An empirical biometric-based study for user identification with different neural networks in the online game League of Legends

DA SILVA, V R and DA COSTA ABREU, Marjory <http://orcid.org/0000-0001-7461-7570>

An empirical biometric-based study for user identification with different neural networks in the online game League of Legends

Valmiro Ribeiro da Silva
DIMAp-UFRN
valmirozunoribeiro@gmail.com

Marjory Da Costa-Abreu
DIMAp-UFRN
marjory@dimap.ufrn.br

***Abstract*—The popularity of computer games has grown exponentially in the last years. Although such games were created to promote competition and promote self-improvement, there are some recurrent issues. One that has received the least amount of attention so far is the problem of "account sharing" which is when a player shares his/her account with more experienced players to make progress in the game. The companies running those games tend to punish this behaviour, but this specific case is hard to identify. Since, the popularity of neural networks has never been higher, the aim of this study is to investigate how different neural network algorithms behave when analysing a database of biometric information (keystroke and mouse dynamics) regarding the game League of Legends, and how those algorithms are affected by how frequently a sample is collected.**

I. Introduction

Online games have become very popular and diverse since their beginning in the 80's, and each device gives us several biometric modalities to be exploited, such as gait, keystroke dynamics, mouse dynamics, touch-screen dynamics, etc. The diversity of input data is endless and it makes the game an unique experience for the player.

Even though all the previously listed biometrics are used for the same purpose in the gaming universe, they can be very different when analysed in the traditional security and authentication applications. Thus, it is important to understand its differences from the traditional approaches[1], [2].

As a very simple example of how different the security application is from a traditional authentication task to a game authentication using biometric data, take it the keystroke dynamics modality in a continuous verification scenario:

- In a traditional verification problem, the user's behaviour is expected to suffer very little variation while typing an e-mail.
- On the other hand, in a game verification problem, the user's behaviour is expected to change and that change will be based on the configurations of the game he/she is using, e.g. the role in the game, the abilities it chose to use, the character he/she is playing with and so on.

The second case can not be considered the same as the first, because, despite the fact they are both verification problems and are using the same base data, the user's behaviour is different which makes the security system to model it in a different way. To the best of our knowledge, no other work has tried to investigate this specific problem.

The main goal of this paper is to investigate how different neural networks perform when classifying gaming biometric data, using the software WEKA to run experiments with different datasets created from the same samples, with different time windows to collect features in each one of this datasets.

II. Biometric modalities used in desktop based game playing

The market of egames is huge and, with the recent advancement of virtual reality, the range of consoles (the hardware you need to play games) has increased greatly. The kinds of devices used to play go from simple keyboard and mouse to very expensive virtual reality glasses. However, the most popular is still the computer-based one for it is indisputable the cheapest [3].

In a security point of view, each different kind of console will have different vulnerabilities, but the "black-box" type, the ones you buy and does not need to install any software are, until certain extent, more secure. When we are talking about private computer-based games, we have a limitation in devices that we can use to play, but the possibility of a user to play with another user's account is more evident.

Considering that we are using League of Legends as our case study, the modalities chosen to investigate are mouse and keystroke dynamics, because both peripherals are mandatory used together during the matches.

Keystroke dynamics is the unique timing patterns embedded in an individual's typing and is most often developed in a personal way, hence the use of keyboard dynamics as a biometrics-based identification modality. Processing of such data includes extracting keystroke timing features such as the duration of a key press and the time elapsed between successive key presses [4], [5].

Mouse dynamics is the unique speed movements and frequency of clicks generated by a user using the mouse. The move speed is how fast the user moves the mouse in the 8 possible mapped directions and frequency of clicks is the amount of clicks the users performs in a time interval [6].

Since we are using League of Legends as our case study, it is important to understand how keystroke and mouse dynamics

are used in the context of the game. The next subsection will introduce the basics of the game, as well as how our modalities can be used in context, followed by subsections enumerating the related work to mouse and keystroke dynamics.

A. League of Legends

League of Legends is a Multiplayer Online Battle Arena (MOBA) game. The game is based around matches with two teams of (normally) five players each, where each team tries to destroy the main base of the other. Before each match starts, each player chooses a champion to play, which is an avatar that already exists in the game, with predetermined statistics (stats) and skills. Two players of the same team cannot choose the same champion.

