computers

Article

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

Priya Bansal and Abdelkader Ouda *

Department of Electrical and Computer Engineering, Western University, London, ON N6A 5B9, Canada; pbansa2@uwo.ca
* Correspondence: aouda@uwo.ca

Abstract: This research article delves into the development of a reinforcement learning (RL)-based continuous authentication system utilizing behavioral biometrics for user identification on computing devices. Keystroke dynamics are employed to capture unique behavioral biometric signatures, while a reward-driven RL model is deployed to authenticate users throughout their sessions. The proposed system augments conventional authentication mechanisms, fortifying them with an additional layer of security to create a robust continuous authentication framework compatible with static authentication systems. The methodology entails training an RL model to discern atypical user typing patterns and identify potentially suspicious activities. Each user's historical data are utilized to train an agent, which undergoes preprocessing to generate episodes for learning purposes. The environment involves the retrieval of observations, which are intentionally perturbed to facilitate learning of nonlinear behaviors. The observation vector encompasses both ongoing and summarized features. A binary and minimalist reward function is employed, with principal component analysis (PCA) utilized for encoding ongoing features, and the double deep Q-network (DDQN) algorithm implemented through a fully connected neural network serving as the policy net. Evaluation results showcase training accuracy and equal error rate (EER) ranging from 94.7% to 100% and 0 to 0.0126, respectively, while test accuracy and EER fall within the range of approximately 81.06% to 93.5% and 0.0323 to 0.11, respectively, for all users as encoder features increase in number. These outcomes are achieved through RL's iterative refinement of rewards via trial and error, leading to enhanced accuracy over time as more data are processed and incorporated into the system.

Keywords: behavioral biometrics; continuous authentication; keystroke dynamics; Markov decision process (MDP); Q-learning; reinforcement learning (RL); static authentication; user authentication; identification

check for updates

Citation: Bansal, P.; Ouda, A. Continuous Authentication in the Digital Age: An Analysis of Reinforcement Learning and Behavioral Biometrics. *Computers* **2024**, *13*, 103. https://doi.org/10.3390/computers13040103

Academic Editors: Leandros Maglaras, Helge Janicke, Mohamed Amine Ferrag and Francisco J. Aparicio-Navarro

Received: 13 March 2024
Revised: 10 April 2024
Accepted: 17 April 2024
Published: 18 April 2024

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 the Verizon 2021 Data Breach Investigations Report [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 USD 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 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 multifactor authentication methods [6–8], 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 [9]. 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 carried out by analyzing the user's behavior, such as keystroke dynamics, without the need for external devices. 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 (RL)-based anomaly detection method to be integrated with the current exiting static authentication architectures. To achieve this goal, the following has been investigated and proposed:

- **Innovative Approach to existing Continuous Authentication:** Investigated the most advanced continuous authentication technology currently available, with a focus on keystroke dynamics as a form of the behavioral biometrics. This approach aims to enhance existing static authentication systems.
- **Development of Reinforcement Learning Model:** 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 reinforcement learning (RL) environment gym is developed and the proposed model has been implemented from scratch to provide a proof of concept application.
- **Open Source Contribution:** This reinforcement learning (RL) gym-based environment code is made available on GitHub (**GitHub:** to the domain researchers to explore and utilize. https://github.com/PriyaBansal68/Continuous-Authentication-Reinforcement-Learning-and-Behavioural-Biometrics) (accessed on 18 April 2024).

The rest of article is structured as follows. Section 2 reviews the existing research on behavioral-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

Continuous authentication involves continuously verifying and validating a user's identity during their entire session or interaction with a system, rather than relying solely on a single authentication event at the beginning. It helps to enhance security [10] by constantly monitoring and assessing the user's behavior, characteristics, or biometrics to ensure their identity remains valid and authorized. The following are some of the ways user can be authenticated continuously are behavioral biometrics, Facial or voice recognition.

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 come across any noteworthy information regarding reinforcement learning. The remainder of this section will concentrate on the most significant studies in the subject of behavioral biometrics in Table 1.

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

Study	# of Users	Behavioral Biometrics	Features Used	ML Type	ML Model	Performance
[11]	5	Keystroke	Hold and flight time, latency, interkey time, acceleration	Supervised	Neural network	FAR 2.2%, FRR 8.67%, EER 5.43%
[12]	10	Keystroke	Dwell and flight time, latency, interkey time	Supervised	Bayesian network classifier	Accuracy 82.18%, FAR 2.0%, FRR 17.8%
[13]	42	Keystroke	Dwell and flight time, latency, interkey time and pressures	Supervised	Random forest classifier, Bayes network classifier, K-NN	EER 3% (2-class), EER 7% (1-class)
[14]	15	Keystroke and gyroscope	Hold time, flight time, latency, interkey time	Supervised	Distance algorithm, 1-class classification	EER 6.93%
[15]	10	Keystroke and finger pressure	Dwell and flight time, latency, interkey time	Supervised	Probabilistic neural network	Accuracy 99%, EER hold-time (H) 35%, EER interkey (I) 40%, EER finger pressure (P) 1%
[16]	63	Keystroke	Dwell and flight time, latency, interkey time and pressures	Supervised	Weighted probabilistic classifier, Bayesian-like classifiers	Accuracy 83.22% to 92.14%
[17]	N/A	Keystroke	Dwell and flight time, latency, interkey time	Unsupervised, supervised	K-means, Bayes net, and neural networks	FRR 1.45% FAR 1.89%
[18]	54	Keystroke	Dwell and flight time, latency, interkey time, key pressures	Supervised	Random forest classifier, Bayes net algorithms, and KNN	EER random forest classifier: for second-order feature set 5%; for full feature set 3%
[19]	81	Mouse	Dwell and flight time, latency, interkey time, touch pressure	Supervised	Long short-term memory (LSTM)	Random impostor: 80–87%; AUC skilled impostor: 62–69% AUC
[20]	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%
[21]	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
[22]	54	Keystroke	Interval, dwell time, latency, flight time, up to up	Supervised	Neural network layers—convolutional, recurrent, and LSTM	Accuracy: 88%
[23]	103	Keystroke	Dwell and 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%)
This study	117	Keystroke	Key, dwell and flight time, interkey	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

In a paper, Asma Salem et al. [11] investigated if an identity and verification system can be used on touch-screen-based mobile devices. Using WEKA, the authors build a multilayer 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. [12] 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 compiled it throughout the classification phase.

Antal M. et al. [13] conducted research on mobile device keystroke authentication using one-class and two-class classification algorithms. To examine the EER values for two-class classification, they trained a dataset on random forest classifiers and Bayesian networks. One-class classification was used to identify the user, whereas two-class classification was used to validate the user after separating them from outliers. According to the authors' research, random forest has the highest EER value for a dataset of 42 users and 71 characteristics, and all one-class classifiers performed better when categorizing the negative class than the positive class.

Lee et al. [14] used one one-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.

A classification accuracy of 99% was attained with efficiency by P. Bhattarakosol et al. [15]. Using a notebook as the input device, they gathered data from eight females and four male users. The k-NN model was created by the authors using three features: hold time, the interkey, and finger pressure. The accuracy falls to 71% when only hold duration and the interkey elements are used, but increases to 91% when all three features are utilized, according to the authors.

In order to address cybersecurity issues such as network intrusion and malicious assaults, Monrose [16] 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 visualizing 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. [17] 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 keystroke information such as 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. [18] 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. [19], to allow for 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. [20] 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 multiclass classification, which reached a peak accuracy of 92.48%.

A group of researchers from Syracuse University [21] analyzed 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 were trained using SVM, RF, and NLP models using the eight least correlated features. As a result of the experiments, they were able to achieve 97% accuracy and a type1 (false positive) and type2 (false negative) error less than 3%.

