Journal of Modern Technology and Engineering
Vol.10, No.3, 2025, pp.159-172
https://doi.org/10.62476/jmte.103159

Jomard Publishing

MULTIMODAL BEHAVIOURAL BIOMETRIC PROFILING AND BASELINE MODELLING FOR CONTINUOUS AUTHENTICATION IN ENDPOINTS

K.N. Asgarov*

Azerbaijan Technical University, Baku, Azerbaijan

Abstract. Continuous authentication is an approach to endpoint security that enhances standard authentication methods. Traditional methods lose effectiveness after the session begins, as they do not monitor who continues to use the system. This work proposes a continuous authentication system based on modelling user behavioural profiles using multimodal telemetry data: mouse dynamics, keystroke dynamics, and application and graphical user interface usage statistics. The system models user behaviour by analyzing data from these modalities at one-minute intervals using a sliding window principle. Based on deep encoders, the system classifies user behaviour by comparing it with profiles built during endpoint baseline establishment. Experiments were conducted using data collected from participants who performed a chosen task for one hour. All user interactions were recorded based on the features proposed in this work. The results show that the proposed model successfully handles the task of classifying and identifying authenticated users and impostors with an overall accuracy of over 0.80, an average false acceptance rate of 0.24 and an average false rejection rate of 0.08.

Keywords: Anomaly detection, endpoint security, continuous authentication, behavioural biometrics.
Corresponding author: Kamran N. Asgarov, Department of Engineering Mathematics and Artificial Intelligence, Azerbaijan Technical University, PhD Candidate, Baku, Azerbaijan, Tel.: +994 50 541 54 19,
e-mail: *kamran.asgarov.n@student.aztu.edu.az*
Received: 16 September 2025; Revised: 22 November 2025; Accepted: 5 December 2025;
Published: 22 December 2025.

1 Introduction

Standard authentication systems in endpoint devices, such as desktop computers and laptops, rely on a one-time verification at system login. This initial verification may consist of checking a password, token, or biometric data such as fingerprints or facial recognition, and may also include two-factor or multi-factor authentication. However, after passing this initial verification, the system typically assumes that the authenticated user maintains control over the endpoint device throughout the entire session. But in reality, after the session is activated, this endpoint may be used by another unauthenticated user. This can occur both with the consent and supervision of the authenticated user, as in cases where colleagues temporarily exchange their devices or pass the device to family members for personal use, but also without the user's consent and supervision, such as when they step away from their computer for a short period. In such cases, users who have gained access to an open authenticated system can perform any actions on behalf of the person who has already once gone through the secure verification and login process. This disconnect between initial verification and actual use creates a critical security gap. Insider threats and account takeovers can remain undetected for extended periods of time.

Continuous authentication is designed to address this gap by extending the authentication

process beyond a single point in time, transforming it into an ongoing verification of the user's identity throughout the entire session. Continuous authentication incorporates all the same initial checks using standard authentication methods, but also continues to reassess the user after the session begins after the session begins using various methods. One such method is continuous facial recognition of the user, where the endpoint device's camera remains constantly active and monitors the user operating it, thereby periodically authenticating the user using facial recognition (Otta et al., 2022). There are also methods where camera monitoring is replaced by other physical devices such as smart rings, bracelets, or watches, in which the endpoint session remains active only if the wearable device is in close proximity to it (Mare et al., 2019).

Continuous authentication methods based on physical devices are not a complete solution to this problem due to numerous drawbacks. For example, using a camera for continuous authentication requires constant monitoring of the user. The user must sit in an almost unchanging position under good lighting so that their face is clearly visible to the camera. This creates inconvenience and causes psychological discomfort from constant surveillance. Methods based on verifying the proximity of a smart device also have disadvantages. The user may leave their device next to the computer, pass it to another person, or simply not want to wear an additional wearable device. Therefore, our research within the scope of this article examines the task of continuous authentication without using additional physical devices. Instead, we use only telemetry data from the endpoint device itself that the user operates.

Each person has a unique way of operating computers shaped by mouse control habits, application usage patterns, typing speed, and routine tasks. These interactions form their behavioural biometrics. Like any other biometric data, this information can identify users (Gupta et al., 2023). Behavioural biometrics create a unique user profile that is difficult to forge or copy, as it forms naturally during daily work. Using machine learning, continuous authentication systems analyze these patterns in real time and detect deviations that may indicate a user change. Behavioural biometrics is a non-intrusive verification method that requires no additional user actions and operates invisibly in the background.

2 Related Works

In recent years, behavioural biometrics and continuous authentication have become active areas of research. Numerous studies examine continuous authentication systems based on behavioural biometrics, using data from user interactions with devices. These studies propose various approaches to using telemetry data for modelling user profiles (Baig & Eskeland, 2021). For example, different studies extract different features from mouse cursor movement dynamics data or smartphone keystroke dynamics. There is no single correct feature or set of features that unambiguously model a user profile. Therefore, even studies that propose extracting different features from the same telemetry sources are of great value.

Mouse movement dynamics is one of the most important indicators of user behaviour when working with desktop computers and laptops. Users perform many tasks using only the mouse cursor, so they spend most of their time interacting with the mouse. It is not surprising that there are numerous studies on modelling user behaviour based on mouse telemetry data. For example, Shen et al. (2013) proposed a method based on distance-measurement and eigenspace-transformation techniques followed by the application of a one-class learning algorithm, in which they managed to achieve an 8.74% false-acceptance rate. Subsequent works and specialized research in the field of mouse dynamics biometrics have made it possible to further improve feature modelling methods and confirmed the suitability of mouse signals for continuous and session-based verification (Khan et al., 2024).

