Information Sciences 304 (2015) 28–53

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

A computational approach to the continuous authentication biometric system

Soumik Mondal, Patrick Bours *

NISlab, Gjøvik University College, Norway

ARTICLE INFO

Article history:
Received 17 December 2013
Received in revised form 18 December 2014
Accepted 28 December 2014
Available online 31 January 2015

Keywords:
Continuous Authentication
Mouse Dynamics
Multi-modal Biometrics
Trust Model
Weighted Fusion Scheme
Score Boost

ABSTRACT

In this paper, we investigate the performance of a continuous biometric authentication system under various different analysis techniques. We test these on a publicly available continuous mouse dynamics database, but the techniques can be applied to other biometric modalities in a continuous setting also. We test all different combinations of fusion techniques, threshold settings, score boosting techniques and static versus dynamic trust models. We extensively describe the way that performance is reported when analyzing the performance of a continuous authentication system. Contrary to a biometric system for access control at the start of a session can the performance not simply be reported by a single EER value or a DET curve. We show that the optimal performance we can reach with our new techniques improves significantly over the best known performance on the same dataset.

1. Introduction

People use access control mechanisms, like username/password, token, or biometrics, to protect against unauthorized access of another person. This means that a user needs to give proof of his/her identity when starting a computer or when unlocking a PC. However, in many cases people leave the computer unattended for shorter or longer periods when it is unlocked, *e.g.* to get a cup of coffee, to go and talk to a colleague, or simply because they do not have the habit of locking a computer because of the inconvenience.

Access control to a computer is generally implemented as a one-time proof of identity during the initial log on procedure. The validity of the user is assumed to be the same during the full session. Unfortunately, when a computer is left unattended, any person can have access to the same sources as the genuine user. This type of access control is referred to as static authentication. On the other hand, we have *Continuous Authentication* (also called *Active Authentication*), where the genuineness of a user is continuously verified based on the activity of the current user operating on the machine. When doubt arises about the genuineness of the user, the system can lock, and the user has to revert to the static authentication access control mechanism to continue working. A continuous authentication system should, with very high probability, never lock out the genuine user. On the other hand, it should detect an impostor user within a short period of time, to limit the potential damage that can be done by this impostor user to information available on the computer and to limit the disclosure of restricted information.

Research on continuous authentication started in 1995 when Shepherd [29] and Monrose et al. [20] showed some impressive result on continuous authentication using keystroke dynamics. These days continuous authentication is getting more

* Corresponding author.
E-mail addresses: soumik.mondal@hig.no (S. Mondal), patrick.bours@hig.no (P. Bours).

http://dx.doi.org/10.1016/j.ins.2014.12.045
0020-0255/

popular because of the security requirements in office environments and the DARPA's Active Authentication project announced in 2012.[1] In our research, we use *Mouse Dynamics* as a biometric modality, but our contributions could be useful for continuous authentication when using any biometric modality.

The remainder of this paper is organized as follows. We provide a state of the art on continuous authentication systems using mouse dynamics in Section 2. In Section 3, we will discuss some background knowledge for better understanding the techniques used in this paper. We will discuss the data and the features used for analysis in Section 4. In Section 5, we will discuss our contributed algorithms for continuous authentication systems. We will provide the system architecture for this research in Section 6. In Section 7, we are going to discuss the experimental results we have found from this research. Section 8, provides discussion related to some general concern and recommendations related to the usefulness of our research for the future aspects. Finally, we conclude this paper with future work in Section 9.

2. State of the art

Research on mouse dynamics based biometric authentication started in 2003 when Everitt et al. [10] showed a promising result of static authentication but most recently Shen et al. [28] showed the potential of mouse dynamics for static authentication as an alternate access control solution.

Gamboa et al. [12,13] considered mouse strokes for game based authentication technique. 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 *Equal Error Rate (EER)* is the function of iteration time and the number of strokes per user. They reported results on 1 stroke (EER of 48.9%), 50 strokes (EER of 2%) and 100 strokes (EER of 0.7%).

Pusara et al. [24] 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 used Decision Tree Classifier with smoothing filters for classification. They reported a *False Non-Match Rate (FNMR)* of 0.43% and a *False Match Rate (FMR)* of 1.75% with the verification time ranging from 1 min to 15 min.

Schulz [26] investigated a continuous authentication system using mouse dynamics. He collected in his experiment data of 72 users. He 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 he used Euclidean distance for classification. He 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.

Nakkabi et al. [22] build a continuous authentication mechanism with mouse dynamics. They collected data of 48 users and achieved a FMR of 0% and a FNMR of 0.36%. Their dataset is publicly available and was also used for 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.

Zheng et al. [31] used angle-based metrics of mouse movements for user verification. They used data of 30 users (different ages, educational backgrounds, and occupations) and SVM as a classification tool. They reported an EER of 1.3% with the requirement of 20 mouse clicks.

Jorgensen et al. [15] reported some existing flows on state of the art continuous authentication systems using mouse dynamics like, time required for authentication, influence of environmental variables and remote access scenario. We believe that the environmental variables should not be controlled because, this is not a natural representation of users behavior. They collected data from 17 participants who performed some predefined tasks for the duration of 30 min and showed some improvement over the existing systems. However, this data set is too small to compare the results with other research.

Feher et al. [11] used individual mouse actions (contrary to using a histogram over a number of mouse actions) as a feature for continuous authentication. They used 25 volunteers (21 male and 4 female) to collect data in their experiment and used Random Forest Classifier for data analysis. They achieved an EER of 8.53% (for 30 actions) with the identification time of less than 2 min.

Lin et al. [18] build a continuous authentication system by using everyday mouse interaction data on a windows computer. They 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, because users are more consistent on their dynamics for the file related operations which occurs always in a similar manner.

Shen et al. [27] 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 behavior patterns and used different classifier for classification. The best result was an FMR of 0.37% and an FNMR of 1.12%, obtained using a One Class SVM detector. The authors also showed that the obtained result is directly proportional to the number of operations performed by the user. Also, Chi et al. [7] studied the dimensionality reduction based approach for the same dataset and obtained the best result as FMR of 1.02% and an FNMR of 2.67%.

[1] http://www.darpa.mil/Our_Work/I2O/Programs/Active_Authentication.aspx.

To the best of our knowledge is there no publicly available dataset which contains the biometric data for continuous mouse dynamics with a significant number of users in order to provide the statistical significance of the analysis, except the dataset we have used in this research as well as the research conducted by Nakkabi et al. [22]. This dataset was build based on free, continuous computer mouse usage. Therefore, we have done our research based on mouse dynamics and tested our algorithms on this dataset. The proposed algorithms can however be useful for continuous authentication when using any biometric modality.

We found that the current research on continuous authentication reports the results in terms of *Equal Error Rate (EER)* or *False Match Rate (FMR)* and *False Non-Match Rate (FNMR)* over either the whole test set or over chunks of a large, fixed number of actions. This means that an impostor can perform a number of actions before the system checks his identity for the first time. 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 action performed by the user.
- The new evaluation metric has been introduced for Continuous Authentication Biometric Systems.
- Score Boost algorithm to improve the system performance.
- Dynamic Trust Model for continuous authentication.
- Efficient weighted fusion scheme.

3. Background knowledge

For our analysis, we test two different classification algorithms in a multi-modal architecture. These are *Support Vector Machine (SVM)* and *Artificial Neural Network (ANN)*. Some details of SVM and ANN and the basic understanding of multi-classifier fusion are presented below.

3.1. Support Vector Machine

Support Vector Machine (SVM) is a very well-known supervised learning algorithm which can be used for classification problem [6]. This classifier is capable of creating a linear 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 use the LibSVM software distribution for the SVM classifier [8]. The main motivation to use the LibSVM is not only because it is a well implemented optimization technique for the cost function of SVM and widely used in the research community, but also because it provides the classification score (probability) along with the class label. In our research, we use only the classification score for our analysis. 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.

3.2. Artificial Neural Network

Artificial Neural Network (ANN) is a combination of multiple artificial neurons which can be used for classification and regression analysis [4]. In our research, the neurons consist of a *linear* activation function with a 2-layer *Feed-Forward* neural network. In this research, we use the NETLAB software distribution for the ANN classifier [21]. NETLAB has a good implementation of *Scaled Conjugate Gradient* algorithm which is efficient to optimize the cost function and also it will reduce the ANN training time. We select different numbers of hidden nodes, and different regularization parameter values (α) for different users to maintain the trade off between *Bias* and *Variance* and the training time of the classifier model.

3.3. Multi Classifier Fusion

Multi Classifier Fusion (MCF) is a technique to combine multiple classifiers on the same biometric modality to improve the performance of that modality [14,16]. Researchers generally prefer to use multiple classifiers on a same modality when the modality is considered to be a weak modality. The architecture of the MCF technique is very much similar to the multi-modal biometric technique but, with the MCF technique, only score level and decision level fusion are possible [25]. For our analysis, we use SVM and ANN classification algorithms in a multi-modal architecture with score level fusion to improve the performance of the mouse dynamics modality for continuous authentication [19].

3.4. Trust Model

The concept of the *Trust Model* was first introduced by Bours [5]. In this model, the behavior of the current user is compared to the template of the genuine user. Based on each single action performed by the user, the trust in the genuineness of

that user will be adjusted. If the trust of the system in the genuineness of the user is too low, then the user will be locked out of the system. In particular if the trust drops below a pre-defined threshold $T_{lockout}$ then the system locks itself and will require static authentication of the user to continue working.

The basic idea is that the trust of the system in the genuineness of the current user depends on the deviations from the way this user performs various actions on the system. If a specific action is performed in accordance with how the genuine user would perform the task (*i.e.* as it is stored in the template), then the systems trust in the genuineness of this user will increase, which is called *Reward*. If there is a large deviation between the behavior of the genuine user and the current user, then the trust of the system in that user will decrease, which is called *Penalty*. The amount of change of the trust level can be fixed or variable [5]. A small deviation from the behavior of the user, when compared to the template, could lead to a small decrease in trust, while a large deviation could lead to a larger decrease.

No single person is able to always behave in exactly the same manner [3]. For the genuine user this means that he will also sometimes deviate more from his normal behavior, which leads to a decrease in trust. However, the majority of actions that a genuine user performs will be close to his normal behavior, *i.e.* leads to an increase of trust. Overall this would lead to a high level of trust. For an impostor however the opposite holds. In some cases he behaves as the genuine user, increasing his level of trust, but the majority of actions will lead to a decrease in trust due to the larger deviation from the behavior of the genuine user. This then leads to a general decrease of the trust over time for an impostor user. Obviously an ideal system should perform in such a way that the trust in anyone other then the genuine user decreases fast to a value below the threshold $T_{lockout}$. In such an ideal system, also a genuine user never reaches a trust level that results in a lockout, *i.e.* the genuine user does not notice the influence of the continuous authentication system in his daily activities.

4. Data description and feature extraction

In this section, we describe the dataset that we use in this research. We provide the raw data description in Section 4.1. In Section 4.2, we describe the features that are extracted from the raw data for the analysis we perform. In Section 4.3, we discuss the data separation process for training and testing the system.

4.1. Raw data description

We use a publicly available continuous authentication dataset for this research. This dataset contains the mouse dynamics data collected from 49 volunteers but, for the analysis Nakkabi et al. [22] have used only the data of 48 users. 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. There is a huge variation in the number of samples per user (minimum 3736, maximum 333,789, average 60,701). The data collection software stored the following 4 raw features for each mouse action from a volunteer:

1. Type of action (1: Mouse Move; 2: Silence; 3: Point and Click; or 4: Drag and Drop).
2. Travelled distance in pixels.
3. Elapsed time in second (with a 0.25 second *Sampling Interval*).

