Article

Not peer-reviewed version

Continuous Authentication in the Digital Age: An Analysis of Reinforcement Learning and Behavioral Biometrics

Priya Bansal and Abdelkader Ouda *

Posted Date: 14 June 2023

doi: 10.20944/preprints202306.1005.v1

Keywords: Continuous Authentication; Static Authentication; Behavioral Biometrics; Reinforcement Learning (RL); Q-learning; Keystroke Dynamics; Anomaly Detection; Machine Learning; Supervised Learning; User Authentication; Identification

Article

Continuous Authentication in the Digital Age: An Analysis of Reinforcement Learning and Behavioral Biometrics

Priya Bansal [1] and Abdelkader Ouda [1]*

Department of Electrical and Computer Engineering, Western University, London, Ontario, N6A 5B9, Canada; pbansa2@uwo.ca

* Correspondence: aouda@uwo.ca

Abstract: This article focuses on developing a continuous authentication system using behavioral biometrics to recognize users accessing computing devices. The user's distinct behavioral biometric is captured through keystroke dynamics, and reward-based reinforcement learning (RL) ideas are applied to recognize them throughout the session. The suggested system adds an extra layer of security to traditional authentication methods, forming a robust continuous authentication system that can be added to static authentication systems. The methodology involves training a RL model to detect unusual user typing patterns and flag suspicious activity. Each user has an agent trained on their historical data, which is preprocessed and used to create episodes for the agent to learn from. The environment involves fetching observations and randomly corrupting them to learn out-of-order behavior. The observation vector includes both running features and summary features. The re-ward function is binary and minimalistic. The Principal Component Analysis (PCA) model is used to encode the running features, and the Double Deep Q-Network (DDQN) algorithm with a fully connected neural network is used as the policy net. The evaluation achieved an average training accuracy and EER (equal error rate) of 94.7% and 0.0126 and test accuracy and ERR of 81.06% and 0.0323 for all users when the number of encoder features was increased. Therefore, it is concluded that by continuously learning and adapting to changing behavior patterns, this approach can provide more secure and personalized authentication, lowering the possibility of unauthorized access and cyberattacks. Overall, the use of reinforcement learning and behavioral biometrics for continuous authentication has the potential to significantly enhance security in the digital age and are effective in identifying each user.

Keywords: Continuous Authentication; Static Authentication; Behavioral Biometrics; Reinforcement Learning (RL); Q-learning; Keystroke Dynamics; Anomaly Detection; Machine Learning; Supervised Learning; User Authentication; Identification.

1. Introduction

In today's fast-paced business environment, traditional methods of security are becoming increasingly inadequate [1]. With the rise of sophisticated cyberattacks, it has become easier for hackers to gain access to systems and steal user identities. Even if an organization has a strong security system in place, employees may still inadvertently compromise security by sharing passwords or digital keys [2,3]. As a result, businesses are facing significant losses due to weakened security systems that rely solely on static authentication methods. Research studies have shown that relying solely on static authentication methods, such as usernames and passwords, is no longer enough in preventing cyberattacks. In fact, according to Breach Investigations Report of Verizon 2021 Data [4], stolen credentials were the most common initial access vector in data breaches. This highlights the need for a more reliable and secure authentication system.

Moreover, many businesses have experienced significant financial losses due to data breaches. For example, the Equifax data breach in 2017 cost the company over $1.4 billion in settlements and legal fees. Therefore, implementing a trusted authentication system that continuously verifies user identity is crucial for businesses to protect their assets and maintain customer trust.

Furthermore, traditional authentication methods such as knowledge-based authentication (KBA) or two-factor authentication (2FA) have been proven to be ineffective against social engineering attacks, where attackers manipulate users into revealing sensitive information [5]. This emphasizes the need for a more advanced and secure authentication system that can resist such attacks.

To mitigate these risks, it is crucial for businesses to have a system that is reliable and trusted for identifying and authenticating users, to protect sensitive assets and financial data. To achieve this, there is a growing need for a trusted authentication system that does not rely on third-party apps. To further strengthen security, it is important for the system to continuously authenticate users in addition to static authentication methods. Authentication can be broadly classified into two types: static (one-time) authentication and continuous authentication. Static authentication typically involves the use of a password or multi-factor authentication methods [6–9], where users enter their credentials at the time of logging in to the system, and the backend database is where the verification takes place. The user is allowed to enter, access, or remain in the system if their credentials match and typing pattern matches; else, access is refused [10]. Contrarily, continuous authentication calculates the probability that a user who logs in repeatedly during a session is the same person they first claimed to be. This is done by analyzing the user's behavior, such as keystroke dynamics, without the need for external devices. This type of authentication can assist in securing the network from phishing and stuffing threats, while also providing an enhanced user experience as there is no interruption from external devices such as multi-factor authentication. It is important to use both static and continuous authentication methods to provide a more secure and user-friendly authentication system. Static authentication provides an initial level of security while logging in, while continuous authentication continuously monitors the user's behavior to ensure that the same person is accessing the system throughout the session.

This article presents a new approach to continuous authentication using a reinforcement learning-based anomaly detection method to be integrated with the current exiting static authentication architectures. In short, the goal is to utilize reinforcement learning (RL) techniques to identify and authenticate users in real-time, by continuously monitoring their behavior. To achieve this goal, the following have been achieved. investigated:

- Investigated the most advanced continuous authentication technology currently available, with a focus on keystroke dynamics as a form of the behavioral biometrics.
- Developed a novel reinforcement learning-based anomaly detection model for continuous authentication of keystroke dynamics.
- Evaluated the proposed model using real-world data, and a comparison with existing methods.
- The RL environment gym is developed and the proposed model has been implemented using this RL environment to provide a proof of concept application.
- This RL gym-based environment code is made available to the domain researchers on GitHub. **GitHub URL**: https://github.com/PriyaBansal68/Continuous-Authentication-Reinforcement-Learning-and-Behavioural-Biometrics.

The rest of article is structured as follows. Section 2 reviews the existing research on behavioural-based user authentication using machine learning. Section 3 explains how reinforcement learning fits into the broader field of machine learning and reviews the essential concepts. The proposed methodology comes in Section 4 after introducing the general reinforcement learning framework and exploring the different methodologies that can be used to train the reinforcement learning models. Finally, the results of this research are presented and discussed in Section 5.

2. Background and Literature Review

The concept of continuous authentication is not a new area of research, but it has been gaining popularity in recent years due to the increasing need for security in various domains such as finance, healthcare, and government. Continuous authentication is a process where the authentication of the user is done continuously throughout the session after the static authentication is successful, as opposed to traditional authentication methods where the user is only authenticated at the beginning of the session. This is done to ensure that the user is the same person throughout the session and to protect against unauthorized access.

User behavioural biometrics are a form of biometric authentication that uses the unique behavioural characteristics of the user to authenticate them. These characteristics include keystroke dynamics, mouse dynamics, and gait analysis [11]. These methods are non-intrusive and do not require any additional hardware. However, they are also more susceptible to attacks such as impersonation and replay attacks. Various machine learning algorithms have been proposed to detect these attacks, including decision trees, SVM, and neural networks [12].

Reinforcement learning (RL) is a machine learning approach that concentrates on training agents to make decisions in an environment to maximize the rewards they obtain. RL has been applied to various domains, including robotics, game playing, and natural language processing. However, its application to user authentication is relatively new.

Significant endeavors have been dedicated to the development of user recognition systems based on keystroke dynamics, aiming to enhance efficiency. This becomes particularly crucial considering the substantial volume of data produced by users, which may fluctuate over time due to contextual factors. While the quantity of studies exploring keystroke dynamics specifically in relation to text-based input is comparatively lower than those examining fixed text, there have been several notable studies conducted in this domain. During our background review on this topic, we encountered both supervised and unsupervised techniques, but we did not find any noteworthy information regarding reinforcement learning. The remainder of this section will concentrate on the most significant studies in the subject of behavioural biometrics, with the results of this debate summarised in Table 1.

In a paper they co-wrote, Asma Salem et al., [13] investigated if identity and verification system can be used on touch screen based mobile devices. Using WEKA, the authors build a multi-layer perceptron (MLP) neural network-based categorization model. Timing and non-timing elements are combined in the article, and it concludes that non-timing features raise the bar for security. Five users are included in the study, and the dataset has four attributes that are taken out. The writers bring up the issue of using different keyboards and create a virtual keyboard for collecting the data.

Jeanjaitrong et al., [14] conducted a literature review on keystroke dynamics and touch dynamics, highlighting the authentication process based on biometric behavior. The authors stressed the significance of protecting mobile devices because they are essential to daily life and pose a high risk of data theft. To categorize the data, the scientists retrieved 4 features: dwell duration, interval timing ratio, button spacing and interval time. Ten users were asked to choose one of sixteen four-symbol passwords to enter data. To determine the relationship between feature elements, the authors created a Bayesian Network and com-piled it throughout the classification phase.

Antal M. et al., [15] have conducted research on mobile device keystroke authentication using 1-class and 2-class classification algorithms. To examine the EER values for 2-class classification, they trained dataset on random forest classifiers and Bayesian networks. The 1-class classification was used to identify the user, whereas the 2-class classification was used to validate the user after separating them from outliers. Ac-cording to the authors' research, Random Forest has the highest EER value for a dataset of 42 users and 71 characteristics, and all 1-class classifiers performed better when categorizing the negative class than the positive class.

Lee et al., [16] used one 1-class classification technique to perform research on keystroke authentication for mobile devices. To determine the user's typing pattern, the authors presented a feature ex-traction method combining accelerometer and gyroscope sensors. The model was developed using a test population of 15 users, and the authors preprocessed, scaled, and standardized their data to provide good EER results.

99% classification accuracy was attained with efficiency by P. Bhattarakosol et al. [17]. Using a notebook as the input device, they gathered data from eight females and 4 male users. The k-NN model was created by the authors using 3 features: hold time, the inter-key, and finger pressure. The accuracy falls to 71% when only hold duration and the inter-key elements are used but increases to 91% when all three features are utilised, according to the authors.

In order to address cybersecurity issues such network intrusion and malicious assaults, Monrose [18] used factor analysis to evaluate user typing patterns to provide a lower dimensional representation based on correlations and dependencies among features, which he then used to build dynamic biometric approaches. The generated feature subset contained examples of both common and uncommon user typing patterns. Monrose employed a k-NN (nearest neighbour) classifier to classify data by visualising covariance matrices for several features. Keystroke dynamics has the potential to be coupled with any authentication system to increase its security layer, according to Monrose's conclusion.

The goal of the research by C. F. Araujo et al., [19] is to develop time delay features that will enhance authentication and reduce the incidence of erroneous rejection and false acceptance rates. They suggest an adaptive method that replaces outdated templates with fresh ones made from fresh samples. This method creates a two-trial authentication system by altering the conventional deviation and thresholds for each feature. While the user types on the screen, the biometric system logs key-stroke information such key up, key down, and ASCII codes. When the password is not a secret, the authors improve the current password authentication process using four key elements.

