Jyväskylän yliopiston julkaisuarkisto
Jyväskylä University Digital Archive

UNIVERSITY OF JYVÄSKYLÄ

This is an electronic reprint of the original article.
This reprint *may differ* from the original in pagination and typographic detail.

Author(s): Ivannikova, Elena; David, Gil; Hämäläinen, Timo

Title: Anomaly detection approach to keystroke dynamics based user authentication

Year: 2017

Version:

Please cite the original version:

Ivannikova, E., David, G., & Hämäläinen, T. (2017). Anomaly detection approach to keystroke dynamics based user authentication. In ISCC 2017 : Proceedings of the 2017 IEEE Symposium on Computers and Communications (pp. 885-889). IEEE. Proceedings : IEEE Symposium on Computers and Communications. https://doi.org/10.1109/ISCC.2017.8024638

All material supplied via JYX is protected by copyright and other intellectual property rights, and duplication or sale of all or part of any of the repository collections is not permitted, except that material may be duplicated by you for your research use or educational purposes in electronic or print form. You must obtain permission for any other use. Electronic or print copies may not be offered, whether for sale or otherwise to anyone who is not an authorised user.

Anomaly Detection Approach to Keystroke Dynamics Based User Authentication

Elena Ivannikova, Gil David and Timo Hämäläinen
Department of Mathematical Information Technology
University of Jyväskylä
POBox 35 (Agora), 40014 Jyväskylä, Finland
Email: elena.v.ivannikova@student.jyu.fi, gil.david@jyu.fi, timo.hamalainen@jyu.fi

***Abstract*—Keystroke dynamics is one of the authentication mechanisms which uses natural typing pattern of a user for identification. In this work, we introduced Dependence Clustering based approach to user authentication using keystroke dynamics. In addition, we applied a k-NN-based approach that demonstrated strong results. Most of the existing approaches use only genuine users data for training and validation. We designed a cross validation procedure with artificially generated impostor samples that improves the learning process yet allows fair comparison to previous works. We evaluated the methods using the CMU keystroke dynamics benchmark dataset. Both proposed approaches outperformed the previous state-of-the-art results for the CMU dataset for unsupervised learning.**

I. Introduction

With the rapidly expanding internet services industry and, thus, the increasing importance of cyber security, the user identification and authentication problems have become a focus for many research labs. User's identity is normally verified by an access control mechanism, which performs the authentication task. Traditional approaches to access control provide good performance, however, they have some limitations. For example, passwords or PINs can be forgotten, lost or stolen which threatens security. Hence, new forms of authentication based on conjunction of traditional methods with biometrics are becoming more popular within computer security.

Biometrics are defined as the physical traits and behavioral characteristics that identify a living person [1]. Among them, keystroke dynamics [2] is considered as a strong behavioral biometric based authentication system. Keystroke dynamics statistics allow extracting timing features containing information about human's typing rhythms, i.e. time intervals between (a) the key presses of consecutive keys, (b) pressing a key and releasing next key, (c) pressing and releasing a key. Such typing patterns can serve as a human's identifier due to their ability to activate similar behavioral and cognitive mechanisms cf. handwritten signatures. Moreover, implementation cost of keystroke dynamics is very low as only a keyboard is needed for data collection. Among other advantages are the ability to operate in hidden mode, high user acceptance and ease of integration to existing security systems [3].

Despite all the advantages keystroke dynamics can be influenced by external factors, e.g. environmental conditions or keyboard device, or emotional state producing some noise and causing lower accuracy and permanence [4], [3]. With the help of statistical, machine learning or other algorithms researchers can identify behavioral patterns which allow distinguishing among the users based on specific characteristics. Analysis of typing patterns can be a powerful security tool for detecting intrusions or threats, or for distinguishing between genuine users and impostors during authentication [5], [6], [7], [8].

The rest of the paper is organized as follows. Section II shortly reviews the current state of keystroke dynamics techniques. Section III briefly describes the dataset used in this work and the data collection procedure. Section IV provides descriptions of the main concepts and methods used in the paper. Sections IV-A - IV-D explain the DC algorithm and the DC based anomaly detection method, while Section IV-E refers to k-NN based anomaly detection approach. Evaluation procedure is described in Section IV-F. Meanwhile, Section V is devoted to the experimental results. It describes parameter estimation procedure and results of the performance tests. Finally, Section VI concludes the paper.

