Summer 7-1-2022

Adversarial Activity Detection and Prediction Using Behavioral Biometrics

Amin Fallahi
Syracuse University

Abstract

Behavioral biometrics can be used in different security applications like authentication, identification, etc. One of the trending applications is predicting future activities of people and guessing whether they will engage in malicious activities in the future. In this research, we study the possibility of predicting future activities and propose novel methods for near-future activity prediction.

First, we study gait signals captured using smartphone accelerometer sensor and build a model to predict a future gait signal. Activity recognition using body movements captured from mobile phone sensors has been a major point of interest in recent research. Data that is being continuously read from mobile sensors can be used to recognize user activity. We propose a model for predicting human body movements based on the previous activity that has been read from sensors and continuously updating our prediction as new data becomes available. Our results show that our model can predict the future movement signal with a high accuracy that can contribute to several applications in the area.

Second, we study keystroke acoustics and build a model for predicting future activities of the users by recording their keystrokes audio. Using keystroke acoustics to predict typed text has significant advantages, such as being recorded covertly from a distance and requiring no physical access to the computer system. Recently, some studies have been done on keystroke acoustics, however, to the best of our knowledge none have used them to predict adversarial activities. On a dataset of two million keystrokes consisting of seven adversarial and one benign activity, we use a signal processing approach to extract keystrokes from the audio and a clustering method to recover the typed letters followed by a text recovery module to regenerate the typed words. Furthermore, we use a neural network model to classify the benign and adversarial activities and achieve significant results: (1) we extract individual keystroke sounds from the raw audio with

91% accuracy and recover words from audio recordings in a noisy environment with 71% average top-10 accuracy. (2) We classify adversarial activities with 93% to 98% average accuracy under different operating scenarios.

Third, we study the correlation between the personality traits of users with their keystroke and mouse dynamics. Even with the availability of multiple interfaces, such as voice, touch, etc., keyboard and mouse remain the primary interfaces to a computer. Any insights on the relation between keyboard and mouse dynamics with the personality type of the users can provide foundations for various applications, such as advertisement, social media, etc. We use a dataset of keystroke and mouse dynamics collected from 104 users together with their responses to two personality tests to analyze how their interaction with the computer relates to their personality. Our findings show that there are considerable trends and patterns in keystroke and mouse dynamics that are correlated with each personality type.

Adversarial Activity Detection and Prediction Using Behavioral Biometrics

by

Amin Fallahi

B.S., Iran University of Science and Technology, 2015

Submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

in

Computer & Information Science and Engineering

Syracuse University

July 2022

Dedication

To my family; may this achievement in my life and career gets me closer to reunion with them after years of distance due to immigration complexities.

Acknowledgements

First and foremost, I would like to praise and thank God, who has granted me strength and encouragement throughout all the challenging moments of this journey.

I would like to thank my advisor, Prof. Vir V. Phoha, for his kind guidance and support throughout my Ph.D. career. His great vision and technical knowledge have helped me a lot to achieve success in this research.

I would like to thank the great faculty of Syracuse University Department of Electrical Engineering and Computer Science and the Syracuse University family at large for providing me with a great learning and living experience. I am thankful for the support I got from the leadership, including Prof. Jae C. Oh and all the department staff who have supported me through the challenges in my Ph.D. career.

I want to thank my great dissertation committee, Professors Shiu-Kai Chin, Can Isik, Venkata S.S. Gandikota, and Edmund Szu-Li Yu for their guidance and constructive criticism. Further, I am thankful to Prof. Dawit Negussey for chairing my doctoral defense.

Most of all, this dissertation is dedicated to my beloved parents and my sister for their unconditional love and support.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview of the Dissertation

The use of behavioral biometrics for security applications has been a trending topic in recent research. Behavioral biometrics refers to the patterns and activities which are unique to the human activities and the way humans behave. In contrast, physical biometrics refers to the natural characteristics of people [1]. Behavioral biometrics can be used for different applications including authentication, identification, and attack detection. Among the applications of behavioral biometrics is to detect malicious activities and adversarial behavior in computing environments. As defined by NIST, malicious activities are "Activities, other than those authorized by or in accordance with U.S. law, that seek to compromise or impair the confidentiality, integrity, or availability of computers, information or communications systems, networks, physical or virtual infrastructure controlled by computers or information systems, or information resident thereon." [2]. Examples of malicious activities include viruses and malwares, activities that result in data loss and theft, scanning and attacking networks, spamming emails, and disabling or removing security monitoring systems [3]. Furthermore, NIST

defines adversary as "An entity that is not authorized to access or modify information, or who works to defeat any protections afforded the information." [4]. In the scope of our research, we define adversaries as the users of a computer system who intend to interfere with the proper functioning of the system. Similar to the NIST definition, we define malicious activities as the activities that the adversaries do to interfere with the proper functioning of the system.

In this research, we study and analyze different aspects of adversarial activity detection using behavioral biometrics through three projects:

First, we try to project the future activities of the users so we can predict whether the user will engage in malicious activities in the future. To address this, we use the motion sensors of smartphones, particularly the accelerometer sensor to analyze the movement patterns of the user during different activities like walking and biking. Then, we try to predict the future movements of the user based on the past movement signal. Having the future movements signal of the user can help us predict whether the user is likely to engage in malicious activities. The use of mobile phone motion sensors has been addressed in several studies before, in applications like identification and authentication. However, our goal is not to identify the user, but to get more information about the future gait signal of the user.

Second, we try to predict adversarial activities without having access to the users' devices or violating their privacy by installing software on their mobile phones to collect their movement data. For this, we study other important biometric sensors which are keystrokes and audio. We focus on predicting the activity of the user by listening to the user's typing sound. Research has shown different keys on a keyboard make different sounds that are distinguishable and it is possible to recover the pressed keystrokes by recording the typing session audio. So, we propose a model for recovering the typed keys using their audio and then using the recovered text to predict the activity of

the user and whether the user will engage in malicious activity in the near future by assigning a threat score to the user's activity session. We make this problem much more challenging by using a noisy dataset in an environment similar to a real-world office with several people and noises present.

Third, we do an empirical analysis of the effects of personal traits and characteristics on the way the user interacts with the computer. More specifically, we use two psychological tests to determine the personal characteristics of the users and analyze the correlation between the personal traits of the user with the user's typing dynamics.

As a result of this research, we have studied different aspects of behavioral biometrics and how they can be used in detecting and predicting adversarial activities. To summarize, our thesis statement is to find out how different behavioral biometrics expose the vulnerability of the systems or the security threats against them.

1.2 Dissertation Roadmap

The rest of the dissertation is organized as follows: in chapter 2, we present our work on human gait prediction using motion sensors and discuss it in detail. Chapter 3 discusses our study on detecting threats against a system by analyzing the typing audio of its users. Chapter 4 presents an analysis of the correlation between personal traits of people and their behavior in interaction with a computer system. Finally, in chapter 5, we conclude the thesis by summarizing our findings.

Chapter 2

Looking Into Your Future: A Continuous Human Gait Prediction for Near Future

2.1 Introduction

The ability to predict the future actions of human beings is an interesting topic in forensics research. Early prediction of future activities leads to the detection of malicious and criminal activities. Many criminal activities are connected to human body movements because a criminal usually walks and moves when committing a crime. Also, with the emergence of the COVID-19 pandemic as one of today's major problems, people use masks in public and cover their faces which makes identification more challenging. Body movements during walking activity can be used for analyzing the behavior and the action of the criminals. Furthermore, being able to predict the future movements of the human body, can give early information about the future activities of a criminal. For example, by observing a gradual speed-up in the criminal's walking behavior, we

can predict that they may be escaping from something and take action to prevent it. Our research addresses this problem by proposing a model for forecasting the future gait signal of humans to be used for future activity detection.

The use of smartphone sensors has been an interesting topic of research during the past years. The data that can be captured from smartphone sensors have been used in different applications. For example, instead of traditional authentication methods like passwords, pins, and fingerprints, smartphone sensor recordings have been used to authenticate users based on their movements and behavioral activities. Behavioral biometrics are harder to exploit and safer to implement in these applications. Another application of these sensors is activity recognition. For example, smartphone sensors can be used for counting the user's steps or monitoring their health. Also, these sensors can be used for applications in digital forensics. For example, by fingerprinting the movements or walking patterns of a user, it is possible to predict their activity [5, 6], find out if they are committing a crime [7], or identify a specific person [8, 9]. But the challenge in this area is the availability of data. Because usually, we do not have access to enough data from the users to be able to accurately identify them, predict their activity, or analyze their behavior.

Neural networks have achieved notable results in predicting the future based on observing a history of available data. Among those, Long Short-Term Memory (LSTM) [10] has been used to forecast speech, handwriting, and other time-series data. In this work, we build a prediction model based on a neural network that learns the user's real-time phone movements and predicts the near future movements. Our model generates a future signal without having information about the activity which is being performed. It is trained with data from several activities such as walking and biking and predicts when similar activities are performed. For our application, we use data that is captured using the smartphone accelerometer sensor when it is in the user's pocket.

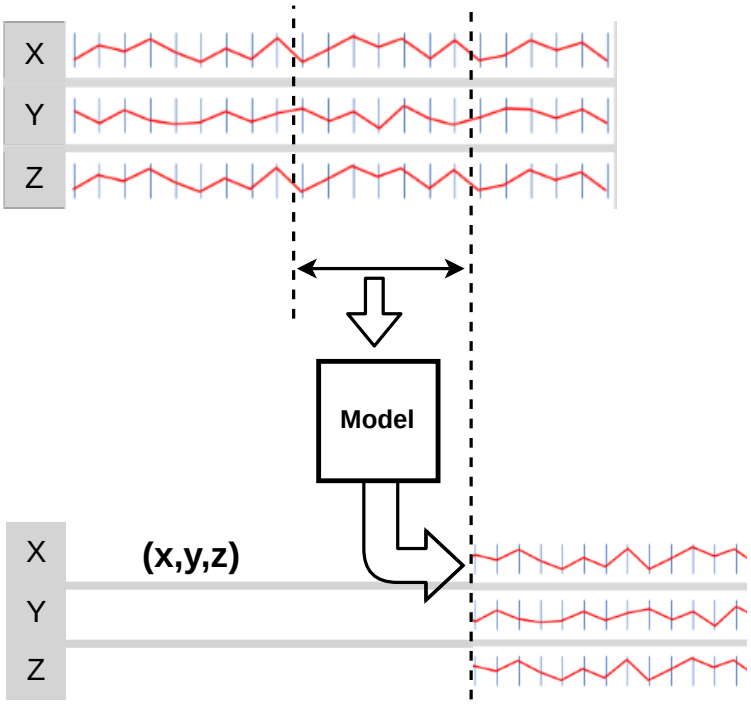


Figure 2.1: Overview of the problem. Movement signal (x, y, z) is fed into the model as input and the model generates the forecast of the future signal in the output.

The main contributions of this research are:

- A previously unexplored application of LSTM for forecasting the users' movement sequence while they perform different activities.
- Evaluating and comparing the performance of our movement prediction model for multiple users and different activities with the user's gait data captured using the accelerometer sensor in the user's smartphone. The analysis of the dataset of four different activities performed by 9 different users shows that our prediction model forecasts the near future movement signal with a small DTW distance between the predicted and actual signal.
- Discussion on various applications of the user's future movement prediction.

2.2 Related Work

As surveyed by Saini et al. [7], several biometric sensors can be used for forensic identification, from older technologies like fingerprint, palm print, voice, etc. to newer emerging ones like gait, odor, and keystroke dynamics. The use of gait for forensic identification has many advantages in comparison with other biometrics. First, most people walk, and gait data can be collected from them. Second, gait information can be collected non-intrusively, without the subjects knowing that it is being collected. In contrast with biometrics like fingerprint that needs the subject to collaborate and provide their finger for scanning, this is an important feature for gait that it does not directly need the subject's consent.

Multiple studies have addressed human identification using gait video data recorded by CCTV cameras. Zhang et al. [8] use gait image data with a CNN model to identify the subjects and correlate it with their demographic information. They do a thorough study on multiple gait factors, compare their results with similar work, and achieve better results in identification. While some of the vision-based methods mention and address the several problems that come with person identification using images, including different obstacles, camera angles, and clothing of the subjects, these models are still vulnerable to the looks of the criminals in forensics applications, because they can easily be modified and obscured by the subjects. Studies that use motion sensors to capture gait patterns and identify subjects are less prone to these vulnerabilities because they are usually not affected by the change in the subject's appearance, camera placement, obstacles, etc.

Several studies have made use of biometric features of the human body using different sensors for security and privacy applications. One major point of study in this area is the use of wearable and mobile sensors for continuously authenticating users

to access systems. Continuous authentication uses time-series data captured from biometric sensors to build a trust model on users as they continue interacting with the system and more data is becoming available. Kumar et al. [11] collect movement data from multiple users while performing different kinds of activities using smartphone sensors and build a model to classify users for continuous authentication with unlabeled data. Their classification results using various algorithms show that their model does not work for every smartphone user when the collected data is for movements of arbitrary activity. Our proposed model uses similar unlabeled sensor readings to predict the upcoming movements and may be able to improve their results because, with our predictions, more data is available during each session. Kwapisz et al. [9] collect smartphone sensors data for different activities while the phone is in the user's pocket and propose a model to identify and authenticate the users based on their data. Their model can identify the users in a sequence of activities as low as 10 seconds. Similarly, they authenticate the user using 10 seconds of data. After identifying the user, since the users' activity patterns are known and the model is already trained with them, our model can predict the 10 seconds of data that is needed for authentication based on fewer observations, thus making decisions faster. Several other works [12, 13, 14, 15, 16] have studied continuous authentication systems using human biometrics which can make use of our model for access to more data and potentially earlier authentication.

In another line of research, Guan et al. [5] design an LSTM-based model for activity recognition. However, their model is limited to recognition and cannot foresee future events and activities. Minor et al. [6] collect a dataset of users' activities based on their location during an extended amount of time using multiple smart-home sensors. They propose a model for predicting activity occurrence times in the future. The difference between their work and ours is that they label the data with the activity and use a sequence of activities to predict future activities. Also, their model is based on static data and is not designed to predict using live and continuous data readings. Cao et

al. [17] focus on recognizing human activities based on partially observed streams of videos. However, they are not filling up the missing parts of the videos. Our model can contribute to filling the missing parts and improve the accuracy of the proposed methods for activity recognition based on sensors.

Finally, most of the research in this area has been done for user identification or activity recognition. Our research focuses on forecasting human movements based on their history of activities and generating a predicted signal that is close to the actual signal. One of the major concerns in continuous authentication and activity recognition systems is to authenticate the user with maximum trust in a short amount of time. However, the data is not available or not enough for decision-making until the user does the activity for a certain amount of time. Our research can provide data for a longer period of user activity based on a small amount of data and can make the decision-making in authentication and activity recognition systems faster.

2.3 Methodology

In this section, we present the data we use for our studies and discuss its features and characteristics. Then, we discuss the architecture of our proposed forecast model in detail.

2.3.1 Data Description

For this study, we use Heterogeneity Activity Recognition Data Set [18] which is collected by Stisen et al. and is publicly available at the UCI Machine Learning Repository [19]. The data consists of accelerometer and gyroscope recordings collected from smartphone and smartwatch sensors for 9 users performing 6 actions during multiple sessions.

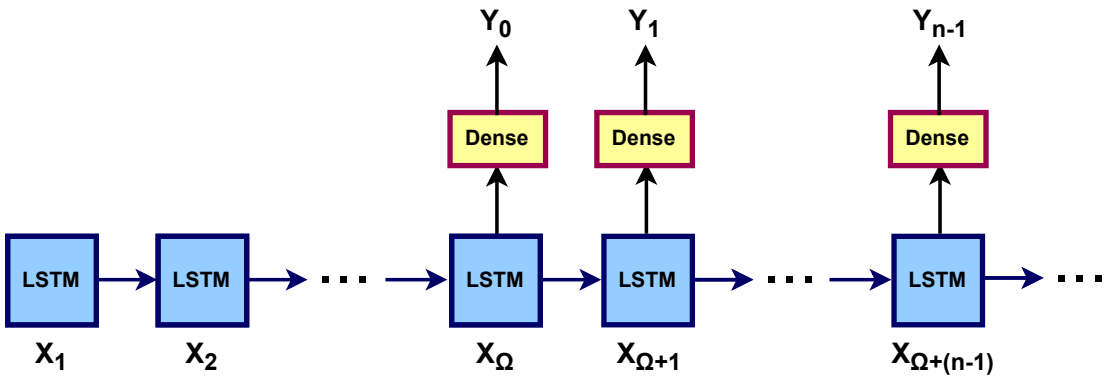


Figure 2.2: Architecture of the proposed model for forecasting the future signal.

The actions during each session are walking, biking, stairs up, stairs down, sitting down, and standing up and the sensor data has been read at 100Hz. We only use accelerometer data and discard sitting down and standing up data which have limited samples captured and do not have a significant variation to fit our model. Also, we only keep x, y, and z values of the raw accelerometer data and discard all other features.

2.3.2 Model Architecture

LSTM neural networks are famous for their ability to remember and forecast time-series data. They have been commonly used to forecast weather conditions [20] and stock prices [21]. In biometrics, LSTM networks have been widely used for recognizing speech [22] and handwriting [23] sequences. They are also used in natural language processing systems for predicting context based on the sequence of words that appeared in the past [24]. Another feature of LSTM is that it pays more attention to the most recent data [24], so we may only feed the latest available readings for making predictions. This leads to a faster and more accurate prediction that does not take old readings into account. However, feeding only a few late actual readings will result in the predictions being made based on small data that can be noisy, inaccurate, or inadequate. So, the prediction will be unreliable.

The properties of LSTM make it promising for our application. Thus, we develop an LSTM neural network to predict the gait signal based on previous actual information captured from users' smartphones. Our training session starts by reading the first activity session for each user. We scale the x, y, and z readings of accelerometer data to fall in a fixed range. Each of our training samples consists of Ω readings as the input data that is used for making the prediction and a single reading which is the result of the prediction. So, the input data is a matrix of shape $\Omega \times 3$ where Ω is the number of input samples and 3 is the size of the x, y, z vector. The output of the network is a single vector of size 3 which is the predicted data for the next $(\Omega + 1^{st})$ sample. As can be seen in Fig. 2.2, inputs X_1 to X_Ω will be fed to the LSTM layer. Then, the network generates a single output Y_0. Afterwards, the next input $X_{\Omega+1}$ is fed to the network to generate the next prediction Y_1. New samples are continuously fed into the network as they become available. We build the LSTM network with mean absolute error (MAE) loss function and Adam optimizer [25]. We use a different session of user activity as validation data for each training epoch.

The testing data includes a separate session of all 4 activities each continuously done by every user. Each testing data instance is used to predict a single sample. Furthermore, each predicted sample contributes to the prediction of the next sample. When $\Omega = 10$, let 10 actual samples be $a_1..a_{10}$. Then, the model is expected to predict p_{11}^1 which corresponds to a_{11} where p indicates a predicted sample and a indicates an actual sample. While the data is captured at 100Hz, predicting a single sample only indicates 0.01 seconds of activity. So, in a real-world scenario, it is needed to predict more future samples. When the actual data is available, we can still use it for predicting the next step ignoring past predictions, otherwise, we can mix the information we have from actual and predicted data for our next prediction. In this case, if a_{11} is captured and available, the model uses $a_2..a_{11}$ for predicting p_{12}^1, otherwise, it uses $a_2, .., a_{10}, p_{11}$ to generate p_{12}^2. The subscript in p_{12}^2 is the sequence number and the superscript is the

prediction index which is used to show different predictions of a sample in the future using different available data.

$$
\begin{aligned}
a_1, a_2, ..., a_{10} &\rightarrow p^1_{11} \\
a_2, a_3, ..., a_{10}, a_{11} &\rightarrow p^1_{12} \\
a_2, a_3, ..., a_{10}, p_{11} &\rightarrow p^2_{12} \\
a_3, a_4, ..., a_{11}, a_{12} &\rightarrow p^1_{13} \\
a_3, a_4, ..., a_{11}, p_{12} &\rightarrow p^2_{13} \\
a_3, a_4, ..., p_{11}, p_{12} &\rightarrow p^3_{13} \\
&... \\
\end{aligned}
\tag{2.1}
$$

When actual data is available, the model will add more weight to samples that are predicted based on more actual samples and present an average:

$$
\begin{aligned}
P_{12} &= \frac{\alpha_1 p^1_{12} + \alpha_2 p^2_{12}}{\alpha_1 + \alpha_2} && \alpha_1 > \alpha_2 \\
P_{13} &= \frac{\alpha_1 p^1_{13} + \alpha_2 p^2_{13} + \alpha_3 p^3_{13}}{\alpha_1 + \alpha_2 + \alpha_3} && \alpha_1 > \alpha_2 > \alpha_3 \\
&... \\
P_x &= \frac{\sum_{i=1}^{n} \alpha_i p^i_x}{\sum_{i=1}^{n} \alpha_i} && \alpha_1 > ... > \alpha_n
\end{aligned}
\tag{2.2}
$$

2.4 Evaluation and Results

We performed a detailed analysis and evaluation of our prediction model using our dataset (see Section 2.3.1) that consists of various activities performed by 9 different users. We use one session of accelerometer data to test our model performance that includes the data corresponding to all the activities performed by a user in our dataset.

Table 2.1: Results of the KS two sample test for walking data, comparing the actual and the predicted signal for each coordinate and user.

User	**Statistic**			**P-value**		
	x	**y**	**z**	**x**	**y**	**z**
1	4.9E-1	4.8E-1	3.8E-1	5.6E-17	6.9E-16	1.3E-10
2	6.3E-1	6.1E-1	5.2E-1	9.2E-32	1.1E-28	1.8E-23
3	6.3E-1	7.5E-1	7.6E-1	6.7E-34	1.1E-45	3.7E-40
4	4.9E-1	3.8E-1	3.8E-1	1.9E-22	1.3E-12	2.8E-12
5	6.8E-1	6.4E-1	6.2E-1	7.3E-40	1.1E-33	8.0E-33
6	4.4E-1	4.3E-1	5.6E-1	1.5E-17	1.2E-14	3.5E-28
7	5.0E-1	9.0E-1	4.7E-1	3.5E-22	2.2E-68	8.8E-13
8	7.5E-1	2.9E-1	3.5E-1	1.8E-51	1.8E-7	7.4E-10
9	3.6E-1	3.1E-1	6.5E-1	1.7E-11	2.1E-7	1.5E-33
Avg	5.5E-1	5.3E-1	5.2E-1	1.9E-12	4.4E-8	9.7E-11

Table 2.2: Results of the KS two sample test for biking data, comparing the actual and the predicted signal for each coordinate and user.