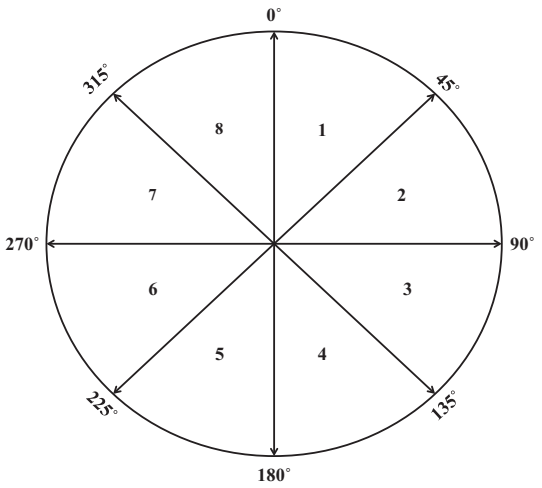

Fig. 1. Direction of the mouse movements.

4. Direction of movement (a value between 1 and 8 according to the movement of the mouse. See Fig. 1 for which direction corresponded to which value).

We encountered the following limitations for this dataset:

- This dataset shows only mouse move distance not the trajectory of the mouse move. The trajectory of the mouse move could be useful to derive some more features which will hopefully improve the system performance.
- There is no separation between left mouse button click and right button click.
- There is no time information about mouse click.
- The time granularity of the capture software is 0.25 second which is very coarse and we miss some important behavioral information form the dataset.
- Due to limited information about the number of sessions and separation of the sessions for each user we are unable to separate the data based on the sessions (Here, session means different day of work or a long pause during the work.).

4.2. Feature extraction

In many works on continuous authentication (e.g. [1,22]) 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. Therefore, we cannot use statistical features derived from the raw data, but are we look for single event based features. We extract the following 5 features from the raw data:

Type of action: We explicitly remove the *Silence Events* from the raw data because we want to focus on the behavior of the user. We therefore, only use the other actions that were recorded, *i.e.* Mouse Move (MM), Point and Click (PC), and Drag and Drop (DD).

Direction: Taken directly from the raw data.

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

Reciprocal Acceleration of the mouse action: Equal to the Elapsed time/Speed of the action.

Travelled distance in bins: We do not use the travelled distance in its raw form, but 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 is between 1 and 50 pixels then we assign bin 1, if the travelled distance is within 51–100 pixels we assign bin 2, and so on. After that the bins grow in range, in particular we use 38 bins according to the following schedule:

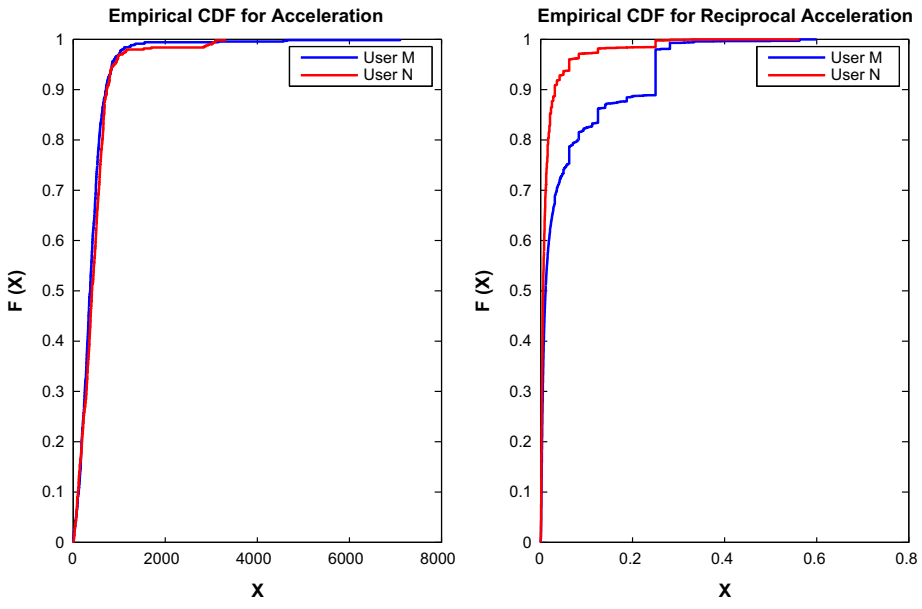

Fig. 2. Empirical Cumulative Distribution Function plot for Acceleration and Reciprocal of the Acceleration for two users.

	X_{tr}	X_{te}
Y_{tr}	2	
	3	
	...	
	48	
	49	

Fig. 3. Data separation for *VP-1*.

- 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.

We first tried to use the *Acceleration* (Speed of the action/the Elapsed time) of the mouse action, but we got much better results when using the reciprocal of the acceleration. Fig. 2, shows the *Empirical Cumulative Distribution Function* plot for *Acceleration* and *Reciprocal of the Acceleration* feature for two users. From this pattern we can distinctly understand that *Reciprocal of the Acceleration* is highly separable in comparison with the *Acceleration* feature. The mentioned formulas for the calculation of speed and acceleration are not the exact formulas to calculate the speed and acceleration of mouse actions [11]. As we discussed in the Section 4.1, there are some limitations of this dataset. Due to inaccessibility of the small distance change with time within a mouse move trajectory, we are unable to estimate the precise speed and acceleration of the mouse actions. Based on our calculation of speed and acceleration, we have found that the acceleration has a very high correlation with the speed of the action. Therefore, we got worse results when we are using the acceleration.

4.3. Data separation

In our research, we use three verification processes which we describe below. We split the data of each of the users into a part for training and a part for testing. In all cases the classifiers are trained with genuine and impostor (training) data. The amount of training data of the genuine user is 50% of the total amount of data of that user, with a maximum of 20,000 actions.[2] The training data from the impostor users is taken such that the total amount of data of all impostors together is equal to the amount of training data of the genuine user. This was done to avoid bias towards either the genuine or the impostor class. The three verification processes described below may be seen to correspond with an "internal system" (*VP-1*), an "external system" (*VP-3*) or a combination of both (*VP-2*).

As we mentioned before (see Section 4.1), due to some limitations of this dataset we are unable to build the validation set for this research.[3] Therefore, we choose the parameters for the different algorithms (see Section 5) used in this research based on the training set.

4.3.1. Verification Process 1 (VP-1)

In this case the impostor part of the training data is taken from all 48 impostors and each of the 48 impostors contributes approximately with the same amount of data for the training of the classifier. This can for example be done internally in an organization where all users provide data from training the various classifiers. In this verification process all the imposter users are known to the system.

For the testing we use all the data of the genuine user and the impostor users that has not been used for the training of the classifier. This means that we have 1 genuine set of test data and 48 impostor sets of test data for each user. Fig. 3 explain this separation process for the first user, where $|X_{tr}| \approx |Y_{tr}|$.

4.3.2. Verification Process 2 (VP-2)

This verification process can be seen as a being a combination of an internal and external process. The classifiers is trained with data from the genuine user as well as data of 24 of the impostor users. In this verification process 50% of the imposter users are known to the system.

Also here we do test the system with all genuine and impostor data that has not been used for training. For each user this means, besides his own genuine test set, the full dataset of 24 of the impostors and for the other 24 impostors the full data set with exclusion of the training data is used for testing. Fig. 4 explains this separation process for the first user, where $|X_{tr}| \approx |Y_{tr}|$.

[2] We set this maximum limit of 20,000 actions primarily because of two reasons. First is to keep a significant amount of data for the testing of all imposters and second is to reduce the classifier's training time. We found that due to this maximum limit the classifier's training accuracy also improved.

[3] Here, validation set means a separate dataset based on which we can optimize our algorithmic parameters. In some state of the art research this set is also named as development dataset.

4.3.3. Verification Process 3 (VP-3)

This verification process can be seen as an external system, where the training and the testing of the system is done by separate sets of impostor users. This can be a situation where a new user will provide his own training data and the remaining training data will be provided by an external organization. This means that data of impostors of the system will never have been used for training the classifiers. In this case we have two training datasets per genuine user. We did split the group of impostor users in 2 sets of 24 impostors. Fig. 5 explain this separation process for the first user, where $|X_{tr}| \approx |Y_{tr}|$. First, we train the classifiers with the first set of training data of the genuine user and the training data of the first set of 24 impostor users, exactly as we have done in *VP-2* (see Fig. 5(a)). Next we tested this system with the testing data of the genuine user and all of the data of the second set of 24 impostor users (see Fig. 5(b)). This process is then repeated with the second set of training data where the impostor users swapping roles, *i.e.* the data of the second set of 24 impostors will be used to train the classifiers, while it is tested with the data of the first set of impostors (see Fig. 5(a)). In this verification process no imposter users are known to the system.

5. Contributed algorithms

In this section we describe our contributed algorithms that are used in the analysis. These algorithms are as following:

5.1. Score Boost

In this section, we discuss our score boosting algorithm which we have applied. As we are dealing with the classification score of the current action to decide the genuineness of the current user, we found that there is a huge overlap between

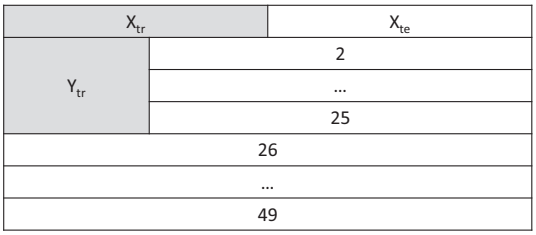

Fig. 4. Data separation for *VP-2*.

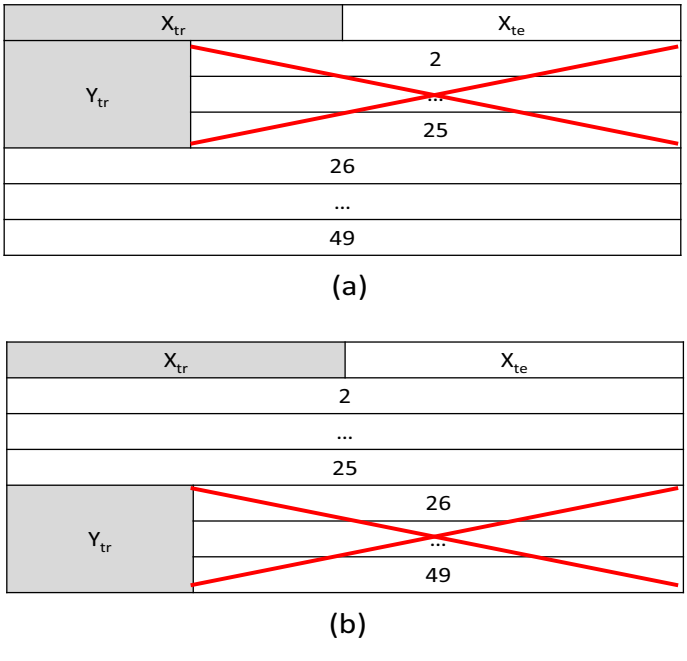

Fig. 5. Data separation for *VP-3*.

genuine and impostor scores on the training set. Therefore, we try to find some method which can significantly improve the performance of the system in this situation. The Score Boost technique will significantly boost the score in a particular range. Therefore, the genuine action will get a higher reward and the impostor action will get a higher penalty. In Algorithm 1, we show the Score Boost technique. This algorithm is capable of boosting the classification score of the current action based on some parameters. The score will boost towards a higher order if the score is above parameter C and if the score is below this parameter it will boost towards a lower order. The parameter W is the width where the score will remain unchanged from $C - \frac{W}{2}$ to $C + \frac{W}{2}$. The parameter P is used for the order of the Score Boost.