Keystroke dynamics is another important source of user behavioural data, as the keyboard is the second most commonly used input tool on endpoint devices. Even classic studies, such as the work of Monrose & Rubin (2000), have shown that temporal patterns between key presses

and releases can uniquely characterize users and contribute to both their identification and verification. Recent research in this area confirms the findings of classical works, improving methods and adding new features and patterns that can be extracted from keystroke dynamics (Ceker & Upadhyaya, 2016).

Among research on behavioural profiling and continuous authentication, there are also works with a multimodal approach, where not just one modality is used but several, which are combined for more accurate profiling. For example, only mouse dynamics or keystroke dynamics alone may be insufficient, so researchers combine them. Subash et al. (2025) examined the adaptability of systems using both keystroke and mouse movement data in biometric system tasks. They also investigated how profiling data obtained from multiple devices, such as smartphones and personal computers, can be combined, highlighting the benefits of cross-device behavioural modelling.

3 The Main Part

The main problem addressed in this work is that standard authentication systems require user verification using passwords, tokens, or other methods only at the initial stage, leaving subsequent user activity largely unverified. After the first login, the system assumes that the endpoint device is being used by the same person who logged in initially.

Our solution proposes a continuous authentication system that models user behaviour based on multimodal data. This data includes mouse movement dynamics, keyboard dynamics, and graphical user interface application usage. After modelling the profile of each authenticated user of the endpoint device, the system continuously monitors the current user and classifies them, determining which of the authenticated users is in the system.

If the classified user differs from the one who is active in the session, the system requires re-authentication. There may also be cases where the user operating the computer is not any of the authenticated users. In such cases, the user is redirected to the login page, and the incident is recorded as an anomaly and a potential cyberattack attempt. If the system's classification matches the data of the active session user, the system continues monitoring without intervention.

3.1 Data Collection and Behavioural Modalities

To build the behavioural profile of authenticated system users, it is necessary to collect data about their work during active sessions, as shown in Figure 1. Figure 1 shows how two users work during their active sessions. First, we define a time period that should encompass the work of all authenticated users to build their behavioural profiles. We called this period the "Training Period". During this period, we collect user data using several modalities of data sources about user interactions with the endpoint device. The duration of the training period depends on the company where the system is being installed. Different companies may have different work cycles. For example, some users have routine work that is the same every day, while others have work that repeats throughout the week but varies by day.

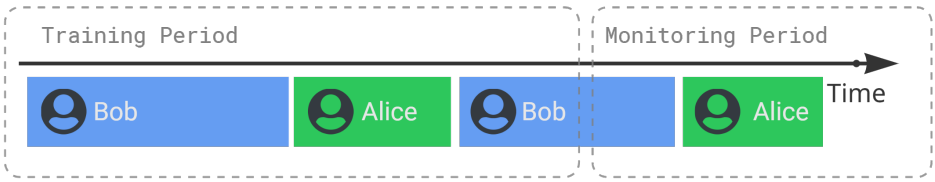


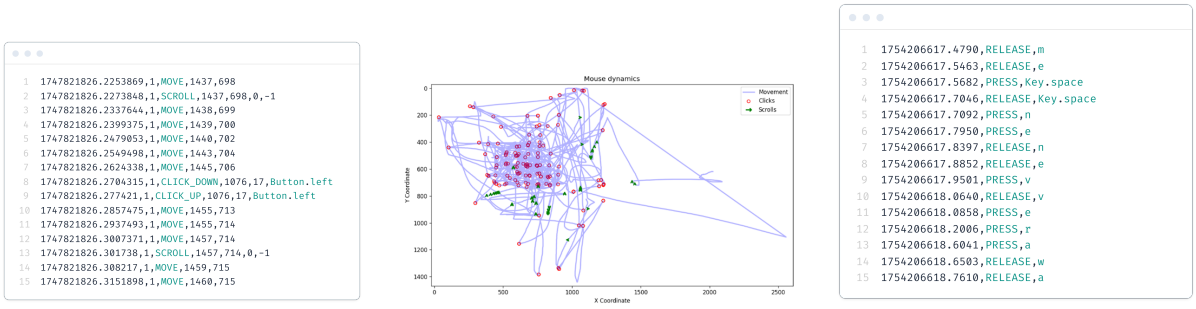
Figure 1: Timeline of multiple user sessions.

At the end of the "Training Period", we train a deep learning model on the collected data. After this, the system is ready to monitor user behaviour and launch the continuous authentica-

tion process. At this second stage, we compare the behaviour of the active session user with the profiles formed in the first period. Thus, at any given moment, the system classifies the active user. It determines whether they are one of the previously authenticated users, or an impostor if their behaviour does not match any of the previously profiled patterns.

3.1.1 Collected Interaction Features

During training, the system collects data from three modalities: mouse, keyboard, and graphical user interface applications. However, all three modalities are complex to work with directly in their raw form. For example, Figure 2 shows the data we receive from the endpoint device's operating system. The operating system transmits this data in the form of events. Figure 2a shows an example of a list of mouse events, Figure 2c shows an example of keyboard events, and Figure 2b shows a visualization of mouse events. These events are transmitted as they occur in the system. Each mouse movement and each press or release of a button on the mouse or keyboard represents individual events.