Attinà et al. [22] 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. Pawel Kasprowski, Zaneta Borowska, and Katarzyna Harezlak [23] investigated the impact of altering neural network architecture and hyperparameters on biometric identification using keystroke dynamics. A publicly available dataset of keystrokes was utilized to train models with diverse parameters. The neural network configurations encompassed convolutional, recurrent, and dense layers in various arrangements, combined with pooling and dropout layers. The outcomes were subsequently compared with those achieved by the state-of-the-art model, using the identical dataset. The results exhibited variability, with the highest attained accuracy reaching 82–88% for the identification task involving 20 subjects.

All the existing approaches are performed on fixed text, whereas this study takes into account the free text and any change in environment that can cause change in user behavior. It can detect unusual patterns and detect suspicious activities by interacting with an environment, receiving feedback in the form of rewards or penalties, and maximizing cumulative rewards over time.

3. Deep-Dive Analysis for the Proposed Methodology

A subset of machine learning called reinforcement learning (RL) sheds light on teaching models for how to make decisions in ambiguous situations. In the context of behavioral 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 behavioral patterns. The main idea behind reinforcement learning [24] is to train an agent to make decisions that will lead to the best outcome, or reward, over time. However, its application to user authentication is relatively new.

In the case of behavioral 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 [25].

One of the advantages of using reinforcement learning for behavioral biometrics is that it allows for continuous and dynamic adaptation of the model to the user's behavior 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 does not have to actively participate in the authentication process [26].

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 the machine learning subset, supervised learning is where 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 and unseen data [24]. 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. Thus, the model can predict the user identity depending on their keystroke data, as shown in Figure 1.

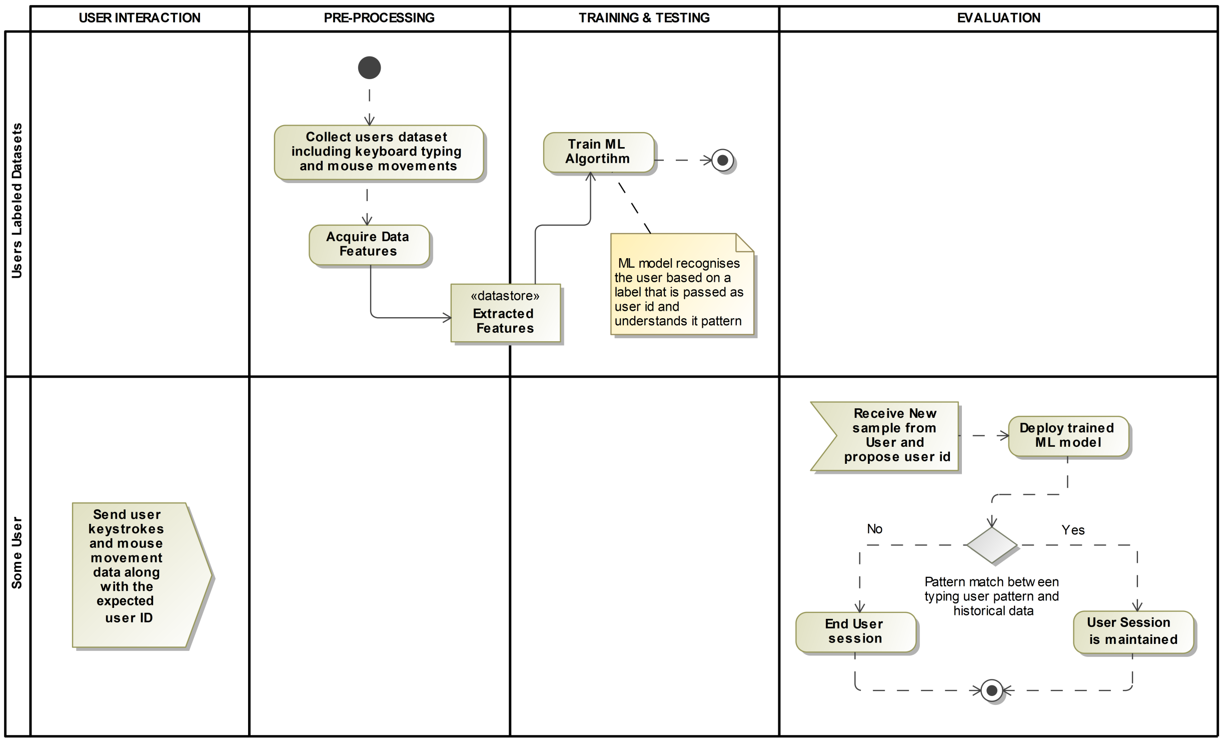
USER INTERACTION
PRE-PROCESSING
TRAINING & TESTING
EVALUATION
Users Labeled Datasets
Some User
Collect users dataset including keyboard typing and mouse movements
Acquire Data Features
«datastore» Extracted Features
Train ML Algortihm
ML model recognises the user based on a label that is passed as user 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 RL Framework

To begin with reinforcement Learning, we formulated our problem in reinforcement learning (RL) mathematically in MDP [24]. The mathematical framework known as the Markov decision process is the foundation of any 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 two 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.

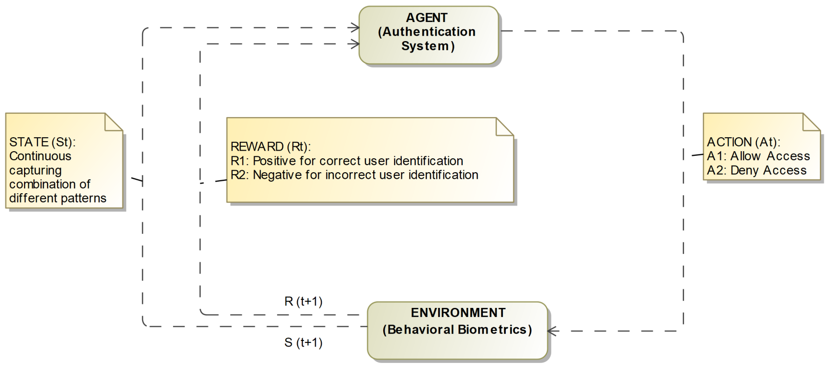
AGENT (Authentication System)
STATE (St): Continuous capturing combination of different patterns
REWARD (Rt):
R1: Positive for correct user identification
R2: Negative for incorrect user identification
ACTION (At):
A1: Allow Access
A2: Deny Access
R (t+1)
S (t+1)
ENVIRONMENT (Behavioral Biometrics)

Figure 2. Proposed MDP diagram for continuous authentication using behavioral biometrics.

The proposed reinforcement learning (RL)-based model (shown in Figure 3) for keystroke dynamics that provide continuous authentication would involve the following main elements which is in continuous interaction with agent unlike the traditional framework in Figure 1:

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 determining 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.

The below image (Figure 3) shows how a reinforcement learning model integrated to develop a learning-based user authentication system analyzing keystroke dynamics.

