Benchmarking Keystroke Authentication Algorithms

Jiaju Huang, Daqing Hou, Stephanie Schuckers, Timothy Law, and Adam Sherwin
Electrical and Computer Engineering Department
Clarkson University, Potsdam NY USA 13699
jiajhua, dhou, sschucke, lawtp, sherwial@clarkson.edu

***Abstract*—Free-text keystroke dynamics is a behavioral biometric that has the strong potential to offer unobtrusive and continuous user authentication. Such behavioral biometrics are important as they may serve as an additional layer of protection over other one-stop authentication methods such as the user ID and passwords. Unfortunately, evaluation and comparison of keystroke dynamics algorithms are still lacking due to the absence of large, shared free-text datasets. In this research, we present a novel keystroke dynamics algorithm, based on kernel density estimation (KDE), and contrast it with two other state-of-the-art algorithms, namely Gunetti & Picardi's and Buffalo's SVM algorithms, using three published datasets, as well as our own new, unconstrained dataset that is an order of magnitude larger than the previous ones. We modify the algorithms when necessary such that they have comparable settings, including profile and test sample sizes. Both Gunetti & Picardi's and our own KDE algorithms have performed much better than Buffalo's SVM algorithm. Although much simpler, the newly developed KDE algorithm is shown to perform similarly as Gunetti & Picardi's algorithm on the three constrained datasets, but the best on our new unconstrained dataset. All three algorithms perform significantly better on the three prior datasets, which are constrained in one way or another, than our new dataset, which is truly unconstrained. This highlights the importance of our unconstrained dataset in representing the real-world scenarios for keystroke dynamics. Lastly, the new KDE algorithm degrades the least in performance on our new dataset.**

I. Introduction

The increasing adoption of automated information systems along with the pervasive use of mobile devices has dramatically simplified and empowered our lives, but has also made us overwhelmingly dependent on computers and digital networks. In the digital age, safe and secure user authentication is never more critical than now for preventing online crimes and maintaining a secure cyber space.

Keystroke dynamics, a kind of behavioral biometric [16], is emerging as an attractive tool for defending the cyber space [17]. For example, Coursera uses keystroke dynamics to verify online students [18]. Unlike other one-stop authentication, such as verifying the user ID and password, keystroke dynamics offers continuous authentication, where users are continuously authenticated whenever they are using the keyboard [19]. This additional layer of protection has the potential to significantly reduce the amount and the severity of identity thefts that are active on the Internet nowadays. It may also help detect and mitigate insider threats. Furthermore, unlike other more expensive biometric modalities, keystroke dynamics is almost free. The only hardware required is a keyboard, something that a user already possesses. Keystroke dynamics is also unobtrusive, so users can work unhindered as long as they pass authentication.

WIFS'2017, December, 4-7, 2017, Rennes, France. 978-1-5090-6769-5/17/$31.00

As shown in Table I and Table II, significant progress has already been made in free-text keystroke dynamics research. However, to advance the start of the art, several problems must be overcome. One is the lack of shared, truly unconstrained dataset that can be used to fully explore the wide scope of real-world keystroke dynamics. The few public datasets in Table I are all collected in settings that are constrained in one way or another. To this end, we contribute a large, truly unconstrained dataset that we will share with the community [11].

The second problem is the lack of benchmarking studies to test out the generalizability of algorithm performance across all available datasets. Common practice so far has been for each research group to publish a study using their own dataset. Replication studies are rare. As shown in Table II, a notable exception is Gunetti & Picardi [4], which has been extended by multiple groups [12] [5] [14] [20]. More benchmarking studies are necessary for further advancement in this field.

Keystroke dynamics authenticates a user by testing the similarity of a test sample against samples in a user's reference profile. As suggested by several prior studies [4] [5] [20], to achieve adequate authentication performance during free text, a profile of at least 10,000 keystrokes and a test sample of about 1,000 keystrokes is necessary. This requirement implies that a sufficiently large dataset is needed in order to thoroughly evaluate the performance of free text keystroke dynamics.