(a) mouse events **(b)** mouse events visualised **(c)** keystrokes events

Figure 2: Collected modalities data in raw form.

Working with such time series data and developing a machine learning or deep learning model is a very complex task. Therefore, instead of working with this data in its raw time series form, we transform it into a convenient format and extract features that may be useful. The main idea of data collection is the sliding window principle, which is shown in Figure 3. We select a specific interval Δt for the sliding window (within this work, this interval equals one minute), during which we collect data from all sources of the three modalities, and then extract features that we consider useful and important. Thus, we store in one telemetry row all the information about what happened during that interval.

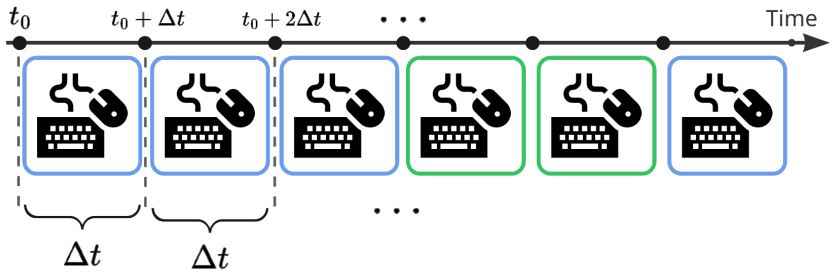


Figure 3: Sliding window principle.

The data obtained from mouse dynamics telemetry was transformed into two modalities. The first modality is a heatmap of mouse cursor activity with dimensions of 16×9 pixels. The value of each pixel ranges from 0 to 1 depending on cursor control activity in that zone, where 1 indicates maximum activity and 0 indicates minimum activity as visualised in Figure 4.

Representing cursor activity as a heatmap simplifies working with the data and allows the use of various machine learning algorithms, including those based on convolutional neural networks (Asgarov, 2025a). The second modality includes seven features that characterize mouse control: average mouse movement speed, mean and standard deviations of pauses between clicks, mean and standard deviations of left mouse button hold duration, as well as mean and standard deviations of right mouse button hold duration. As a result, eight features were obtained: a heatmap containing 144 pixels, and seven numerical features. All these features reflect user behaviour when controlling the mouse during the sliding window interval.

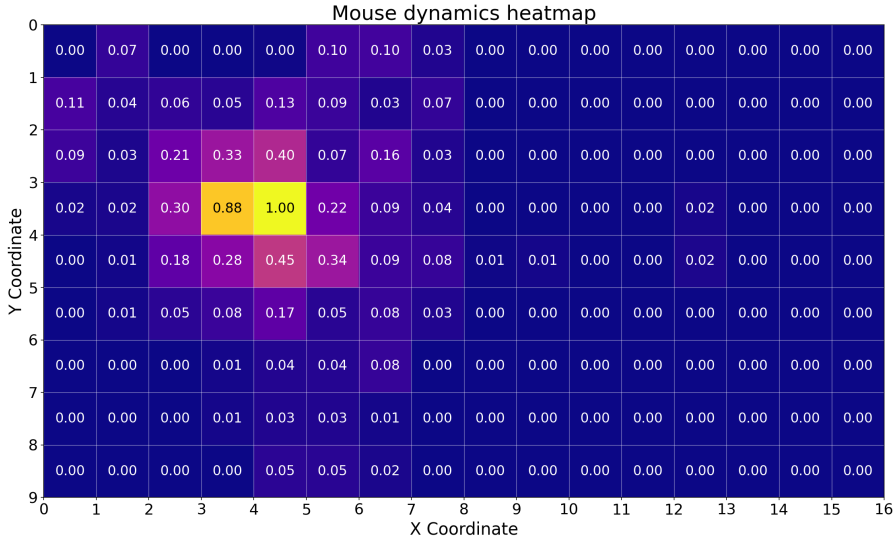


Figure 4: Mouse activity heatmap representation.

The next source of telemetry data is keystroke dynamics. This modality helps profile user behaviour when interacting with the endpoint device. We collect a list of all keystroke events that occurred during the sliding window interval and extract the following features from them: the number of keystrokes, mean and standard deviation of key hold time, mean and standard deviation of intervals between consecutive key presses, mean and standard deviation of intervals between releasing one key and pressing the next, mean and standard deviation of intervals between key releases, as well as the number of keystrokes per minute. In total, we extract 9 features from the user's keystroke dynamics.

The final modality is graphical user interface application usage, which tracks which applications the user works with and how they switch between them. During each sliding window interval, we extract four features: total focus time across all applications, the number of application switches, the number of unique applications used, and the total count of window events.

3.1.2 Technical Implementation and Resource Consumption

A program was written in the Python programming language to collect user interaction data with the endpoint device, using the `pynput` library to collect mouse and keyboard dynamics events, as well as the `pywin32` library to collect graphical interface and application events. The source code of the program, like all other programs and datasets used in this work, is located in the author's public GitHub repository (Asgarov GitHub Repository, 2025).

In order to ensure that the data collection component does not introduce a noticeable overhead on the endpoint, the implementation was instrumented to monitor its own resource consumption. During operation, the Python agent periodically samples its CPU and memory usage using the `psutil` library and measures the time required to aggregate raw mouse, keyboard and graphical user interface events into one-minute windows and to write the corresponding feature

vectors to disk.