Each champion has four unique skills, where three are common skills and the last is a ultimate skill. Skills can grant passive or active abilities, where each skill is activated by the keys 'Q', 'W', 'E' and 'R', the last one used to activate the ultimate ability.

Each team member follows one of the defined roles during a match:

- Top Laner: know as "top", this player starts at the top of the map, and usually is a melee attacker;
- Jungler: This player spends most of his time defeating jungle's monsters in order to gain bonus statistics to the team. Champions with high mobility usually take this role;
- Mid Laner: Also known as mid, this player starts in the middle of the map, and uses his skill set to create combos to deal damage. Champions with synergic skill set usually take the role;
- Carry: also known as ADC (attack damage carry) is responsible take down buildings and clear minions waves. Ranged champions with high damage usually take the role.
- Support: Starts in the bottom lane and is responsible to support the ADC ans later the whole team. Champions with good supporting abilities or tanks usually take the role;

The main features used in League of Legends selected for the analysis of this paper from both keyboard and mouse dynamics can be described as follows:

- Keystroke dynamics: 'Q', 'W', E', 'R' (for the unique skills) and 'SPACE' (used to make the camera follow the player's champion);
- Mouse dynamics: Move the character (point and click in a empty space using the right mouse button), basic attacks (clicking in a enemy using the right mouse button) and target skills (using the left mouse button);

As already said previously, to the best of our knowledge, there is no other work which has investigates the individual variations of keystroke dynamics and mouse dynamics (biometrics) in the context of online games. Section III will present the main works that can be found using mouse and keystroke dynamics.

III. Keystroke dynamics, mouse dynamics and game-related work

Keystroke dynamics is a much older biometrics modality than the mouse dynamics, thus the number of databases available is larger. This is expected because the use of "mouse" is very much associated with the personal computer whereas the keystroke exists since the use of Morse code.

In [7] data from 16 graduate students from Seul National University was collected to try to detect user/password sharing, where each of them typed a fixed set of 25 username/passwords 30 times using their own keyboards. A generic approach was used, using the information known about keystroke dynamics, where a user's pattern is consistent and different from other individuals, and assuming that each user's patterns form a "cluster" in Euclidean space, making possible to estimate the number of sharers by the number of clusters formed. This experiment had an EER (Equal Error Rate) estimating the sharers of 6.51%.

In [8] a database containing soft-biometrics and keystroke dynamics from 110 volunteers from France and Norway is presented, where users were classified using Support Vector Machine (SVM) with an EER of 21% to 4% when the soft-biometrics were added. The work was expanded in [9] using a fusion approach, where the SVM algorithm was used to classify the fused data, with an EER of 10%.

In [10] a new approach to emotion recognition using pressure sensor keyboards was described. Fear, happiness, anger, sadness, surprise and neutral emotions were tested, with a EER of 12.02% when using only traditional keystroke methods to classify the subjects with the KNN algorithm.

In [11], a database using various fusion approach on keystroke dynamics was collected. Each user was allowed to choose their preferable username and password during the enrolment process and they were asked to type one fixed text for fifteen consecutive times, with an EER of 9% using SVM and combining features.

A login method for accessing computer systems using mouse dynamics was described by [6]. 28 users performed a fixed task of moving the mouse between two lines. They were classified using Levenshtein distance to calculate similarities, also know as edit distance, with an EER of 26.8%.

In [12] the experiment focused in behavioral variability, where they provided free use of the mouse. Data from 5 users were gathered. A preprocessing was made using PCA, decreasing the number of important mouse features to 8, and after that an artificial neural network was used to classify the volunteers, with an EER of 5%. Later, in [13] the goal was to compare two hypo-optimum feature selectors and evaluate the methods to obtain the best combination of features for continuous identity authentication and monitoring. 20 users were asked to install on his/her workstations and to continue to work normally while the mouse data was collected. After a preprocessing to select the best features, the users were classified using SVM with an EER of 5.32%.

Pattern-growth-based mining was used to extract frequent behavior segments in obtaining stable mouse characteristics

in [14], using classification algorithms to perform continuous user authentication. 22 users performed Internet surfing, word processing, online chatting and programming for 30 minutes. The best result was an EER of 1.49% using a One-Class SVM detector.

The literature does not have a large amount of multimodal systems using mouse and keystroke biometric data. Additionally, work related to online games are very limited.