II. Related Work

Most of the previous research works in keystroke dynamics refer to authentication systems. For user's authentication via keystroke dynamics either *static* or *free* text models can be used. The majority of previous works focus on *static* text when users type specific predefined text such as password [7], [2]. In more advanced and secure systems, users are being continually authenticated and monitored based on *free* text models when users type arbitrary input text of any length [9], [10], [2].

A vast amount of performance results obtained using various keystroke dynamics datasets have been reported by studies. Most of the studies collected own data, therefore, making performance comparison among the works difficult. To tackle this issue, in [11] authors present a comprehensive comparative study of detecting anomalies using keystroke dynamics dataset (CMU), thus, making a good benchmark. The authors collected data and implemented 14 anomaly detection algorithms for detecting anomalies in keystroke dynamics. Each detector was trained on a set of timing vectors of a true user, thus, providing a true-user behavioral model. Then the rest of data samples were tested against the true-user behavioral model and assigned an anomaly score. The parameters tuning was not performed in the experiments of this study due to a possible

bias in the evaluation results. Instead, the authors used the parameters reported in the source studies. In addition to [11] there are a number of works presenting performance results for the CMU keystroke dynamics benchmark dataset. According to [11] the top detectors demonstrating best results on the CMU dataset are the scaled Manhattan distance [12] and the nearest neighbor with Mahalanobis distance [13] with equal error rate (EER) values of 0.096 and 0.100, correspondingly. The best zero-miss false-alarm (ZMFAR) rate belongs to the nearest neighbor detector using Mahalanobis distance with the value of 0.482. Classical Mahalanobis [14] and the normed Mahalanobis [15] detectors demonstrate equal ZMFAR values of 0.482.

Following [11] a number of works proposed new algorithms and compared their performance with existing results. In [16], the authors followed the study in [11] by introducing new detectors and improving the results by using the same protocol and evaluation procedure in order to guarantee fair performance comparison. They introduced a new distance measure by combining Mahalanobis and Manhattan measures and used it in combination with the nearest neighbor classifier. They improved EER by 0.9% and ZMFAR by 4.5% compared to the best results in [11]. Furthermore, the Gaussian mixture model (GMM) was applied [17] producing EER of 0.087. Despite the authors also reported additional even better results we omit them here as the testing procedure was different from the one described in [11] making the fair comparison impossible. Another work [18] devoted to applications of neural networks to keystroke data reports improved results compared to the best performance values from [11]. The authors got EER of 0.0773 by using Levenberge-Marquardt backpropagation network. However, they incorporated negative examples into the training set. This prevents fair comparison with [11] where detectors were trained with the use of only positive samples. All aforementioned performance results can be found in Table I.

In this work, we propose two anomaly detection approaches based on *k*-nearest neighbors (*k*-NN) and dependence clustering (DC) [19]. Despite its simplicity, k-NN is a strong benchmark and often provides state-of-the art results in different tasks including biometric identification and authentification [20]. In our study, we employ a k-NN based approach combining with Manhattan distance. DC is a spectral clustering type algorithm which has been used for the clustering tasks. In this study, we adapt DC to solving anomaly detection problems. We test the methods on the CMU dataset and compare the obtained results with the performance reported in [11]. We reproduce training and evaluation procedures according to [11] to ensure fair comparison among detectors. Both proposed methods outperform the previous known state-of-the-art results on the CMU dataset.

III. Data

Detailed information of how the data was collected can be found in [11]. In this section, we provide a brief description of the data and the data collection procedure. For password generation and verifying its strength, publicly available tools were used [21], [22].

The password was generated as a sequence of 10 characters containing letters, numbers and punctuation signs. The obtained sequence was manually modified by altering some punctuation and casing in order to better meet requirements of a strong password. This resulted in password *.tie5Roanl* that still was rated strong. During data collection the same password was typed by all subjects.

