2018 IEEE Symposium on Security and Privacy Workshops

Mouse Authentication without the Temporal Aspect – What does a 2D-CNN learn?

Penny Chong*‡, Yi Xiang Marcus Tan*‡, Juan Guarnizo*‡, Yuval Elovici*† and Alexander Binder*‡

*ST Electronics-SUTD Cyber Security Laboratory
‡Information Systems Technology and Design (ISTD) Pillar
Singapore University of Technology and Design, 8 Somapah Road, 487372, Singapore
†Department of Information Systems Engineering, Ben-Gurion University of the Negev, Beer-Sheva, 84105 Israel
†Deutsche Telekom Innovation Laboratories at Ben Gurion University of the Negev, Beer-Sheva, 84105 Israel

***Abstract*—Mouse dynamics as behavioral biometrics are under investigation for their effectiveness in computer security systems. Previous state-of-the-art methods relied on heuristic feature engineering for the extraction of features. Our work addresses this issue by learning the features with a convolutional neural network (CNN), thereby eliminating the need for manual feature design. Contrary to time-series-based modeling approaches, we propose to use a two-dimensional CNN with images as inputs. While counterintuitive at first sight, it permits to profit from well-initialized lower-layer kernels obtained via transfer learning. We demonstrate our results on two public datasets, Balabit and TWOS, and compare against a 1D-CNN and a classical baseline relying on hand-crafted features, which are both outperformed. We show that a position-independent variant of the 2D-CNN loses little performance yet we learned that the trained classifier is very sensitive to simulated resolution shifts at test time. In a final step, we analyze and visualize the learned features on single test curves using layer-wise relevance propagation (LRP). This analysis reveals that the 2D-CNN uses curve information only sparsely, with a tendency to assign little relevance to straight segments and artifactual curve crossings.**

***Keywords*-authentication; CNN; LRP; mouse dynamics;**

I. Introduction

Behavioral biometrics are gaining attention for potential use in authentication systems. The predominant approach in this field relies on physiological biometrics, such as fingerprints, iris and facial recognition. They are reliable yet costly as they require additional, sometimes costly, hardware to deploy. On the other hand, behavioral biometric systems are capable of capturing both motor and physiological differences among users, and permit a continuous mode of authentication. In the context of user authentication for computer systems, behavioral biometrics such as keystrokes and mouse dynamics can be employed to authenticate users. These approaches are cost effective as they do not require additional hardware to be deployed, which makes behavioral biometrics attractive. In this work we focus on mouse-based authentication.

To the best of our knowledge, the majority of state-of-the-art methods extract a set of fixed mouse features such as angle, curvature and velocity in a heuristic manner and feed them into a shallow machine learning model, such as random forests or support vector machines (SVM). Given the immense success of deep models in many fields, we investigate the usability of neural networks with their intrinsic feature learning for this problem. Deep learning requires two components for success, a suitable classifier model and sufficiently large sample size. Regarding the model, it is natural to exploit the time series structure of mouse movement data, for example by using long short-term memory (LSTM) [1] over a suitable representation, or one-dimensional (1D-) CNN [2] as they are common in natural language processing. However due to the small sample sizes in typical mouse biometric datasets, in the order of only a few thousand samples, learning deep models with random initializations is a challenging task. It is well known, that successful deep learning models perform abysmally when trained from scratch with too small sample sizes [3]. For this reason we investigate the usage of those deep learning models, for which we can use transfer learning to initialize the convolutional layers with pretrained kernels. This paves the way for a two-dimensional convolutional deep learning model for user authentication, with images of mouse movement trajectories and weights initialized from transfer learning, to overcome problems of small sample sizes. We introduce a model that learns jointly the multi-label classifiers for each user and show its results on two datasets. We investigate sensitivity to position and resolution-invariance to obtain insights into strengths and fail cases of the model. As such a model is counterintuitive compared to 1D-CNNs or time-series models, we proceed to analyze the features using layer-wise relevance propagation [4]. Key contributions of this work include:

1) A CNN model with joint multi-label training for mouse-based user authentication in comparison with a fixed-feature extraction and a 1D-CNN baseline.
2) Exploration of various preprocessing methods for conversion of mouse movement sequences into suitable inputs for 2D-CNNs.
3) Exploration on the robustness of the approach against changes in the environment.
4) Visualization of the mouse movement parts used in the

DOI 10.1109/SPW.2018.00011

decision of the 2D-CNN for single test instances.