Antal, M. et al., [20] discussed different types of biometric systems used for authentication, including static and dynamic methods, as well as continuous authentication, which involves monitoring how the user interacts with the system over time.The author does, however, draw attention to the difficulty of cross-device authentication, which necessitates a model trained to identify users across various computing devices due to the possibility of differing keyboard layouts and screen coordinates.

A standardized experimental methodology and benchmark have been created by G. Stragapede, et al., [21] to allow fair comparisons of emerging approaches with currently used ones in the area. They suggest a system that employs an LSTM architecture (Long-Short Term Memory). At the score level, the architecture has triplet loss and modality fusion. The average AUC of the individual modalities is 58.75%, whereas the best modality's average AUC is 66.22%, representing a relative improvement of 12.71%. With a score of 68.72% AUC, the model performs best when all modalities are combined for the keystroke task. As comparison to using touch data alone, the combination of modalities yields an improvement of about 10%. The other modalities' performance is comparable.

N. Siddiqui, et al., [22] used three distinct machine learning and deep learning algorithms to evaluate a dataset of 40 users. The authors looked at two evaluation scenarios, one using multi-class classification and the other utilizing binary classifiers for user authentication. A one-dimensional convolutional neural network, which had the average test accuracy of (average) 85.73% for top ten users, was the best performer for binary classification. The maximum accuracy on the chosen dataset was attained with the help of artificial neural network (ANN) for multi-class classification, which reached a peak accuracy of 92.48%.

A group of researchers from Syracuse University [23] analyzed on typing behavior to categorize it under benign or adversarial activity. They collected the data from users and asked the users to perform certain tasks. They proposed 14 additional features for analysis. The data was trained using SVM, RF and NLP models using the 8 least correlated features. As a result of the experiments, they were able to achieve 97% accuracy and the type1 (False Positive) and type2 (False Negative) error less than 3%.

Attinà et al. [24] propose a convolutional neural network (CNN) with cut-out regularization. A hybrid model combining a CNN and a recurrent neural network (RNN) is also developed. The study uses the Buffalo free-text keystroke dataset. Two models are evaluated, with a CNN applied to the KDI image-like features, while a hybrid CNN-RNN model is applied to the KDS features. The Clarkson

II keystroke dataset is also analyzed, which is a free-text keystroke dynamics dataset collected from 101 subjects in a completely uncontrolled and natural setting over a period of 2.5 years. The study uses five time-based features - duration, down-down time (DD-time), up-down time (UD-time), up-up time (UU-time), and down-up time (DU-time) - extracted from consecutive keystroke events. The performance of the models is evaluated using accuracy and equal error rate (EER). The results show that the CNN model generated better results than the RNN-CNN model, and the performance on the Buffalo dataset was better than that of the Clarkson II dataset, likely due to noisier data in the latter. In conclusion, the study proposes effective feature engineering strategies and compares two feature structures for authentication based on free-text keystroke dynamics.

Table 1. Comparison study of the features, model, dataset used, and models of the recent literature.

Study	# of Users	Behavioural Biometrics	Features Used	ML Type	ML Model	Performance
[13]	5	Keystroke	Hold & Flight Time, Latency, Inter-key time, Acceleration	Supervised	Neural Network	FAR 2.2%, FRR 8.67%, EER 5.43%
[14]	10	keystroke	Dwell & Flight Time, Latency, Inter-key time	Supervised	Bayesian network classifier	Accuracy 82.18% FAR 2.0% FRR 17.8%
[15]	42	keystroke	Dwell & Flight Time, Latency, Inter-key time & Pressures	Supervised	Random Forests classifier, Bayes Network classifier, K-NN	EER 3% (2-class) EER 7% (1-class)
[16]	15	Keystroke & gyroscope	Hold Time, Flight Time, Latency, Inter-key time	Supervised	distance algorithm, 1-class classification	EER 6.93%
[17]	10	Keystroke and Finger Pressure	Dwell & Flight Time, Latency, Inter-key time	Supervised	Probabilistic Neural Network	Accuracy 99% EER hold-time (H) 35%, EER inter-key (I) 40%, EER finger pressure (P) 1 %
[18]	63	keystroke	Dwell & Flight Time, Latency, Inter-key time & Pressures	Supervised	weighted probabilistic classifier, Bayesian-like classifiers	Accuracy 83.22% to 92.14%
[19]	N/A	keystroke	Dwell & Flight Time, Latency, Inter-key time	Unsupervised, supervised	K-means, Bayes Net, and Neural networks	FRR 1.45% FAR 1.89%
[20]	54	keystroke	Dwell & Flight Time, Latency, Inter-key time, Key Pressures	Supervised	Random forests classifier, Bayes Net algorithms, and KNN	EER Random Forests classifier: for second order feature set 5% for full feature set 3%
[21]	81	Mouse	Dwell & Flight Time, Latency, Inter-key time, Touch Pressure	Supervised	Long-Short Term Memory (LSTM)	random impostor: 80%–87% AUC skilled impostor: 62%–69% AUC
[22]	40	Mouse	Speed, Clicks, Movement	Supervised	1-dimensional CNN, artificial neural network (ANN)	test accuracy 85.73% for the top-10 users,peak accuracy 92.48%
[23]	103	Keystroke	DDwell & Flight Time, Latency, Interkey time	Supervised	SVM, Random Forest (RF), Multilayer Perceptron (MLP)	accuracy (93% to 97%) Type I and Type II errors (3% to 8%)
[24]	73/80	Keystroke	down-down time, down-up time up-up time, up-down time	Supervised	MLP, CNN, RNN, CNN-RNN	Buffalo Dataset: Accuracy: 98.56% EER: 0.0088 Clarkson Dataset: Accuracy: 91.74 EER:0.0755
This study	117	Keystroke	Key, Dwell & Flight Time, Inter-key	Reinforcement Learning (RL)	Double Deep Q Networks (DDQN)	Train: Acc: 94.77% EER: 0.0255 FAR: 0.0126 FRR: 0.045 Test: Acc: 81.06% EER : 0.0323 FAR: 0.0356 FRR: 0.0174

3. Deep Dive Analysis for the Proposed Methodology

A subset of machine learning called reinforcement learning lights on teaching models how to make decisions in ambiguous situations. In the context of behavioural biometrics, reinforcement learning can be used to train models to make decisions about a user's identity dependent on their typing dynamics, mouse movement, and other behavioural patterns. The main idea behind reinforcement learning [25] is to train an agent to make decisions that will lead to the best outcome, or reward, over time.

In the case of behavioural biometrics, the agent would be trained on a dataset of typing dynamics or other behavioral patterns from a set of users. The agent would then use this training to make decisions about whether a new user is the same person as the one who was previously authenticated, or if they are an imposter [26].

One of the advantages of using reinforcement learning for behavioural biometrics is that it allows for continuous and dynamic adaptation of the model to the user's behaviour changes over time. This is because the agent can learn from its past decisions and update its decision-making strategy accordingly. Additionally, reinforcement learning can be used in a transparent way to the user, which means that the user doesn't have to actively participate in the authentication process [11].

In the subsequent section, we will delve into the conventional framework employed for behavioral biometrics, prior to introducing our proposed RL framework.

3.1. Traditional Frameworks

In Machine Learning Subset, Supervised learning in which a labeled dataset is used to train an algorithm, with each input having a corresponding known output or target. The objective is to develop a function that can map inputs to their respective outputs, allowing the model to forecast fresh events, unseen data [25]. In the context of keystroke dynamics, this involves training a supervised learning model on a dataset of keystroke data from known users, where the output is the user's identity. By doing so, the model can predict the user identity depending on their keystroke data as shown in Figure 1.

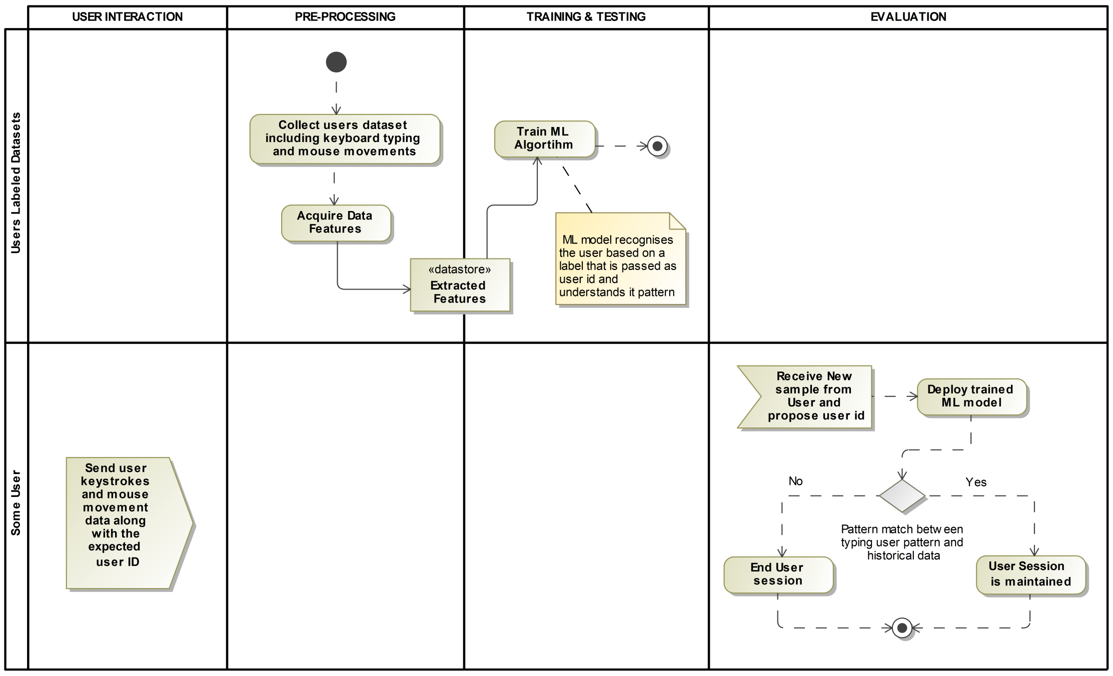

USER INTERACTION
PRE-PROCESSING
TRAINING & TESTING
EVALUATION
Users Labeled Datasets
Some User
Collect users dataset including keyboard typing and mouse movements
Acquire Data Features
«datastore» Extracted Features
Train ML Algortihm
ML model recognises the user based on a label that is passed as user id and understands it pattern
Send user keystrokes and mouse movement data along with the expected user ID
Receive New sample from User and propose user id
Deploy trained ML model
No
Yes
Pattern match between typing user pattern and historical data
End User session
User Session is maintained

Figure 1. Existing Continuous Authentication Framework for Behavioral Biometrics.