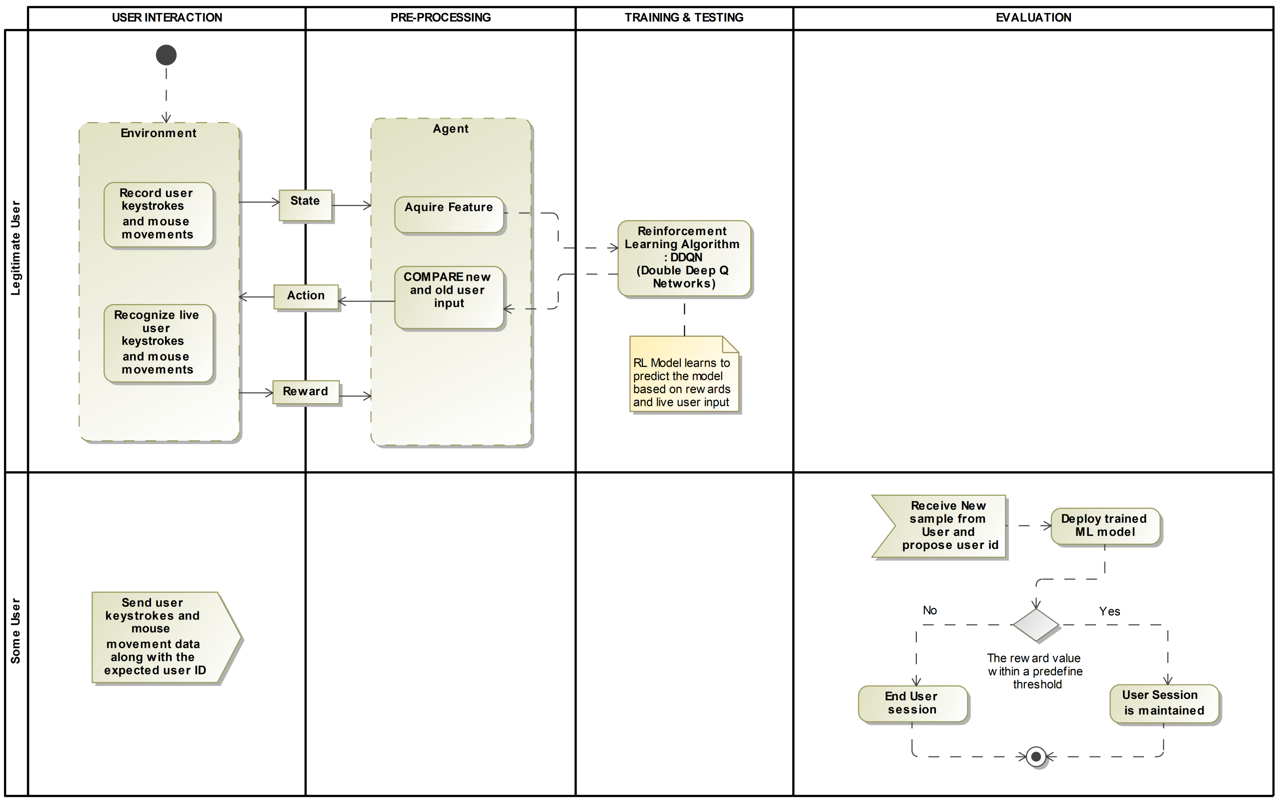
USER INTERACTION
PRE-PROCESSING
TRAINING & TESTING
EVALUATION
Legitimate User
Some User
Environment
Record user keystrokes and mouse movements
Recognize live user keystrokes and mouse movements
State
Action
Reward
Agent
Aquire Feature
COMPARE new and old user input
Reinforcement Learning Algorithm : 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 of reinforcement learning and behavioral biometrics, we have developed a framework as shown in Figure 3 from scratch 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 (RL)-based user authentication system using keystroke dynamics. The detailed 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 [30], 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. **Preprocess 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 (RL) 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 performed using a variety of reinforcement learning (RL) 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:

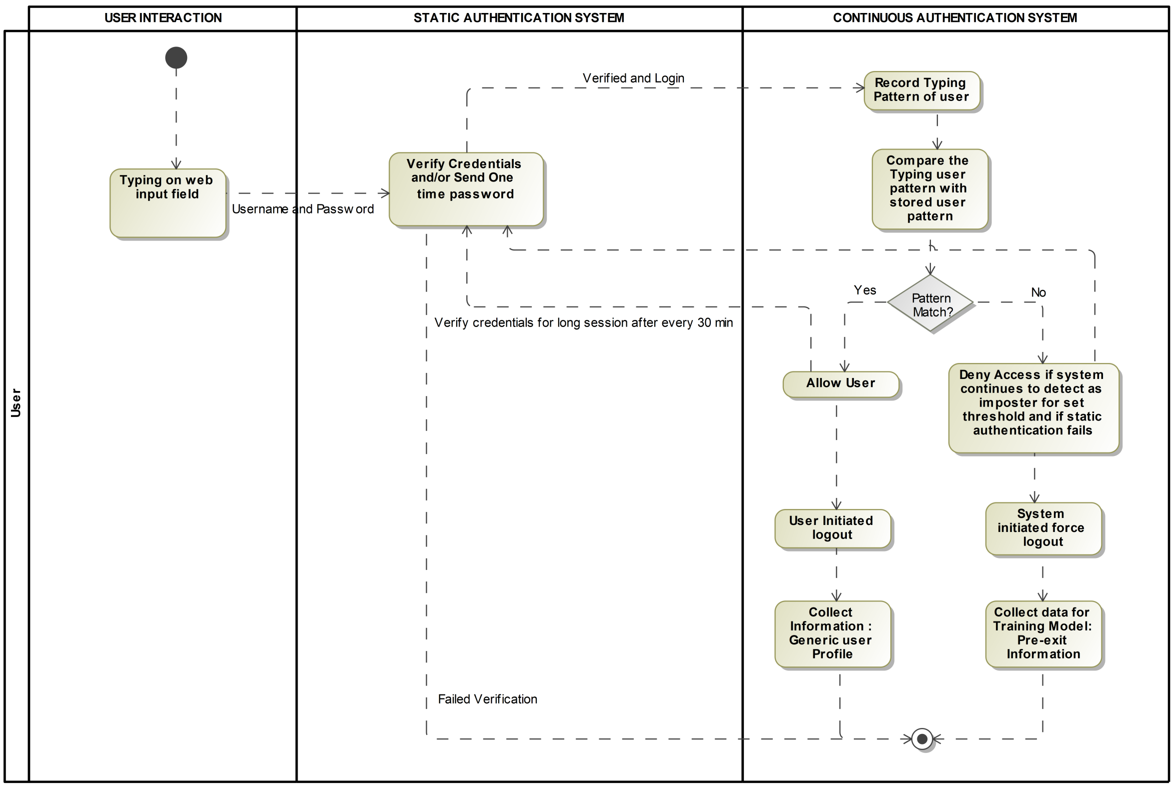
USER INTERACTION
STATIC AUTHENTICATION SYSTEM
CONTINUOUS AUTHENTICATION SYSTEM
User
Typing on web input field
Username and Password
Verify Credentials and/or Send One time password
Verified and Login
Record Typing Pattern of user
Compare the Typing user pattern with stored user pattern
Pattern Match?
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 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. These data are then preprocessed to clean and format them 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 are 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 reward is given for correctly identifying an authorized user and a negative reward is given for incorrectly identifying a hacker.
6. **Train on multiple-episode 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.