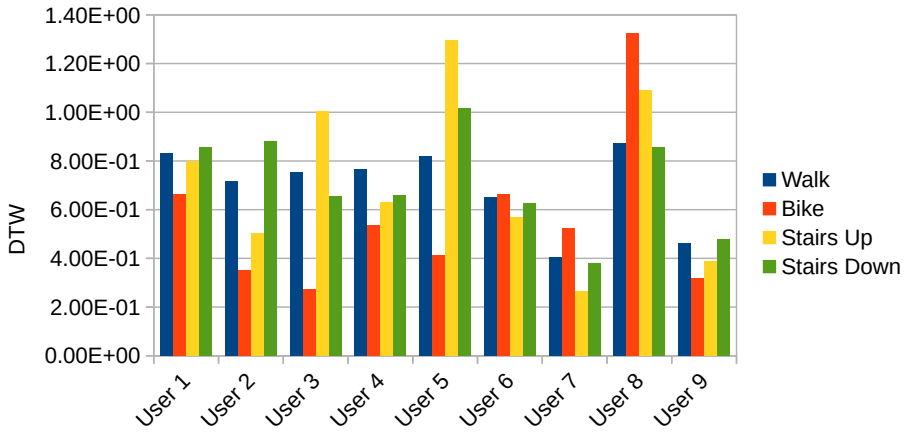
User	**Statistic**			**P-value**		
	x	**y**	**z**	**x**	**y**	**z**
1	7.1E-1	3.4E-1	4.7E-1	3.6E-35	1.3E-6	2.5E-19
2	5.9E-1	4.8E-1	8.9E-1	1.1E-24	4.5E-19	5.8E-67
3	6.6E-1	4.6E-1	9.4E-1	1.5E-35	8.8E-19	5.2E-77
4	9.1E-1	4.7E-1	8.5E-1	5.5E-69	2.8E-20	1.2E-63
5	6.1E-1	5.6E-1	8.4E-1	5.5E-28	6.8E-20	3.4E-49
6	9.2E-1	5.3E-1	8.6E-1	3.5E-76	1.8E-25	1.2E-63
7	8.4E-1	3.8E-1	3.5E-1	2.1E-61	2.8E-12	1.8E-7
8	8.8E-1	3.9E-1	4.1E-1	3.3E-68	1.3E-12	2.4E-11
9	4.1E-1	6.0E-1	9.8E-1	4.1E-13	6.7E-27	2.6E-87
Avg	7.3E-1	4.7E-1	7.3E-1	4.6E-14	1.4E-7	2.0E-8

Table 2.3: Results of the KS two sample test for stairs up data, comparing the actual and the predicted signal for each coordinate and user.

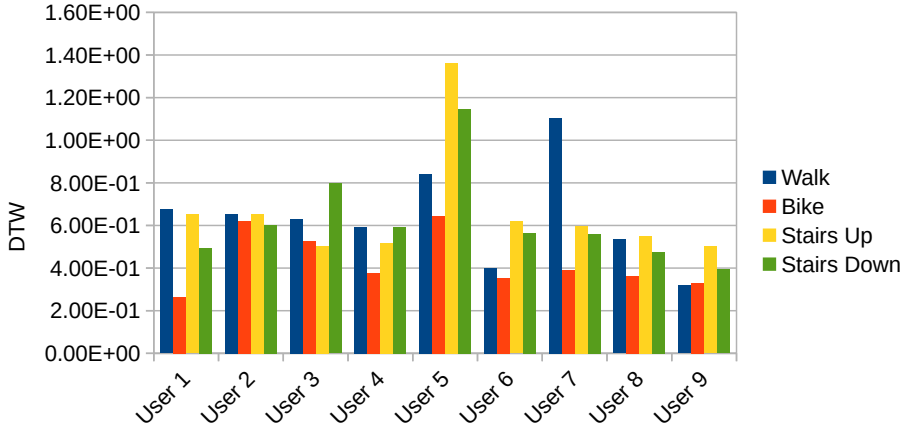
User	**Statistic**			**P-value**		
	x	**y**	**z**	**x**	**y**	**z**
1	5.7E-1	6.8E-1	3.9E-1	2.4E-28	1.3E-34	8.9E-12
2	5.7E-1	5.9E-1	5.3E-1	1.0E-27	2.0E-20	1.4E-22
3	8.7E-1	6.3E-1	9.5E-1	6.7E-66	1.8E-23	2.3E-75
4	5.0E-1	6.1E-1	4.8E-1	1.0E-21	3.4E-29	2.0E-20
5	5.6E-1	5.9E-1	6.3E-1	3.5E-28	3.4E-28	1.0E-26
6	4.6E-1	7.0E-1	7.3E-1	8.8E-19	6.0E-45	1.3E-47
7	5.0E-1	9.0E-1	4.7E-1	3.5E-22	2.2E-68	8.8E-13
8	8.0E-1	3.6E-1	4.0E-1	3.2E-52	4.1E-12	1.9E-10
9	5.7E-1	7.1E-1	4.5E-1	1.4E-25	1.7E-29	3.0E-14
Avg	6.0E-1	6.4E-1	5.6E-1	9.8E-2	4.5E-13	2.2E-11

Table 2.4: Results of the KS two sample test for stairs down data, comparing the actual and the predicted signal for each coordinate and user.

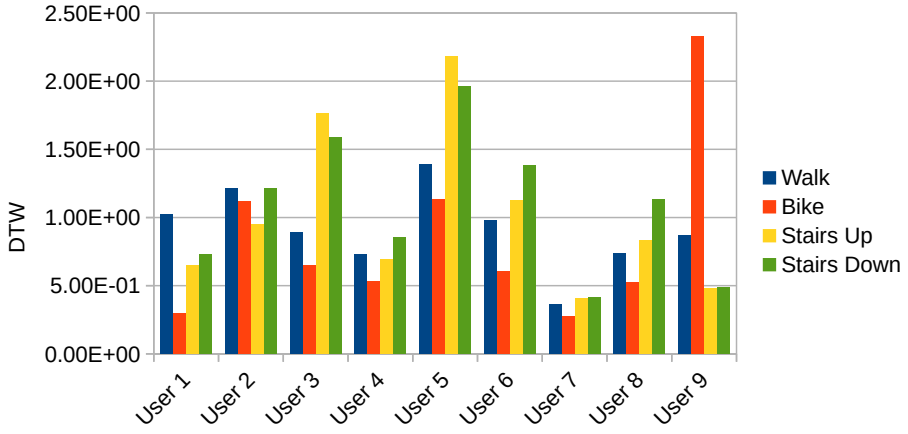
User	**Statistic**			**P-value**		
	x	**y**	**z**	**x**	**y**	**z**
1	4.2E-1	5.8E-1	3.7E-1	4.4E-13	2.7E-25	7.8E-11
2	7.3E-1	3.5E-1	7.0E-1	9.7E-40	3.8E-8	2.5E-40
3	3.9E-1	7.6E-1	8.5E-1	1.3E-12	3.2E-52	1.2E-63
4	4.4E-1	3.7E-1	3.15-1	2.9E-12	4.7E-9	1.2E-6
5	3.9E-1	8.4E-1	7.4E-1	2.4E-12	1.8E-57	1.5E-40
6	5.1E-1	6.1E-1	7.9E-1	2.4E-19	4.7E-32	7.6E-52
7	6.3E-1	7.9E-1	4.6E-1	8.0E-33	6.7E-53	1.8E-16
8	5.5E-1	3.0E-1	3.9E-1	7.0E-25	6.3E-7	1.9E-13
9	8.1E-1	3.5E-1	2.6E-1	1.5E-51	5.0E-11	4.0E-4
Avg	5.4E-1	5.5E-1	5.4E-1	7.9E-13	7.5E-8	4.5E-5



(a) X signal vector



(b) Y signal vector



(c) Z signal vector

Figure 2.3: Comparison of DTW distance between actual and predicted signals for 200 samples for all users during different activities. The predictions are made based on 50 previous samples of actual and prediction data mixed as discussed in Section 2.4.

The model was trained using the rest of the sessions' data (consisting of all the activities) that has on average 200000 samples that are captured during 33 minutes. Another session's data of the same size is used for the model validation. We use different LSTM configurations to fit our model and present the results obtained using an optimal configuration of the LSTM model having 50 neurons at one hidden layer, trained with 64 vector samples per batch during 100 epochs. We decided to choose this setting after trying different configurations and finding out that our model performs the best with our dataset using this configuration.

For this experiment, we train our model with $\Omega = 50$ to predict one sample based on 50 previously seen samples. Furthermore, we extend our prediction to 200 samples in the future using the method we discussed in Section 2.3.2. So, since the data has been captured at 100Hz, we look at the past 0.5 seconds and predict the future 2 seconds which is sufficient for most applications. We use a different session having an average of 100000 samples per user indicating 16 minutes of captured activities as the test data. Each testing vector consists of 50 samples as input and one as output. With every 50 samples as input, to predict more than one future sample, we use already predicted samples with previously available ones by moving the window of 50 input samples. We use all the possible vectors of actual data and predicted data to generate all the possible prediction vectors as shown in Eq. 2.1. For generating the prediction signals, we use two windows. The first window moves on 200 samples of actual data and the second window moves on 100 samples of actual data followed by predicted data that is generated during each prediction. In each movement of the windows, we discard the oldest sample. The first window only accepts actual data while the second window starts with actual data and will discard actual data and accept predicted data during each prediction. Here is the algorithm for this scenario:

1. Add 50 samples from actual data to Window1

2. Let Window2=Window1
3. Predict a new sample based on Window2
4. Remove the first (oldest) sample from Window2
5. Append the new predicted sample to Window2
6. Repeat step 3 to 5 for 100 times
7. Remove the first (oldest) sample from Window1
8. Append next sample from actual data to Window1
9. Repeat steps 2 to 8 for 200 times

The above algorithm results in vectors of predicted values based on actual data and a mix of actual data and predicted data. We use Eq. 2.2 to calculate the average for each sample. Setting $\alpha_1 = 1$ and all other coefficients to zero results in the case that all the predictions have been made based on all actual data is far from the real-world scenario. So, we assume the predicted samples from all actual data are more accurate, if available, but we also give weights to other predicted samples which are not completely made based on actual data. Hence, we set $\alpha_1 = n, \alpha_2 = n-1, \alpha_3 = n-2, ..., \alpha_n = 1$ paying more attention to the predicted samples made based on actual data.

The final result of the test will be a vector of 200 predicted samples. We use the corresponding 200 samples from the actual signal to evaluate the performance of our predictor. To present a measure for the distance between actual and prediction signals, we use the Dynamic Time Warping (DTW) algorithm and calculate the distance between each x, y, z pairs of signals. The significant feature of DTW distance is that when comparing two signals distributed over time, it can ignore the differences that are caused by time. In our case, DTW can ignore gait speed and acceleration differences between two users. So, when comparing the signals using DTW, we can assume the users have performed the same amount of activity (e.g. steps for walking) in the period we are using for comparison (200 samples). We also perform a Kolmogorov-Smirnov

(KS) test on our data to examine the statistical significance of the signals.

2.4.1 DTW Analysis

We run the test for all 9 users with walking, biking, stairs up, and stairs down data with several batches of 200 continuous samples from the activity signals. Then, we calculate the DTW distance between the actual and predicted signal for each test and each coordinate. Finally, we calculate the average of each coordinate from the results for every user during each activity and present the results in Fig. 2.3.

The results show that the DTW falls in a maximum distance of 1.30 for x, 1.35 for y, and 2.30 for z while according to our data the maximum possible ranges for x, y, and z are 40, 30, and 30 respectively after removing the outliers. The z coordinate shows the highest average distance, while x and y are significantly lower. So, the model may be tuned to pay more attention to z.

Among different activities, the model has been most successful in predicting the biking signal. Among different users, user 5 shows the highest average distance in all coordinates.

2.4.2 Statistical Analysis of the Predicted Signal

We use KS two-sample test to compare the actual and predicted signals and estimate how the signals follow each other. For each test, we calculate the test statistic value (D) and P-value. For 200 samples at a 5% significance level, the critical value D_α is 0.136. The null hypothesis says that the actual and predicted signals are from the same distribution. Our test results as shown in Tables 2.1-2.4 indicate that 99% of statistic values are over the critical value. Also, the P-values are significantly small. So, the

test results reject the null hypothesis. The test with a 1% significance level leads to the same conclusion. However, we cannot rely on the outcome of the KS two-sample test for our experiment, because it calculates the maximum difference between the cumulative distribution function (CDF) of the actual and the predicted signal, and CDF is not reflective of the time sequence pattern due to the variations in gait speed.

2.5 Summary

We proposed a model for predicting future activity signal for accelerometer data captured from a smartphone during different activities and executed multiple experiments to evaluate our model. Our results show that our model is capable of generating the future movement signal with significant accuracy.

Following our study on gait biometrics, we are also interested to explore another category of behavioral biometrics related to human computer interaction. So, in the next chapter, we study user interactions with input devices to a computer using keystroke sounds. Furthermore, we study the security applications of behavioral biometrics and build a model to examine the threats against a computer system using keystrokes audio.

Chapter 3

Adversarial Activity Detection Using Keystroke Acoustics

3.1 Introduction

Keystroke acoustics can be used to detect adversarial activities and threats against a system by monitoring users' behavior without interrupting their interaction with the computer. There is no need to install specific software or hardware, such as a keylogger on the users' machines to monitor their activity. Therefore, the typing activity of a malicious attacker may be monitored covertly without the attacker's awareness. So, it is less likely that the attacker takes preventive measures to evade detection. In Section 3.4.2, we discuss how our method compares with installing a simple keylogger on the users' machines which is one of the traditional monitoring methods.

In this research, we propose a model consisting of multiple components that gets the audio data of the users' typing sessions as input and detects the type of their activity and the threat level against the system as output. As depicted in Fig. 3.1, we use audio

Table 3.1: Comparison of the prior work on text recovery using keystrokes audio with our work.

Research	**Data Quality**	**Data Type**	**Data Collection Method**	**Recovery Method**	**Mics**
Asonov & Agrawal [26]	Clean & Noisy	Random	Fixed	Keys	2
Zhuang et al. [27]	Moderate Noise	Benign	Fixed	Words	1
Berger et al. [28]	Clean	Benign	Fixed	Words	1
Liu et al. [29]	Moderate Noise	Random	Random	Keys	2
Roth et al. [30, 31]	Clean & Noisy	Benign	Free & Fixed	N/A	1
Campagno et al. [32]	Clean & Noisy	Random	Fixed	Keys	1
This Work	**Very Noisy**	**Adversarial & Benign**	**Free**	**Words**	2

signals collected using multiple recording devices to recover the keystrokes and typed words of the users. Then, we develop an LSTM-based neural network model to identify the user's activity type based on the typed words. Finally, we use the user's activity and its preassigned threat level to generate a threat score for the session.

3.1.1 Related Work

Prior studies have proposed multiple approaches for recovering letters and words using audio recordings of typing sessions. In a classic approach to this problem, Asonov and Agrawal [26] propose a method to extract individual keystroke sounds from raw audio using amplitude threshold and signal peaks. Then, they extract FFT (Fast Fourier Transform) coefficients from each keystroke as a feature vector and train a neural network to detect the keystrokes. Zhuang et al. [27] improve Asonov and Agrawal's work using an unsupervised approach instead of a labeled training set. They use Mel-frequency Cepstral Coefficients (MFCC) instead of FFT coefficients. Berger et al. [28] further improve the previous work by proposing a dictionary attack to extract words based on the keystroke audio recordings and achieve 73% overall accuracy for recovering the words. However, their approach is not applicable for recovering random text

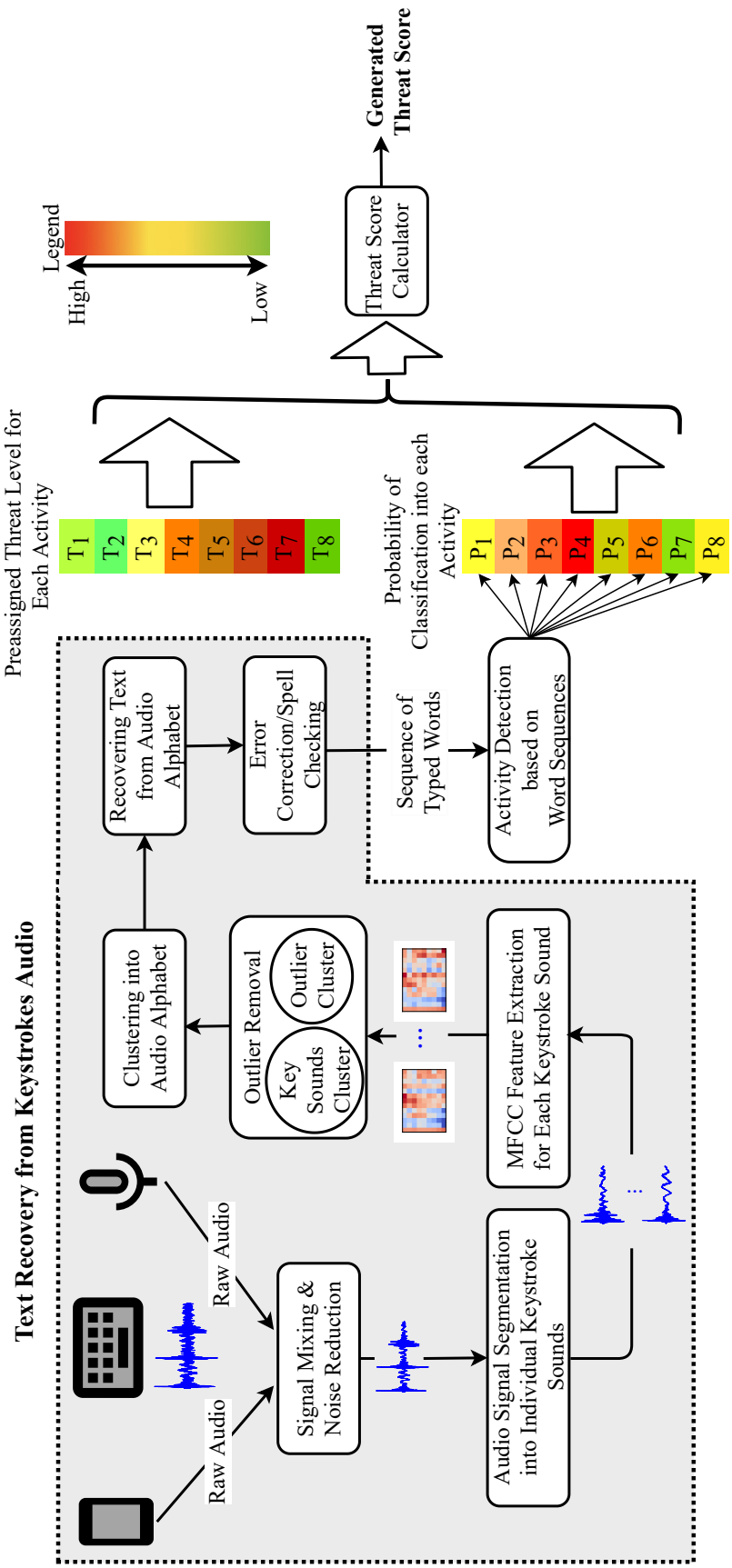


Figure 3.1: Overview of our proposed components from recording the typing session audio to quantifying the threat level.

like passwords. In another effort, Liu et al. [29] combine the time of arrival and acoustic features of keystrokes to extract individual keystrokes from audio recordings with a mobile phone. They recover 97% of the text without training or depending on linguistic context. Campagno et al. [32] propose Skype & Type which focuses on recovering the keystrokes using audio transmitted over VoIP applications. They use a peak detection method for individual keystroke sound extraction and MFCC features for classification. In another noteworthy work, Wang et al. [33] study the detection of combined keystrokes using acoustic signals.

Another application that has been studied by the prior work is attack detection and classification using keystrokes. Belman et al. [34] extract several features from keystroke dynamics of benign and malicious users and train conventional classifiers for adversarial activity detection. However, their method assumes that they have access to keystroke dynamics, including the exact pressed keys and the timing of keystrokes. In our method, we use audio recordings instead of keystroke dynamics and pay attention to the context of the typed text.

Furthermore, in another series of studies, Roth et al. [30, 31] use keystroke audio emanations for user identification. They use a threshold-based method and MFCC features for building an authentication model based on keystroke acoustics.

3.2 The Significance of Our Work

Although prior studies have addressed the use of keystroke acoustics for various applications, none have used them for malicious activity and threat-level detection to the extent of our knowledge. As summarized in Table 3.1, this study stands out from prior work and contributes to the research in multiple aspects:

Table 3.2: A comparison between the characteristics of the attack and the benign dataset. The numbers show the averages for the entire dataset.

	Attack	**Benign**
Duration	4884.62s	2915.32s
Enters	160.21	68.57
Enters/hour	119.5	85.44
Letters/Sentence	11.18	65.28
Words/Sentence	2.65	13.1
Words in Dict	21.46%	31.59%

- We present a model for text recovery from raw audio data of users' typing sessions, especially suited to noisy environments, such as where multiple people are present in a room (similar to a real-world office environment). Our results on a particularly challenging dataset with multiple inaudible keystrokes and a noisy environment show that our text recovery model is able to recover the typed words with an average 71% top-10 accuracy.

- We show that it is possible to classify a sequence of typed words which are recovered from audio into 8 different activities with up to 98.07% accuracy by proposing an LSTM-based model which is unique for our application. While the number of activities is limited to our dataset, our work is easily extendable to other activities.

- Our large dataset of nearly two million keystrokes is collected during both adversarial and benign sessions from 103 and 117 users respectively. The users have interacted with the keyboard freely, without having to type certain fixed words several times in contrast to most of the prior studies.

- Although other studies have collected and used data from noisy environments, as discussed in Section 3.5.1, the amount of noise in our data is significantly high, making the text recovery task very challenging.

3.3 Typing Differences in Adversarial and Benign Environments

User interaction with the input devices varies during different activities. In benign activities like web browsing, email writing, etc., the usage of specific keys and the overall usage of the keyboard are different. For example, the use of the enter key is more frequent in adversarial activities because the commands are separated by the enter key and are shorter than sentences typed in a benign session. Also, the use of special characters is more frequent in adversarial activities and dictionary words appear more in benign activities. Table 3.2 shows a few statistical differences between the two datasets. The average number of enters per hour is higher for the attack sessions which can show higher terminal usage with shorter commands. The average numbers of letters and words are higher for the benign activity showing longer words and sentences. Also, a higher percentage of the words typed during the benign session appear in an English dictionary. Therefore, several features can distinguish between the benign and adversarial activities as studied in [34]. However, these features may misclassify benign terminal sessions with a lot of safe commands, resulting in a high false-positive rate. Also, they cannot distinguish between different types of adversarial activities.

Benign and adversarial environments are also different in terms of the typed words and commands. Some of the commands typed in a terminal are more dangerous in nature. For example, "sudo" is used to gain administrative privileges and is more common in malicious activities than daily tasks. Similarly, the commands for creating filesystems, installing applications, etc. can be more dangerous than a simple execution of "ls" to list the files in a directory.

3.3.1 How These Differences Are Useful in Our Application?

The presence of specific words which are unique to one environment and are absent from another, helps us train a neural network model to distinguish between the adversarial and benign activities as discussed in Section 3.7. Since a single malicious command may not be sufficient for evaluating the threat level of the entire typing session, we assign a threat level to each activity instead of each command. Our model learns to classify a sequence of words into an activity. Then, we use the threat level of the activity to detect the maliciousness level of the entire typing session (Section 3.7.2).

3.4 Threat Model and Adversarial Capabilities

In our application, we assume that we have already identified the user and their input devices. The user can be identified using methods based on behavioral biometrics or through classic user name/password authentication. So, we are aware of the user access privileges in the system and assume the adversary has limited access to a system, network, or sensitive data for a legitimate purpose and can use a Linux terminal to exploit the system and a web browser to search for attack instructions and downloading malware. However, we do not make any assumptions about the presence of any vulnerabilities in the system. The adversary works with a mouse and a keyboard and can recognize if any eavesdropping or surveillance software or hardware have been directly installed on the machine. The only information captured is the environment sound.