3.2. The Proposed Reinforcement Learning (RL) Framework

To begin with reinforcement Learning, we formulated our problem in RL mathematically in Markov Decision Process (MDP)[25]. The mathematical framework known as the Markov Decision Process (MDP) is the foundation of any reinforcement learning (RL) to model sequential decision-making problems. It consists of various states, actions, and rewards, and some rules for transitioning between states based on the actions taken. In the context of behavioral biometrics, MDP (in Figure 2) can be used to model the process of authenticating a user depending on their keystroke dynamics. The states in the MDP could represent different observations of the user's keystrokes, such as the timing between 2 subsequent key presses, the duration of time between of each key press, or the sequences of characters typed. The actions in the MDP could represent different authentication decisions, such as allowing access or denying access [27]. And the rewards in the MDP could represent the level of confidence in the authentication decision, with higher rewards assigned to more confident decisions and lower rewards assigned to less confident decisions.

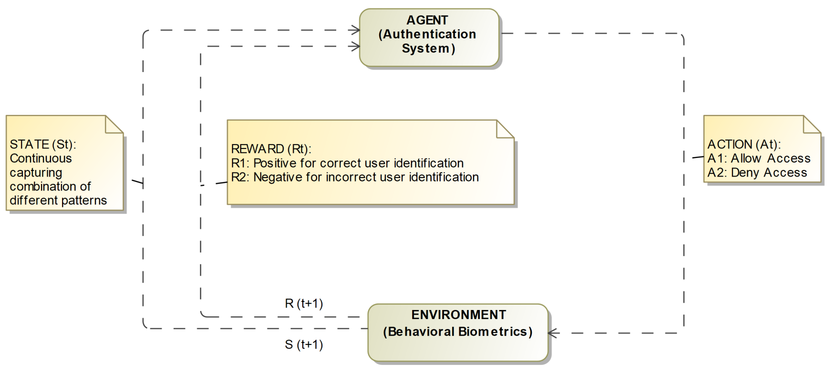

AGENT (Authentication System)
STATE (St): Continuous capturing combination of different patterns
REWARD (Rt): R1: Positive for correct user identification R2: Negative for incorrect user identification
ACTION (At): A1: Allow Access A2: Deny Access
R (t+1)
S (t+1)
ENVIRONMENT (Behavioral Biometrics)

Figure 2. Proposed MDP Diagram for Continuous Authentication using Behavioral Biometrics.

The proposed RL-base model for keystroke dynamics that provide continuous authentication would involve the following main elements:

1. **Agent:** The agent is the system that makes decisions based on the keystroke data. The agent is responsible for analyzing the user's keystroke patterns and deter-mining whether the user is who they claim to be.
2. **Reward:** The reward is a scalar value that the agent receives after each step of the authentication process. A positive reward is given when the agent correctly identifies the user, while a negative reward is given when the agent fails to identify the user. The agent attempts to maximize the long-term reward accumulated.
3. **Action:** This is the decision that the agent makes, based on the keystroke data. In this case, the action would be to either authenticate or reject the user.
4. **Environment:** This is the overall system that the agent interacts with. It includes the user's keystroke data, the decision-making process of the agent, and the feedback from the system.
5. **State:** The state represents the current typing pattern of a user, including factors such as typing speed, rhythm, and key press duration. The state could also include other features such as mouse movement, website activity, and other behavioral data that can be used to identify the user [28]. The state is an essential component of the MDP because it is used to inform the decision-making process of the agent and determine which action to take. This process is dependent on the present/current state and the rewards it receives for different actions.

These elements collaborate such that the machine learning (ML) algorithm will learn to act in a way that maximizes a reward signal. In order to get better over time, the algorithm interacts with the environment, learns from its actions and receives rewards or punishments accordingly, updating its internal state. In the context of keystroke dynamics, this would involve training a reinforcement learning model to recognize keystroke patterns from known users, where the rewards would be given when the algorithm correctly identifies a user and penalties would be given when it makes a mistake. The below image (Figure 3) shows how a reinforcement learning model integrated to develop a learning-based user authentication system analysing keystroke dynamics.

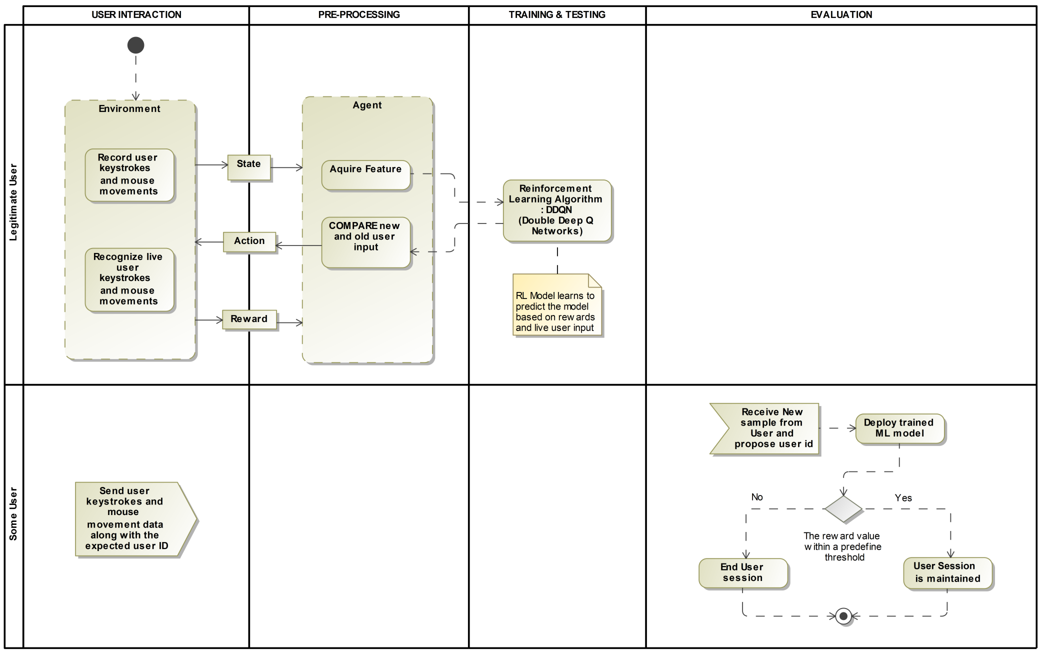

USER INTERACTION
PRE-PROCESSING
TRAINING & TESTING
EVALUATION
Legitimate User
Some User
Environment
Record user keystrokes and mouse movements
Recognize live user keystrokes and mouse movements
State
Action
Reward
Agent
Aquire Feature
COMPARE new and old user input
Reinforcement Learning Algorithm : DDQN (Double Deep Q Networks)
RL Model learns to predict the model based on rewards and live user input
Send user keystrokes and mouse movement data along with the expected user ID
Receive New sample from User and propose user id
Deploy trained ML model
No
Yes
The reward value within a predefine threshold
End User session
User Session is maintained

Figure 3. Proposed RL framework.

4. Methodology

By combining the two approaches reinforcement learning and behavioral biometrics, we have created a system that can continuously learn and adapt to changing user behavior and environmental conditions, providing reliable user authentication. We will discuss the various components of the proposed methodology, including data collection, feature extraction, reinforcement learning algorithms, and evaluation metrics [29]. Additionally, we will provide insights into the implementation and experimental results of our proposed method.

The following is a high-level overview of our approach to construct the reinforcement learning-based user authentication system using keystroke dynamics. The detail construction is described in the subsections that fellow.

1. **Collect a dataset of keystroke dynamics** data from several users. This should include a variety of different typing patterns, such as the time difference between key presses and the duration of time of each key press. In our case, we used the data from IEEE dataport website called BB-MA DATASET [23] as the data collection is a time-consuming task. As an addition, we collected our own data of keystrokes and trained the agent for testing purposes.
2. **Pre-process the data** to extract relevant features that can be used as inputs to the reinforcement learning algorithm. This might include mean, median, or the standard deviation of various keystroke features, and other statistical measures.
3. **Define the reinforcement learning environment.** This could be a simple decision tree, where the agent must choose between two actions: "accept" or "reject" the user's authentication request.
4. **Define the reward function.** This will determine what the agent is trying to optimize for. In the case of user authentication, the reward could be dependent on the accuracy of the agent's predictions. For example, the agent could receive a high reward for correctly accepting an authentic user and a low reward for incorrectly rejecting an authentic user.
5. **Train the agent** using the collected keystroke dynamics data and the defined reward function. This could be done using a variety of reinforcement learning algorithms, such as Q-learning or SARSA.
6. **Test the trained agent** on a separate dataset to evaluate its performance.

Figure 4 shows the flow of data and how the user would be authenticated at each step:

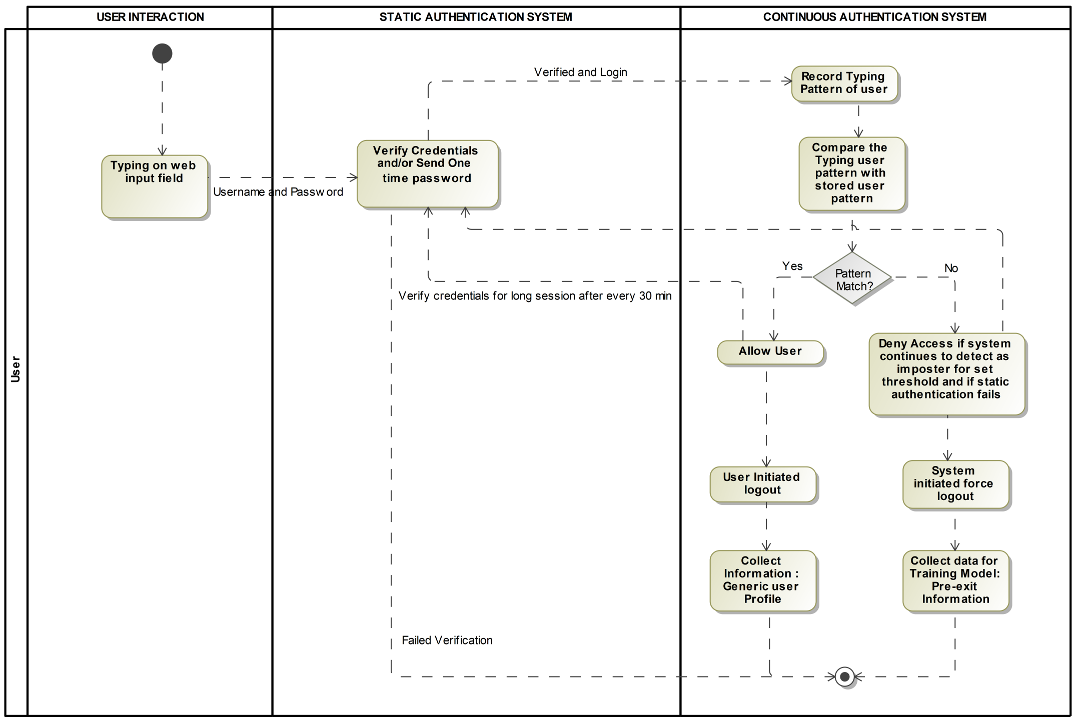

USER INTERACTION
STATIC AUTHENTICATION SYSTEM
CONTINUOUS AUTHENTICATION SYSTEM
User
Typing on web input field
Username and Password
Verify Credentials and/or Send One time password
Verified and Login
Record Typing Pattern of user
Compare the Typing user pattern with stored user pattern
Pattern Match?
Yes
No
Verify credentials for long session after every 30 min
Allow User
Deny Access if system continues to detect as imposter for set threshold and if static authentication fails
User Initiated logout
System initiated force logout
Collect Information : Generic user Profile
Collect data for Training Model: Pre-exit Information
Failed Verification

Figure 4. Data process flow for RL model.

