4-18-2023 12:30 PM

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

Priya Bansal, *The University of Western Ontario*

Supervisor: Dr. Abdelkader Ouda, *The University of Western Ontario*

A thesis submitted in partial fulfillment of the requirements for the Master of Engineering Science degree in Electrical and Computer Engineering

Abstract

This study aims to develop a design and architecture of continuous authentication system using behavioral biometrics for recognizing users accessing computing devices. The proposed system utilizes keystroke dynamics as a behavioral biometric, and reward-based reinforcement learning (RL) concepts are applied to recognize users throughout the session. The purpose of this research is to investigate the effectiveness of behavioral biometrics for user authentication using reinforcement learning. This study applies feature extraction techniques to enhance performance metrics.

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 reward function is binary and minimalistic. A PCA (Principal Component Analysis) model is used to encode the running features, and a DDQN (Double Deep Q-Network) 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.

The proposed system provides an additional layer of security to traditional authentication methods, forming a robust continuous authentication system utilizing the concepts of reinforcement learning that can be added to existing static authentication systems.

Keywords

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

Summary for Lay Audience

Continuous authentication is a way to make sure that someone is who they say they are while they use a computer or other digital device. It's important because there are many bad people who try to steal information or cause problems online.

This study looked at using a type of computer learning called "reinforcement learning" and a way of analyzing a person's behavior called "behavioral biometrics" to make a better continuous authentication system. Reinforcement learning helps a computer learn to make decisions based on what it learns from trying different things, while behavioral biometrics looks at things like how someone types to figure out if it's really them.

The study found that using these methods together was better at detecting when someone was trying to pretend to be someone else than other ways of doing it. This means that it can help keep digital systems safe and secure without being too inconvenient for people to use. It can also help prevent bad people from accessing someone's computer if they leave it unattended.

Dedication

I would like to express my gratitude to my Family for their unwavering support throughout my academic journey. My siblings have been a constant source of encouragement and have played a crucial role in helping me shape my identity as a woman. I also extend my heartfelt thanks to my spouse, whose unwavering support and motivation have been instrumental in my success. This thesis is dedicated to my loved ones who have played an integral part in my personal and academic growth.

Acknowledgments

First and foremost, I express my gratitude to God for bestowing upon me the ability to achieve my goals and guiding me in my pursuit of knowledge. He has been my protector throughout my academic journey and the sole source of all my accomplishments.

I would also like to extend my heartfelt thanks to my beloved parents, who have consistently offered their daily supplications for my success and well-being in all aspects of my life. Furthermore, I am deeply grateful to my husband for his unwavering patience, support, and encouragement throughout my studies.

My sincere appreciation goes to Dr. Abdelkader Ouda, my supervisor during my MESc. program, for his invaluable guidance, insightful advice, and constructive feedback. His academic expertise has significantly enhanced my research skills and his motivation and support have bolstered my confidence and determination to achieve my goals.

Finally, I acknowledge and appreciate the support provided by Mitacs.

Table of Contents

List of Tables

List of Figures

List of Acronyms

Abbreviation	Meaning
ML	Machine Learning
RL	Reinforcement learning
MDP	Markov decision process
EER	Equal Error rate
FRR	False Rejection rate
FAR	False Acceptance Rate
SA	Static authentication
CSA	Continuous and static authentication
MFA	Multi-factor Authentication
OTP	One-time password
TP	True Positives
TN	True Negatives
FP	False Positives
FN	False Negatives

Chapter 1

1. Introduction

In today's fast-paced business environment, traditional methods of security are becoming [1] increasingly inadequate. 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 [1] in place, employees may still inadvertently compromise security by sharing passwords or digital keys. 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 effective in preventing cyberattacks. In fact, according to the Verizon 2021 Data Breach Investigations Report, 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. 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 reliable system in place 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 [2], it is important for the system to continuously authenticate users in addition to static authentication methods.

The purpose of this thesis is to design and develop a dynamic, continuous authentication system that provides a high level of security and reliability for businesses. The system will leverage machine learning algorithms and behavioral biometrics to continuously authenticate users, reducing the risk of data breaches and unauthorized access. The system will be evaluated against traditional static and dynamic authentication methods to measure its effectiveness in preventing cyberattacks and enhancing security. The goal of this research is to provide businesses with a trusted authentication system that can protect their sensitive data and assets in the face of sophisticated cyberattacks.

1.1 Overview on Authentication

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, where users enter their credentials at the time of logging in to the system, and the verification happens in the backend database. If the credentials match, the user is granted access to the system, otherwise, the user is denied access.

While static authentication methods have indeed become less secure over time, it is not necessarily true that all current methods are vulnerable to online fraud. For example, multi-factor authentication (MFA) has been shown to be a much more effective way of securing user accounts than single-factor authentication. However, even MFA is not immune to certain types of attacks, such as phishing or SIM swapping. Therefore, it is important to continually evaluate and update security measures to stay ahead of evolving threats. [3]. This has led to a growing need for a more robust authentication system that can secure applications in today's fast-paced environment, especially with the increase in work from home scenarios.

Continuous authentication, on the other hand, estimates the likelihood that the user accessing the system throughout the session is the same user who initially claimed to be. This is done by analysing the user's behaviour, 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.

Therefore, 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 behaviour to ensure that the same person is accessing the system throughout the session [4].

Table 1: Benefits of using continuous and static authentication (CSA) over only current static authentication solutions (SA):

Parameter	**Continuous and Static authentication (CSA)**	**Static authentication (SA)**	**Evidence**
Enhanced security	An effective continuous authentication system considers various environmental and human factors, instead of solely relying on trustworthiness. CSA provides an additional layer of security by continuously monitoring the user's behavior and ensuring that the same person is accessing the system throughout the session. This can help to prevent unauthorized access and protect against various types of cyberattacks, such as phishing and stuffing.	Comparing one-time authenticated logins to checking someone in at a gate, it is difficult to monitor their activities once they gain access to the system. Static authentication methods are insufficient to handle the dynamic nature of modern environments, such as multi-terminal access, shared accounts, and remote work setups. Hackers have found ways to bypass inactivity monitoring, such as by simulating user activity or by stealing authentication credentials. Therefore, while inactivity monitoring can be a	• According to the Verizon Data Breach Investigations Report 2020, stolen credentials are still the leading cause of data breaches, accounting for 80% of hacking-related breaches. • A study by the Ponemon Institute found that 65% of attacks against small and medium-sized businesses were due to employee negligence, such as weak passwords and using public Wi-Fi.

		useful tool, it should be combined with other security measures to provide a comprehensive security solution.	
Reduced risk of fraud	By continuously monitoring the user's behavior, CSA can detect any anomalies or changes in behavior that may indicate a potential fraud attempt. This can help to reduce the risk of financial and identity-related fraud.	Static authentication poses a risk of fraud, as it relies on a fixed password or other unchanging credentials, which can be compromised through various means such as theft, phishing, or other malicious attacks.	• A report by the Association of Certified Fraud Examiners found that organizations lose 5% of their revenue to fraud each year. • According to the Identity Theft Resource Center, there were 1,108 data breaches in 2020, exposing over 300 million records.
Improved user experience	CSA does not require the use of external devices such as multi-factor authentication, which can be interruptive and time-consuming for the user. Instead, it analyses the user's behavior without interruption, providing a more seamless and user-friendly experience.	The User experience is not that good. In most cases, users are only required to perform MFA once at the beginning of a session. However, in some cases, such as when accessing sensitive information or performing critical	• A survey conducted by HYPR found that 93% of consumers prefer biometric authentication over passwords, citing convenience as a primary reason. • A study by IBM found that the

		tasks, users may be required to re-authenticate at specific intervals for added security.	average person has to remember 191 passwords, leading to password fatigue and frustration.
Cost-effective	Implementing CSA does require the use of additional hardware depending upon the level of security needed and number of people, such as tokens or smart cards, which can be expensive to deploy and manage. This can make CSA a more cost-effective solution for organizations.	Additional hardware is needed to authenticate the user which has cost factor attached to it. However, the impact of the cost factor of using hardware in security systems is dependent on various factors such as the specific type of hardware, the size of the organization, and the level of security required. The cost of hardware-based authentication systems can vary widely, depending on the type of hardware and the specific implementation, but it can be substantial for some organizations.	• According to a report by the Identity Theft Resource Center, the average cost of a data breach in the US was $8.19 million in 2019. • The cost of a security token can range from $15 to $50 per user, depending on the type and quantity purchased.

Greater flexibility	Organizations can enhance their authentication processes by implementing CSA alongside a range of other methods, including passwords, biometrics, and multi-factor authentication. This approach offers greater flexibility, enabling organizations to select the most suitable authentication method according to their specific requirements.	Static authentication systems rely on a fixed set of credentials, such as a username and password, for user authentication. However, if the user needs to change their device or access the system from a different location, they may need to go through a tedious process to update their authentication details. This lack of flexibility can be a major inconvenience for users, especially in today's fast-paced and mobile work environments where users often need to switch devices or work remotely.	• According to a survey by LastPass, 95% of businesses use more than one method of authentication, with 45% using three or more methods. • A report by Gartner predicts that by 2022, 60% of large and global enterprises, and 90% of midsize enterprises, will implement password less methods in more than 50% of use cases.

This study presents a new approach to continuous authentication using a reinforcement learning-based anomaly detection method to be added with current existing security architectures to enhance the security. The proposed model utilizes reinforcement learning techniques to identify and authenticate users in real-time, by continuously monitoring their behaviour.

1.2. Research Motivation:

The below steps explain in detail what motivated us to further research in this domain:

- When logging in to a computer or website, the most frequent interaction with an authentication mechanism involves entering a password. Once we type in the password, the operating system verifies it, and then grants access to our session. For example, when logging into an online banking website, a user enters their username and password to access their account [19].
- In certain scenarios, a second authentication factor such as a physical device, one-time passcode, or biometric sample may be necessary. However, all these authentication methods have a common limitation: they authenticate the user only once. Once the authentication is successful, there are no further checks or restrictions to prevent unauthorized access to the resource. For example, when logging into a sensitive system or network, a user may be required to enter a password and then provide a fingerprint scan for additional authentication.
- When a user leaves their computer unattended, there is a risk that an unauthorized person may gain access to it. This raise concerns in environments where sensitive data is stored, such as hospitals or financial institutions with millions of bank accounts. For example, if an employee leaves their workstation unlocked and unattended, a colleague or outsider could easily access their files and data.
- To mitigate this risk, organizations need to implement robust security policies that go beyond the initial authentication. Continuous authentication is a mechanism that addresses this issue by regularly verifying the user's authenticity throughout the session. This approach aims to prevent common security breaches, including tailgating and piggybacking [19], where an unauthorized person gains access to an unattended computer. For example, a system could monitor a user's behavior while they are logged in, such as their typing patterns or mouse movements, and flag any abnormal activity as potential unauthorized access.

By implementing continuous authentication, organizations can reduce the risk of data breaches and protect their networks from security threats, particularly at the most vulnerable endpoint - the user. Continuous authentication could help prevent hackers from accessing

sensitive data even if they manage to bypass the initial authentication step, such as by stealing a user's password.

Consequently, the most effective continuous authentication mechanism should incorporate multiple authentication schemes to optimize the end-user experience and enhance security for administrators. Moreover, it should operate seamlessly in the background without any user intervention while running continuously. Importantly, the user should be unaware of the mechanism, ensuring complete transparency. All the recent hacks/outages happened due to the organization’s ignoring the importance of continuous authentication in their application. Hence, to meet the needs of the changing market for continuous authentication motivated us to research further on this topic from a different perspective and reach a solution which is easily integrated with current system without the need of any new hardware and is also cost-effective.

The motivation for this work stems from the limitations of traditional methods of user authentication, which have become increasingly ineffective due to the rise of sophisticated cyberattacks. Additionally, current methods for keystroke dynamics-based authentication are not well-suited for continuous authentication, as they do not consider the dynamic nature of user behavior over time (discussed in Chapter 4).

The main objective of this study is to explore the potential of reinforcement learning for continuous authentication of behavioral biometrics.

1.3. Research Objective:

The main objective of this study is to design and develop a solution that leverages keystroke dynamics for user authentication in a web environment. The proposed solution aims to aid in the detection of potential illegitimate users by building user behavior profiles using the Markov Decision Process (MDP) [15]. The goal is to automatically and accurately detect whether the user is genuine or an imposter.

- The study will centre on creating a behavioral model of the authorized user's computer usage patterns to identify non-conforming activities. This involves scrutinizing typing, website activity, and mouse movements to ascertain the user's presence. By creating a unique behavioral signature, such as typing speed or rhythm, the authentication software can prevent unauthorized access by detecting deviations in the user's real-time computer usage compared to their behavioral model.
- To re-iterate as stated earlier, the goal of this research is to design and implement an accurate and comprehensive model for user authentication and identification. This can help to secure user accounts and can improve the security and privacy of online systems, as well as the user experience.

1.4. Research Methodology:

- In this study, the research methodology will employ sequential decision-making techniques from the domain of machine learning. Sequential decision-making refers to the process of using prior experiences to determine the sequence of actions to accomplish objectives in an ambiguous environment. This method has broad possibilities for application in various fields, including security, finance, healthcare, robotics, smart grids, self-driving cars, and others.
- The research focuses on the domain of behavioral biometrics and utilizes reinforcement learning concepts to solve the problem of continuous authentication. This approach has been chosen after thorough research, as it has the potential to overcome some of the limitations of traditional methods, such as supervised learning techniques.
- The solution proposed aims to create a behavioural model of the authenticated user to identify non-conformal computer usage over time, by analysing typing to determine the user's presence. The model will be built using reinforcement learning techniques (explained in detailed in chapter 4) to establish the user's behavioural signature, such as typing dynamics or "typing rhythm." It will then be capable of preventing access when real-time use of the computer deviates from the established behavioral model.

In addition to the above points, there are benefits (Table 3) of using RL over Supervised and unsupervised.

Table 2: RL over Supervised and unsupervised:

Based on	RL over supervised and unsupervised
Adaptability	RL can adapt to changes in user behavior over time, making it suitable for continuous authentication. This has been demonstrated in various studies where RL algorithms have been used to learn the evolving patterns of user behavior [15].
Dynamic decision-making	RL can make dynamic decisions based on the current state of the system, allowing for real-time identification and authentication of users. This is important in scenarios where prompt action is required, such as fraud detection.
Handling uncertainty	RL is well-suited for handling uncertainty and dealing with unknown or unseen situations. This is important in the case of keystroke dynamics, where a user's behavior may change over time or in different contexts.
Exploration	RL can explore different options and strategies to learn from its own experiences, rather than relying on pre-labelled data.[15] This allows for more robust and accurate authentication.
Reward-based learning	RL can learn from reward signals, which are useful in the context of authentication where correct decisions are rewarded, and incorrect decisions are penalized. This has been shown to improve the accuracy of RL-based authentication systems [15]
Scalability	RL can be applied to large and complex systems, such as continuous authentication of keystroke dynamics, making it more scalable than supervised and unsupervised learning methods. This is particularly relevant in modern-day scenarios where user behavior is dynamic and constantly evolving.
Continuous learning	RL allows the agent to learn from its past actions and improve its decision-making over time. This is particularly useful in the context of continuous authentication, where the user's behavior may change over time.

	This capability ensures that the authentication system is up-to-date and effective in detecting fraud.

To achieve the discussed objective, we have implemented a reinforcement learning-based anomaly detection model to identify and authenticate users in real-time. This includes the use of Q-learning algorithm, which is a popular RL algorithm for solving MDPs [15], as well as the implementation of a deep neural network to approximate the Q-function (All details about this section are discussed in detail in chapters 3-4).

Objective is to evaluate the proposed model using real-world data, and a comparison with existing methods. To do this, we have used a dataset of keystroke dynamics [14] from multiple users and used it to train and test the proposed model. Details about data have been discussed in chapter 4. We have also compared the performance of our model with traditional methods of user authentication such as supervised and unsupervised learning algorithms. The performance of the proposed RL model is evaluated based on metrics such as accuracy, False Acceptance Rate (FAR), Equal error rate (EER) and False rejection rate (FRR).

1.5. Research Contribution:

To achieve the objective, the following was investigated:

1. The current state of the art in continuous authentication, with a focus on behavioural biometrics (keystroke dynamics, mouse etc).
2. The potential of reinforcement learning for continuous authentication of keystroke dynamics, including the challenges and limitations of this approach.
3. The development and evaluation of a novel reinforcement learning-based anomaly detection model for continuous authentication of keystroke dynamics.

The study makes the following contributions to the field of continuous authentication using behavioural biometrics:

1. A comprehensive analysis of the potential benefits and limitations of using reinforcement learning for continuous authentication of keystroke dynamics.
2. A Feature engineering enhancing technique to get better evaluation results that can be used in real world scenarios.
3. An implementation of a novel reinforcement learning-based anomaly detection model for continuous authentication of keystroke dynamics.