We have collected a novel dataset of individuals during their normal computing behavior [11]. For subjects who signed consent, keystrokes were recorded in an open computing lab and on their personal laptops while subjects performed their normal work (homework, email, etc.). Unlike previous datasets where there is some control of users during the free text collection, most often in a laboratory, this collection was obtained from unconstrained, natural behavior. This dataset contains over 12.9M keystrokes, large enough for us to benchmark keystroke dynamics algorithms, the subject of this paper.

In the remaining paper, along with two existing authentication algorithms, Gunetti & Picardi's algorithm [4] and Buffalo's SVM based algorithm [15], we propose a new algorithm that uses Kernel Density Estimation (KDE) as the basis for calculating the distance score between two samples

Study	#User	Time Span	Logger Setting	#Keystrokes	Availability
Monrose and Rubin [1]	31	7 weeks	UI, set tasks	N/A	NO
Monrose and Rubin [2]	63	11 months	UI, set tasks	N/A	NO
Dowland and Furnell [3]	35	3 months	Uncontrolled, logger	3.4 M	NO
Gunetti and Picardi [4] (Torino)	40/165	6 months	Web UI	400 K	YES
Messerman et al. [5]	55	12 months	Web UI (email)	293 K	NO
Ahmed and Traore [6]	53	5 months	Uncontrolled, logger	9.5 M	NO
Janakiraman and Sim [7]	22	2 weeks	Uncontrolled, logger	9.5 M	NO
Stewart et al. [8]	30	4 tests in 3 weeks	Web UI, short answers	720 K	NO
Vural et al. [9] (Clarkson I)	39	2 sessions	Web UI, tasks	840 K	YES
Sun et al. [10] (Buffalo)	148	28 days	Web UI, tasks	2.14 M	YES
Ours [11] (Clarkson II)	103	2.5 years	Uncontrolled	12.9 M	YES

Tab. I: Free-text keystroke datasets from the literature. There are 165 imposters in Gunetti and Picardi's dataset [4].

Study	Feature & Dataset Used	Algorithm	FAR (%)	FRR (%)	EER or accuracy (%)
Gunetti and Picardi [4]	n-graph flight time (n=2,3,4)	Sum of degree of disorder and n-graph reject ratio	0.5	5	
Davoudi and Kabir [12]	digraph, 21 / 165 [4]	Sum of degree of disorder and difference of histogram-based density estimation	0.14	1.59	
Xi et al. [13]	digraph, 21 / 165 [4]	Degree of disorder, cross correlation	1.65	2.75	
Shimshon et al. [14]	digraph, 21 / 165 [4]	Random forests	3.47	0	1.73
Messerman et al. [5]	n-graph	Degree of disorder	2.02	1.84	
Monrose and Rubin [1] [2]	digraph	Euclidean distance, (weighted) probability			91.1-93.2
Dowland and Furnell [3]	digraph	acceptance window, threshold on number of alerts	4.9	0	
Janakiraman and Sim [7]	key dwell time and digraph flight time	Bhattacharyya distance			74-100
Stewart et al. [8]	digraph	KNN			0.5
Ahmed and Traore [6]	key dwell time and digraph flight time	ratio of n-graph acceptance, Neural network for learning missing features	0.0152	4.82	2.5
Ceker et al. [15]	key dwell time and digraph flight time [9]	One-class SVM	0	0	0

Tab. II: Authentication algorithms and performance based on free-text keystroke dynamics.

(Section III). We benchmark the three algorithms using four datasets, the Torino dataset [4], the Clarkson I dataset [9], the Buffalo dataset [10], and our own new, unconstrained dataset (Clarkson II) [11] (Section II). We present the results in Section IV, and conclude our paper in Section V.

II. Datasets

Four datasets are used to benchmark our KDE based algorithm with the other two. In the following, we introduce these datasets in the chronological order by which they are created.

A. Torino Dataset

The Torino dataset is collected by Gunetti & Picardi [4]. This dataset is in Italian. Participants are asked to open a webpage in their browser to fill up HTML forms with whatever they feel comfortable to type. Each session is about 800 keystrokes, and a participant can type at most one session each day. The dataset contains 400K keystrokes in total, from 40 participants [4]. It also contains another 165 users who contribute only one sample, to be used as attack. This dataset contains only key press time, but without the key release time.