4.1. Process flow

The process flow of training an agent for continuous authentication using reinforcement learning (RL) with behavioral biometrics is as follows (Figure 5):

1. **Preprocessing the historical data:** The first step is to gather a dataset of historical keystroke data from users. This data is then preprocessed to clean and format it for training. This may include removing any irrelevant data, normalizing the data, and dividing the data into sets for training and testing.
2. **Creating episodes on the cleaned data:** Next, the cleaned data is used to create episodes for training the agent. An episode is a sequence of observations and actions that the agent takes to learn from. Each episode is created by randomly selecting a user from the dataset and creating a sequence of observations and actions based on their keystroke data.
3. **Fetching observation from the environment:** The agent then fetches an observation from the environment. An observation is a set of data that the agent uses to decide. In this case, the observation is the keystroke data for a user.
4. **Predicting user or hacker on the given observation:** Using the observation, the agent makes a prediction of whether the user is an authorized user or a hacker. The agent's prediction is according to the user typing patterns and characteristics it has learned from the training data.
5. **Giving feedback to user in form of rewards:** The agent then receives feedback in the rewards form. A reward is a value that the agent receives for making a prediction. The reward is according to the accuracy of the agent's prediction. A positive re-ward is given for correctly identifying an authorized user and a negative reward is given for incorrectly identifying a hacker.
6. **Train on multiple episodes runs:** The agent is then trained on multiple episodes, with each episode providing the agent with new observations and rewards. As the agent receives feedback in the form of rewards, it updates its parameters and improves its ability to predict whether a user is an authorized user or a hacker. This process is repeated over multiple episodes until the agent reaches a satisfactory level of accuracy. This process flow is repeated for every user, to create an agent per user, which can be used to continuously authenticate users throughout a session by monitoring their behavior and predicting whether they are authorized users or imposters.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 14 June 2023 doi:10.20944/preprints202306.1005.v1

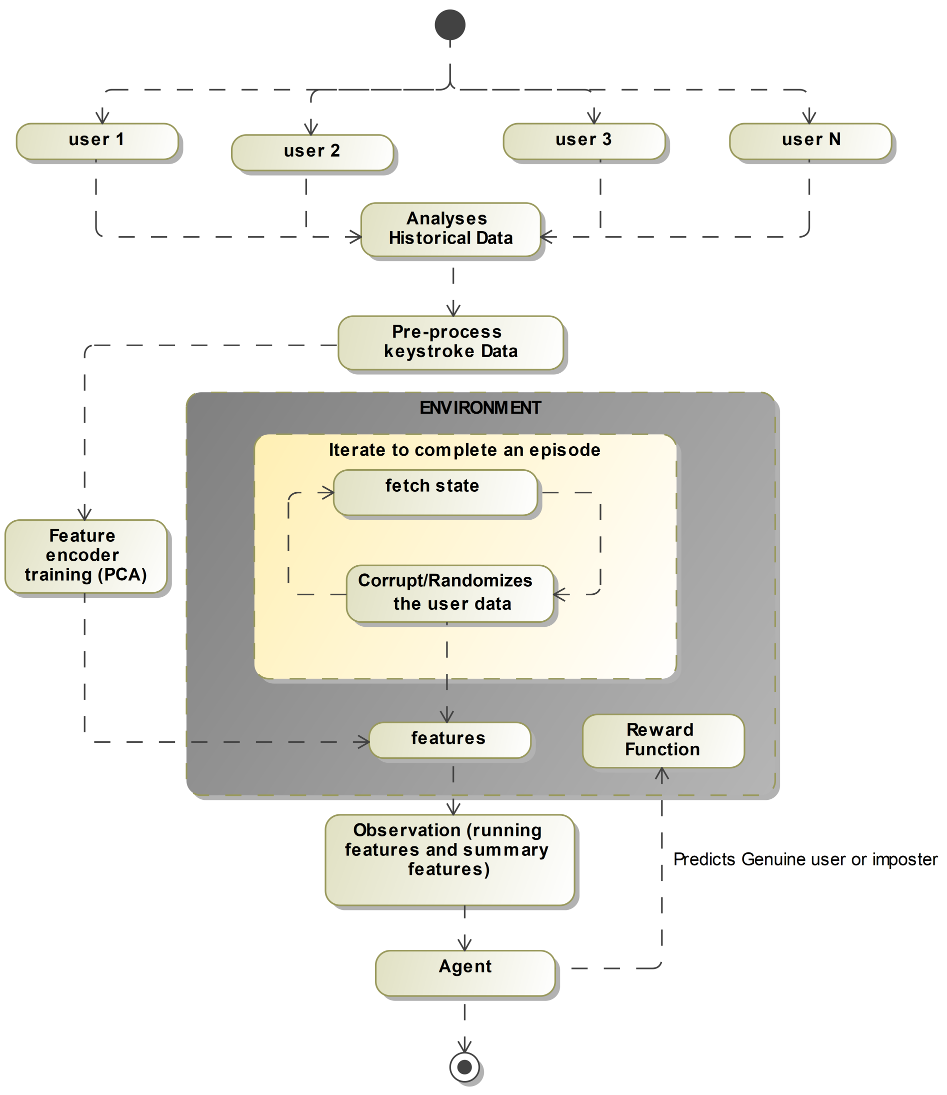

user 1
user 2
user 3
user N
Analyses Historical Data
Pre-process keystroke Data
ENVIRONMENT
Iterate to complete an episode
fetch state
Corrupt/Randomizes the user data
Feature encoder training (PCA)
features
Reward Function
Observation (running features and summary features)
Predicts Genuine user or imposter
Agent

Figure 5. Code Flow.

4.1.1. Data Preprocessing

About Original Dataset: The SU-AIS BB-MAS dataset [23] is a collection of keystroke data from multiple users performing various activities on different devices. The dataset was created by Syracuse University and Assured Information Security to provide a benchmark for behavioral biometrics research. The dataset was initially released in 2017 and latest updated in 2020 and contains data from 117 users performing 6 different activities on 5 different devices. The activities performed by the users include typing a predefined paragraph, free-form typing, copying, and pasting, web browsing, reading a PDF document, and playing a game. The devices used in the study include a desktop computer, a laptop computer, a tablet, a smartphone, and a smartwatch. The dataset includes both sin-gle-device and multi-device sessions. For this research, we are making use of Key-strokes data. For each keystroke, the dataset provides the timestamp, the key pressed, the key release time, and the user ID. The dataset also includes metadata about each session, such as the device used, and the activity performed. The keystroke data is provided in CSV format and is accompanied by documentation describing the dataset and its collection process. The SU-AIS BB-MAS dataset [23] has been used in various studies in behavioral bio-metrics research, including keystroke dynamics, multi-device authentication, and user identification. The dataset provides a valuable resource for researchers in this field, allowing them to compare their algorithms and techniques with a standardized benchmark.

A. Exploratory Data Analysis: Time difference between two consecutive events As a part of analysing the data for the 117 users in the data, we observed that none of the users has consistent typing pattern throughout the session which made it difficult for us to train the model with the features in the dataset. As a result, we re-searched and cameup with additional features for training. The Figures 6 and 7 below show the time difference between two consecutive events of 2 different users from the selected dataset.

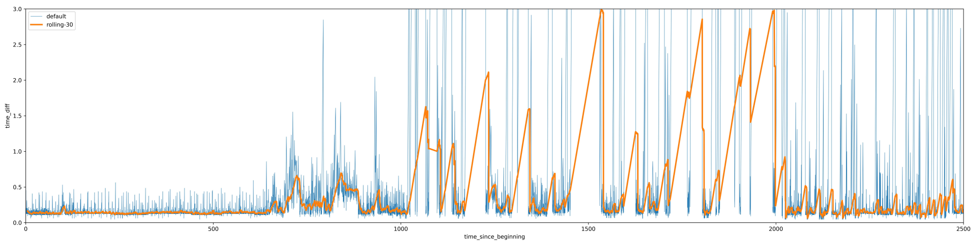


Figure 6. Time (ms) difference between two consecutive events for user 11.

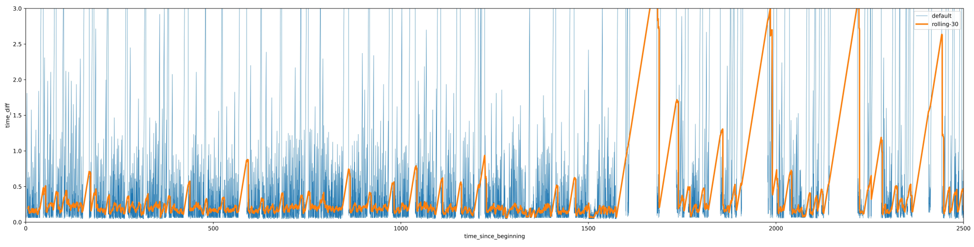


Figure 7. Time (ms) difference between two consecutive events for user 16.

B. Exploratory Data Analysis: Keys hold time The Figures 8 below show the key holding time (ms) for key ‘t’.

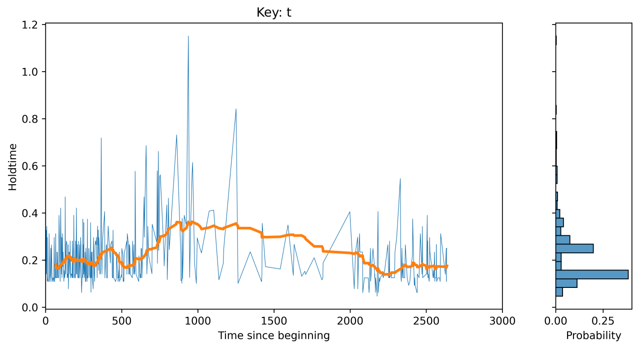


Figure 8. Key (‘t’) time (ms) for user 11.