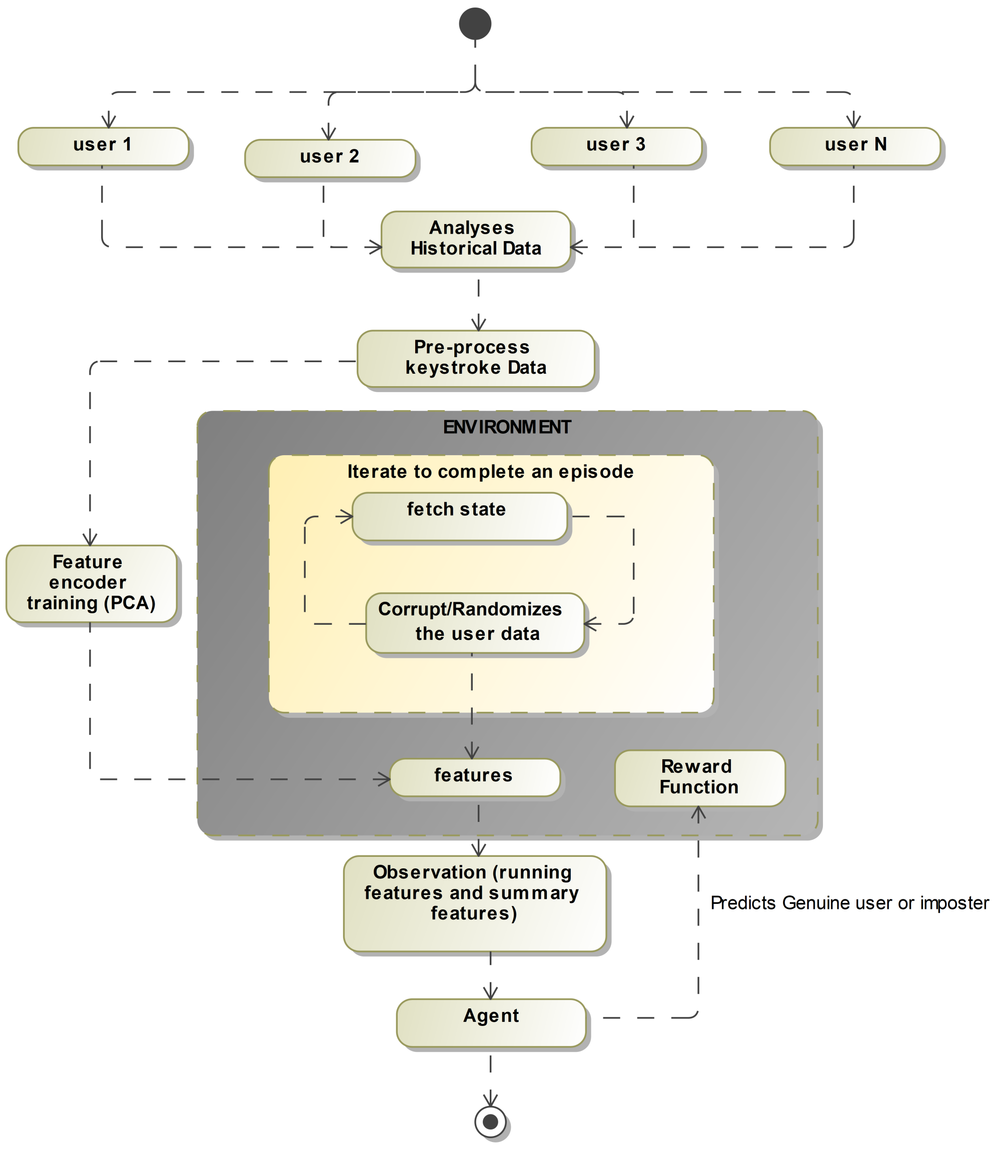
user 1
user 2
user 3
user N
Analyses Historical Data
Pre-process keystroke Data
ENVIRONMENT
Iterate to complete an episode
fetch state
Corrupt/Randomizes the user data
Feature encoder training (PCA)
features
Reward Function
Observation (running features and summary features)
Predicts Genuine user or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

Figure 5. Code flow.

4.1.1. Data Preprocessing

About Original Dataset: The SU-AIS BB-MAS dataset [30] 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 contains data from 117 users performing 6 different activities on 5 different devices. 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 analyzing 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 researched and came up with additional features for training. Figures 6 and 7 below show the time difference between two consecutive events of two different users from the selected dataset.

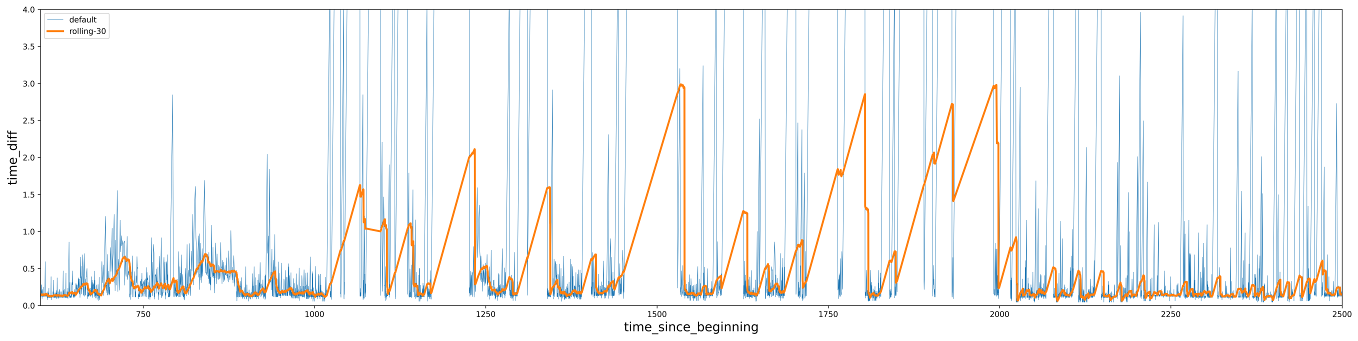

Figure 6. Time (ms) difference between two consecutive events for user 11.

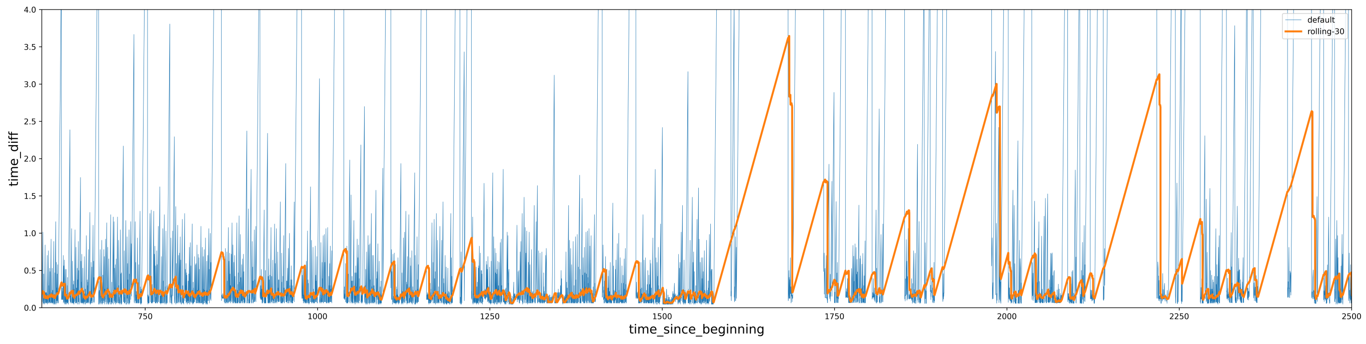

Figure 7. Time (ms) difference between two consecutive events for user 16.

B. Exploratory Data Analysis: Key hold time: The Figure 8 below show the key holding time (ms) for key 't'.

C. Exploratory Data Analysis: Keyboard time length vs. full session: Keyboard time length vs. full session (Figure 9) was an interesting development of all the features. This is violin plot, which is hybrid of a box plot and a kernel density plot, which shows peaks in the data similar to box plot. It reveals that a significant portion of users, approximately 70–98%, spent their session time engaged in keyboard typing. This finding suggests that keyboard interaction is a primary activity during user sessions, which can be used to analyze their level of engagement and productivity.