In Fig. 6, we have shown the nature of this algorithm for different parameters.

Algorithm 1. Algorithm for Score Boost.

Data:

$x_i \rightarrow$ Classification score i_{th} action

$C \rightarrow$ Center

$W \rightarrow$ Width of left unchanged

$P \rightarrow$ Power of boost

Result:

$X_i \rightarrow$ Boosted classification score i_{th} action

begin
  if $x_i < (C - \frac{W}{2})$ **then**
    if $x_i \geq 0$ **then**
      $X_i = \frac{(x_i)^P}{C - \frac{W}{2}}$
    else
      $X_i = -|x_i|^{\frac{1}{P}} - (C - \frac{W}{2})$
    end
  else
    if $(C - \frac{W}{2}) \leq x_i \leq (C + \frac{W}{2})$ **then**
      $X_i = x_i$
    else
      $X_i = C + \frac{W}{2} + ((1 - c - \frac{W}{2}) \times (x_i - c - \frac{W}{2}))^{\frac{1}{P}}$
    end
  end
end

5.2. Weighted fusion scheme

In Algorithm 2, we explain the weighted fusion scheme. This algorithm takes several parameters and computes the weights for the fusion of the SVM (WT_{SVM}) and ANN (WT_{ANN}) classifiers. Based on the accuracy difference of the two classifiers this algorithm will decide the weights for score fusion. We use *Tolerance* to neglect the amount of accuracy difference and put the same weight for both the classifiers. The slope of the straight line of this function can be adjusted by *Slope* parameter. According to the primary principals of weighted fusion scheme [30], the algorithm should satisfy the two criteria, $0 \leqslant WT_i \leqslant 1,\ \forall i$ and $\sum_{i=1}^{n} WT_i = 1$, where $n =$ number of classifiers used in the system. The maximum weight on a classifier can be adjusted by *UpperLimit* parameter. If we set the *UpperLimit* parameter to 1, this algorithm can be used as a classifier selection.

In Fig. 7, we show the Accuracy difference vs. Weight calculation from the Eq. (1) for different parameters.

Algorithm 2. Algorithm for Weighted Fusion Scheme.

Data:

$Accuracy_{svm} \rightarrow$ Accuracy of SVM classifier

$Accuracy_{ann} \rightarrow$ Accuracy of ANN classifier

$Slope \rightarrow$ Slope of the straight line on the function

$Tolerance \rightarrow$ Parameter for tolerance of the classifiers accuracy difference

$UpperLimit \rightarrow$ Upper limit of the classifier weight ($0.5 < UpperLimit \leq 1$)

Result:

$WT_{SVM} \rightarrow$ Weight for SVM classifier

$WT_{ANN} \rightarrow$ Weight for ANN classifier

begin

  $Abs_{diff} = |Accuracy_{svm} - Accuracy_{ann}|$

  $Diff_{acc} = Abs_{diff} - Tolerance$

  $$WT = \min\{0.5 + \max\{Diff_{acc} \times Slope \times 0.1, 0\}, UpperLimit\} \quad (1)$$

  if $Accuracy_{svm} > Accuracy_{ann}$ **then**

    $WT_{SVM} = WT$

    $WT_{ANN} = 1 - WT$

  else

    $WT_{SVM} = 1 - WT$

    $WT_{ANN} = WT$

  end

end

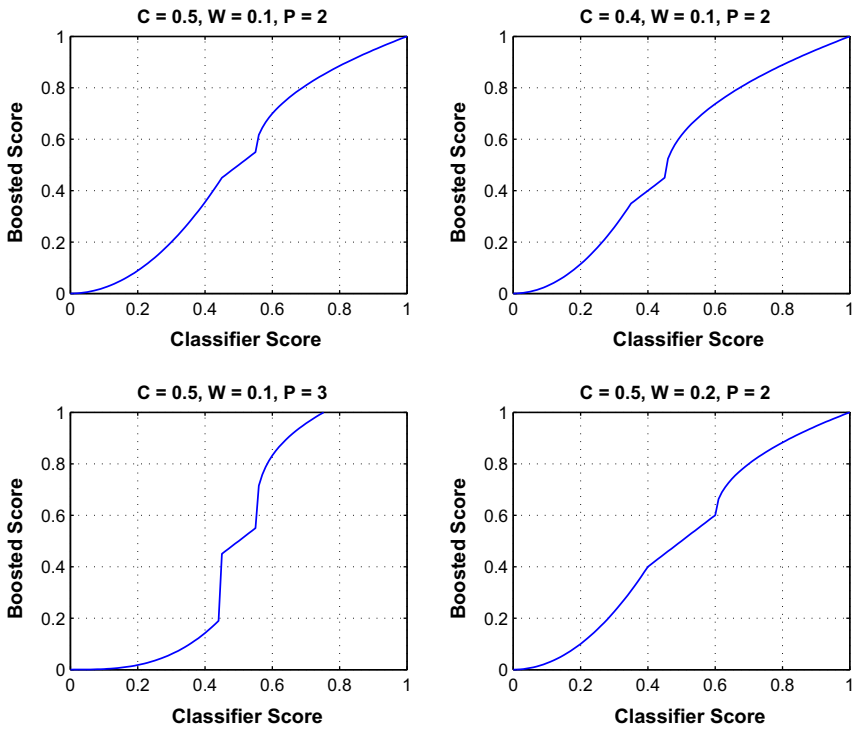

Fig. 6. Classifier score vs. Boosted score from Algorithm 1 with different parameters.

5.3. Static Trust Model

Bours [5] 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 use the classifier score (that is the probability of the genuineness of that event) to increase or decrease the trust value (denoted by *Trust*). Based on the different levels of penalty and reward and also different functions for penalty and reward value we create different trust models. These are explained below;

Both the model by Bours [5] and by BehavioSec [9] are two level trust models. This means that these models have one threshold value which decides if the current action leads to a penalty or to a reward. The main difference between these two models is the calculation for the penalty and reward value.

5.3.1. 3-level Static Trust Model

In [19], a three level trust model is used. In Algorithm 3 we explain this 3-level Static Trust Model algorithm. This model has 1 level for a reward and 2 levels for penalties. In our research, we use this trust model with different threshold parameters and also with different penalty and reward functions.

Algorithm 3. Algorithm for 3-level Static Trust Model.

Data:

$x_i \rightarrow$ Classification score for the i^{th} action

$Tr \rightarrow$ Threshold for Penalty or Reward

$Tr_P \rightarrow$ Threshold for 2^{nd} level Penalty

$Trust_{i-1} \rightarrow$ System trust after $(i-1)^{th}$ action

Result:

$Trust_i \rightarrow$ System trust on the user after i^{th} action

```
1  begin
2      if x_i ≥ Tr then
3          ΔTrust = f_Reward(x_i)
4      else
5          if Tr_P ≤ x_i < Tr then
6              ΔTrust = −f^1_Penalty(x_i)
7          else
8              ΔTrust = −f^2_Penalty(x_i)
9          end
10     end
11     Trust_i = min {max {Trust_{i−1} + ΔTrust , 0} , 100}
12 end
```

5.3.2. 4-level Static Trust Model

We also use a four level trust model. In Algorithm 4 we explain this 4-level Static Trust Model algorithm. This model has 2 levels of rewards and 2 levels of penalties. We use this trust model with different threshold parameters and also with different penalty and reward functions.

Algorithm 4. Algorithm for 4-level Static Trust Model.

Data:

$x_i \rightarrow$ Classification score for the i^{th} action

$Tr \rightarrow$ Threshold for Penalty or Reward

$Tr_R \rightarrow$ Threshold for 2^{nd} level Reward

$Tr_P \rightarrow$ Threshold for 2^{nd} level Penalty

$Trust_{i-1} \rightarrow$ System trust after $(i-1)^{th}$ action

Result:

$Trust_i \rightarrow$ System trust on the user after i^{th} action

```
begin
    if x_i ≥ Tr then
        if x_i ≥ Tr_R then
            ΔTrust = f^1_Reward(x_i)
        else
            ΔTrust = f^2_Reward(x_i)
        end
    else
        if x_i ≥ Tr_P then
            ΔTrust = −f^1_Penalty(x_i)
        else
            ΔTrust = −f^2_Penalty(x_i)
        end
    end
    Trust_i = min {max {Trust_{i−1} + ΔTrust , 0} , 100}
end
```

5.3.3. Discussion

We have some limitations and drawbacks for Static Trust Models. These are given below,

- Very difficult to find optimal threshold parameters for different levels.
- There can be a large difference in change in trust in case the classifier score is just below or just above the threshold.
- Difficult to find what type of Static Trust Model should be used on a system.
- There is a discontinuous relation for penalty and reward value according to the classifier's score.

5.4. Dynamic Trust Model

To overcome some of the limitations of the Static Trust Model, we come up with a new Dynamic Trust Model. Algorithm 5 explains the idea behind the Dynamic Trust Model. This algorithm takes several parameters and returns the system trust of the genuineness of the user after the current action performed by the user. All the parameters for this algorithm can be different for different users. Also, we can change the parameters for different kind of actions performed by the users. For an example, we can use three different sets of parameters for the same user for the three different types of actions (see Section 4.2).

Eq. (2) in Algorithm 5 is capable of calculating the small change required on the system trust ($\Delta Trust$) based on the classification score of the current action performed by the user. In Fig. 8, we show the $\Delta Trust$ produced by the Eq. (2) based on the classification score of the current action for different parameters.

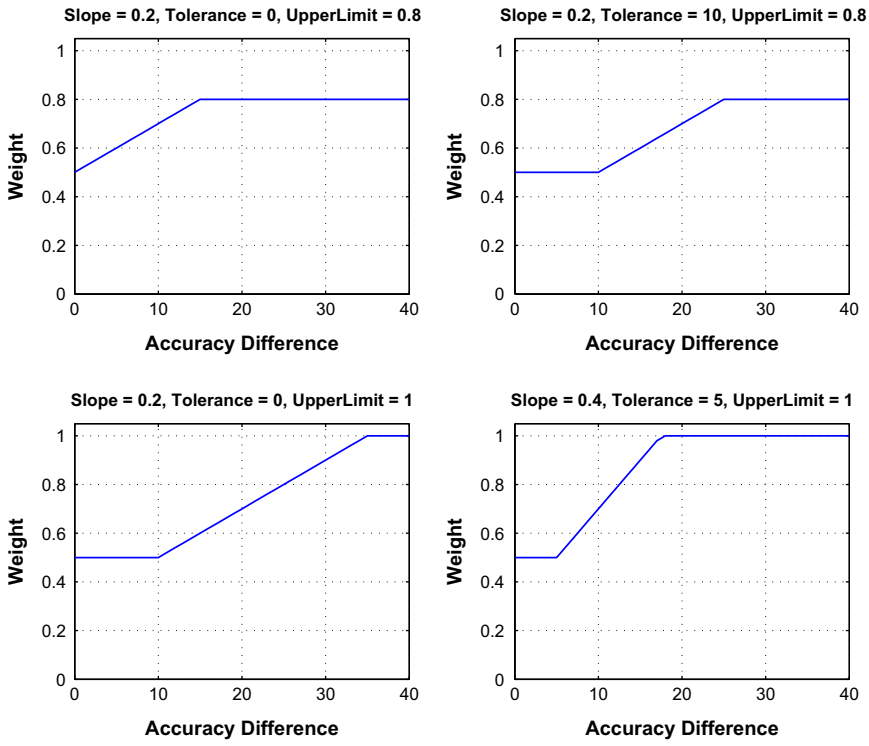

Fig. 7. Accuracy difference vs. weight from Eq. (1) with different parameters.