An approach using game-play activities was proposed in [15] with the purpose to attack the account sharing problem, where the idle time distribution of a player in-game was proved to be a representative feature, and the RET scheme was proposed for user identification, which is based on the Kullback-Leibler divergence between idle time distributions. The results showed that the RET scheme achieves higher than 90% accuracy with a 20-minute detection time given a 200-minute history size.

According to [16], the behavior of a player in a match can be used as a metric for identification in some cases. The authors used poker as case study, calculating the percentage of folds, calls, checks, raises, re-raises and all-ins, using euclidean distance to calculate similarity to verify 30 players identities, with an EER of 22.67%.

In [17], a total of 24 subjects were asked to perform a fixed task using their own computers, where the objective was to analyze both biometric modalities together and isolated. The classifier had an EER of 8.21%. The mouse dynamics isolated had a EER of 22.41% and the keystroke dynamics alone had an EER of 24.78%.

In [18], 25 volunteers participated in the experiment and four different machine learning approach were use:, Decision Tree (DT), Counter-Propagation Artificial Neural Network (CPANN), Artificial Neural Network (ANN) and SVM. In this work, an identification accuracy of 62.2% was obtained in a closed-set experiment and a Detection and Identification Rate of 58.9% in an open-set experiment.

For this work we have used the biometrics database collected in [3] using League of Legends as case study. Data from 56 different users were collected, using the same type of keyboard and mouse to all volunteers, where 18 users played more than one time, sometimes using different characters and/or positions. Our analysis will focus on this group. The original proposition was to verify if a user is himself/herself, attacking the account sharing problem more precisely, but the amount of data collected was not sufficient to make the experiments, thus changing the subject to try to identify the the volunteers. More accurate, real data could not be used to experiment because the game is from a private company, making it difficult to have access to the company's data.

The goal in [3], and then [1] and [2], was to use the database for identification. For this purpose, the software WEKA was utilised in order to run machine learning algorithms trying to identify correctly each user. The best result combining keystroke and mouse dynamics in [1] and [2] was 90.77% using the Random Forest algorithm, as shown in both works.

Each sample collected in these works have data of 33 different features:

- 13 keystroke features:
  - Three combination of keys, using the distance between keys, C1 (Q & W, W & E, E & R), C2 (Q & E, W & R) and C3 (Q & R) , also called combos;
  - Frequency (per minute of match) for each key pressed (FQ, FW, FE, FR and FSPACE);
  - Latency for each key pressed (Q, W, E, R and SPACE).
- 20 mouse features:
  - Move speed of the 8 directions - 'Down', 'Down + Left', 'Left', 'Up + Left', 'Up', 'Up + Right', 'Right' and 'Down + Right - represented by D1, D2, D3, D4, D5, D6, D7 and D8, respectively;
  - The acceleration for each direction, represented by AD1, AD2, AD3, AD4, AD5, AD6, AD7 and AD8, respectively;
  - Frequency and Latency for right and left clicks, represented as CFR, CFL (for frequency) and CTR and CTL (for latency).

IV. Experimental and statistical analysis

In order to discover which network can give us the best results for our data the software WEKA (Waikato Environment for Knowledge Analysis) was used to perform experiments [19]. We have selected all the available networks in this platform to evaluate the data: Multilayer Perceptron, Bayesian Network and Radial Basis Function Network. The three next subsections will describe the networks and the parameters used in our experiments, and the later subsections will detail how the more appropriate time stamp to extract the features was chosen and also how the different networks performed when trying to classify the data.

A. Multilayer Perceptron (MLP)

MultiLayer Perceptron (MLP) is an artificial network model with at least three layers that maps inputs to a set of appropriate outputs [20]. The layers commonly are input layer, hidden (or intermediate) layers and output layer.

Input layers are responsible for receiving and propagate the input information, without the need of do any processing. Hidden layers are composed by nodes, and are responsible for processing data and transmitting the information through connections between inputs and outputs. To ensure that a network is keeping knowledge, these connections save the weights that will be multiplied by the inputs. Output layers are composed of neurons, which receives the information from the hidden layers, yielding the output (answer to the problem).

WEKA's implementation [21] treat all nodes as sigmoid (can be used in backpropagation), and it let us change atributtes as Momentum Rate for backpropagation, Learning Rate, training time, and the number of hidden Layers.