C. Exploratory Data Analysis: Keyboard time length vs full session Keyboard time length vs full session was an interesting development of all the features, where noticed that most of the users spend almost 70-90% of their time on keyboard typing.

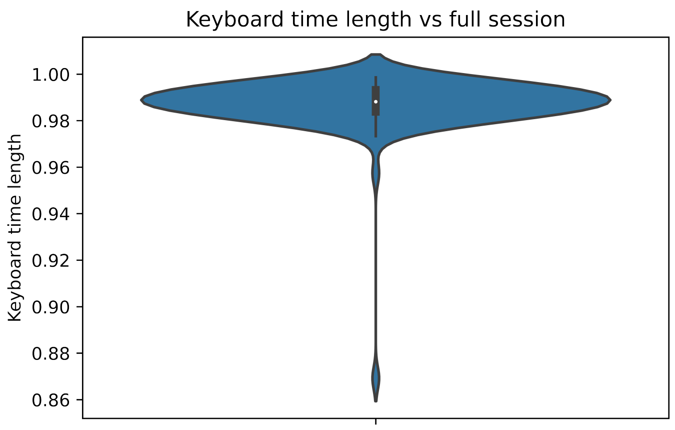


Figure 9. Keyboard time length vs full session.

Feature engineering and data preprocessing are important steps in training an agent for continuous authentication using reinforcement learning (RL) with behavioral biometrics. We performed below steps as pre-processing and designed the running and summary features in addition to the normal features:

1. **Standardized key names:** One of the first steps in data preprocessing is to standardize key names. This means making sure that all keys are represented in the same format and that there are no inconsistencies. This can help to en-sure that the data is clean and easy to work with clean data [28].
2. **Removed consecutive duplicate pairs (key, direction):** To reduce the dimensionality of the data, consecutive duplicate pairs of key and direction are removed. This can help to reduce the amount of data the agent needs to process, making the training process more efficient.
3. **Add column "time_diff" which is time difference between consecutive events:** To capture the timing information of keystrokes, a column "time_diff" is added which represents the difference of time among consecutive press and release events. This can help to capture the unique typing rhythm of an individual, which is a key behavioral biometric.
4. **Add column "time_since_beginning" that is cumulative sum of time difference column:** A cumulative sum of the time difference column is added, this column is called "time_since_beginning" which captures the time elapsed since the beginning of the typing session. This column can be used to capture the changes in behavior over time, which can be useful for detecting anomalies or changes in the user's behavior that may indicate a security threat [21].
5. Added new flight features such as press_to_press (P2P), release_to_press (R2P), hold_time (HT). press_to_press: Assuming, we have 2 keys, let's say I and K, then press time is I presstime- K presstime. release_to_press: I presstime – K releasetime
   hold_time: I releasetime – I presstime
6. Removed direction of the key as the features, we only considered press direction for our analysis.

Feature engineering and data preprocessing plays important role in training an agent for continuous authentication using RL with behavioral biometrics. These steps can help to ensure that the data is clean, easy to work with and that the agent is able to learn from the most relevant and informative features of the data.

4.1.2. Feature Engineering

Running Features: Running features are a technique used to capture the dynamics of the user's behaviour over time. They are particularly useful for the problem of continuous authentication using reinforcement learning (RL) with behavioural biometrics. This is because they allow the agent to learn from the user's behaviour over time, which can be important for detecting anomalies or the user's behavioural changes that may indicate a security threat.

A vector of size (n,) is created where the value is the hold time for the each key. This vector is calculated for a single event. For example, if there are n unique keys, and a user pressed the key 'a' for 2 seconds, the vector for that event would be [0, 2, 0, ... 0], where the first value represents the key 'a'.

If there are k multiple consecutive events, these vectors can be combined in a 2D vector (k, n). This 2D vector captures the dynamics of the behavior of user over time, by showing the hold time for each key across multiple events.

Additionally, for k events "time_diff" column is appended. This captures the difference of time between 2 consecutive press and release events. Therefore, in the end, we have a 2D vector of size (k+1, n) that captures the dynamics of the behavior over time, including the hold time for each key across multiple events and the difference of time between consecutive press and release events.

The 2D vector can then be used as an input to the RL agent, which can use it to learn from the dynamics of the user's behavior over time and make predictions about whether the user is an authorized user or a hacker.

Summary features: Summary features are a technique used to capture a summary or aggregate of the user's behavior over multiple events. They are particularly useful for the problem of continuous authentication using reinforcement learning (RL) with behavioral biometrics. This is because they allow the agent to gain insight from the broad trends and traits of the user's behavior, which can be important for detecting anomalies or changes in the user's behavior that may indicate a security threat.

Summary features can be calculated from k multiple consecutive events like typing speed, time_diff standard deviation, etc. [30]. These features summarize the user's behavior into a single value or set of values, making it easier for the agent to learn from the data.

For example, typing speed is a feature that can be calculated simply dividing the total time spent by the number of characters typed. This feature captures the overall typing speed of the user, which can be important for identifying unique typing rhythms. Time_diff standard deviation is another feature that can be calculated from k multiple consecutive events. It captures the variability in the difference between 2 consecutive key press and release events, which can also be used to identify unique typing rhythms.

4.2. ENVIRONMENT

The environment for a reinforcement learning (RL) model consists of the user's keystroke patterns and other behavioral biometric data. The RL model would be trained on this data to learn the user's unique keystroke patterns and other behavioral characteristics, and then use this knowledge to continuously authenticate the user. The RL agent would interact with the environment by observing the user's key-stroke patterns and other behavioral data, and then deciding on whether to authenticate the user based on this information. The agent's decisions would be based on its learned policy, which is updated as it gets feedback from the environment in the form of rewards or penalties. The RL algorithm would be trained on a dataset of keystroke patterns and other behavioral data from multiple users to learn to generalize to new users. The training data would be labelled with the identity of the user, so that the agent can learn to differentiate between different users' keystroke patterns and other behavioral data.

4.2.1. Fetch State

For training an agent for continuous authentication using reinforcement learning (RL) with behavioral biometrics, the environment plays an important role in fetching the state. In addition to this, the environment is accounted for providing the agent with the data it needs to make predictions. There are two important parameters to understand when fetching the state:

1. No: Number of events in an observation. This parameter determines the number of keystroke events that will be included in each observation.
2. Nh: Events that must occur before moving on to the following observation. This parameter determines the number of keystroke events that will be skipped before creating the next observation.

For example, if No=10 and Nh=4, the environment will create an observation from keystroke events 0-10 on the first iteration, keystroke events 4-14 on the second iteration, and so on. This allows the agent to learn from different parts of the keystroke data of each user. A user's past data is iterated upon to generate an episode using the above stated pattern. An episode is terminated if there are not enough data points to create the observation.

4.2.2. Corruption/Randomization

Corruption or randomization of user keystroke data in reinforcement learning can be used to provide the robustness of the model. Generally, when we speak of machine learning (ML) models, a model is trained on data which is usually a sample of the real data. If this sample is not representative of the real data, the model can be less accurate or perform poorly. By corrupting or randomizing the user keystroke data, it helps the model to generalize better and be more robust to different variations of the data. Corruption or randomization is a technique used to introduce variability and randomness into the training data, to help the agent learn to handle out-of-order behaviors and unexpected situations [19].

Corruption can also increase the diversity of the training data, making it less likely that the model will be an overfit to the training data. As a result, the model is able to generalize better to fresh and untested data.

Randomization of user keystroke data can also be used to make the model more robust to adversarial attacks. Adversarial attacks are attempts to fool the model by providing it with input that is specifically designed to cause an error. By randomizing the data, the model can learn to be more robust to variations in the data, which can make it more difficult for an attacker to fool the model [18].

This is necessary so that model is not always predicting the same user. This process is being referred to as corruption.

In this context, while iterating on the episode, some of the events are randomly chosen from different user's data to be introduced into the episode [15]. This is done with a certain probability, for example, 50% of the events are corrupted. This can help the agent to learn to recognize patterns that deviate from the usual pattern, and to handle unexpected situations.

4.2.3. Process to Create Observation

After fetching and randomly corrupting the state with some probability, the next step is to create an observation for the agent. This is done in 3 steps:

1. **Calculate running features of the state:** The first step is to calculate the running features of the state. This includes calculating the hold time for each key across multiple events and the difference of time be-tween 2 consecutive key press and release events. Running features provide the agent with information about the dynamics of the behavior over time.
2. **Encode the running features using trained encoder model:** The second step is to encode the running features using a trained encoder model. This can help to reduce the data dimensionality and make it more manageable for the agent to learn from.
3. **Calculate summary features and concatenate it with the encoded features:** The final step is to calculate summary features and concatenate them with the encoded features. Summary features are a set of aggregate characteristics of the user's behavior, such as typing speed, time_diff standard deviation, etc. By concatenating the summary and encoded features, the agent can learn from both the dynamics of the user's behavior over time and the overall patterns and characteristics of the user's behavior.

The final observation vector size is (f1+f2,), where f1 is the no. of summary features and f2 is the no. of encoded features. If the batch size is n, then the tensor that is fed to the agent is (n, f1+f2). This tensor provides the agent with the information it needs to make predictions about whether the user is an authorized user or a hacker.

4.2.4. Reward Function

The reward function is employed in reinforcement learning to give the agent feedback on the effectiveness of its actions. The agent's learning process and the ideal conduct are both guided by the reward function.

For the problem of continuous authentication using RL with behavioural biometrics, a minimalistic binary reward function is used to propagate rewards. The reward function assigns 1 for true negatives and true positives and 0 for false negatives and false positives.

True positive is when a corrupted observation is made, and the model correctly predicts that it is a hacker. A true negative is when a normal observation is made, and the model correctly predicts that it is an authorized user. A false positive (FP) is when a normal observation is made, but the model incorrectly predicts that it is a hacker. A false negative (FN) is when a corrupted observation is made, but the model incorrectly predicts that it is an authorized user. The reward function is used to guide and improve the overall learning process of agent by providing positive feedback for correct predictions and negative feedback for incorrect predictions. This can help the agent to learn to make better predictions and to improve its overall performance.

4.2.5. Feature Encoder

A feature encoder is a technique used to reduce dimensionality of data and make it more manageable for the agent to learn from. In this case, the feature encoder used is Principal Component Analysis (PCA) model. This technique PCA is used to recognize patterns in data, by finding the directions of maximum variance in the data. The PCA model is trained on the cleaned observations, which are the running and summary features calculated from the keystroke da-ta. After training, the PCA model can reduce the data dimensionality by identifying the most important features and discarding the less important ones. It was observed that for some users, even up to 10 components were able to ex-plain 99% variance. This means that even with just 10 components, the PCA model can capture most of the variation in the data. This is useful because it allows the agent to learn from a smaller set of features, which can make the learning process more efficient and less computationally expensive.