II. Related Works

In an early work by [5], a user identification system via signature written with mouse was introduced. Gradually the biometric technology evolved to non-signature based mouse dynamics as introduced by [6]. Mouse features were extracted heuristically in the behavior analysis unit and later fed into the behavior comparison unit consisting of a 3-layer artificial neural network (ANN) to discriminate between users. Subsequent work [7] adopted the mouse dynamics biometrics for user identity verification. Each user is verified based on a single action instead of an aggregate of actions. They constructed a hierarchy of mouse features. A random forest classifier were trained with these features. In [8]–[11], the authors used a SVM classifier over various feature types. The work [8] introduced procedural mouse features. These procedural features together with holistic features were fed into a one-class SVM. In [9], the authors evaluated BayesNets, SVMs and decision trees as the classification algorithms. Recently, a multi classifier fusion (MCF) architecture consisting of ANN, a counter-propagation artificial neural network (CPANN) and SVM were proposed in [12] for both keystroke and mouse dynamics. The work [13] took a different approach by using mouse gestures for static authentication. A gesture is a combination of mouse movements and clicks in a way that is recognizable as a command to the system. From the mouse gestures, the authors extracted features and employed a learning vector quantization (LVQ) neural network for classification. Further work on mouse dynamics can be found in [14]–[17].

III. Methodology

A. Baselines

The first baseline is the model from [10], which uses fixed feature extraction pipeline of 66 features from [7] without smoothing and a support vector machine classifier.

The second baseline is a 1D-CNN, implemented in PyTorch, which can be obtained from the authors. One input sample for the neural network is a sequence of length 130. The sequence element for one time step is the 2-dimensional, time-normalized vector $\left(\frac{dx}{dt}, \frac{dy}{dt}\right)$ of position differences. Thus one input sequence consists of a $(130, 2)$ vector. The convolution dimension is the time axis. The network with exponential linear activations (elu) after the convolutions is given in Figure 1. It aggregates differences in a first layer using two different scales, followed by a second layer with shared weights. By its input this model is invariant to translation of mouse movement sequences.

B. 2D-CNNs and Transfer Learning

Given the representation of mouse movements as time series, a natural choice for a deep learning architecture relies on one-dimensional convolutions, possibly in combination with recurrent nets (RNNs) in higher layers. As known mouse movement datasets have usually sample sizes in the order of tens of thousands samples, training of deep 1D-CNN models in this domain can lead to overfitting. Alternatives consist of learning shallower models or the use of transfer learning. While pretrained 1D-CNNs are abundant in natural language processing, it is unclear how such models over high-dimensional word embeddings as inputs can be transferred to the domain of two-dimensional mouse movements. As for RNNs, a preliminary experiment showed poor performance when trained directly on the time series. Unsurprisingly, a RNN needs to be fed with features aggregated on a larger time scale. Our choice of a 2D-CNN was motivated by the fact that transfer learning achieves good performance while training from random initializations performs poorly [3], and the experience that models have over images are transferable even when image statistics are highly different. We used a Googlenet architecture [18] from the Caffe package [19]. There are two approaches to transfer learning that address which layer(s) to be modified during training: either only the last layer or all layers. Both of these approaches were explored in this work and we found that fine-tuning all the layers of the network performs better.

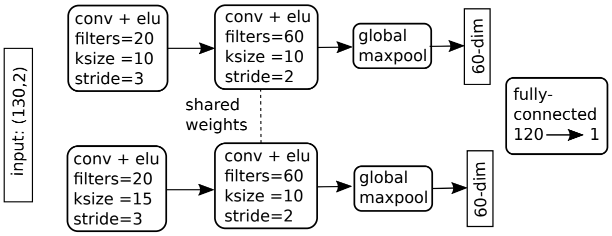

Figure 1: 1D-CNN baseline.

