User Authentication via Multifaceted Mouse Movements and Outlier Exposure

Jennifer J. Matthiesen[1(✉)], Hanne Hastedt[2], and Ulf Brefeld[1]

[1] Leuphana University of Lüneburg, Lüneburg, Germany
{jennifer.matthiesen,brefeld}@leuphana.de
[2] Georg-August-University of Göttingen, Göttingen, Germany

Abstract. Gaining information about how users interact with systems is key to behavioural biometrics. Particularly mouse movements of users have been proven beneficial to authentication tasks for being inexpensive and non-intrusive. State-of-the-art approaches consider this problem an instance of supervised classification tasks. In this paper, we argue that the problem is actually closer to unsupervised one-class classification tasks. We thus propose to view behavioural user authentication as an unsupervised task and learn individual models using data from a single user only. We further show that, by being purely unsupervised, losses in performance can be counterbalanced by augmenting additional data into the training processes (outlier exposure). Empirical results show that our approach is very effective and outperforms the state-of-the-art in several performance metrics.

Keywords: User Authentication · Mouse Dynamics · Anomaly Detection

1 Introduction

User authentication is most commonly based on user-determinant keys, such as passwords or pin codes. In contrast to these traditional systems, biometric authentication [22,40] provides an additional layer of security. But these approaches come at the cost of storing sensitive data like fingerprints and require dedicated pieces of hardware.

An inexpensive and readily-available alternative is offered by *behavioural biometrics* [14]. In contrast to their biometric peers, these methods are non-intrusive and continuously analyze user behaviour for authentication during a session. Behavioural traits of users have been analysed in handwriting [9], voice [41], or keyboard and mouse dynamics [26]. Particularly the latter suggests itself for user authentication in computer-based systems since keyboard and mouse are considered standard equipment. While keystroke dynamics may contain sensitive personal information like passwords, mouse movements offer an implicit and non-sensitive measurement of idiosyncratic behaviour [17]. In fact, Rodden et al. [29] show that eye and mouse movement are significantly correlated and conclude that mouse movement serves as an appropriate proxy to address implicit user behaviour.

Mouse movement dynamics are typically handled in a fully-supervised multi-class or multi-label setup, where class labels are identified with user IDs. While being purely

B. Crémilleux et al. (Eds.): IDA 2023, LNCS 13876, pp. 300–313, 2023.
https://doi.org/10.1007/978-3-031-30047-9_24

supervised can add to predictive accuracy and detection performance, there are important limitations with this formalization. Firstly, the above approaches are often biased towards the data of other users present in training due to learning discriminating functions. Secondly, maintaining a multi-class approach in practice is close to being infeasible as every new user requires a full re-training of all models.

By contrast, we consider user authentication in an unsupervised approach, which learns a user's representations by using only the data of this very user. Learning individual models of normality for every user allows to create features which are independent from other users. This allows the model to generalise better with respect to the target user, especially in the presence of unknown users who have not been seen during training.

In this paper, we propose a novel methodology for user authentication using mouse dynamics and a deep one-class setup. Our contributions are as follows: (i) We phrase user authentication as a deep one-class machine learning problem that can enhance user authentication and (ii) show the effectiveness of using a multifaced input to extract appropriate features from mouse data. Finally, (iii) we visualise the individual and relevant parts of the user's mouse trajectory to gain an understanding of our approach, showing our model indeed focuses on characteristics previously known to be important from hand-crafted features but unattended in unsupervised approaches so far.

2 Related Work

Many studies have shown that mouse dynamics can deliver insights into a user's mood [42], satisfaction and frustration [6] or mental state [13]. Consequentially, mouse movement has received much attention in behavioural analyses [2,3,24,38].

The identification of users based on mouse strokes has been first investigated on the example of mouse-written signatures [9]. Many algorithmic approaches in dealing with mouse movement rely on hand-crafted features [11,14]. Such representations are often extended by peers to increase expressiveness and/or incorporate additional characteristics like the number of pauses or pause length [25]. Matthiesen et al. [25] showed that using a fixed feature set for every user does not cover all of them equally. Similar conclusions were made by [34]. Therefore, an individual feature set for every single user is required. Neural approaches try to overcome the dependency on hand-crafted features while mapping the input data to a feature space using corresponding objectives. Using mouse dynamics to tell users apart can have two main objectives: User identification and user authentication. Many of the previous work addresses the topic of user identification, which is to detect the right user in a set of all users [1,11,14,21,36]. The usual setup for this is a supervised multi-class classification. In contrast, user authentication underlies a binary decision, e.g. is the target user/ is not the target user [7,21,26,34,35,39]. Note that the common approach here is still supervised, i.e. using data of both classes. Chong et al. [7] are the first to investigate the potential of deep neural networks for mouse dynamics. They investigate multiple network architectures while casting the problem as a supervised multi-label problem. Applying a similar architecture, [1] propose a one-dimensional convolutional network (1D-CNN) for modelling temporal aspects of mouse movement. They train the models in a supervised one vs. rest manner using a binary-cross entropy loss.