B. Clarkson I Dataset

The Clarkson I dataset is collected in a laboratory at Clarkson University [9]. This dataset contains two sessions for each participant that are separated by at least a few days. All the keystrokes are collected in the lab by using the same computer and keyboard, and running the same web browser. They are mostly free text keystrokes where participants transcribe given passages or answer survey questions that are carefully designed around the interests of the subject population to ensure fluent response. The dataset contains a total of 840K keystrokes, obtained from 39 participants.

C. Buffalo Dataset

The Buffalo dataset is collected by researchers at SUNY Buffalo [10]. This dataset is a mix of (long) fixed text and free text keystrokes, collected in a laboratory. There are 148 participants, with a total of 2.14M keystrokes in this dataset. The average user contributes 1.44K keystrokes, with a max of 18.3K. Most users have visited the lab in 3 sessions [10]. Four different kinds of keyboards are used. Some participants are deliberately required to use different kinds of keyboards

during different sessions. So this dataset can be used to test out any difference that the keyboards may make on authentication.

D. Our Own New, Unconstrained Dataset (Clarkson II)

Unlike the other three datasets described above, our new dataset, Clarkson II, is completely unconstrained [11]. A Windows based logger loaded on a user's own computer passively records all keystrokes from a user's natural behavior, without requiring or restricting them to do anything special. This dataset contains a total of 12.9M keystrokes, from 103 participants. The average user contributes 125K keystrokes, with a max of 625K. Our dataset is collected over a long span of 2.5 years, so it can also be used to study the long-term consistency of a user's typing behavior.

III. Algorithms

We compare our Kernel Density Estimation (KDE) based algorithm with two existing algorithms: Gunetti & Picardi algorithm [4] and Buffalo's SVM based algorithm [15]. We re-implement Gunetti & Picardi's algorithm [4] as the evaluation algorithm because it represents the state-of-the-art in free text keystroke dynamics. As shown in Tab. II, this algorithm has been the subject of multiple follow-up studies [12] [5] [14] [20]. We choose the SVM based algorithm because it represents a machine learning based approach [15].

Informed by prior studies [4] [5] [20], we use 10,000 keystrokes as the profile, and 1,000 keystrokes as a test sample. In all cases, we calculate a distance score, and compare it with an acceptance threshold to make accept/reject decisions. By systematically varying the thresholds, we are able to create DET (Detection Error Tradeoff) curves and calculate EER (Equal Error Rate) for each algorithm/dataset pair.

A. KDE Based Algorithm

In statistics, kernel density estimation (KDE) is a non-parametric way to estimate the probability density function (PDF) of a random variable. Kernel density estimation is a fundamental data smoothing problem where inferences about the population are made based on a finite data sample [21]. We use KDE to estimate the probability density of a digraph in the profile and in a test sample. The estimated PDFs are used to calculate the distance between reference profile samples and test samples. Specifically, in our implementation, we use a function in Python that is called *statsmodels.api.nonparametric.KDEUnivariate* to estimate PDFs.

Under ideal circumstances (the distribution is known and density is well behaved), a sample size of at least 4 data points is required in order to run KDE for one dimensional data [21]. This is in agreement with our experience. When we estimate the PDFs using KDE for only digraphs that have 4 or more instances in both the profile and test samples, we get the best performance results. When we also apply KDE to digraphs that have fewer than 4 instances, our performance results become worse. On the other hand, when we only use those digraphs that have enough instances, as features, fewer digraphs qualify. As a result, fewer features are used in authentication, resulting in worse performance. By using KDE for those digraphs that have enough instances, but using average density for those that do not, we get the best performance. The details of fusing these two are presented as follows.

For each digraph that has four or more instances in both the profile and the test samples, we estimate two PDFs using KDE. Note that the PDFs are functions of digraph flight time. For each value between 0 to 500 ms, we calculate the absolute difference of the two PDFs. We sum up all the differences for the digraph. Finally, we calculate an average PDF difference, D, for all such digraphs (those with four or more instances).

For each digraph that has four or more instances in the profile sample, but not so in the test sample, we use the PDF estimated for the profile sample to calculate the average probability density for all the instances in the test sample. Finally, we calculate an average probability density, d, for all such digraphs (those with fewer than four instances).