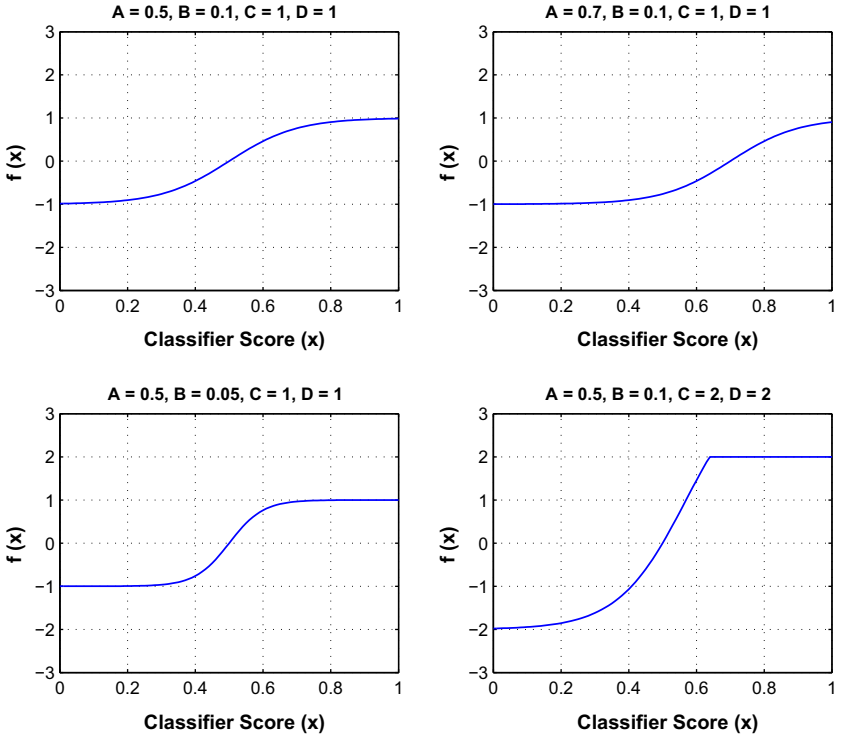

Fig. 8. Classifier score vs. $f(x)$ from Eq. (2) with different parameters.

In this algorithm, the parameter A is the threshold value for penalty or reward for the trust model. If the classification score (x_i) of the current action is equal to this threshold then $\Delta Trust = 0$. If $x_i > A$ then $\Delta Trust > 0$, *i.e.* a reward, and if $x_i < A$ then $\Delta Trust < 0$, *i.e.* a penalty. The parameter B is the width of the sigmoid for this function (see Fig. 8). The parameters C and D are the upper limit of the reward and penalty.

Algorithm 5. Algorithm for Dynamic Trust Model.

Data:

$x_i \rightarrow$ Classification score for the i^{th} action

$A \rightarrow$ Threshold for penalty or reward

$B \rightarrow$ Width of the sigmoid

$C \rightarrow$ Maximum reward

$D \rightarrow$ Maximum penalty

$Trust_{i-1} \rightarrow$ System trust after $(i-1)^{th}$ action

Result:

$Trust_i \rightarrow$ System trust on the user after i^{th} action

1 **begin**

2

$$\Delta Trust(x_i) = \min\{-D + D \times (\frac{1 + \frac{1}{C}}{\frac{1}{C} + \exp(-\frac{x_i - A}{B})}), C\} \quad (2)$$

3 $Trust_i = \min\{\max\{Trust_{i-1} + \Delta Trust(x_i), 0\}, 100\}$

4 **end**

6. System architecture

In this section, we discuss the methodology of our analysis. The system is divided into two basic phases (see Fig. 9).

I. *Training Phase:* In the training phase, the training data (see Section 4.3) is used to build the classifier models and store the models in a database for use in the testing phase. Each genuine user has his/her own two classifier models (SVM

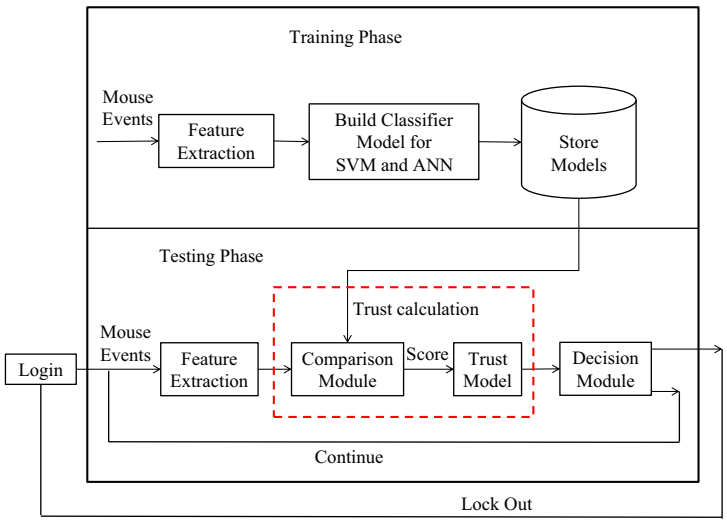

Fig. 9. Block diagram of the system.

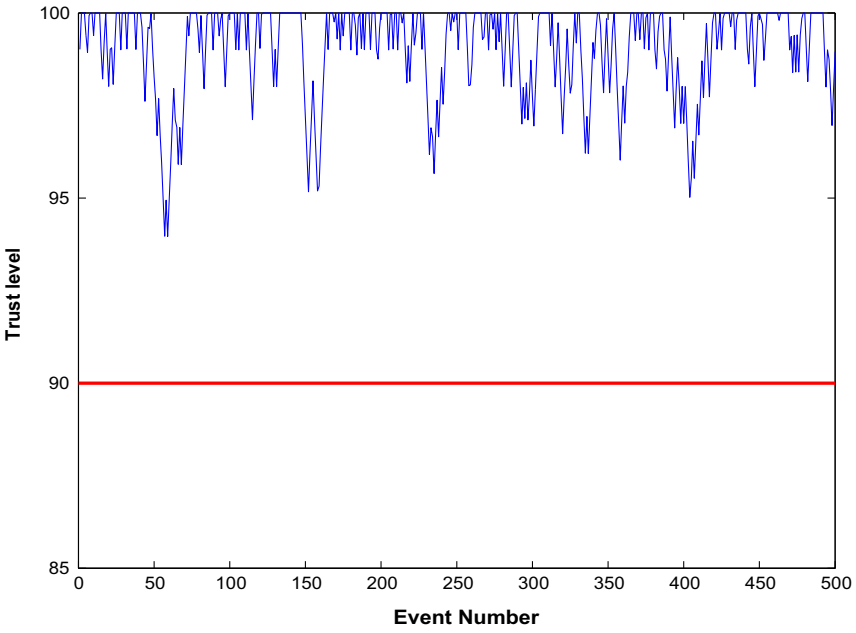

Fig. 10. Trust value for genuine user tested with the genuine test set.

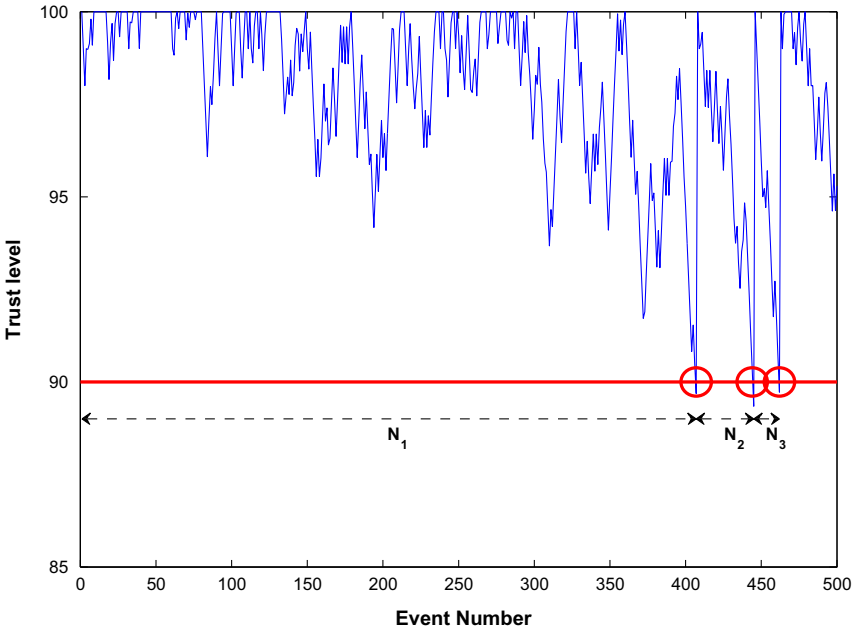

Fig. 11. Trust value for genuine user tested with the impostor test set.

and ANN). This means that we have build two sets of 49×2 different classifier models for *VP-1* and *VP-2* and one set of 49×4 different classifier models for *VP-3*.

II. *Testing Phase:* In the testing phase, we use the test data, which was separated from the training data, for comparison. In the comparison, we use the models stored in the database and obtain the classifier score (probability) on each sample of the test data. This score is then used to update the trust value *Trust* in the trust model (see Sections 5.3 and 5.4). Finally, the trust value *Trust* is used in the decision module, to determine if the user will be locked out or can continue using the PC. This decision is made based on the current trust value and the lockout threshold ($T_{lockout}$).

In the testing phase we measure the performance of the system in terms of *Average Number of Genuine Actions (ANGA)* and *Average Number of Impostor Actions (ANIA)* [19]. 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. We count the number of test data samples of the genuine or an impostor user that can be performed in the trust model before the user is locked out. A user will always start at the trust value $Trust = 100$ and in case of a trusted test data sample, *i.e.* the probability outputted by the classifier model is above threshold, this trust value will go up (with a maximum of 100). In case the probability from the classifier model is below threshold then the trust value *Trust* will decrease. This means that if the user's working is in accordance with the classifier model, then the trust in the genuineness increases, otherwise it will decrease.

In Figs. 10 and 11, we elaborate on this concept. In Fig. 10, we see how the trust level changes when we compare a model with test data of the genuine user. We see that the trust level will never drop below the lockout threshold ($T_{lockout}$ is equal to 90). Fig. 11, shows that if the same model is compared to test data of an impostor user, the trust will drop (in this example) 3 times below the lockout threshold within 500 user actions. The value of ANIA in this example equals $\frac{1}{3}\sum_{i=1}^{3} N_i$. We can calculate ANGA in the same way, if the genuine user is locked out based on his own test data.

The data samples from the genuine user will in most cases get a high classification score from the classifier and sometimes a low score. 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 behavior 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.