The data was collected from 51 subjects of different age, sex and handedness groups. Each subject resulted in 400 password-typing samples. After all data had been collected, a set of timing features were extracted from raw data. Finally, 31 timing features were generated. Despite known correlations and linear dependency among timing features all of them were left in the data with a purpose of being useful in evaluation of future works. Possible adverse effect on some detectors can be avoided through a careful feature selection procedure [23].

IV. Methods

A. Preliminary concepts

Dependence Clustering (DC) is a method which considers geometric structures of data and is based on maximizing the group dependence measure. First, let us introduce essential assumptions regarding the data and concepts foundational this algorithm.

Given a set of data points $\Omega = \{x_i | i = 1, ..., N; x_i \in \mathbb{R}^n\}$ we form a graph defined by a similarity matrix S of size $N \times N$. By scaling rows of S so that elements in each row sum up to one we transform S to the transition matrix P and the corresponding t-step transition matrix P^t, thus, defining the Markov chain on this graph. The t-step transition matrix is calculated as $\mathrm{P}^t : P^t_{i,j} = Pr(X_t = j | X_0 = i)$, where X_0 represents probability at the initial state and X_t is a random walk representing a node at the t-th transition. Further, we assume that the whole chain is ergodic and that any two nodes in the graph can be connected through Markovian transitions. This enables calculation of statistical dependence between graph nodes in a certain evolution step $D_{i,j,t} = Dep(X_0 = i, X_t = j)$ [24] that is defined by the following equation:

$$D_{i,j,t} = \frac{Pr(X_0 = i, X_t = j)}{Pr(X_0 = i) Pr(X_t = j)}. \tag{1}$$

Statistical dependence serves as a measure of closeness between data points.

Then we define group dependence D_t as

$$D_t(\mathbf{s}) = \sum_{x_i, x_j \in \Omega} \big(D_{i,j,t} - d_0\big)(s_i s_j + 1)/2, \tag{2}$$

where $D_{i,j,t}$ is defined in (1), $\mathbf{s} = [s_1, ..., s_N]$ is a group assignment vector, where decision variable $s_i = 1$ if data point i belongs to group 1 and $s_i = -1$ if it belongs to group 2 and $d_0 = 1 + \epsilon_d$ is the baseline dependence level which is usually set to 1. More detailed information about parameter settings and optimization procedure can be found in [19].

B. DC for two groups

The actual optimization is carried out in the domain of real numbers $\mathbb{R}$. Hence, we relax the original formulation (2) so that elements of $\mathbf{s}$ become real. We start with a simple case of bisecting a graph. By varying the group assignment $\mathbf{s}$ of all N points and constraining the L_2 norm of $\mathbf{s}$ to be equal to one: $||\mathbf{s}||_2 = 1$ we obtain a good partition through maximizing the group dependence measure $D_t(\mathbf{s})$ as follows:

$$\begin{aligned}\arg\max_{||\mathbf{s}||=1} D_t(\mathbf{s}) &= \arg\max_{||\mathbf{s}||=1} \sum_{i,j}(D_{i,j,t} - 1)(s_i s_j + 1)/2 \\ &= \arg\max_{||\mathbf{s}||=1} \mathbf{s}^T(\mathrm{P}^t(\mathrm{B}^{(t)})^{-1} - \mathbf{1}\mathbf{1}^T)\mathbf{s} \qquad (3)\end{aligned}$$

where $\mathrm{B}^{(t)} : B^{(t)}_{j,j} = Pr(X_t = j) = [x_0^T \mathrm{P}^t]_j$, x_0 is the initial probability vector representing prior information related to the initial states, ϵ_d is assumed to be 0, for simplicity. The division of the data points is made based on the signs of the eigenvector corresponding to the largest positive eigenvalue of G_0 and stops when we get no positive eigenvalues. The matrix G_0 is defined by

$$\mathrm{G}_0 = \mathrm{P}^t(\mathrm{B}^{(t)})^{-1} - \mathbf{1}\mathbf{1}^T. \qquad (4)$$

Note that generally matrix G_0 is not symmetric. However, the nature of many datasets, including the CMU dataset used in this study, implies symmetry of similarity relation which obeys commutative property. Therefore, in this paper we make divisions based on eigendecomposition of a symmetric matrix $\mathrm{G} = \mathrm{G}_0 + \mathrm{G}_0^{\mathrm{T}}$.