4. A unique feature encoder technique to reduce dimensionality of the data.
5. An evaluation of the proposed model using real-world data, and a comparison with existing methods.
6. The RL environment gym code is made available for the public through GitHub to utilize the work.
   GitHub URL: https://github.com/PriyaBansal68/Continuous-Authentication-Reinforcement-Learning-and-Behavioural-Biometrics

1.6. Research Outline:

The thesis is structured in 7 chapters as described below:

Chapter 1: Introduction

This chapter provides an overview of different types of authentication methods and how machine learning, specifically reinforcement learning, can be used to improve continuous authentication methods to better protect critical information in today's fast-paced digital environment.

Chapter 2: Literature Review

This chapter reviews existing research on user authentication using behavioural biometrics and machine learning, with a focus on the use of reinforcement learning in this area.

Chapter 3: Deep dive analysis of Proposed Method

It explains how reinforcement learning fits into the broader field of machine learning and reviews the essential concepts.

Chapter 4: Reinforcement Learning Framework (Methodology)

This chapter introduces the general reinforcement learning framework and explores the different methodologies that can be used to train an RL agent. It also examines the concept of Markov Decision Process (MDP) and its importance in the context of reinforcement learning. Also, this chapter provides a general introduction to the field of machine learning and the reinforcement learning approach.

Chapter 5: Results and analysis

In this chapter, we present the results of our experiments evaluating the performance of the proposed reinforcement learning-based algorithm for continuous authentication of keystroke dynamics. We analyse the results and compare them with existing methods to demonstrate the effectiveness of our approach. The proposed model utilizing reinforcement learning techniques for continuous authentication of keystroke dynamics is demonstrated to be effective in identifying and authenticating users in real-time. The results of the evaluation demonstrate the potential of this approach for addressing the limitations of traditional methods of user authentication and its suitability for the dynamic nature of user behavior over time.

Chapter 6: Future Direction for RL model implementation

In the discussion chapter, we will delve deeper into the results obtained from the experiments and analyze the performance of the proposed model in comparison to existing methods. We will also explore potential avenues for future research and improvements to the proposed model.

Chapter 7: Model Deployment: Integration of Machine Learning Model and FastAPI.

In this chapter, the approach to deploy the model on production is covered. It also covers the benefits of selected framework and compares with other available frameworks that can be used to deploy the model.

Chapter 8: Conclusion

The conclusion chapter summarizes the key findings and contributions of the research.

Chapter 2

2. Literature Review and Background

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 [16]. 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 [35].

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.

2.1 Literature Review

Considerable effort has been devoted to developing systems for user recognition based on keystroke dynamics with a focus on improving efficiency. This is especially critical given the vast amounts of data generated by users, which can vary over time due to contextual factors. Although the number of studies on keystroke dynamics with respect to text-based input is smaller than for fixed text, several notable studies have been conducted.

As per the background review for this topic, we came across supervised and unsupervised techniques and we didn’t come across anything significant on reinforcement learning.

Please find below some of the related work in the domain of behavioural biometrics that helped us to make progress with this study. Results of each have been stated in table 3:

2.1.1 Analysis of Strong Password Using Keystroke Dynamics Authentication in Touch Screen Devices [3]

A paper authored by Asma Salem and Dema Zaidan investigates the utilization of a verification and identification system for touch screen mobile devices. The authors construct a classification model based on a multi-layer perceptron neural network. The paper also combines timing and non-timing features, and it concludes that non-timing features enhance the security level. The study involves five users, and four features are extracted from the dataset. The authors raise the issue of using various keyboard types and develop a virtual keyboard for data collection.

2.1.2 Feasibility study on authentication-based keystroke dynamics over touch-screen devices [4]

Jeanjaitrong and Bhattarakosol conducted a literature review on keystroke dynamics and touch dynamics, highlighting the authentication process based on biometric behavior. They emphasized the importance of securing mobile devices since they are integral to daily life and the risk of data theft is high if compromised. The authors extracted four features, namely dwell time, interval time, interval timing ratio, and the distance between buttons, to classify the data. They collected data from ten users by having them enter a four-symbol password out of 16. The authors developed a Bayesian Network to determine the relationship between feature factors and summarized them in the classification phase.

2.1.3 Evaluation of One-Class and Two-Class Classification Algorithms on Mobile Devices [5]

Margit Antal and Laszlo Zsolt Szabo have conducted research on mobile device keystroke authentication using one-class and two-class classification algorithms. They have applied Bayesian networks and random forest classifiers on the dataset to compare the EER values for two-class classification. The one-class classification is used to verify the user by distinguishing them from outliers, while the two-class classification is used to identify the user. The authors' findings show that Random Forest provides the best EER value for a

dataset containing 42 users and 71 features, and all one-class classifiers perform better in classifying the negative class than the positive class.

2.1.4 Keystroke dynamics for authentication in smartphones [6]

Roh et al. and Lee conducted research on mobile device keystroke authentication using one class classification techniques. They collected features such as time interval, strength, position, and usage angle using smartphone sensors, along with the user's posture characteristics, including walk, hand, and table. The authors proposed a feature extraction algorithm using accelerometer and gyroscope sensors to identify the user's keystroke pattern. A test population of 15 users was used to build the model, and the authors performed pre-processing, scaling, and standardization on their data, resulting in good EER values.

2.1.5 Authenticating User Using Keystroke Dynamics and Finger Pressure [7]

P. Bhattarakosol and H. Saevanee achieved classification accuracy of 99%. They collected data from six female and four male users using a notebook as the input device. The authors utilized three features, namely inter-key, hold time, and finger pressure, to develop the k-NN model. The authors observed that the accuracy drops to 71% when only inter-key and hold time features are used, while using all three features together achieves an accuracy of 91%. They further concluded that the finger pressure feature significantly contributes to the high accuracy scores. However, the experiment suffers from a major limitation of statistical insignificance due to the small number of users involved in the study. Despite the promising results of keystroke dynamics and finger pressure for continuous authentication, some limitations still need to be addressed. For example, these features may not be reliable for users with physical disabilities that affect their typing behavior. Additionally, keystroke dynamics and finger pressure may not be effective against sophisticated attacks that mimic the user's behavior. Nonetheless, with the continuous advancements in machine learning and biometrics technology, keystroke dynamics and finger pressure-based authentication may provide a promising solution for enhancing the security and usability of authentication systems.

2.1.6 Keystroke dynamics as a biometric for authentication [8]

Monrose and Rubin conducted experiments on several participants from Bell Communications Lab to address cybersecurity threats such as network intrusion and

malicious attacks. To develop dynamic biometric techniques, Monrose analyzed user typing patterns using factor analysis to obtain a lower dimensional representation based on correlations and dependencies among features. The resulting feature subset included instances of similar and dissimilar user typing patterns. Monrose visualized covariance matrices for different features and used a k-NN (nearest neighbour) classifier for classification.

The authors achieved a classification accuracy of over 80% using a feature subset consisting of just four features. They also compared the performance of keystroke dynamics with that of other biometric authentication methods such as fingerprint and voice recognition and found that keystroke dynamics performed well.

The authors conclude that keystroke dynamics can be an effective biometric for authentication and can be combined with other authentication methods to provide an additional layer of security. They also discuss the limitations of keystroke dynamics, such as the impact of environmental factors on typing patterns, and the potential for attacks such as replay attack.

2.1.7 User authentication through typing biometrics features [9]

The author's work focuses on generating timing latency features to improve the authentication process and minimize the occurrence of false rejection and false acceptance rates. They propose an adaptive mechanism that uses new samples to create a new template and discard old ones. This approach modifies the standard deviation and thresholds for each feature, resulting in a two-trial authentication system. The biometric system records keystroke data such as key up, key down, and ASCII codes while the user types on the screen. The authors use four major features to enhance the existing password authentication mechanism when the password is not a secret.

The authors describe the various ways in which typing behavior can be analyzed, including keystroke dynamics, typing rhythm, and finger pressure. They also discuss the challenges associated with typing biometrics, such as the need for large datasets and the variability of typing behavior over time. The article then presents various studies and experiments that have been conducted to investigate the efficacy of typing biometrics for user authentication.

The authors review several different approaches to typing biometrics authentication, including feature-based methods, machine learning-based methods, and hybrid approaches that combine both. They also discuss the use of keystroke dynamics as a continuous authentication method, where the user is constantly monitored for changes in their typing behavior that may indicate an imposter.

2.1.8 The MOBIKEY Keystroke Dynamics Password Database [10]

This research provides a comprehensive review of the literature on keystroke dynamics and outlines the process of authenticating users based on their biometric behavior. The author discusses the 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. Various biometric modalities such as facial recognition, iris scanning, hand vein patterns, fingerprint recognition, and keystroke dynamics are discussed as effective methods for biometric authentication. However, the author highlights the challenge of cross-device authentication, which requires a model trained to recognize users across various computing devices, as different devices may have different keyboards and screen coordinates.

This aimed to evaluate the performance of the MOBIKEY system by analyzing a database of keystroke dynamics data collected from 54 users. The data was collected using a custom software application that recorded users' keystrokes as they typed a predefined password. The dataset included data on dwell time, flight time, hold time, and inter-key time, as well as other features such as typing speed and error rates.

The study used machine learning techniques to analyze the dataset and evaluate the performance of the MOBIKEY system. The authors used a range of machine learning algorithms, including decision trees, k-nearest neighbors, and support vector machines, to classify users based on their keystroke dynamics data.

The results showed that the MOBIKEY system performed well, with an overall accuracy of over 95%. The authors also compared the performance of different machine learning algorithms and found that decision trees and k-nearest neighbours performed best for this task.

2.1.9 BehavePassDB: Benchmarking Mobile Behavioral Biometrics [11]

Giuseppe Stragapede, Ruben Vera-Rodriguez, Ruben Tolosana and Aythami Morales have developed a standardized experimental protocol and benchmark to enable fair comparisons of novel approaches with existing ones in the field. The data collected included keystroke dynamics, swipe patterns, and accelerometer data. The dataset includes both genuine and imposter data to enable the evaluation of different biometric techniques.

The authors also developed a machine learning-based evaluation framework to evaluate the performance of different behavioral biometric techniques. They evaluated a range of techniques, including time-based, frequency-based, and deep learning-based approaches. They propose a system that uses a Long-Short Term Memory (LSTM) architecture with triplet loss and modality fusion at the score level. The individual modalities achieve an average of 58.75% AUC, whereas the average AUC of the best modality combinations is 66.22% AUC, representing a relative improvement of 12.71%. The best performance is obtained when using all modalities together for the task of keystroke, with a result of 68.72% AUC. The fusion of modalities results in an absolute improvement of around 10% compared to using touch data only. The performance of the other modalities is similar, except for tapping which does not improve much.

2.1.10 Machine and Deep Learning Applications to Mouse Dynamics for Continuous User Authentication [12]

A group of researchers consisting of Nyle Siddiqui, Rushit Dave, Mounika Vanamala, and Naeem Seliya evaluated a dataset of 40 users using three different machine learning and deep learning algorithms. Two evaluation scenarios were considered, one using binary classifiers for user authentication and the other using multi-class classification. The top performer for binary classification was a 1-dimensional convolutional neural network (1D-CNN) with a peak average test accuracy of 85.73% across the top 10 users. For multi-class classification, an artificial neural network (ANN) achieved a peak accuracy of 92.48%, the highest accuracy the researchers had seen for any classifier on their selected dataset.

The paper suggests that mouse dynamics can be a promising modality for continuous user authentication, and that deep learning techniques can be effectively applied to this problem. The authors also provide insights into the selection of appropriate feature extraction techniques and the optimization of machine learning and deep learning models for this application.

2.1.11 Classification of Threat Level in Typing Activity Through Keystroke Dynamics [13]

Amith K. Belman, Swathi Sridhara and Vir V. Phoha from Syracuse University analysed 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 authors of the study proposed a set of features for detecting whether a particular keystroke behavior originated from benign or adversarial activity. They observed high accuracies in classification and very low Type 1 and Type 2 errors, with most values lying between 85% to 97%. They also found that the RF classifier performed the poorest among the three classifiers (SVM, RF, and MLP), but still had better results than the conventional features.

To analyze the tradeoff of eliminating some features, the authors performed pairwise correlation analysis of the proposed features and retained the eight least correlated features for training and testing. They found that the classifiers using only the eight selected features performed very well, and the reduction of six features did not seem to affect its performance drastically. Type 1 and Type 2 errors increased by a marginal amount, but the accuracies decreased only slightly. This suggests that even with a smaller set of features, text samples can be classified accurately to their activity of origin.

The data was trained using SVM, RF and MLP 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%.

2.1.12 Free-Text Keystroke Dynamics for User Authentication [38]

The study focuses on the authentication of users based on keystroke dynamics obtained from free-text. The researchers propose a new feature engineering method that generates image-like transition matrices, which are then fed to a convolutional neural network (CNN) with cut-out regularization for better results. A hybrid model combining a CNN and a recurrent neural network (RNN) is also developed, which outperforms previous research in this area.

The study uses the Buffalo free-text keystroke dataset, collected from 148 research subjects who transcribed Steve Jobs' Stanford commencement speech and responded to free-text questions. Two kinds of 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 popular 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-based model consistently generates better results than the RNN-CNN based model, and the performance on the Buffalo dataset is consistently higher 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 keystroke dynamics in free-text.

Table 3:The below table shows all the above in a tabular form to show the features, model, dataset used, and results:

Study	No. of users	Behavioral biometrics types	Features Used	ML Type	ML Model	Performance
2.1.1	5	Keystroke	Hold Time, Flight Time, Latency, Inter-key time, Acceleration	Supervised	Neural Network	FAR 2.2%, FRR 8.67%, EER 5.43%

2.1.2	10	keystroke	Dwell Time, Flight Time, Latency, Inter-key time	Supervised	Bayesian network classifier	Accuracy 82.18% FAR 2.0 % FRR 17.8%
2.1.3	42	keystroke	Dwell Time, Flight Time, Latency, Inter-key time, Key Pressures	Supervised	Random Forests classifier, Bayes Network classifier, K-NN	EER 3% (two-class) EER 7% (one-class)
2.1.4	15	Keystroke and gyroscope	Hold Time, Flight Time, Latency, Inter-key time	Supervised	distance algorithm within the one-class classification	EER 6.93%
2.1.5	10	Keystroke and Finger Pressure	Dwell Time, 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 %

2.1.6	63	keystroke	Dwell Time, Flight Time, Latency, Inter-key time, Key Pressures	Supervised	weighted probabilistic classifier, Bayesian-like classifiers	Accuracy 83.22% to 92.14%
2.1.7	Not mentioned	keystroke	Dwell Time, Flight Time, Latency, Inter-key time	Unsupervised and supervised	K-means, Bayes Net algorithms, and Neural networks	FRR 1.45% FAR 1.89%
2.1.8	54	keystroke	Dwell Time, Flight Time, Latency, Inter-key time, Key Pressures	Supervised	Random forests classifier, Bayes Net algorithms, and KNN	EER Random Forests classifier, around 5 % for the second order feature set and around for the full feature set 3 %
2.1.9	81	Mouse	Dwell Time, Flight Time,	Supervised	Long-Short Term	80%–87% AUC range in the

			Latency, Inter-key time, Touch Pressure		Memory (LSTM)	random impostor case and 62%–69% AUC in the skilled impostor case
2.1.10	40	Mouse	Speed, Clicks, Movement	Supervised	1-dimensional convolutional neural network, artificial neural network (ANN)	test accuracy 85.73% for the top-10 users, peak accuracy 92.48%
2.1.11	103	Keystroke	Dwell Time, Flight Time, Latency, Inter-key time	Supervised	Support Vector Machine (SVM), Random Forest (RF) and Multilayer Perceptron (MLP)	accuracies (93% to 97%) Type 1 and Type 2 errors (3% to 8%)
2.1.12	73/ 80	Keystroke	down-down time, up-down time, up-up time,	Supervised	MLP, CNN, RNN, CNN-RNN	**Buffalo Dataset:** Accuracy: 98.56%

			and down-up time			EER: 0.0088 **Clarkson Dataset:** Accuracy: 91.74 EER:0.0755
This study	117	Keystroke	Key, Dwell time, Flight Time, Inter-key time, Key_id, key category	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

2.2 Inspiration from the Previous Work

This chapter has shown that there is a need for more efficient and effective continuous authentication methods, particularly in text-based user behavioral biometrics. Existing research in this field has primarily focused on supervised and unsupervised techniques, with

limited work on reinforcement learning [32]. Our proposed method aims to address the limitations of previous research by using keystroke dynamics and reinforcement learning to form a robust authentication system that can adapt to constantly changing user patterns and environments.

The previous research focused on limitations related to handling large datasets, updating the model to reflect changing patterns due to various factors like walking or talking on the phone, and the number of features used in the experiments [30]. These studies mainly considered eccentric user methods for connecting with input devices, without considering other user characteristics, interests, or behaviours. However, research has shown that human practices that are strongly influenced by a user's aptitude, knowledge, and interests also exhibit unique individual characteristics [31]. To address this issue, we propose a reinforcement learning-based method that can effectively identify users without relying on previously collected data and can adapt to the constantly changing environment.