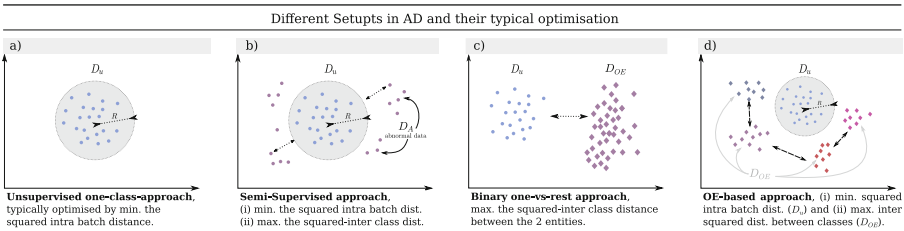


Fig. 1. Setups of AD, from only using the target class to incorporate an additional dataset.

Multi-class approaches imply retraining the whole model when adding/deleting users and are not feasible in dynamic environments. Binary one-vs-all strategies, on the other hand, are often biased towards the seen anomalies. Thus, we argue that an unsupervised anomaly detection (AD) method is more suitable for mouse-dynamics-based user authentication in real-world applications.

In the context of AD, unsupervised one-class approaches are appealing because they find minimum volume summarization of data at hand through hyperplanes [33] or hyperspheres [37]. Neural peers [20,30] introduce improvements by identifying anomalies through their alterity in feature space. This is often done by including additional data that are not part of the target concept, into the training process, for example by semi-supervised learning [16], pre-training [19,28], reference data [27,28] or outlier exposure (OE) [19]. We will make use of the latter in the remainder. An overview of common setups incorporating additional data into AD and their typical optimisation can be found in Fig. 1. For example, in contrast to a binary one-vs-rest strategy, the OE-approach extract rich descriptive features from the mouse trajectories instead of focussing on increasing the distance between the two entities (normal and anomalous data). Note that the concept of OE in AD can be found widely in the literature under various terms, such as reference dataset [27,28], auxiliary or OE dataset [19,31]. This auxiliary dataset enables the anomaly detector to generalize better for unseen data.

3 Representing Mouse Trajectories

Formally, a mouse trajectory is given by a sequence of spatial (x, y) coordinates ordered in time τ. In addition to the spatio-temporal information, mouse data contains events, for example $e \in \{\emptyset, c_L, c_R, c_M, s_{\text{up}}, s_{\text{down}}\}$, with left ($c_L$), right ($c_R$) or center ($c_M$) clicks, up $\{s_{\text{up}}\}$ and down $\{s_{\text{down}}\}$ scrolls, or $\emptyset$ in case there is no event. We represent a mouse trajectory as a sequence $\boldsymbol{x} = \langle(\tau_1, x_1, y_1, e_1), \ldots, (\tau_T, x_T, y_T, e_T)\rangle$.

Mouse data records consist likewise of movements for interacting with the application as well as idiosyncratic movements. In the following, we aim at devising a representation that is as independent as possible from actual user interface (UI) and rather aim at capturing *how* a user moves the pointer to a certain location instead of where exactly an action has been performed. While such velocities are translation invariant, they also render certain patterns almost undetectable (e.g. loops).

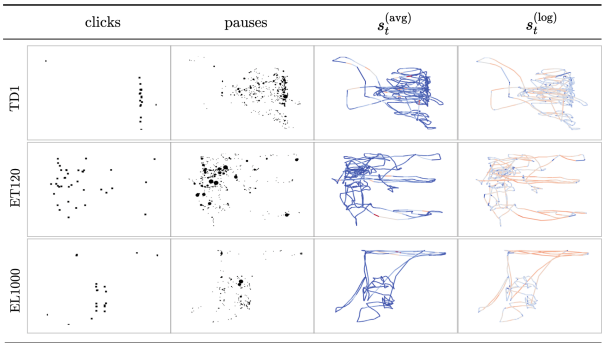


Fig. 2. Views of mouse movement with different splitting criterions.