C. Multi-label Joint Training

Unlike the conventional approach of training n different binary classifiers for n users, we employed joint multi-label training to train a single multi-label classifier to predict n labels. Given a single instance, a multi-label classifier predicts a set of target labels where each label represents a different classification problem. On the other hand, a multi-class classifier predicts a single label or class in which the labels or classes are mutually exclusive. The chosen approach of training a single multi-label classifier has two notable advantages. Firstly, learning a common set of features reduces issues with overfitting given the small sample sizes available. Secondly, it greatly reduces the constant in the training time and space complexity for n users. We observed the first point in preliminary experiments which are omitted here. The joint multi-label approach performed better. The choice of adopting the multi-label joint training is advantageous, since n different binary classifiers can be obtained simultaneously by training one single model with features, that are shared except for the last layer. That layer with $2000n$ weights contains only a fraction of the weights of all convolutional layers (5.8 million parameters [18]), resulting in a reduction of the constant by a factor of 20 for 20 users. During the training, the model parameters were updated through a sum of n weighted cross-entropy loss functions, one for each user. Weighting the cross-entropy loss gives more emphasis on the losses of the smaller classes by setting the class weight to be inversely proportional to the frequency of the class in the training data. The original multi-class Googlenet architecture was modified to a multi-label architecture by setting the number of outputs in the last fully connected (FC) layer to $2n$, corresponding to n multi-label problems, each with its own two-class output. Then, the FC layer was splitted and branched to n separate softmax layers, one for each multi-label problem.

Besides employing the multi-label joint training, other means of training were also explored with the aim to tackle the imbalanced class problem. This includes setting a fixed positive to negative samples ratio in stochastic gradient descent (SGD) mini-batches when training n different single-label binary classifiers separately. Nevertheless, this training approach underperformed compared to the multi-label joint training.

D. Data Preprocessing

The 2D-CNN approach requires to generate images of mouse movement sequences from raw mouse coordinates. We have explored 3 different ways to generate the mouse movement plots. Firstly, using a moving window stride, secondly, a fixed time window split and thirdly, a time difference split. The fixed time window and time difference split methods depend greatly on the location of the coordinates with respect to the entire screen resolution, while the moving window stride method being similar to the 1D-CNN baseline, is position-invariant and focuses solely on a small region of movement contained in the moving window.

1) Moving Window Stride: In this approach, plots of mouse movement sequences were extracted from a fixed window size of 448×448 by sliding the window along connected coordinates of mouse movements. As mentioned, this approach will only extract a small region of the curve centered on the fixed window, eliminating the location of the coordinates relative to the screen resolution. We considered all consecutive mouse coordinates contained within the fixed window to be plotted as a mouse sequence and set the first coordinate that falls outside the window to be the midpoint of the next new moving window. Therefore, the generated mouse sequence plots will be overlapping.

2) Fixed Time Window Split: As an alternative to the fixed size moving window stride approach, the total time elapsed from the start to the end of a mouse sequence was used as a splitting criterion to group consecutive mouse coordinates into sequences. Instead of fixing a constant window size, we fixed a constant time window of 10 seconds. Thus, all coordinates that fall within this time window are considered as one sequence. The next new sequence will have a 5% overlap with the previous sequence provided that the time elapsed from the start to the end of this new sequence does not exceed the time window of 10 seconds. In the case where the inclusion of overlapping coordinates caused the new sequence to exceed the fixed time window, this new sequence will not overlap with the previous sequence. Doing so, one can account for long period of inactivity and reduce the risk of splitting a supposedly continuous sequence or combining two supposedly disjoint sequences. The curves were plotted with respect to the screen resolutions. Later on, the plots were resized to a fixed size of 448×448 before passing it to the CNN. The details of this resizing together with its impact on the prediction performance will be discussed in section IV-E.

3) Time Difference Split - Unfused and Fused Curves: Apart from the two preprocessing methods, the mouse coordinates were also splitted by considering the time difference between two consecutive coordinates. A coordinate belongs to a new mouse movement sequence if the time difference between this coordinate and the preceding coordinate exceeds the one second threshold. One has no control over the length of the sequences which may lead to the generation of extremely short sequences. We called these generated plots *unfused* curves. Too short sequences may not be discriminative. In order to ensure the generated sequences are long enough and contain discriminative movements among users, short curves will be fused together to generate a longer curve, called the *fused* curve. These *fused* curves are ensured to meet a minimum length of either 33%, 50% or 100% of the screen width. Similarly, the plots were plotted with respect to the screen resolutions and resized to 448×448.