Table 1 summarizes the resource consumption of the telemetry agent in a representative experimental run. The collector maintains low CPU and memory usage, and the per-window storage footprint of the extracted features remains modest. This confirms that continuous data collection runs unobtrusively in the background without degrading the user experience on the endpoint.

Table 1: Resource consumption of the behavioural telemetry collection agent.

Metric	Value
Average window processing time	3.43 ms
Maximum window processing time	5.92 ms
Average CPU utilisation	0.11%
Peak CPU utilisation	3.10%
Average RAM usage	23.09 MB
Peak RAM usage	23.17 MB
Average storage per window	0.96 kB

3.2 Behavioural Profiling Modelling

After collecting interaction data from all authenticated users of the endpoint device, the system applies the sliding window principle and divides the collected dataset into two subsets: 80% and 20%. 80% of the data is used for training the deep learning model, while 20% is used for optimizing coefficients to better distinguish between authenticated users. After model training, which takes only a few seconds due to the model's lightweight nature, the system enters continuous monitoring mode. In this mode, the model classifies each interaction dataset, which is again collected using the sliding window principle, just as during the training data collection phase. This data is fed to the model, which classifies the user and determines whether the observed behaviour belongs to one of the authenticated users, or whether it differs completely from all previously modeled behavioural profiles. In the latter case, the model classifies the dataset as "unknown" or "impostor", which is also considered an anomaly.

3.2.1 Encoder Networks for Each Modality

Let $\mathcal{M} = \{\text{heatmap}, \text{mouse}, \text{keys}, \text{gui}\}$ denote the four modalities. For every sliding window t and modality $m \in \mathcal{M}$, the program constructs a feature input $\mathbf{x}_t^{(m)}$ and feeds it to a modality-specific encoder network $f_\theta^{(m)}$. Each encoder outputs a 32-dimensional ℓ_2-normalised embedding

$$\mathbf{z}_t^{(m)} = f_\theta^{(m)}\left(\mathbf{x}_t^{(m)}\right) \in \mathbb{R}^{32}, \qquad \left\|\mathbf{z}_t^{(m)}\right\|_2 = 1.$$

For the mouse heatmap modality the input is a normalised matrix $\mathbf{H}_t \in [0,1]^{9\times16}$, which is reshaped to a tensor of shape $(1, 9, 16)$ and processed by a compact convolutional neural network. The heatmap encoder $f_\theta^{(\text{heatmap})}$ has the following architecture:

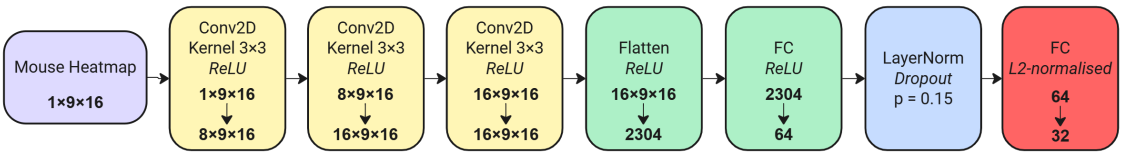


Figure 5: Convolutional encoder for the mouse activity heatmap.

All convolutional layers use 3×3 kernels with unit padding to preserve the spatial resolution of 9×16. Layer normalisation and dropout in the penultimate layer help to stabilise training and reduce overfitting, while the final 32-dimensional latent space captures spatial cursor-activity patterns in a compact form suitable for similarity-based classification.

The three tabular modalities are processed by feed-forward multilayer perceptrons (MLPs) with identical structure. For each modality m the input is a vector $\mathbf{x}_t^{(m)} \in \mathbb{R}^{d_m}$, where $d_{\text{mouse}} = 7$, $d_{\text{keys}} = 9$, and $d_{\text{gui}} = 4$ in this work. The encoder $f_\theta^{(m)}$ has the following architecture:

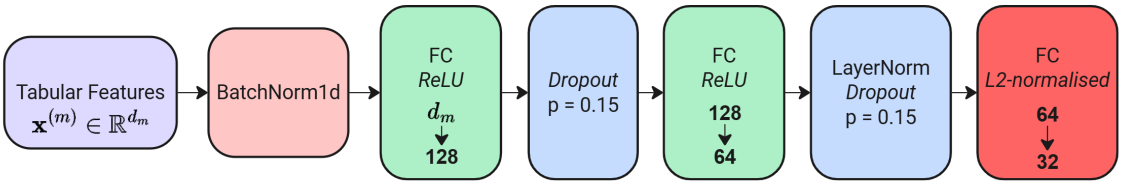


Figure 6: Shared MLP encoder architecture for tabular modalities .

Batch normalisation reduces covariate shift between mini-batches, and layer normalisation in the second hidden layer improves robustness when the feature dimensionality is small. The final ℓ_2-normalisation ensures that all embeddings lie on the unit hypersphere, which makes cosine similarity and Euclidean distance equivalent and simplifies prototype-based classification.