3.1 Image-Based Tensor Representations

We propose to represent different *views* of mouse trajectories (e.g., trajectory, speed, click, pause) as an image, which allows to access shape of motion as well as characteristic patterns like loop or hesitation [4,7], see Fig. 2 for examples. A convolutional neural network (CNN) can detect edges very well and is therefore perfectly suited for such shapes. For the *trajectory view*, sub-sequences of the trajectory are re-scaled, plotted and saved as images. We adapt the size of the plot to the range of the respective trajectory to assure no bias from the positioning on the screen. Later, those images will be transformed into a multidimensional tensor to serve as an input for the network. To maintain the temporal information, we encode the *speed of the movement* with a colour interval, where the colour is determined by the actual speed of movement s_t at that position. To encode the speed value, we test two different normalisation approaches. We report on experiments with different normalisations in Sect. 6.1. Both ground on the speed $s_t = \frac{d_t}{\tau_t}$ of the movement, where d is the Euclidean distance, but are normalized (i) by the average speed: $s_t^{(avg)} = \frac{s_t}{\frac{1}{T}\sum_{t=1}^{T}\frac{d_t}{\tau_t}}$ and (ii) with a log-variant with $\tilde{s}_t = \log(1+s_t)$ $s_t^{(log)} = \frac{\tilde{s}_t - \tilde{s}_{\max}}{\tilde{s}_{\max} - \tilde{s}_{\min}}$, respectively, where $\tilde{s}_{max} = \max_t \log(1+s_t)$ and $\tilde{s}_{min}$ analogously. A *click view* is simply containing indicators at click positions which are visualized by black crosses in the figure. The *pause view* contains the length of the pauses at the observed positions and is visualized by circles with radii corresponding to the length of the pause. We scale every pause so that the radius of the smallest pauses starts at a fixed radius. The different layers are aggregated to a multi-dimensional tensor, using one channel each, and serve as input to the model. We experiment with several combinations of input information and report the results in Table 1 (left).

3.2 Splitting Sessions

The overall sequences of our data cover a whole session of a user. Therefore we divide the total session into sub-sequences. The length of those sequences in the images is another aspect of representing mouse trajectory data. Since it is not obvious how to split

a long user session into smaller meaningful pieces, we study three different splitting criteria in the empirical evaluation in Sect. 6.1 and describe them in the following.

Time Difference Split (TD). TD [7] splits a sequence when the time difference between two consecutive mouse operations (movement or click) exceeds a predefined threshold $\rho \in \{1s, 60s\}$. Since this may result in very short sub-trajectories, we only split if the resulting sub-sequences contain at least 100 data points.

Equal Length Split (EL). EL [25] splits the data into sub-sequences of the same length, $\omega \in \{200, 1000\}$, irrespective of occurring events or movements. The last sequence is naturally shorter and usually discarded. In contrast to the TD method, the resulting sequences have the same length. Note that the identical number of data points does not result in the same amount of coloured pixels in the generated image.

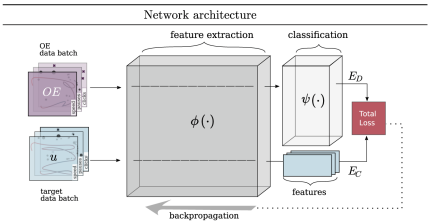
Equal Time Split (ET). There exist two main ways of recording trajectories: (i) record the position in equal time stamps, so that a static position will results in duplicates coordinates or (ii) recording on movements, meaning the distances between consecutive points will not be the same. Since the latter is the case in the *Balabit* data, we introduce an additional method: The Equal Time Split (ET). It is a temporal analogy to the previous splitting criterion and splits the trajectory after a fixed amount of time. We experiment with the thresholds $\upsilon \in \{10s, 120s\}$. Since the used mouse data is not recorded using equal time stamps but rather recorded on movements, this splitting method will not result in equally sized sub-sequences. Although this extension is straightforward, there does not seem to exist related work on this method.

4 Deep Anomaly Detection and Outlier Exposure

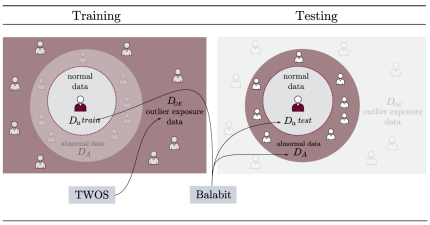
The main goal in an AD task is to obtain a feature map ϕ which separates the normal data from the outliers either linearly [33] or spherically [37]. Following the latter approach, a neural variant of the so-called Support Vector Data Description (SVDD), maps the data into a feature space $\mathcal{F}$ and finds the minimal enclosing sphere with center c and radius $R > 0$ that contains the majority of points. We derive two important characteristics for features for our one-class setup similar to [28].

(i) **Compactness.** Following the above-described cluster assumption [32], we want a similar feature representation extracted from the same class lie compactly in an enclosing hypersphere within feature space. Similar to the SVDD, the objective is to minimise the R of the hypersphere will result in reducing its volume and a more compact representation. However, if no further constraints are included this will directly result in the *hypersphere collapse* [8], when all data is mapped to the same point.

(ii) **Descriptiveness.** A feature map that gives rise to compact representations may not be rich enough to distinguish other users. We thus aim to devise a rich feature representation that is general enough to not only summarize individual user data well but, at the same time, allows us to identify other users because of their unique traits in moving the mouse. Thus, we need *descriptiveness* but cannot give up on compactness either (cf. [27]). Producing descriptive features is likewise a desired characteristic in multi-class classification, where those features would ensure a large inter-class distance.



(a) The network ϕ extracts features from both Balabit and TWOS, while additional layers of ψ are used for the auxiliary data to calculate the descriptive loss E_D.



(b) For training we use the data of the target user (Balabit) and a sample of OE data (TWOS). For testing we use the legal users from Balabit.

Fig. 3. Depiction of the network (a) structure and of (b) the process for training and testing.