C. DC for multiple groups

In order to obtain divisions to multiple groups a standard subsequent division approach is applied [25]. At every step we make binary splits of each group already found during the previous iterations, following the procedure described in Section IV-B. We proceed with an optimal division as the one which resulted in the maximal increase of group dependence for the whole dataset. Moreover, the following condition must fulfill: $D_t(\mathrm{sgn}(\mathbf{s'}_C)) - D_t(\mathbf{s}_C) > N\Delta_d$, where $\mathbf{s}_C$ and $\mathbf{s'}_C$ are within-cluster group configurations of a split candidate cluster $\Omega_C \subseteq \Omega$ before and after the division, correspondingly, $D_t(\mathbf{s})$ is defined by (2) and N is the size of Ω_C. $\Delta_d = \delta_d \sum_{g_{i,j}>0} \mathrm{G}$ is the minimal dependence gain required to split a cluster into two sub-clusters where $\delta_d \in [0, 1]$ is a dependence gain parameter. δ_d serves as a regularization parameter which prevents the algorithm from defining too small or too unclear clusters. Figure 1 displays pseudo code summarizing the above steps.

D. DC-based anomaly detection

For discovering anomalies we use the following procedure. First, we apply z-score normalization [26] to the data. Next, we build a similarity matrix by first computing pairwise distance matrix using either Manhattan or Euclidean distance. We transform the distance matrix to the similarity matrix by using the following non-linearity $s_{ij} = \exp(-\alpha d_{ij})$, where d_{ij} denotes an element of the pairwise distance matrix, s_{ij} denotes an element of the similarity matrix and α is a scale parameter. We anticipate that this non-linearity matches distribution of the pairwise distances well. In other words, it gives more resolution in the region of the distances we are interested in the most. Further, using the DC algorithm we divide the training set into clusters. The median Manhattan distance between a test sample and each cluster mean is then used as anomaly score for this test sample.

Require: Set of data points Ω, similarity matrix S, parameters t, δ_d, ϵ_d.
Initialize a set $\Psi = \{\Omega\}$, Δ_d as described in Section IV-C;
Compute transition matrix P from S (see Section IV-A);
while true do
  Set $\Omega'_C = \emptyset$, $\Omega' = \emptyset$;
  for all Ω_C in Ψ **do**
    Find a sub-division $\mathbf{s'}_C$ of Ω_C by solving (3);
    Define $\Omega^1_C = \{x_i | \mathrm{sgn}(\mathbf{s'}_C(i)) > 0\}$ and $\Omega^2_C = \{x_i | \mathrm{sgn}(\mathbf{s'}_C(i)) \leq 0\}$;
    if $D_t(\Omega^1_C) + D_t(\Omega^2_C) - D_t(\Omega_C) > N\Delta_d$ **then**
      $\Omega'_C = \Omega_C$, $\Omega' = \{\Omega^1_C, \Omega^2_C\}$;
    end if
  end for
  if $\Omega'_C \neq \emptyset$ **then**
    Apply the sub-division of Ω'_C by replacing Ω'_C in Ψ by the two elements of the set Ω';
  else
    Finish the loop.
  end if
end while

Fig. 1. Pseudo code for DC algorithm

E. k-NN-based anomaly detection

k-NN [14] is a well-known method used for classification or regression. It belongs to the category of instance-based learning methods when training phase is essentially missing. All computations are done locally with k nearest neighbors from the training set during testing phase.

For discovering anomalies we use the following procedure. First, we apply z-score normalization [26] to the data. Then for every test sample we compute Manhattan distance between itself and the mean value of its k-nearest neighbors. This distance is used as anomaly score for the test sample.