The upcoming chapter will discuss the suggested approach for creating a secure authentication system utilizing keystroke dynamics and reinforcement learning.

Chapter 3

3. Deep Dive Analysis of Proposed Method

Reinforcement learning is a type of machine learning that focuses on training models to make decisions in an uncertain environment. In the context of behavioural biometrics, reinforcement learning can be used to train models to make decisions about a user's identity based on their typing dynamics, mouse movement, and other behavioural patterns [13].

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. The agent would be rewarded for making correct decisions and penalized for making incorrect decisions. Over time, the agent would learn to make more accurate decisions based on the feedback it receives.

One of the advantages of using reinforcement learning for behavioural biometrics is that it allows for continuous and dynamic adaptation of the model to changes in the user's behaviour 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 [29].

3.1 Supervised Machine Learning

In the context of keystroke dynamics, supervised learning involves training a supervised learning model on a dataset of keystroke data from known users, where the output is the user's identity [1-12]. By doing so, the model can predict the identity of a user based on their keystroke data. The figure 1 below illustrates how a supervised learning model would operate in this scenario [16] [19].

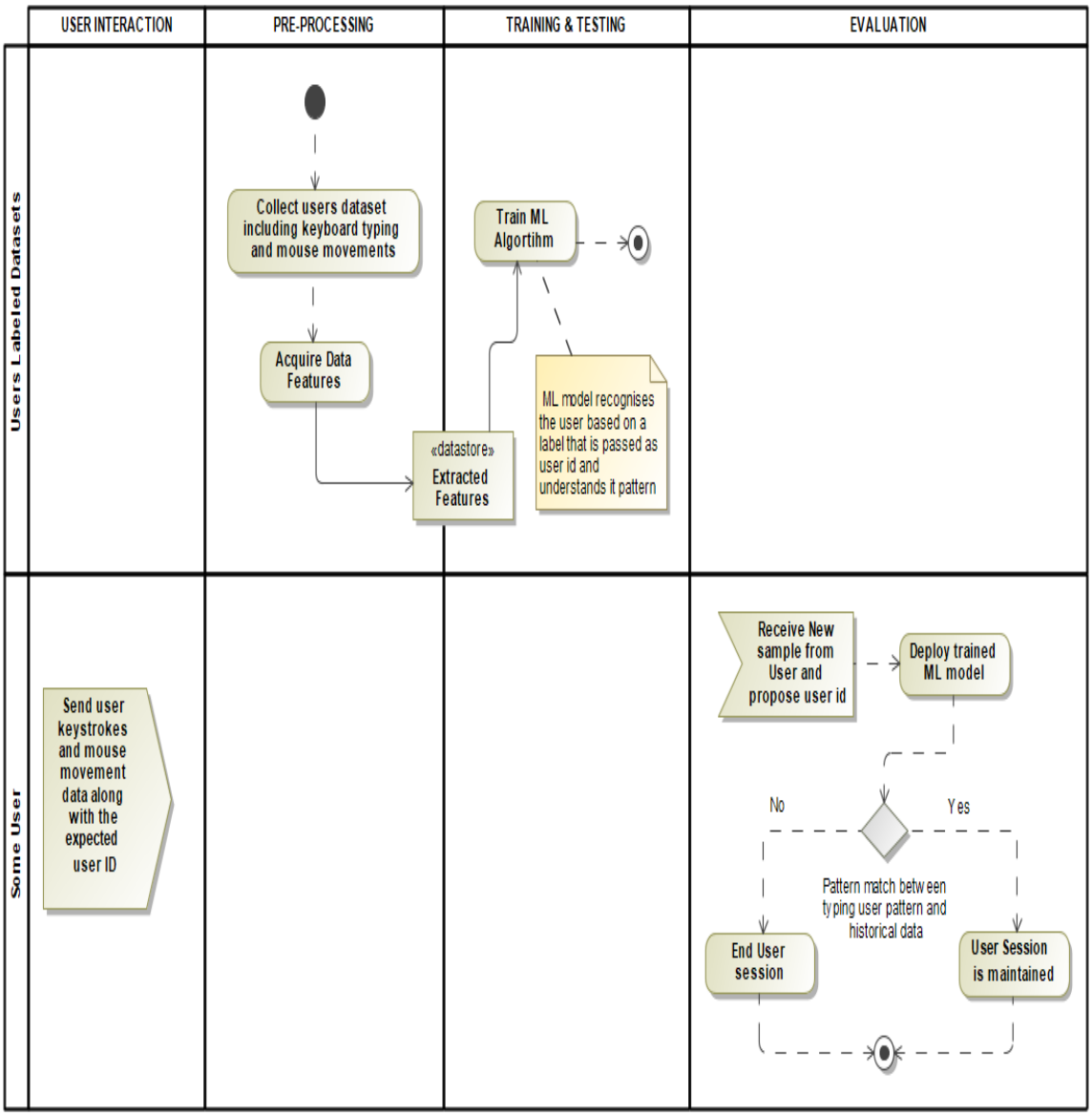

Figure 1: Existing Continuous Authentication Framework for Behavioral Biometrics

3.2 Proposed RL Framework

Reinforcement learning, on the other hand, is a type of machine learning where the algorithm learns to take actions in an environment to maximize a reward signal. The algorithm interacts with the environment, receives rewards or penalties based on its actions, and updates its internal state to improve its performance over time. 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 2) shows how a reinforcement learning model would differ from the proposed method.

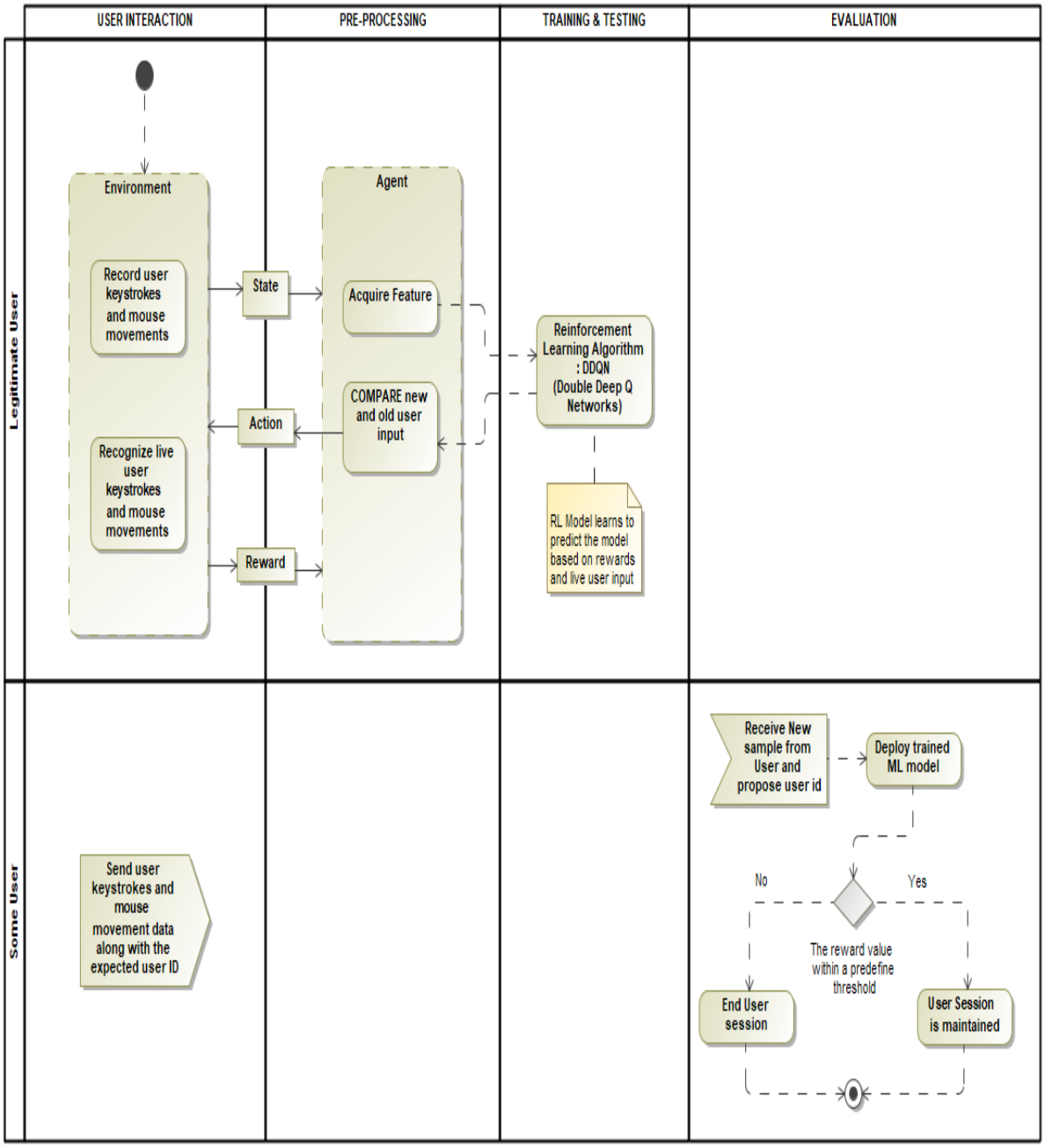

Figure 2: Proposed RL Framework

3.3 RL Model Flow for Continuous Authentication

The following diagram (figure 3) shows the flow of data and how the user would be authenticated at each step.

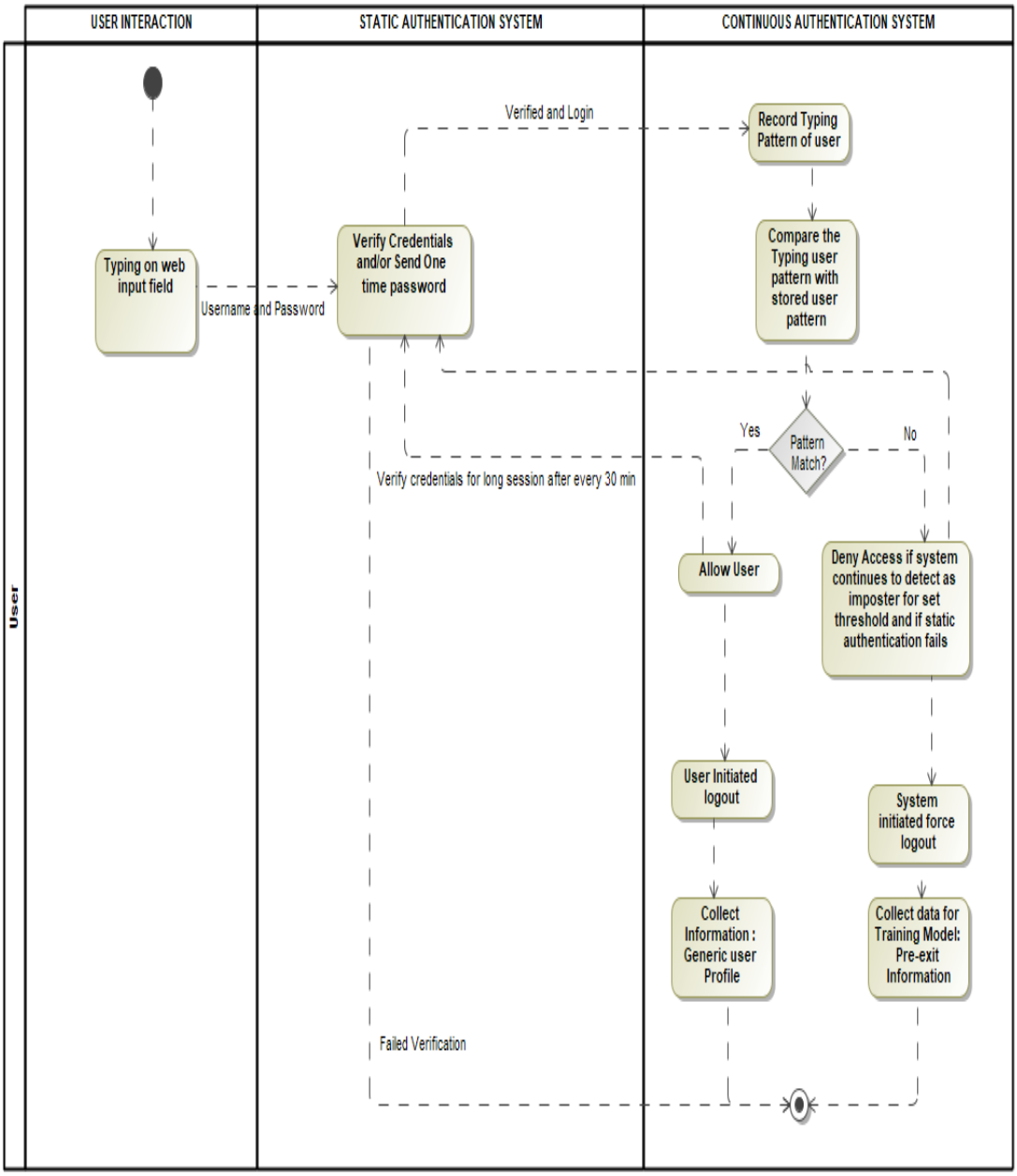

Figure 3: Data process flow for RL model

Overview of Reinforcement Learning from Behavioural biometrics perspective:

To begin with reinforcement Learning, we formulated our problem in RL mathematically in Markov Decision Process (MDP).

The Markov Decision Process (MDP) is a mathematical framework that is commonly used in reinforcement learning (RL) to model sequential decision-making problems. It consists of a set of states, actions, and rewards, and a set of rules for transitioning between states based on the actions taken [13].

In the context of behavioral biometrics, MDP can be used to model the process of authenticating a user based on their keystroke dynamics. The states in the MDP could represent different observations of the user's keystrokes, such as the timing between key presses, the duration of key presses, or the sequences of characters typed. The actions in the MDP could represent different authentication decisions, such as allowing access or denying access. 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.

3.4 MDP Algorithm

The MDP algorithm helps to determine the optimal sequence of actions to take in each state to maximize the expected reward over time, by using the concepts of value and policy. The value of a state is the expected long-term reward of being in that state, and the policy is the strategy for selecting actions in each state to maximize the expected reward. The model will be trained on the dataset of the user's keystroke dynamics which will be used to predict the user's keystroke pattern and the model will be able to detect the anomalies, in other words, the model will be able to identify if the current user is the same as the user that was authenticated earlier. The below image (figure 4) shows the MDP for our proposed method.

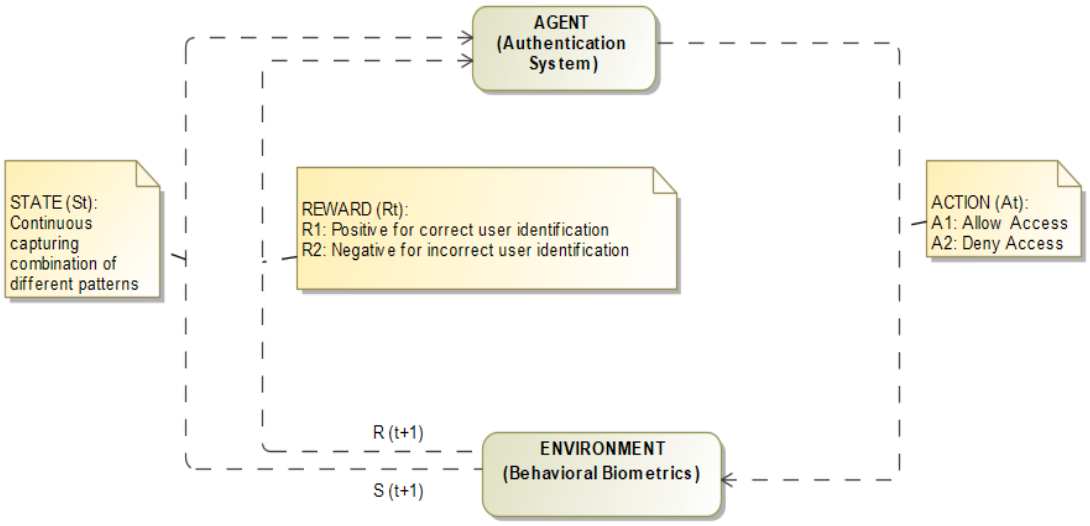

Figure 4: Proposed MDP Diagram for Continuous Authentication using Behavioral Biometrics

An RL model for keystroke dynamics to authenticate a user and provide continuous authentication would involve the following main components:

1. Agent: The agent is the system that makes decisions based on the keystroke data. The agent is responsible for analysing 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 aims to maximize the accumulated reward over time.
3. Action: The action 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: The environment is the overall system that the agent interacts with. It includes the user's keystroke data, the agent's decision-making process, 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 behavioural data that can be used to identify the user. The state is an essential component of the MDP because it is used to inform the agent's decision-making process and determine which action to take. The agent's decision-making process is based on the current state and the rewards it receives for different actions.