4.1 Outlier Exposure

The idea of OE originates in the observation that when learning a target concept, myriads of labelled examples exist that live in the same space but are known to *not* match the target concept [19]. While this insight borders on triviality, it is particularly powerful in unsupervised learning tasks like one-class and density estimation problems. Instead of only feeding observations of the desired target concept, additional data from possibly very different origins and sources is made available to the learner that now faces contrastive tasks: Ultimately, the goal is to provide a minimal description of the desired target concept but additionally, there is a classification problem that needs to be solved simultaneously using only the auxiliary data. The goal of the learning process is to identify a set of features that not only accounts for minimal description of the target concept but also induces high predictive accuracies on the auxiliary data. Note that OE-based approaches are generally classifies as unsupervised methods across literature [19,27,28,31] since the standard approach uses only data of one data (normal data) from the target dataset during training.

5 Authentication of Users

For the underlying architecture for our model, we consider the AlexNet CNN architecture [23]. We modify the network and separate it into the two parts ϕ und ψ for

feature extraction and classification layers respectively. The network architecture is depicted in Fig. 3a. The feature extraction component of the network $\phi : \mathcal{X} \to \mathbb{R}^p$ acts on both sources, the target classes D_u and the OE data D_{OE}, to render learning compact as well as descriptive representations feasible. While the auxiliary data then branches into a standard feed-forward classification component $\psi : \mathbb{R}^p \to \mathbb{R}^{|\mathcal{Y}|}$ with a final softmax layer for descriptiveness, the compactness of the user data is evaluated by a variance-based criterion. The task in the optimization is now to find an appropriate feature extraction ϕ, such that the classification error and variance of user data is small, while simultaneously ensuring a compact representation of the features for D_u. This is achieved by deploying dedicated loss functions for controlling compactness and descriptiveness, respectively, and minimizing the two losses simultaneously (cf. [28]). The loss-controlling compactness measures the squared intra-batch distance

$$E_C = \frac{1}{N} \sum_{n=1}^{N} (\phi(\boldsymbol{x}_n; \theta) - \bar{\boldsymbol{x}}_{\neg n})^2, \tag{1}$$

with mean $\bar{\boldsymbol{x}}_{\neg n} = \frac{1}{N-1} \sum_{j \neq n} (\phi(\boldsymbol{x}_j; \theta))$ of the leave-one-out set $D_u \setminus \{\boldsymbol{x}_n\}$. The descriptiveness loss is given by the cross entropy over all involved classes $\mathcal{Y}$, given by

$$E_D = -\frac{1}{M} \sum_{m=1}^{M} \sum_{\bar{y} \in \mathcal{Y}} \delta_{\bar{y}, y_{N+m}} \log \left(\psi(\boldsymbol{x}_{N+m}; \theta) \right), \tag{2}$$

where δ is the Kronecker delta. The joint objective function for the entire architecture is given by aggregating Eqs. (1) and (2). We minimize $E_C(D_u) + \lambda E_D(D_{OE})$, where $\lambda > 0$ is a balancing term. In addition to user data D_u, we introduce an auxiliary and labelled M-sample $D_{OE} = \{(\boldsymbol{x}_{N+1}, y_{N+1}), \ldots, (\boldsymbol{x}_{N+M}, y_{N+M})\}$ with $\boldsymbol{x}_{N+m} \in \mathcal{X}$ and $y_{N+m} \in \mathcal{Y}$ for $1 \leq m \leq M$ and $\mathcal{Y}$ denotes the set of (arbitrary) class labels of the auxiliary data. Recall that both user observations $\boldsymbol{x}_n^{(u)}$ and auxiliary data $\boldsymbol{x}_{N+m}$ live in the same space $\mathcal{X}$ for all n, m. Here, we propose a neural architecture that combines learning a compact representation of target data D_u *and* a descriptive feature space on target and auxiliary data.

6 Empirical Results

We evaluate on the Balabit Mouse Dynamics Challenge [12] for sampling D_u and incorporate instances of the Wolf of SUTD (TWOS) [18] data as D_{OE}. *Balabit* contains mouse movements from 10 users from 65 sessions between 13640 and 83091 data points each, recorded during a set of unspecified but common administrative tasks. Since the screen resolution is not given, we normalize the trajectories based on the maximum coordinates. The TWOS data is the outcome of a gamified competition among competing companies over five days. It consists of 320 h of activity of 24 users and comprises a mouse, keyboard and other actions and logs, where we only use legal mouse movements in our experiments.