F. Evaluation

In our experiments we used the same procedure of building training and testing sets as described in [11]. Thus, for each user the first 200 samples were assigned to the training set and the last 200 samples were assigned to the test set as positive samples (true user's patterns). Moreover, the first 5 repetitions from every other user were added to the test set as negative samples (impostor's patterns). Note that training sets are constructed so that they contain only positive examples. To be able to perform cross-validation we need, however, negative samples as well. Using the true impostors' patterns during cross-validation would lead to unfair comparison with the results from [11] as our algorithms will indirectly learn to discriminate among positive and negative samples. Therefore, we

TABLE I

PERFORMANCE COMPARISON OF THE BEST REPORTED ALGORITHMS ON THE SAME CMU DATASET. MEAN AND (STANDARD DEVIATION) ARE SHOWN FOR THE EQUAL ERROR RATE (EER) AND ZERO-MISS FALSE-ALARM RATES (ZMFAR).

Algorithm	Parameters	EER	ZMFAR
Nearest Neighbor (Mahalanobis)		0.100 (0.064)	0.468 (0.272)
Manhattan (scaled)		0.096 (0.069)	0.601 (0.337)
GMM		0.087 (0.058)	-
Combined Mahalanobis and Mahattan distance		0.084 (0.056)	-
k-NN (Manhattan)	k=3	0.078 (0.055)	0.385 (0.267)
k-NN (Manhattan)	k=6	0.078 (0.056)	0.377 (0.272)
k-NN (Manhattan)	k=8	0.078 (0.057)	0.377 (0.276)
DC (Euclidean)	*	0.078 (0.056)	0.390 (0.280)
DC (Manhattan)	**	**0.077 (0.055)**	**0.358 (0.256)**

* $\delta_d = 0$, $\epsilon_d = 0.001$, $\alpha = 3.33$, $t = 1$, Manhattan distance; ** $\delta_d = 0$, $\epsilon_d = 0.01$, $\alpha = 10$, $t = 1$, Manhattan distance

designed a procedure to generate artificial negative samples. For each user we generated 200 additional negative samples by randomly sampling values for every feature independently from the entire data. Thus, for every user we obtained 200 new artificial samples served as impostors. Finally, we merged the obtained negative samples with the original training sets. We ran 10-fold cross validation procedure on the obtained validation sets of 400 samples independently for each user. Note that the randomly obtained artificial samples are not the points from the original dataset. Therefore, with this procedure we avoid using real test samples in the training/validation phase.

After the optimal parameter values had been set up the methods were red trained and tested using the same procedure as described in [11]. The obtained anomaly scores were converted into standard measures of error. The following two measures have been used for estimating performance of all detectors.

- **Equal error rate** (EER) [27] corresponds to a point on the ROC curve where miss rate and false-alarm rate are equal.
- **Zero-miss false-alarm rate** (ZMFAR) [13] is a measure minimizing false-alarm rate constraining the miss rate is zero.

V. RESULTS

The results corresponding to the best parameter values obtained during cross validation as well as the parameter values themselves are shown in Table I. For reference, Table I also displays the best results from the previous studies related to the CMU dataset, reported in Section II. One can see that the presented k-NN and DC approaches demonstrate improved results compared to the previous works. Thus, EER was improved by 7% compared to combined Mahalanobis and Manhattan distance detector [16]. We cannot compare our ZMFAR with the results from [16] as it was not reported in their work. Although, we improved ZMFAR by 11% compared to Nearest Neighbor (Mahalanobis) demonstrating the lowest ZMFAR in [11].

The best results have been shown by the DC method when using Manhattan distance in both stages of the algorithm: generating similarity matrix at the step of clustering and calculating distance at the step of discovering anomalies. We pay attention to the fact that not only the EER and ZMFAR performance measures have been improved but also their standard deviation values (displayed in brackets) became lower, which signifies superior robustness of the proposed methods.

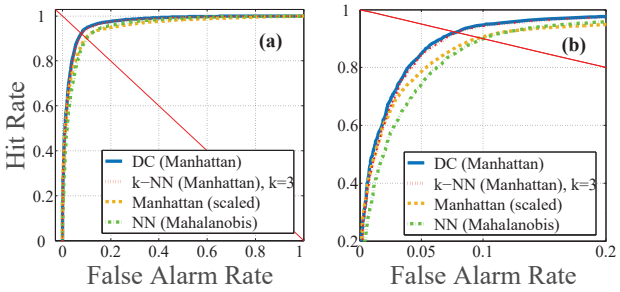

Fig. 2. (a) - ROC curves, (b) - zoomed in ROC curves for anomaly detection algorithms.