3.4.1 Real World Attack Scenario

This work addresses any attack that can be done in public spaces with a keyboard on a computer. A routine scenario can be an employee working with a computer in

an office or a college student using a public computer available on campus. After authentication, the user will have limited access to the computer and the network. The type of the keyboard is fixed for the specific public computer and the user is identified through authentication. A voice recorder has been set up in the room which records the environment audio and broadcasts it to a server that runs our model for threat level evaluation.

3.4.2 Using Keystroke Acoustics Versus Keylogger

A common eavesdropping method to detect potentially harmful behavior is to install software or hardware keyloggers on the users' machines. The keylogger reads all the keystrokes and broadcasts them to a server for processing. However, audio recording can be done covertly and from distance without physical access to the computer and the user's awareness. So, it is more practical in the real-world scenario.

3.5 Description of Benign and Adversarial Datasets

Following Institutional Review Board (IRB) approval, two separate datasets have been collected in our lab. In both, several modalities have been collected from a variety of participants. The SU-AIS BBMAS [35] dataset is collected from the participants during their everyday routines which we consider as benign activity. The second dataset is collected from participants while performing predefined malicious tasks on a computer to gain access to a remote machine and extract sensitive information. This dataset will be published and made publicly available in IEEE Dataport by our lab. In both datasets, the user has used a standard computer QWERTY keyboard (Dell KB212-B). Two devices have been used for recording the audio: a Blue Yeti Professional Wired

Table 3.3: Description of activities performed by the users in the adversarial and benign datasets, example command that may be entered by the user, and their threat level. Example commands only show one method of doing a specific task and the participants were asked to use their own method for fulfilling the task.

Welcome Activity	**Description**	Opening a terminal on a Linux machine.
	Example	Use GUI, CTRL+T, gnome-terminal, xterm
	Threat Level	Low
Network Discovery	**Description**	Identifying IP addresses and open ports.
	Example	nmap -sV 192.0.1.1/24
	Threat Level	Medium
Target Identification	**Description**	Identifying an attack to access a vulnerable machine.
	Example	ssh 192.0.1.3
	Threat Level	Medium
Password Dictionary Attack	**Description**	Cracking a user/pass of the remote machine.
	Example	hydra –L user.txt –P pass.txt 192.0.1.3 ssh
	Threat Level	High
Privilege Escalation	**Description**	Obtaining root access to the remote machine.
	Example	sudo, chown, chmod, scp
	Threat Level	High
Data Exfiltration	**Description**	Downloading sensitive files from the remote machine.
	Example	chown, chmod, scp
	Threat Level	High
Credential Stealing	**Description**	Accessing the remote machine by stealing credentials.
	Example	ssh -i id_rsa jesse@192.0.1.12
	Threat Level	High
Benign	**Description**	Daily tasks like online shopping, note-taking, etc.
	Example	Arbitrary benign command/text
	Threat Level	Low/Benign

Table 3.4: The number of audible, inaudible, and noisy keystroke sounds out of 100 randomly selected ones for 5 randomly selected users from the dataset as discussed in Section 3.5.1.

User	**Audible**	**Inaudible**	**Noisy**
1	39	57	4
2	50	43	7
3	56	42	2
4	71	25	4
5	65	32	3

Microphone which was placed 1 inch away from the top of the keyboard and a Samsung Galaxy S6 smartphone which was placed 1 foot to the left of the keyboard.

The benign data is collected from 117 participants during regular daily tasks like text transcription, online shopping, note-taking, and typing the answers to a set of questions. Most of the participants were students with computer science or related majors with ages of 19 to 35 years. Among the participants 72 were male and 45 were female. The adversarial dataset has been collected from 103 students in the age range of 19-46 years among which 22 were female and 81 were male, mostly with computer science background during a sequence of activities that result in discovering a vulnerability in a remote machine, getting access to it, and extracting information from it. The tasks are defined in a way to reproduce a regular procedure for a common form of attack in a network. The users were asked to do 7 different activities to eventually succeed in the attack. They were given a Linux machine with a web browser to search for information and tutorials to aid them to fulfill the requirements of the tasks. The users were free to use their method for completing the tasks and were not asked to follow a fixed set of instructions or commands. Among the available data, we use keystrokes data and audio data which is recorded using a standalone microphone and a mobile phone in this work. The activities and their threat level against the system which are manually set by us are described in detail in Table 3.3.

3.5.1 Noisiness of the Data

Our data is collected in a lab room with the presence of other researchers and an instructor directing the participant to do the tasks and their voice is also recorded in the background. Based on the typing behavior of the user, some keys may be pressed gently without making any audible sound to be recorded. Furthermore, our data is collected using simple and cheap equipment like mobile phones. Therefore, several keystroke sounds are inaudible or overlap with human voice and background noise. To evaluate the quality of the data, we use the exact keystrokes times to extract their sound from the session audio. Then, we listen to 100 random keystroke sounds from 5 users manually to see if they are audible by the human ear. We refer to a keystroke sound as inaudible if nothing can be heard from the audio and refer to it as noisy if some noise from the environment such as human voice talking in the background overlaps with the keystroke sound in a way it cannot be heard and recognized by the human ear. We show the statistics for this experiment in Table 3.4.

3.6 Text Extraction from Audio

As shown in Fig. 3.1, to recover the typed text from audio, first we preprocess the raw audio signal to improve the quality and reduce the noise. Then, we extract individual keystroke sounds from it. Next, we generate features for each keystroke audio signal, remove the outliers, and cluster them into a set of audio letters. Finally, we present a mathematical model to recover the words using the audio alphabet and use a spell-checking method to fix the errors in the recovered words. In this section, we go over each component in detail. Since the audio recordings from the benign dataset are not available, we only use the attack dataset for extracting text from audio.

3.6.1 Audio Signal Preprocessing by Signal Mixing and Noise Reduction

We process and clean the audio file by mixing audio data from two separate recording devices to improve the signal quality. In summary, these are the steps taken to preprocess the raw audio:

- Matching and time synchronizing the recordings from the microphone and the smartphone. Because the exact start times of the recordings are available in the dataset and both tracks have been recorded with equal sample rates, there is no need for up-sampling or down-sampling and the effect of distance from the keyboard is diminished.
- Mixing both signals into one stereo audio track using Sound eXchange (SoX) mixing tool [36].
- Obtaining a noise profile to reduce the background noise such as human voice, fan, door sound, etc.
- Using a generic spectral noise gating algorithm with the noise profile to reduce the background noise in the audio.
- Applying voice reduction and isolation filter using Audacity [37] tool to remove the human voices.
- Normalizing the amplitude using two passes of ffmpeg loudnorm filter with EBU R 128 [38] algorithm.

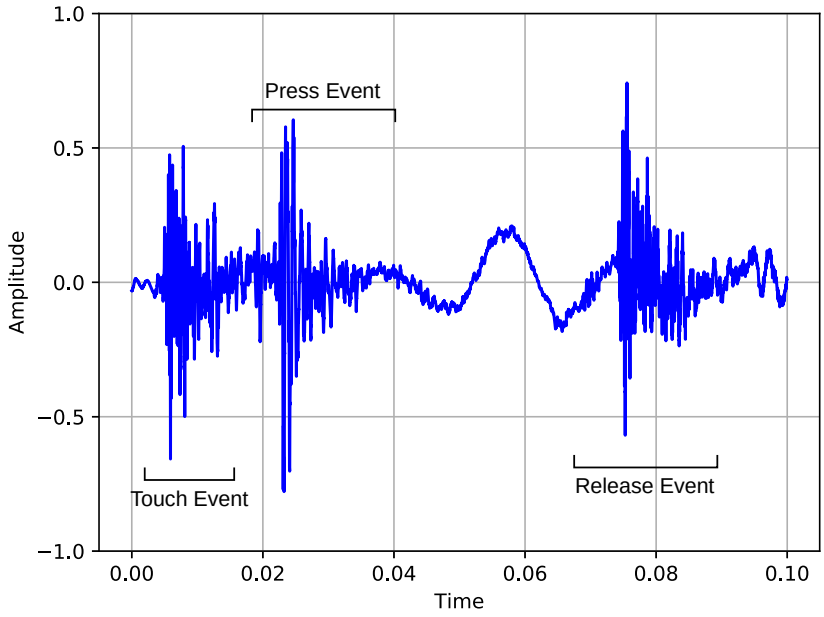


Figure 3.2: Example of an audio signal for a single keystroke. Each keystroke event consists of a touch, a press, and a release event. The horizontal axis shows time in seconds and the vertical axis shows signal amplitude which is scaled between -1 and 1.

3.6.2 Audio Signal Segmentation Into Individual Keystroke Sounds

As shown in Fig. 3.2, each keystroke event is composed of three stages: touch, press, and release [27]. Since the most distinguishable and significant peak of the signal is the press event, we use it for detecting the keystroke signal. We use a similar approach to prior research [29, 27, 30, 31] with different parameters and modifications to fit our data and application. First, we calculate FFT coefficients for small sliding windows of 10ms. With a 44100Hz signal sample rate, we slide the window in steps of 0.02ms. Then, we get the sum of FFT coefficients for each window as the signal energy and calculate the 99% percentile of the resulting sum values from all the windows. Finally, we accept the windows greater than the 99% percentile as peaks. As a result, each keystroke sound will be detected multiple times. To avoid this, we use a threshold of 113ms which is obtained through experiments as the minimum delay between two

press events. We take the past 10ms and the future 90ms of the peak from the signal to include the touch event, assuming each keystroke event takes 100ms. The 100ms keystroke delay has shown to be sufficient for our purpose in prior research [26, 27, 29].

To further improve the accuracy of our method, we use an amplitude and a frequency filter to accept the detected keystroke sounds. Based on experiments, a minimum cutoff of 12000 for amplitude and a maximum cutoff of 150Hz for frequency achieve the best results.

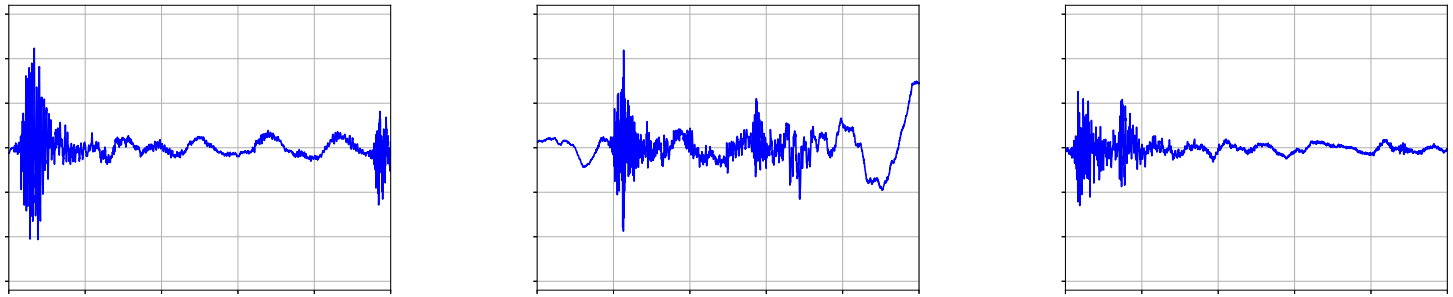
This process is only used with the testing data because the the ground truth (exact keystroke timings) for all the keystrokes is available in the dataset. Thus, instead of using this process to extract the individual keystroke sounds, we directly use the timings to extract them from the preprocessed audio and extend them to 100ms.

3.6.3 Using MFCCs as Audio Features

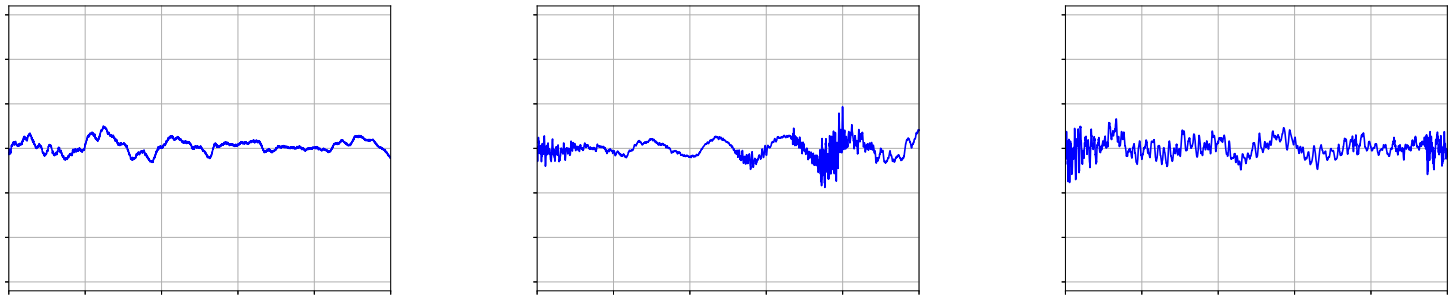
MFCCs have been frequently used as features in speech recognition applications [39]. Research has shown that they can also be effective for recovering keystrokes using their sound [30, 31, 33, 29, 27]. Thus, we generate MFCCs as features for each individual keystroke sound. We use a window length of 10ms and 2.5ms step between windows, 32 filters in the filter bank, and 16 coefficients with a frequency range of 400-14000Hz. The shape of the MFCC feature set will be 30x13 for each keystroke sound which we flatten to get a linear feature vector of length 390.

3.6.4 Removing Inaudible and Noisy Keystrokes

As mentioned in Section 3.5.1, there are several keystrokes that have not made any audible sound and are not recorded. We show the signals for a few samples randomly



(a) Valid Keystroke Signals



(b) Inaudible Keystroke Signals (Outliers)

Figure 3.3: Randomly selected samples from the keystroke audio signals showing correctly recorded and inaudible keystrokes which we consider as outliers and discard from our training data. In contrast with Fig. 3.2, the signals in this figure do not include the touch event segment of the signal, because the signals have been extracted using the exact keystroke times. So, the start of the signal is the press event. The horizontal axis shows time in seconds and the vertical axis shows signal amplitude which is scaled between -1 and 1.

selected from keystroke sounds in Fig. 3.3. The signals that look similar to the one we have shown in Fig. 3.2 are usually recorded correctly and the signals that are not in the same shape are from the keypresses which have been pressed very gently and have not made any audible sound to be recorded by the devices. So, we need to remove them from our training set as outliers. To do this, after generating the feature vectors, we use the K-Means clustering algorithm [40] to split keystroke sounds into two clusters. As a result, one cluster includes all the correctly recorded keystroke sounds and the other cluster includes noisy or inaudible ones. For each cluster, we calculate the mean of the absolute amplitude values. The cluster with the lower mean contains the outliers which we remove from the data.

3.6.5 Clustering Keystroke Sounds Into Audio Alphabet

The sound a keystroke makes is not always unique. So, similar to [30, 31], we define an audio alphabet containing a set of audio letters. Our audio alphabet size is larger than the number of keys, allowing each key to be assigned to different audio letters. We use the Gaussian Mixture clustering algorithm [41] (implemented in Python sklearn.mixture.GaussianMixture package) to assign keystroke sounds into 104 different clusters (the number of clusters is chosen through experiments reported in Section 3.8.3). We only use alphanumeric, special character, space, and enter keys (total 52 keys) for generating the clusters using training data. For each user, the training data consists of individual keystroke sounds extracted from the audio recordings using the ground truth timings and values available in the dataset.

Table 3.5: (a) Example of 22 keystroke sounds clustered into 8 audio letters. (b) The value of h'_{kc_j} probability for each letter. (c) Assigned keys to each cluster during the training session, sorted by h'_{kc_j} probability.
The values for the clusters which are not among the example query discussed in Section 3.6.6 are not shown in (b) and (c).

(a)

Cluster	Keys
c_1	d b d d
c_2	e b b e
c_3	e a a e c
c_4	b a a
c_5	a c
c_6	c c
c_7	a
c_8	d

(b)

Key (k)	Probability		
	h'_{kc_1}	h'_{kc_2}	h'_{kc_3}
a	0	0	2/5
b	1/4	1/2	0
c	0	0	1/5
d	3/4	0	0
e	0	1/2	2/5

(c)

Cluster	Keys
c_1	d b
c_2	b e
c_3	a e c

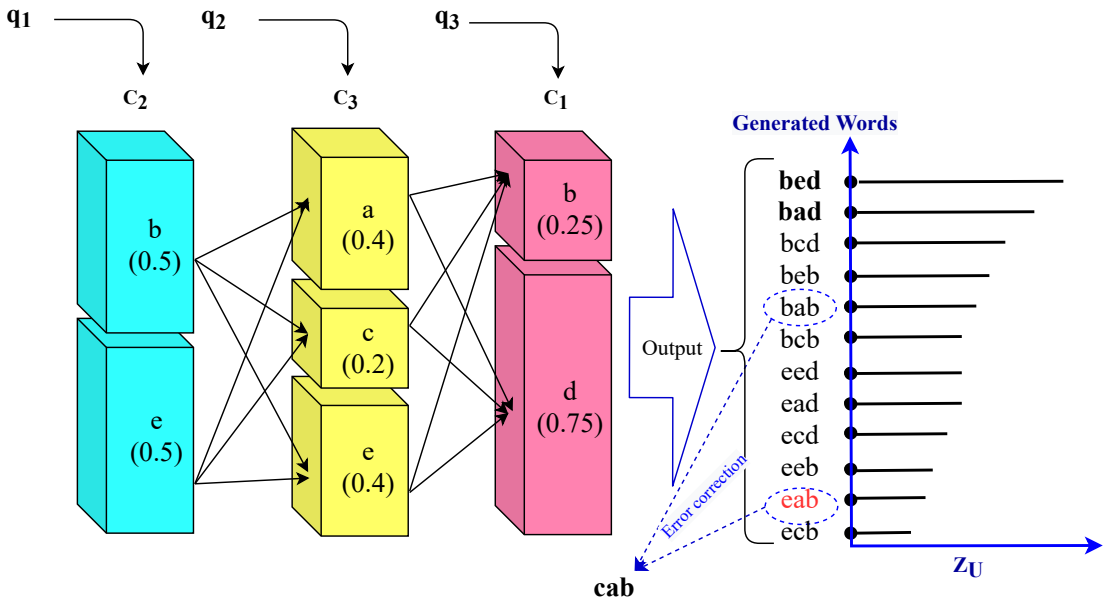


Figure 3.4: Generating words for the query $Q = \{q_1, q_2, q_3\}$ clustered into $Y = \{c_2, c_3, c_1\}$ where each c_i is a letter from the audio alphabet (clusters). All the possible permutations from the clusters are generated and then sorted by Z_U which shows the correctness probability of the word. "bad" and "bed" are already correct English words. "bab" and "eab" will be corrected to "cab" by the error correction module.

3.6.6 Text Recovery Using Audio Alphabet

We build a mathematical model based on the probabilities that we have to recover the typed words. Let $T = \{t_1, t_2, ..., t_n\}$ be the set of all the training samples, the probability that keystroke sound t_i being assigned to cluster c_j calculated by the Gaussian Mixture clustering method be $g_{c_j t_i}$, the total number of training samples assigned to cluster c_j be N_j, and $A = \{a_1, a_2, ..., a_n\}$ be the set of all keystroke sounds (t_i's) that represent key k. We calculate h_{kc_j} which is the probability of key k in cluster c_j:

$$h_{kc_j} = p(k|c_j) = \frac{\sum_{i=1}^{n} g_{c_j a_i}}{N_j} \tag{3.1}$$

For each a_i representing k, Gaussian Mixture clustering method generates a vector $v_{a_i} = \{g_{c_1 a_i}, g_{c_2 a_i}, ..., g_{c_j a_i}\}$. We take the maximum $g_{c_j a_i}$ for each a_i, set it to 1, and set the rest to zero. So, we assign each keystroke sound to only one cluster:

$$J = \{j | g_{c_j a_i} = max(v_{a_i})\} \tag{3.2}$$

$$G_{c_j a_i} = \begin{cases} 0 & j \neq J \\ 1 & j = J \end{cases} \tag{3.3}$$

Then, we define h'_{kc_j} and H_{kc_j} probabilities:

$$N'_j = \sum_{i=1}^{n} G_{c_j t_i} \tag{3.4}$$

$$h'_{kc_j} = \frac{\sum_{i=1}^{n} G_{c_j a_i}}{N'_j} \tag{3.5}$$

$$H_{kc_j} = max(h_{kc_j}, h'_{kc_j}) \tag{3.6}$$

We sort the keys in each cluster by H_{kc_j}. So, the keys in cluster c_j will be $K_{c_j} = \{k_1, k_2, ..., k_{52}\}$ where $H_{k_1c_j} \geq H_{k_2c_j} \geq ... \geq H_{k_{52}c_j}$. Note that samples representing some keys will never be assigned to some of the clusters and the majority of h probabilities will be zero resulting in $h = h'$ for most of the samples and clusters.

Now, if we have a query of keystroke sounds $Q = \{q_1, q_2, ..., q_m\}$ where each q_i is a keystroke sound sample from a testing set, we use the Gaussian Mixture clustering method with the already trained model to classify each q_i to the clusters. Then, we only take the cluster c_j with the highest $H_{c_jq_i}$ for each q_i. As a result, we assign each q_i to only one cluster. So we have a set of clusters $Y = \{y_1, y_2, ..., y_m\}$ where $q_1 \in y_1, q_2 \in y_2, ..., q_m \in y_m$. Now, we generate all the permutations of keys from the clusters, taking one key from each cluster, starting with the key with the highest H_{kc_j} in each cluster c_j.

We use a dictionary of English words [42] combined with a list of Linux commands and common words typed in terminals [43]. If we have k_i representing letter l_i, we define $d_{k_i} = p(l_i|l_{i-1})$. If the number of times l_i and l_{i-1} appear sequentially in all the words in the dictionary is j and the number of all occurrences of l_i in the dictionary is L, we calculate $d_{k_i} = j/L$.

Let $U = k_1, k_2, ..., k_m$ be a permutation of keys (word) where k_i is selected from c_i. We define:

$$Z_U = \prod_{i=1}^{m} \frac{\alpha d_{k_i} + \beta H_{k_ic_i}}{\alpha + \beta} \tag{3.7}$$

Z_U represents the correctness probability of a permutation retrieved from a query of clusters based on the probability of the letters in the dictionary (d) and the probability of each key in each cluster (H). α and β are factors for changing the weight of d and H and are manually set through experiments (Section 3.8.3). We calculate Z_U for each permutation and sort them based on Z_U. As a result, our model generates words that

are recovered from the query of keystroke sounds sorted by their correctness probability.

A Simplified Example

We show an example of the training and testing procedure with a limited number of samples. To minimize the number of possible permutations generated from a query of clusters, we assume that each training sample is only assigned to one cluster, so $H_{kc_j} = h'_{kc_j}$. Let $T = \{t_1, ..., t_{22}\}$ and $T' = \{l_1, ..., l_{22}\}$ where T contains all the keystroke sounds in the training set and T' is the set of letters represented by the keystroke sounds in T and l_n is the key represented by t_n. In our example, we set $T' = \{b, d, d, d, e, b, a, e, c, b, c, a, b, a, e, a, a, c, e, c, a, d\}$ and assume there are 8 clusters and each sample has been assigned to one of the clusters as shown in Table 3.5a. Also, we have query $Q = \{q_1, q_2, q_3\}$ clustered into $Y = \{c_2, c_3, c_1\}$. Then, we calculate h'_{kc_j} for all keys in all clusters which shows the probability of key k, if cluster c_j is selected. We show the calculated h' values in Table 3.5b for c_1, c_2, c_3 because these are the only clusters that have appeared in our sample query. By sorting the keys in each cluster based on h', we get the clusters in Table 3.5c. Then, we generate all the permutations of keys as shown in Fig. 3.4. Next, we calculate Z_U based on d_k and h'. For simplicity, we just illustrate the Z_U values for each permutation in Fig. 3.4 as the calculations are redundant and straightforward.