Setup. The two parts of the network ϕ and ψ are trained jointly with samples from both D_u and D_{OE}. For each user u, we train an individual model using only data

Table 1. Results of the experiments on different views (left) and splitting criteria (right). * Results are averaged over 9 users, since some images of user 07 resulted in numerical issues for those representations (see further details in Sect. 8).

	avg. AUC	avg. EER
trajectory	0.670	0.420
s_{avg}	0.656	0.268
s_{avg}, pause	0.698*	0.697*
$s_{\log}$, pause	0.710*	0.179*
s_{avg}, pause, click	0.755	0.272
$s_{\log}$, pause, click	0.777	0.240

	Avg. No. Img.	avg. AUC	avg. EER
TD1	1074	0.517	0.017
TD60	77	0.576	0.476
ET10	2861	0.516	0.867
ET120	344	0.606	0.600
EL200	950	0.723	0.409
EL1000	188	0.777	0.240

from that user $D_{u_{\text{train}}}$ (Balabit) plus a sample from the 24 users as additional OE data D_{OE} (TWOS). More formally, let the size of a batch be n. Then, for every i^{th} sample $x_i^{D_u} \in \mathbb{R}^k$, where $1 \leq i \leq n$, we calculate the distance between the networks output and the rest of the batch. For every i^{th} sample $(x_i^{D_{OE}}, y_i) \in \mathbb{R}^k$, where $1 \leq i \leq n$, we calculate a loss for each class label y_i and sum the result. Hyperparameters are found via grid search and given by $\lambda = 1.0$, 300 epochs and a learning rate of $\eta = 0.0001$ on balanced batches containing 100 user and OE samples. We observe that a rather low learning rate results in better performance since it prevents overfitting on the OE data while still assuring convergence on the compactness loss. At test time, we use independent data of the target user $D_{u_{\text{test}}}$ and the nine remaining users from Balabit, similar to [15,20,28,30]. Note that the model has never seen the other users from *Balabit* during training. In this way, we ensure that the model can even distinguish from unseen users. This allows scalability and does not require retraining of the model, even if more new users are added.

6.1 Results

We first execute preliminary experiments, learning the optimal input and splitting strategy as described in Sect. 3.2. Using the resulting best-performing representation of mouse trajectories, we train the presented model in two different setups: (i) To show the influence of the utilisation of the OE data, we first train the model without the usage of the additional data. (ii) We build upon that and show the improvement in performance reached through the usage of OE data. We report average Areas under the ROC curve (AUCs) and equal error rates (EERs) over five repetitions.

Results for Optimal Representation. Table 1 (left) shows the results for different views (layers of input tensors), presented in Sect. 3. The results lay out that including likewise the pause and click *view* in the tensor leads to higher detection rates and the log-average performs slightly better than the global average. Table 1 (right) shows the results for different splitting criteria, also presented in Sect. 3. Firstly, the table nicely shows that the heuristics lead to considerably different numbers of training instances, however, recall that fewer instances contain longer parts of the respective user sessions.

Table 2. Detection performance per class. Results are retrieved over 5 random seeds.

	EER (↓)				AUC (↑)			
	[30] (features [25])	[30] (images)	ours D_u	ours $D_u \cup D_{OE}$	[30] (features [25])	[30] (images)	ours D_u	ours $D_u \cup D_{OE}$
user 07	0.302	0.399	0.414	0.181	**0.879**	0.542	0.714	0.857
user 09	0.207	0.608	0.451	0.237	**0.911**	0.474	0.661	0.80
user 12	0.629	0.263	0.325	0.090	0.250	0.607	0.594	**0.838**
user 15	0.532	0.437	0.152	0.219	0.426	0.550	0.555	**0.714**
user 16	0.552	0.492	0.418	0.418	0.424	0.515	0.682	**0.720**
user 20	0.402	0.463	0.716	0.290	**0.788**	0.496	0.490	0.728
user 21	0.476	0.332	0.281	0.138	0.548	0.554	0.625	**0.825**
user 23	0.619	0.403	0.216	0.305	0.267	0.577	0.662	**0.730**
user 29	0.619	0.500	0.533	0.320	0.346	0.472	0.7	**0.813**
user 35	0.609	0.420	0.265	0.200	0.283	0.554	0.662	**0.747**
Mean	**0.495**	**0.432**	**0.378**	**0.240**	**0.512**	**0.534**	**0.634**	**0.777**

While most splitting methods are just slightly better than random guessing, the EL split performs notably better. With $\omega = 1000$, decreases the EER by almost half.

Results for User Authentication. We now use the best-performing representation to compare to related work. As a baseline, we use the performance of the deepSVDD proposed in [30] as well as the user authentication approach using features proposed for Balabit from [25]. To show the influence of OE data, we also compare our approach to a variant that does not leverage OE data. To prevent hypersphere collapse, we incorporated an additional regularizer into Eq. (1) similar to [37]. To additionally show the benefit of our representation over state-of-the-art handcrafted features, we train a deepSVDD on features taken from [25] and another one on our tensor representation.

The results are shown in Table 2. Interestingly, the baseline on features and tensors leaves a mixed picture in terms of AUC (right part of the table). For users 07, 09, and 20, hand-crafted features outperform the tensor-based deepSVDD as well as the proposed approach. For the other users, the image-based representation is favourable, often by a large margin as seen for users 12, 23, or 35 which is also reflected by a slightly better average AUC over all users. However, even without including OE data, our proposed approach performs either on par or improves over the stronger baselines. This result impressively improved by including OE. Our proposed approach already constitutes an improvement in AUC by a factor of 1.2 over the baselines when no OE data is included in the training. Note that in this case, a regularizer has to be added to Eq. 1 to avoid a collapse of the hypersphere. The results for including OE are even better and raise the improvement in AUC by a factor of 1.5.