MDP is a powerful tool for modelling sequential decision-making problems in behavioural biometrics, which can be used to learn a policy for authenticating users based on their keystroke dynamics. This approach can be seen as a more advanced and sophisticated alternative to supervised machine learning where it allows for the system to learn and adapt over time and make better decision based on the changing behaviour of the user, whereas supervised machine learning model makes decision based on the labelled data.

Additionally, RL also allows for the incorporation of additional information about the user and the environment into the decision-making process, which can lead to more accurate and robust authentication decisions.

In the next methodology chapter, we will go into more detail about the specific RL algorithms and techniques that will be used to implement the proposed method. We will also

discuss the experimental design and evaluation metrics that will be used to assess the performance of the proposed method.

Chapter 4

4. Methodology

In this chapter, we will discuss the methodology used for building a robust authentication system using behavioral biometrics and reinforcement learning. Behavioral biometrics is a rapidly growing field that focuses on using unique behavioral patterns of individuals, such as keystroke dynamics, to authenticate their identity.

By combining these two approaches, we aim to create 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. Additionally, we will provide insights into the implementation and experimental results of our proposed method.

High Level Overview:

Here is a high-level overview of how we approached building a reinforcement learning-based user authentication system using keystroke dynamics:

1. Collect a dataset of keystroke dynamics data from several users. This should include a variety of different typing patterns, such as the time between key presses and the duration of key presses. In our case, we used the data from IEEE dataport website called BB-MAS_DATASET [12] 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 the mean, median, and standard deviation of various keystroke features, as well as 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 based 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 the test dataset to evaluate its performance.

4.1 Process Flow

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

1. Pre-processing the historical data: The first step is to gather a dataset of historical keystroke data from users. This data is then pre-processed to clean and format it for training. This may include removing any irrelevant data, normalizing the data, and splitting the data into training and testing sets.
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 based on the 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 form of rewards. A reward is a value that the agent receives for making a prediction. The reward is based on 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 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 as shown in figure 5 until the agent reaches a satisfactory level of accuracy.

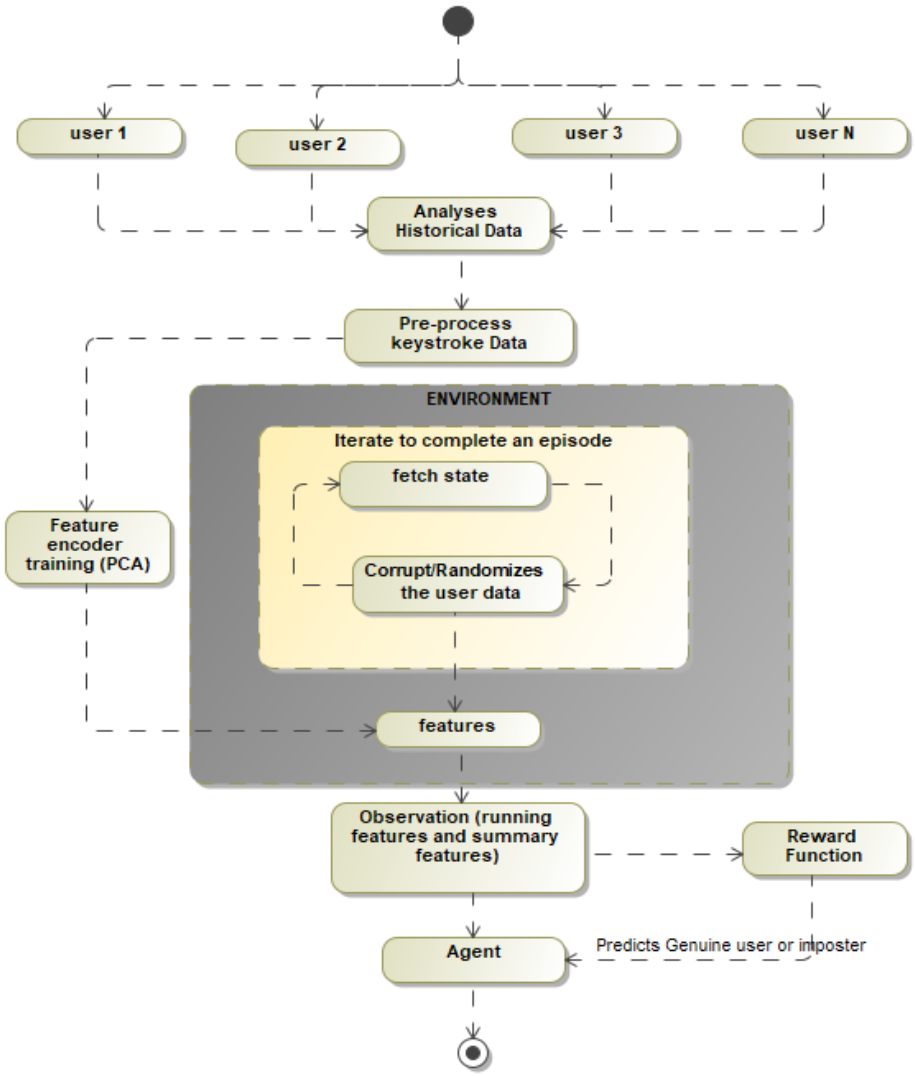

Figure 5: Code Flow

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.

4.1.1 Data pre-processing:

About Original Dataset:

The SU-AIS BB-MAS dataset [12] 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 pre-defined 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 single-device and multi-device sessions. For this research, we are making use of Keystrokes 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 [12] has been used in various studies in behavioral biometrics 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 diff 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 researched and came up with additional features for training.

The below 2 images (figure 6 and figure 7) shows the time difference between two consecutive events of 2 different users from the selected dataset.

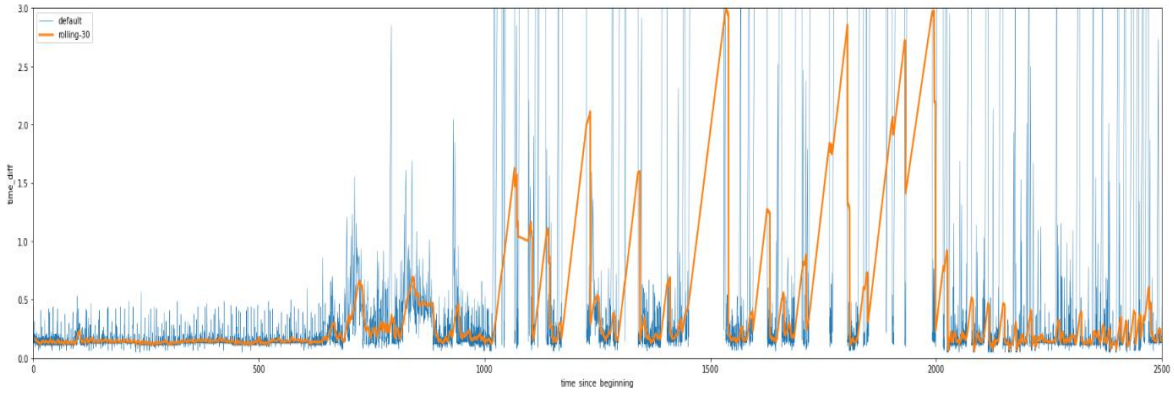

Figure 6: Time diff between two consecutive events for user 11

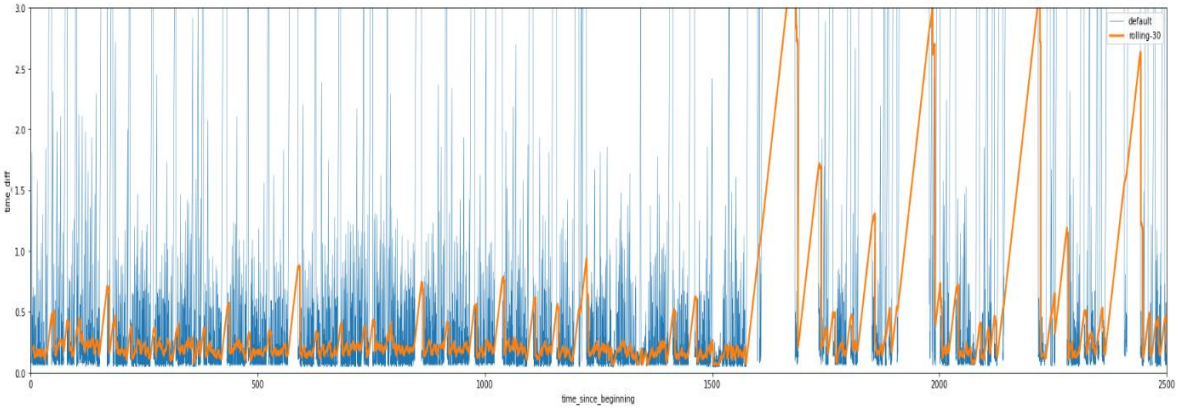

Figure 7: Time diff between two consecutive events for user 16

B. Exploratory Data Analysis: Keys hold time

The below two images (figure 8 and figure 9) shows the key holding time for key ‘t’.

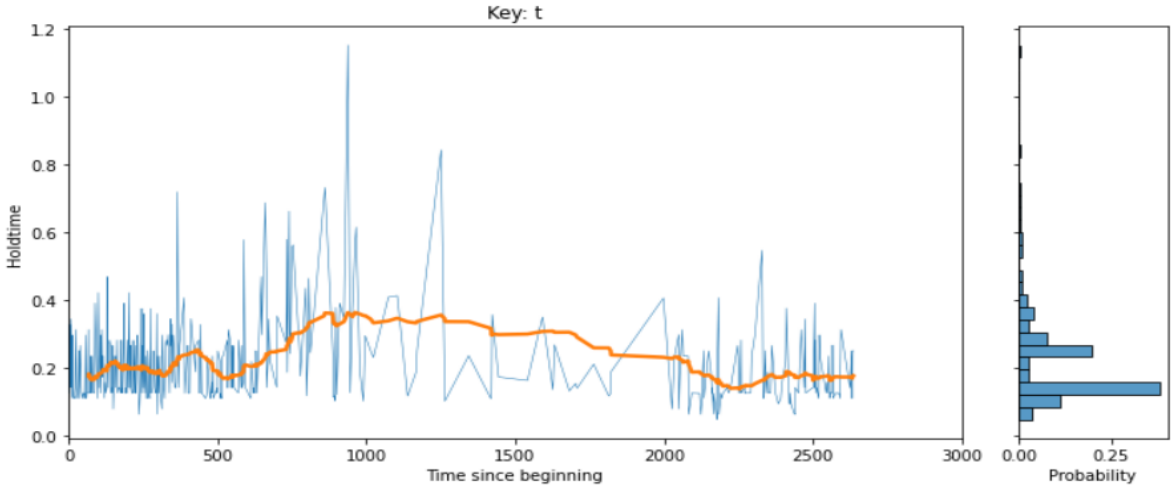

Figure 8: Key (‘t’) time (ms) for user 11

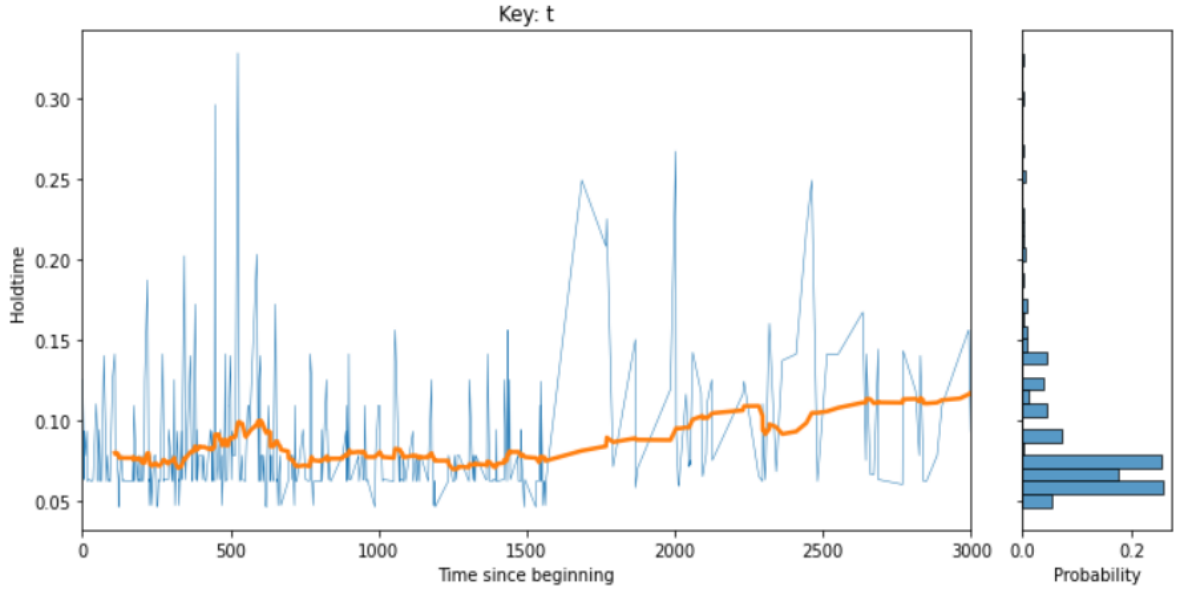

Figure 9 : Key ('t') time (ms) for user 16

C. Exploratory Data Analysis: Keyboard time length vs full session

Keyboard time length vs full session was an interesting development of all the features. Here, it is observed (also shown in figure 10) that most of the users spend almost 70-90% of their time on keyboard typing.

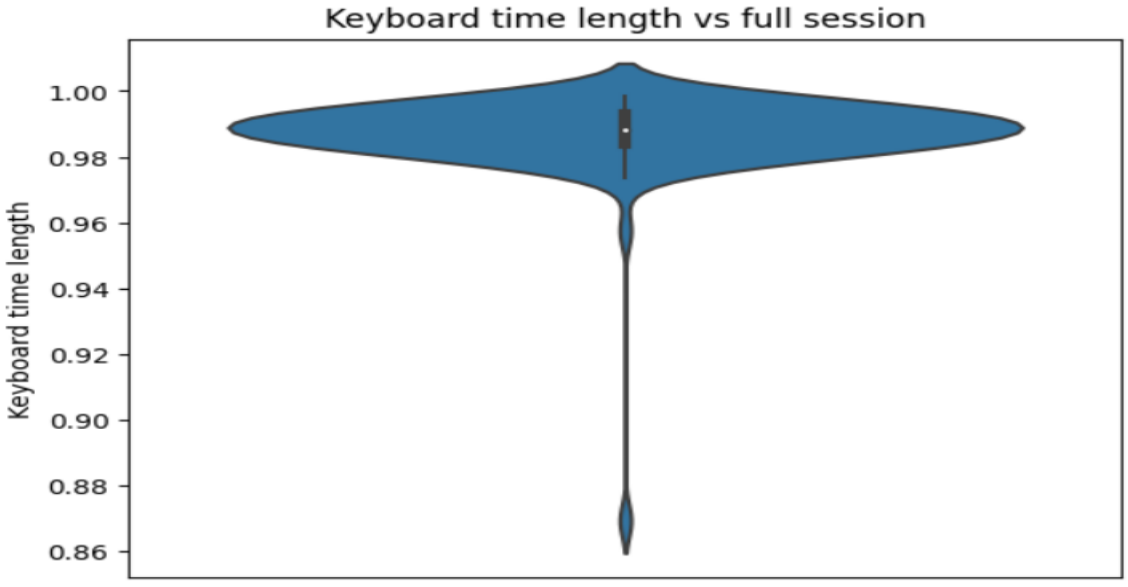

Figure 10: Keyboard time length vs full session

Feature engineering and data pre-processing 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 ensure that the data is clean and easy to work with clean data [22].
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 average time difference between consecutive key 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 [25].
5. Added new flight features such as press_to_press, release_to_press, hold_time.
6. 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
7. 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 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 changes in the user's behaviour over time, which can be important for detecting anomalies or changes in the user's behaviour that may indicate a security threat.

In this context, a vector of size (n,) is created. 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 user's behavior 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 time difference between consecutive key press and release events. Therefore, in the end, we have a 2D vector of size (k+1, n) that captures the dynamics of the user's behavior over time, including the hold time for each key across multiple events and the time difference between consecutive key 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 learn from the overall patterns and characteristics 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. 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 by dividing the number of characters typed by the total time taken. 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 time difference between consecutive key press and release events, which can also be used to identify unique typing rhythms.

The final vector size would be (p,) if there are p features. Each feature is a scalar value that summarizes the user's behavior over multiple events, making it easier for the agent to learn from the data.

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 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 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 based on the rewards received between different users' keystroke patterns and other behavioral data.

4.2.1 Fetch State:

In the context of training an agent for continuous authentication using reinforcement learning (RL) with behavioral biometrics, the environment plays an important role in fetching the state. The environment is responsible for providing the agent with the data it needs to make predictions.

There are two important parameters to understand when fetching the state:

- **No: Number of events in an observation.**

This parameter determines the number of keystroke events that will be included in each observation.

- **Nh: Number of events to hop to move to next 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 user's keystroke data.

An episode is created by iterating on the user's historical data using the above pattern. An episode is terminated if there are not enough data points to create the observation.