To visualize the results we plot ROC curves in Figure 2. Figure 2(b) is a zoomed version of Figure 2(a). ROC curves corresponding to the DC and k-NN based approaches proposed in this paper are displayed by plain and dotted lines, correspondingly. Furthermore, by dashed and dash-dot lines we plot ROC curves for Manhattan (Scaled) and Nearest Neighbor (Mahalanobis) detectors which demonstrated the best EER and ZMFAR in [11]. The threshold for calculating EER is chosen so that detector miss and false-alarm rates are equal. In Figure 2 it corresponds to the point where the line $y = 1 - x$ displayed in red crosses the detector ROC curve.

While computing ROC curves we used concatenated prediction results for all users, i.e. the varied parameter for building ROC curves represented global user-wide threshold (anomaly score) values. Note, this approach is different from the one used for calculation of EER and ZMFAR scores in Table I as the scores were taken as averages from per-user ROC curves. In the latter case the ROC curve parameter was varied for each user independently representing individual user threshold. The obtained ROC curve is a smoothed version of the individual

user ROC curves due to substantial variation in ZMFAR rate, in particular. Average EER and ZMFAR rates are higher than the ones measured from the user-wide ROC curve, since by varying anomaly score individually for every user we have more degrees of freedom, i.e. more powerful model. The ROC curves show that the methods generalize well even using global anomaly score threshold and can be further improved by choosing specialized per-user thresholds.

VI. Conclusions

In this work we proposed two approaches for detecting anomalies in the CMU dataset. One approach is based on the well-known k-NN and the other approach is based on DC thoroughly described here. Both proposed approaches outperform previous results that we know for this dataset respecting unsupervised learning setting. The main contributions of our work include:

(a) We improved upon the previous state-of-the-art results for the CMU dataset reported in [11], [16] for the unsupervised learning.

(b) We designed a cross validation procedure with artificially generated negative samples that allows avoiding the use of true negative samples in the learning process. Using the true negative samples during cross-validation would prevent fair comparison with the previous studies and result in violation of purely unsupervised learning setting. The improved results for the CMU dataset, indirectly, justify the validity of this procedure.

(c) We adapted a spectral clustering style algorithm previously used only for clustering problems to anomaly detection.

The practical implications of the presented results manifest in enabling more accurate and robust intrusion detection systems.

In the future the proposed approaches can be extended to other datasets related to more diverse security threats. Moreover, we plan to extend the DC approach to using dependence distance when computing anomaly score or determining the closest cluster of a test point. This extension requires changes in implementation of the DC algorithm that will allow performing real-time incremental computations with test points.

References

[1] J. Poss, D. Boye, and M. Mobley, "Biometric voice authentication," Jun. 10 2008, uS Patent 7,386,448. [Online]. Available: https://www.google.ch/patents/US7386448

[2] F. Monrose and A. D. Rubin, "Keystroke dynamics as a biometric for authentication," *Future Generation Computer Systems*, vol. 16, no. 4, pp. 351–359, 2000.

[3] P. S. Teh, A. B. Jin Teoh, and S. Yue, "A survey of keystroke dynamics biometrics," *The Scientific World Journal*, vol. 2013, p. 24 pages, 2013.

[4] C. Epp, M. Lippold, and R. L. Mandryk, "Identifying emotional states using keystroke dynamics," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI '11. New York, NY, USA: ACM, 2011, pp. 715–724.

[5] R. Joyce and G. Gupta, "Identity authentication based on keystroke latencies," *Commun. ACM*, vol. 33, no. 2, pp. 168–176, Feb. 1990.

[6] J. Ho and D. Kang, "Sequence alignment with dynamic divisor generation for keystroke dynamics based user authentication," *J. Sensors*, vol. 2015, pp. 935 986:1–935 986:14, 2015.

[7] F. Bergadano, D. Gunetti, and C. Picardi, "Identity verification through dynamic keystroke analysis," *Intell. Data Anal.*, vol. 7, no. 5, pp. 469–496, Oct. 2003.