Our experiments with MLP were conducted using two different configurations, and only the training time was changed. Bellow we can see the configurations used in our experiments.

Config1: *hiddenLayers = a, learning Hate = 0.7, momentum = 0.6, trainingTime = 5000, threshold = 20, validationSet = 0.*

Config2: *hiddenLayers = a, learning Hate = 0.7, momentum = 0.6, trainingTime = 500, threshold = 20, validationSet = 0.*

B. Bayesian Network

Bayesian Network is a probabilistic graphical model that represents a set of random variables and their conditional dependencies using a directed acyclic graph (DAG). Nodes represent random variables and edges represent conditional dependencies. When two different nodes are not connected this represents that they are conditionally independent. Each node then is associated with a probability function, which takes as inputs a set of values for the node's parent variables, giving the probability of the variable represented by the node as output [22].

Weka's implementation of Bayesian Networks can use different search algorithms and quality measures [23]. The search algorithms used in this experiment were K2, that uses a hill climbing strategy restricted by an order on the variables [24][25][26], and the Tabu Search, which is a hill climbing until an optimum is reached, where the following step in the search is the least worst step [27][28]. The only estimator used for our experiments is the Simple Estimator, estimating directly from data the conditional probability tables of a Bayes network once the structure has been learned [29].

Two different configurations of Bayesian Networks were tested in our experiments, using the following configurations:

Config1: *Estimator = simpleEstimator (default configuration), search algorithm = TabuSearch.*

Config1: *Estimator = simpleEstimator (default configuration), search algorithm = K2.*

C. Radial Basis Function Network (RBF Network)

A Radial Basis Function (RBF) Network is a neural network that uses radial basis functions as activation functions. RBF networks can be regarded as feedfoward neural networks with a single layer for hidden nodes. The inputs for these neural networks is the distance between the input vector (activation) and its center (location) [30].

Their performance depends on the number and positions of the radial basis functions, their shape, and the method used to determine the weights. The outputs for this kind of neural network is a linear combination of radial basis functions of the inputs and neuron parameters.

WEKA's implementation of RBF Network uses the k-means clustering algorithm to provide the basis functions[31], which uses the gradient-descent method until convergence as default, making larger sets to run extremely slowly [32]. Putting a limit to the number of interactions can make de algorithm faster, but we can not guarantee that it will converge, thus making it hard to tell if there is a possibility to improve results.

Two configurations were tested, one with and one without limiting the number of iterations, with the following parameters:

Config1: *Clusttering seed = 1; maxIterations = -1 (until convergence), minStdDev = 0.1, number of Clusters = 2.*

Config2: *Clusttering seed = 1; maxIterations = 5, minStdDev = 0.1, number of Clusters = 2.*

Config3: *Clusttering seed = 1; maxIterations = 10, minStdDev = 0.1, number of Clusters = 2.*

D. Time Stamp Selection

In [3] the authors extracted the features every 5 minutes played, but it was not certain that this time frame was the best for the problem. In order to verify which time stamp is good enough to give us information to classify correctly our game biometric data a preprocessing using the original information of each user was applied to create five new bases, each one with a different time frame to extract the features (5 minutes, 4 minutes, 3 minutes, 2 minutes and 1 minute).

We may incorrectly think that shrinking the time window will give us more information simply because it generates more samples, but these samples do not always have enough significant information to perform the identification correctly.

E. Results

Tables I, II, IV, V,VII, VIII and IX show us the classification algorithms results. In Table I and II, we can see that the best cases were with the databases of 5 minutes and 4 minutes. However, the 4 minutes database have a lower standard deviation and more instances, indication that 4 minutes is a time window big enough to be representative to the MLP algorithm. A simple T-Test show us that the difference between the two databases is not big enough to be representative, consequently, 4 minutes is a better time stamp to run this algorithm. Here, the results using config1 for MLP is better, because this set o parameters has a bigger training time.