7 Visualisation of Important Information of the Mouse Dynamics

Since the performance using all three *views* and the splitting method EL1000 result in the best authentication performance, we can now investigate which parts of the input lead to creating compact and descriptive features for each user. To achieve this, we

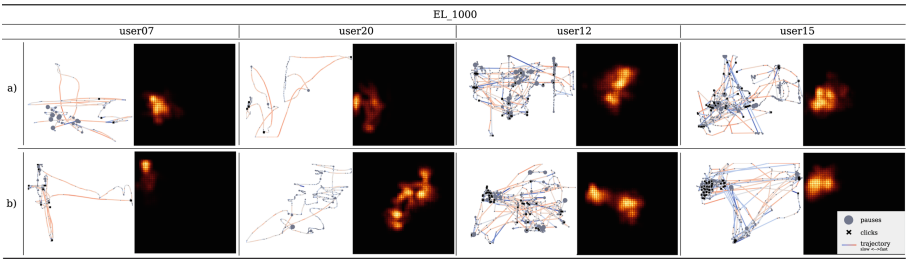


Fig. 4. Two examples per user of trajectories images and their LRP visualisation using the splitting criterion EL1000. We refer to the left example as (a) and the right as example (b).

utilise the layer-wise relevance propagation (LRP) [5]. It can help identify the parts of the input while highlighting input features that were decisive for the network's decision. The relevance R of every neuron is computed as follows: $R_i^{(l)} = \sum_j \frac{a_i w_{ij}}{\sum_{i'} a_{i'} w_{i'j}} R_j^{(l+1)}$, where $R_i^{(l)}$ and $R_i^{(l+1)}$ represent the relevance score of the neurons i and j in the layers l and $l+1$ respectively. The activation of neuron i is represented as a_i and the weight connecting neuron i and j as w_{ij}. The LRP heatmap is then obtained by applying this principle to all layers. In addition, we implement the z^+-rule and a relevance filter as suggested in [10]: We adapt the threshold value for the filter to $k = 0.05$.

The results are shown in Fig. 4. Note that the trajectory is reconstructed for better legibility. The images used as input carry one channel per *view*, where different shadings are hard to detect for the human eye. The trajectory of user 07 (a) shows clearly the advantage of incorporating pauses into the input image. It can be seen that the pauses got a much higher relevance score than the clicks (black cross). Locally overlapping occurrences of pauses and clicks are likewise relevant. In contrast, the clicks are much more relevant to user 20 as can be seen in example a). When no clicks are made, the pauses are getting more relevant. In [7] only the plotted trajectory was used as input for the CNN. It was shown that the edges are the relevant element for the decision process of the network. Added pauses and clicks carry even more relevant information for user authentication and should not be left out in image-based deep learning approaches.

8 Discussion and Limitation

In this study, we cast user authentication base on mouse dynamics as a one-class problem. Multiple *views* of the trajectories are used as input to a CNN for extracting features using the objective of compactness and descriptiveness. Related work using deep neural networks for mouse trajectory data view the problem as a purely supervised task and often rely on pieces of information that is not always present, such as screen resolution [1,7]. We remove this implicit dependency on the screen to avoid identifying users based on their personal preferences or hardware but still report state-of-the-art results.

In our setup, we reached the best performance by equally weighted both losses, setting $\lambda = 1.0$. We did not detect a large difference in performance though. With the EL1000 split the trajectories of users 07, 09 and 20 cover much shorter (pixel-wise)

distances than the remaining users. Interestingly, these are exactly the users for which the hand-crafted features were performing well. However, our results are in line with [25] and show that even shorter sequences for the remaining users did not enhance the performance. Mouse trajectories are not translation invariant. While some movements, like patterns of confidence (e.g. straight and direct movements), can still be detected in mirrored or rotated images, other mouse movement motifs can not be orientation invariant and lose their idiosyncratic characteristic. Therefore, we did not include additional data augmentation to generate more data (e.g. through mirroring or rotation).

In contrast to our setup, a binary one-vs-rest strategy as used in [1] assumes that the "rest" classes (e.g. anomaly samples) are representative for all other occurring anomalies. Often the same classes are used in training as well as testing, resulting in a high accuracy, but introducing a selection bias. Using an auxiliary dataset from the same field as the target dataset has been shown beneficial to increase authentication performance in mouse trajectories. Since the auxiliary dataset is just used for training but not for testing, the model even performs well when testing against trajectories of unseen users. In comparison to previous methods [25], the CNN-based model overcomes the dependency on hand-crafted features while learning to extract an individual feature set for every user.