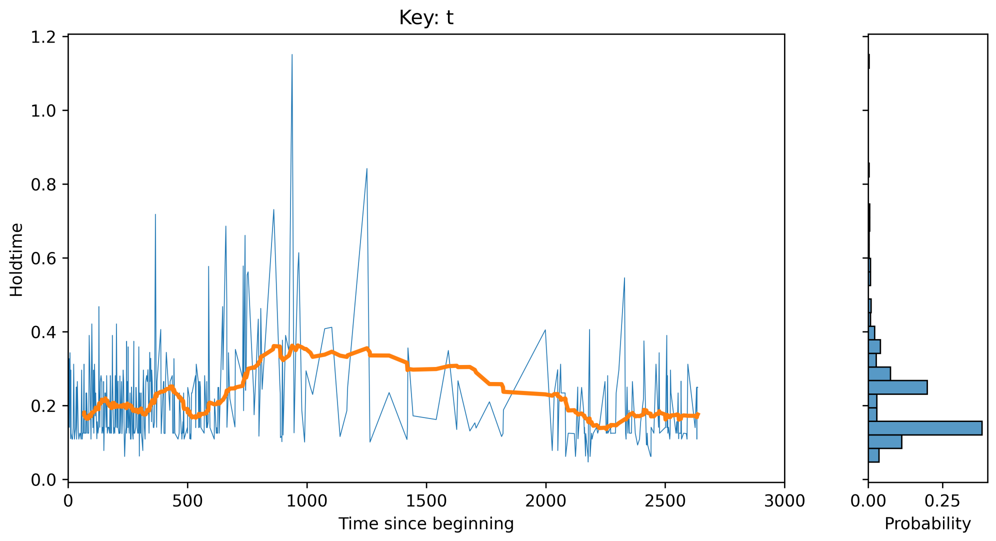

Figure 8. Key ('t') time (ms) for user 11.

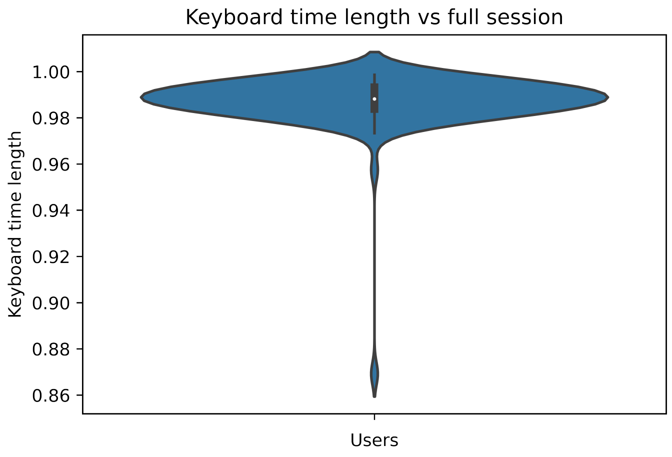

Figure 9. Keyboard time length vs. full session.

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

1. **Standardized key names:** To establish consistent naming conventions to make it easier to read and understand what each event mean [28].
2. **Removed consecutive duplicate pairs (key, direction):** This can help to reduce the number of data the agent needs to process, making the training process more efficient.
3. **Added column "time_diff", which is the time difference between consecutive events:** This can help to capture the unique typing rhythm of an individual, which is a key behavioral biometric.
4. **Added column "time_since_beginning", which is the cumulative sum of the time difference column:** This column is 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 [19].
5. Added new flight features such as press_to_press (P2P), release_to_press (R2P), hold_time (HT).
   press_to_press: Assuming, we have two keys, 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.

4.1.2. Feature Engineering

Running features: Running features are used to capture the dynamics of the user's behavior over time. This feature was useful for the training reinforcement learning (RL) with behavioral biometrics. This allows the agent to learn from the user's behavior over time, which can be important for detecting anomalies or the user's behavioral 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 s, the vector for that event would be [0, 2, 0, . . . 0], where the first value represents the key 'a'.

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 used to capture a summary or aggregate of the user's behavior over multiple events. This allows 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. [31]. 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.

4.2. Environment

The reinforcement learning (RL) agent would interact with the environment by observing the user's keystroke 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 receives feedback from the environment in the form of rewards or penalties. 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. The below equation plays a crucial role in the learning process of RL algorithms by guiding them to make decisions that maximize cumulative rewards over time. By iteratively updating the Q-values through this equation, RL models can learn optimal behavior in an environment by exploring different actions and observing their outcomes, ultimately leading to the acquisition of a strategy that yields the highest reward.

$$Q(s,a) = [1 - Lr]Q(s,a) + Lr[R(s,a) + ymaxQ(ss,aa)] \tag{1}$$

where $Q(s,a)$ represents the Q-value for state-action pair (s,a). Lr is the learning rate. R(s,a) denotes the immediate reward for taking action a in state s. y is the discount factor. ss and aa represent the next state and action.

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 are 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, in 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 [17].

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 [16].

This is necessary so that model is not always predicting the same user. This process is being referred to as corruption [13].

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 performed in three steps:

1. **Calculate running features of the state:** Running features provide the agent with information about the dynamics of the behavior over time.
2. **Encode the running features using trained encoder model:** This helps 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.

4.2.4. Reward Function

The reward function is employed in reinforcement learning (RL) 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. The reward function assigns 1 for true negatives and true positives and 0 for false negatives and false positives.

Several key considerations in designing the reward function for RL in continuous authentication include:

Task-Specifying Objective Reward Function: This reward function evaluates the performance of the agent based on predefined objectives. It is used to assess how well the agent is achieving its goals in the authentication process.

Internal Reward Function: The internal reward function guides the behavior of the agent by providing feedback on actions taken during interactions. It plays a crucial role in shaping the learning process and influencing decision making within the system.

Optimal Internal Reward Function: The optimal internal reward function is determined based on the agent's architecture, limitations, and objectives. It is essential to align this internal reward function with the system's goals to ensure effective learning and decision-making.

Distinguishing Objective and Internal Rewards: Separating the task-specifying objective reward from the internal reward allows for a clear distinction between evaluating performance and guiding behavior. This separation enhances the system's adaptability and performance in continuous authentication scenarios.

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 the 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. 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 reinforcement learning (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 activation 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.

RL Algorithm: DDQN

The Q-learning method, a kind of RL algorithm used 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.

For keystroke dynamics, DDQN would learn to predict the user's keystroke patterns and other behavioral characteristics, and then use this information to decide on whether to authenticate the user. 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 accuracy and effectiveness of the existing authentication system.

In the next section, we evaluate the model's performance.

Table 2. Reasons for choosing double deep Q-network (DDQN) for this task.

Target Network	There are 2 Q-networks, the primary network, and the target network. The target network estimates the Q-values for the next state, which then updates the primary network.
Action Selection	This is dependent on the primary network, and the Q-value estimation is performed 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.

5. Results and Discussion

Evaluation is the process of assessing the performance of the RL model using behavioral biometrics for continuous authentication. The evaluation is performed 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 the GitHub (**GitHub:** https://github.com/PriyaBansal68/Continuous-Authentication-Reinforcement-Learning-and-Behavioural-Biometrics/tree/main/output) (accessed on 18 April 2024) in output folder.

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):

$$Accuracy = (TP + TN)/(TP + TN + FP + FN) \tag{2}$$

where TP is the number of true positives and TN is the number of true negatives.