IV. Experiments

A. Datasets

The Balabit Mouse Dynamics Challenge [20] data and the TWOS [21] data were used in our experiments. The Balabit data consists of a train and test set, containing information on timing and positions without screen resolutions. Thus, the screen resolution for each user was estimated by computing the maximum coordinate and mapping it to a set of finite screen resolutions. Here we assumed that each user used only one screen resolution and we chose the closest possible mapping. We constructed a 5-fold cross-validation set for training and validation. The experiments were reported on the test set, after removing the illegal sessions. Illegal sessions are sessions conducted by users that are not owner of the account but tries to masquerade as the actual owner.

On the other hand for the TWOS data, only 20 out of 24 users were used in training and testing. The 4 random users omitted here are to be used in a future test case to test the trained classifier with samples from unseen users.

The results reported on TWOS data are the average from a 5-fold cross-validation set.

All experiments if not mentioned explicitly, were conducted on the standard Balabit data and tested on the Balabit public test set without illegal users.

B. Model Hyperparameters

For training of the 2D-CNN, we used a base learning rate of 0.0001, the SGD optimizer with a batch size 20, gamma value of 0.96, momentum of 0.9, weight decay rate of 0.0002 and 100k as the maximum number of iterations. The weights and biases from all layers of CNN were fine-tuned, emphasizing on the last FC layer. The learning rate for the last FC layer was set to be larger than the learning rates in the preceding layers. For training of the 1D-CNN, Adam with a learning rate of 0.01, decaying every 12 epochs by a gamma of 0.2, weight decay of $5e-4$, and 45 epochs with batch size 8 were used.

C. Performance Metrics

In the testing phase, we reported the performance of our classifiers in terms of area under the curve (AUC) of a receiver operating characteristic (ROC) curve and the equal error rate (EER), which is obtained by using a threshold such that both false positive rate (FPR) and false negative rate (FNR) are equal.

D. Experimental Results

The first result concerns the comparison of 2D-CNN against two baselines: a combination of fixed features with a SVM and a 1D-CNN trained from scratch. One can see from Table I that the proposed model outperforms the baselines by a clear margin. While it does not rule out the possibility to obtain better 1D-CNN models, it does show the strength of the proposed combination of 2D-CNN, transfer learning and joint multi-label training.

Table I: Comparison of baselines against proposed 2D-CNN.

Dataset	Model	Avg AUC	Avg EER
Balabit	Fixed features+SVM [10]	0.87	0.20
	1D-CNN	0.90	0.11
	Proposed 2D-CNN	**0.96**	**0.10**
TWOS	Fixed features+SVM [10]	0.88	0.18
	1D-CNN	0.77	0.23
	Proposed 2D-CNN	**0.93**	**0.13**

The second result addresses the concern of which preprocessing method works best for generation of images. It is observed in Table II that fused curves of longer lengths perform better when compared to the fixed time window of 10 seconds and moving window stride. This is in line with the intuition that longer curves tend to contain more discriminative information. A similar observation was made for the 1D-CNN baseline regarding larger and smaller kernel sizes. Among these approaches, the model trained with plots extracted using the moving window stride approach performed the worst while the model trained with fused curve of minimum length 1.00 performed the best, slightly outperforming the fixed time window approach. The average AUC and EER values reported for moving window stride were far worse than the other two approaches. This gap in performance may be due to the nature of the moving window approach that removed positioning information relative to the screen resolution, while the other two approaches retained this information. This information may be crucial in distinguishing mouse movements between users, if every user has a preference to work only in certain regions of the screen. To verify this, heatmaps representing the frequency a pixel is visited were plotted for every user.

Table II: Comparison of various preprocessing methods.