Our final distance score between two samples is a weighted sum of the average PDF difference, D, and the average probability density, d, namely, $D - 20 * d$, where the weight coefficient 20 is determined by trial and error.

Finally, in the extremely unlikely case where none of the digraphs shared by the profile sample and the test sample contains enough instances, the test sample will be rejected.

B. Gunetti & Picardi's Algorithm

Given a test sample X and a user A, Gunetti & Picardi's algorithm verifies whether X comes from A [4]. Specifically, it calculates the distance between samples in the profile of A and test sample X. If the distance is larger than the set threshold, it rejects the test sample as from an intruder. Otherwise, it accepts the test sample as from the genuine user A.

The distance between two typing samples S1 and S2 is measured in terms of shared n-graphs (n consecutive keystrokes) , using 'A' measure (for "*A*bsolute") and 'R' measure (for "*R*elative"). 'A' measure uses the difference in average n-graph latencies to determine distance. 'R' measure uses the normalized rank difference between two ordered vectors of average n-graph latencies to determine distance [4].

The 'A' measure between $S1$ and $S2$ is defined in terms of the n-graphs they share and a constant value t:

$A_{t,n}(S1, S2) = 1 - \#similar/\#shared$

where $\#similar$ represents the number of similar n-graphs shared by $S1$ and $S2$, and $\#shared$ the total number of n-graphs shared.

The constant t is needed for defining n-graph similarity. Specifically, let $G_{S1,d1}$ and $G_{S2,d2}$ be the same n-graph occurring in typing samples $S1$ and $S2$, with duration $d1$ and $d2$, respectively. We say that $G_{S1,d1}$ and $G_{S2,d2}$ are similar if $1 <= max(d1, d2)/min(d1, d2) <= t$, where t is a constant greater than 1 (we used $t = 1.25$).

C. Buffalo's SVM Based Algorithm

Buffalo's algorithm uses the most common D digraphs from a dataset as features. Three features are created for each digraph instance: the two dwell times for the two letters and

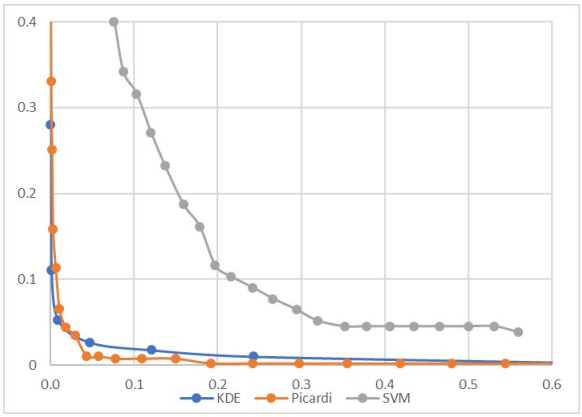

Fig. 1: DET curves for Torino dataset (X-axis: False Accept Rate, Y-axis: False Reject Rate). EER for the three algorithms, KDE, Gunetti & Picardi, and SVM, are 0.0348, 0.0311, and 0.1688, respectively.

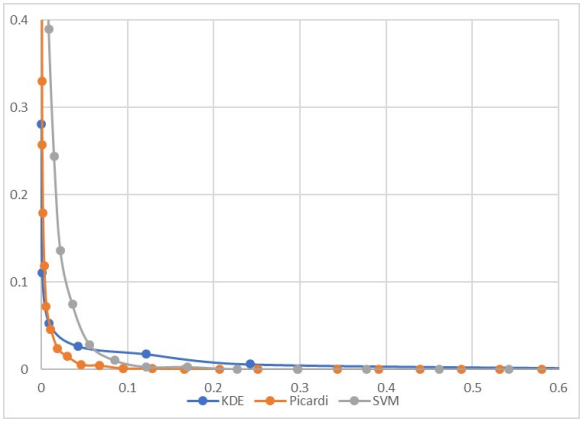

Fig. 2: DET curves for Clarkson Dataset I. EER for KDE, Gunetti & Picardi, and SVM are 0.0336, 0.0217, and 0.0377, respectively.