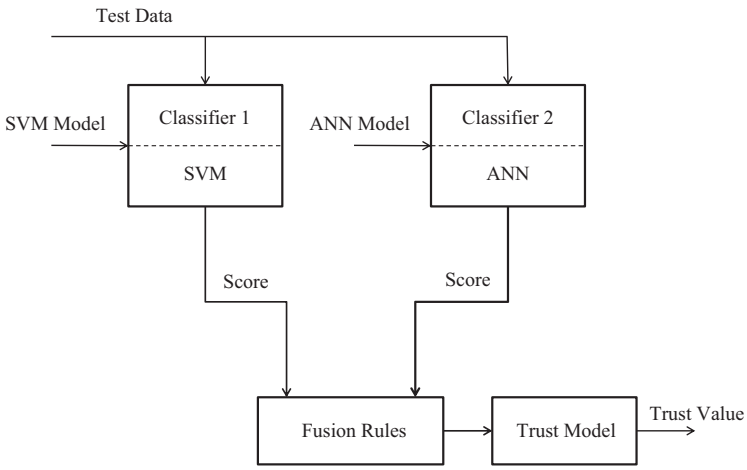

Fig. 12. Block diagram of the *Trust Calculation* module (see Fig. 9) without sore normalization or score boosting.

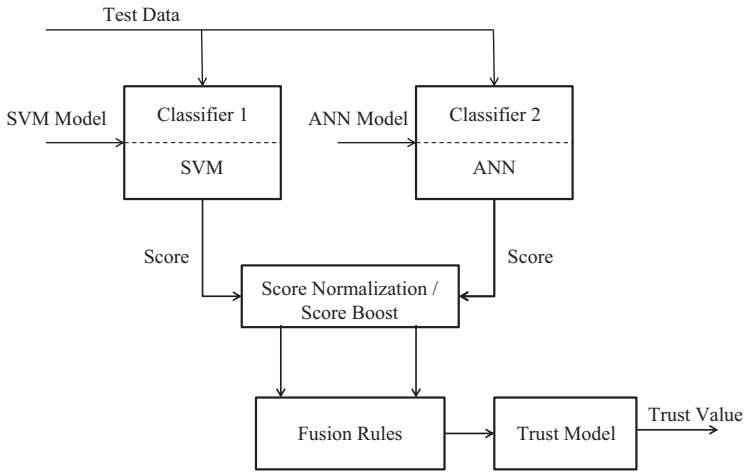

Fig. 13. Block diagram of the *Trust Calculation* module (see Fig. 9) with sore normalization or score boosting.

Table 1
Example of extended performance reporting from state of the art research.

Ref.	#Users	FNMR (%)	FMR (%)	Blocksize	ANGA	ANIA
[22]	48	0.36	0	2000	555,555	2000
[31]	30	1.3	1.3	20	1538	20
[27]	28	1.69	1.22	2000	118,343	2025
[11]	25	7.5	7.5	100	1333	108
[7]	28	3.38	4.51	2000	59,172	2094

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.

In Fig. 12, we show the extended block diagram for the Trust Calculation module (see Fig. 9) without any classifier score normalization and score boosting as discussed in Section 5.1. In Fig. 13, we show the extended block diagram for the Trust Calculation module (see Fig. 9) with classifier score normalization or score boosting technique. The score normalization or Score Boost is used for the pre-processing of the raw scores as they are produced by the classifiers. We made this step optional and done separate performance analysis with and without the pre-processed score, which we discuss in Section 7. We also measure the performance of the system by applying min–max score normalization techniques [30] and our proposed score boosting technique (see Section 5.1). After this step, both classifier scores will go to the Fusion Rules module. In this module, we apply different state of the art fusion rules like, average, min, max [25] and our proposed weighted fusion scheme (see Section 5.2). After that the fused score will go to the Trust Model and calculate the current system trust on the genuineness of the user. Based on the current system trust the system decides if the user can continue his/her work or if the user should be locked out from the system.

6.1. FNMR/FMR to ANGA/ANIA conversion

In this section, we show how we can express FNMR and FMR in terms of ANGA and ANIA. Assume FNMR equals p and the system operates on chunks of m actions. The genuine user can always do m actions and with the probability of $(1-p)$ he can continue and do m more actions. After that, again with the probability of $(1-p)$, he can do m more actions, etc. So in total we find:

$$\text{ANGA} = m + (1-p)\times m + (1-p)^2 \times m + (1-p)^3 \times m + \ldots$$
$$\text{So, ANGA} = \frac{m}{1-(1-p)} = \frac{m}{p}$$

Similarly, if FMR is p, then the imposter user can always do m actions and with the probability of p he can continue and do m more actions. After that again with probability p he can do m more actions, etc. So in total we find:

$$\text{ANIA} = m + p \times m + p^2 \times m + p^3 \times m + \cdots$$
$$\text{So, ANIA} = \frac{m}{1-p}$$

For example, Feher et al. [11] got an EER of 8.53% with chunks of 30 actions. So, $p = 0.0853$ and $m = 30$; then $\text{ANGA} = \frac{30}{0.0853} \approx 352$ and $\text{ANIA} = \frac{30}{1-0.0853} \approx 33$. It is clear that an impostor can do at least m actions because the system checks the identity of the user for the first time after m actions. Table 1, shows the extended performance reporting from state of the art research.

7. Result analysis

In this section, we analyse our results that we got by applying the algorithms discussed in Section 5. We divide our analysis into two major parts based on the type of trust model (see Sections 5.3 and 5.4). As mentioned before, in this research the total number of data sets of genuine users is 49 and the total number of data sets of impostor users is 2352 (49×48). We report the results from a one-hold-out cross validation test in terms of ANIA and ANGA along with the total number of impostors not detected for different lockout threshold ($T_{lockout}$). Also, we report the results with the user specific lockout threshold (T_{us}), where the threshold for lockout will satisfy $50 \leqslant T_{us} < \min(Trust_{genuine})$.

Besides reporting the performance in terms of ANIA and ANGA and in terms of the 4 categories described below, will we also report the results in terms of FMR and FNMR. We do realize that this is a slight abuse of terminology in the context of continuous authentication with our analysis methods, but decided to do so to clarify the results in known terminology. These results can be found in Section 7.3.

Table 2
Results for *VP-1* with the analysis method of Static Trust Model without Score Boost.

$T_{lockout}$	Categories	Average fusion				Weighted fusion			
		# User	ANGA	ANIA	# Imp. ND	# User	ANGA	ANIA	# Imp. ND
T_{us}	+/+	43		68		45		84	
	+/−	3		440	3	4		1326	32
	−/+								
	−/−	3	2066	2076	42				
90	+/+	35		92		34		88	
	+/−	6		1000	22	5		600	13
	−/+	7	4216	103		9	7585	84	
	−/−	1	323	602	5	1	5782	2171	10

Interpretation of the tables: When presenting the results from our analysis, the results are shown for 4 possible categories. The categories are divided based on genuine user lockout (+ if the genuine user is not locked out and − if he is locked out) and non-detection of impostor users (+ if an impostor user is locked out and − otherwise). Each of the 49 users is thus classified in one of these 4 categories:

- *All Positive (+/+):* This is the best case category. In this category, the genuine user is never locked out, and all the 48 impostors are detected as impostors.
- *Positive vs. Negative (+/−):* In this category, the genuine user is not locked out but some impostors are not detected by the system.
- *Negative vs. Positive (−/+):* In this category, the genuine user is locked out by the system. On the other hand are all impostors detected.
- *All Negative (−/−):* This is the worst case category. In this category, the genuine user is locked out by the system and also, some of the impostors are not detected.

The column # *Users* shows how many users will fall within each of these categories (*i.e.* the values sum up to 49). In the column ANGA a value will indicate the Average Number of Genuine Actions in case indeed genuine users are locked out by the system. If the genuine users are not locked out, then we actually cannot calculate ANGA. The column ANIA will display the Average Number of Impostor Actions, and is based on all impostors that are detected. The actions of the impostors that are not detected are not used in this calculation, but the number of impostors that is not detected is given in the column # *Imp. ND*. This number should be seen in relation to the number of users in that particular category. For example, in the Average Fusion '+/−' category in Table 2, we see that # *Users* equals 3, *i.e.* there are $3 \times 48 = 144$ impostor test sets, and only 3 imposters within these 144 impostors are not found by the system as being an impostor.

Ideally is the value of ANGA high or not given (meaning that the genuine users are never locked out) and the value of ANIA low. Also the number of impostors not detected should be as low as possible.

7.1. Analysis of Static Trust Model

We apply the 3-level and 4-level static trust model with different penalty and reward thresholds (see Section 5.3 for the algorithms). Also, we apply different functions to calculate the $\Delta Trust$. We do the separate analysis with the boosted classifier scores and without boosted classifier scores.

We apply the Static Trust Model with and without the Score Boost followed by the Trust Calculation architecture given in Fig. 12. After applying different threshold parameters for 3-level and 4-level trust models, we get the best results from the 3-level static trust model. We apply different fusion rules and find the best results from the Average Fusion $\left(x_i = \frac{SCORE_{SVM}+SCORE_{ANN}}{2}\right)$ rule and our proposed Weighted Fusion scheme ($x_i = WT_{SVM} \times SCORE_{SVM} + WT_{ANN} \times SCORE_{ANN}$). We also test the different parameter values for our proposed weighted fusion scheme (see Section 5.2 for algorithm) to optimize the system performance.

Algorithmic parameters for the above results are given below:

3-level Static Trust Model: The parameters for Algorithm 3 are:

- Threshold parameters, $Tr = 0.5$ and $Tr_P = 0.3 \sim 0.4$.
- Reward function, $f_{Reward}(x_i) = 1 \times x_i$, where x_i is the fused classification score.
- Penalty functions, $f^1_{Penalty}(x_i) = 1 - x_i$ and $f^2_{Penalty}(x_i) = 1$, where x_i is the fused classification score.

Weighted Fusion: The parameters for Algorithm 2 are, $Slope = 0.4,\ Tolerance = 10,\ UpperLimit = 1$.

Table 3
Results for *VP-2* with the analysis method of Static Trust Model without Score Boost.

$T_{lockout}$	Categories	Average fusion				Weighted fusion			
		# User	ANGA	ANIA	# Imp. ND	# User	ANGA	ANIA	# Imp. ND
T_{us}	+/+	48		109		48		131	
	+/−	1		366	1	1		366	1
	−/+								
	−/−								
90	+/+	35		171		43		210	
	+/−	6		781	11	3		543	3
	−/+	8	32,311	138		3	35,009	98	
	−/−								

Table 4
Results for *VP-3* with the analysis method of Static Trust Model without Score Boost.

$T_{lockout}$	Categories	Average fusion				Weighted fusion			
		# User	ANGA	ANIA	# Imp. ND	# User	ANGA	ANIA	# Imp. ND
T_{us}	+/+	47		130		47		91	
	+/−	2		1026	2	2		592	4
	−/+								
	−/−								
90	+/+	34		172		41		171	
	+/−	8		662	16	6		597	8
	−/+	5	31,450	93		2	54,401	158	
	−/−	2	81,004	960	2				

7.1.1. Without Score Boost

Table 2 shows the optimal result we get from this analysis for *VP-1* (see Section 4.3.1). The table is divided into two parts based on the fusion rules. For the Average fusion rule, only 43 users fall into the best case category for the user specific lockout threshold; whereas for weighted fusion, this number is 45. We can clearly observe from the table that if we go from user specific lockout threshold (T_{us}) to fixed lockout threshold ($T_{lockout} = 90$) the results get worse. In the fixed lockout threshold, the number of best performing users decreases, and the numbers for *Positive vs. Negative* and *Negative vs. Positive* increase. We observed from the analysis that, some of the users have $T_{us} > 90$ and those users contribute to the *Positive vs. Negative* category and some of the users have $T_{us} < 90$ and those users contribute to the *Negative vs. Positive* category.