Overall, the environment plays an important role in fetching the state for training the agent by providing the agent with the keystroke data it needs to make predictions, it iterates on the user's historical data to create an episode and terminates the episode 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 improve the robustness of the model. In machine learning models, the model is trained on a dataset 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 behaviours and unexpected situations [24].

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. This can increase the model's ability to generalize to new, unseen 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 [15]. 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 [30].

Furthermore, by randomly corrupting or randomizing the keystroke data, it increases the entropy of the training dataset, making the model more robust to the presence of outliers or anomalies.

While iterating on the episode, we have to develop a strategy in order to also learn out of order behaviours i.e. something which does not follow the usual pattern. To perform this, we simply introduced random events from different user's data.

Corruption is imperative to incorporate measures that prevent the model from constantly predicting the same user, to improve its accuracy and prevent it from becoming stagnant.

In this context, during iteration on the episode, certain events from different user's data are randomly selected and introduced into the episode with a designated probability, such as 50% [25]. This approach aims to enable the agent to learn to recognize patterns that deviate from the usual pattern and handle unexpected situations.

4.2.3 Process to create observation:

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

- **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 time difference between consecutive key press and release events. The running features provide the agent with information about the dynamics of the user's behavior over time.

- **Encode the running features using trained encoder model:**

The second step is to encode the running features using a trained encoder model such as PCA (discussed in chapter 5). This can help to reduce the dimensionality of the data and make it more manageable for the agent to learn from.

- **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 (figure 11). 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.

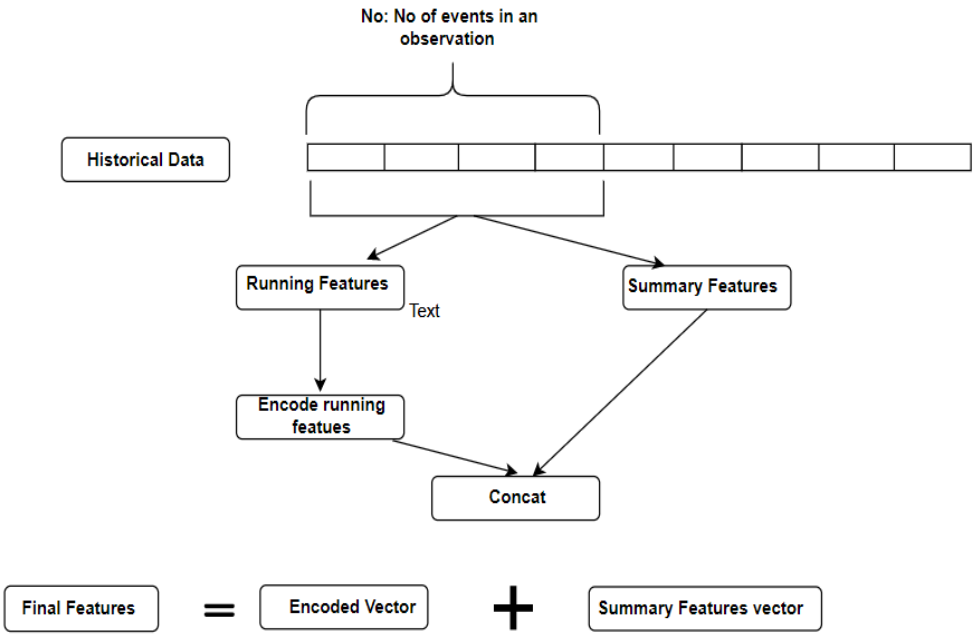

Figure 11: Process adopted to calculate the final features.

The final observation vector size is (f1+f2,), where f1 is the number of summary features and f2 is the number 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:

In reinforcement learning, the reward function is used to provide feedback to the agent about the quality of its actions (accept or deny from the authentication system). The reward function is used to guide the agent's learning process and to determine the optimal behaviour.

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 a value of 1 for true positives (TP) and true negatives (TN) if the systems correctly predict the user and a value of 0 for false positives (FP) and false negatives (FN) otherwise.

A true positive (TP) is when a corrupted observation is made, and the model correctly predicts that it is a hacker. A true negative (TN) 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 can be used to guide the agent's learning process 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 the dimensionality of the 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.

PCA is a technique used to identify 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 data. After training, the PCA model can reduce the dimensionality of the data by identifying the most important features and discarding the less important ones.

In our experiments, it was observed that for some users, even up to 10 components were able to explain ~99% variance. This means that even with just 10 components, the PCA model was able to 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 consists 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.

The output layer of the network has two nodes, one node represents the value of the action "user" (0) and another node represents the value of the action "hacker" (1). The agent uses the values of these two nodes to make predictions about the user.

In all the experiments, the capacity of the replay memory is set to 10000 observations. The replay memory is used to store the observations and actions taken by the agent, so that it can learn from its past experiences. The Adam optimizer is used with a learning rate of 1e-3. The optimizer is used to update the parameters of the neural network. The loss function used is Huber L1 loss (nn.SmoothL1Loss). Huber L1 loss is used to measure the difference between

the predicted and target values. The agent uses this loss function to learn from the data and improve its predictions.

The loss function used is Huber L1 loss (nn.SmoothL1Loss), which measures the difference between the predicted and target values. The agent uses this loss function to learn from the data and improve its predictions. Huber Loss is a loss function that is less sensitive to outliers than the mean squared error loss function.

4.3.1 RL algorithm: DDQN

DDQN is an extension of the Q-learning algorithm [13], which is a type of RL algorithm that is used to learn the optimal action-value function for a given environment. In DDQN, two separate Q-networks are used: a primary Q-network and a target Q-network. The primary Q-network is used to make predictions about the action-value function, while the target Q-network is used to generate the target values for the primary Q-network during training. The idea behind this is that it can reduce the correlation between the action-value estimates and the target values, which can help to stabilize the training process and improve the performance of the algorithm. It also addresses some of the problems that can arise with Q-learning, such as overestimating Q-values [13]. DDQN uses two neural networks: a Q-network and a target network. The Q-network is used to estimate the Q-values, while the target network is used to generate the targets for the Q-network.

Table 4 : Reasons for choosing DDQN for this task:

Target Network	There are two Q-networks, the primary network, and the target network. The target network is used to estimate the Q-values for the next state, which is then used to update the primary network.
Action Selection	The action selection is based on the primary network, and the Q-value estimation is done using the target network.
Learning Stability	DDQN has good 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 estimate the Q-values.

Performance	DDQN has been shown to outperform in various applications. This is due to the improved learning stability, which leads to better convergence to the optimal policy. Additionally, DDQN can learn faster and requires fewer training samples compared to DQN.

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. 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 between 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 improve the overall accuracy and effectiveness of the authentication system.

4.4 EVALUATION:

Evaluation is the process of assessing the performance of the reinforcement learning model for continuous authentication using behavioral biometrics. The evaluation is done on a test set and several parameters are varied to evaluate the model's performance.

Following parameter values were randomly chosen to start the experiment:

1. No = 25: Number of events in an observation
2. Nh = 5: Number of events to hop to move to next observation
3. Num encoder features = 10: Number of encoded features used for each observation
4. C_update = 2: Update target net after 2 episodes
5. Eps_decay = 200: Exploration decay rate

4.5 Live Runner:

A live runner was also created for live demonstration purposes. Basically, the user's historical data is simulated, and the model is continuously authenticating on it in real-time. We can come as an intruder by pressing the trigger key and start typing. Whenever the model detects it as a hacker, it throws an alert.

A live runner is a tool that allows for real-time demonstration of the reinforcement learning model for continuous authentication using behavioral biometrics. It simulates the user's historical data, and the model continuously authenticates it in real-time.

The live runner allows for a user to simulate an intruder by pressing a trigger key and start typing. The model is then able to detect if the typing pattern is that of an authorized user or a hacker. If the model detects the intruder as a hacker, it throws an alert.

The live runner is useful for demonstrating the capabilities of the model in a real-world scenario and allows for testing of the model's performance under different conditions. It can also be used to evaluate the model's performance in identifying hackers in real-time.

Chapter 5

5. Results and Analysis

This chapter presents the findings obtained from the experiments and discusses their significance and implications for the study. This chapter provides an in-depth analysis of the results and their implications for continuous authentication using keystroke dynamics with the proposed RL framework. We performed multiple experiments after changing the parameter combinations in config.json file. The experiment that gave the best results has been discussed 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

Performance of keystroke analysis is typically measured in terms of various error rates, namely Accuracy, FAR, FRR, EER and ROC Curve.

Accuracy:

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

Accuracy = (number of true positives + number of true negatives) / (total number of genuine users + total number of imposters) x 100% (1)

False Rejection Rate (FRR) /Type I error:

It measures the percent of valid users who are rejected as impostors. In statistics this type of errors is referred to as a Type I error. FRR described in equation (2)

FRR = number of refused genuine/Total number of genuine (2)

FRR = FNR = FN/(FN + TP) = 1-TPR

False Acceptance Rate (FAR) / Type II error:

The probability of an unauthorized user gaining access to a secured system is known as the false acceptance rate, or Type II error in statistics. The ideal scenario is to have both the false acceptance rate and false rejection rate (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 = number of accepted imposters/Total number of imposters (3)

FAR = FPR = FP/(FP + TN)

Equal Error Rate (EER):

The equal error rate (EER) is a widely used metric to evaluate biometric systems, which determines the point where the rates of false acceptance (FAR) and false rejection (FRR) are equal. A lower EER value indicates higher accuracy of the biometric system. EER described in equation (4)

ERR = (FRR+FAR)/2 (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 [20]. They can also help to identify the optimal threshold value for the classifier algorithm, balancing the sensitivity and specificity of the system, 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%. In practice, the curve is usually not perfect, and the area under the ROC curve (AUC) is used as a measure of the system's overall performance. A higher AUC indicates better performance, with a value of 1 indicating perfect discrimination and a value of 0.5 indicating a system that performs no better than random chance.

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_end": 0.01,
10. "eps_decay": 200,
11. "train_split": 0.7

5.3 Results:

For the initial results, over 20 episode runs, an average accuracy of 74.8% was achieved for live user. This means that the model was able to correctly predict the user as an authorized user in 74.8% of the cases on average.

When the number of encoded features used for each observation is increased to 1096, the average accuracy for a specific user (user 11) increased to 92.7% over 20 episode runs. This means that the model was able to correctly predict the user as an authorized user in 92.7% of the cases on average. Below (Table 6 and 7) are results of the experiments we performed and figures 12 and 13 are the heatmaps showing the results.

Table 5 : 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

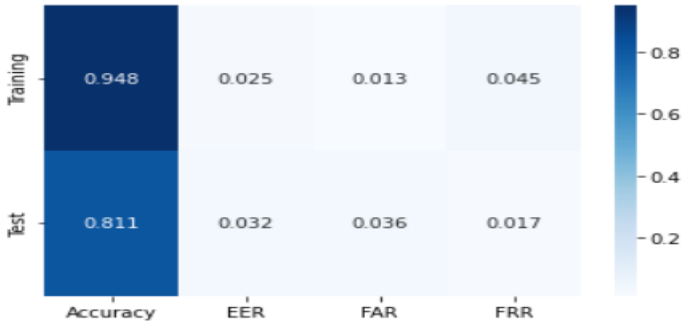

Figure 12: Heatmap results on full dataset of 117 users.

Table 6 : 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%	0.00	0.07	0.31	0.29
71	96.7%	93.5%	0.00	0.11	0.39	0.28
99	96.8%	81.2%	0.049	0.06	0.19	0.34
116	100%	85.7%	0.00	0.13	0.26	0.32

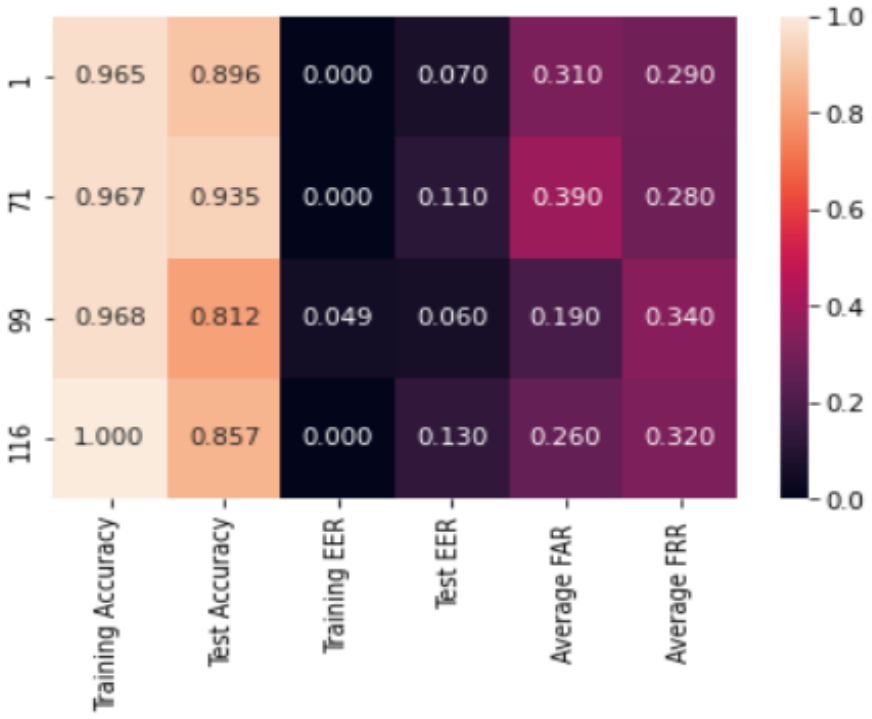

Figure 13: Heatmap results on users (randomly chosen) 1,71,99,116.

Training Accuracy/EER:

The below (figures 13,14,15,16) are the snapshot of the training performed on individual users. Figure 17 shows how the EER decreases with increasing no. of iterations. It signifies, if we increase the no. of iterations per episode then the EER decrease and ultimately, we can even achieve near to zero EER.

```
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 14: Training for User 1 showing Accuracy and EER

```
INFO:__main__:Mode: train
INFO:__main__:Number of users: 1
INFO:__main__:Processing user ID: 71
INFO:__main__:Selected corrupted user IDs: ['40', '97', '45', '8', '1', '114', '81', '58', '82', '90']
INFO:root:Episode length: 31; reward: 18.0; cm: {'TP': 15, 'FP': 12, 'TN': 3, 'FN': 1}; exploration: 4; eer: 0.40
INFO:root:Episode length: 31; reward: 17.0; cm: {'TP': 17, 'FP': 13, 'TN': 0, 'FN': 1}; exploration: 6; eer: 0.62
INFO:root:Episode length: 31; reward: 15.0; cm: {'TP': 10, 'FP': 16, 'TN': 5, 'FN': 0}; exploration: 11; eer: 0.48
INFO:root:Episode length: 31; reward: 15.0; cm: {'TP': 13, 'FP': 14, 'TN': 2, 'FN': 2}; exploration: 13; eer: 0.50
INFO:root:Episode length: 31; reward: 14.0; cm: {'TP': 11, 'FP': 12, 'TN': 3, 'FN': 5}; exploration: 17; eer: 0.53
INFO:root:Episode length: 31; reward: 22.0; cm: {'TP': 14, 'FP': 7, 'TN': 8, 'FN': 2}; exploration: 20; eer: 0.19
INFO:root:Episode length: 31; reward: 21.0; cm: {'TP': 8, 'FP': 8, 'TN': 13, 'FN': 2}; exploration: 26; eer: 0.33
INFO:root:Episode length: 31; reward: 26.0; cm: {'TP': 15, 'FP': 1, 'TN': 11, 'FN': 4}; exploration: 27; eer: 0.17
INFO:root:Episode length: 31; reward: 29.0; cm: {'TP': 20, 'FP': 1, 'TN': 9, 'FN': 1}; exploration: 28; eer: 0.10
INFO:root:Episode length: 31; reward: 29.0; cm: {'TP': 11, 'FP': 1, 'TN': 18, 'FN': 1}; exploration: 31; eer: 0.08
INFO:root:Episode length: 31; reward: 29.0; cm: {'TP': 14, 'FP': 1, 'TN': 15, 'FN': 1}; exploration: 33; eer: 0.07
INFO:root:Episode length: 31; reward: 25.0; cm: {'TP': 13, 'FP': 2, 'TN': 12, 'FN': 4}; exploration: 37; eer: 0.21
INFO:root:Episode length: 31; reward: 26.0; cm: {'TP': 9, 'FP': 0, 'TN': 17, 'FN': 5}; exploration: 40; eer: 0.21
INFO:root:Episode length: 31; reward: 30.0; cm: {'TP': 16, 'FP': 0, 'TN': 14, 'FN': 1}; exploration: 44; eer: 0.06
INFO:root:Episode length: 31; reward: 30.0; cm: {'TP': 12, 'FP': 0, 'TN': 18, 'FN': 1}; exploration: 44; eer: 0.00
INFO:root:Episode length: 31; reward: 29.0; cm: {'TP': 14, 'FP': 2, 'TN': 15, 'FN': 0}; exploration: 47; eer: 0.12
INFO:root:Episode length: 31; reward: 27.0; cm: {'TP': 11, 'FP': 1, 'TN': 16, 'FN': 3}; exploration: 53; eer: 0.18
INFO:root:Episode length: 31; reward: 28.0; cm: {'TP': 11, 'FP': 1, 'TN': 17, 'FN': 2}; exploration: 54; eer: 0.06
INFO:root:Episode length: 31; reward: 28.0; cm: {'TP': 12, 'FP': 2, 'TN': 16, 'FN': 1}; exploration: 57; eer: 0.11
INFO:root:Episode length: 31; reward: 29.0; cm: {'TP': 11, 'FP': 0, 'TN': 18, 'FN': 2}; exploration: 60; eer: 0.08
INFO:root:Metrics on best EER iteration: {'eer': 0.0, 'accuracy': 0.967741935483871}
```

Figure 15: Training for User 71 showing Accuracy and EER