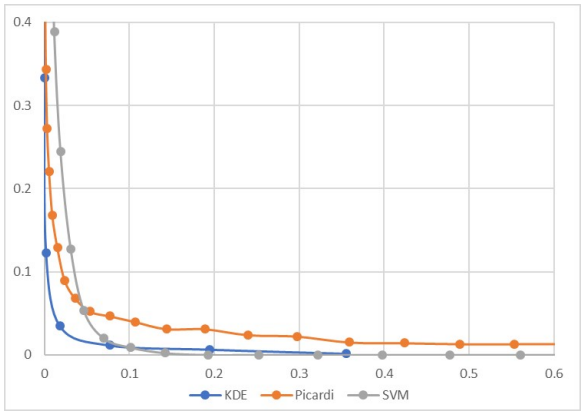

Fig. 3: DET curves for Buffalo dataset. EER for KDE, Gunetti & Picardi, and SVM are 0.0195, 0.0375, and 0.0493, respectively.

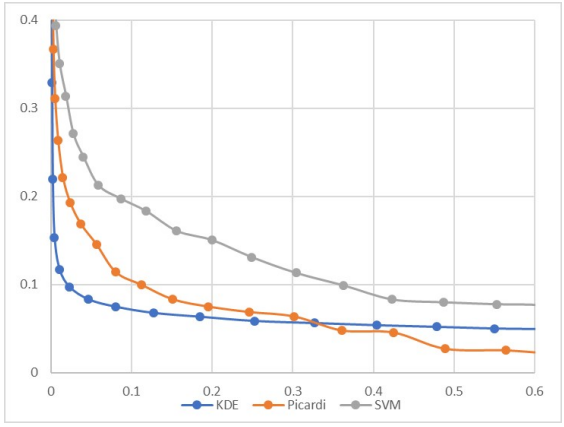

Fig. 4: DET curves for our unconstrained dataset (Clarkson II). EER for KDE, Gunetti & Picardi, and SVM are 0.0759, 0.1036, and 0.1567, respectively.

the flight time. A $D * 3$ dimension feature vector is created for each digraph instance, where 3 slots represent the features for the instance, and all the other slots are filled with 0s. They train a one-class SVM classifier using 80% of the data for each of the 34 users. Authentication decisions are made by picking the user whose classifier yields the highest total score for all the instances in the remaining 20% of the data [15]. Instead of an 80/20% split, in our benchmarking, all three algorithms have used the same profile size of 10,000 and test sample size of 1,000 keystrokes.

The more digraphs the SVM algorithm uses, the better its performance [15]. For the Clarkson I dataset, we use the same 27 most frequent digraphs used in the original paper [15]. We scan the other two datasets to identify the most frequent digraphs. The most frequent digraphs for the Buffalo dataset and our new dataset are almost identical to those found in Clarkson I, but a very different set is used for the Torino dataset as it is in a different language (Italian).

D. Evaluation Metrics

We measure the authentication performance for keystroke dynamics using three standard metrics:

- FAR (False Accept Rate): The ratio of impostor attacks that are falsely accepted as genuine users.
- FRR (False Reject Rate): The ratio of genuine tests that are falsely rejected as impostors.
- EER (Equal Error Rate): The point on a DET curve where FAR and FRR are equal.

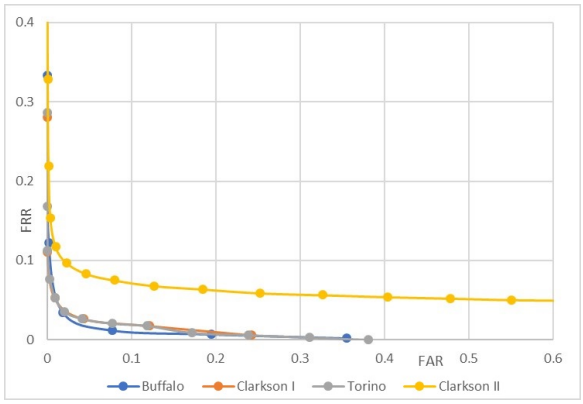

Fig. 5: DET curves for the KDE algorithm.