For each modality, if at least two enrolled users are present, the encoder is augmented with a linear classification head $g_\phi^{(m)} : \mathbb{R}^{32} \to \mathbb{R}^K$, where K is the number of known users. The head applies a simple affine transformation

$$\mathbf{o}_t^{(m)} = g_\phi^{(m)}\left(\mathbf{z}_t^{(m)}\right) = W^{(m)}\mathbf{z}_t^{(m)} + \mathbf{b}^{(m)},$$

and the resulting logits are used to compute a cross-entropy loss over the known identities. In the special case of monitoring a single user, the program omits this head and instead introduces a learnable centre vector $\mathbf{c}^{(m)} \in \mathbb{R}^{32}$ on the unit hypersphere; the encoder is then trained to maximise the cosine similarity between embeddings and this centre, yielding a tight, user-specific cluster in latent space.

3.2.2 Training Procedure

For modalities with at least two known authenticated users the training objective is the standard cross-entropy loss over the enrolled identities. For an input $\mathbf{x}_t^{(m)}$ with label $y_t \in \{1, \ldots, K\}$, the model produces logits $\mathbf{o}_t^{(m)}$ and class probabilities

$$p^{(m)}(y \mid \mathbf{x}_t^{(m)}) = \frac{\exp(o_{t,y}^{(m)})}{\sum_{k=1}^{K} \exp(o_{t,k}^{(m)})},$$

and the loss over a mini-batch $\mathcal{B}$ is

$$\mathcal{L}_{\text{CE}}^{(m)} = -\frac{1}{|\mathcal{B}|} \sum_{t \in \mathcal{B}} \log p^{(m)}(y_t \mid \mathbf{x}_t^{(m)}).$$

For modalities with only one known user the loss becomes a cosine centre loss,

$$\mathcal{L}_{\text{centre}}^{(m)} = \frac{1}{|\mathcal{B}|} \sum_{t \in \mathcal{B}} \left(1 - \cos(\mathbf{z}_t^{(m)}, \mathbf{c}^{(m)})\right),$$

where both the encoder parameters and the centre vector $\mathbf{c}^{(m)}$ are learned jointly. The centre is ℓ_2-normalised after each update to keep it on the unit sphere.

Training is implemented in PyTorch using the Adam optimiser with learning rate $\eta = 10^{-3}$ and weight decay $\lambda = 10^{-4}$ for ℓ_2 regularisation. Gradients are clipped to a maximum norm of 1.0, and mini-batches of windows are sampled uniformly from the pooled training data of all known users. After each epoch the program evaluates the loss on the optimisation subset for the corresponding modality. The model parameters that achieve the lowest optimisation loss are retained, and early stopping is triggered if no improvement is observed for several successive epochs.

3.2.3 Prototype-Based Open-Set Classification

After training, the encoder networks are used purely as feature extractors, classification heads are discarded, and the continuous authentication logic is implemented via prototypes and open-set thresholds. For each modality m and known user u_k the program computes embeddings for all windows in the training subset:

$$\mathcal{Z}_k^{(m)} = \left\{ \mathbf{z}_t^{(m)} : t \in \mathcal{D}_k^{\text{train}} \right\}.$$

The class prototype for user u_k in modality m is defined as the mean of these embeddings, re-normalised to unit length:

$$\boldsymbol{\mu}_k^{(m)} = \frac{\frac{1}{|\mathcal{Z}_k^{(m)}|} \sum_{\mathbf{z} \in \mathcal{Z}_k^{(m)}} \mathbf{z}}{\left\| \frac{1}{|\mathcal{Z}_k^{(m)}|} \sum_{\mathbf{z} \in \mathcal{Z}_k^{(m)}} \mathbf{z} \right\|_2}.$$

This prototype represents the typical embedding of user u_k for modality m (Geng et al., 2020). To support open-set rejection, the program calibrates user-specific distance thresholds from the optimisation subset. For each $t \in \mathcal{D}_k^{\text{opt}}$ and modality m, the cosine similarity between the embedding and the prototype is computed as

$$s_{k,t}^{(m)} = \cos\big(\mathbf{z}_t^{(m)}, \boldsymbol{\mu}_k^{(m)}\big) = \frac{\mathbf{z}_t^{(m)} \cdot \boldsymbol{\mu}_k^{(m)}}{\|\mathbf{z}_t^{(m)}\|_2 \, \|\boldsymbol{\mu}_k^{(m)}\|_2},$$

and the cosine distance is defined as $d_{k,t}^{(m)} = 1 - s_{k,t}^{(m)}$. The threshold $\tau_k^{(m)}$ is then chosen as a high quantile q of the distance distribution:

$$\tau_k^{(m)} = \text{Quantile}_q \left(\{ d_{k,t}^{(m)} : t \in \mathcal{D}_k^{\text{opt}} \} \right).$$

Intuitively, $\tau_k^{(m)}$ defines a modality-specific acceptance region around the prototype $\boldsymbol{\mu}_k^{(m)}$ that covers the majority of typical behaviour for user u_k. During continuous monitoring, each new window is encoded to $\mathbf{z}^{(m)}$ and compared to all user prototypes for that modality. The system computes the cosine similarity to every prototype and selects the nearest one:

$$k^{*(m)} = \arg\max_k \cos\big(\mathbf{z}^{(m)}, \boldsymbol{\mu}_k^{(m)}\big).$$

Let $d^{(m)} = 1 - \cos(\mathbf{z}^{(m)}, \boldsymbol{\mu}_{k^{*(m)}}^{(m)})$ be the distance to this best-matching prototype. The open-set decision for modality m is

$$\hat{y}^{(m)} = \begin{cases} u_{k^{*(m)}} & \text{if } d^{(m)} \le \tau_{k^{*(m)}}^{(m)}, \\ \text{Unknown} & \text{otherwise.} \end{cases}$$

This decision rule is applied online to each incoming sliding window, providing a continuous stream of identity predictions and unknown-user detections.