4.3. AGENT

The agent is the component of the reinforcement learning system that takes actions and interacts with the environment. In the context of continuous authentication using RL with behavioral biometrics, the agent is responsible for predicting whether the user is an authorized user or a hacker. The standard DDQN (DDQN is an extension of the Q-learning algorithm, explained in next section) algorithm was implemented for the agent. The architecture of the agent has of a fully connected neural network which is used as the policy net in DDQN. The network has the following architecture:

- Hidden layer 1: 32 nodes
- Hidden layer 2: 16 nodes
- Output layer: 2 nodes
- The activation function used in each layer except the last one is the ReLU activa-tion function.
- The activation function used in the last layer is the SoftMax activation function.
- The optimizer used is the Adam optimizer, with a learning rate of 0.001.

4.3.1. RL algorithm: DDQN

The Q-learning method, a kind of RL algorithm uses to learn the best action-value function for a certain environment, is extended by DDQN (as listed in Table 2). A primary Q-network and a target Q-network are the two distinct Q-networks employed in DDQN. While the target Q-network is used to create the target values for the primary Q-network during training, the primary Q-network is utilized to make predictions about the action-value function. This is done with the hope that it will lessen the cor-relation between the action-value estimations and the target values, stabilizing the training process and enhancing algorithm performance. Also, it addresses a few issues with Q-learning, such as overestimating Q-values. In DDQN, 2 neural networks namely: Q-network and target network. The Q-network estimates the Q-values, while the target network generates the targets for the Q-network.

Table 2. Reasons for choosing DDQN for this task.

Target Network	There are 2 Q-networks, the primary network, and the target net-work. The target network estimates the Q-values for the next state, which is then updates the primary network.
Action Selection	It is dependent on the primary network, and the Q-value estimation is done using the target network.
Learning Stability	DDQN has better learning stability. This is because DDQN reduces the overestimation of Q-values. This improvement in learning stability is due to the use of the target network in DDQN.
Exploration-Exploitation Trade-off	In DDQN, the exploration-exploitation trade-off is balanced by using the target network to approximates the Q-values.
Performance	DDQN has improved learning stability, which leads to better convergence to the optimal policy. Additionally, DDQN can learn faster and requires fewer training samples.

For keystroke dynamics, DDQN would learn to predict the user's keystroke pat-terns and other behavioral characteristics, and then use this information to decide on whether to authenticate the user. The agent would be trained on a dataset of keystroke patterns and other behavioral data from multiple users, and the training data would be labelled with the identity of the user, so that the agent can learn to differentiate be-tween different users' keystroke patterns and other behavioral data. DDQN proved beneficial because it allows for more accurate and reliable predictions of user behavior. By reducing overestimation bias, DDQN can better capture the nuances of user behavior and adapt to changes in that behavior over time. This can help to enhance the ac-curacy and effectiveness of the existing authentication system. In the next section, we would be evaluating the model's performance.

5. Results and Discussion

Evaluation is the process of assessing the performance of the reinforcement learning model using behavioral biometrics for continuous authentication. The evaluation is done on a test set and several parameters are varied to evaluate the model's performance. We performed multiple experiments after changing the parameter combinations in config.json file. The experiment that gave the best results has been dis-cussed below in this chapter. All the other experiments results are upload on GitHub in output folder (https://github.com/PriyaBansal68/Continuous-Authentication-Reinforcement-Learning-and -Behavioural-Biometrics/tree/main/output).

5.1. Evaluation Metrics

The effectiveness of keystroke analysis is often assessed using a variety of error rates, including Accuracy, FAR, FRR, EER and ROC Curve.

Accuracy: This metric represents the overall effectiveness of the keystroke dynamics system. It is calculated as follows in Equation (1):

$$Accuracy = ((TP + TN)/(TP + TN + FP + FN))x100\% \tag{1}$$

where, TP be the number of true positives, TN be the number of the true negatives.

False Rejection Rate (FRR) /Type I error: It calculates the proportion of legitimate users that are flagged as impostors. This kind of error is known as a Type I error in statistics. FRR described in equation (2)

$$FRR = FN/(FN + TP) = 1 - TPR \tag{2}$$

False Acceptance Rate (FAR) / Type II error: In statistics, FAR is the likelihood that an unauthorized person will get access to a secured system. The ideal scenario is to have both the FAR and FRR (Type I error) at 0%. While minimizing false acceptance is crucial from a security perspective, it is also important to minimize false rejection as legitimate users may become frustrated if they are mistakenly rejected by the system. FAR described in equation (3)

$$FAR = FPR = FP/(FP + TN) \tag{3}$$

Equal Error Rate (EER): EER is a widely used metric to evaluate biometric systems, which determines the point where the FAR and FRR are equal. A lower EER value indicates higher accuracy of the biometric system. EER described in equation (4)

$$ERR = (FRR + FAR)/2 \tag{4}$$

ROC Curve: In the context of continuous authentication of behavioral biometrics using Reinforcement Learning, the ROC curve and AUC can help system developers to evaluate and compare the performance of different machine learning algorithms, feature sets, or training methods [10]. They can also help to identify the optimal threshold value for the classifier algorithm, balancing the system sensitivity and specificity, and ultimately improve the accuracy and reliability of the authentication system.A perfect classification system would have an ROC curve that passes through the point (0,1), which represents a TPR of 100% and a FPR of 0%. The area under the ROC curve (AUC), a measure of the system's overall performance, is employed in practise because the curve is typically not perfect. With a value of 1 showing perfect discrimination and a value of 0.5 suggesting performance no better than random chance, a higher AUC indicates greater performance.

5.2. Hyperparameter Tuning

We tuned the model on the below hyperparameters that are also available in the public version (in config.json file). Trying different combination of below parameters, the model can be tuned further.

1. No: 100,
2. Nh: 50,
3. num_encoder_features:10,
4. num_corrupted_users: 10,
5. corrupt_bad_probability: 0.5,
6. num_episodes: 20,
7. c_update: 2,
8. eps_start: 0.1,
9. eps_decay: 200,
10. eps_end: 0.01,
11. train_split: 0.7

5.3. Results

Below are results (Tables 3, 4 and 5) of the experiments we performed:

Table 3. Below are the results when the full dataset of 117 users is run.

Metrics	Accuracy	EER	FAR	FRR
Training	94.77%	0.0255	0.0126	0.045
Test	81.06%	0.0323	0.0356	0.0174

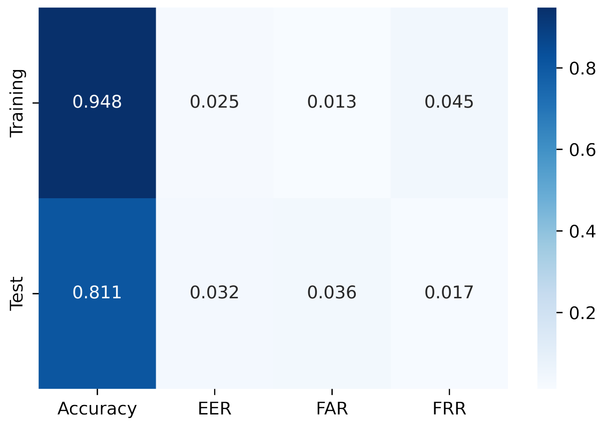


Figure 10. Heatmap results on full dataset of 117 users.

Table 4. Below are results when each user is run separately

User	Training Accuracy	Test Accuracy	Training EER	Test EER	Average FAR	Average FRR
1	96.5%	89.6%	0.0	0.07	0.31	0.29
71	96.7%	93.5%	0.0	0.11	0.39	0.28
99	96.8%	81.2%	0.049	0.06	0.19	0.34
116	100%	85.7%	0.0	0.13	0.26	0.32

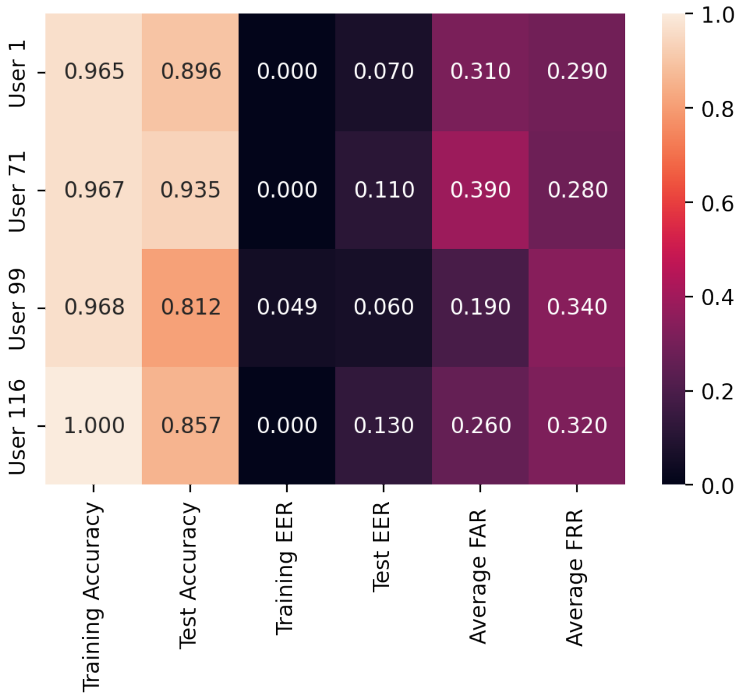
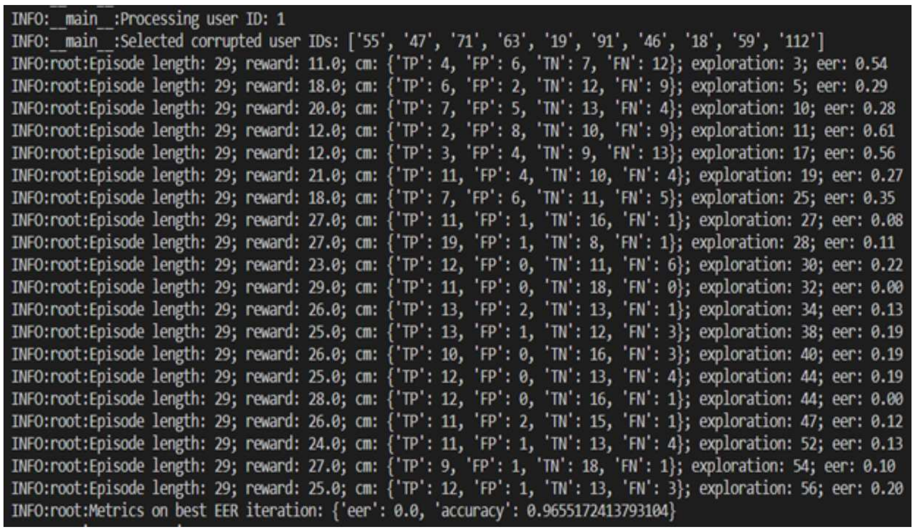


Figure 11. Heatmap results on users (randomly chosen) 1,71,99,116.