Preprocessing Method	Avg AUC	Avg EER
Moving window stride	0.7729	0.2891
Fixed time window of 10s	0.9506	0.1154
Fused curve *(min length 0.33)*	0.9508	0.1126
Fused curve *(min length 0.50)*	0.9546	0.1062
Fused curve *(min length 1.00)*	**0.9584**	**0.0984**

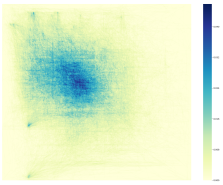

Figure 2: Frequency heatmap from Balabit user 12.

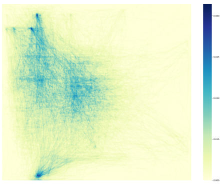

Figure 3: Frequency heatmap from Balabit user 20.

Comparing the frequency heatmaps from Fig. 2 and Fig. 3, users preferences to work in different regions of the screen are clearly visible. User 12 preferred to work with its mouse close to the center of the screen. On the contrary, user 20 used its mouse over a larger region with preference to work in the left region of the screen, concentrating mostly on the top-left or the bottom-left of the screen. Through these heatmaps, one can assume that classifiers based on absolute coordinates rather than difference, may include such position information into their decisions. We leave it to the reader to decide whether the inclusion of position information is considered a feature or a bias in a mouse movement model.

In addition to using only one curvelet in the evaluation, the models were also evaluated by averaging the prediction scores of 5, 10 and 15 consecutive curvelets. The idea behind this evaluation is to determine the performance of the classifier in biometric systems, when given a longer sequence of mouse movements for identification. Alterna-

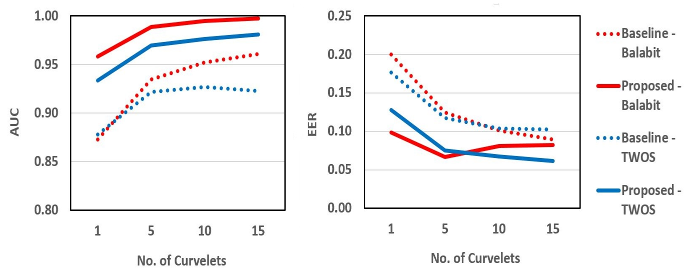

Figure 4: Average area under the curve (AUC) and equal error rate (EER).

tively, one can think of this longer sequence of movements as a longer period of time allocated for user to authenticate himself/herself. Based on Fig. 4, the average AUC and EER scores improved as the number of curvelets increased further. We remark here that the proposed 2D-CNN still remained superior to the baseline SVM method as the number of curvelets increased.

There are now two questions left to be answered: to what extent is the 2D-CNN sensitive to position bias and resolution changes, and which part of the images are actually used in the network decision?

E. Position and Resolution Dependency

In an attempt to answer one of the questions raised, we investigated the robustness of our preprocessing method towards shifts in positions and/or resolutions. As mentioned, the fused curve plots and fixed time window plots capture position information relative to the screen resolution. Therefore, the plots depend on the screen resolutions and are resized to a fixed size of 448×448. While maintaining the aspect ratio of the plot, the larger side of the plot (width) is resized to 448. Since the length of the shorter side will be less than 448 pixels, all sides of the plot are equally padded if necessary, to obtain the desired size of 448×448. The resized plots will still reflect the position information of the coordinates relative to the resolution. This information can be used by the classifier. We investigated the question of performance decrease, if the 2D-CNN is trained in a position-invariant manner. From the results in Table II, it is clear that the moving window stride approach is unsuitable to preserve both position invariance and prediction performance. Thus, we added an additional preprocessing step over the resized fused plots to remove position dependencies from the plots and increase robustness. The additional step involved shifting the minimum bounding box of the curve towards the center of the plot, thereby removing all traces of position dependencies. This was performed for training and test data. The comparison can be seen in the first two rows of Table III.

Table III: Robustness against changes in environment.

Pos. Dependency	Reso. Dependency	Avg AUC	Avg EER
✓	✓	0.9584	0.0984
✗	✓	0.9476	0.1124
✓	✗	0.4672	0.5292
✗	✗	0.4458	0.5434

The removal of position dependencies from the model leads only to a slight drop in performance. Note that this centering can be performed at test time easily. However, one can identify a fail case of the model, namely when resolution changes at test time. The Balabit dataset does not support such a diversity. Given the resized mouse movement plots of 448×448, one can simulate resolution change by shrinking the mouse curve towards the upper left corner while retaining the size of the image. Once resolution dependencies are removed in such a way, i.e., resolutions for train and test sets are different, the model's performance plunged down severely, as seen in the last two rows of Table III. These observations are indicators that the CNN model is insensitive to change in positions but highly sensitive to change in resolutions at test time.