3.2.4 Multimodal Late Fusion

Although each modality can independently perform open-set classification, the final system combines them through a late fusion scheme that operates directly in the similarity space (Fridman et al., 2014). The motivation for this design choice is supported by prior empirical comparisons between unimodal and multimodal approaches. In our previous study, unimodal and multimodal deep learning models were evaluated on the same endpoint interaction modalities, including mouse dynamics, keystroke dynamics, and graphical user interface activity (Asgarov, 2025b). The results demonstrated that multimodal models consistently outperformed unimodal counterparts, achieving higher classification accuracy and F_1 scores, which confirms that combining complementary behavioural signals provides more robust user profiling than relying on any single modality alone.

For every modality m the program evaluates the encoder in a closed-set setting on the optimisation subset by assigning each optimisation embedding to the nearest prototype and computing the macro-averaged F_1-score $F_{\text{opt}}^{(m)}$ across all enrolled users. These scores are then normalised to obtain non-negative fusion weights w_m that sum to one:

$$w_m = \frac{F_{\text{opt}}^{(m)}}{\sum_{m' \in \mathcal{M}} F_{\text{opt}}^{(m')}}, \qquad \sum_{m \in \mathcal{M}} w_m = 1.$$

Given a monitoring window, each modality produces a vector of cosine similarities $s^{(m)}(k)$ between its embedding and the prototypes $\boldsymbol{\mu}_k^{(m)}$ of the known users. The fused similarity for user u_k is defined as the weighted sum

$$s^{\text{fused}}(k) = \sum_{m \in \mathcal{M}} w_m \, s^{(m)}(k),$$

and the candidate user is chosen as

$$k^* = \arg\max_k s^{\text{fused}}(k).$$

The corresponding fused distance is $d^{\text{fused}} = 1 - s^{\text{fused}}(k^*)$. To preserve the open-set interpretation in the multimodal setting, the per-modality thresholds $\tau_k^{(m)}$ are also combined using the same weights, yielding a fused threshold for user u_k:

$$\tau_k^{\text{fused}} = \sum_{m \in \mathcal{M}} w_m \, \tau_k^{(m)}.$$

The final multimodal decision is therefore

$$\hat{y} = \begin{cases} u_{k^*} & \text{if } d^{\text{fused}} \le \tau_{k^*}^{\text{fused}}, \\ \text{Unknown} & \text{otherwise.} \end{cases}$$

In the continuous authentication scenario this multimodal decision is evaluated for every sliding window in real time. If the predicted identity differs from the currently logged-in user, or if the activity is classified as "Unknown", the system can either request an additional authentication step or lock the session and record an anomaly event.

4 Results

As part of the author's PhD research, experiments were conducted with 12 participants who were asked to choose a task and complete it over the course of one hour. All participants selected a task for themselves and performed it for 60 minutes on a personal computer. Throughout this

time, the data collection program described in section 3.1.1 ran in the background. The data was collected, transformed, relevant features were extracted, and then written to `.csv` files. Each `.csv` file contains 60 rows, and each row encompasses 164 features. These features include a mouse activity heatmap, mouse control usage characteristics, keyboard interaction features, as well as features of working with applications through the graphical user interface. The collected data from all 12 participants is available in the author's public GitHub repository (Asgarov GitHub Repository, 2025).

4.1 Experimental Setup

Using data from the `.csv` files, an experiment was designed to test and evaluate the proposed continuous authentication method. Three experiments were conducted with different numbers of known users: 1, 2, and 3 respectively. The model was trained on data from these known users, who in the continuous authentication task are the authenticated users whose data was collected during the training period.

Since the dataset contained 60 rows of data for each user, covering 60 minutes, we divided the known users' data into three parts, which are depicted in Figure 7. The data from the first 45 minutes was used for training: 36 minutes were fed to the model for training, and 9 minutes were used for method optimization. The remaining 15 minutes were used in subsequent tests to evaluate the classification accuracy of the constructed model.

Figure 7: Data distribution of known user for experimental setup.

We evaluate three operating scenarios with different numbers of enrolled (known) users, $N \in \{1, 2, 3\}$. For a given scenario, a subset of N participants is treated as enrolled users whose profiles are modelled during the training period. The remaining participants are treated as non-enrolled users and are used to test open-set rejection (the "Unknown" outcome). For each N, we repeat the experiment across multiple different selections of enrolled users and report the mean performance across runs.

In this evaluation, "impostor" samples correspond to interaction windows produced by other participants who are not enrolled in the current run. This corresponds to a standard *zero-effort* impostor setting, which is commonly used as a baseline in behavioural-biometric authentication studies. Similar limitations are commonly highlighted in recent continuous-authentication studies and surveys (Aversano et al., 2021; Uslu et al., 2023). More challenging attacker models, such as active mimicry attempts including automated or assisted imitation and synthetic generated interaction traces (Yu et al., 2023), are outside the scope of the present dataset and will be addressed in future work.