False Rejection Rate (FRR)/Type I error: This 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 is described in Equation (2):

$$FRR = FN/(FN + TP) = 1 - TPR \tag{3}$$

False Acceptance Rate (FAR)/Type II error: In statistics, FAR is the likelihood that an unauthorized person will obtain 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 is described in Equation (3):

$$FAR = FPR = FP/(FP + TN) \tag{4}$$

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 is described in Equation (4):

$$ERR = (FRR + FAR)/2 \tag{5}$$

ROC Curve: In the context of continuous authentication of behavioral biometrics using reinforcement learning (RL), 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 [9]. 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 combinations of the 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 (Figures 10 and 11, Tables 3–5) of the experiments we performed:

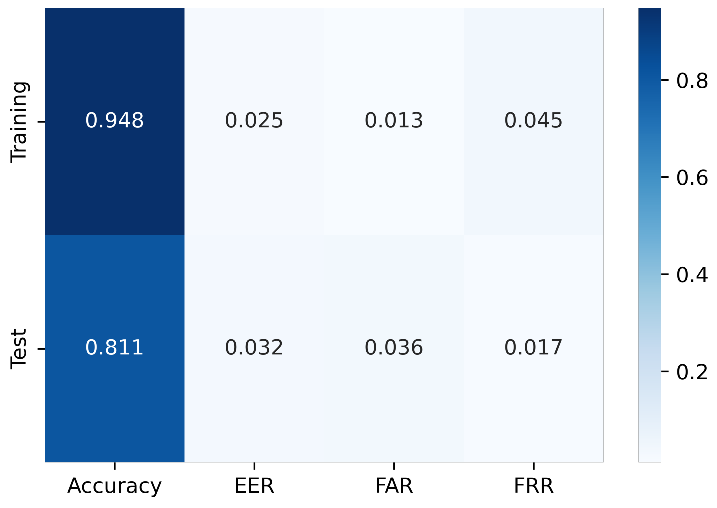

Figure 10. Heatmap results on full dataset of 117 users.

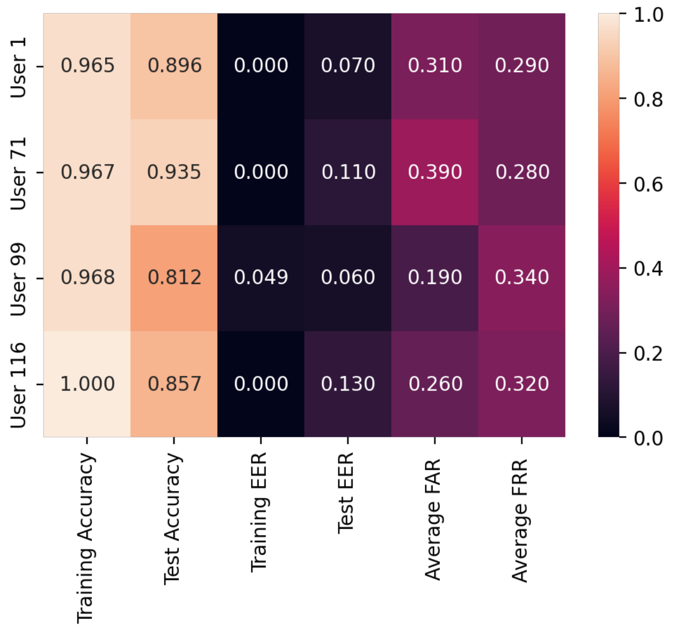

Figure 11. Heatmap results on users (randomly chosen) 1, 71, 99, 116.

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

- In the testing phase, the accuracy achieved was 84.06%, indicating that the system successfully identified users with a high level of accuracy during the testing process.
- The EER achieved during testing was 0.0323, indicating that the system reached a balance where the rates of false acceptance and false rejection were roughly equal.
- In the testing phase, the system had a FAR of 0.0356, suggesting that approximately 3.56% of unauthorized users were incorrectly accepted.
- In the testing phase, the FRR achieved was 0.0174, indicating that around 1.74% of legitimate users were falsely rejected.
- The drop in accuracy from train to test is due to the inconsistent typing behavior of users that affect the model's ability to accurately recognize their keystroke patterns during testing.

It can be observed from the results in Table 4 that the relationship between the user parameters and the accuracy of the model is not linear. Several factors that lead to this nonlinear relationship are as below:

1. **Individual Differences**: Each user has different characteristics, behaviors, or patterns that affect the performance of the model. These individual differences result in variations in accuracy, even if the user parameters are changing linearly.
2. **Exploration vs. Exploitation**: Users need to balance exploration (trying out new actions to learn more about the environment) and exploitation (leveraging known actions for optimal results). The user with 100% accuracy may have effectively balanced exploration and exploitation strategies, leading to superior performance during training. In contrast, the user with 96.5% accuracy might have faced difficulties in finding the right balance, resulting in slightly lower overall accuracy.
3. **Complexity of User Parameters/Behavior**: The user parameters considered in the table capture intricate interactions and dependencies. As a consequence, minor changes in these parameters lead to substantial fluctuations in the model's performance, resulting in nonlinear relationships between the parameters and accuracy.

4. **Data Availability and Quality**: The number and quality of data available for each user parameter differ. Insufficient or noisy data for certain parameter values affect the model's ability to accurately learn and generalize, leading to nonlinear accuracy trends.

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

Training Accuracy/EER: Figure 12 shows the graphical representation of the training phase of user 116. It shows that with each iteration of the episode, the agent tries to increase rewards by exploring the environment and targets to reduce the EER.

Figure 13 shows how the EER is decreasing with each iteration for four randomly chosen users.

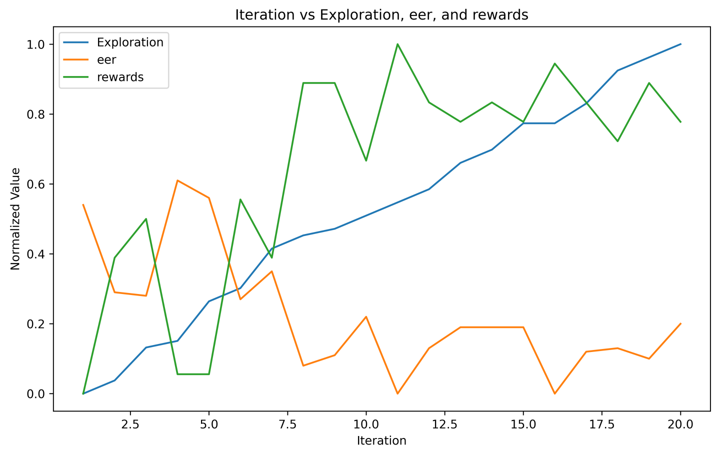

Figure 12. Iterations for user 116 vs. exploration, rewards, and EER.

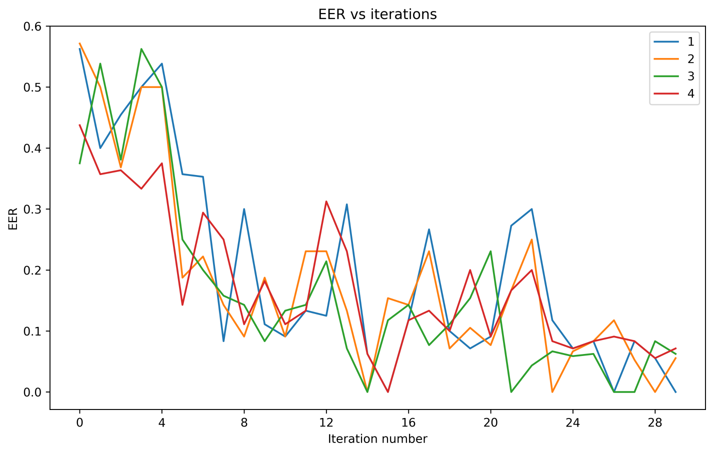

Figure 13. Graph for users 1, 2, 3, and 4 showing decreasing EER with no. of iterations.

Test Accuracy/EER: Figure 13 shows the testing performance of the continuous authentication system on four randomly chosen users.

This test 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 [21]. 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 [26].

All the factors listed for Table 4 are significant reasons for the difference in train and test phases of the ROC curve (Figures 14 and 15) at user level. For ideal scenario, the ROC in training and testing should look alike. The major factor for difference in ROC curves is the user behavior, which is not consistent due to multiple external factors. In this case, the model is able to recognize the pattern of users 1, 2, and 3 (above the assumed threshold of 75%) in comparison to user 4. Hence, the model can retrained for user 4.