F. Explaining what the Classifier used

To see which parts of the mouse movement images are important for the decision of the neural network, we employed the LRP method [4]. It decomposes the prediction into a relevance score for every pixel so that the pixel-wise scores sum up to the prediction score. The pixel-wise scores can be visualized with a relevance heatmap over pixels. LRP was chosen because it has been shown to outperform gradient/sensitivity-based methods [22]. For a theoretical explanation see [23]. The generated heatmaps are shown in Fig. 5, highlighting the regions of the curve used in the network's decision, with positive evidences being yellow and negative being cyan. One can observe that the gradient-based heatmap appears to be more noisy, because the gradient explains how to change the image to increase the prediction score rather than to decompose it into parts. The LRP rule used here was a hybrid rule: for fully connected layers the ϵ-rule (cf. [4]) with $\epsilon = 0.1$, for convolutional layers the β-rule (cf. [4]) with $\beta = 0$. Since we had to keep figure size small, we postprocessed the heatmaps, except for Figure 5: we applied a squareroot on the absolute value of the heatmap scores and five steps of morphological thickening for values above 0.1 for thicker lines.

One can see several aspects. Firstly, the relevance is distributed sparsely. Edges and curved segments tend to have higher assigned relevance than long straight lines, see Figure 6. Secondly, artifactual crossings of the movement tend to receive lower absolute scores, which appear fainter than other parts. These are observed in Figures 7 and 8 which demonstrate the ability of deep architectures to mitigate artifacts. Thirdly, the LRP scores for the same sample but different user classifiers are not a complement of each

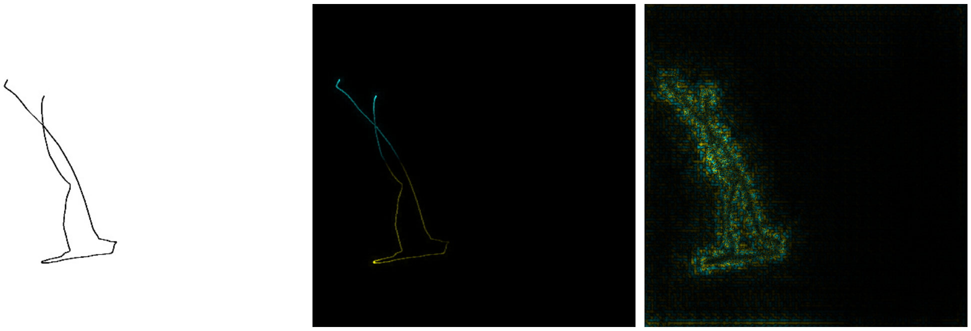

Figure 5: Original plot with LRP heatmap (Middle) and gradient heatmap (Right), both with original sparsity of outputs.

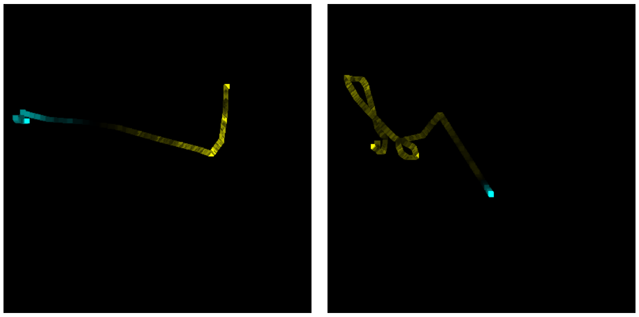

Figure 6: Sparsity altered by morphological thickening for improved perception, also for the following plots.