For the presented approach, we compare different setups and *views*. To ensure a fair comparison we left the structure of the underlying model untouched. However, when taking the [s_{avg}, pause] *view* or the [$s_{\log}$, pause] *view*, some images form user 07 caused numerical instabilities. There is no obvious visual difference in data between user 07 and other similar users. We excluded the models with these setups for further analysis and emphasize to utilise other combinations of trajectory representations when using this particular model.

9 Conclusion and Future Work

In this paper, we proposed an unsupervised learning approach for user authentication using only the data of one user for training. We showed that incorporating additional data can enhance the model's performance so that a distinction even to unknown users, which were never seen during training, becomes possible. This enables a deeper understanding of mouse cursor movements by visualising important key parts of the mouse trajectory for single users. Future research efforts should be directed to improve the discovery of mouse cursor motifs for individual users and their interplay with pauses.
We thank the web-netz GmbH for funding this research and all former reviewers for the valuable feedback.

References

1. Antal, M., Fejér, N.: Mouse dynamics based user recognition using deep learning. Acta Universitatis Sapientiae, Informatica **12**, 39–50 (2020)
2. Arapakis, I., Lalmas, M., Valkanas, G.: Understanding within-content engagement through pattern analysis of mouse gestures. In: Proc. of the 23rd ACM CIKM, pp. 1439–1448 (2014)

3. Arapakis, I., Leiva, L.A.: Predicting user engagement with direct displays using mouse cursor information. In: Proc. of the 39th Int. ACM SIGIR Conf. on Research and Development in Inform. Retrieval, pp. 599–608. ACM (2016)
4. Atterer, R., Wnuk, M., Schmidt, A.: Knowing the user's every move: user activity tracking for website usability evaluation and implicit interaction. In: Proc. of the 15th Int. Conf. on WWW (2006)
5. Bach, S., Binder, A., Montavon, G., Klauschen, F., Müller, K.R., Samek, W.: On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. PLoS ONE **10**(7), 1–46 (2015). https://doi.org/10.1371/journal.pone.0130140
6. Cepeda, C., et al.: Mouse tracking measures and movement patterns with application for online surveys. In: Holzinger, A., Kieseberg, P., Tjoa, A.M., Weippl, E. (eds.) CD-MAKE 2018. LNCS, vol. 11015, pp. 28–42. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-99740-7_3
7. Chong, P., Elovici, Y., Binder, A.: User authentication based on mouse dynamics using deep neural networks: a comprehensive study. In: IEEE Transactions on Information Forensics and Security, vol. 15, pp. 1086–1101 (2020)
8. Chong, P., Ruff, L., Kloft, M., Binder, A.: Simple and effective prevention of mode collapse in deep one-class classification. CoRR abs/2001.08873 (2020)
9. Everitt, R., McOwan, P.: Java-based internet biometric authentication system. IEEE Trans. Pattern Anal. Mach. Intell. **25**(9), 1166–1172 (2003). https://doi.org/10.1109/TPAMI.2003.1227991
10. Fabi, K.: Layer-wise relevance propagation for pytorch (2021). https://github.com/KaiFabi/PyTorchRelevancePropagation
11. Feher, C., Elovici, Y., Moskovitch, R., Rokach, L., Schclar, A.: User identity verification via mouse dynamics. Inf. Sci. **201**, 19–36 (2012)
12. Fülöp, A., Kovács, L., Kurics, T., Windhager-Pokol, E.: Balabit mouse dynamics challenge data set (2016). https://github.com/balabit/Mouse-Dynamics-Challenge
13. Gajos, K., et al.: Computer mouse use captures ataxia and parkinsonism, enabling accurate measurement and detection. Mov. Disord. **35**(2), 354–358 (2019). https://doi.org/10.1002/mds.27915
14. Gamboa, H., Fred, A.: A behavioral biometric system based on human-computer interaction. In: Jain, A.K., Ratha, N.K. (eds.) Biometric Technology for Human Identification, vol. 5404, pp. 381–392. SPIE (2004)
15. Golan, I., El-Yaniv, R.: Deep anomaly detection using geometric transformations. In: Advances in Neural Information Processing Systems (NeurIPS) (2018)
16. Görnitz, N., Kloft, M., Rieck, K., Brefeld, U.: Toward supervised anomaly detection. J. Artif. Intell. Res. **46**, 235–262 (2013)
17. Haider, P., Chiarandini, L., Brefeld, U.: Discriminative clustering for market segmentation. In: Proc. of the ACM SIGKDD (2012)
18. Harilal, A., Toffalini, F., Castellanos, J., Guarnizo, J., Homoliak, I., Ochoa, M.: Twos: a dataset of malicious insider threat behavior based on a gamified competition. In: Proc. of MIST (MIST 2017), pp. 45–56. Association for Comp. Machinery, New York, NY, USA (2017)
19. Hendrycks, D., Mazeika, M., Dietterich, T.: Deep anomaly detection with outlier exposure. In: International Conference on Learning Representations (2019)
20. Hendrycks, D., Mazeika, M., Kadavath, S., Song, D.: Using self-supervised learning can improve model robustness and uncertainty. In: Wallach, H., Larochelle, H., Beygelzimer, A., d' Alché-Buc, F., Fox, E., Garnett, R. (eds.) Advances in Neural Information Processing Systems, vol. 32. Curran Associates, Inc. (2019)