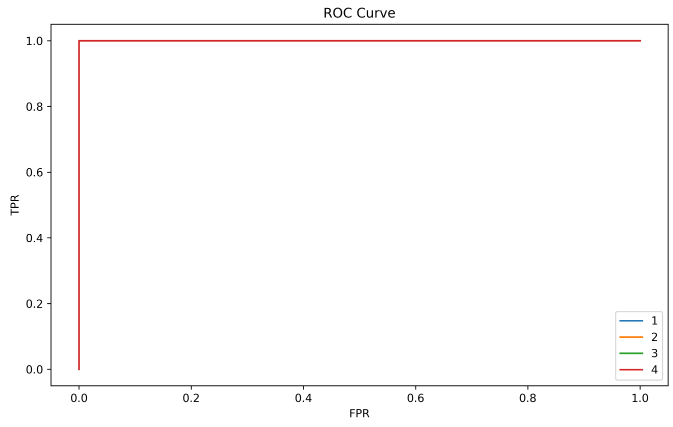

Figure 14. (TRAIN) ROC curve for users 1, 2, 3, and 4.

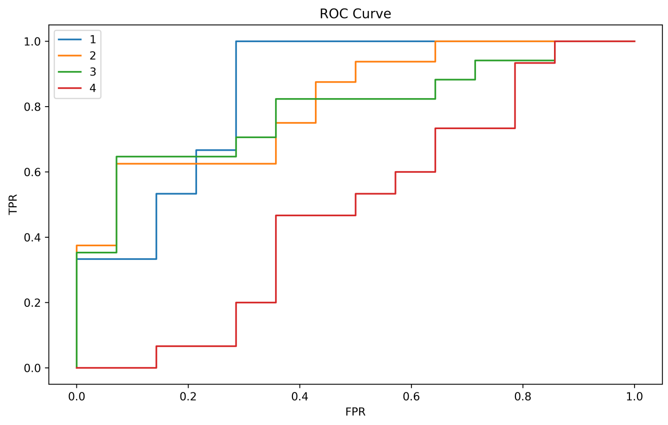

Figure 15. (TEST) ROC curve for users 1, 2, 3, and 4.

Comparison of Supervised Learning vs. Reinforcement Learning results:

Table 5 shows the comparison between supervised learning and reinforcement learning. The same dataset was used to perform this comparison. It was observed that the reinforcement learning accuracy is greater even when all the keys are used. We also made a comparison to one of the literature review who had used subset of the same dataset used in this study. The literature review [30] used a subset of the dataset and used 10 unigraphs and 5 digraphs. The EER in the case of reinforcement learning (RL) is quite lower than in the case of supervised learning for both of these studies. Figure 16 shows the heatmap representation of train accuracy, test accuracy and EER of these three results. Continuous authentication systems employ the EER as a crucial performance metric, quantifying the balance between false acceptance and false rejection rates. Lower EER values indicate superior performance, presenting multiple advantages for continuous authentication.

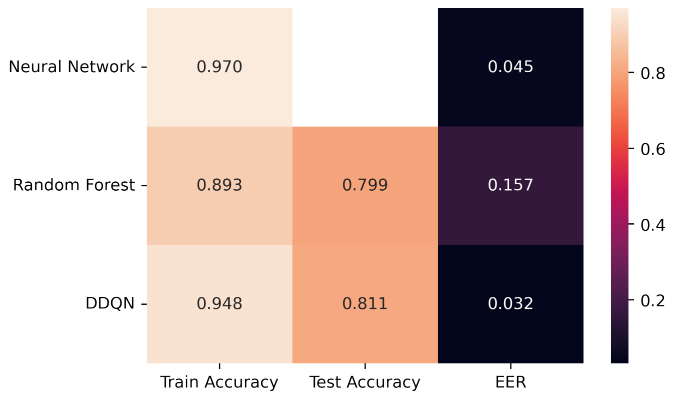

Figure 16. Heatmap results for comparison between supervised learning and reinforcement learning.

Table 5. Supervised learning vs. reinforcement learning result.

Metrics	Literature Review [21]	State-of-the-Art Model [23]	This Study (SL)	This Study (RL)
ML type	Supervised learning	Supervised learning	Supervised learning	Reinforcement learning
Dataset	SU-AIS BBMAS subset	Buffalo and Clarkson dataset	SU-AIS BBMAS [30]	SU-AIS BBMAS [30]
No. of users	102	54	117	117
Keys	10 unigraphs, 5 digraphs	NA	All keys	All keys
Model	Neural network	Neural network	Random forest	DDQN
Train/test accuracy	97%/NA	88%	89.34%/79.89%	94.77%/81.06%
EER	0.03–0.06 = average 0.045	NA	0.157	0.0323

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. As a result, 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 is effective in identifying each user.

RL models can be deployed on the client side where the model can adapt to learn the change in user behavior, and diminishes the need to retrain the model, unlike supervised learning models.

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.

Moreover, to conclude, we achieved benchmark results on Keystroke dynamics using RL, on the full dataset with overall Training and test accuracy as 94.77% and 84.06% and EER as 0.0255 and 0.0323, respectively.

As an addition, the dependency on the data would decrease as the model would learn eventually to recognize the pattern on its 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 are 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 are highly variable and subject to noise.

Below are some of the areas where the continuous authentication applications proves to be very beneficial:

1. **Healthcare:** In a hospital setting, medical professionals access sensitive patient data on computers located in public areas. With continuous authentication, the system verifies 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 prevents 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 is particularly useful in remote work environments, where employees may be working from unsecured locations or using unsecured devices. It is also used to track employee's productivity of the day.
4. **Behavioral Profiling for User Insights:** Keystroke dynamics are used for behavioral profiling to gain insights into user behavior, preferences, and typing habits, helping in personalized user experiences and targeted marketing.

Future Work

The proposed reinforcement learning model for continuous authentication using behavioral biometrics has potential for future improvements.

1. **Autoencoder feature encoder**: In the current model, a PCA model is used as a feature encoder. However, autoencoders can also be used as a feature encoder. Autoencoders are neural networks that are trained to reconstruct their inputs. They can be used to reduce the dimensionality of the data and extract features that are important for the task. Using an autoencoder as a feature encoder can improve the accuracy of the model by reducing the dimensionality of the data and extracting important features.

2. **Augmentation techniques**: In the current model, the data are not augmented. However, augmentation techniques such as adding small noise to the data can be used to improve the robustness of the model. Augmenting the data can help the model to generalize better to new data and make it more robust to variations in the data.

3. One way to carry this out could be to **use a large language model** (LLM) to generate text that the user would type, and then use reinforcement learning to authenticate the user based on their typing patterns. This could involve pretraining on a large dataset of text and then fine-tuning it on a smaller dataset of text that is specific to the user. The user would then be prompted to type the text generated by LLM models and their typing patterns would be analyzed by the reinforcement learning model.

However, there can be some challenges that might occur in adopting such applications in the real world, related to privacy concerns, variability and adaptability of user data, robustness and anti-spoofing measures, etc.

The article concludes with a call to action for researchers, practitioners, and developers to collaborate and advance the field of continuous authentication using keystroke dynamics, highlighting the importance of this technology in ensuring secure and reliable access to information systems.

Author Contributions: Conceptualization, P.B. and A.O.; methodology, P.B. and A.O.; software, P.B.; validation, P.B. and A.O.; formal analysis, P.B. and A.O.; investigation, P.B. and A.O.; resources, P.B. and A.O.; data curation, P.B. and A.O.; writing—original draft, P.B.; writing—review and editing, P.B. and A.O.; project administration, A.O.; funding acquisition, A.O. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) [Grant no. RGPIN-2018-06250]. These support is greatly appreciated.