TABLE I
MLP RESULTS FOR DIFFERENT TIME WINDOWS (CONFIG1)

	5min	4min	3min	2min	1min
Mean	86.27%	86.14%	82.46%	70.7%	8.21%
StdDev	5.25%	3.81%	3.75%	8.36%	4.75%
Median	86.27%	86.76%	82.11%	73.15%	7.07%

TABLE II
MLP RESULTS FOR DIFFERENT TIME WINDOWS (CONFIG2)

	5min	4min	3min	2min	1min
Mean	83.84%	85.39%	81.96%	68.45%	7.74%
StdDev	5.58%	3.67%	3.75%	9.62%	2.74%
Median	84.62%	85.29%	81.05%	70.47%	7.23%

For the Bayesian Network, we can see in Table IV and V that the 3 minutes database being better than the other datasets following the same logic we used to pick the 4 minutes database for the MLP algorithm, but the result is not as good as the MLP algorithm. Table VI shows that there is a significant difference between the 3 minutes and 4 minutes databases, indicating that 3 minutes is indeed the best pick for this case.

An important observation about both Table IV and Table V is that when we shrink the time frame to extract data

TABLE III
TWO TAILED T-TEST RESULT FOR THE MLP ALGORITHM, WITH A LEVEL OF SIGNIFICANCE $\alpha = 0,05$

	4min	5min
Run 1	84,690518	84,15912519
Run 2	86,15891133	86,30467572
Run 3	86,29938543	83,97435897
Run 4	86,31035996	83,99698341
Run 5	86,89201054	86,50452489
Run 6	85,85820896	85,88612368
Run 7	87,33318701	87,25113122
Run 8	86,4596137	84,7586727
Run 9	86,29719052	84,7586727
Run 10	85,11633011	85,31297134
P = 0,068503993		

TABLE IV
BAYESIAN NETWORK RESULTS FOR DIFFERENT TIME WINDOWS (CONFIG1)

	5min	4min	3min	2min	1min
Mean	72.50%	72.20%	75.55%	73.85%	65.60%
StdDev	4.64%	4.20%	4.25%	3.50%	2.70%
Median	72.80%	75%	75.53%	73.15%	65.60%

TABLE V
BAYESIAN NETWORK RESULTS FOR DIFFERENT TIME WINDOWS (CONFIG2)

	5min	4min	3min	2min	1min
Mean	75.5%	75.16%	75.31%	71.36%	65.58%
StdDev	4.61%	4.43%	4.25%	3.38%	2.7%
Median	72.81%	75%	75.53%	73.15%	65.6%

TABLE VI
TWO TAILED T-TEST RESULT FOR THE BAYESIAN NETWORK, WITH A LEVEL OF SIGNIFICANCE $\alpha = 0,05$

	3min	4min
Run 1	75,59014558	75,26558385
Run 2	74,74580067	74,52589991
Run 3	75,46696529	74,51931519
Run 4	75,67861142	75,55750658
Run 5	74,8331467	75,11413521
Run 6	74,73348264	75,84723442
Run 7	75,16573348	73,77085162
Run 8	75,78051512	74,67515364
Run 9	76,82866741	76,28841089
Run 10	74,3225084	76,13257243
p = 0,6761545659		

the standard deviation also decreases, indicating that with more data to infer statistic rules the Bayesian Network can be more accurate within itself. Another important information from these tables is that this algorithm is powerful enough to classify correctly our users with the 1 minute dataset, implying a certain advantage over the MLP algorithm when we see how much info the algorithms need to correctly classify users.

For this algorithm, the one using Tabu Search had a result a little bit better, even with a lot of common results between tables, indicating that hill climbing approaches, in this case, lead us to similar results.

Table VII shows us that, with exception of the 1 minute time frame, the longer the time window shrinks the better the algorithm works. This happens because with the increased number of samples of each class (users) more radial basis functions are needed to try to classify and learn patterns, increasing the accuracy. However, the processing of the RBF Network increases a lot with a higher number of instances, so, a smaller dataset with a reasonable result is more indicated for this classifier.

In Tables VII,VIII and IX, we can see how the database behave when the RBF Network algorithm is applied. We can treat the case where the algorithm executed until convergence as the general case, because in forensic investigation to discover whether some specific user is playing as another user the ideal would be to let the algorithm run until convergence. The mean and median results show us that 5 iterations of the algorithm give us the best result overall, needing the least amount of computational time to achieve them. Th is algorithm also behave better than the MLP when the samples are collected with the 1 minute time stamp, but it is worse when compared to the Bayes algorithm. Since the difference between the best and the second to best results are not close enough, a T-test is not needed to infer that 3 minutes in this cases the better time-stamp.