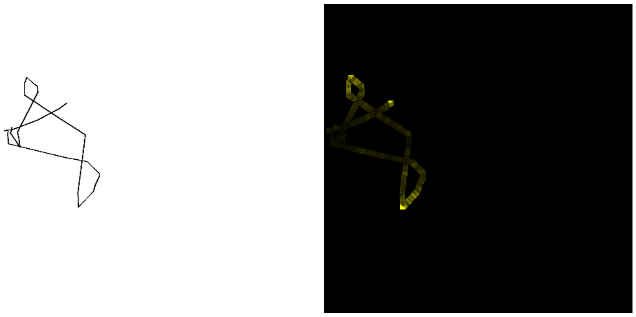

Figure 7: Artifactual crossings of the curve receive lower absolute scores.

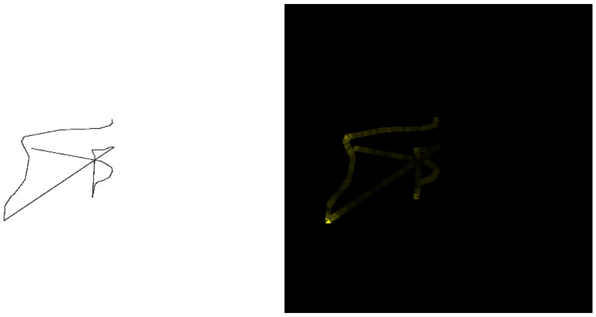

Figure 8: Artifactual crossings of the curve receive lower absolute scores.

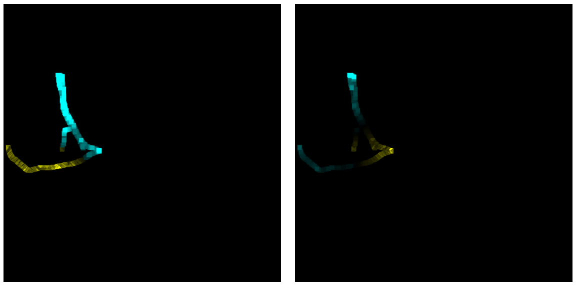

Figure 9: Left and right figures show the LRP heatmaps of Balabit user 9 extracted from the classifiers of user 9 and user 7 respectively.

other, that is negative scores (cyan) do not turn into their positive complement (yellow) and vice versa, as can be seen in Figure 9. It shows that the joint feature learning does not learn very similar models for the different classifiers. This serves as a sanity check for the multi-label model.

V. Conclusion

CNNs, both 1D and 2D are able to perform well on two datasets, though a combination of them together with the fixed feature approach might improve. For the 2D-CNN, LRP helps to obtain insights on the learned features, showing a sparse distribution of relevance scores, and low weights on crossing artifacts. While preventing overfitting, the multi-label joint training reduces the training time for multiple users. The 2D-CNN is sensitive to resolution shifts at test time, an aspect which needs to be tackled for serious usability and checked for other methods as well. Regarding performance, trust models [24], [25] can be used to reduce the impact of prediction errors.

Acknowledgment

The authors are grateful for the support by the ST Electronics-SUTD Cyber Security Laboratory.

REFERENCES

[1] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997. [Online]. Available: http://dx.doi.org/10.1162/neco.1997.9.8.1735

[2] Y. Kim, “Convolutional neural networks for sentence classification,” *arXiv preprint arXiv:1408.5882*, 2014.

[3] M. D. Zeiler and R. Fergus, “Visualizing and understanding convolutional networks,” in *ECCV*, 2014, pp. 818–833.

[4] S. Bach, A. Binder, G. Montavon, F. Klauschen, K.-R. Müller, and W. Samek, “On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation,” *PloS one*, vol. 10, no. 7, p. e0130140, 2015.

[5] A. F. Syukri, E. Okamoto, and M. Mambo, “A user identification system using signature written with mouse,” in *Australasian Conference on Information Security and Privacy*. Springer, 1998, pp. 403–414.

[6] A. A. E. Ahmed and I. Traore, “A new biometric technology based on mouse dynamics,” *IEEE Transactions on dependable and secure computing*, vol. 4, no. 3, 2007.

[7] C. Feher, Y. Elovici, R. Moskovitch, L. Rokach, and A. Schclar, “User identity verification via mouse dynamics,” *Information Sciences*, vol. 201, pp. 19–36, 2012.