3.6.7 Error Correction using Dictionary

Having a set of words sorted by Z_U, we still recover words that are not completely correct or not in the dictionary. So, after generating the possible words, we use a spell-checking algorithm [44] with the same dictionaries used in the text recovery module [42, 43]. We filter out the words with numbers and special characters using regular

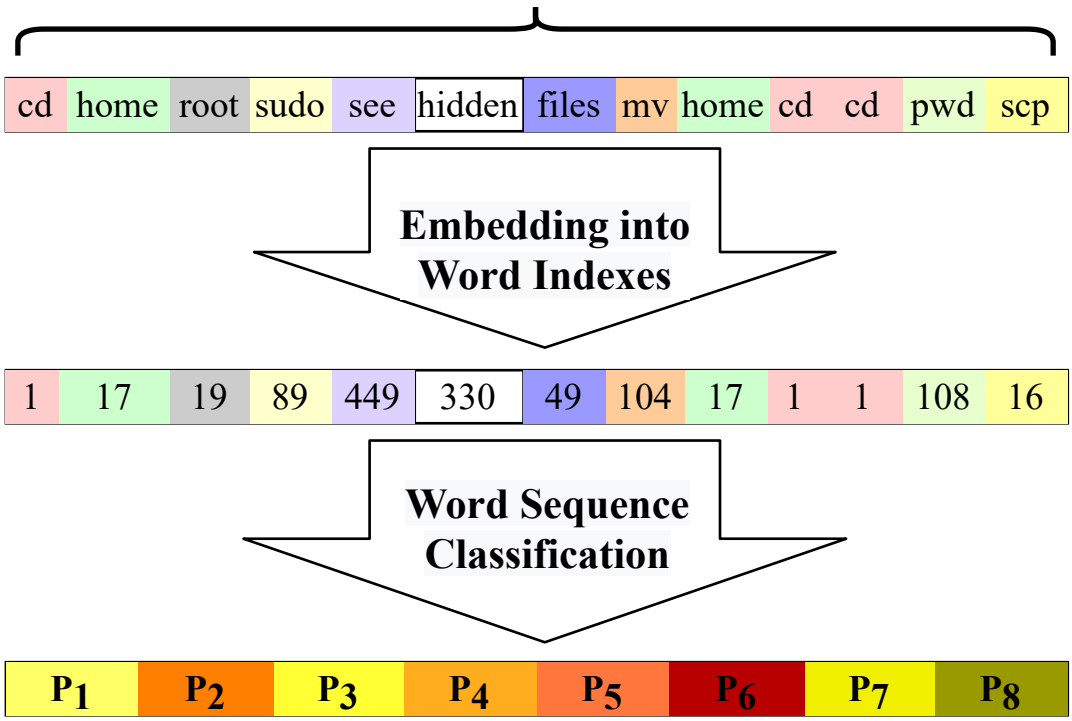


Figure 3.5: An example of a sequence of words typed by one of the users during activity 6. Red shows higher probability while green indicates lower probability. (See Section 3.7.1)

expressions and do not pass them to this module.

3.7 Adversarial Activity Detection and Classification

Among the different prominent architectures of Artificial Neural Networks (ANNs) that have been implemented as software and hardware components [45], recurrent neural networks are the preferred methods of learning from temporally correlated datasets. Long Short-Term Memory (LSTM) is a variant of RNN that has shown notable performance in the classification of sequential data. Hence, we propose an LSTM-based model to classify each sequence of words into one of the 8 classes. To build the training dataset, first, we get all the keystrokes from all the users and take each sequential set of keys

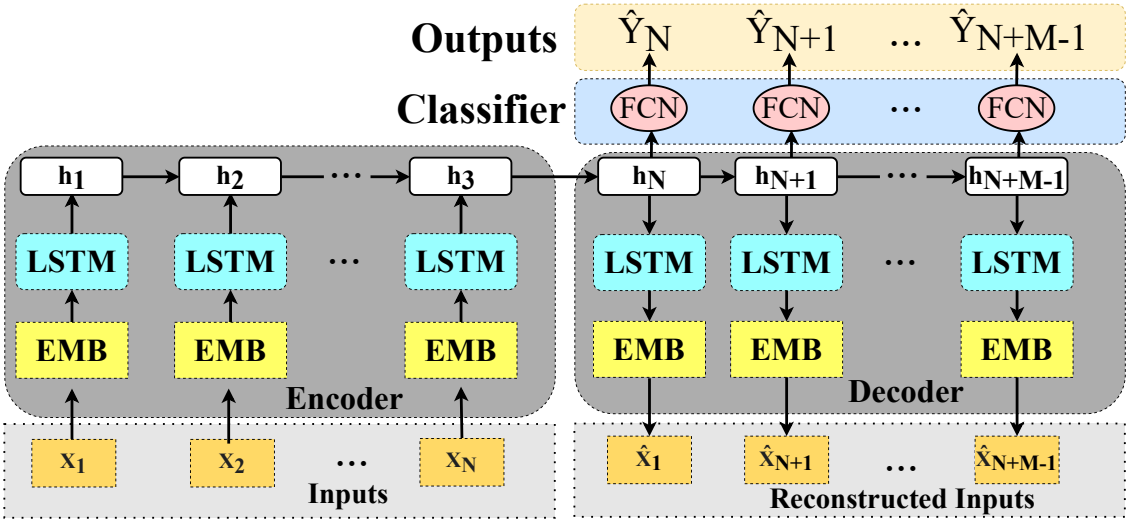


Figure 3.6: Data flow over N time-steps with inputs x_1, x_2, x_N (words) for the encoder/decoder LSTM architecture as discussed in Section 3.7.

(separated by space or enter) as a typed word. Then, we use every n sequential words for each activity of each user as one data sample. To feed the sequence into the network, we fix its length to n words. When n words are not available, we pad the sequence with zeros and when a sequence is longer than n words, we break it into segments of n words. As a result, we have fixed-length sequences of words and their corresponding activity to be used for training the model. Also, we get all the words from the dataset and sort them based on their frequency. Then, we assign an index to each word, with the most frequent word indexed with 1. The procedure for classifying a sample sequence of words has been demonstrated in Fig. 3.5.

Fig. 3.6 shows the flow of the data across three time steps. Similar to the previous work on time series data classification [46, 47], each input word x is fed to an embedding layer (EMB) to be represented in a lower-dimensional space. Then, the word embedding is passed to an LSTM layer that performs the main task of classification using Adam optimizer [25]. Pavllo et al. [48], show that regularizing the model by adding another network to form an encoder-decoder architecture that learns to reconstruct the input data increases the robustness and generalization capability of the LSTM layer and

helps the model to learn and represent the features of the data in a more effective way. So, we add such a component to our network. The latent state of the LSTM, which is represented by h_t in Fig. 3.6, is shared among the encoder and the decoder. For the classification task, h is passed through a fully connected network (FCN) with one hidden layer. During the training process, h is also used for reconstruction of the input data $\hat{x}_t$.

Eq. 3.8 shows the loss function that is used for training the model. L_R denotes the loss value obtained for the reconstruction of input in the output of the decoder and L_C represents the classification loss, which is calculated by comparing the expected output with the predicted class from the model. α and β are coefficients for controlling the effect of L_C and L_R on the total loss.

$$L_{total} = \alpha L_C(y, \hat{y}) + \beta L_R(x, \hat{x}) \tag{3.8}$$

3.7.1 Examples of Adversarial Activity Detection using Typed Sentences

We obtain a set of sequentially typed words from the attack dataset: {how, to, scan, open, ports, in, kali}

The set of words will be embedded into word indexes. In this example, we only consider the top 1000 words, so, the word "ports" is embedded to zero because it is not among the top 1000 words in the dataset: {121, 12, 720, 150, 0 ,10, 990}

Finally, the adversarial activity probability vector will be generated by the attack detection module: {0.04, 0.26, 0.15, 0.11, 0.09, 0.07, 0.09, 0.19}

As can be seen, the typed sentence will be classified to the network discovery ac-

tivity (Table 3.3) with the highest probability. Also, the benign activity probability is relatively high which is expected as the typed sentence is not necessarily threatening.

Similarly, we show an example of a sentence taken from the benign dataset. Here, we also show the sentence before error correction, so, the effect of error correction module can be seen: {slice, it, and, then, place, a, layer, of, tomato, on, top, of, the, laye, rof, onion} → {slice, it, and, then, place, a, layer, of, tomato, on, top, of, the, layer, of, onion} → {818, 17, 5, 21, 220, 4, 561, 14, 0, 16, 41, 14, 6, 561, 14, 0} → {0.06, 0.00, 0.00, 0.00, 0.04, 0.01, 0.00, 0.89}

3.7.2 Quantifying The Threat Level

The output of the activity classification network is a probability vector $v = \{p_1, p_2, ..., p_8\}$ indicating the probability that word sequence B belongs to each activity a_1 to a_8 respectively. We also define $D = \{d_1, d_2, ..., d_8\}$ where d_i indicates the threat level for each activity and is manually set. Since d_8 represents the benign activity, we set it to a small value ϵ. The rest of the d values are set on a scale of 0 to 1 based on our evaluation of the maliciousness of each activity as we have defined them where 0 shows a benign activity and 1 shows a highly malicious activity. Finally, we define X as a measure for the threat level of B:

$$X = d_1p_1 + d_2p_2 + ... + d_8p_8 \tag{3.9}$$

3.7.3 Continuous Real-Time Monitoring of the Threat Level

After calculating the threat level score in each time step, we use a separate scoring model as shown in Eq. 3.10 to continuously update the threat level in real-time. We

use a coefficient z^i which grows exponentially to give more attention to the latest seen events.

$$T = \frac{\sum_{i=1}^{n}(z^i \sum_{j=1}^{8} p_j d_j)}{\sum_{i=0}^{n} z^i} \tag{3.10}$$

Later, in our experiments in Section 3.9, we use $d = \{0.25, 0.5, 0.5, 1, 1, 1, 1, 0\}$ as the activities threat level vector by assigning 0 to the benign activity, 0.25 to low threat activities, 0.5 to medium threat activities, and 1 to high threat activities. As a result, with this d vector, the value of T in Eq. 3.10 will converge to 0.65 (Eq. 3.11).

$$\lim_{n\to\infty} T = \lim_{n\to\infty} \frac{\sum_{j=1}^{8} d_j}{8} = 0.65 \tag{3.11}$$

3.8 Performance Evaluation of the Proposed Components

In this section, we evaluate and analyze the performance of each module in our proposed method through several experiments.

3.8.1 Audio Signal Segmentation into Individual Keystroke Sounds

After preprocessing and outlier removal, we use the audio segmentation module to extract the keystroke times. For each keystroke time in the ground truth, we find the closest value from the audio segmentation module predictions. Then, we use an error threshold to consider the predicted time as correct or incorrect. With a threshold of 100ms, our audio segmentation module can detect 91% of the keystroke sounds on

average. The mean squared error averaged for all the users is 0.09. We calculate the mean squared error as follows when Y_i is the actual start time of the keystroke and $\hat{Y}_i$ is our predicted start time: $MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$.

The Impact of Using a Single Microphone

As mentioned in Section 3.6.1, the data has been recorded using microphones and we mix the recorded signals to improve the quality and reduce the noise. To find out whether the use of two separate microphones contribute to the performance of the audio segmentation module, we repeat the same experiments with the recordings from each of the microphones, without mixing the signals. Our results show that using only the stand-alone microphone, with a threshold of 100ms, the audio segmentation module detects 89% of the keystroke sounds on average with a mean squared error of 0.10. When using the mobile microphone only, 84% of the keystrokes are detected with a mean squared error of 0.10.

3.8.2 Clustering into Audio Alphabet

To evaluate the clustering method, we assume each training sample is assigned to only one cluster and define three metrics to calculate the accuracy. For all test samples $t_1, t_2, ..., t_n$ representing keys $k_1, k_2, ..., k_n$ clustered into $c_1, c_2, ..., c_n$ respectively, we calculate A_1 as an accuracy metric by considering a classification correct if at least one training sample that represents k_i has been assigned to c_i during the training procedure ($h'_{c_i k_i} > 0$).

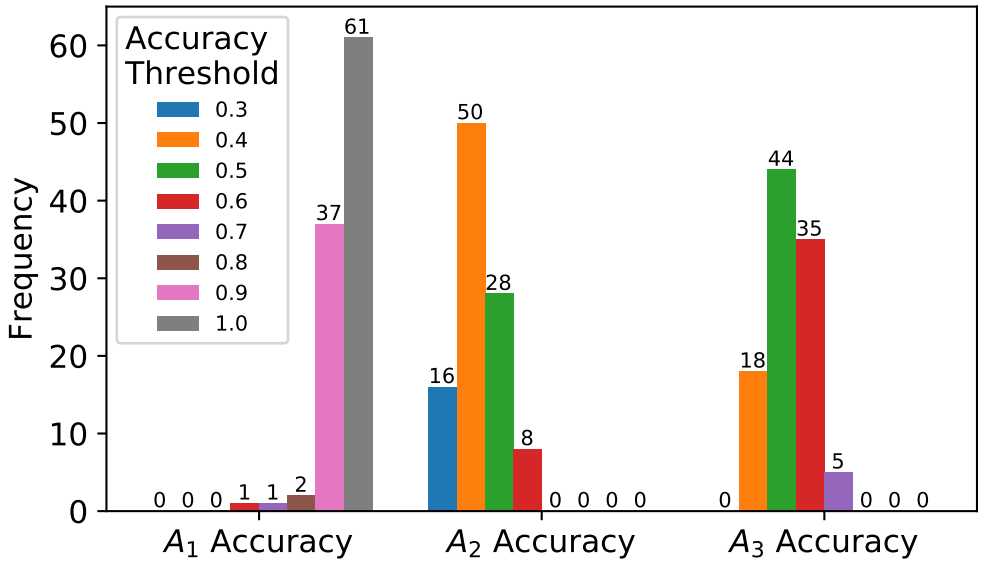


Figure 3.7: Distribution of the number of users based on A_1, A_2, A_3 accuracy metrics as discussed in Section 3.8.2.

$$A_1 = \frac{\sum_{i=1}^{n} \begin{cases} 0 & h'_{k_i c_i} = 0 \\ 1 & h'_{k_i c_i} \neq 0 \end{cases}}{n} \tag{3.12}$$

We also define A_2 as another accuracy metric which is the mean of $h'_{k_i c_i}$ for all key-cluster pairs:

$$A_2 = \frac{\sum_{i=1}^{n} h'_{k_i c_i}}{n} \tag{3.13}$$

We consider sample t_i representing key k_i clustered into c_i as correct if k_i has the highest probability $h'_{k_i c_i}$ among all the keys that have been clustered into c_i during the

training procedure. If N is the total number of the keys we calculate A_3:

$$A_3 = \frac{\sum_{i=1}^{n} \begin{cases} 0 & h'_{k_i c_i} \neq max_{j=0}^{N}(h'_{k_j c_i}) \\ 1 & h'_{k_i c_i} = max_{j=0}^{N}(h'_{k_j c_i}) \end{cases}}{n} \tag{3.14}$$

As an example, take the clusters of Table 3.5a. Assume we have a set of testing samples t_1, t_2, t_3 representing keys a, b, c clustered into c_1, c_2, c_3. Then, we have:

$$h'_{ac_1}, h'_{bc_2}, h'_{cc_3} = 0, 1/2, 1/5 \implies \begin{cases} A_1 = \frac{0+1+1}{3} = 0.66 \\ A_2 = \frac{0+1/2+1/5}{3} = 0.23 \\ A_3 = \frac{0+1+0}{3} = 0.33 \end{cases}$$

We take 70% of keystroke sounds for each user and group them into 104 clusters. Then, we predict the cluster for each sample keystroke sound from the test set. It is possible that the keystroke sounds representing a specific letter be assigned to different clusters. Also, multiple keystroke sounds representing the same character can be assigned to the same cluster. The average A_1, A_2, and A_3 values for all the users are 90%, 38%, 48% respectively. We show the distribution of the number of users based on the accuracy values in Fig. 3.7. Note that an A_2 value of 100% is only possible when all the samples that have been assigned to each cluster only represent a specific letter. In this case, each cluster represents only one letter. Similarly, an A_3 value of 100% means all the samples from the test set that represent a specific key are assigned to only one cluster. This also leads to each cluster representing only one letter. As a result, A_2 and A_3 can reach 100% in theory while it is not possible in practice and values of 38% and 48% do not necessarily show a low accuracy for the model.

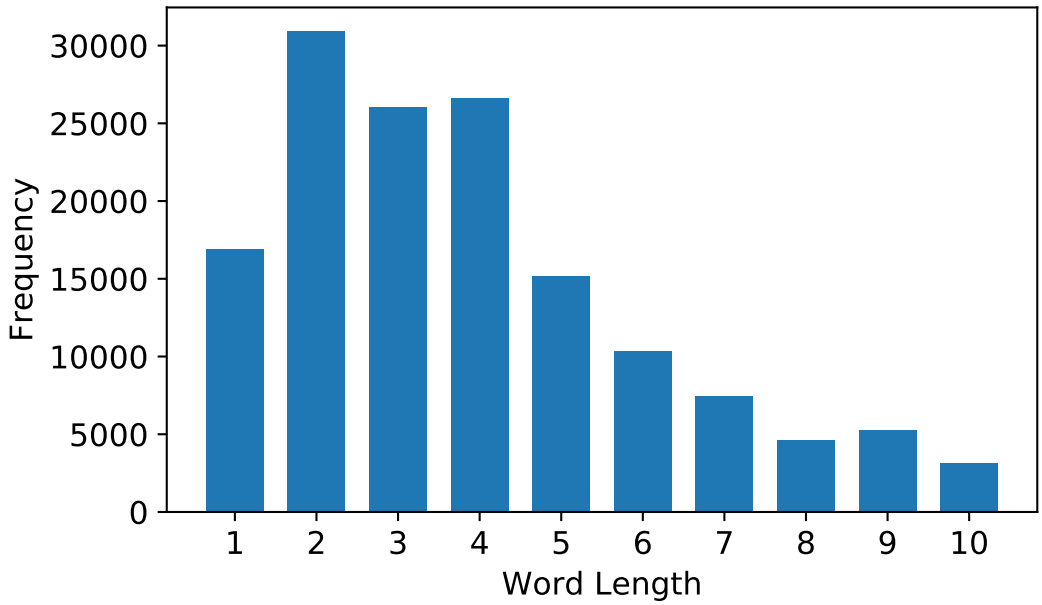
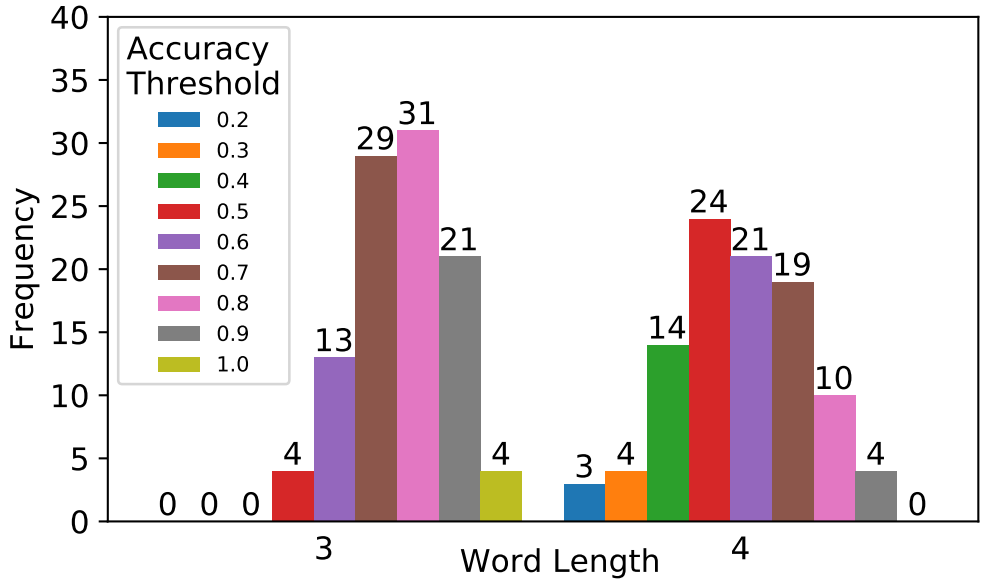


Figure 3.8: Word length distribution in benign and attack datasets combined.

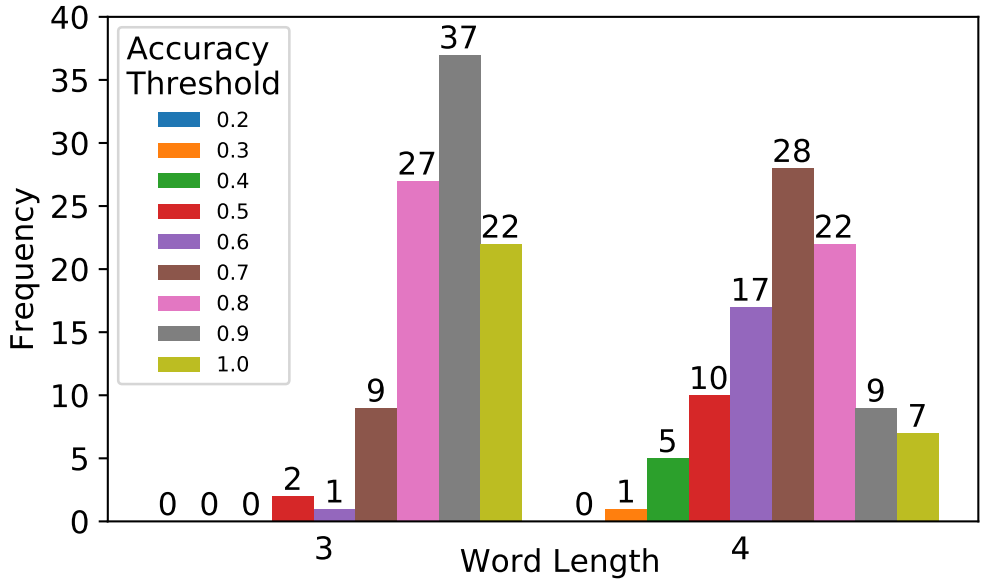
3.8.3 Recovering Text from Clusters

For each user, we get the keystroke sounds from 70% of the typed words to train the clusters and use the remaining 30% for testing. While we use the ground truth for identifying words using spaces and enters as separators for our evaluation, in a real-world scenario, we identify the separators by assigning them to a specific cluster that represents them. The cluster with the highest number of enters (or spaces) assigned to it in the training session, represents enter (or space). Later, when a keystroke sound is assigned to such cluster, we consider it as a possible separator. We take the words with a fixed length. Most of the typed words are between 2 to 4 letters, so we use those in most of our experiments. The distribution of words length in the datasets is shown in Fig. 3.8.

In Fig. 3.9, we show the distribution of the top-1000 accuracy for all of the users for 3 and 4-letter words with and without using the error correction module. With 3-letter words, the accuracy for most of the users is 80-90% while for 4-letter words it is 50-60%.