TABLE VII
RBF NETWORK RESULTS FOR DIFFERENT TIME WINDOWS (CONFIG1, UNTIL CONVERGENCE)

	5min	4min	3min	2min	1min
Mean	74.92%	76%	76.1%	70.36%	55.66%
StdDev	5.45%	5.36%	4.33%	6.08%	3.7%
Median	75	75%	75.55%	69.8	53.1

TABLE VIII
RBF NETWORK RESULTS FOR DIFFERENT TIME WINDOWS (CONFIG2, 5 ITERATIONS)

	5min	4min	3min	2min	1min
Mean	75.37%	75.19%	76.26%	70.29%	53.06%
StdDev	6.26%	5.45%	4.48%	3.61%	3.23
Median	75%	75%	76.71%	69.60%	52.73

TABLE IX
RBF NETWORK RESULTS FOR DIFFERENT TIME WINDOWS (CONFIG3, 10 ITERATIONS)

	5min	4min	3min	2min	1min
Mean	75.76%	75.7%	75.22%	68.56%	57.09%
StdDev	6.09%	5.72%	4.79%	4.72%	3.12%
Median	74.75%	75%	74.73%	68.68%	56.75%

V. FINAL REMARKS

The results of our experiments showed us how different neural networks behave with the League of Legends biometric data and databases. The classifier with the better accuracy was the Multilayer Perceptron, achieving its better result when the features take a little bit more time to be gathered. In the other hand, RBF and Bayesain networks indicate that its possible to achieve or even increase results when we collect the samples more often, sometimes under the cost processing.

Future work can go to a route where a separation between early, mid and late game samples are analyzed individually, because the current state of the games forces players to change roles, which can lead to a changes in their biometric profiles, or to a route where we investigate the impact of each set of features in the outcome of machine learnng algorithms.

REFERENCES

[1] I. da Silva Beserra, "Using keystroke dynamics for user identification in the online collaborative game League of Legends." 2017, master's Thesis (Systems and Computing), UFRN (Universidade Federal do Rio Grande do Norte), Natal, Brazil.

[2] L. Camara, "Acquisition and analysis of the first mouse dynamics biometrics database for user identification in the online collaborative game League of Legends." 2017, master's Thesis (Systems and Computing), UFRN (Universidade Federal do Rio Grande do Norte), Natal, Brazil.

[3] I. da Silva Beserra, L. Camara, and M. Da Costa-Abreu, "Using keystroke and mouse dynamics for user identification in the online collaborative game league of legends," 2016.

[4] F. Bergadano, D. Gunetti, and C. Picardi, "User authentication through keystroke dynamics," *ACM Transactions on Information and System Security (TISSEC)*, vol. 5, no. 4, pp. 367–397, 2002.

[5] S. P. Banerjee and D. L. Woodard, "Biometric authentication and identification using keystroke dynamics: A survey," *Journal of Pattern Recognition Research*, vol. 7, no. 1, pp. 116–139, 2012.

[6] P. Bours and C. J. Fullu, "A login system using mouse dynamics," in *Intelligent Information Hiding and Multimedia Signal Processing, 2009. IIH-MSP'09. Fifth International Conference on*. IEEE, 2009, pp. 1072–1077.

[7] S.-s. Hwang, H.-j. Lee, and S. Cho, "Account-sharing detection through keystroke dynamics analysis," *International Journal of Electronic Commerce*, vol. 14, no. 2, pp. 109–126, 2009.

[8] S. Z. S. Idrus, E. Cherrier, C. Rosenberger, and P. Bours, "Soft biometrics database: a benchmark for keystroke dynamics biometric systems," in *Biometrics Special Interest Group (BIOSIG), 2013 international conference of the*. IEEE, 2013, pp. 1–8.

[9] S. Z. S. Idrus, E. Cherrier, C. Rosenberger, S. Mondal, and P. Bours, "Keystroke dynamics performance enhancement with soft biometrics," in *Identity, Security and Behavior Analysis (ISBA), 2015 IEEE International Conference on*. IEEE, 2015, pp. 1–7.

[10] H.-R. Lv, Z.-L. Lin, W.-J. Yin, and J. Dong, "Emotion recognition based on pressure sensor keyboards," in *Multimedia and Expo, 2008 IEEE International Conference on*. IEEE, 2008, pp. 1089–1092.

[11] R. Thanganayagam and A. Thangadurai, "Fusion approach on keystroke dynamics to enhance the performance of password authentication," in *Electrical, Computer and Communication Technologies (ICECCT), 2015 IEEE International Conference on*. IEEE, 2015, pp. 1–6.