Finally, we note that the current dataset is sufficient to assess short-term separability of behavioural profiles under controlled conditions, but it cannot fully capture long-term behavioural drift or broader diversity in work patterns. Collecting a larger longitudinal dataset is an important next step for improving external validity for future work.

4.2 Method Evaluation

To evaluate the proposed continuous authentication method, the following metrics were used:

Fraction of correctly classified windows:

$$\text{Overall Accuracy} = \frac{\text{number of correct predictions}}{\text{total number of test samples}}.$$

Mean recall over all classes $\mathcal{C}$, including *Unknown*:

$$\text{Balanced Accuracy} = \frac{1}{|\mathcal{C}|} \sum_{c \in \mathcal{C}} \text{TPR}_c.$$

Genuine windows from known users rejected as *Unknown*:

$$\text{FRR} = \frac{N_{\text{FR}}}{N_{\text{genuine}}}.$$

Impostor windows from unknown users accepted as known:

$$\text{FAR} = \frac{N_{\text{FA}}}{N_{\text{impostor}}}.$$

ROC–AUC (area under the ROC curve) is computed for the binary task "known user" vs. "unknown user" using the fused similarity score as the decision variable.

Table 2 shows the experimental results according to the method's evaluation metrics. The results demonstrate that the method achieved an overall accuracy of 0.80 or higher in all cases—with one, two, and three known users. FRR remains relatively low in all cases, which indicates successful identification of known users by the system. The FAR value, ranging from 0.22 to 0.27, indicates that some impostor users were incorrectly identified as authenticated users. This reflects the inherent trade-off in continuous authentication between usability (low FRR) and security (low FAR), which in practice can be adjusted by tuning the open-set rejection threshold.

Table 2: Average performance of the continuous authentication method for different numbers of known users N.

Metric	$N = 1$	$N = 2$	$N = 3$
Overall accuracy	0.84	0.81	0.80
Balanced accuracy	0.87	0.84	0.84
FRR	0.04	0.11	0.10
FAR	0.22	0.24	0.27
ROC–AUC	0.92	0.91	0.87

Finally, increasing the number of known users from $N = 1$ to $N = 3$ slightly reduces overall accuracy and ROC-AUC, suggesting that classification and open-set rejection become more challenging as more behavioural profiles are modeled in the same embedding space. Nonetheless, the method maintains stable balanced accuracy around 0.84, indicating that performance remains relatively even across classes, including the "Unknown" class.

5 Discussion

Recent research emphasises that continuous authentication can be applied at endpoint usage behavioural data. These data form the behavioural biometrics of a human interacting with a computer (Baig & Eskeland, 2021; Gupta et al., 2023; Uslu et al., 2023). Studies show that mouse dynamics and keystroke dynamics remain among the most informative desktop modalities for modelling user behaviour and subsequent user classification (Khan et al., 2024; Subash et al., 2025). These data are collected passively in the background, using minimal endpoint resources. Our results are consistent with these findings and show that short-term behavioural profiles

can be reliably separated in an open-set scenario. The proposed system achieved an overall accuracy above 0.80 with a low FRR (averaging 0.08), whilst the FAR remained moderate (averaging 0.24). This reflects the typical trade-off between security and usability highlighted in the continuous authentication literature (Baig & Eskeland, 2021).

The main contribution of this work is the use of a multimodal approach to modelling user behaviour when working with an endpoint device, followed by the fusion of these modalities. Our previous research comparing multimodal and single-modal approaches to user behaviour modelling showed that multimodal approaches are more effective. This is supported by other studies, for example the work of Subash et al. (2025), which notes that individual modalities can be noisy or periodically absent, and that combining complementary signals generally allows for the creation of more robust user profiles. Another distinguishing feature of this work is the extraction of features from each modality and the elimination of a major complexity in developing machine learning methods: working with time series. For example, Li et al. (2024) used an autoencoder with long short-term memory to process time series data, which can require a large amount of memory. Our method deliberately uses a lightweight and efficient fusion strategy, where modality weights are derived from validation results and applied directly in the similarity space. This allows the system to maintain open-set recognition capability whilst simultaneously improving stability in the presence of multiple enrolled users. At the same time, the proposed approach remains compatible with single-modal operation, which is important in practice when some sensors are unavailable or limited.

From a practical perspective, the proposed method can be integrated into endpoint security as a background continuous-authentication layer that does not require additional devices such as cameras or wearables and does not require explicit user actions. The system outputs a per-window identity decision and an "Unknown" rejection outcome, which can be mapped to standard response policies such as step-up re-authentication, session lock, or incident logging, depending on organisational risk tolerance. The presented evaluation focuses on a standard zero-effort impostor setting, recent work has shown that stronger attacker models such as mimicry attempts can challenge keystroke-based systems (Yu et al., 2023), therefore future work should expand the dataset longitudinally and include adversarial evaluations to better quantify robustness under targeted attacks.

6 Conclusion

This work examined the task of continuous authentication on endpoint devices, which consists of constantly verifying that the user currently controlling the device is the same user who completed the login process at the beginning of the session. As a solution, we chose to model the user's behavioural profile using a multimodal source of interaction data with the endpoint device, which includes mouse dynamics, keystroke dynamics, and graphical user interface interaction data. To model the behavioural profile, encoders were used for each modality, with later mapping the collected features into a 32-dimensional ℓ_2-normalized embedding space with a subsequent fusion mechanism. An experiment was then conducted with data collected from 12 participants who performed a chosen task for 60 minutes. Throughout this time, data was recorded to a `.csv` file. Experiments were conducted with 1, 2, and 3 known participants. The constructed model had to classify and identify known users, as well as detect unauthenticated users, that is, unknown impostor users. The experimental results showed an overall accuracy of 0.80 in all three experiments, and also demonstrated good FRR results, which reflects the model's ability to correctly identify known users.