[8] A. Messerman, T. Mustafic, S. A. Çamtepe, and S. Albayrak, "Continuous and non-intrusive identity verification in real-time environments based on free-text keystroke dynamics," in *2011 IEEE International Joint Conference on Biometrics, IJCB 2011, Washington, DC, USA, October 11-13, 2011*, 2011, pp. 1–8.

[9] D. Gunetti and C. Picardi, "Keystroke analysis of free text," *ACM Trans. Inf. Syst. Secur.*, vol. 8, no. 3, pp. 312–347, Aug. 2005.

[10] P. Kang and S. Cho, "Keystroke dynamics-based user authentication using long and free text strings from various input devices," *Information Sciences*, vol. 308, pp. 72 – 93, 2015.

[11] K. S. Killourhy and R. A. Maxion, "Comparing anomaly-detection algorithms for keystroke dynamics." in *DSN*. IEEE Computer Society, 2009, pp. 125–134.

[12] L. C. F. Araújo, L. H. R. Sucupira, M. G. Lizárraga, L. L. Ling, and J. B. T. Yabu-uti, *User Authentication through Typing Biometrics Features*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 694–700.

[13] S. Cho, C. Han, D. H. Han, and H. il Kim, "Web based keystroke dynamics identity verification using neural network," *Journal of Organizational Computing and Electronic Commerce*, vol. 10, pp. 295–307, 2000.

[14] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification (2Nd Edition)*. Wiley-Interscience, 2000.

[15] S. Bleha, C. Slivinsky, and B. Hussien, "Computer-access security systems using keystroke dynamics." *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 12, no. 12, pp. 1217–1222, 1990.

[16] Y. Zhong, Y. Deng, and A. K. Jain, "Keystroke dynamics for user authentication." in *CVPR Workshops*. IEEE Computer Society, 2012, pp. 117–123.

[17] Y. Deng and Y. Zhong, "Keystroke dynamics user authentication based on gaussian mixture model and deep belief nets." *ISRN Signal Processing*, vol. 2013, p. 7 pages, 2013.

[18] Y. Uzun and K. Bicakci, "A second look at the performance of neural networks for keystroke dynamics using a publicly available dataset," *Comput. Secur.*, vol. 31, no. 5, pp. 717–726, Jul. 2012.

[19] H. Park and K. Lee, "Dependence clustering, a method revealing community structure with group dependence," *Know.-Based Syst.*, vol. 60, pp. 58–72, Apr. 2014.

[20] J. Hu, D. Gingrich, and A. Sentosa, "A k-nearest neighbor approach for user authentication through biometric keystroke dynamics." in *ICC*. IEEE, 2008, pp. 1556–1560.

[21] P. Tools, "Security guide for windowsrandom password generator," http://www.pctools.com/guides/password/, 2008.

[22] Microsoft, "Password checker," Available at http://www.microsoft.com/protect/yourself/password/checker.mspx., 2008.

[23] E. Yu and S. Cho, "Ga-svm wrapper approach for feature subset selection in keystroke dynamics identity verification," in *Proceedings of the International Joint Conference on Neural Networks*. IEEE Press, 2003, pp. 2253–2257.

[24] K. Lee, A. Gray, and H. Kim, "Dependence maps, a dimensionality reduction with dependence distance for high-dimensional data," *Data Min. Knowl. Discov.*, vol. 26, no. 3, pp. 512–532, May 2013.

[25] M. E. J. Newman, "Modularity and community structure in networks," *Proceedings of the National Academy of Sciences*, vol. 103, no. 23, pp. 8577–8582, 2006.

[26] T. J. Hastie, R. J. Tibshirani, and J. H. Friedman, *The elements of statistical learning : data mining, inference, and prediction*, ser. Springer series in statistics. New York: Springer, 2009.

[27] P. Kang, S.-s. Hwang, and S. Cho, "Continual retraining of keystroke dynamics based authenticator," in *Proceedings of the 2007 International Conference on Advances in Biometrics*, ser. ICB'07. Berlin, Heidelberg: Springer-Verlag, 2007, pp. 1203–1211.