[12] C. Shen, Z. Cai, X. Guan, H. Sha, and J. Du, "Feature analysis of mouse dynamics in identity authentication and monitoring," in *Communications, 2009. ICC'09. IEEE International Conference on*. IEEE, 2009, pp. 1–5.

[13] C. Shen, Z. Cai, X. Guan, and J. Cai, "A hypo-optimum feature selection strategy for mouse dynamics in continuous identity authentication and monitoring," in *Information Theory and Information Security (ICITIS), 2010 IEEE International Conference on*. IEEE, 2010, pp. 349–353.

[14] C. Shen, Z. Cai, and X. Guan, "Continuous authentication for mouse dynamics: A pattern-growth approach," in *Dependable Systems and Networks (DSN), 2012 42nd Annual IEEE/IFIP International Conference on*. IEEE, 2012, pp. 1–12.

[15] K.-T. Chen and L.-W. Hong, "User identification based on game-play activity patterns," in *Proceedings of the 6th ACM SIGCOMM workshop on Network and system support for games*. ACM, 2007, pp. 7–12.

[16] R. V. Yampolskiy and V. Govindaraju, "Use of behavioral biometrics in intrusion detection and online gaming," in *Proc. of SPIE Vol*, vol. 6202, 2006, pp. 62 020U–1.

[17] I. Traore, I. Woungang, M. S. Obaidat, Y. Nakkabi, and I. Lai, "Combining mouse and keystroke dynamics biometrics for risk-based authentication in web environments," in *Digital Home (ICDH), 2012 Fourth International Conference on*. IEEE, 2012, pp. 138–145.

[18] S. Mondal and P. Bours, "Combining keystroke and mouse dynamics for continuous user authentication and identification," in *Identity, Security and Behavior Analysis (ISBA), 2016 IEEE International Conference on*. IEEE, 2016, pp. 1–8.

[19] WEKA. Waikato environment for knowledge analysis. [Online]. Available: https://www.cs.waikato.ac.nz/ml/weka

[20] F. Rosenblatt, "Principles of neurodynamics. perceptrons and the theory of brain mechanisms," CORNELL AERONAUTICAL LAB INC BUFFALO NY, Tech. Rep., 1961.

[21] WEKA. Class multilayerperceptron. [Online]. Available: http://weka.sourceforge.net/doc.dev/weka/classifiers/functions/MultilayerPerceptron.html

[22] N. Friedman, D. Geiger, and M. Goldszmidt, "Bayesian network classifiers," *Machine learning*, vol. 29, no. 2-3, pp. 131–163, 1997.

[23] WEKA. Class bayesnet. [Online]. Available: http://weka.sourceforge.net/doc.dev/weka/classifiers/bayes/BayesNet.html

[24] ——. Class k2 search. [Online]. Available: http://weka.sourceforge.net/doc.dev/weka/classifiers/bayes/net/search/local/K2.html

[25] *A Bayesian method for constructing Bayesian belief networks from databases*, 1990.

[26] G. Cooper and E. Herskovits, "A bayesian method for the induction of probabilistic networks from data," *Machine Learning*, vol. 9, no. 4, pp. 309–347, 1992.

[27] WEKA. Class tabu search. [Online]. Available: http://weka.sourceforge.net/doc.stable/weka/classifiers/bayes/net/search/global/TabuSearch.html

[28] R. Bouckaert, "Bayesian belief networks: from construction to inference," Ph.D. dissertation, Utrecht, Netherlands, 1995.

[29] WEKA. Class bayes simple estimator. [Online]. Available: http://weka.sourceforge.net/doc.stable/weka/classifiers/bayes/net/estimate/SimpleEstimator.html

[30] N. B. Karayiannis and G. W. Mi, "Growing radial basis neural networks: merging supervised and unsupervised learning with network growth techniques," *IEEE Transactions on Neural networks*, vol. 8, no. 6, pp. 1492–1506, 1997.

[31] WEKA. Class rbfnetwork. [Online]. Available: http://weka.sourceforge.net/doc.packages/RBFNetwork/

[32] Y. Wu, H. Wang, B. Zhang, and K.-L. Du, "Using radial basis function networks for function approximation and classification," *ISRN Applied Mathematics*, vol. 2012, 2012.