(a) Without error correction



(b) With error correction

Figure 3.9: Distribution of the number of users based on the top-1000 accuracy of the word recovery module for 3-letter and 4-letter words before and after using the error correction module.

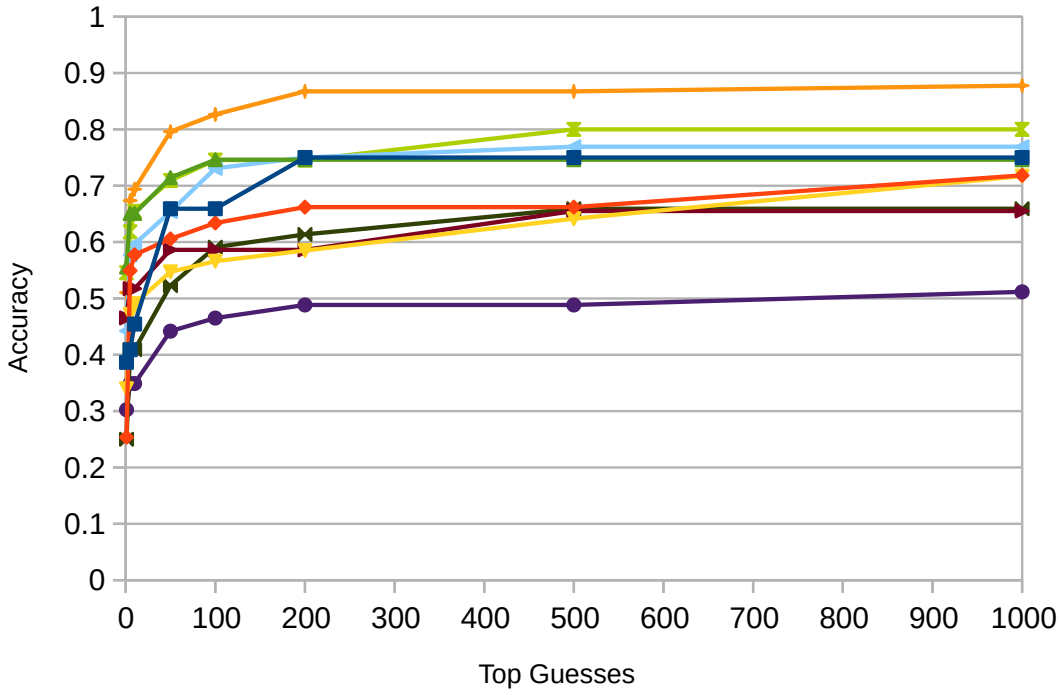


Figure 3.10: The accuracy of the text recovery module for 10 users when changing the number of top guesses which are considered as correct (Top-n Accuracy).

The average accuracy for all the users is 72% for 3-letter words and 53% for 4-letter words. The average top-10 accuracy for words of length 2-4 and all the users is 54% which will later be improved by the error correction module. We change the number of top guesses which are considered correct and show the accuracy for words with 2 to 4 letters for 10 users in Fig. 3.10.

Error Correction

We apply the error correction method (Section 3.6.7) to the guessed words in the previous experiment. Fig. 3.9 shows how the error correction module improves the performance of the word recovery module. With error correction, the average top-10 accuracy for words of 2 to 4 letters will be improved to 71% which is an acceptable accuracy considering our noisy dataset. Also, since text recovery is not the final goal of our model

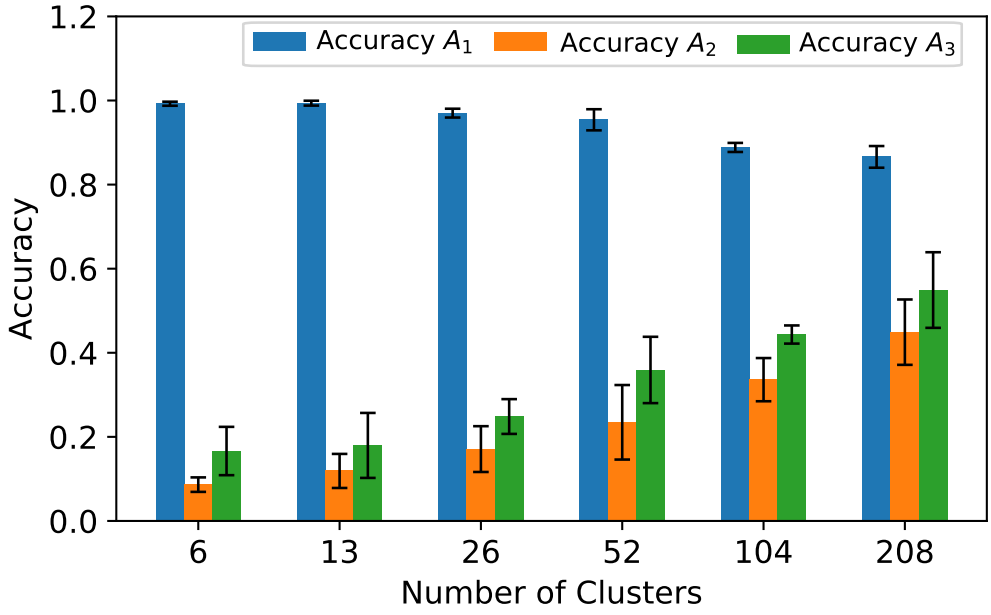


Figure 3.11: Accuracy of the clustering method evaluated using three metrics, changing number of the clusters.

and the recovered words will be passed to the activity detection module, the effect of accuracy for the text recovery module will be lowered. In Section 3.8.4, we show that if we use our activity detection model with the words directly fetched from the dataset (without extracting them from audio), the accuracy gain in comparison with using the words produced by the text recovery module is insignificant.

Choosing the Number of the Clusters

To find the optimal number of clusters, we take 30% of all the keystroke sounds from all the users as test data and classify them into the pre-trained clusters. Fig. 3.11 shows the evaluation results using our three metrics. When the number of clusters grows, the probability that a keystroke sound representing a key k be assigned to a cluster with at least one sample representing k from the training set decreases. At the same time, if a keystroke sound is clustered correctly because the number of samples is lower in each cluster, the probability of the key in the cluster (h') would be higher, resulting

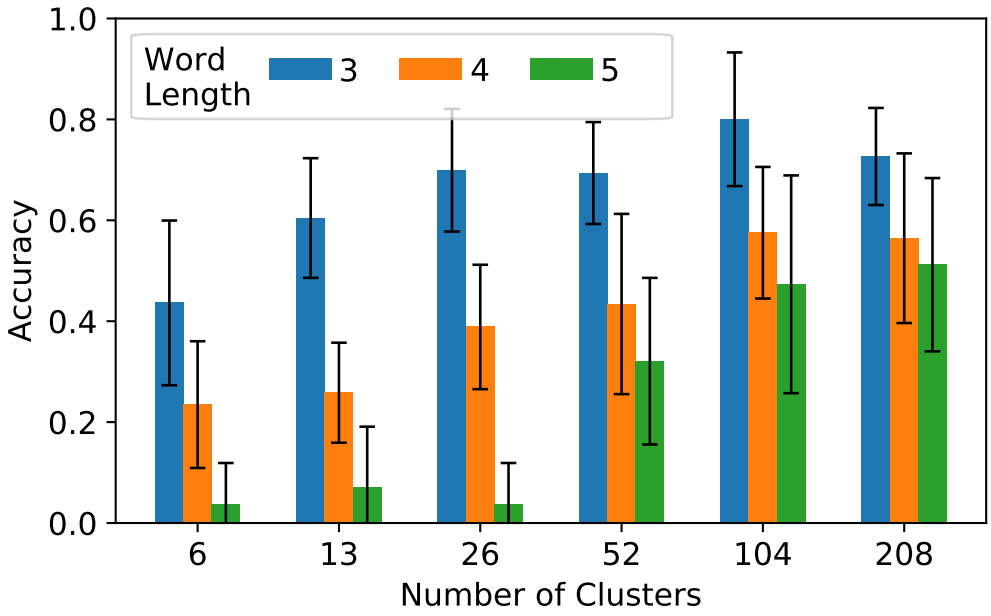


Figure 3.12: Top-1000 accuracy of the word recovery module for different word lengths and number of clusters.

in higher A_2 and A_3 values. Overall, the results show that using 104 and 208 clusters achieves the best performance.

In another experiment, we run the word extraction algorithm for each user with different word lengths and number of clusters. First, we get 70% of the words for each user and generate the clusters using the keystroke sounds from each word. Then, we take 30% of the words as the test data and classify them into clusters. Next, we use the predicted cluster to regenerate the words and consider a prediction correct if it is among the 1000 most probable words predicted by the algorithm. As shown in Fig. 3.12, 104 and 208 clusters achieve the best overall performance and the sum of performance metrics for 104 clusters is the highest. Therefore, based on the findings in Fig. 3.11 and Fig. 3.12, we choose to use 104 clusters for our experiments.

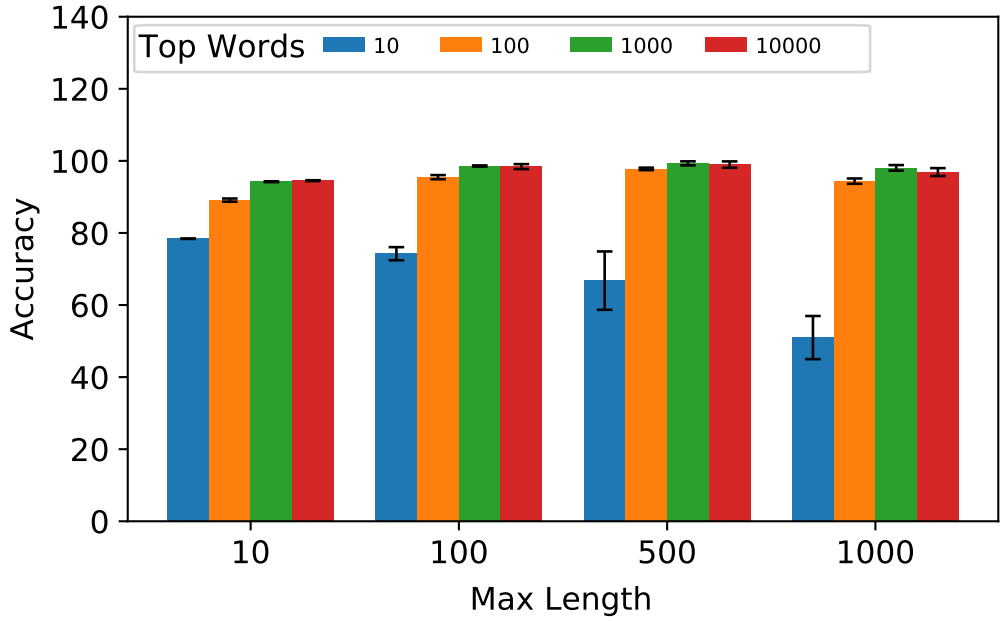


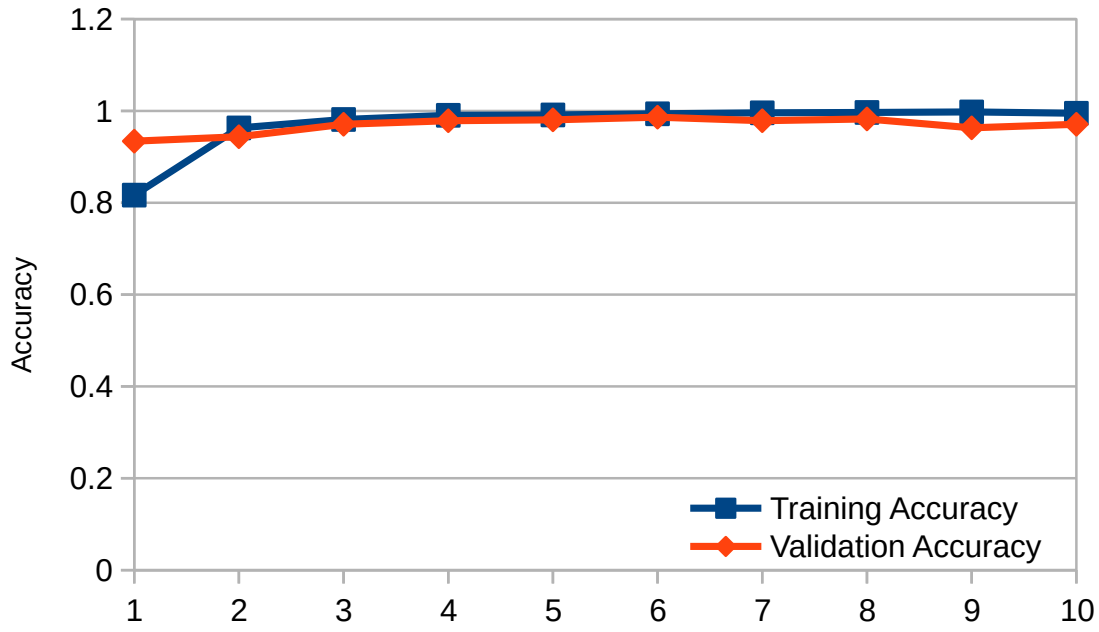
Figure 3.13: Accuracy of the activity classifier with different word sequence lengths and number of most frequent words.

Choosing the Values of α and β

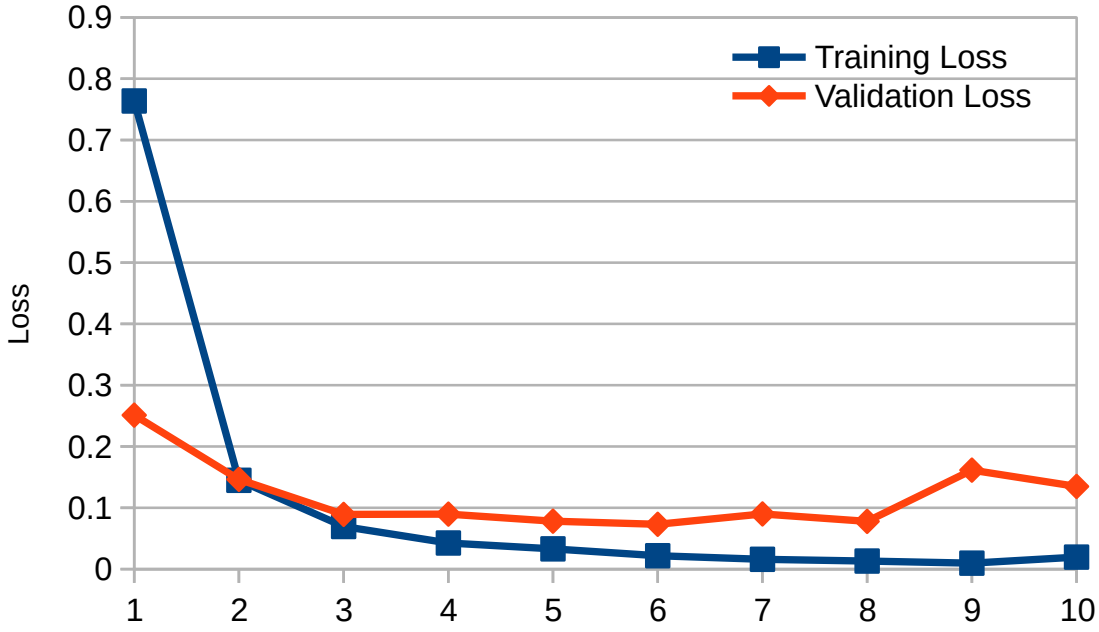
To choose the best values for α and β in Eq. 3.7 which control the effect of probability in the dictionary and the probability in the clusters for each recovered word, we run the same experiment that we used for evaluating the text recovery module, varying the values of α and β between 1 to 1000. Based on the experiment on the 3-letter and 4-letter words, the best results are achieved when $\alpha = 1$ and $\beta = 10$. So, we use these values for our experiments.

3.8.4 Adversarial Activity Detection and Classification

We directly take the ground truth for all the typed word sequences from the attack and benign datasets instead of the words generated by the text recovery module and fix their length. We only take the n most frequent words (top words) typed by all the users and



(a) Accuracy Curve



(b) Loss Curve

Figure 3.14: Training and validation loss and accuracy for activity detection algorithm with word sequences of length 50 and considering only the 1000 top words.

Table 3.6: Comparison of the performance of different algorithms for classifying word sequences to activities. The input data is fixed-length zero-padded sequences of 50 words considering only the 2000 most frequent words in the dataset.

Model	**Hyper-parameters**		**Accuracy**	
			Mean	**STD**
Nearest Neighbor	Number of Neighbors	3	0.65	0.01
RBF SVM	Regularization gamma	2 1	0.67	0.10
Decision Tree	Max depth	5	0.75	0.04
Random Forest	Max depth Number of estimators Max features	5 10 1	0.69	0.09
AdaBoost	Number of estimator Base estimatior	50 1	0.38	0.8
Naive Bayes	Variance smoothing	1e-9	0.28	0.04
QDA	Threshold	1e-4	0.46	0.03
Our Model	Embedding vector length LSTM layer size Number of epochs Batch size	512 100 10 32	0.98	0.01

discard (zero out) the rest. We train and test our model with different sequence lengths and top words and show the results in Fig. 3.13. Based on the sequence length, it is possible that a chosen sentence for classification include words from multiple commands or words from a part of a command. The best accuracy gained is 98.07% when we take only the 1000 top words and fix the sequence length to 50 words. The training and validation loss and accuracy of the model for sequences of 50 words and considering the 1000 most frequent words are shown in Fig. 3.14.

Furthermore, we use the words generated by the text recovery module to evaluate our activity detection model. We use sequences of 2 to 4 letter words from the dataset to train the activity detection model. Then, we use the text recovery module to generate words of length 2-4 using keystroke audio samples and feed them as test data to the activity detection module. We use sequences of 50 subsequent words for this experiment. Since we do not have the audio for the benign dataset, we directly get the words from it without recovering them from the audio. As a result of this evaluation, our model achieves 93.11% accuracy in detecting and classifying the activities.

In another experiment, we fine-tune the hyper-parameters of our model and compare the best-performing configuration with other classifiers. We use the fixed-length zero-padded sequences of words as features, and activities as labels for the training and testing data. For each classifier, we fine-tune the hyper-parameters separately to get the best possible accuracy. Among the several algorithms and configurations we used, decision tree results in the best performance with 75% accuracy which is significantly less than our model's performance. Detailed results of this experiment are reported in Table 3.6.

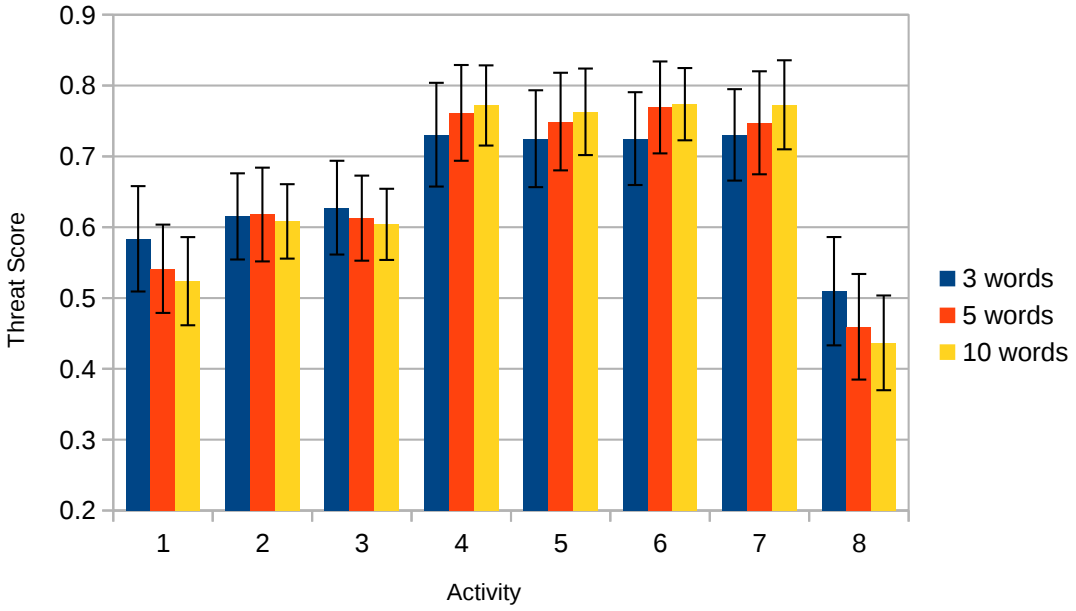


Figure 3.15: Average threat score for 100 samples of word sequences of length 3, 5, and 10 for different activities.

Classifying Real-Time Data

In our experiments, we assume that a sequence of typed words by the user is available for classification. However, some of the sequences are zero-padded to have same length sequences of words. Considering this, any number of words can be used for classification using our model. So, based on each new word that is detected from the user's typing audio, we can get a new classification decision and continuously improve the accuracy of threat level score. Furthermore, it is possible to update the threat level continuously after getting each k words.

3.9 Evaluating the Threat Score

We proposed a model for generating a threat score based on a sequence of words that have been typed by a user and then developed it into a continuous model to update

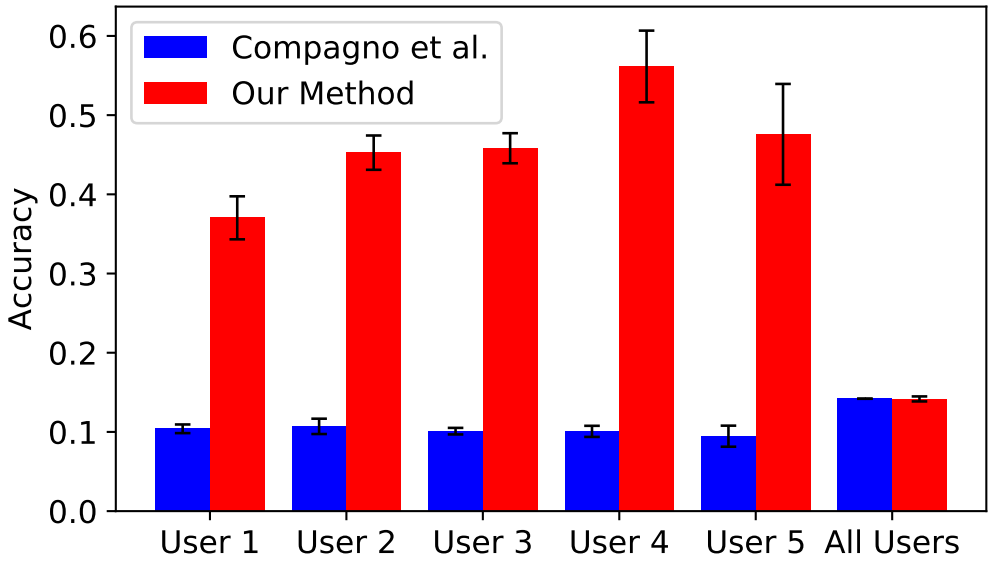


Figure 3.16: Comparison of the performance of our method with S&T [32] for retrieving single letters from keystroke sounds.

the threat score in real time. However, we did not provide a metric to evaluate how the numerical value of the threat score reflects the threat against a system. To relate the threat score to the actual threat level, we run a experiment by feeding sets of 3, 5, and 10 consecutive words from different users in different activities in the dataset to the activity detection module and then, we generate the threat score for each. We show the average for 100 repetitions of each experiment in Fig. 3.15. The threat score is generated using Eq. 3.9 with $d = \{0.25, 0.5, 0.5, 1, 1, 1, 1, 0\}$. As we can see, for the activity 8 which is the benign activity, the average threat score is less than 0.5 while for the activities with highest maliciousness level, the average threat score is between 0.7 and 0.8. So, we can define the range for our threat score between 0 and 1 while we consider scores below 0.5 to be completely safe and scores closer to 1 to show the highest maliciousness level.