Table 3, shows the optimal result, we get from this analysis for *VP-2* (See Section 4.3.2) and Table 4 shows the optimal result for *VP-3* (See Section 4.3.3). By comparing all these three tables we can observe that *VP-2* gives better results compared to the other VPs. Besides, we note that weighted fusion with user specific threshold produces the best result where all the genuine users are never locked out from the system and 2% of the imposters are not being detected for one genuine user.

We notice furthermore that the results for VP-1 are not the best, although this might have been unexpected beforehand because the models are trained with data of all the impostors.

7.1.2. With Score Boost

Next we apply the Static Trust Model with the Score Boost technique followed by the Trust Calculation architecture given in Fig. 13. Similar to the previous settings we find that the 3-level static trust model with average fusion and weighted fusion performs better than the other settings. Algorithm 1 parameters are, $C = 0.5,\ W = 0.00 \sim 0.05$ and $P = 2$.

Table 5
Results for *VP-1* with the analysis method of Static Trust Model with Score Boost.

$T_{lockout}$	Categories	Average fusion				Weighted fusion			
		# User	ANGA	ANIA	# Imp. ND	# User	ANGA	ANIA	# Imp. ND
T_{us}	+/+	43		79		45		87	
	+/−	4		1231	28	4		1837	25
	−/+								
	−/−	2	366	1085	10				
90	+/+	32		87		34		85	
	+/−	7		777	31	5		598	15
	−/+	9	2575	94		9	4807	76	
	−/−	1	1890	1401	10	1	12,037	3543	13

Table 6
Results for *VP-2* with the analysis method of Static Trust Model with Score Boost.

$T_{lockout}$	Categories	Average fusion				Weighted fusion			
		# User	ANGA	ANIA	# Imp. ND	# User	ANGA	ANIA	# Imp. ND
T_{us}	+/+	48		127		49		109	
	+/−	1		589	1				
	−/+								
	−/−								
90	+/+	29		142		41		150	
	+/−	8		669	12				
	−/+	12	13,626	133		8	2880	72	
	−/−								

Table 7
Results for *VP-3* with the analysis method of Static Trust Model with Score Boost.

$T_{lockout}$	Categories	Average fusion				Weighted fusion			
		# User	ANGA	ANIA	# Imp. ND	# User	ANGA	ANIA	# Imp. ND
T_{us}	+/+	41		99		44		77	
	+/−	4		807	7	4		523	4
	−/+	3	4630	164		1	34,457	185	
	−/−	1	90,936	512	1				
90	+/+	26		89		34		88	
	+/−	4		1351	9	2		948	4
	−/+	18	6971	100		13	14,008	94	
	−/-	1	80,279	383	1				

We also apply the min–max score normalization techniques on *VP-1* and find that the number of best performing users is low (30 users) and the ANIA is high (129 actions) for the user specific lockout threshold.

Table 5, shows the optimal results we obtained from this analysis for *VP-1*. Here, the trust model and weighted fusion parameters are the same as for the previous analysis. The results we get from this analysis are comparable to the previous settings. According to this analysis we can say that the previous setting is performing slightly better than this system due to the nature of the static trust model.

Tables 6 and 7 show the optimal results we obtain from this analysis for *VP-2* and *VP-3* respectively. Similar to the previous analysis, we find that *VP-2* performs better than the other two and more precisely, weighted fusion with user specific threshold performs best where all the genuine users are never locked out and all the imposters are detected by the system.

7.2. Analysis of Dynamic Trust Model

To overcome the limitations of the Static Trust Model (see Section 5.3) and to further improve on the performance of the system, we apply the Dynamic Trust Model. The algorithm of the Dynamic Trust Model is provided in Section 5. This model is very similar to the 2-level trust model except that we use an efficient function to calculate the $\Delta Trust$ for this model. The main advantages of this model are that (1) the penalty and reward values are defined via a continuous function and (2) there is not a large difference in the trust change for classification scores just below or above the threshold. Due to the limited number of features we can extract from an action, there is a huge overlap between impostor and genuine actions scores. For this reason does this model work extremely well on the given dataset.

Algorithmic parameters for the above results are given below,

Dynamic Trust Model: The parameters for Algorithm 5 are, $A = \frac{1}{2n}\sum_{i=1}^{n} T_i^{svm} + T_i^{ann}$ (where, T^{svm} and T^{ann} are the training score set for SVM and ANN classifiers and n is the number of samples in the training set), $B = 0.01 \sim 0.5$, $C = 1$ and $D = 1$.
Weighted Fusion: The parameters for Algorithm 2 are, $Slope = 0.5,\ Tolerance = 10,\ UpperLimit = 1$.

7.2.1. Without Score Boost

In this section, we present the results we get from the Dynamic Trust Model (see Section 5.4) without applying the Score Boost technique followed by the Trust Calculation architecture given in Fig. 12. After exploring all the options for fusion rules, we find that the average and our proposed weighted fusion rules perform better than the other fusion rules.

Table 8, shows the optimal result we find from this analysis for *VP-1*. This result is obtained after exploring all the parameters of Eq. (2). These experimental results clearly indicate the potential improvement on the performance of the system over

Table 8
Results for *VP-1* with the analysis method of Dynamic Trust Model without Score Boost.

$T_{lockout}$	Categories	Average Fusion				Weighted Fusion			
		# User	ANGA	ANIA	# Imp. ND	# User	ANGA	ANIA	# Imp. ND
T_{us}	+/+	46		67		47		69	
	+/−					2		445	2
	−/+								
	−/−	3	1716	3988	70				
90	+/+	39		95		15		287	
	+/−	7		384	11	34		872	69
	−/+								
	−/−	3	850	279	16				

Table 9
Results for *VP-2* with the analysis method of Dynamic Trust Model without Score Boost.

$T_{lockout}$	Categories	Average fusion				Weighted fusion			
		# User	ANGA	ANIA	# Imp. ND	# User	ANGA	ANIA	# Imp. ND
T_{us}	+/+	49		83		49		79	
	+/−								
	−/+								
	−/−								
90	+/+	48		86		49		86	
	+/−	1		1665	1				
	−/+								
	−/−								

Table 10
Results for *VP-3* with the analysis method of Dynamic Trust Model without Score Boost.

$T_{lockout}$	Categories	Average fusion				Weighted fusion			
		# User	ANGA	ANIA	# Imp. ND	# User	ANGA	ANIA	# Imp. ND
T_{us}	+/+	46		74		48		101	
	+/−	3		390	3	1		337	1
	−/+								
	−/−								
90	+/+	46		81		46		68	
	+/−	3		509	3	3		440	3
	−/+								
	−/−								

Table 11
Results for *VP-1* with the analysis method of Dynamic Trust Model with Score Boost.

$T_{lockout}$	Categories	Average fusion				Weighted fusion			
		# User	ANGA	ANIA	# Imp. ND	# User	ANGA	ANIA	# Imp. ND
T_{us}	+/+	46		62		48		71	
	+/−					1		805	1
	−/+								
	−/−	3	10,498	14,462	114				
90	+/+	26		227		28		311	
	+/−	20		654	42	21		662	31
	−/+								
	−/−	3	1925	5002	76				

the analysis discussed above. From these results we can clearly conclude that all the users have $T_{us} > 90$ for weighted fusion. Note specifically also that in case we apply weighted fusion no genuine user is locked out by the system.

Tables 9 and 10, show the optimal results we obtain from this analysis for *VP-2* and *VP-3* respectively. Again we find that *VP-2* outperforms the other two for both the fusion techniques with user specific threshold, where all the genuine users are never locked out and all the imposters are detected by the system. However, a fixed threshold with weighted fusion technique also produces similar result on *VP-2* which is not seen in the previous analysis techniques.

Table 12
Results for *VP-2* with the analysis method of Dynamic Trust Model with Score Boost.

$T_{lockout}$	Categories	Average fusion				Weighted fusion			
		# User	ANGA	ANIA	# Imp. ND	# User	ANGA	ANIA	# Imp. ND
T_{us}	+/+	49		87		49		70	
	+/−								
	−/+								
	−/−								
90	+/+	49		92		49		75	
	+/−								
	−/+								
	−/−								

Table 13
Results for *VP-3* with the analysis method of Dynamic Trust Model with Score Boost.

$T_{lockout}$	Categories	Average fusion				Weighted fusion			
		# User	ANGA	ANIA	# Imp. ND	# User	ANGA	ANIA	# Imp. ND
T_{us}	+/+	47		82		48		87	
	+/−	2		540	2	1		364	1
	−/+								
	−/−								
90	+/+	47		87		48		93	
	+/−	2		733	2	1		364	1
	−/+								
	−/−								

In this analysis, significant difference between *VP-1* (see Table 8) and *VP-3* (see Table 10) are observed. All the techniques in *VP-3* produce the result where genuine users never locked out, but in case of *VP-1* only weighted fusion produces the same result. Also we find 4.2% of undetected imposters for 34 users in case of *VP-1* with weighted fusion and fixed threshold, whereas the same techniques produce only 2% of undetected imposters for 3 users in the case of *VP-3*.

7.2.2. With Score Boost

In this section, we show the results we get from the Dynamic Trust Model with applying the Score Boost technique followed by the Trust Calculation architecture given in Fig. 13. The parameters for Algorithm 1 are, $C = 0.5,\ W = 0.00 \sim 0.05$ and $P = 2$.

Table 11, shows the optimal results we found from this analysis for *VP-1*. These results clearly show that this method outperforms all other methods discussed in this paper. The total number of users for the 'All Positive' category equals 48 and for 1 genuine user, there was 1 impostor not detected.

Tables 12 and 13 show the optimal results we obtained from this analysis for *VP-2* and *VP-3* respectively. Overall, we find this analysis is better than the all previous analyses where each Verification Process produces better results over other analyses. In this analysis, *VP-2* produces the best result of all the users for all the techniques with both the thresholds where genuine users never locked out and there are no undetected imposters.

7.3. Performance reporting in FMR and FNMR

In this section we report the overall system performance in terms of FMR and FNMR, although we are aware that this is a slight abuse of terminology in the context of continuous authentication with our analysis methods. In this case we consider

Table 14
Results in terms of (FNMR, FMR).

Techniques	*VP-1*	*VP-2*	*VP-3*
STM-WSB	(0%, 1.36%)	(0%, 0.04%)	(0%, 0.17%)
STM-SB	(0%, 1.06%)	(0%, 0%)	(2.04%, 0.17%)
DTM-WSB	(0%, 0.09%)	(0%, 0%)	(0%, 0.04%)
DTM-SB	(0%, 0.04%)	(0%, 0%)	(0%, 0.04%)

FMR as the probability that an impostor user is not detected when his test data is compared to the classification model of a genuine user. Each impostor data was tested against 49 genuine users, which means that the total number of impostor tests equals $49 \times 48 = 2352$. Similarly we also define the FNMR here as the probability that a genuine user is falsely locked out by the system.

We report here the FNMR and FMR values for all the tests done in Sections 7.1 and 7.2. We restrict ourselves to only the optimal settings, *i.e.* the user dependent threshold T_{us} and the Weighted Fusion. For example we see in Table 7 that in that case 1 genuine user is locked out (see category −/+) and that 4 impostor users are not detected (category +/−). This implies that $FNMR = 1/49 = 2.0\%$ and $FMR = 4/2352 = 0.17\%$. All the results are presented in Table 14. The columns in this table show the results for the various VPs and the rows represent the evaluation scenarios. Here STM stands for Static Trust Model and DTM stand for Dynamic Trust Model. Also SB means Score Boost and WSB means Without Score Boost.