```
INFO:__main__:Mode: train
INFO:__main__:Number of users: 1
INFO:__main__:Processing user ID: 99
INFO:__main__:Selected corrupted user IDs: ['92', '90', '74', '78', '81', '9', '64', '61', '104', '113']
INFO:root:Episode length: 32; reward: 16.0; cm: {'TP': 1, 'FP': 0, 'TN': 15, 'FN': 16}; exploration: 4; eer: 0.47
INFO:root:Episode length: 32; reward: 13.0; cm: {'TP': 0, 'FP': 1, 'TN': 13, 'FN': 18}; exploration: 6; eer: 0.64
INFO:root:Episode length: 32; reward: 23.0; cm: {'TP': 1, 'FP': 0, 'TN': 22, 'FN': 9}; exploration: 11; eer: 0.40
INFO:root:Episode length: 32; reward: 16.0; cm: {'TP': 0, 'FP': 0, 'TN': 16, 'FN': 16}; exploration: 14; eer: 0.69
INFO:root:Episode length: 32; reward: 17.0; cm: {'TP': 2, 'FP': 1, 'TN': 15, 'FN': 14}; exploration: 17; eer: 0.31
INFO:root:Episode length: 32; reward: 26.0; cm: {'TP': 11, 'FP': 1, 'TN': 15, 'FN': 5}; exploration: 23; eer: 0.25
INFO:root:Episode length: 32; reward: 26.0; cm: {'TP': 7, 'FP': 2, 'TN': 19, 'FN': 4}; exploration: 26; eer: 0.19
INFO:root:Episode length: 32; reward: 27.0; cm: {'TP': 16, 'FP': 0, 'TN': 11, 'FN': 5}; exploration: 28; eer: 0.19
INFO:root:Episode length: 32; reward: 31.0; cm: {'TP': 19, 'FP': 0, 'TN': 12, 'FN': 1}; exploration: 30; eer: 0.05
INFO:root:Episode length: 32; reward: 30.0; cm: {'TP': 10, 'FP': 0, 'TN': 20, 'FN': 2}; exploration: 32; eer: 0.08
INFO:root:Episode length: 32; reward: 28.0; cm: {'TP': 15, 'FP': 2, 'TN': 13, 'FN': 2}; exploration: 34; eer: 0.13
INFO:root:Episode length: 32; reward: 23.0; cm: {'TP': 8, 'FP': 1, 'TN': 15, 'FN': 8}; exploration: 39; eer: 0.25
INFO:root:Episode length: 32; reward: 28.0; cm: {'TP': 10, 'FP': 0, 'TN': 18, 'FN': 4}; exploration: 42; eer: 0.14
INFO:root:Episode length: 32; reward: 30.0; cm: {'TP': 15, 'FP': 0, 'TN': 15, 'FN': 2}; exploration: 44; eer: 0.12
INFO:root:Episode length: 32; reward: 28.0; cm: {'TP': 12, 'FP': 2, 'TN': 16, 'FN': 2}; exploration: 46; eer: 0.11
INFO:root:Episode length: 32; reward: 28.0; cm: {'TP': 12, 'FP': 1, 'TN': 16, 'FN': 3}; exploration: 50; eer: 0.07
INFO:root:Episode length: 32; reward: 28.0; cm: {'TP': 11, 'FP': 1, 'TN': 17, 'FN': 3}; exploration: 54; eer: 0.17
INFO:root:Episode length: 32; reward: 28.0; cm: {'TP': 12, 'FP': 1, 'TN': 16, 'FN': 3}; exploration: 56; eer: 0.13
INFO:root:Episode length: 32; reward: 30.0; cm: {'TP': 12, 'FP': 1, 'TN': 18, 'FN': 1}; exploration: 58; eer: 0.08
INFO:root:Episode length: 32; reward: 28.0; cm: {'TP': 8, 'FP': 0, 'TN': 20, 'FN': 4}; exploration: 61; eer: 0.17
INFO:root:Metrics on best EER iteration: {'eer': 0.04999999999999991, 'accuracy': 0.96875}
```

Figure 16: Training for User 99 showing Accuracy and EER