Training Accuracy/EER: The below are the snapshot of the training performed on individual users.



```
INFO:__main__:Processing user ID: 1
INFO:__main__:Selected corrupted user IDs: ['55', '47', '71', '63', '19', '91', '46', '18', '59', '112']
INFO:root:Episode length: 29; reward: 11.0; cm: {'TP': 4, 'FP': 6, 'TN': 7, 'FN': 12}; exploration: 3; eer: 0.54
INFO:root:Episode length: 29; reward: 18.0; cm: {'TP': 6, 'FP': 2, 'TN': 12, 'FN': 9}; exploration: 5; eer: 0.29
INFO:root:Episode length: 29; reward: 20.0; cm: {'TP': 7, 'FP': 5, 'TN': 13, 'FN': 4}; exploration: 10; eer: 0.28
INFO:root:Episode length: 29; reward: 12.0; cm: {'TP': 2, 'FP': 8, 'TN': 10, 'FN': 9}; exploration: 11; eer: 0.61
INFO:root:Episode length: 29; reward: 12.0; cm: {'TP': 3, 'FP': 4, 'TN': 9, 'FN': 13}; exploration: 17; eer: 0.56
INFO:root:Episode length: 29; reward: 21.0; cm: {'TP': 11, 'FP': 4, 'TN': 10, 'FN': 4}; exploration: 19; eer: 0.27
INFO:root:Episode length: 29; reward: 18.0; cm: {'TP': 7, 'FP': 6, 'TN': 11, 'FN': 5}; exploration: 25; eer: 0.35
INFO:root:Episode length: 29; reward: 27.0; cm: {'TP': 11, 'FP': 1, 'TN': 16, 'FN': 1}; exploration: 27; eer: 0.08
INFO:root:Episode length: 29; reward: 27.0; cm: {'TP': 19, 'FP': 1, 'TN': 8, 'FN': 1}; exploration: 28; eer: 0.11
INFO:root:Episode length: 29; reward: 23.0; cm: {'TP': 12, 'FP': 0, 'TN': 11, 'FN': 6}; exploration: 30; eer: 0.22
INFO:root:Episode length: 29; reward: 29.0; cm: {'TP': 11, 'FP': 0, 'TN': 18, 'FN': 0}; exploration: 32; eer: 0.00
INFO:root:Episode length: 29; reward: 26.0; cm: {'TP': 13, 'FP': 2, 'TN': 13, 'FN': 1}; exploration: 34; eer: 0.13
INFO:root:Episode length: 29; reward: 25.0; cm: {'TP': 13, 'FP': 1, 'TN': 12, 'FN': 3}; exploration: 38; eer: 0.19
INFO:root:Episode length: 29; reward: 26.0; cm: {'TP': 10, 'FP': 0, 'TN': 16, 'FN': 3}; exploration: 40; eer: 0.19
INFO:root:Episode length: 29; reward: 25.0; cm: {'TP': 12, 'FP': 0, 'TN': 13, 'FN': 4}; exploration: 44; eer: 0.19
INFO:root:Episode length: 29; reward: 28.0; cm: {'TP': 12, 'FP': 0, 'TN': 16, 'FN': 1}; exploration: 44; eer: 0.00
INFO:root:Episode length: 29; reward: 26.0; cm: {'TP': 11, 'FP': 2, 'TN': 15, 'FN': 1}; exploration: 47; eer: 0.12
INFO:root:Episode length: 29; reward: 24.0; cm: {'TP': 11, 'FP': 1, 'TN': 13, 'FN': 4}; exploration: 52; eer: 0.13
INFO:root:Episode length: 29; reward: 27.0; cm: {'TP': 9, 'FP': 1, 'TN': 18, 'FN': 1}; exploration: 54; eer: 0.10
INFO:root:Episode length: 29; reward: 25.0; cm: {'TP': 12, 'FP': 1, 'TN': 13, 'FN': 3}; exploration: 56; eer: 0.20
INFO:root:Metrics on best EER iteration: {'eer': 0.0, 'accuracy': 0.9655172413793104}
```

Figure 12. Training for User 1 showing Accuracy and EER.

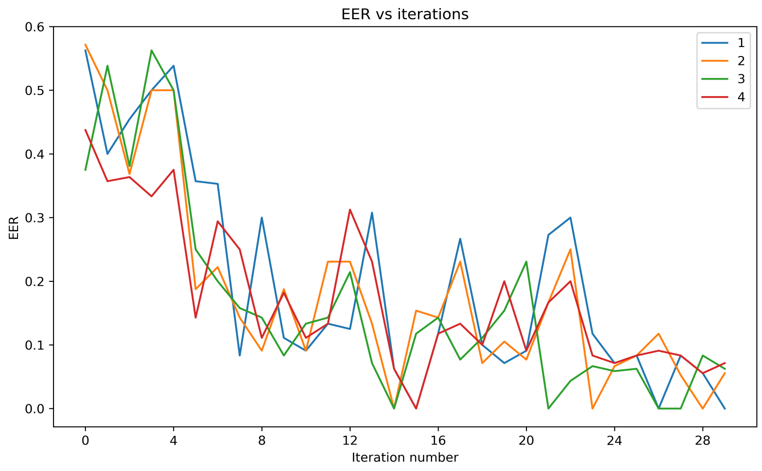
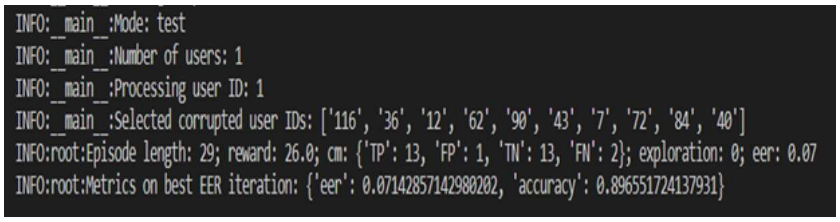


Figure 13. Graph for users 1,2,3 and 4 showing decreasing EER with no. of iterations.

Test Accuracy/EER: The snapshot, Figure 14, of the testing performed on individual users is a crucial aspect of evaluating the performance of the continuous authentication system. This testing provides insight into the accuracy and reliability of the system for individual users, which is particularly important for personalized systems, such as those used in healthcare or financial applications.

By analyzing the test accuracy and EER values for individual users, system developers can identify potential areas for improvement and tailor the system to the specific needs and characteristics of each user [31]. For example, if a user consistently exhibits unique typing patterns that are difficult to distinguish from those of unauthorized users, the system can be fine-tuned to better recognize the user's individual typing patterns and reduce false rejections [11].



```
INFO:__main__:Mode: test
INFO:__main__:Number of users: 1
INFO:__main__:Processing user ID: 1
INFO:__main__:Selected corrupted user IDs: ['116', '36', '12', '62', '90', '43', '7', '72', '84', '40']
INFO:root:Episode length: 29; reward: 26.0; cm: {'TP': 13, 'FP': 1, 'TN': 13, 'FN': 2}; exploration: 0; eer: 0.07
INFO:root:Metrics on best EER iteration: {'eer': 0.07142857142980202, 'accuracy': 0.896551724137931}
```

Figure 14. Testing for User 1 showing Accuracy and EER.

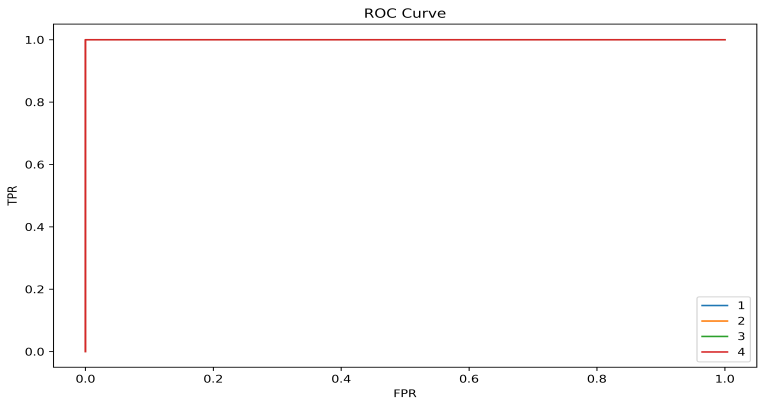


Figure 15. (TRAIN) ROC Curve for user 1 ,2 3 and 4.

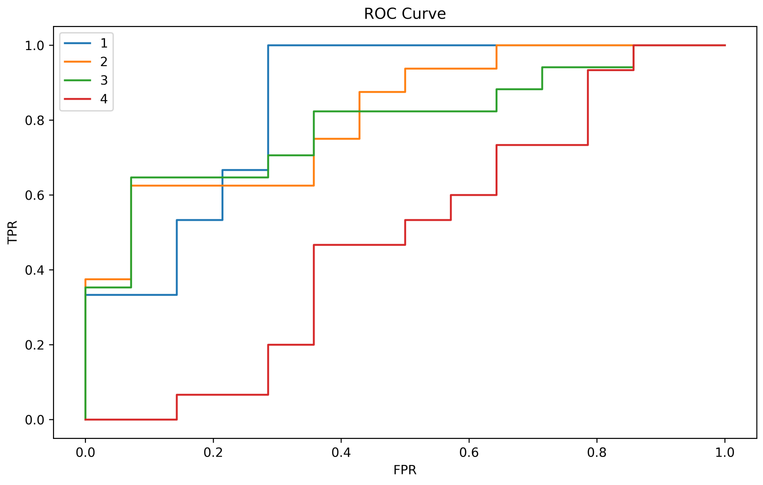


Figure 16. (TEST) ROC Curve for user 1 ,2, 3 and 4.

Comparison of Supervised Learning vs Reinforcement Learning results:

Table 5. Supervised learning vs Reinforcement learning results from same dataset

Metrics	Literature Review	This Study	This Study
ML Type	Supervised Learning	Supervised Learning	Reinforcement learning
Dataset	SU-AIS BBMAS subset	SU-AIS BBMAS	SU-AIS BBMAS
No. of users	102	117	117
Keys	10 Unigraphs, 5 Digraphs	All keys	All keys
Model	Neural Network	Random Forest	DDQN
Train/Test Accuracy	97% / NA	89.34% / 79.89%	94.77% / 81.06%
EER	80.03 - 0.06 = Average 0.045	0.157	0.0323

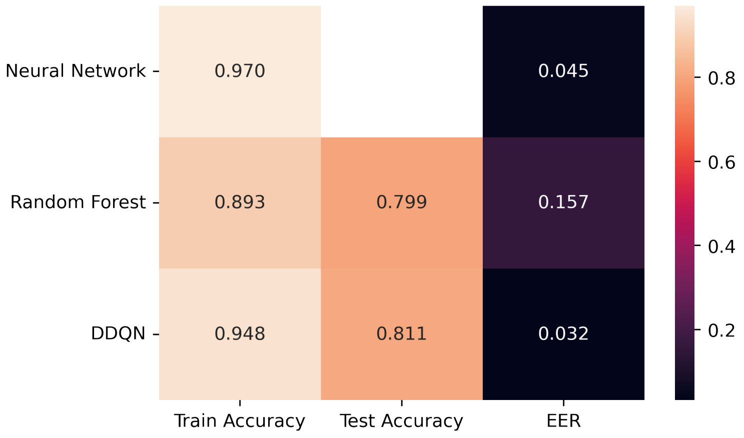


Figure 17. Heatmap results for comparison between supervised learning and reinforcement learning.