References

Asgarov, K.N. (2025). User Behaviour Endpoint Baseline Deviation Using Mouse Dynamics Heatmaps. *The 10th Republican Scientific and Technical Conference of Students and Young Researchers Advanced Technologies and Innovations*, 299-303.

Asgarov, K.N. (2025). Endpoint Behavior Modelling from User Interaction Metrics via Unsupervised Deep Learning for Anomaly Detection. *2025 IEEE 19th International Conference on Application of Information and Communication Technologies (AICT)*, Al Ain, United Arab Emirates, pp. 1–6. https://doi.org/10.1109/AICT67988.2025.11268628

Asgarov, K.N. (2025). Behavioural Biometric Profiling. *GitHub Repository* https://github.com/tivole/anomaly-behavioural-biometric-profiling

Aversano, L., Bernardi, M.L., Cimitile, M., & Pecori, R. (2021). A deep neural networks ensemble to enhance user authentication using keystroke dynamics. *PeerJ Computer Science*. https://doi.org/10.7717/peerj-cs.525

Baig, A.F., Eskeland, S. (2021). Security, privacy, and usability in continuous authentication: A survey. *Sensors*, 21(17), 5967. https://doi.org/10.3390/s21175967

Ceker, H., Upadhyaya, S. (2016). User authentication with keystroke dynamics in long-text data. *IEEE 8th International Conference on Biometrics Theory, Applications and Systems (BTAS)*, 1-6. https://doi.org/10.1109/btas.2016.7791182

Fridman, L., Stolerman, A., Acharya, S., Brennan, P., Juola, P., Greenstadt, R., & Kam, M. (2014). Multi-modal decision fusion for continuous authentication. *Computers & Electrical Engineering.*, 41, 142–156. https://doi.org/10.1016/j.compeleceng.2014.10.018

Geng, C., Huang, S., & Chen, S. (2020). Recent advances in open set Recognition: a survey. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *43*(10), 3614–3631. https://doi.org/10.1109/tpami.2020.2981604

Gupta, S., Maple, C., Crispo, B., Raja, K., Yautsiukhin, A., & Martinelli, F. (2023). A survey of human-computer interaction (HCI) & natural habits-based behavioural biometric modalities for user recognition schemes. In *Pattern Recognition*, 139, 109453. https://doi.org/10.1016/j.patcog.2023.109453

Khan, S., Devlen, C., Manno, M., & Hou, D. (2024). Mouse dynamics behavioral biometrics: A survey. *ACM Computing Surveys*, *56*(6), 154:1–154:33. https://doi.org/10.1145/3640311

Li, J., Yi, Q., Lim, M. K., Yi, S., Zhu, P., & Huang, X. (2024). MBBFAUTH: Multimodal Behavioral Biometrics Fusion for Continuous Authentication on Non-Portable Devices. *IEEE Transactions on Information Forensics and Security,*, 19, pp. 10000–10015. https://doi.org/10.1109/tifs.2024.3480363

Mare, S., Rawassizadeh, R., Peterson, R., & Kotz, D. (2019). Continuous Smartphone Authentication using Wristbands. In *Workshop on Usable Security (USEC)*. https://doi.org/10.14722/usec.2019.23013

Monrose, F., Rubin, A.D. (2000). Keystroke dynamics as a biometric for authentication. *Future Generation Computer Systems*, *16*(4), 351–359. https://doi.org/10.1016/S0167-739X(99)00059-X

Otta, S.P., Kolipara, S., Malhotra, V.K., Singh, A.R., Panda, S. & Hota, C. (2022). Continuous Cloud User Authentication By Efficient Facial Recognition. In *5th International Conference on Computational Intelligence and Networks (CINE)* (pp. 01-06). https://doi.org/10.1109/CINE56307.2022.10037567

Shen, C., Cai, Z., Guan, X., Du, Y., & Maxion, R.A. (2013). User authentication through mouse dynamics. *IEEE Transactions on Information Forensics and Security*, 8(1), 16–30. https://doi.org/10.1109/TIFS.2012.2223677

Subash, A., Song, I., Lee, I., & Lee, K. (2025). Adaptability of current keystroke and mouse behavioral biometric systems: A survey. *Computers & Security*, 160, 104731. https://doi.org/10.1016/j.cose.2025.104731

Uslu, U., İncel, Ö.D., & Alptekin, G.I. (2023). Evaluation of deep learning models for continuous user authentication based on behavioral biometrics. *Procedia Computer Science*, 225, 1272-1281. https://doi.org/10.1016/j.procs.2023.10.115

Yu, R., Kizilkaya, B., Meng, Z., Li, E., Zhao, G. & Imran, M. (2023). Robot Mimicry Attack on Keystroke-Dynamics User Identification and Authentication System. *2023 IEEE International Conference on Robotics and Automation (ICRA)*, London, UK, pp. 9879-9884. https://doi.org/10.1109/ICRA48891.2023.10161423