```
INFO:__main__:Mode: train
INFO:__main__:Number of users: 1
INFO:__main__:Processing user ID: 116
INFO:__main__:Selected corrupted user IDs: ['67', '24', '40', '90', '63', '44', '56', '18', '8', '109']
INFO:root:Episode length: 28; reward: 11.0; cm: {'TP': 1, 'FP': 3, 'TN': 10, 'FN': 14}; exploration: 3; eer: 0.69
INFO:root:Episode length: 28; reward: 12.0; cm: {'TP': 0, 'FP': 2, 'TN': 12, 'FN': 14}; exploration: 5; eer: 0.71
INFO:root:Episode length: 28; reward: 14.0; cm: {'TP': 1, 'FP': 3, 'TN': 13, 'FN': 11}; exploration: 10; eer: 0.50
INFO:root:Episode length: 28; reward: 16.0; cm: {'TP': 1, 'FP': 2, 'TN': 15, 'FN': 10}; exploration: 11; eer: 0.82
INFO:root:Episode length: 28; reward: 10.0; cm: {'TP': 0, 'FP': 3, 'TN': 10, 'FN': 15}; exploration: 16; eer: 0.67
INFO:root:Episode length: 28; reward: 24.0; cm: {'TP': 12, 'FP': 1, 'TN': 12, 'FN': 3}; exploration: 18; eer: 0.20
INFO:root:Episode length: 28; reward: 23.0; cm: {'TP': 8, 'FP': 2, 'TN': 15, 'FN': 3}; exploration: 23; eer: 0.18
INFO:root:Episode length: 28; reward: 25.0; cm: {'TP': 10, 'FP': 2, 'TN': 15, 'FN': 1}; exploration: 26; eer: 0.12
INFO:root:Episode length: 28; reward: 26.0; cm: {'TP': 16, 'FP': 0, 'TN': 10, 'FN': 2}; exploration: 27; eer: 0.00
INFO:root:Episode length: 28; reward: 27.0; cm: {'TP': 19, 'FP': 0, 'TN': 8, 'FN': 1}; exploration: 29; eer: 0.00
INFO:root:Episode length: 28; reward: 26.0; cm: {'TP': 9, 'FP': 0, 'TN': 17, 'FN': 2}; exploration: 31; eer: 0.00
INFO:root:Episode length: 28; reward: 26.0; cm: {'TP': 12, 'FP': 1, 'TN': 14, 'FN': 1}; exploration: 33; eer: 0.07
INFO:root:Episode length: 28; reward: 21.0; cm: {'TP': 11, 'FP': 2, 'TN': 10, 'FN': 5}; exploration: 37; eer: 0.17
INFO:root:Episode length: 28; reward: 25.0; cm: {'TP': 8, 'FP': 0, 'TN': 17, 'FN': 3}; exploration: 40; eer: 0.27
INFO:root:Episode length: 28; reward: 23.0; cm: {'TP': 9, 'FP': 0, 'TN': 14, 'FN': 5}; exploration: 43; eer: 0.14
INFO:root:Episode length: 28; reward: 28.0; cm: {'TP': 14, 'FP': 0, 'TN': 14, 'FN': 0}; exploration: 44; eer: 0.00
INFO:root:Episode length: 28; reward: 26.0; cm: {'TP': 10, 'FP': 1, 'TN': 16, 'FN': 1}; exploration: 45; eer: 0.06
INFO:root:Episode length: 28; reward: 26.0; cm: {'TP': 13, 'FP': 1, 'TN': 13, 'FN': 1}; exploration: 47; eer: 0.07
INFO:root:Episode length: 28; reward: 24.0; cm: {'TP': 9, 'FP': 1, 'TN': 15, 'FN': 3}; exploration: 53; eer: 0.17
INFO:root:Episode length: 28; reward: 26.0; cm: {'TP': 11, 'FP': 1, 'TN': 15, 'FN': 1}; exploration: 54; eer: 0.08
INFO:root:Metrics on best EER iteration: {'eer': 0.0, 'accuracy': 1.0}
```

Figure 17: Training for User 116 showing Accuracy and EER

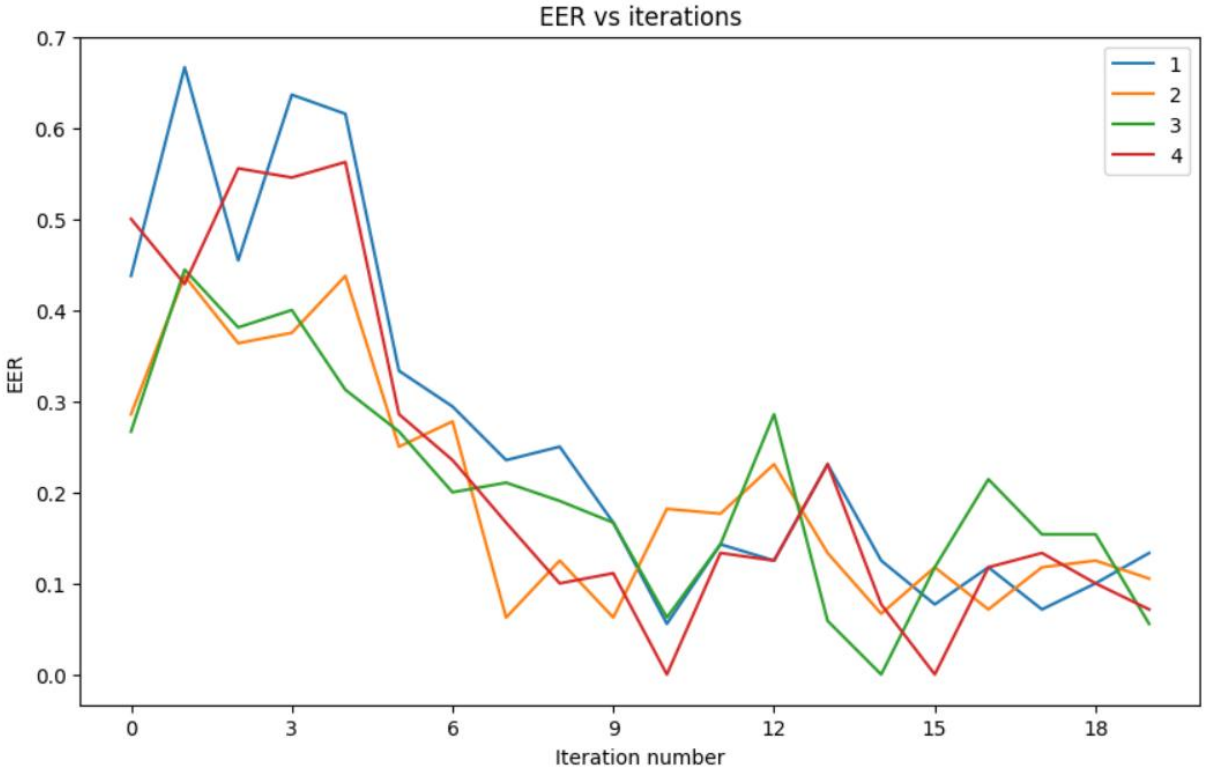

Figure 18: Graph for users 1,2,3 and 4 showing decreasing EER with no. of iterations.

Test Accuracy/EER:

The snapshot (figures 19, 20, 21 and 22) 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. 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 [21].

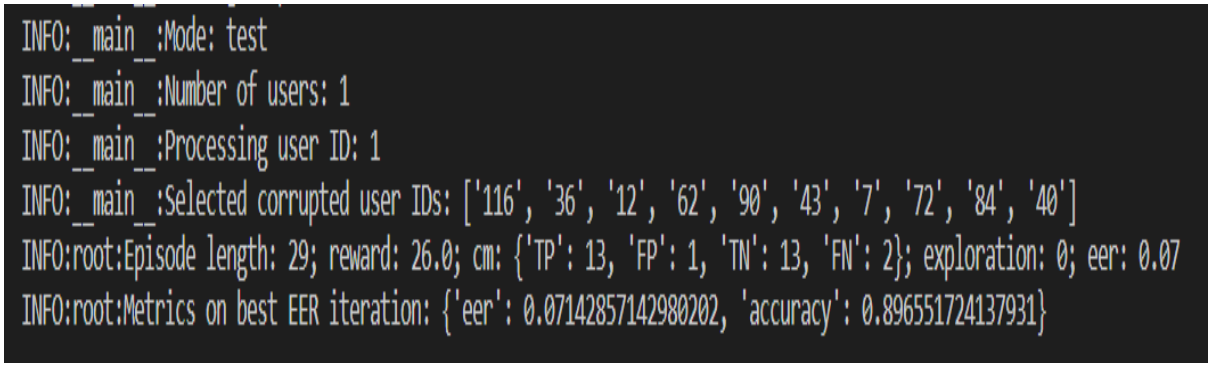

Figure 19: Testing for User 1 showing Accuracy and EER

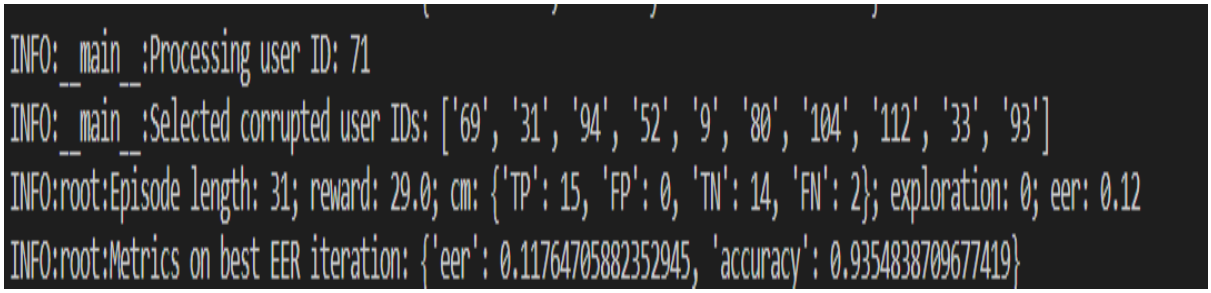

Figure 20: Testing for User 71 showing Accuracy and EER

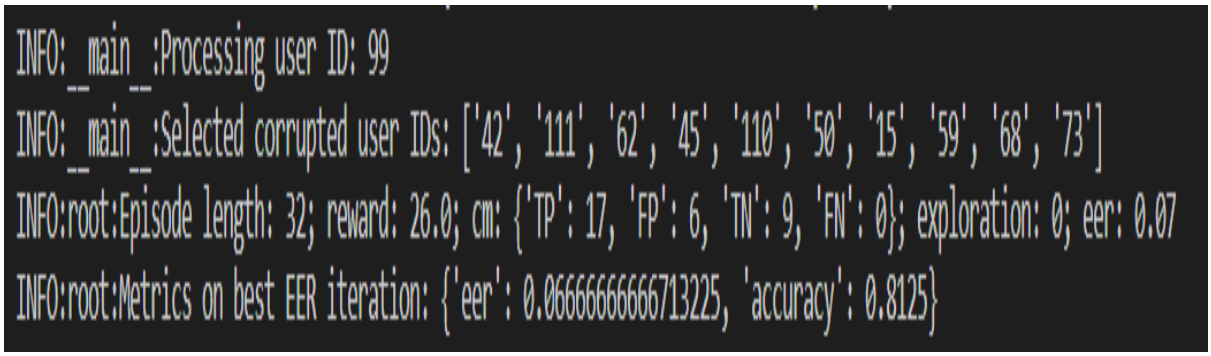

Figure 21: Testing for User 99 showing Accuracy and EER

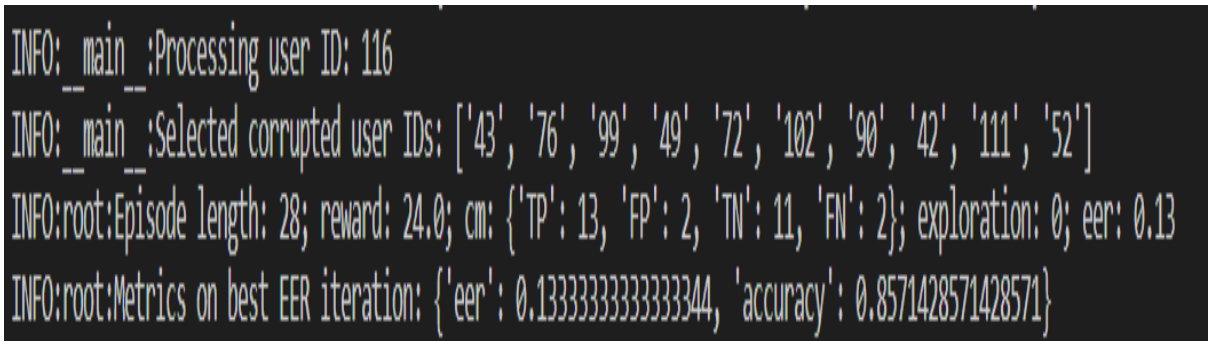

Figure 22: Testing for User 116 showing Accuracy and EER

ROC Curve:

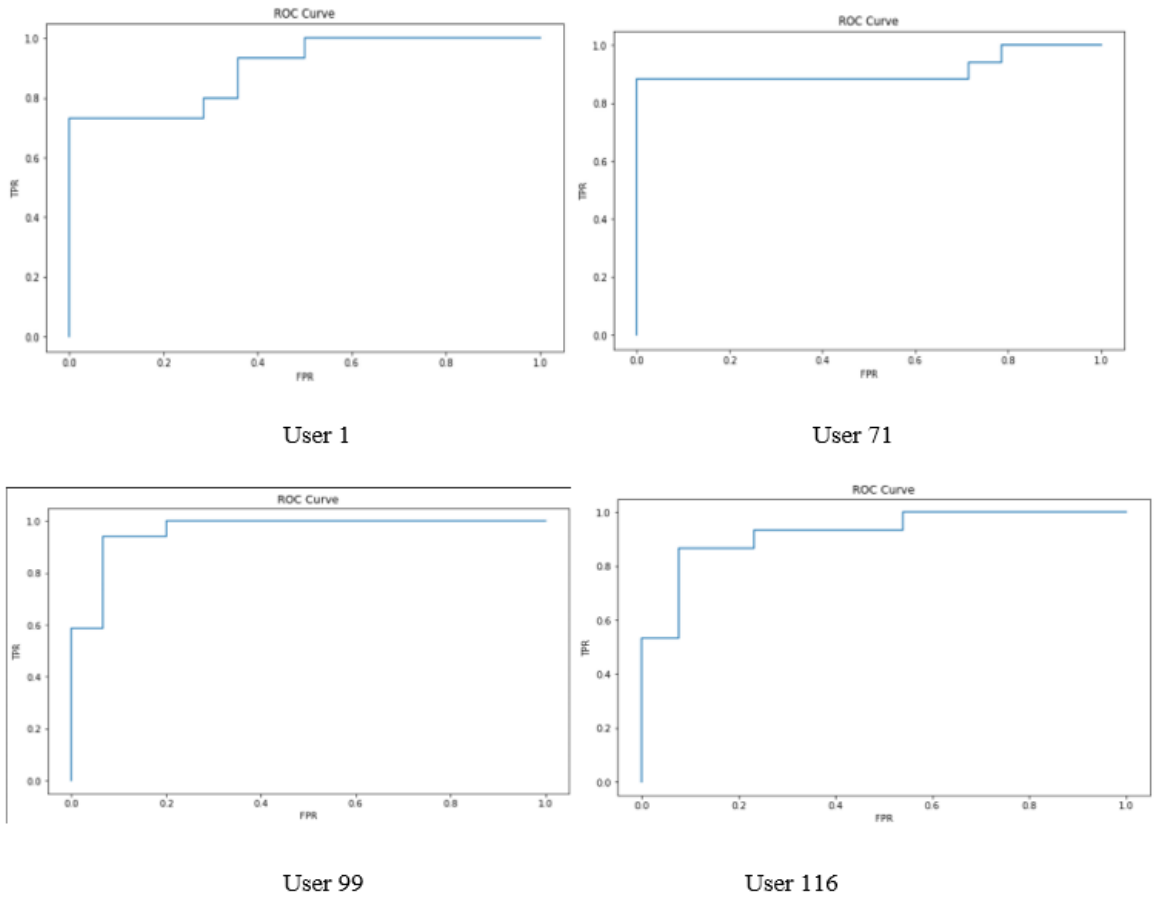

Figure 23: ROC Curve for 4 users

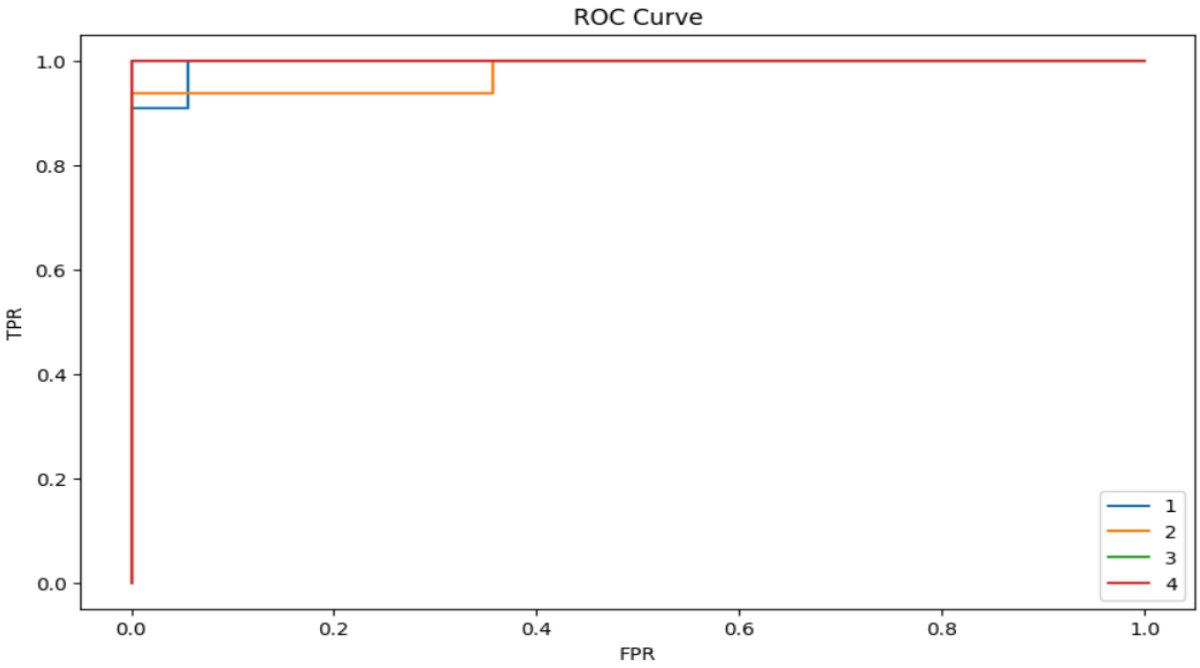

Figure 24 : ROC Curve for user 1, 2, 3 and 4

5.4 Comparison of Supervised Learning vs Reinforcement Learning results

Table 7: Supervised learning vs Reinforcement learning results from same dataset.

Metrics	Literature Review (2.1.11)	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	0.03 - 0.06 = Average 0.045	0.157	0.0323

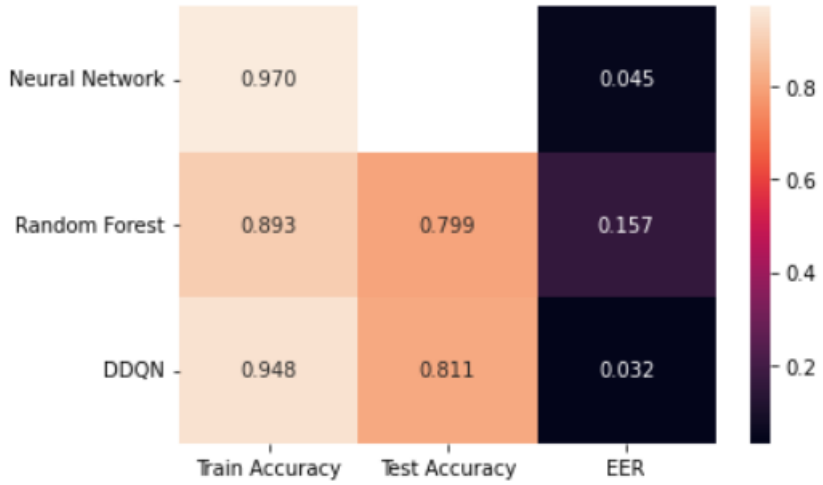

Figure 25 : Heatmap results for comparison between supervised learning and reinforcement learning.

Chapter 6

6.1 Future Scope/Direction:

The chapter explores the possibilities of refining existing methods and techniques for keystroke dynamics analysis, such as improving the accuracy of keystroke feature extraction algorithms or enhancing the performance of machine learning models. It also highlights opportunities for exploring new keystroke dynamics features, such as touch pressure, typing rhythm, or cursor movements, that could improve the accuracy of continuous authentication.

Moreover, the Future Scope/Direction discusses the practical applications of continuous authentication using keystroke dynamics, such as enhancing security in the workplace, online transactions, and other sensitive systems, and highlights areas where continuous authentication can be further explored, such as e-learning systems or remote workspaces.

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

1. **Mouse data**: In the current model, only keystroke data is used for authentication. However, mouse data can also provide valuable information about a user's typing pattern. Mouse data includes information such as cursor movement, clicks, and scrolling. Incorporating mouse data into the model can improve the accuracy of the model as it can provide additional information about the user's typing pattern.
2. **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.
3. **Augmentation techniques**: In the current model, the data is 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.

4. **Use auxiliary keys**: In the current model, only the duration of the key press and release events are used. However, the duration of press and release of auxiliary keys such as space, ctrl, etc can also be used to improve the accuracy of the model. Auxiliary keys can provide additional information about the user's typing pattern and improve the accuracy of the model.
5. One way to do this could be to use 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 pre-training 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.

6.2 Limitations:

The proposed reinforcement learning model for continuous authentication using behavioral biometrics also has some of the below stated limitations:

1. One limitation is that the recorded data is not diverse enough. The same text has been freely typed under the same constraints, which makes it difficult to capture the same user’s typing pattern in any other scenarios.
2. Another limitation is that it is difficult to train models on short length episodes or if the No parameter is too low because of the high variability among the events in small duration. The No parameter is the number of events in an observation, a low value for this parameter can lead to high variability among the events, making it difficult for the model to learn patterns.
3. Additionally, long pauses in typing can also make it tricky to understand the patterns. It is not clear how much longer a pause should be treated as an anomaly. Currently, the hold time priors are reset when any sample is corrupted.

The chapter 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.

Chapter 7

7. Model Deployment: Integrating FastAPI and Machine Learning for Continuous Behavioral Biometric Authentication

The traditional practices of security are failing slowly; new systems are needed to protect the information in the cyber world. The user authentication should be such that the systems are continuously learning and improving, and development should be fast paced without consuming too much time. The currently used continuous authentication systems have significant weaknesses in the huge data handling mechanisms including the run-time overhead taken in analysing the user profiles. The objective of this work is to overcome these weaknesses to be able to handle multiple requests simultaneously, improve the overall performance, and decrease the cost of the behavioural biometrics-based authentication systems. In other words, we aim to create a machine learning algorithm to create user-profiles that are capable to user's behavioral data of 64 bytes per second. The algorithm would provide over millions of user-profile recognitions per day through predictive techniques. To reach this target, we integrate the biometric behavior detection machine learning (ML) model, that doesn't natively run on the web, with frontend using FastAPI services. These services enable the users to access the model detection using the web browser for continuous authentication using behavioral biometrics. The evaluation and the experimental results showed that the performance of the ML model by using the FastAPI has been improved by almost 45% as compared to Flask. This research is published [35] and peer reviewed.

Though we have tested this approach for supervised learning Model. Based on the results obtained from the machine learning used for this experiment, we can say that RL model's performance would be better when it is deployed using FastAPI than Flask.

The major challenge nowadays is the selection of a framework that can deliver optimal performance for the large datasets suitable for production environments. The

industries while working on development solutions in python look for quick deployments that are fast and highly responsive for the users to access on web browsers. We performed some comparison results in two of the python frameworks suitable for our behaviour biometrics dataset to enhance the continuous authentication experience.

We performed some literature review and comparison for the frontend APIs (Table 8) currently used in the market.

Table 8: Based on the below comparison shown in Table, we then decided to compare the FastAPI and Flask for our hypothesis.

Based on	**Flask**	**FastAPI**
Complex API's	Not Suitable	very much suitable
Huge amount of data	Not Suitable	advisable for huge data
Query handling	Results in error and cannot handle multiple requests simultaneously.	can handle multiple requests simultaneously without affecting the response time and not errors reported.
Asynchronous tasks	No support for such tasks	Supports Asynchronous tasks
Security Breaches	Prone to security breaches	No Security breaches
Performance	Slower than FastAPI	Speed is quite good
Production	Not production ready, Requires lot of work	Scalable for deploying ML models quickly.

This proposed integration aims to present the new generation of continuous authentication system of behavioral biometrics using Machine Learning and FastAPI to reduce the fraud in cyberworld. **This study is different from the other integration models based on the below specified points:**

1. New methodology of continuous authentication using a combination of Machine Learning and FastAPI (a python framework).
2. Verifying the genuineness of user at every click and movement.
3. This adds a security layer to prevent the loss of information.

7.1 FastAPI Implementation:

Implementation of Fast API will make the ML model to predict the results faster and scalable. In case of any new features addition, it is easy to modify the Fast API end points.

Python Pickling for Authentication Prediction Dataset

Once the data set is processed with training set and test set, the best algorithm which is producing the highest accuracy is selected. Model behavior can be saved by using python pickle method. The python pickle model is used to serialize or de-serializing the python structure. Python object would then be pickled, and it will be saved on disk. The python pickle file is a character stream that contains all the required data that is needed to create the object in Fast API script as displayed in Fig 26. The Fig 26. is applicable for any Restful API services for Machine learning models irrespective of the framework used [34].

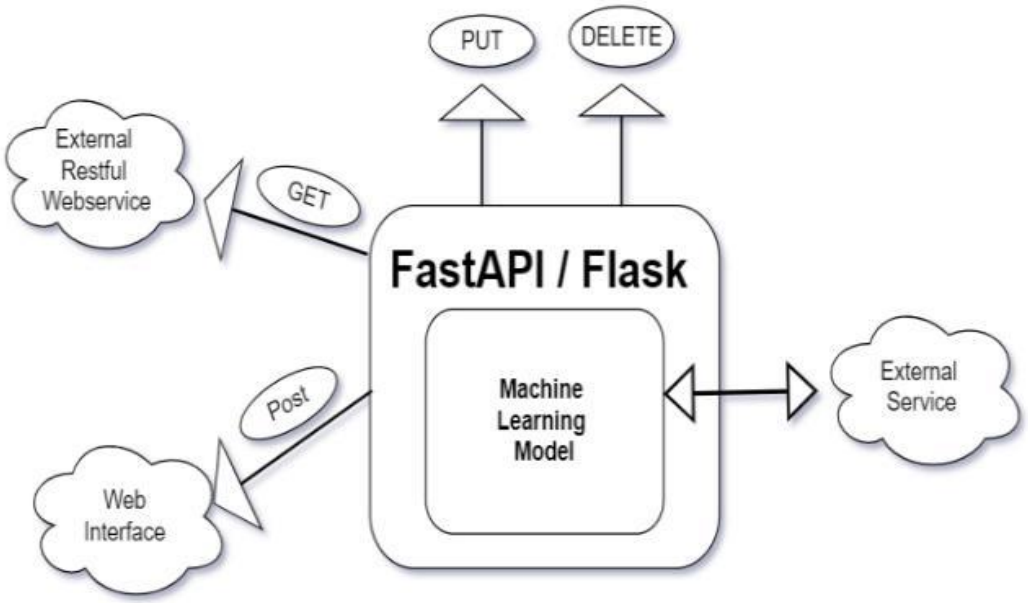

Figure 26: FastAPI Model

Building this model with FastAPI, accelerated the development. While setting up the model with FastAPI, the API endpoints accepting the query parameters can be set up as per the requirement of the application, with some minor changes in the original code and generate the automatic docs for the endpoints. The implementation with FastAPI, authentication takes very less time for data validation, allows execution of block of code without interfering the entire thread (asynchronous code) which is not possible with Flask. Also, there were no errors reported in the queries, and the response time for database sessions is reduced significantly [22]. As a result, the number of simultaneous users that this application can support will be much larger than in the case of Flask.

While setting up the model with Flask, the deployment time was lesser than that of FastAPI, but the major drawback is in terms of the time. Flask takes more time to process the requests as it does it in turns (one after the other) as it is not capable of handling multiple requests. It does start quickly for the first API call but when the data increases in the application, FastAPI remains persistent in performance than Flask.

In the static authentication systems where authentication happens only once, it becomes easier for the hackers to enter the system as no monitoring is performed after the session is unlocked. In this proposed continuous authentication system, the authenticity of the user is validated throughout the session even when the user is not present in front of the computer system which doesn't let the hackers to enter [26]. Hence, it helps to safeguard the user information and provide high security standards.

After the Fast API setup is done, the model can be deployed as microservice in production using Docker, Kubernetes, and a Linux system.

7.2 Experiments and Result:

Once API is designed with endpoints URLs, ML model is consumed with endpoints at front end. The api calls from user application to ML model follows the process shown in the above figure 26. When a user accesses the API, the client application would be sending a parameter or combination of parameters in the URL such as click length and button type along with user id to the Continuous authentication system without interrupting the user. Then the Fast API will invoke the trained ML model and returns the status of the authentication whether the user should be allowed to log in or not. When an event is detected on user application, API

inspection is an efficient way to monitor the behavior of keystroke loggers, since the use of keystroke related APIs is a typical behaviour of any software. The list of Click_length, button type and screen id along with user_id would be passed to the endpoint as a get/put method. The API would then run the prediction based on the biometrics parameters passed [23].