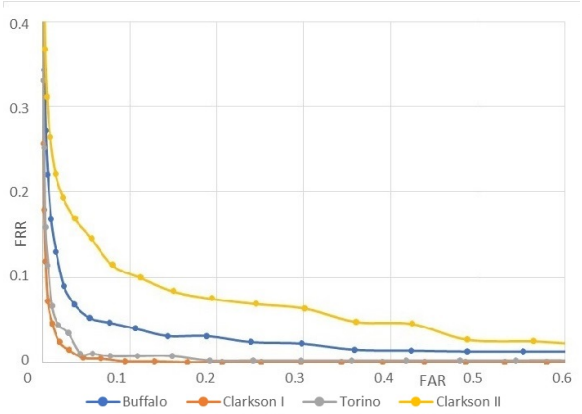

Fig. 6: DET curves for the Gunetti & Picardi algorithm.

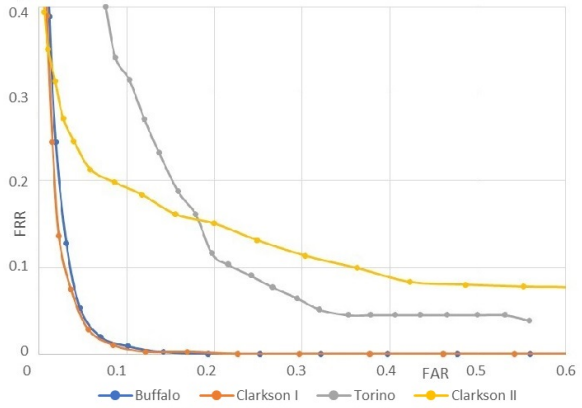

Fig. 7: DET curves for the SVM algorithm.

IV. Results of Evaluation

Three algorithms, our own KDE-based algorithm, Gunetti & Picardi [4], and the SVM based algorithm [15], are benchmarked using the four datasets described in Section II. All three algorithms are tested out using the same profile size of 10,000 keystrokes and a test sample size of 1,000 keystrokes.

A. DET Curves by Datasets

Figures 1, 2, 3, and 4 depict the Detection Error Tradeoff (DET) curves of running the three algorithms on the four datasets, Torino, Clarkson I, Buffalo, and our own new, unconstrained dataset (Clarkson II), respectively.

As shown in Figure 1, using the Torino dataset, our KDE algorithm performs slightly better than Gunetti & Picardi. Both are much better than SVM. The degradation on SVM's performance on this dataset is very likely due to the fact that the Torino dataset does not contain the key release time that is needed for calculating key dwell time. As a result, no dwell time is calculated for each key, and running SVM algorithm without the dwell time makes it perform much worse.

Dataset	KDE	Gunetti & Picardi [4]	SVM [15]
Torino [4]	0.0348	0.0311	0.1688
Clarkson I [9]	0.0336	0.0217	0.0377
Buffalo [10]	0.0195	0.0375	0.0493
Clarkson II [11]	0.0759	0.1036	0.1567

Tab. III: EERs for the three algorithms on the four datasets.

Figure 2 shows the DET curves of running the three algorithms on the Clarkson I dataset. All three algorithms perform better on this dataset than Torino. Moreover, Gunetti & Picardi does better than the other two. Note that the SVM algorithm has been reported to produce much better performance results on the same dataset [15] than our experiment. This is very likely due to the fact that different data sizes are used in the two experiments. They have used a much larger profile size and test sample size: 80% and 20% of each user's data [15]. In contrast, to enable a fair comparison, we have used a profile size of 10,000 keystrokes and a test sample size of 1,000.

Figure 3 shows the DET curves for the Buffalo dataset. On this dataset, out of the three, the KDE algorithm achieves the best performance (EER of 0.0195), and the SVM algorithm performs the worst (EER of 0.0493).

Figure 4 shows the DET curves for our new, unconstrained dataset (Clarkson II), where KDE performs the best and SVM the worst. Note that all three algorithms perform the worst on our new dataset than on the other three datasets as shown in Figures 1, 2, and 3. Moreover, the performance difference between the three algorithms on our new dataset is also quite large. We believe that this performance degradation and variation are due to the fact that our new dataset is unconstrained and thus contains much more 'noisy' keystrokes than the other three.