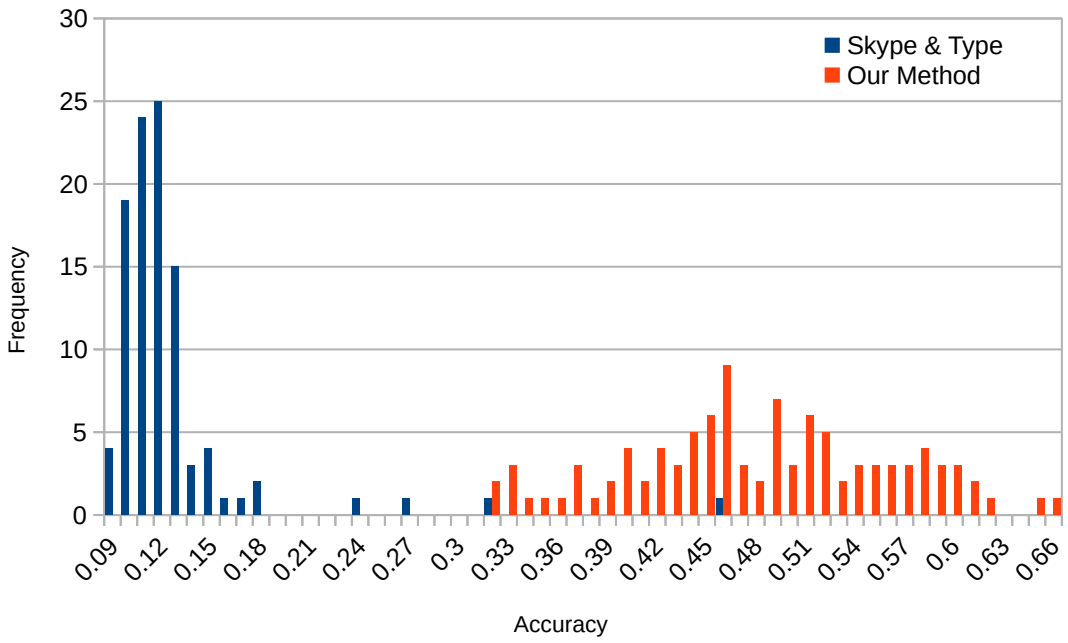


Figure 3.17: Distribution of users based on accuracy for classifying each user's key sounds using S&T [32] method and our method.

3.10 Comparison with Other Work

There are a limited number of prior studies with publicly available source code to be evaluated with our data. Among those, we evaluate a similar work *Skype & Type!* (S&T) [32] with our dataset. S&T has different components similar to our work. The audio segmentation module uses a peak detection method to extract individual keystorke sounds from the raw audio file. However, this component as implemented does not work on our dataset because of the presence of noise and inaudible keystrokes. So, we remove this component and use already extracted individual keystroke sounds and their labels with their classification model. S&T generates MFCC features with similar hyper-parameters to ours and uses a feature selection algorithm to reduce the number of features. We run their model using different classifiers with the audio data from 5 of the users in our dataset separately and once with the audio data of all the users combined. Note that their model directly classifies the feature vector and assigns it

to a key while our classification method generates a probability for multiple possible audio letters (clusters) representing each MFCC feature vector and does not give an explicit key for each sample. We use A_3 as the metric for calculating the accuracy of our work. Using this metric is fair for comparison because for each sample keystroke sound, we output one cluster and for each cluster, we only take the most probable letter as correct. So, in this evaluation method, our classifier gets a keystroke sound as input and outputs a key (E.g. letter) like what S&T does. The results for all users and 5 users are presented in Fig. 3.16 showing significantly better accuracy for our model in comparison to S&T for single users and similar accuracy for all users combined. In Fig. 3.17, we show the frequency of the users based on the classification accuracy of their keystroke sound samples for both S&T and our method. We need to mention that our method can achieve even better accuracy because we build a probability model to consider all the possible letters in the cluster and do not only take the most probable letter as the result of the classification. So, our model does not assign a specific key to a keystroke sound sample and the subsequent modules (text extraction, error correction) will decide about the final chosen key.

3.11 Summary

We showed that it is possible to detect different adversarial activities and evaluate the threat level against a computer system using the audio recordings of typing sessions with significant accuracy. To show that, we proposed a model with multiple components to extract text from typing audio, detect activity based on extracted words, and quantify the threat level based on the activity.

Following the detection and prediction of the user's activity, we are also interested to find out whether the activities and the behavior of the users in their interaction with

a computer are correlated with their personality. So, in the following chapter, we try to get insights into the personality of people and how it is related to their usage of keyboard and mouse.

Chapter 4

On the Correlation of Keystroke and Mouse Dynamics With Personality

4.1 Introduction

Research in the past has directed significant attention to keystroke and mouse dynamics of individuals because they provide a primary interface to a desktop computer. While many studies in the past have addressed authentication [49, 50, 51] and identification [52, 53], (and some isolated works related to emotions, social networks, etc. (see Section 4.2)); to the best of our knowledge, none have analyzed the personality of people and its correlation with the users' interactions with a computer through keystroke dynamics and mouse usage.

Creating the personality and psychological profile of users based on their interaction with the computer can be used for different applications, such as showing personality-directed content to the users, advertisement, law enforcement, and malicious behavior detection and prevention. It can also be used in chatting and match-making applica-

tions to connect people with similar personalities to each other.

In this research, using data from 104 participants, we present how characteristics of keyboard and mouse usage relate to the users' personality traits, as inferred through two psychological batteries–*The Holland Code Test* [54] and *The Big Five Personalities Test* [55].

The contributions of this research are:

- We introduce several empirically derived features related to the keystroke dynamics and mouse usage of the users. Our features are unique because they inherently capture the usage patterns of certain tasks and may be used to find the correlation with users' personality types.
- We show that certain personality traits such as extraversion are correlated with the typing dynamics of the users such as the number of clicks and keys and different personalities present different behaviors in their interaction with a computer. For example, people with conventional personality type tend to spend more time with a computer to do a certain task and people with social personality type use the mouse relatively more than the keyboard in their interaction with the computer (see Section 4.5 for more details).
- We posit that this study along with the dataset will serve as a benchmark to build further analysis linking psychological profiles of individuals with computer usage. Our dataset provides unique advantages because it is large (104 users) and includes typing and mouse data obtained through multiple activities together with the answers to questionnaires from which the personality and psychological profile of the users may be interpreted.

Table 4.1: Description and statistics of the six personalities that are categorized by the Holland Code Test [56]. The shaded columns show the statistics of the results on our dataset.

Personality	**Description**	**Count**	**Gender**		**Avg Age**
			M	**F**	
Realistic	Practical, physical, concrete, hands-on	2	2	0	27.50
Investigative	analytical, intellectual, scientific	21	15	6	25.47
Artistic	creative, original, independent	18	16	2	23.50
Social	cooperative, supporting, patient	37	25	12	24.16
Enterprising	competitive, leadership, persuading	8	8	0	23.87
Conventional	detail-oriented, organizing, careful	14	12	2	23.57

Table 4.2: Five personalities evaluated by the Big Five Personalities Test [57]. The shaded columns show the statistics of the results on our dataset for users with high (H), medium (M), and low (L) personality scores. The criteria for score range selection is described in Section 4.4.5.

Personality	**Description**		**Average Age**			**Male**			**Female**		
	High Score	**Low Score**	**H**	**M**	**L**	**H**	**M**	**L**	**H**	**M**	**L**
Extraversion	Outgoing, warm	Quiet, reserved	23.13	24.82	24.16	18	41	18	4	12	7
Agreeableness	Helpful, trusting	Critical, uncooperative	23.84	24.50	24.33	24	47	6	8	15	0
Conscientiousness	Hardworking, organized	Impulsive, careless	23.17	25.22	23.66	30	37	10	4	14	5
Neuroticism	Anxious, unhappy	Calm, secure	24.60	24.27	23.96	35	40	2	4	15	4
Openness to Exp.	Curious, independent	Practical, conventional	24.82	24.03	22.50	32	42	3	7	15	1

4.2 Related Work

Keystroke and mouse dynamics have been used for different applications like identification, authentication, and emotion detection. Nahin et al. [58] study emotion identification based on keystroke dynamics. They collect a dataset with self-reported emotional states and keystrokes from users and build a model for emotion detection. Multiple other studies address emotion [59, 60], gender [61, 62], age [62, 63], and handedness [64] prediction and identification using keystroke dynamics in different environments and applications like social networks [65] and chatting platforms [61]. Roy et al. [66] address the same problem using touch screen interactions on the mobile. Our work is unique in comparison with the prior work because we use personality types determined using standard psychological tests and study their correlation with mouse and keyboard dynamics. In a closer approach to our study, Katerina et al. [67] extract features from mouse and keystroke dynamics and study their correlation with behavioral variables like self-efficacy, risk perception, etc. They use a questionnaire survey for evaluating the level of the behavioral variables for users and their analysis methods and behavioral variables are different from ours. They also use a different set of features related to the typing patterns and mouse movements.

4.3 Keyboard, Mouse, and Personality Dataset

After appropriate IRB approval, our dataset is collected from 104 participants during a series of predefined activities on a computer in our lab which is connected to the school network. Throughout the session, participants were free to search over the internet and surf the web to get help and instructions on how to perform the requirements of the tasks and activities. The data collection scenario consisted of seven different activities that complement each other. The activities require the users to do several tasks that lead

to connecting to a vulnerable virtual machine on a network and stealing information from it. Several modalities have been collected in this data collection effort, including keystrokes and keystroke timings, mouse clicks and movements, audio recordings during the session, and system events. For this research, we use the keystroke and mouse recordings. On average, 3261 keystrokes and 303 mouse clicks have been collected over a course of about 90 minutes from each of 83 male and 21 female participants.

Table 4.3: Description and statistics of the features that are used for our analysis. The average and standard deviation are calculated for all the users in our dataset after outlier removal.

Feature	**Description**	**Average**	**STD**
Duration	The duration of the session.	4776 s	999 s
Keys	The number of keystrokes for the session.	3274	1267
Clicks	The number of left mouse button clicks for the session.	316	132
Clicks/Keys	The ratio of clicks to keys.	0.10	0.03
Latency	Average inter-key latency for sequential keystrokes.	0.24 s	0.04 s
Dclick	Mouse double-click latency with 500ms threshold.	0.29 s	0.06 s

4.3.1 Personality Tests

All the participants of the dataset have also participated in two personality tests. The Holland Code Test [54] asks 48 career-related questions and categorizes the participants into 6 different personality types which are shown in Table 4.1. This Test focuses on career-related personalities and how each person fits into different kinds of occupations. The results of this test can help us understand whether career-related personalities can affect people's interaction with the computer through mouse and keyboard.

The second questionnaire is the Big Five Personality Test [55] which asks 50 personality-related questions from the participants and indicates the level of five different personality traits for them as shown in Table 4.2. This test indicates the way a person acts

and how the person's personality is structured. Unlike the Holland Code Test, this test does not categorize a person into a certain category. Instead, it will determine how much each of the traits defines the person's character.

4.3.2 Questionnaire Samples

Here we list some of the questions from the personality test questionnaires used in this study.

Sample Questions From the Holland Code Test

In the Holland Code Test, each question is linked to a career-related personality type and is ranked on a Likert scale:

- Test the quality of parts before shipment
- Study the structure of the human body
- Conduct a musical choir
- Give career guidance to people
- Sell restaurant franchises to individuals

Sample Questions From the Big Five Personalities Test

Similar to the Holland Code Test, the questions in the Big Five Personalities Test are ranked on a Likert scale:

- I am the life of the party.
- I feel little concern for others.

- I am always prepared.
- I get stressed out easily.
- I have a rich vocabulary.

4.4 Methodology

In this section, we describe the methods and steps that we take to pre-process the data and generate the results for our analysis.

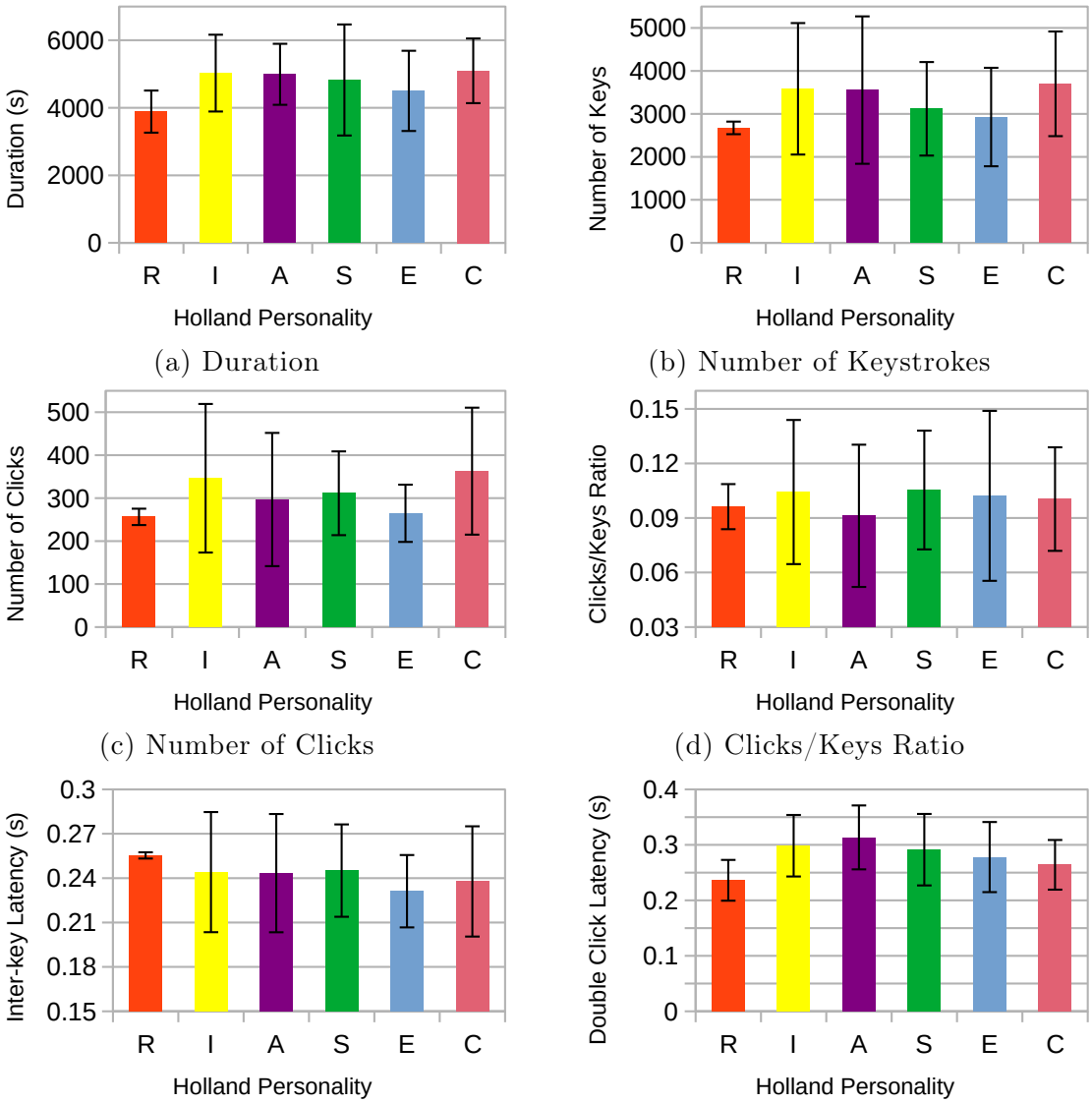
4.4.1 Feature Extraction

With all the keystrokes, mouse clicks, and mouse movements of the users recorded, we generate several features that are useful to extract information about behavioral and personality patterns in people's interaction with the computer. We show the extracted features together with the average and standard deviation for each feature among all the users in our dataset in Table 4.3. The duration of each activity is available for all the users. We use the total session duration as a feature for our analysis. Similarly, we use the total number of keystrokes and left mouse button clicks as features. Later, in Section 4.5.4, we analyze each feature for different activities in more details. The number of right mouse button clicks and the mouse wheel usage is low for each user since they are not frequently used in a regular session with a computer, so, we do not use them for our analysis. The clicks/keys ratio is a useful feature to determine the experience and skill of the users in their interaction with a computer, because more experienced users use the keyboard more for doing their tasks and are also more familiar with keyboard shortcuts. So, we expect their clicks/keys ratio be relatively lower. We use this feature to determine which input device is relatively used more

during the session. Another feature that we extract for our analysis is the inter-key latency. We get the average time difference between each sequential keystrokes pair. We choose a 1-second cut-off threshold based on our experiments and do not include the time difference between any two sequential keys that have not been pressed within 1 second in calculating the average because the participant is not always typing during the session and may pause for thinking, viewing and reading the monitor, using the mouse, and scrolling. Inter-key latency can be used as a metric for evaluating the user's typing speed. A higher inter-key latency indicates lower typing speed. Lastly, we use the double-click latency as another feature which is calculated by taking the time difference between every two sequential left mouse button clicks that does not exceed 500 milliseconds. Any sequential click with a time difference of more than 500ms is not considered double-click because this is the default double-click delay set in the data collection machine.

4.4.2 Outlier Removal Based on Extracted Features

We discard the data for 4 out of 104 participants through our outlier detection process and use the data from the 100 remaining participants for our analysis in the rest of this study. As an example, the age of one participant is 46 while the average age in our data is 26, and when this participant is removed, the age range will be from 20 to 33. There are also other participants with short session duration which means they have not completed all the activities and the tasks of the data collection and should be considered outliers. To find and remove these participants, we use the Z-score outlier detection method on each feature. For example, for the total duration of the activity, we calculate the z-score as $Z = \frac{x-\mu}{\sigma}$ where x is each user's duration and μ and σ are the mean and the standard deviation of the duration for all users. We take records with a z-score of more than 3 or less than -3 as outliers. For the duration, the z-score for



(a) Duration

(b) Number of Keystrokes

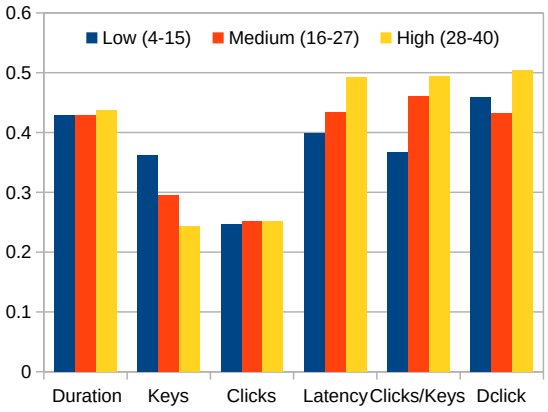
(c) Number of Clicks

(d) Clicks/Keys Ratio

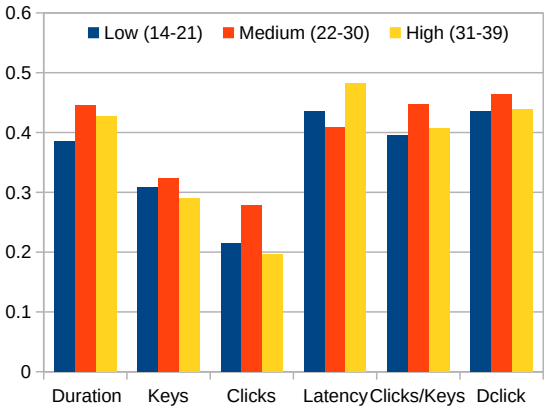
(e) Inter-key Latency

(f) Double-click Latency

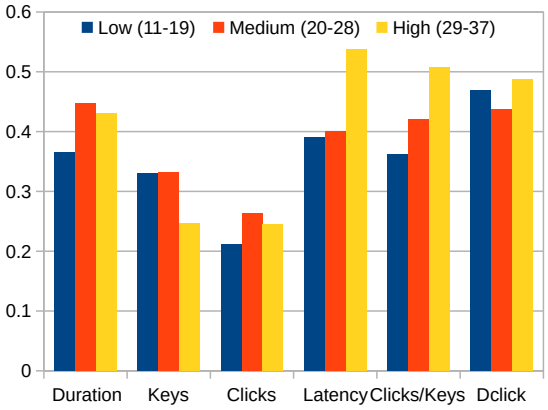
Figure 4.1: Comparison of feature values average for each of the Holland Code Test personality types.



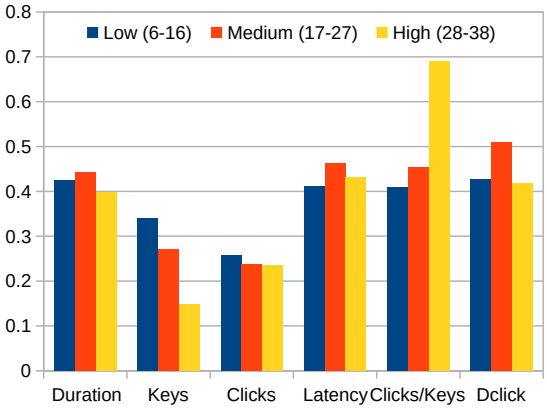
(a) Extraversion



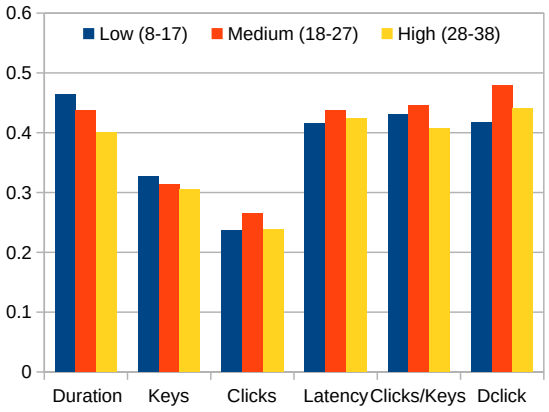
(b) Agreeableness



(c) Conscientiousness



(d) Neuroticism



(e) Openness to Experience

Figure 4.2: Comparison of feature values average for the personality types of the Big Five Personalities Test. The Y-axis shows the normalized values for each feature. All the feature values are normalized between 0 and 1 to be shown in the same figure.

one of the users is 6.06 which is significantly higher than the population. The user's duration value is 13111 seconds while $\mu = 2810$ and $\sigma = 1369$. An abnormally high duration can be due to a temporal halt during the data collection, for example, a phone call or an emergency might have caused the user to idle during the session.

4.4.3 Correlation Analysis Methods

To evaluate the correlation between each Holland personality type and our features, we use the point-biserial correlation coefficient [68]. This method is useful for calculating the correlation when one of the variables is categorical. Because the categorical variable should be binary, we set it separately for each personality type. For example, for realistic personality type, we assign 1 to all the users who have been categorized as realistic and 0 to the rest of the users. Then we use the point-biserial method to calculate the correlation between this vector and each feature vector. To calculate the correlation coefficient r_{pb}, we split the users into two groups 0 and 1 based on their binary variable value. We show the mean of the feature values for group 0 with M_0 and for group 1 with M_1. Respectively, we show the number of users in each group with n_0 and n_1. With s_n as the standard deviation of the feature vector, we calculate: $r_{pb} = \frac{M_1 - M_0}{s_n} \sqrt{\frac{n_1 n_0}{n_1 + n_0}}$.