21. Kaixin, W., Hongri, L., Bailing, W., Shujie, H., Jia, S.: A user authentication and identification model based on mouse dynamics. In: Proceedings of the 6th Int. Conf. on Information Engineering (ICIE 2017), Association for Computing Machinery, New York, NY, USA (2017)
22. Komarinski, P.: Automated fingerprint identification systems (AFIS). Elsevier (2005)
23. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Pereira, F., Burges, C., Bottou, L., Weinberger, K. (eds.) Advances in Neural Information Processing Systems, vol. 25. Curran Associates, Inc. (2012)
24. Lagun, D., Ageev, M., Guo, Q., Agichtein, E.: Discovering common motifs in cursor movement data for improving web search. In: Proc. of the 7th ACM Int. Conf. on Web Search and Data Mining, pp. 183–192. ACM (2014)
25. Matthiesen, J.J., Brefeld, U.: Assessing user behavior by mouse movements. In: Stephanidis, C., Antona, M. (eds.) HCII 2020. CCIS, vol. 1224, pp. 68–75. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-50726-8_9
26. Mondal, S., Bours, P.: A study on continuous authentication using a combination of keystroke and mouse biometrics. Neurocomputing **230**, 1–22 (2017)
27. Perera, P., Patel, V.M.: Deep transfer learning for multiple class novelty detection. CoRR abs/1903.02196 (2019)
28. Perera, P., Patel, V.M.: Learning deep features for one-class classification. IEEE Trans. Image Process. **28**(11), 5450–5463 (2019)
29. Rodden, K., Fu, X., Aula, A., Spiro, I.: Eye-mouse coordination patterns on web search results pages. In: Extended Abstracts on Human Factors in Computing Systems (CHI EA 2008), pp. 2997–3002. Assoc. for Computing Machinery, New York, NY, USA (2008)
30. Ruff, L., et al.: Deep one-class classification. In: ICML, pp. 4393–4402 (2018)
31. Ruff, L., Vandermeulen, R.A., Franks, B.J., Müller, K.R., Kloft, M.: Rethinking assumptions in deep anomaly detection (2020)
32. Ruff, L., et al.: Deep semi-supervised anomaly detection. In: Int. Conf. on Learning Representations (2020)
33. Schölkopf, B., Platt, J.C., Shawe-Taylor, J., Smola, A.J., Williamson, R.C.: Estimating the support of a high-dimensional distribution. Neural Comput. **13**(7), 1443–1471 (2001)
34. Shen, C., Cai, Z., Guan, X., Du, Y., Maxion, R.A.: User authentication through mouse dynamics. In: IEEE Trans. Inf. Forensics Security, vol. 8, pp. 16–30 (2013)
35. Shen, C., Cai, Z., Liu, X., Guan, X., Maxion, R.A.: MouseIdentity: modeling mouse-interaction behavior for a user verification system. IEEE Trans. Hum.-Mach. Syst. **46**(5), 734–748 (2016)
36. Shen, C., Cai, Z., Maxion, R.A., Xiang, G., Guan, X.: Comparing classification algorithm for mouse dynamics based user identification. In: 2012 IEEE Fifth International Conference on Biometrics: Theory, Applications and Systems (BTAS), pp. 61–66 (2012)
37. Tax, D.M., Duin, R.P.: Support vector data description. ML **54**(1), 45–66 (2004)
38. Tzafilkou, K., Protogeros, N.: Mouse behavioral patterns and keystroke dynamics in end-user development: what can they tell us about users' behavioral attributes? In: Computers in Human Behavior, vol. 83, pp. 288–305 (2018)
39. Wei, A., Zhao, Y., Cai, Z.: A deep learning approach to web bot detection using mouse behavioral biometrics. In: Sun, Z., He, R., Feng, J., Shan, S., Guo, Z. (eds.) CCBR 2019. LNCS, vol. 11818, pp. 388–395. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-31456-9_43
40. Yao, B., Ai, H., Lao, S.: Person-specific face recognition in unconstrained environments: a combination of offline and online learning. In: 2008 8th IEEE International Conference on Automatic Face & Gesture Recognition, pp. 1–8 (2008)

41. Zhang, L., Tan, S., Yang, J., Chen, Y.: VoiceLive: a phoneme localization based liveness detection for voice authentication on smartphones. In: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (CCS 2016), pp. 1080–1091. Association for Computing Machinery, New York, NY, USA (2016)
42. Zimmermann, P., Guttormsen, S., Danuser, B., Gomez, P.: Affective computing - measuring mood with mouse and keyboard. Int. J. Occup. Saf. Ergon. **9**, 539–551 (2003)