7.4. Result discussion

It is not surprising that from Tables 2–13 we consistently see that the personal threshold performs better than the fixed system threshold. When only considering the personal threshold, we can also see in almost all of the tables that the Average Fusion does not perform as well as the Weighted Fusion. The differences in performance between using Score Boost and not using Score Boost are not that large. We can see this from comparing the results in Sections 7.1.1 and 7.2.1 with the results in Sections 7.1.2 and 7.2.2. On the other hand, we can see that the Dynamic Trust Model outperforms the Static Trust Model.

We can consistently see that *VP-2* performs better than the other verification processes and also *VP-3* performs better that *VP-1*. Due to the varied nature of the behavioral biometric paradigm, we can clearly assume that the increment of the number of imposters on the training phase can also affect the classifier models to recognize the actions. We can validate this assumption by looking at the classifier's training accuracy where we note that the classifier's training accuracy improves for *VP-2* and *VP-3* when compare to *VP-1*. Similar findings were reported in the research conducted by Pusara [23].

In Continuous Authentication Systems, the objective is not just to achieve 0% FMR and FNMR but also to reduce the number of actions performed by the imposters before getting detected by the system and increase the number of actions performed by the genuine users before falsely being locked out by the system. In total, we can conclude that the best performance is reached wit an average of 70 actions required to detect an imposter when using personal thresholds in combination with boosted score and weighted fusion in a dynamic trust model where none of the genuine users falsely lockout from the system.

Table 15 shows the best performance for all the verification processes. In this table, we can clearly see that none of the genuine users are lockout from the system for all the three verification processes (*i.e. VP-1*, *VP-2*, and *VP-3*). We also observe that for *VP-1*, 48 users are the best performing users (*i.e.* +/+ category) with an ANIA of 71 (Standard Deviation (SD) = 47, *t*-value = 0.92, *p*-value = 0.46) and one user has an ANIA of 805 with one undetected imposter. For *VP-2* we have all the 49 users are best performing users with an ANIA of 70 (SD = 56, *t*-value = 0.82, *p*-value = 0.5). In case of *VP-3*, we find that 48 users are the best performing users with an ANIA of 87 (Standard Deviation (SD) = 57, *t*-value = 0.93, *p*-value = 0.4) and one user has an ANIA of 364 with one undetected imposter. In all cases we see that the *p*-values indicate that the results found in this research are statistically significant [17].

8. General discussion and recommendations

In this section, we discuss some general issues related to this research and provide some recommendations related to the usefulness of our research for the future.

8.1. Significance of this new scheme

We find that the state of the art research on continuous authentication uses either the whole test set or chunks of a large, fixed number of actions for the analysis. This implies that an impostor at least can perform that fixed number of actions before the system checks the identity. This is then in fact no longer continuous authentication, only at best periodic

Table 15
Best performance for all the verification processes.

Verification process	Type-II error		Type-I error				
	# User	ANGA	# User	ANIA	SD	*t*-value	*p*-value
VP-1	49	∞	48	71	47	0.92	0.46
			1	805 (# Imp. ND = 1)			
VP-2	49	∞	49	70	56	0.82	0.5
VP-3	49	∞	48	87	57	0.93	0.4
			1	364 (# Imp. ND = 1)			

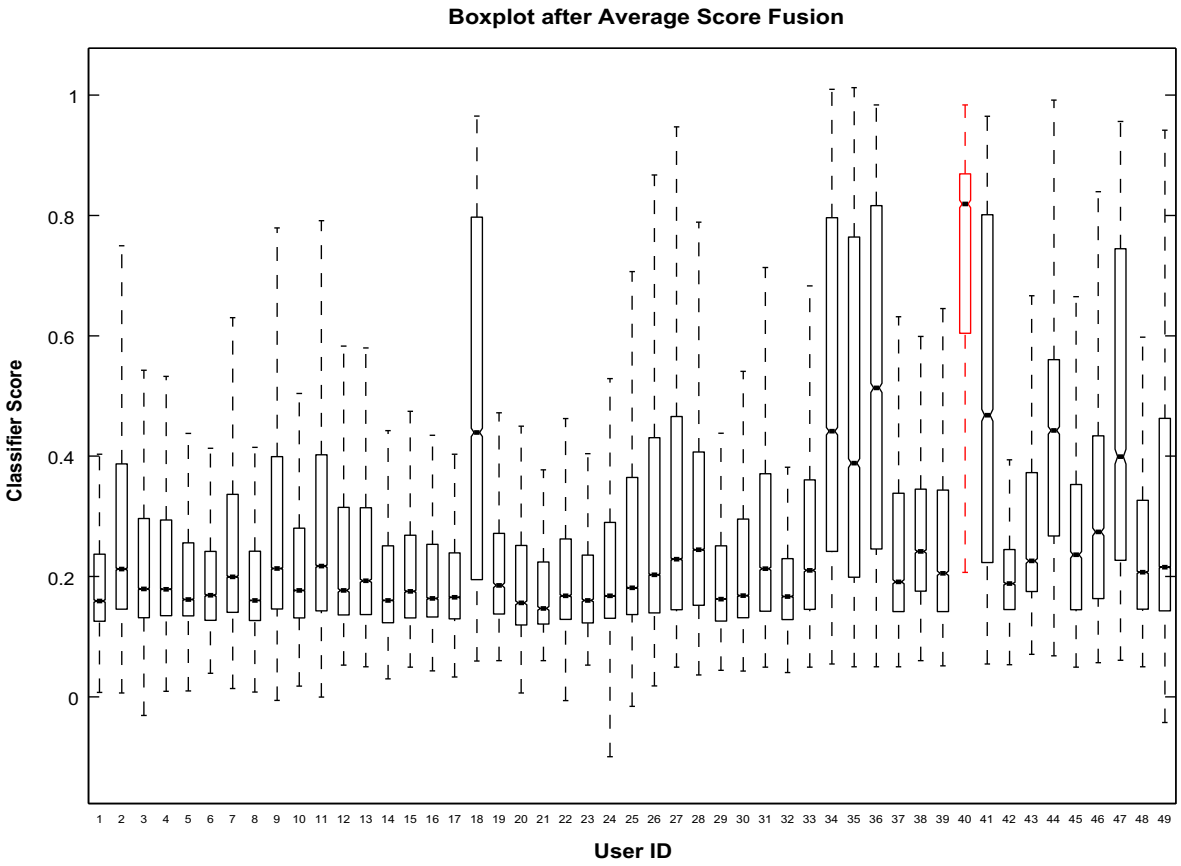

Fig. 14. Distribution of the classifier score after average fusion for the best performing user for *VP-2*. Here, the genuine user is number 40.

authentication. As for our understanding actual continuous authentication should react on every single action performed by the user. By doing this the challenges are eminently because, the system cannot apply any *Schematic features* (*i.e.* Mouse action histogram, Percentage of silent periods, Distribution of cursor positions, etc.) for the analysis which may be helpful to recognize the user. However, still we believe that this new scheme should be the correct approach for continuous authentication systems.

8.2. Significance of Dynamic Trust Model

The proposed Dynamic Trust Model trust model is a very significant contribution to the Continuous Authentication research. This algorithm is a powerful algorithm to handle the huge overlapping between genuine user scores and the imposter users scores. In Fig. 14, we show the distribution of the classifier scores after average fusion for the best performing user where on an average 9 actions are required to detect an imposter for that user. We see in this figure that the median score of user is higher than 75% of all scores of any other user. But for other users the distribution of the classifier score after average fusion looks more like Fig. 15. In this case the score distribution of the genuine user does not differ a lot from the distributions of scores of other users. However, the algorithm described in this paper will still produce satisfactory results. Please note that these two figures were generated by the classifier score without using Score Boost technique.

8.3. Algorithmic parameters

In this research, all the algorithmic parameters are optimized based on the training dataset, more precisely, the score distribution of the training data. Apart from that we apply sequential searching for the parameters to obtain the optimized results. We first try different fixed threshold as $T_{lockout}$ ranging from 50 to 90 and then choose the user specific lockout threshold (T_{us}), where we try to avoid the genuine user from being locked out from the system.

We would like to mention that all the algorithms used in this research are sensitive to the parameters, meaning that the choice of one wrong parameter can lead to a disappointing result. We can understand this by observing Fig. 15, where the room for wrong parameter selection is tiny. Using a cross validation set with optimization algorithms (*i.e.* Genetic Algorithm, Particle Swarm Intelligence, etc.) for algorithmic parameters optimization can improve the results, but due to limitations of

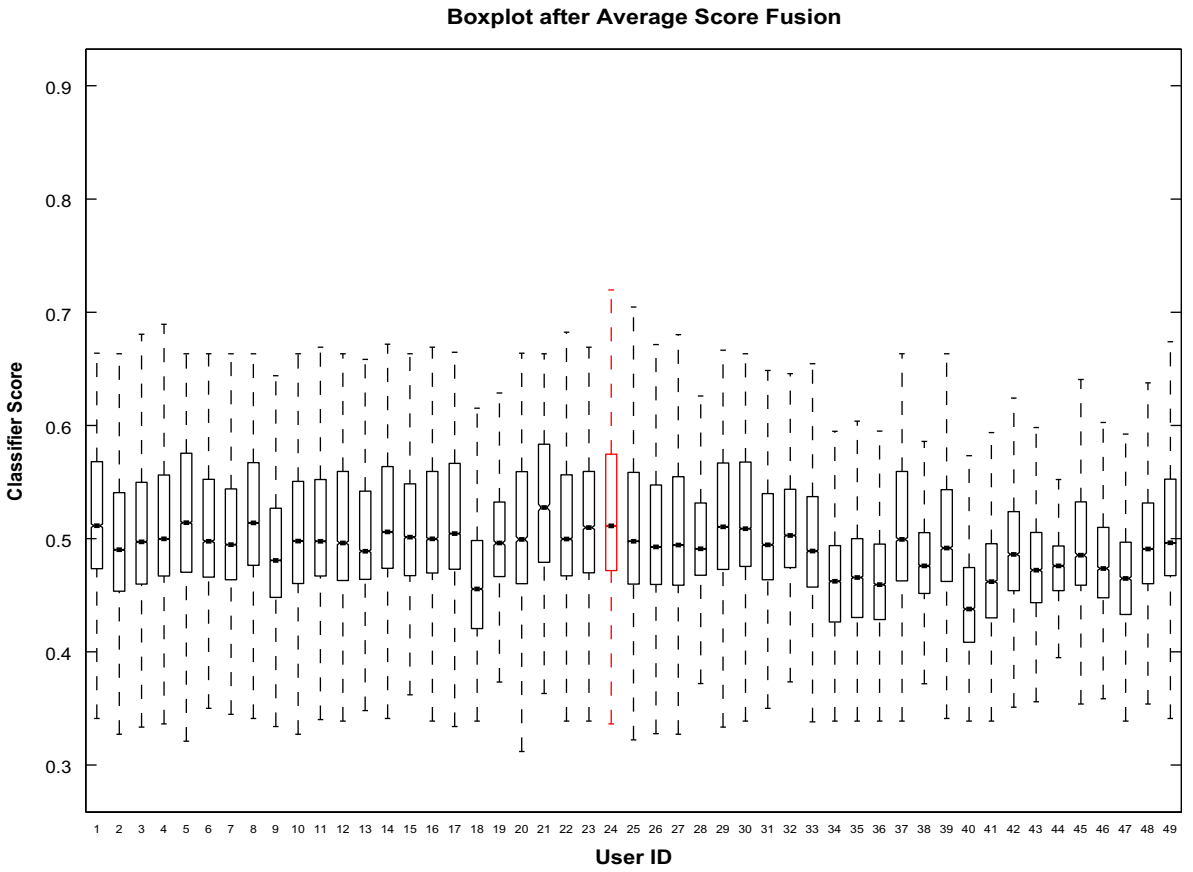

Fig. 15. Distribution of the classifier score after average fusion for the below average performing user for *VP-2*. Here, the genuine user is number 24.

this dataset (see Section 4.1), we are unable to build a validation set. Because of the unavailability of any other proper publicly available continuous authentication dataset, we were bound to apply our research on this dataset. However, the algorithms and methods applied in this research can be applied on any continuous authentication dataset, irrespective of the biometric modality, as long as classification scores are available.