As an output, it would show 1 for successfully authenticating the legitimate user and 0 if the user authentication fails and hence will not be allowed to login. It is possible to have multiple endpoints and microservices using FastAPI depending on the expected output. The Table 9 shows some of the prediction values for click length list passed for sample users.

To measure the response time and compare the results between FastAPI and Flask, the readings can be observed from the response headers tab for each API call. Additionally, async entry point can be added to compare the results. Another methodology that can be used is through get the response time and details through Postman platform.

In the below graph (Figure 27) shows the comparison between the response time of FastAPI and Flask. It can be observed that the FastAPI's response time is quite lesser than Flask for all the API' calls. The response time is measure in milliseconds. As our goal is to choose the highly responsive framework, there are multiple other performance metrics that can be compared other than the response time. The metrics could be time per concurrent request, total data transferred, transfer rate, etc. The first API call in FastAPI takes the longest time than the other subsequent calls [24]. This indicates that the start-up time is high. In Flask, all the calls take similar time, but the response time is quite high than FastAPI overall. The FastAPI response time for even the longest call is 7.2x lesser than that of Flask. When we measure it in terms of percentage, the FastAPI' s performance is 45% faster on average depending upon the number of requests sent. In some cases, FastAPI outperforms flask framework by more than 100%.

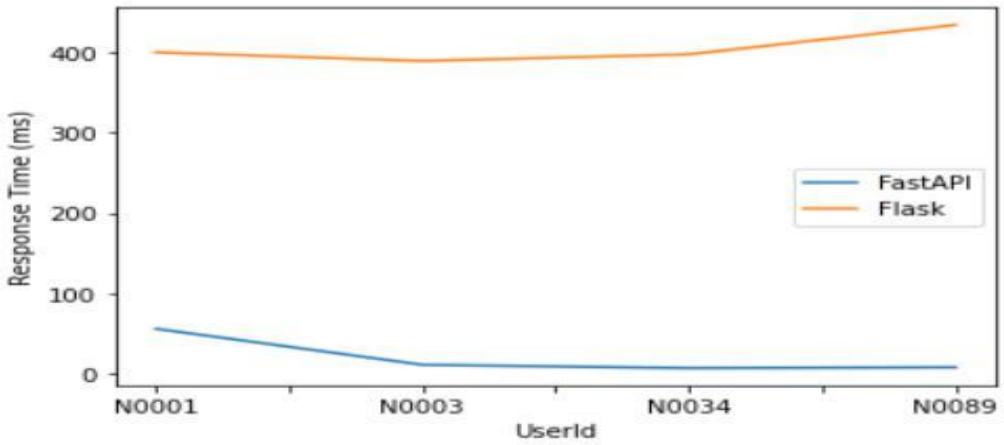

Figure 27: Comparison response time between FastAPI and Flask

Based on experimental results (in Table 10) obtained with more than 800 records per user, the proposed method achieved the best accuracy (0.23) with an authentication time of less than 5 sec using machine learning and FastAPI. The threshold of 800 records is used for this study and this can be changed as per the data or the business requirement.

Table 9: Prediction Result

Sample User id	Actual Click_length list	Predicted Click_length list	Output	FastAPI Response Time (ms)	Flask Response Time (ms)
N0001	[32,556,767]	[33,556,765]	1	55.4	400.3
N0003	[23,665,33,56]	[15,454,67,46]	0	10.5	389.6
N0034	[34,767,8787,435,65]	[33,765,8785, 436,65]	1	6.4	397.8
N0089	[128,256,512,1024]	[128,256,515,1024]	1	7.4	434.5

7.3 Conclusion:

Behaviour biometrics have promising potentials to strengthen the security of user account and reduce number of the threats and vulnerabilities without additional hardware requirements. This study gives a comprehensive review of the research work or efforts made on integration of ML model with FastAPI for behavioural biometrics to provide the cost and time efficient solution. The development is fast due to editor autocompletion, and automatic error checks offered by the framework [27].

There are very few numbers of features in each dataset, consequently, the increased number of users might reduce the accuracy of the model, as the values in each dataset will repeat and cause difficulty to distinguish between users. In this study, if any user has more than the

threshold records in the database is allowed to go through the prediction model. If the record per user is less than the set threshold, then the user is automatically authenticated until the numbers of events for him reaches the threshold. This is done to reduce to False rejection rate (FRR).

7.4 Future Scope:

There is a wide scope for research in continuous authentication using behavioural biometrics such as keystroke dynamics and Mouse pointer movement. The idea of implementing the Continuous Authentication model specifically with FastAPI is novel. Furthermore, model can further be hyper tuned to improve the accuracy and generate better results. The accuracy of the model can further be improved after removing the outliers using Standard deviation method depending upon the data collected for analysis.

Chapter 8

8. Conclusion

From the results discussed in chapter 5, it can be 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 cyber-attacks and unauthorized access.
- RL model can be deployed on client side where the model can adapt to learn the change in user behaviour 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, unigraphs or digraphs [11] are included as a part of the analysis. One of the key benefits to include all keys (full keyboard) is that authentication system doesn't have to wait for the user to specifically press the keys for it to analyse and then classify it as genuine or imposter.
- 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 which is at par with any of the supervised learning models discussed in the literature review chapter along with overcoming the limitations of supervised learning.

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.

Moreover, the model can detect when the user’s 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.

Finally, the chapter 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.

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 of 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 user's identity throughout the session, reducing the risk of identity theft and fraud.

References

1. Jianwei Li, Han-Chih Chang, Mark Stamp, “Free-Text Keystroke Dynamics for User Authentication”, 2021, doi: https://doi.org/10.48550/arXiv.2107.07009
2. GOLD'S TAKE ON TECH “Traditional security is dead -- why cognitive-based security will matter”11 May 2016 [Online], Available: https://www.computerworld.com/article/3068185/traditional-security-is-dead-why-cognitive-based-security-will-matter.html
3. A. Salem, D. Zaidan, A. Swidan and R. Saifan, "Analysis of Strong Password Using Keystroke Dynamics Authentication in Touch Screen Devices," 2016 Cybersecurity and Cyberforensics Conference (CCC), Amman, Jordan, 2016, pp. 15-21, doi: 10.1109/CCC.2016.11.
4. N. Jeanjaitrong and P. Bhattarakosol, "Feasibility study on authentication based keystroke dynamic over touch-screen devices," 2013 13th International Symposium on Communications and Information Technologies (ISCIT), Surat Thani, Thailand, 2013, pp. 238-242, doi: 10.1109/ISCIT.2013.6645856.
5. M. Antal and L. Z. Szabó, "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, Bucharest, Romania, 2015, pp. 343-350, doi: 10.1109/CSCS.2015.16.
6. J. -h. Roh, S. -H. Lee and S. Kim, "Keystroke dynamics for authentication in smartphone," 2016 International Conference on Information and Communication Technology Convergence (ICTC), Jeju, Korea (South), 2016, pp. 1155-1159, doi: 10.1109/ICTC.2016.7763394.
7. H. Saevanee and P. Bhattarakosol, "Authenticating User Using Keystroke Dynamics and Finger Pressure," 2009 6th IEEE Consumer Communications and Networking Conference, Las Vegas, NV, USA, 2009, pp. 1-2, doi: 10.1109/CCNC.2009.4784783.
8. L. C. F. Araujo, L. H. R. Sucupira, M. G. Lizarraga, L. L. Ling and J. B. T. Yabu-Uti, "User authentication through typing biometrics features," in IEEE Transactions on Signal Processing, vol. 53, no. 2, pp. 851-855, Feb. 2005, doi: 10.1109/TSP.2004.839903.

9. Fabian Monrose, Aviel D. Rubin, Keystroke dynamics as a biometric for authentication, Future Generation Computer Systems, Volume 16, Issue 4, 2000, Pages 351-359, ISSN 0167-739X, https://doi.org/10.1016/S0167-739X(99)00059-X.
10. Antal, M., & Nemes, L. (2016). The MOBIKEY Keystroke Dynamics Password Database: Benchmark Results. *Computer Science On-line Conference*.
11. Giuseppe Stragapede, Ruben Vera-Rodriguez, Ruben Tolosana, Aythami Morales, BehavePassDB: Public Database for Mobile Behavioral Biometrics and Benchmark Evaluation, Pattern Recognition, Volume 134, 2023, 109089, ISSN 0031-3203,
12. 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. https://doi.org/10.3390/make4020023
13. A. K. Belman, S. Sridhara and V. V. Phoha, "Classification of Threat Level in Typing Activity Through Keystroke Dynamics," 2020 International Conference on Artificial Intelligence and Signal Processing (AISP), Amaravati, India, 2020, pp. 1-6, doi: 10.1109/AISP48273.2020.9073079.
14. Amith K. Belman, Li Wang, Sundaraja S. Iyengar, Pawel Sniatala, Robert Wright, Robert Dora, Jacob Baldwin, Zhanpeng Jin, Vir V. Phoha. (2019). SU-AIS BB-MAS (Syracuse University and Assured Information Security - Behavioral Biometrics Multi-device and multi-Activity data from Same users) Dataset . IEEE Dataport. https://dx.doi.org/10.21227/rpaz-0h66
15. Sutton, R. S., & Barto, A. G. (2018). Reinforcement learning: An introduction (2nd ed.). The MIT Press.
16. Gupta, Shruti (2022): USER ATTRIBUTION IN DIGITAL FORENSICS THROUGH MODELING KEYSTROKE AND MOUSE USAGE DATA USING XGBOOST. Purdue University Graduate School. Thesis. https://doi.org/10.25394/PGS.19570726.v1
17. J. Huang, D. Hou, S. Schuckers, T. Law and A. Sherwin, "Benchmarking keystroke authentication algorithms," 2017 IEEE Workshop on Information Forensics and Security (WIFS), Rennes, France, 2017, pp. 1-6, doi: 10.1109/WIFS.2017.8267670.
18. R. Chandok, V. Bhoir and S. Chinnaswamy, "Behavioural Biometric Authentication using Keystroke Features with Machine Learning," 2022 IEEE 19th India Council International Conference (INDICON), Kochi, India, 2022, pp. 1-6, doi: 10.1109/INDICON56171.2022.10040168.

19. Sedat Akleylek, Kadir Yunus Koc and Nursah Cevik “Keystroke Dynamics Based Authentication System”- 2021 6th International Conference on Computer Science and Engineering (UBMK)
20. Ananya and Saurabh Singh’s Keystroke Dynamics for Continuous Authentication- 2018 8th International Conference on Cloud Computing, Data Science & Engineering (Confluence)
21. Yunji Liang , Sagar Samtani, Bin Guo , and Zhiwen Yu ‘s Behavioural Biometrics for Continuous Authentication in the Internet-of-Things Era: An Artificial Intelligence Perspective- IEEE Internet of Things Journal (Volume: 7, Issue: 9, Sept. 2020)
22. Jigyasa Handa, Saurabh Singh and Shipra Saraswat’s A comparative study of Mouse and Keystroke Based Authentication- 2019 9th International Conference on Cloud Computing, Data Science & Engineering (Confluence)
23. Soumik Mondal and Patrick Bours Continuous Authentication using Behavioural Biometrics- Collaborative European Research Conference (CERC) 2013
24. Claudia Audia Trianti, Budhi Kristiano and Hendry’s Integration of Flask and Python on The Face Recognition Based Attendance System- 2021 2nd International Conference on Innovative and Creative Information Technology (ICITech)
25. C. A. Trianti, B. Kristianto and Hendry, "Integration of Flask and Python on The Face Recognition Based Attendance System," 2021 2nd International Conference on Innovative and Creative Information Technology (ICITech), 2021, pp. 164-168, doi: 10.1109/ICITech50181.2021.9590122.
26. Tivadar Danka’s “How to properly ship and deploy your machine learning model” Jan 29, 2020 [Online]. Available: https://towardsdatascience.com/how-to- properly-ship-and-deploy-your-machine-learning- model-8a8664b763c4
27. A. G. Mirabella, A. Martin-Lopez, S. Segura, L. Valencia-Cabrera and A. Ruiz-Cortés, "Deep Learning- Based Prediction of Test Input Validity for RESTful APIs," 2021 IEEE/ACM Third International Workshop on Deep Learning for Testing and Testing for Deep Learning (DeepTest), 2021, pp. 9-16, doi: 10.1109/DeepTest52559.2021.00008.
28. I. Deutschmann, P. Nordström and L. Nilsson, "Continuous Authentication Using Behavioral Biometrics," in IT Professional, vol. 15, no. 4, pp. 12-15, July-Aug. 2013, doi: 10.1109/MITP.2013.50.
29. Y. Barlas, O. E. Basar, Y. Akan, M. Isbilen, G. I. Alptekin and O. D. Incel, "DAKOTA: Continuous Authentication with Behavioral Biometrics in a Mobile

Banking Application," 2020 5th International Conference on Computer Science and Engineering (UBMK), 2020, pp. 1-6, doi: 10.1109/UBMK50275.2020.9219365.

30. P. M. A. B. Estrela, R. de Oliveira Albuquerque, D. M. Amaral, W. F. Giozza, G. D. A. Nze and F. L. L. de Mendonça, "Biotouch: a framework based on behavioral biometrics and location for continuous authentication on mobile banking applications," 2020 15th Iberian Conference on Information Systems and Technologies (CISTI),2020, pp.1-6, doi: 10.23919/CISTI49556.2020.9140948.
31. E. Opoku-Mensah, Y. S. Bandoh, J. Li, J. B. Ayekai and B.C. Mawuli, "Behavioral Biometrics for Human Identity Corroboration based on Gesture-Signature with Deep Learning," 2020 17th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP), 2020, pp. 146- 152, doi: 10.1109/ICCWAMTIP51612.2020.9317330.
32. Sailpoint "What is Identity-as-a-Service (IDaaS)?" (2021, Dec 21), [Online]. Available: https://www.sailpoint.com/identity-library/identity-as-a- service/
33. W.-H. Lee and R. B. Lee, "Implicit smartphone user authentication with sensors and contextual machine learning," in Proc. 47th Annu. IEEE/IFIP Int. Conf. Depend. Syst. Netw. (DSN), 2017, pp. 297–308
34. C. A. Trianti, B. Kristianto and Hendry, "Integration of Flask and Python on The Face Recognition Based Attendance System," 2021 2nd International Conference on Innovative and Creative Information Technology (ICITech), 2021, pp. 164-168, doi: 10.1109/ICITech50181.2021.9590122.
35. Nikhil Verma, Krishna Prasad," Responsive parallelized architecture for deploying deep learning models in production environments" arXiv:2112.08933 on 15 Dec 2021
36. Suman Das "Concurrency with FastAPI "19 Jan 2022 [Online], Available: https://medium.com/cuddle- ai/concurrency-with-fastapi-1bd809916130
37. P. Bansal and A. Ouda, "Study on Integration of FastAPI and Machine Learning for Continuous Authentication of Behavioral Biometrics," 2022 International Symposium on Networks, Computers and Communications (ISNCC), Shenzhen, China, 2022, pp. 1-6, doi: 10.1109/ISNCC55209.2022.9851790.
38. Attinà, F. (2016). Traditional Security Issues. In: Wang, J., Song, W. (eds) China, the European Union, and the International Politics of Global Governance. Palgrave Macmillan, New York. https://doi.org/10.1057/9781137514004_10

Curriculum Vitae

Name: Priya Bansal

Post-secondary Education and Degrees:

Chitkara University
Baddi, Himachal Pradesh, India
2010-2014 B.E.

The University of Western Ontario
London, Ontario, Canada
2021-2023 M.E.Sc.

Related Work Experience

Teaching Assistant
The University of Western Ontario
2021-2023

Data Science Intern
F8th Inc
2021-2022

Data Science Intern
XLScout Pvt. Ltd.
2022-2023

Publications:

P. Bansal and A. Ouda, "Study on Integration of FastAPI and Machine Learning for Continuous Authentication of Behavioral Biometrics," 2022 International Symposium on Networks, Computers and Communications (ISNCC), Shenzhen, China, 2022, pp. 1-6, doi: 10.1109/ISNCC55209.2022.9851790.