B. DET Curves by Algorithms

To facilitate comparison, Figures 5, 6, and 7 depict the DET curves of running the three algorithms on the four datasets, and Table III depicts the EERs.

Figure 5 shows the DET curves for running the KDE algorithm on the four datasets. As shown, the KDE algorithm has similar performance on the first three datasets, which are much cleaner than Clarkson II due to the ways they are collected. It is noteworthy that the KDE algorithm still performs reasonably well on Clarkson II, despite the fact that it is unconstrained and contains much more 'noisy' data.

Figure 6 shows DET curves for the Gunetti & Picardi algorithm [4]. This algorithm achieves similar performance on the Torino dataset and Clarkson I, but degrades significantly on the Buffalo dataset, and even worse on our Clarkson II.

Figure 7 shows DET curves for the SVM algorithm [15]. This algorithm performs similarly on Clarkson I and the Buffalo dataset, but significantly worse on Torino and Clarkson II. As we note earlier, the poor performance on the Torino dataset can be attributed to the fact that it does not have the key release time that SVM requires to calculate the key dwell time. Moreover, the poor performance on our Clarkson II indicates that it is sensitive to the 'noise' in the data.

C. Summary

As shown in Table III, both the KDE and Gunetti & Picardi's algorithms perform better than the SVM algorithm

consistently across the four datasets. The main reason for this difference is that the former two use many more features, often hundreds of n-graphs, than the two dozens or so features used in the SVM algorithm [15]. As a result, the dwell time is especially important for the SVM algorithm. When dwell times are missing in the Torino dataset, its performance degrades significantly (Figure 7).

As shown in Table III, the four datasets disagree on whether our KDE algorithm is better than Gunetti & Picardi's algorithm [4]. However, it is clear that all of the algorithms do not perform as well on unconstrained datasets as laboratory datasets. Our KDE algorithm responds to unconstrained, 'noisy' data in the most robust way, and performs the best.

V. Conclusion

Our new KDE-based algorithm has the best EER on the Buffalo dataset and our new, unconstrained dataset (Clarkson II), whereas Gunetti & Picardi's algorithm has better EER on the other two datasets. Buffalo's SVM has the worst performance on all four datasets.

The KDE algorithm shows similar performance on the Buffalo, Clarkson, and Torino datasets. Overall, the KDE creates the most consistent, stable performance. Not only does it has the most similar performance on other datasets, its performance on our unconstrained dataset (Clarkson II) degrades the least. The Gunetti & Picardi algorithm performs worse on the Buffalo dataset, but similar on the other two. The SVM algorithm has a similar performance on the Buffalo dataset and Clarkson I, but much worse on Torino because it does not have the key release time the algorithm requires.

We may conclude that because all three datasets are collected in similar settings, they have similar performance. The differences between English and Italian seem to have played little, if any role. On the other hand, our own new, unconstrained dataset (Clarkson II) is composed of typing during a participant's normal computing behavior and includes keystrokes from a variety of activities, as well as noisy keystrokes from activities such as gaming. It creates the worst performance for all three algorithms, but it is more likely to reflect the reality of real-world keystroke dynamics [22].

The number of features is critical for keystroke dynamics authentication. Buffalo's SVM paper [15] has shown that the more digraphs used, the better the performance. Yet the number of features that it uses is still far less than those used by the KDE and Gunetti & Picardi's algorithms. Therefore, much worse performance is found for this algorithm. In particular, since there is no key release time in the Torino dataset, only one of the three required features is used. As a result, we see a sharp decline in its authentication performance.

Acknowledgment

This work was supported by NSF Award CNS-1314792.

References

[1] F. Monrose and A. Rubin, "Authentication via keystroke dynamics," in *Proceedings of the 4th ACM conference on Computer and communications security*. ACM, 1997, pp. 48–56.

[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. Dowland and S. M. Furnell, "A long-term trial of keystroke profiling using digraph, trigraph and keyword latencies," in *Security and Protection in Information Processing Systems: IFIP 18th World Computer Congress TC11 19th International Information Security Conference, Toulouse, France*, Y. Deswarte, F. Cuppens, S. Jajodia, and L. Wang, Eds. Springer US, 2004, pp. 275–289.