To evaluate the correlation between each feature and each personality type for the big five personalities test, we use Pearson correlation coefficient which is calculated as $\rho_{P,F} = \frac{cov(P,F)}{\sigma_P \sigma_F}$ where P is the personality score, F is the feature value, and σ_P, σ_F are the standard deviations for P and F.

4.4.4 Holland Code Test Analysis Method

The Holland Code Test assigns a score in the range of 5-40 for each of the six personality types to each user and considers the one with the highest score as the final personality type for the user. To interpret the patterns and trends in the results of the test, for the users assigned to each personality type, we calculate the average and the standard deviation of each feature. Then, we observe the personality types that have higher or lower values in most features and the ones that have the highest or lowest averages in each feature. We analyze the patterns and trends that can be inferred from the results of this test in Section 4.5.1. In our analysis, we try to reason about the characteristics of each personality type and how they are related to the features extracted from the interaction with the computer. However, accurate and theoretical psychological analysis of the personalities are out of the scope of this research.

4.4.5 The Big Five Personalities Test Analysis Method

The big five personalities test assigns a score in the range of 0-40 to each personality trait for each user. We split the score for each personality trait into three different ranges: low, medium, and high. If m is the minimum, and M is the maximum score for personality trait p among all the users, we divide the range $M - m$ into three equal ranges and round up the ranges in case $M - m$ is not divisible by 3. Then, we calculate the average of each feature for all the users in each range. So, for each feature, we know the average for the low, medium, and high range. Then, we can use these values to evaluate the trends for each feature. The results and analysis for this personality test are reported in Section 4.5.2.

Table 4.4: Correlation statistics for Holland Code test calculated using point-biserial correlation coefficient method. For each personality, all the users with that personality are assigned the value 1 and rest have been assigned 0 as the binary categorical variable for point-biserial correlation calculation. More significant values with p-value<=0.05 are shown in bold.

Personality	**Duration**		**Keys**		**Clicks**		**Latency**		**Clicks/Keys**		**Dclick**	
	Corr	**p-val**	**Corr**	**p-val**	**Corr**	**p-val**	**Corr**	**p-val**	**Corr**	**p-val**	**Corr**	**p-val**
Realistic	-0.11	0.27	-0.07	0.47	-0.07	0.51	-0.15	0.13	-0.02	0.84	-0.01	0.88
Investigative	0.06	0.59	0.09	0.37	0.11	0.28	**0.29**	**0.00**	0.04	0.68	**0.17**	**0.05**
Artistic	0.04	0.71	0.07	0.48	-0.08	0.45	-0.01	0.91	**-0.14**	**0.02**	0.01	0.94
Social	-0.04	0.69	-0.13	0.18	-0.04	0.69	-0.09	0.35	**0.09**	**0.04**	**-0.23**	**0.02**
Enterprising	**-0.09**	**0.03**	**-0.09**	**0.03**	**-0.12**	**0.05**	-0.04	0.71	0.01	0.95	-0.04	0.68
Conventional	0.06	0.52	**0.11**	**0.03**	**0.14**	**0.05**	-0.11	0.29	-0.01	0.90	0.15	0.12

Table 4.5: Correlation statistics for big five personality test calculated using Pearson correlation coefficient method. More significant values with p-value<=0.05 are shown in bold.

Personality	**Duration**		**Keys**		**Clicks**		**Latency**		**Clicks/Keys**		**Dclick**	
	Corr	**p-val**	**Corr**	**p-val**	**Corr**	**p-val**	**Corr**	**p-val**	**Corr**	**p-val**	**Corr**	**p-val**
Extraversion	-0.04	0.70	**-0.24**	**0.02**	-0.03	0.78	0.11	0.30	**0.19**	**0.05**	0.00	0.99
Agreeableness	**0.19**	**0.05**	0.03	0.77	0.06	0.55	0.03	0.78	0.06	0.53	0.00	1.00
Conscientiousness	0.14	0.16	-0.11	0.27	0.06	0.57	**0.16**	**0.01**	0.18	0.07	-0.03	0.77
Neuroticism	0.05	0.61	**-0.13**	**0.02**	0.00	0.98	0.12	0.24	**0.21**	**0.04**	0.03	0.77
Openness to Exp.	**-0.06**	**0.04**	0.00	0.96	0.00	0.96	-0.05	0.64	-0.06	0.55	0.06	0.59

4.5 Results and Analysis

In this section, we present the results of our study and analyze the trends and the patterns that exist in the relation between personality types and the extracted features from the mouse and keyboard for each personality test.

4.5.1 Holland Code Test Results Analysis

As shown in Fig. 4.1, participants with conventional personality type have the highest number of keys, clicks, and the longest average duration. The correlation statistics (Table 4.4) also approves that conventional personality trait is slightly correlated with the number of keys and clicks. This can be expected as people with this personality tend to be detail-oriented, careful, and organizing. So, they spend more time and effort to deliver the best performance in fulfilling the tasks.

On the other hand, the competitive characteristic of enterprising people can be the reason for their shorter time to success and fewer keys and clicks as shown in Fig. 4.1 and Table 4.4.

Most of the feature values for the realistic group are lower, but we do not rely on them to make strong statements because the number of participants assigned to this group is low.

In Fig. 4.1d, we can see that the clicks/keys ratio for the people with artistic personality type is the lowest, so, they use the keyboard more than the mouse in their interaction with the computer. Also, Table 4.4 shows a slight negative correlation between the artistic personality type and clicks/keys ratio.

In contrast, participants with social personality have the highest clicks/keys ratio.

One characteristic of the social people is patience which is reflected in their high duration, number of clicks, and clicks/keys ratio which indicates higher usage of the mouse. Working with the mouse usually takes longer and needs more effort for doing tasks with the computer while it makes the interaction easier for less experienced users in comparison with the keyboard. In addition, Table 4.4 shows negative correlation between social personality trait and double-click latency.

Investigative personality is positively correlated with inter-key latency. This shows that people with this personality are usually slower typists. Furthermore, there is a slight positive correlation between investigative personality and double-click latency.

As shown in Table 4.1, fewer people have been assigned to enterprising and realistic personality types. While most of our participants are from computer science background, we can conclude that realistic and enterprising personalities are less common among the students with computer science background in the 20-30 years old age range in our dataset. Users with realistic and investigative personality types have the highest average age while artistic users are the youngest in our dataset. Artistic users have the highest and social users have the lowest male/female ratio and most of the users have been categorized as social.

4.5.2 The Big Five Personalities Test Results Analysis

To present the results and findings based on the big five personalities test, we partition the personality trait scores into three segments as described in Section 4.4.5. For example, after removing the outliers, the extraversion score for the users falls in the range of 4-40. So, we partition this range into three ranges of 4-15, 16-27, and 28-40 and refer to them as low, medium, and high respectively. The results for each personality trait are shown in Fig. 4.2.

There are various patterns and information we can extract based on the trends of each feature for each personality trait. As shown in Fig. 4.2a, people with higher extraversion score have higher clicks/keys ratio and lower number of keystrokes, so they use the mouse relatively more than the keyboard and their total keyboard usage is lower. Also, their inter-key latency is higher, showing a slower typing speed. Correlation results in Table 4.5 also show that extraversion is slightly correlated with clicks/keys ratio and negatively correlated with the number of keys. So, people with higher extraversion score, they tend to use the graphical interface and mouse more.

Fig. 4.2b does not provide any trends or patterns in the typing behavior using our defined features for agreeableness and Table 4.5 only shows a slight correlation with duration. So, we cannot make any conclusion about this personality trait based on our results (Fig. 4.2b).

The trends for conscientiousness personality are similar to those of extraversion. A higher conscientiousness score leads to a lower typing speed (higher inter-key latency) and a higher mouse usage (Fig. 4.2c). The correlation analysis also approves correlation between conscientiousness and inter-key latency (Table 4.5).

People with higher neuroticism scores succeed in fulfilling the requirements of the task with lower usage of keyboard and mouse, while there is not any significant trend in their duration of activity. So, people with this personality trait can do more with less interaction with the computer, which means their usage of the input devices is optimal. They tend to focus on the screen and the task and get the maximum information without having to type multiple commands or browse different websites (Fig. 4.2d). Furthermore, people with higher neuroticism score have a higher average clicks/keys ratio which can also be inferred from Table 4.5.

Users with higher openness to experience scores are able to finish their tasks faster

Table 4.6: Classification results using different machine learning algorithms for classifying user personality types in Holland code test using different feature sets as described in Section 4.5.3.

Model	Hyper-parameters		Accuracy					
			All Features	All Demographic Features	All Typing Features	Selected Features	Selected Demographic Features	Selected Typing Features
Nearest Neighbor	Num of Neighbors	3	0.26	0.26	0.23	0.30	0.26	0.40
Linear SVM			0.36	0.46	0.40	0.43	0.46	0.50
RBF SVM	Regularization gamma	2 1	0.40	0.30	0.40	0.43	0.46	0.50
Gaussian Process			0.23	0.36	0.20	0.43	0.46	0.46
Decision Tree	Max depth	5	0.26	0.26	0.30	0.36	0.30	0.36
Random Forest	Max depth Num of estimators Max features	5 10 1	0.36	0.40	0.30	0.40	0.33	0.36
Neural Net			0.33	0.36	0.26	0.43	0.46	0.50
AdaBoost	Num of estimators Base estimatior	50 1	0.36	0.20	0.30	0.33	0.33	0.43
Naive Bayes	Variance smoothing	1e-9	0.30	0.10	0.43	0.26	0.23	0.36
QDA	Threshold	1e-4	0.30	0.23	0.33	0.40	0.30	0.30

with lower usage of the keyboard. While we cannot get much information about their interaction with the mouse, they seem to be more skillful in performing the assigned tasks (Fig. 4.2e and Table 4.5).

Based on the results shown in Table 4.2, users with high openness to experience score are the oldest, and users with low scores are the youngest in our dataset. Users with high neuroticism and low agreeableness score have the highest male/female ratio.

4.5.3 Classification Results for the Holland Code Test

To find out if it is possible to get insights about the personality types of people using their demographics and typing patterns, we use different machine learning algorithms with multiple feature sets for classifying users' Holland code test personality types. To determine the most selective features for classifying users into different personality types, we run the CFS feature selection algorithm with BestFirst search method on our

Table 4.7: Correlation between the big five personality traits and the duration of each activity for all users.

Duration	**1**		**2**		**3**		**4**		**5**		**6**		**7**	
	Corr	p-val	Corr	p-val	Corr	p-val	Corr	p-val	Corr	p-val	Corr	p-val	Corr	p-val
Extraversion	-0.06	0.57	-0.12	0.24	0.00	1.00	-0.18	0.08	-0.09	0.37	**-0.21**	**0.04**	0.15	0.15
Agreeableness	-0.09	0.36	-0.09	0.35	-0.11	0.26	-0.16	0.12	-0.08	0.45	0.00	0.98	0.24	0.02
Conscientiousness	-0.03	0.74	-0.08	0.44	-0.10	0.32	-0.04	0.70	-0.11	0.30	0.10	0.34	0.07	0.52
Neuroticism	0.01	0.95	0.00	0.98	-0.01	0.89	0.01	0.92	0.03	0.77	0.13	0.21	0.11	0.27
Openness to Exp.	-0.09	0.35	**-0.27**	**0.01**	**-0.31**	**0.00**	-0.14	0.17	**-0.25**	**0.01**	**0.31**	**0.00**	**0.32**	**0.00**

data.

For the Holland Code Test, the results of feature selection show that height, inter-key latency, handedness, age, double-click latency, clicks/keys ratio are the most selective features. If we only consider typing pattern related features, the selected features are inter-key latency, double-click latency, and clicks/keys ratio. Considering only demographic features, height, handedness, and age will be selected.

In each experiment, we train our model with 70% of randomly selected users' data and test it with data from the remaining 30% of users. Since an accurate classification needs a huge dataset and we have only 100 users available, we do not expect perfect results, but we can still get insights about whether these features can be used to get information about the personality types of the users. As shown in Table 4.6, using the typing features, we can get better accuracy in classifying the personality type. While the results show that the personality type of the users is reflected in their typing behavior, we cannot strongly conclude that we can detect the personality type based on typing patterns and demographics.

Table 4.8: Correlation between the big five personality traits and number of keys of each activity for all users.

Keys	**1**		**2**		**3**		**4**		**5**		**6**		**7**	
	Corr	p-val	Corr	p-val	Corr	p-val	Corr	p-val	Corr	p-val	Corr	p-val	Corr	p-val
Extraversion	0.04	0.67	-0.03	0.80	0.19	0.06	-0.10	0.31	0.09	0.39	-0.10	0.31	0.11	0.28
Agreeableness	0.07	0.48	0.14	0.18	0.11	0.30	0.04	0.68	0.14	0.16	0.10	0.34	0.15	0.15
Conscientiousness	0.01	0.93	-0.04	0.70	-0.03	0.74	0.05	0.59	0.00	0.98	**0.23**	**0.02**	0.18	0.07
Neuroticism	-0.03	0.77	**-0.20**	**0.05**	-0.07	0.47	-0.02	0.81	-0.02	0.81	0.08	0.45	0.05	0.64
Openness to Exp.	-0.01	0.89	0.04	0.68	-0.11	0.29	0.05	0.60	-0.06	0.54	**0.31**	**0.00**	**0.30**	**0.00**

Table 4.9: Correlation between the big five personality traits and number of clicks of each activity for all users.

Clicks	**1**		**2**		**3**		**4**		**5**		**6**		**7**	
	Corr	p-val	Corr	p-val	Corr	p-val	Corr	p-val	Corr	p-val	Corr	p-val	Corr	p-val
Extraversion	-0.06	0.54	-0.04	0.71	0.06	0.53	-0.15	0.13	-0.04	0.68	-0.09	0.37	0.03	0.75
Agreeableness	0.06	0.59	-0.09	0.35	0.00	0.98	-0.15	0.14	-0.09	0.35	0.05	0.63	0.06	0.58
Conscientiousness	0.01	0.88	-0.01	0.89	-0.01	0.92	0.04	0.68	0.02	0.88	0.15	0.13	0.13	0.19
Neuroticism	-0.06	0.53	0.00	0.97	-0.05	0.64	-0.03	0.80	0.00	0.99	0.08	0.43	-0.02	0.83
Openness to Exp.	0.09	0.37	**-0.20**	**0.04**	-0.16	0.12	-0.11	0.30	-0.06	0.54	**0.26**	**0.01**	**0.24**	**0.02**

Table 4.10: Correlation between the big five personality traits and double-click latency of each activity for all users.

Double-click Latency	**1**		**2**		**3**		**4**		**5**		**6**		**7**	
	Corr	p-val	Corr	p-val	Corr	p-val	Corr	p-val	Corr	p-val	Corr	p-val	Corr	p-val
Extraversion	-0.11	0.26	0.00	0.98	**0.22**	**0.03**	0.18	0.07	-0.04	0.70	0.12	0.23	0.04	0.66
Agreeableness	0.03	0.80	0.06	0.55	**0.20**	**0.05**	-0.01	0.91	0.11	0.27	0.07	0.49	**0.27**	**0.01**
Conscientiousness	-0.12	0.26	0.07	0.47	-0.17	0.10	0.05	0.60	**-0.23**	**0.02**	0.08	0.44	-0.14	0.18
Neuroticism	0.09	0.39	-0.05	0.66	0.06	0.53	0.01	0.94	-0.03	0.79	-0.03	0.78	-0.02	0.83
Openness to Exp.	**0.22**	**0.03**	-0.06	0.57	-0.04	0.67	0.05	0.59	-0.03	0.80	0.01	0.94	0.12	0.22

Table 4.11: Correlation between the big five personality traits and clicks/keys ratio of each activity for all users.

Clicks/Keys	**1**		**2**		**3**		**4**		**5**		**6**		**7**	
	Corr	p-val	Corr	p-val	Corr	p-val	Corr	p-val	Corr	p-val	Corr	p-val	Corr	p-val
Extraversion	-0.02	0.84	0.05	0.59	**0.20**	**0.05**	-0.04	0.67	0.06	0.53	-0.01	0.90	0.02	0.86
Agreeableness	0.06	0.57	0.11	0.28	0.06	0.53	0.00	0.97	0.09	0.36	0.08	0.46	0.02	0.85
Conscientiousness	-0.05	0.65	0.04	0.72	-0.02	0.85	0.01	0.90	0.02	0.81	0.12	0.26	0.07	0.47
Neuroticism	0.03	0.77	-0.10	0.30	-0.13	0.20	-0.10	0.33	-0.07	0.51	0.02	0.85	-0.11	0.27
Openness to Exp.	0.14	0.16	0.11	0.27	0.08	0.45	0.08	0.43	**0.25**	**0.01**	0.19	0.06	**0.21**	**0.03**

Table 4.12: Correlation between the big five personality traits and inter-key latency of each activity for all users.

Inter-key Latency	**1**		**2**		**3**		**4**		**5**		**6**		**7**	
	Corr	p-val	Corr	p-val	Corr	p-val	Corr	p-val	Corr	p-val	Corr	p-val	Corr	p-val
Extraversion	**-0.20**	**0.04**	0.02	0.86	-0.15	0.14	0.10	0.32	-0.14	0.17	-0.02	0.81	0.11	0.30
Agreeableness	-0.03	0.79	-0.18	0.07	-0.18	0.07	-0.03	0.80	**-0.27**	**0.01**	-0.07	0.48	-0.03	0.75
Conscientiousness	**-0.22**	**0.03**	-0.14	0.18	-0.08	0.43	-0.16	0.11	**-0.22**	**0.03**	**-0.24**	**0.01**	**-0.26**	**0.01**
Neuroticism	0.09	0.40	0.13	0.21	0.10	0.33	0.00	0.96	0.15	0.15	-0.02	0.84	0.01	0.94
Openness to Exp.	0.09	0.38	-0.06	0.56	0.07	0.46	-0.15	0.15	-0.10	0.31	-0.15	0.14	-0.15	0.13

Table 4.13: Correlation between the Holland code test personality traits and the duration of each activity for all users.

Duration	**1**		**2**		**3**		**4**		**5**		**6**		**7**	
	Corr	p-val	Corr	p-val	Corr	p-val	Corr	p-val	Corr	p-val	Corr	p-val	Corr	p-val
Realistic	0.06	0.53	0.12	0.23	0.03	0.77	0.11	0.29	0.12	0.25	-0.02	0.87	0.02	0.87
Investigative	0.10	0.35	-0.03	0.80	0.08	0.42	-0.10	0.32	0.03	0.79	-0.02	0.85	0.19	0.06
Artistic	0.02	0.86	-0.03	0.80	-0.02	0.88	-0.05	0.60	-0.05	0.60	0.00	1.00	0.05	0.63
Social	-0.19	0.06	0.00	0.99	-0.07	0.51	-0.07	0.51	**-0.27**	**0.01**	0.05	0.63	**-0.23**	**0.02**
Enterprising	0.02	0.81	**0.20**	**0.05**	0.17	0.09	0.09	0.36	0.08	0.41	-0.02	0.83	-0.18	0.07
Conventional	0.14	0.16	0.14	0.16	0.13	0.21	0.01	0.90	**0.20**	**0.04**	-0.01	0.94	-0.12	0.23

Table 4.14: Correlation between the Holland code test personality traits and number of keys of each activity for all users.

Keys	**1**		**2**		**3**		**4**		**5**		**6**		**7**	
	Corr	p-val	Corr	p-val	Corr	p-val	Corr	p-val	Corr	p-val	Corr	p-val	Corr	p-val
Realistic	0.16	0.12	0.16	0.12	0.11	0.30	0.08	0.44	0.17	0.09	-0.10	0.35	-0.10	0.34
Investigative	0.05	0.60	0.09	0.38	0.11	0.26	-0.17	0.10	0.11	0.27	0.01	0.90	**0.23**	**0.02**
Artistic	0.07	0.50	0.09	0.37	0.14	0.17	0.01	0.96	0.07	0.52	0.08	0.43	0.05	0.64
Social	-0.03	0.76	0.08	0.46	0.09	0.36	-0.06	0.56	-0.09	0.39	0.00	0.97	**-0.19**	**0.05**
Enterprising	0.15	0.14	0.10	0.32	**0.28**	**0.00**	**0.22**	**0.03**	0.19	0.06	-0.07	0.51	**-0.22**	**0.03**
Conventional	**0.34**	**0.00**	0.11	0.27	0.14	0.16	0.04	0.72	**0.23**	**0.02**	-0.01	0.96	-0.17	0.10

Table 4.15: Correlation between the Holland code test personality traits and number of clicks of each activity for all users.

Clicks	**1**		**2**		**3**		**4**		**5**		**6**		**7**	
	Corr	p-val	Corr	p-val	Corr	p-val	Corr	p-val	Corr	p-val	Corr	p-val	Corr	p-val
Realistic	0.07	0.52	0.03	0.80	-0.11	0.28	0.02	0.85	-0.03	0.79	-0.18	0.08	-0.15	0.15
Investigative	0.03	0.75	-0.04	0.70	0.08	0.42	-0.04	0.69	0.03	0.78	0.02	0.85	0.07	0.49
Artistic	-0.07	0.47	-0.06	0.55	-0.02	0.84	-0.08	0.41	-0.10	0.31	-0.14	0.18	-0.06	0.55
Social	**-0.23**	**0.02**	-0.10	0.34	-0.09	0.38	-0.01	0.89	**-0.27**	**0.01**	0.04	0.70	-0.08	0.44
Enterprising	0.07	0.48	0.06	0.55	0.12	0.24	0.00	1.00	-0.10	0.33	-0.08	0.46	-0.11	0.27
Conventional	0.17	0.09	0.11	0.28	0.10	0.34	0.00	0.96	0.08	0.42	-0.03	0.79	-0.11	0.30

Table 4.16: Correlation between the Holland code test personality traits and double-click latency of each activity for all users.

Double-click Latency	**1**		**2**		**3**		**4**		**5**		**6**		**7**	
	Corr	p-val	Corr	p-val	Corr	p-val	Corr	p-val	Corr	p-val	Corr	p-val	Corr	p-val
Realistic	-0.04	0.70	-0.04	0.69	0.03	0.77	0.19	0.06	0.05	0.59	0.01	0.93	-0.02	0.85
Investigative	0.00	0.99	-0.08	0.42	-0.05	0.60	0.04	0.68	0.03	0.76	-0.06	0.58	0.15	0.14
Artistic	-0.06	0.58	-0.09	0.36	-0.07	0.50	-0.06	0.57	-0.11	0.30	-0.10	0.34	-0.04	0.72
Social	-0.02	0.82	**-0.20**	**0.05**	-0.05	0.60	-0.05	0.64	-0.18	0.07	-0.05	0.62	0.04	0.71
Enterprising	0.15	0.13	-0.02	0.82	0.14	0.17	**0.20**	**0.05**	-0.10	0.33	-0.05	0.61	-0.12	0.23
Conventional	0.02	0.88	0.02	0.88	0.18	0.08	**0.31**	**0.00**	-0.07	0.48	0.07	0.50	-0.11	0.26

Table 4.17: Correlation between the Holland code test personality traits and clicks/keys ratio of each activity for all users.