[8] C. Shen, Z. Cai, X. Guan, Y. Du, and R. A. Maxion, “User authentication through mouse dynamics,” *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 1, pp. 16–30, 2013.

[9] K. O. Bailey, J. S. Okolica, and G. L. Peterson, “User identification and authentication using multi-modal behavioral biometrics,” *Computers & Security*, vol. 43, pp. 77–89, 2014.

[10] Y. X. M. Tan, A. Binder, and A. Roy, “Insights from curve fitting models in mouse dynamics authentication systems,” in *IEEE Conference on Applications, Information and Network Security (AINS)*, 2017.

[11] N. Zheng, A. Paloski, and H. Wang, “An efficient user verification system via mouse movements,” in *Proceedings of the 18th ACM conference on Computer and communications security*. ACM, 2011, pp. 139–150.

[12] S. Mondal and P. Bours, “A study on continuous authentication using a combination of keystroke and mouse biometrics,” *Neurocomputing*, vol. 230, pp. 1–22, 2017.

[13] B. Sayed, I. Traore, I. Woungang, and M. S. Obaidat, “Biometric authentication using mouse gesture dynamics,” *IEEE Systems Journal*, vol. 7, no. 2, pp. 262–274, 2013.

[14] C. Shen, Z. Cai, X. Guan, and J. Wang, “On the effectiveness and applicability of mouse dynamics biometric for static authentication: A benchmark study,” in *Biometrics (ICB), 2012 5th IAPR International Conference on*. IEEE, 2012, pp. 378–383.

[15] Y. Nakkabi, I. Traoré, and A. A. E. Ahmed, “Improving mouse dynamics biometric performance using variance reduction via extractors with separate features,” *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, vol. 40, no. 6, pp. 1345–1353, 2010.

[16] C. Shen, Z. Cai, X. Guan, and R. Maxion, “Performance evaluation of anomaly-detection algorithms for mouse dynamics,” *Computers & Security*, vol. 45, pp. 156–171, 2014.

[17] Z. Jorgensen and T. Yu, “On mouse dynamics as a behavioral biometric for authentication,” in *Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security*. ACM, 2011, pp. 476–482.

[18] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, “Going deeper with convolutions,” in *CVPR*, 2015, pp. 1–9. [Online]. Available: http://dx.doi.org/10.1109/CVPR.2015.7298594

[19] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. B. Girshick, S. Guadarrama, and T. Darrell, “Caffe: Convolutional architecture for fast feature embedding,” in *Proc. of the ACM Int. Conf. on Multimedia*, 2014, pp. 675–678. [Online]. Available: http://doi.acm.org/10.1145/2647868.2654889

[20] Á. Fülöp, L. Kovács, T. Kurics, and E. Windhager-Pokol. Balabit mouse dynamics challenge data set. [Online]. Available: https://github.com/balabit/Mouse-Dynamics-Challenge

[21] A. Harilal, F. Toffalini, J. Castellanos, J. Guarnizo, I. Homoliak, and M. Ochoa, “Twos: A dataset of malicious insider threat behavior based on a gamified competition,” in *9th ACM CCS International Workshop on Managing Insider Security Threats*, 10 2017, pp. 45–56.

[22] W. Samek, A. Binder, G. Montavon, S. Lapuschkin, and K. Müller, “Evaluating the visualization of what a deep neural network has learned,” *IEEE Trans. Neural Netw. Learning Syst.*, vol. 28, no. 11, pp. 2660–2673, 2017. [Online]. Available: https://doi.org/10.1109/TNNLS.2016.2599820

[23] G. Montavon, S. Lapuschkin, A. Binder, W. Samek, and K.-R. Müller, “Explaining nonlinear classification decisions with deep taylor decomposition,” *Pattern Recognition*, vol. 65, no. Supplement C, pp. 211 – 222, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0031320316303582

[24] S. Mondal and P. Bours, “A computational approach to the continuous authentication biometric system,” *Information Sciences*, vol. 304, pp. 28 – 53, 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0020025514011979

[25] P. Bours, “Continuous keystroke dynamics: A different perspective towards biometric evaluation,” *Information Security Technical Report*, vol. 17, no. 1, pp. 36 – 43, 2012, human Factors and Bio-metrics. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1363412712000027