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

[5] A. Messerman, T. Mustafić, S. A. Camtepe, and S. Albayrak, "Continuous and non-intrusive identity verification in real-time environments based on free-text keystroke dynamics," in *Biometrics (IJCB), 2011 International Joint Conference on*. IEEE, 2011, pp. 1–8.

[6] A. A. Ahmed and I. Traore, "Biometric recognition based on free-text keystroke dynamics," *IEEE Transactions on Cybernetics*, vol. 44, no. 4, pp. 458–472, April 2014.

[7] R. Janakiraman and T. Sim, "Keystroke dynamics in a general setting," *Advances in Biometrics*, pp. 584–593, 2007.

[8] J. C. Stewart, J. V. Monaco, S.-H. Cha, and C. C. Tappert, "An investigation of keystroke and stylometry traits for authenticating online test takers," in *Biometrics (IJCB), 2011 International Joint Conference on*. IEEE, 2011, pp. 1–7.

[9] E. Vural, J. Huang, D. Hou, and S. Schuckers, "Shared research dataset to support development of keystroke authentication," in *International Joint Conference on Biometrics (IJCB), 2014 IEEE International Conference on*, 2014.

[10] Y. Sun, H. Ceker, and S. Upadhyaya, "Shared keystroke dataset for continuous authentication," in *2016 IEEE International Workshop on Information Forensics and Security (WIFS)*, Dec 2016, pp. 1–6.

[11] C. Murphy, J. Huang, D. Hou, and S. Schuckers, "Shared dataset on natural human-computer interaction to support continuous authentication research," in *Biometrics (IJCB), 2017 International Joint Conference on*. IEEE, 2017, pp. 1–6.

[12] H. Davoudi and E. Kabir, "A new distance measure for free text keystroke authentication," in *Computer Conference, 2009. CSICC 2009. 14th International CSI*. IEEE, 2009, pp. 570–575.

[13] K. Xi, Y. Tang, and J. Hu, "Correlation keystroke verification scheme for user access control in cloud computing environment," *The Computer Journal*, p. bxr064, 2011.

[14] T. Shimshon, R. Moskovitch, L. Rokach, and Y. Elovici, "Continuous verification using keystroke dynamics," in *Computational Intelligence and Security (CIS), 2010 International Conference on*. IEEE, 2010, pp. 411–415.

[15] H. Çeker and S. Upadhyaya, "User authentication with keystroke dynamics in long-text data," in *Biometrics Theory, Applications and Systems (BTAS), 2016 IEEE 8th International Conference on*, 2016, pp. 1–6.

[16] R. V. Yampolskiy and V. Govindaraju, "Behavioural biometrics: a survey and classification," *Int. J. Biometrics*, vol. 1, no. 1, pp. 81–113, Jun. 2008. [Online]. Available: http://dx.doi.org/10.1504/IJBM.2008.018665

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

[18] A. Maas, C. Heather, C. T. Do, R. Brandman, D. Koller, and A. Ng, "Offering verified credentials in massive open online courses: Moocs and technology to advance learning and learning research (ubiquity symposium)," *Ubiquity*, vol. 2014, no. May, p. 2, 2014.

[19] P. Bours, "Continuous keystroke dynamics: A different perspective towards biometric evaluation," *Information Security Technical Report*, vol. 17, no. 12, pp. 36 – 43, 2012, human Factors and Bio-metrics.

[20] J. Huang, D. Hou, S. Schuckers, and Z. Hou, "Effect of data size on performance of free-text keystroke authentication," in *Identity, Security and Behavior Analysis (ISBA), 2015 IEEE International Conference on*, March 2015, pp. 1–7.

[21] E. Parzen, "On estimation of a probability density function and mode," pp. 1065–1076, 1962.

[22] J. Huang, D. Hou, S. Schuckers, and S. J. Upadhyaya, "Effects of text filtering on authentication performance of keystroke biometrics," in *2016 IEEE International Workshop on Information Forensics and Security (WIFS)*, Dec 2016, pp. 1–6.