8.4. Discussion on context dependency

As stated previously, the used dataset was collected in an uncontrolled environment in a continuous manner. On average this dataset has 14 hours of data per user. However, we do not know how much variation of the data has been introduced during data collection because this dataset was collected in a completely uncontrolled environment. In our research, we use a single biometric signature per user, but we do feel that performance can be improved by using multiple biometric signatures for the same user, each related to a specific situation. Data clustering can be used to produce multiple signatures for the same user, but we need more experimental evidence to validate this argument which is beyond the scope of this dataset.

8.5. Significance of the upper limit of the trust level

In our analysis, we restrict the upper limit of the trust level to 100. The reason for this is that if the genuine user would be working for a longer time, in principle the trust could rise to a high level, say for example to 1000. This means that if an impostor would take over the computer then, he would not be detected by the system until the trust level drops below the lockout threshold. In case of no upper limit, this means that the impostor could first profit from the trust that was build up by the genuine user, allowing him/her more time for malicious activities.

8.6. Significance of analysis in the action domain

We perform our analysis in the action domain rather than in time domain. Based on our understanding does the number of actions performed by the users within a specific time period depend highly upon the individual's specific behavior.

Table 16
Results for *VP-2* with the analysis method of Static Trust Model without Score Boost.

Categories	SVM				ANN			
	# User	ANGA	ANIA	# Imp. ND	# User	ANGA	ANIA	# Imp. ND
+/+	46		136		43		103	
+/−	3		1958	9	6		412	8
−/+								
−/−								

Table 17
Results for keystroke dynamics with our analysis.

Categories	DTM without SB				DTM with SB			
	# User	ANGA	ANIA	# Imp. ND	# User	ANGA	ANIA	# Imp. ND
+/+	30		155		32		374	
+/−	14		620	29	11		1053	21
−/+	5	2299	220		9	8683	757	
−/−	4	13,172	578	9	1	1717	904	2

Therefore, we have decided to report the performance of the continuous authentication system in the action domain. We analyzed how much time is required to find an imposter in our optimal analysis method and we found that it took on average of 23.4 min ($\pm$18.8 min) to detect an impostor when including the *Silence Periods*.

8.7. Significance of the multiple classifiers

We first apply the 3-level static trust model for *VP-2* with SVM as a classifier on the same dataset and obtained ANIA as 136 only for 46 best category users by using users specific lockout threshold (T_{us}). We then apply ANN with the same analysis technique and obtained ANIA as 103 only for 43 best category users. Table 16, shows the result obtained from this analysis for *VP-2* with the analysis method of Static Trust Model without Score Boost. During this analysis, we observe that some users perform better with a particular classifier than with the other classifier. Therefore, we use the multi-classifier fusion technique to obtain the optimal result. We can compare this result with the result shown in Table 3 where we can observe the clear improvement on the result by using multi-classifier fusion.

8.8. Comparison with previous research

Nakkabi et al. [22] achieved an FMR of 0% and an FNMR of 0.36% for 48 users with the same dataset that we used in our analysis. The number of actions they used in this research is bit unclear; but, from their previous research they have mentioned 2000 actions for the analysis [1]. Therefore, if we consider the same number of actions and convert it in terms of ANIA/ANGA (discussed in Section 6.1) then ANIA $= 2000$ and ANGA $\approx 555,556$. The ANGA is comparable with our result, meaning that the genuine user is in practice never locked out from the system but, the ANIA is very high compare to our result. Also, we need to mention that they have gotten this result based on only 48 of the 49 users.

8.9. Analysis on Keystroke Dynamics

We perform a small analysis on a keystroke dynamic dataset with our best performing techniques. We used our own data logging software and collected data of 53 volunteers, where the data was collected unrestricted and continuously for 5–7 days. The collected data is separated to build and train the system (training dataset), parameter adjustment of the algorithms used in this research (validation dataset) and finally to test the system performance (test dataset). Approximately 35% of the total data is used as training dataset, 10% as validation dataset and the remaining 55% as test dataset. We use state of the art keystroke dynamics features [2] and followed the *VP-2* verification process. In this analysis, all the algorithmic parameters and lockout threshold are optimized based on *Genetic Algorithm*. Table 17 shows the results by using the Dynamic Trust Model (DTM) with and without the Score Boost (SB) technique. We observe that our proposed methods are also applicable to a keystroke dynamics dataset.

9. Conclusion

Our analysis is complete in the sense that we try three different verification processes and all different combinations of the 4 different settings that we used (threshold setting, score boosting, dynamic vs. static trust model, and fusion). Although

we apply the analysis in this research to a dataset with continuous mouse dynamics data, are the techniques general enough that they can also be applied to, for example, keystroke dynamics data or any other biometric data that is used for continuous authentication. We have shown that our results improve significantly over the previously known performance results on this dataset, even though it is hard to compare the results in a straightforward manner.

Our analysis is focused on the Trust Calculation module (see the dotted box in Fig. 9). We like to mention that different pre-process techniques or different classifier selection can influence the results. In the future we will apply the described techniques also on a dataset that contains both keystroke dynamics data and mouse dynamics data, to see if we can improve the results on such datasets as well.

References

[1] A. Ahmed, I. Traore, A new biometric technology based on mouse dynamics, IEEE Trans. Dependable Secure Comput. 4 (3) (2007) 165–179.
[2] L. Araujo, L.H.R. Sucupira Jr., M. Lizarraga, L. Ling, J.B.T. Yabu-Uti, User authentication through typing biometrics features, IEEE Trans. Signal Process. 53 (2) (2005) 851–855.
[3] R.M. Bergner, What is behavior? and so what?, New Ideas Psychol 29 (2) (2011) 147–155.
[4] C.M. Bishop, Neural Networks for Pattern Recognition, Oxford University Press Inc., New York, NY, USA, 1995.
[5] P. Bours, Continuous keystroke dynamics: a different perspective towards biometric evaluation, Inf. Secur. Techn. Rep. 17 (2012) 36–43.
[6] C. Burges, A tutorial on support vector machines for pattern recognition, Data Min. Knowl. Disc. 2 (2) (1998) 121–167.
[7] Z. Cai, C. Shen, X. Guan, Mitigating behavioral variability for mouse dynamics: a dimensionality-reduction-based approach, IEEE Trans. Hum.–Mach. Syst. 44 (2) (2014) 244–255.
[8] C.-C. Chang, C.-J. Lin, Libsvm: a library for support vector machines, ACM Trans. Intell. Syst. Technol. 2 (3) (2011) 27:1–27:27.
[9] I. Deutschmann, J. Lindholm, Behavioral biometrics for DARPA's active authentication program, in: IEEE International Conference of the Biometrics Special Interest Group (BIOSIG'13), IEEE, 2013, pp. 1–8.
[10] R. Everitt, P.W. McOwan, Java-based internet biometric authentication system, IEEE Trans. Pattern Anal. Mach. Intell. 25 (9) (2003) 1166–1172.
[11] C. Feher, Y. Elovici, R. Moskovitch, L. Rokach, A. Schclar, User identity verification via mouse dynamics, Inf. Sci. 201 (0) (2012) 19–36.
[12] H. Gamboa, A. Fred, An identity authentication system based on human computer interaction behavior, in: 3rd International Workshop on Pattern Recognition in Information Systems, 2003.
[13] H. Gamboa, A. Fred, A behavioral biometric system based on human–computer interaction, in: Biometric Technology for Human Identification, SPIE, vol. 5404, 2004, pp. 381–392.
[14] A. Jain, R.P.W. Duin, J. Mao, Statistical pattern recognition: a review, IEEE Trans. Pattern Anal. Mach. Intell. 22 (1) (2000) 4–37.
[15] Z. Jorgensen, T. Yu, On mouse dynamics as a behavioral biometric for authentication, in: 6th ACM Symposium on Information, Computer and Communications Security (ASIACCS '11), ACM, 2011, pp. 476–482.
[16] J. Kittler, M. Hatef, R.P.W. Duin, J. Matas, On combining classifiers, IEEE Trans. Pattern Anal. Mach. Intell. 20 (3) (1998) 226–239.
[17] E.L. Lehmann, J.P. Romano, Testing Statistical Hypotheses, Springer Texts in Statistics, Springer, 2005.
[18] C.-C. Lin, C.-C. Chang, D. Liang, A new non-intrusive authentication approach for data protection based on mouse dynamics, in: IEEE International Symposium on Biometrics and Security Technologies (ISBAST'12), IEEE, 2012, pp. 9–14.
[19] S. Mondal, P. Bours, Continuous authentication using mouse dynamics, in: IEEE International Conference of the Biometrics Special Interest Group (BIOSIG'13), IEEE, 2013, pp. 1–12.
[20] F. Monrose, A. Rubin, Authentication via keystroke dynamics, in: 4th ACM Conference on Computer and Communications Security (CCS '97), ACM, 1997, pp. 48–56.
[21] I.T. Nabney, Netlab: Algorithms for Pattern Recognition, Advances in Computer Vision and Pattern Recognition, vol. XVIII, Springer, 2002.
[22] Y. Nakkabi, I. Traoré, A. Ahmed, Improving mouse dynamics biometric performance using variance reduction via extractors with separate features, IEEE Trans. Syst. Man Cybern. – Part A: Syst. Hum. 40 (2010) 1345–1353.
[23] M. Pusara, An Examination of User Behavior for Re-Authentication. PhD thesis, CERIAS, Purdue University, 2007.
[24] M. Pusara, C. Brodley, User re-authentication via mouse movements, in: ACM Workshop on Visualization and Data Mining for Computer Security (VizSEC/DMSEC '04), ACM, 2004, pp. 1–8.
[25] A. Ross, A. Jain, Information fusion in biometrics, Pattern Recogn. Lett. 24 (13) (2003) 2115–2125.
[26] D. Schulz, Mouse curve biometrics, in: Biometrics Symposium: Special Session Research at the Biometric Consortium Conference, 2006, pp. 1–6.
[27] C. Shen, Z. Cai, X. Guan, Continuous authentication for mouse dynamics: a pattern-growth approach, in: 42nd Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN'12), IEEE, 2012, pp. 1–12.
[28] C. Shen, Z. Cai, X. Guan, Y. Du, R. Maxion, User authentication through mouse dynamics, IEEE Trans. Inf. Forensics Secur. 8 (1) (2013) 16–30.
[29] S. Shepherd, Continuous authentication by analysis of keyboard typing characteristics, in: European Convention on Security and Detection, IEEE, 1995, pp. 111–114.
[30] R. Snelick, U. Uludag, A. Mink, M. Indovina, A. Jain, Large-scale evaluation of multimodal biometric authentication using state-of-the-art systems, IEEE Trans. Pattern Anal. Mach. Intell. 27 (3) (2005) 450–455.
[31] N. Zheng, A. Paloski, H. Wang, An efficient user verification system via mouse movements, in: 18th ACM Conference on Computer and Communications Security (CCS '11), ACM, 2011, pp. 139–150.