6. Conclusions

From the results, it is concluded that combining reinforcement learning and behavioral biometrics can provide a powerful approach to continuous authentication in the digital age. By continuously learning and adapting to changing behavior patterns, this approach can provide more secure and personalized authentication, reducing the risk of cyberattacks and unauthorized access. RL models can be deployed on client side where the model can adapt to learn the change in user behavior and diminishing the need to retrain the model unlike supervised learning models. Overall, the use of reinforcement learning and behavioral biometrics for continuous authentication has the potential to significantly enhance security in the digital age. Another additional advantage of using this approach and feature is that there is no need to get rid of any keys for analysis. We have used all the keys in the research unlike the other research where some have only selected 30 keys, unigraph or bigraphs are included as a part of the analysis. Also, to conclude, we achieved benchmark results on Keystroke dynamics using reinforcement learning, on the full dataset with Training and test accuracy as 94.77% and 81.06% and EER as 0.0255 and 0.0323 respectively. Moreover, the model would be able to detect when the user's behavior/ pattern changes over time. As an addition, the dependency on the data would decrease as the model would learn eventually to recognize the pattern own in own. Reinforcement Learning has the potential in the domain of behavioral biometrics overcoming multiple challenges that occur in supervised learning. By allowing agents to learn from their own experiences, reinforcement learning can adapt to changes in the data and provide accurate predictions even when labelled training data is limited or difficult to obtain. The agent learns from the feedback it receives based on its actions. This approach can be particularly useful in scenarios where the data is highly variable and subject to noise. The article concludes with a summary of the key findings, their practical implications, and recommendations for stakeholders, such as system developers, security professionals, and end-users, in the field of continuous authentication using keystroke dynamics while making use of reinforcement learning based approach. Below are some of the areas where the continuous authentication proves to be very beneficial:

1. **Healthcare:** In a hospital setting, medical professionals often must access sensitive patient data on computers located in public areas. With continuous authentication, the system can verify that only authorized personnel are accessing the data, reducing the risk of unauthorized access, and protecting patient privacy.
2. **Financial institutions:** In the financial sector, it is crucial to ensure that only authorized personnel can access sensitive financial data. Continuous authentication can prevent unauthorized access to banking systems by verifying the identity of users throughout their session.
3. **Remote work:** With an increasing number of employees working from home, it is important for companies to ensure that their networks are secure. Continuous authentication can be particularly useful in remote work environments, where employees may be working from unsecured locations or using unsecured devices.
4. **Online transactions:** With the rise in online shopping and banking, it is important to ensure that users are protected from cyber threats. Continuous authentication can prevent unauthorized access to online accounts by verifying the identity of a user throughout the session, reducing the risk of identity theft and fraud.

References

1. Gold, J. Traditional Security Is Dead – Why Cognitive-Based Security Will Matter. https://www.computerworld.com/article/3068185/traditional-security-is-dead-why-cognitive-based-security-will-matter.html, 2016. Last accessed 25 May 2023.
2. Ventures, C. Cybercrime To Cost The World $10.5 Trillion Annually By 2025. https://www.prnewswire.com/news-releases/cybercrime-to-cost-the-world-10-5-trillion-annually-by-2025--301172786.html, 2020. Last accessed 25 May 2023.
3. Editor, N. 67 Percent of Breaches Caused by Credential Theft, User Error, and Social Attacks. https://www.netsec.news/67-percent-of-breaches-caused-by-credential-theft-user-error-and-social-attacks/#:~:text=The%20report%20revealed%20that%20the, 2020. Last accessed 25 May 2023.
4. Burbidge, T. Cybercrime thrives during pandemic: Verizon 2021 Data Breach Investigations Report. https://www.verizon.com/about/news/verizon-2021-data-breach-investigations-report, 2021. Last accessed 25 May 2023.
5. Ginni. What are the disadvantage of Multi-factor authentication? https://www.tutorialspoint.com/what-are-the-disadvantage-of-multi-factor-authentication, 2022. Last accessed 25 May 2023.
6. Voege, P.; Ouda, A. An Innovative Multi-Factor Authentication Approach. 2022 International Symposium on Networks, Computers and Communications (ISNCC), 2022, pp. 1–6. doi:10.1109/ISNCC55209.2022.9851710.
7. Ouda, A. A framework for next generation user authentication. 2016 3rd MEC International Conference on Big Data and Smart City (ICBDSC), 2016, pp. 1–4. doi:10.1109/ICBDSC.2016.7460349.
8. Voege, P.; Abu Sulayman, I.I.; Ouda, A. Smart chatbot for user authentication. *Electronics* **2022**, *11*, 4016.
9. Abu Sulayman, I.I.M.; Ouda, A. Designing Security User Profiles via Anomaly Detection for User Authentication. 2020 International Symposium on Networks, Computers and Communications (ISNCC), 2020, pp. 1–6. doi:10.1109/ISNCC49221.2020.9297252.
10. Li, J.; Chang, H.C.; Stamp, M. Free-text keystroke dynamics for user authentication. In *Artificial Intelligence for Cybersecurity*; Springer, 2022; pp. 357–380.
11. Gupta, S. USER ATTRIBUTION IN DIGITAL FORENSICS THROUGH MODELING KEYSTROKE AND MOUSE USAGE DATA USING XGBOOST. PhD thesis, Purdue University Graduate School, 2022.
12. Verma, N.; Prasad, K. Responsive parallelized architecture for deploying deep learning models in production environments. *arXiv preprint arXiv:2112.08933* **2021**.

13. Salem, A.; Zaidan, D.; Swidan, A.; Saifan, R. Analysis of strong password using keystroke dynamics authentication in touch screen devices. 2016 Cybersecurity and Cyberforensics Conference (CCC). IEEE, 2016, pp. 15–21.
14. Jeanjaitrong, N.; Bhattarakosol, P. Feasibility study on authentication based keystroke dynamic over touch-screen devices. 2013 13th international symposium on communications and information technologies (ISCIT). IEEE, 2013, pp. 238–242.
15. Antal, M.; Szabó, L.Z. An evaluation of one-class and two-class classification algorithms for keystroke dynamics authentication on mobile devices. 2015 20th International Conference on Control Systems and Computer Science. IEEE, 2015, pp. 343–350.
16. Roh, J.h.; Lee, S.H.; Kim, S. Keystroke dynamics for authentication in smartphone. 2016 international conference on information and communication technology convergence (ICTC). IEEE, 2016, pp. 1155–1159.
17. Saevanee, H.; Bhattarakosol, P. Authenticating user using keystroke dynamics and finger pressure. 2009 6th IEEE Consumer Communications and Networking Conference. IEEE, 2009, pp. 1–2.
18. Monrose, F.; Rubin, A.D. Keystroke dynamics as a biometric for authentication. *Future Generation computer systems* **2000**, *16*, 351–359.
19. Araújo, L.C.; Sucupira, L.H.; Lizarraga, M.G.; Ling, L.L.; Yabu-Uti, J.B.T. User authentication through typing biometrics features. *IEEE transactions on signal processing* **2005**, *53*, 851–855.
20. Antal, M.; Nemes, L. The mobikey keystroke dynamics password database: Benchmark results. Software Engineering Perspectives and Application in Intelligent Systems: Proceedings of the 5th Computer Science On-line Conference 2016 (CSOC2016), Vol 2 5. Springer, 2016, pp. 35–46.
21. Stragapede, G.; Vera-Rodriguez, R.; Tolosana, R.; Morales, A. BehavePassDB: Public Database for Mobile Behavioral Biometrics and Benchmark Evaluation. *Pattern Recognition* **2023**, *134*, 109089.
22. Siddiqui, N.; Dave, R.; Vanamala, M.; Seliya, N. Machine and deep learning applications to mouse dynamics for continuous user authentication. *Machine Learning and Knowledge Extraction* **2022**, *4*, 502–518.
23. Belman, A.K.; Wang, L.; Iyengar, S.S.; Sniatala, P.; Wright, R.; Dora, R.; Baldwin, J.; Jin, Z.; Phoha, V.V. SU-AIS BB-MAS (Syracuse University and Assured Information Security—Behavioral Biometrics Multi-device and multi-Activity data from Same users) Dataset. *IEEE DataPort* **2019**.
24. Attinà, F., Traditional Security Issues. In *China, the European Union, and the International Politics of Global Governance*; Wang, J.; Song, W., Eds.; Palgrave Macmillan US: New York, 2016; pp. 175–193. doi:10.1057/9781137514004_10.
25. Sutton, R.S.; Barto, A.G. *Reinforcement learning: An introduction*; MIT press, 2018.
26. Singh, S.; others. Keystroke dynamics for continuous authentication. 2018 8th International Conference on Cloud Computing, Data Science & Engineering (Confluence). IEEE, 2018, pp. 205–208.
27. Çevik, N.; Akleylek, S.; Koç, K.Y. Keystroke Dynamics Based Authentication System. 2021 6th International Conference on Computer Science and Engineering (UBMK). IEEE, 2021, pp. 644–649.
28. Liang, Y.; Samtani, S.; Guo, B.; Yu, Z. Behavioral biometrics for continuous authentication in the internet-of-things era: An artificial intelligence perspective. *IEEE Internet of Things Journal* **2020**, *7*, 9128–9143.
29. Bansal, P.; Ouda, A. Study on Integration of FastAPI and Machine Learning for Continuous Authentication of Behavioral Biometrics. 2022 International Symposium on Networks, Computers and Communications (ISNCC). IEEE, 2022, pp. 1–6.
30. Huang, J.; Hou, D.; Schuckers, S.; Law, T.; Sherwin, A. Benchmarking keystroke authentication algorithms. 2017 IEEE Workshop on Information Forensics and Security (WIFS). IEEE, 2017, pp. 1–6.
31. Belman, A.K.; Sridhara, S.; Phoha, V.V. Classification of threat level in typing activity through keystroke dynamics. 2020 International Conference on Artificial Intelligence and Signal Processing (AISP). IEEE, 2020, pp. 1–6.