Data Availability Statement: The developed RL gym-based environment code is made available to the domain researchers on the GitHub: https://github.com/PriyaBansal68/Continuous-Authentication-Reinforcement-Learning-and-Behavioural-Biometrics/tree/main/output (accessed on 18 April 2024).

Acknowledgments: The authors acknowledge the NSERC research support from Western University in Canada.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Gold, J. Traditional Security Is Dead—Why Cognitive-Based Security Will Matter. 2016. Available online: https://www.computerworld.com/article/3068185/traditional-security-is-dead-why-cognitive-based-security-will-matter.html (accessed on 25 May 2023).
2. Ventures, C. Cybercrime to Cost the World $10.5 Trillion Annually by 2025. 2020. Available online: https://www.prnewswire.com/news-releases/cybercrime-to-cost-the-world-10-5-trillion-annually-by-2025--301172786.html (accessed on 25 May 2023).
3. Anderson, R. 67 Percent of Breaches Caused by Credential Theft, User Error, and Social Attacks. 2020. Available online: https://www.netsec.news/67-percent-of-breaches-caused-by-credential-theft-user-error-and-social-attacks/ (accessed on 25 May 2023).
4. Burbidge, T. Cybercrime Thrives during Pandemic: Verizon 2021 Data Breach Investigations Report. 2021. Available online: https://www.verizon.com/about/news/verizon-2021-data-breach-investigations-report (accessed on 25 May 2023).
5. Ginni. What Are the Disadvantage of Multi-Factor Authentication? 2022. Available online: https://www.tutorialspoint.com/what-are-the-disadvantage-of-multi-factor-authentication (accessed on 25 May 2023).
6. Ouda, A. A framework for next generation user authentication. In Proceedings of the 2016 3rd MEC International Conference on Big Data and Smart City (ICBDSC), Muscat, Oman, 15–16 March 2016; pp. 1–4. [CrossRef]
7. Voege, P.; Abu Sulayman, I.I.; Ouda, A. Smart chatbot for user authentication. *Electronics* **2022**, *11*, 4016. [CrossRef]
8. Abu Sulayman, I.I.M.; Ouda, A. Designing Security User Profiles via Anomaly Detection for User Authentication. In Proceedings of the 2020 International Symposium on Networks, Computers and Communications (ISNCC), Montreal, QC, Canada, 20–22 October 2020; pp. 1–6. [CrossRef]
9. Li, J.; Chang, H.C.; Stamp, M. Free-Text Keystroke Dynamics for User Authentication. In *Artificial Intelligence for Cybersecurity*; Springer: Berlin/Heidelberg, Germany, 2022; pp. 357–380.
10. Verma, N.; Prasad, K. Responsive parallelized architecture for deploying deep learning models in production environments. *arXiv* **2021**, arXiv:2112.08933.
11. Salem, A.; Zaidan, D.; Swidan, A.; Saifan, R. Analysis of strong password using keystroke dynamics authentication in touch screen devices. In Proceedings of the 2016 Cybersecurity and Cyberforensics Conference (CCC), Amman, Jordan, 2–4 August 2016; pp. 15–21.
12. Jeanjaitrong, N.; Bhattarakosol, P. Feasibility study on authentication based keystroke dynamic over touch-screen devices. In Proceedings of the 2013 13th International Symposium on Communications and Information Technologies (ISCIT), Surat Thani, Thailand, 4–6 September 2013; pp. 238–242.
13. Antal, M.; Szabó, L.Z. An evaluation of one-class and two-class classification algorithms for keystroke dynamics authentication on mobile devices. In Proceedings of the 2015 20th International Conference on Control Systems and Computer Science, Bucharest, Romania, 27–29 May 2015; pp. 343–350.
14. Roh, J.h.; Lee, S.H.; Kim, S. Keystroke dynamics for authentication in smartphone. In Proceedings of the 2016 International Conference on Information and Communication Technology Convergence (ICTC), Jeju, Korea, 19–21 October 2016; pp. 1155–1159.
15. Saevanee, H.; Bhattarakosol, P. Authenticating user using keystroke dynamics and finger pressure. In Proceedings of the 2009 6th IEEE Consumer Communications and Networking Conference, Las Vegas, NV, USA, 10–13 January 2009; pp. 1–2.
16. Monrose, F.; Rubin, A.D. Keystroke dynamics as a biometric for authentication. *Future Gener. Comput. Syst.* **2000**, *16*, 351–359. [CrossRef]
17. Araújo, L.C.; Sucupira, L.H.; Lizarraga, M.G.; Ling, L.L.; Yabu-Uti, J.B.T. User authentication through typing biometrics features. *IEEE Trans. Signal Process.* **2005**, *53*, 851–855. [CrossRef]
18. Antal, M.; Nemes, L. The mobikey keystroke dynamics password database: Benchmark results. In *Software Engineering Perspectives and Application in Intelligent Systems: Proceedings of the 5th Computer Science Online Conference 2016 (CSOC2016)*; Springer: Berlin/Heidelberg, Germany, 2016; Volume 2, pp. 35–46.
19. Stragapede, G.; Vera-Rodriguez, R.; Tolosana, R.; Morales, A. BehavePassDB: Public Database for Mobile Behavioral Biometrics and Benchmark Evaluation. *Pattern Recognit.* **2023**, *134*, 109089. [CrossRef]
20. Siddiqui, N.; Dave, R.; Vanamala, M.; Seliya, N. Machine and deep learning applications to mouse dynamics for continuous user authentication. *Mach. Learn. Knowl. Extr.* **2022**, *4*, 502–518. [CrossRef]

21. Belman, A.K.; Sridhara, S.; Phoha, V.V. Classification of threat level in typing activity through keystroke dynamics. In Proceedings of the 2020 International Conference on Artificial Intelligence and Signal Processing (AISP), Amaravati, India, 10–12 January 2020; pp. 1–6.
22. 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, NY, USA, 2016; pp. 175–193. [CrossRef]
23. Pawel Kasprowski, Borowska, Z.B.; Harezlak, K. Biometric Identification Based on Keystroke Dynamics. *Sensors* **2022**, *22*, 3158. [CrossRef] [PubMed]
24. Sutton, R.S.; Barto, A.G. *Reinforcement Learning: An Introduction*; A Bradford Book: Cambridge, MA, USA, 2018.
25. Ananya; Singh, S. Keystroke dynamics for continuous authentication. In Proceedings of the 2018 8th International Conference on Cloud Computing, Data Science & Engineering (Confluence), Noida, India, 11–12 January 2018; pp. 205–208.
26. Gupta, S. User Attribution in Digital Forensics through Modeling Keystroke and Mouse Usage Data Using Xgboost. Ph.D. Thesis, Purdue University Graduate School, West Lafayette, IN, USA, May 2022.
27. Çevik, N.; Akleylek, S.; Koç, K.Y. Keystroke Dynamics Based Authentication System. In Proceedings of the 2021 6th International Conference on Computer Science and Engineering (UBMK), Ankara, Turkey, 15–17 September 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 Things J.* **2020**, *7*, 9128–9143. [CrossRef]
29. Bansal, P.; Ouda, A. Study on Integration of FastAPI and Machine Learning for Continuous Authentication of Behavioral Biometrics. In Proceedings of the 2022 International Symposium on Networks, Computers and Communications (ISNCC), Shenzhen, China, 19–22 July 2022; pp. 1–6.
30. 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. Available online: https://ieee-dataport.org/open-access/su-ais-bb-mas-syracuse-university-and-assured-information-security-behavioral-biometrics (accessed on 18 April 2024).
31. Huang, J.; Hou, D.; Schuckers, S.; Law, T.; Sherwin, A. Benchmarking keystroke authentication algorithms. In Proceedings of the 2017 IEEE Workshop on Information Forensics and Security (WIFS), Rennes, France, 4–7 December 2017; pp. 1–6.