Clicks/Keys	**1**		**2**		**3**		**4**		**5**		**6**		**7**	
	Corr	p-val	Corr	p-val	Corr	p-val	Corr	p-val	Corr	p-val	Corr	p-val	Corr	p-val
Realistic	-0.06	0.52	-0.08	0.41	**-0.21**	**0.04**	-0.08	0.46	-0.12	0.26	**-0.21**	**0.04**	**-0.21**	**0.03**
Investigative	-0.01	0.89	-0.08	0.43	0.09	0.35	0.02	0.82	0.03	0.74	0.03	0.77	0.04	0.69
Artistic	0.04	0.68	-0.01	0.95	0.04	0.67	-0.05	0.65	-0.07	0.49	-0.06	0.57	-0.06	0.58
Social	-0.09	0.40	0.01	0.95	-0.01	0.92	0.03	0.78	**-0.20**	**0.05**	-0.03	0.80	-0.03	0.74
Enterprising	0.09	0.37	0.04	0.69	0.08	0.45	0.05	0.62	-0.18	0.07	-0.01	0.92	-0.07	0.50
Conventional	0.05	0.60	-0.07	0.52	-0.03	0.75	-0.09	0.38	-0.11	0.27	-0.01	0.92	-0.13	0.19

Table 4.18: Correlation between the Holland code test personality traits and inter-key latency of each activity for all users.

Inter-key Latency	**1**		**2**		**3**		**4**		**5**		**6**		**7**	
	Corr	p-val	Corr	p-val	Corr	p-val	Corr	p-val	Corr	p-val	Corr	p-val	Corr	p-val
Realistic	0.08	0.43	0.02	0.82	0.10	0.35	0.13	0.22	-0.03	0.77	0.08	0.42	0.18	0.08
Investigative	0.13	0.20	-0.15	0.13	-0.07	0.47	0.17	0.09	0.01	0.95	0.02	0.82	0.03	0.77
Artistic	-0.01	0.89	-0.18	0.08	**-0.20**	**0.05**	0.12	0.26	-0.15	0.14	0.03	0.78	0.05	0.64
Social	-0.04	0.69	-0.14	0.17	-0.12	0.23	0.08	0.44	-0.10	0.33	-0.04	0.72	0.06	0.53
Enterprising	-0.04	0.71	-0.01	0.96	-0.04	0.71	0.03	0.73	-0.14	0.16	0.00	0.97	0.17	0.09
Conventional	-0.09	0.35	-0.05	0.64	-0.11	0.30	0.03	0.78	-0.12	0.25	-0.03	0.75	0.14	0.16

4.5.4 Correlation Between Personality Traits and Features for Each Activity

To further analyze the correlation between personality traits and interaction with input devices, we calculate the correlation between each personality trait and the feature vectors for each activity. As we noted earlier, each user has participated in 7 different activities with different risk levels. The description of these activities is shown in Table 3.3. This analysis may help us get insights about the trends in the personality traits in correlation with the features for more or less malicious activities. To do so, for each trait in the big five personality test, we get the scores vector and calculate the Pearson correlation coefficient with the feature values vector for all users. For example, we get the extraversion score for all users as one feature vector and the duration of the first activity for all users as another feature vector and calculate the Pearson correlation coefficient between them. We do the same process for calculating the correlation between features and traits in the Holland code test. The results of these tests are shown in Tables 4.7 to 4.18. The higher correlation values are shown in bold.

Our observations in the aforementioned tables show that there are several correlated pairs of features and traits for different activities. From Table 4.7, we can see that openness to experience is directly correlated with duration for activities 6 and 7 which are more malicious, and inversely correlated with duration for activities 2 and 3 which are less malicious. So, we may conclude that generally, people with higher openness to experience score, tend to spend more time for completing the malicious activities. Furthermore, based on Table 4.8, we can see that people with higher openness to experience, use the keyboard more for fulfilling the most critical activities, because there is a correlation between this trait and number of keys in activities 6 and 7 which are considered as malicious. The same argument can be made based on Table 4.9 for the number of clicks as there is a correlation between openness to experience and the

number of clicks for activities 6 and 7. So, we may conclude that people with higher openness to experience score make more use of the input devices during malicious activities. In Table 4.10 we can see that agreeableness is directly correlated with activity 7 which is a malicious activity. Table 4.11 shows correlation between openness to experience and activities 5 and 7 which are malicious activities, so, the more openness to experience leads to higher clicks/keys ratio for malicious activities showing higher usage of the mouse. Table 4.12 shows multiple inverse correlations, especially between conscientiousness and activities 1, 5, 6, and 7. As a result, we can conclude that people with higher conscientiousness level, are faster in working with the input devices, especially for more malicious activities.

To analyze the results based on the Holland Code Test, we observe Tables 4.13 to 4.18. It is observable that the more social people, spend less time in activities 5 and 7 which are both malicious activities. However, there is no significant correlation between the typing latency of social people and the activities as shown in Table 4.18. There are multiple correlations observable in Table 4.14. More enterprising people use keyboard more in activities 3 and 4 and less in activity 7. Also, more conventional people use the keyboard more in activities 1 and 5. Table 4.15 shows negative correlation between social trait and number of clicks in activities 1 and 5. Table 4.16 shows correlation between enterprising and conventional traits and double-click latency in activity 4. Table 4.17 shows negative correlation between realistic trait and activities 3, 6, and 7. Finally, the only notable correlation in Table 4.18 is negative correlation between artistic trait and activity 3.

4.6 Summary

We extracted multiple features from the keyboard and mouse usage of 100 users. Then, we used the results from two separate questionnaires to perceive the personality traits of the users. We studied the correlation between the personality traits with keystroke and mouse dynamics and analyzed the patterns and trends that exist between them. Our analysis shows that there is a possibility to infer different trends and patterns from the data to the extent and situations where the personality tests and our interpretations are valid. We showed how different personality characteristics of people may be correlated with their typing behavior and interaction with a computer system. This study provides a benchmark for further analysis of correlation between personality types and interaction with computer through mouse and keyboard. Although our analysis shows correlations between the personality traits and user interactions with input devices, the results of our research are based on the mathematical inferences and may not be sufficient to make psychological conclusions. So, psychological analysis of the patterns and the correlations is beyond the scope of this study.

Chapter 5

Conclusion

We studied different aspects of behavioral biometrics and how they can be used in security-related applications. Through our study on predicting the future gait signal of humans using ubiquitous motion sensors, we showed that it is possible to use behavioral biometrics in forensics applications. Our prediction of the future signal can open the path for several future studies in security and forensics applications.

Furthermore, we studied and analyzed two large datasets of different sensors captured from several users during their interaction with a computer system. We analyzed and used the data to be used in different applications. Among those, we chose to study how audio data that has been recorded from typing sessions can be used in predicting possible threats and securing the system. Our study showed that it is possible to use audio to predict the activity of the users in real-time and find out if there is any threat against the system by a particular user.

Finally, we extended our research to find out whether the personality traits and psychological characteristics of the users can give us insights into their behavior in working with a computer system. Our findings showed that there are different corre-

lations between the personality of the users and how they interact with input devices like keyboard and mouse, which are among the main interfaces for interaction with computers.

Our studies have contributed to the research by providing several insights on how behavioral biometrics apply to everyday security challenges. We have unfolded several research directions for the future and believe there is more left for exploration by us and future scientists.

These are some of the limitations of our work that we recommend to be studied further and addressed in the future research:

- In our future gait signal predictions, we evaluate our work by predicting 200 samples in the future. While we narrow our problem to near-future prediction and this might be enough for most applications in the area, evaluating the model with more samples and tuning it to perform well for further predictions in the future is left for future research.

- For predicting the future gait signal, we assume that we have already identified the users and trained a separate model for each individual. The possibility of using a single model for all the users, taking their height, gender, etc. into account as model parameters can be studied in the future.

- In our evaluations, we use a mixture of already predicted samples and actual samples from the past to predict the future signal. Evaluating the performance of the model using only actual past samples and only predicted past samples is left to be explored.

- We use a dataset with a limited number of users (9 users) to conduct our research on gait signal prediction. We also use a limited number of activities (walking,

biking, stairs up, stairs down). The applicability of our method and the extension of our model to support more activities with variations in speed is subject to further research.

- We use the accelerometer captured from mobile phones in users' pockets for future gait signal prediction. The use of different sensors that are available in today's smartphones, different mobile phone placements, and other devices like smartwatches can be studied in the future.

- For recovering keystrokes using their audio, we use a limited recording configuration, placing the microphones near the keyboard. Using high-end microphones that can capture the sound from a higher distance and further placement of microphones can be explored in future research.

- We have separate benign and adversarial datasets. While there are participants that have participated in both data collections, collecting a unified dataset with both adversarial and benign activities from the same participants can be conducted in future research. Furthermore, our benign dataset does not have keystroke audio recordings and we evaluate our text recovery method only on the adversarial dataset. For evaluating our activity prediction model, we get the exact typed text from the benign dataset and do not recover the text using audio. This can be addressed in future research.

- In our dataset, we use a single keyboard and mouse for all participants and develop our model with specific input devices. Extending the model to support different input devices is subject to further study.

- While we capture the sound with a microphone, capturing different waves and signals that are in the inaudible frequency range using different high-end devices can be studied in the future. This way, we may be able to apply the model to the

silent keyboards that are emerging and make acoustic-based methods including ours ineffective.

- While our activity detection model can classify the activities regardless of the user, our text recovery method is dependent on the identity of the user and we train separate models for each user. The possibility of using a unified model for text recovery for all users is subject to future research.

- We use a supervised model for text recovery from keystroke acoustics. However, past research has shown that it is possible to use unsupervised models for this application [27]. The applicability of those models to our dataset and their performance on our problem can be studied in the future.

- In our model, we only use a limited number of keys (alphanumerical, special characters, space, and enter) for text recovery and classification. While we showed that it is possible to classify the activity and measure the threat level using those keys with notable performance, the use of other keys on the keyboard (like tab which is used to auto-complete commands in the terminal) and their effects on the model performance is left for the future research. Also, studying the detection of keystroke combinations which is partially addressed in [33] and its effects on our model performance can be conducted in the future.

- We use a dictionary-based method for recovering meaningful words using keystrokes audio. Whether random sequences of keystrokes can contribute to activity detection is subject to further research.

- We limit our adversarial activities to 7 different categories. Collecting a more complex dataset with other adversarial activities and evaluating the performance of our model based on a higher variation of adversarial activities can be done in the future.

- We exclude analysis based on demographics in our research, however, demographic information is available in our dataset and different insights may be retrieved from them by further study in the future.

- We use a limited number of features for studying the correlation between the usage of input devices and personality. Using a wider collection of features (for example, features related to specific keystrokes, keystroke combinations, mouse movement patterns, etc.) and their fusion can be studied in the future.

- Our study on personalities and their correlation with features related to input devices is a foundational work that opens the path for future studies in this area. It is too early to deduce strong conclusions based on them or utilize them for adversarial activity detection.

- Psychological analysis of the outcomes of the correlations between personality traits and input device features can be further studied in joint work with experts in the psychology field.

- While our problem is not to detect personalities based on keyboard and mouse features and we only focus on finding correlations, building and tuning a classification model can be an interesting problem for research in the future.

While we conduct our research in a lab setting, careless application of artificial intelligence methods in the real world, including ours, may result in different unethical outcomes. It is also important that AI researchers be aware of the problems of stereotyping and labeling people as criminal, malicious, adversary, etc. based on experimental research. So, we need to state that even though our research in gait signal prediction opens the path for many applications in forensics research, applying it to real-world problems and labeling people as criminals based on it is subject to extensive research

and experiments in the future. Similarly, our personality analysis and adversarial activity detection using keystrokes have been studied with limited data and considerations. Our limited dataset only includes a limited group of participants and may be subject to different biases. Generalizing the results of our research to be used with a greater and more diverse population needs further studies and experiments on larger and more diverse data. Furthermore, errors are inevitable in AI applications, including ours and they should be expected and considered carefully. In our keystrokes audio recording scenario, we assume that we can record the user without their knowledge and use the recordings in surveillance applications. However, the application of this scenario in the real world, taking into account the user's privacy, legal, and ethical concerns needs careful consideration.

Related Publications

- (Conference paper) A. Fallahi and V. V. Phoha, "Adversarial Activity Detection Using Keystroke Acoustics," 26th European Symposium on Research in Computer Security (ESORICS) 2021.
- (Scientific meeting) A. Fallahi, D. Shukla, V. V. Phoha, "Looking into Your Future: A Continuous Human Gait Prediction for Near Future ," in *American Academy of Forensic Sciences 2021 Annual Scientific Meeting.*
- (Under revision) A. Fallahi and V. V. Phoha, "On the Correlation of Keystroke and Mouse Dynamics With Personality"

Bibliography

[1] TechTarget Contributor. *What is behavioral biometrics? - definition from whatis.com*. Mar. 2015. URL: https://www.techtarget.com/whatis/definition/behavioral-biometrics.

[2] National Institute of Standards and Technology. *Malicious cyber activity - glossary*. Apr. 2015. URL: https://csrc.nist.gov/glossary/term/malicious_cyber_activity.

[3] Law Insider. *Malicious activity definition*. URL: https://www.lawinsider.com/dictionary/malicious-activity.

[4] Quynh Dang. *Recommendation for Applications Using Approved Hash Algorithms*. en. Aug. 2012. URL: https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=911479.

[5] Yu Guan and Thomas Plötz. "Ensembles of Deep LSTM Learners for Activity Recognition Using Wearables". In: *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 1.2 (June 2017), 11:1–11:28. ISSN: 2474-9567. DOI: 10.1145/3090076. URL: http://doi.acm.org/10.1145/3090076.

[6] Bryan Minor, Janardhan Rao Doppa, and Diane J. Cook. "Data-Driven Activity Prediction: Algorithms, Evaluation Methodology, and Applications". In: *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '15. Sydney, NSW, Australia: ACM, 2015, pp. 805–814.

ISBN: 978-1-4503-3664-2. DOI: 10.1145/2783258.2783408. URL: http://doi.acm.org/10.1145/2783258.2783408.

[7] Monika Saini and Anup Kumar Kapoor. "Biometrics in Forensic Identification: Applications and Challenges". In: *Journal of Forensic Medicine* 1 (2016), pp. 1–6.

[8] Yuqi Zhang et al. "A comprehensive study on gait biometrics using a joint CNN-based method". In: *Pattern Recognition* 93 (2019), pp. 228–236. ISSN: 0031-3203. DOI: https://doi.org/10.1016/j.patcog.2019.04.023. URL: https://www.sciencedirect.com/science/article/pii/S0031320319301694.

[9] J. R. Kwapisz, G. M. Weiss, and S. A. Moore. "Cell phone-based biometric identification". In: *2010 Fourth IEEE International Conference on Biometrics: Theory, Applications and Systems (BTAS)*. Sept. 2010, pp. 1–7. DOI: 10.1109/BTAS.2010.5634532.

[10] Sepp Hochreiter and Jürgen Schmidhuber. "Long Short-term Memory". In: *Neural computation* 9 (Dec. 1997), pp. 1735–80. DOI: 10.1162/neco.1997.9.8.1735.

[11] R. Kumar et al. "Continuous user authentication via unlabeled phone movement patterns". In: *2017 IEEE International Joint Conference on Biometrics (IJCB)*. Oct. 2017, pp. 177–184. DOI: 10.1109/BTAS.2017.8272696.

[12] Upal Mahbub and Rama Chellappa. "PATH: Person Authentication using Trace Histories". In: *CoRR* abs/1610.07935 (2016). arXiv: 1610.07935. URL: http://arxiv.org/abs/1610.07935.

[13] Rajesh Kumar, Vir V Phoha, and Abdul Serwadda. "Continuous authentication of smartphone users by fusing typing, swiping, and phone movement patterns". In: *2016 IEEE 8th International Conference on Biometrics Theory, Applications and Systems (BTAS)*. IEEE. 2016, pp. 1–8.

[14] Rahul Murmuria et al. "Continuous authentication on mobile devices using power consumption, touch gestures and physical movement of users". In: *International Symposium on Recent Advances in Intrusion Detection*. Springer. 2015, pp. 405–424.

[15] Cheng Bo et al. "Continuous user identification via touch and movement behavioral biometrics". In: *2014 IEEE 33rd International Performance Computing and Communications Conference (IPCCC)*. IEEE. 2014, pp. 1–8.

[16] Zahid Akhtar et al. "Multimodal smartphone user authentication using touchstroke, phone-movement and face patterns". In: *2017 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*. IEEE. 2017, pp. 1368–1372.

[17] Y. Cao et al. "Recognize Human Activities from Partially Observed Videos". In: *2013 IEEE Conference on Computer Vision and Pattern Recognition*. June 2013, pp. 2658–2665. DOI: 10.1109/CVPR.2013.343.

[18] Allan Stisen et al. "Smart Devices Are Different: Assessing and MitigatingMobile Sensing Heterogeneities for Activity Recognition". In: *Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems*. SenSys '15. Seoul, South Korea: ACM, 2015, pp. 127–140. ISBN: 978-1-4503-3631-4. DOI: 10.1145/2809695.2809718. URL: http://doi.acm.org/10.1145/2809695.2809718.

[19] Dheeru Dua and Casey Graff. *UCI Machine Learning Repository*. 2017. URL: http://archive.ics.uci.edu/ml.

[20] SHI Xingjian et al. "Convolutional LSTM network: A machine learning approach for precipitation nowcasting". In: *Advances in neural information processing systems*. 2015, pp. 802–810.

[21] David MQ Nelson, Adriano CM Pereira, and Renato A de Oliveira. "Stock market's price movement prediction with LSTM neural networks". In: *2017 Inter-*

national Joint Conference on Neural Networks (IJCNN). IEEE. 2017, pp. 1419–1426.

[22] Alex Graves, Navdeep Jaitly, and Abdel-rahman Mohamed. "Hybrid speech recognition with deep bidirectional LSTM". In: *2013 IEEE workshop on automatic speech recognition and understanding*. IEEE. 2013, pp. 273–278.

[23] Alex Graves and Jürgen Schmidhuber. "Offline handwriting recognition with multidimensional recurrent neural networks". In: *Advances in neural information processing systems*. 2009, pp. 545–552.

[24] Urvashi Khandelwal et al. "Sharp Nearby, Fuzzy Far Away: How Neural Language Models Use Context". In: *CoRR* abs/1805.04623 (2018). arXiv: 1805.04623. URL: http://arxiv.org/abs/1805.04623.

[25] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2017. arXiv: 1412.6980 [cs.LG].

[26] D. Asonov and R. Agrawal. "Keyboard acoustic emanations". In: *IEEE Symposium on Security and Privacy, 2004. Proceedings. 2004*. 2004, pp. 3–11. DOI: 10.1109/SECPRI.2004.1301311.

[27] Li Zhuang, Feng Zhou, and J. D. Tygar. "Keyboard Acoustic Emanations Revisited". In: 13.1 (Nov. 2009). ISSN: 1094-9224. DOI: 10.1145/1609956.1609959. URL: https://doi.org/10.1145/1609956.1609959.

[28] Yigael Berger, Avishai Wool, and Arie Yeredor. "Dictionary Attacks Using Keyboard Acoustic Emanations". In: *Proceedings of the 13th ACM Conference on Computer and Communications Security*. CCS '06. Alexandria, Virginia, USA: Association for Computing Machinery, 2006, pp. 245–254. ISBN: 1595935185. DOI: 10.1145/1180405.1180436. URL: https://doi.org/10.1145/1180405.1180436.

[29] Jian Liu et al. "Snooping Keystrokes with Mm-Level Audio Ranging on a Single Phone". In: *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking*. MobiCom '15. Paris, France: Association for Computing Machinery, 2015, pp. 142–154. ISBN: 9781450336192. DOI: 10.1145/2789168.2790122. URL: https://doi.org/10.1145/2789168.2790122.

[30] J. Roth et al. "Biometric authentication via keystroke sound". In: *2013 International Conference on Biometrics (ICB)*. 2013, pp. 1–8. DOI: 10.1109/ICB.2013.6613015.

[31] J. Roth et al. "Investigating the Discriminative Power of Keystroke Sound". In: *IEEE Transactions on Information Forensics and Security* 10.2 (2015), pp. 333–345. DOI: 10.1109/TIFS.2014.2374424.

[32] Alberto Compagno et al. *Don't Skype & Type! Acoustic Eavesdropping in Voice-Over-IP*. 2017. arXiv: 1609.09359 [cs.CR].

[33] J. Wang et al. "Accurate Combined Keystrokes Detection Using Acoustic Signals". In: *2016 12th International Conference on Mobile Ad-Hoc and Sensor Networks (MSN)*. 2016, pp. 9–14. DOI: 10.1109/MSN.2016.010.

[34] A. K. Belman, S. Sridhara, and V. V. Phoha. "Classification of Threat Level in Typing Activity Through Keystroke Dynamics". In: *2020 International Conference on Artificial Intelligence and Signal Processing (AISP)*. 2020, pp. 1–6. DOI: 10.1109/AISP48273.2020.9073079.

[35] Amith K. Belman et al. *Insights from BB-MAS – A Large Dataset for Typing, Gait and Swipes of the Same Person on Desktop, Tablet and Phone*. 2019. arXiv: 1912.02736 [cs.HC].

[36] Benjamin Barras. "SoX : Sound eXchange". In: (Jan. 2012).

[37] Audacity Team. *Audacity*. Version 3.0.2. Feb. 19, 2010. URL: https://www.audacityteam.org/.

[38] European Broadcasting Union. *R 128 - Loudness normalisation and permitted maximum level of audio signals*. Aug. 2020. URL: https://tech.ebu.ch/docs/r/r128.pdf.

[39] Todor Ganchev, Nikos Fakotakis, and George Kokkinakis. "Comparative evaluation of various MFCC implementations on the speaker verification task". In: *in Proc. of the SPECOM-2005*. 2005, pp. 191–194.

[40] S. Lloyd. "Least squares quantization in PCM". In: *IEEE Transactions on Information Theory* 28.2 (1982), pp. 129–137. DOI: 10.1109/TIT.1982.1056489.

[41] Richard O. Duda and Peter E. Hart. "Pattern classification and scene analysis". In: *A Wiley-Interscience publication*. 1973.

[42] Kevin Atkinson. *SCOWL (Spell Checker Oriented Word Lists) and Friends*. 2020. URL: http://wordlist.aspell.net/.

[43] Binh Nguyen. *Linux Dictionary*. The Linux Documentation Project. URL: https://tldp.org/LDP/Linux-Dictionary/html/index.html.

[44] Peter Norvig. *How to Write a Spelling Corrector*. Aug. 2016. URL: http://norvig.com/spell-correct.html.

[45] F. M. Shakiba and M. Zhou. "Novel Analog Implementation of a Hyperbolic Tangent Neuron in Artificial Neural Networks". In: *IEEE Transactions on Industrial Electronics* (2020), pp. 1–1. DOI: 10.1109/TIE.2020.3034856.

[46] K. T. Baghaei and S. Rahimi. "Sepsis Prediction: An Attention-Based Interpretable Approach". In: *2019 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*. 2019, pp. 1–6. DOI: 10.1109/FUZZ-IEEE.2019.8858808.

[47] Mohan Kopuru, Shahram Rahimi, and Kourosh Teimouri Baghaei. "Recent Approaches in Prognostics: State of the Art". In: Aug. 2019.

[48] Dario Pavllo et al. *3D human pose estimation in video with temporal convolutions and semi-supervised training*. 2019. arXiv: 1811.11742 [cs.CV].

[49] M. A. Haque, N. Z. Khan, and G. Khatoon. "Authentication through keystrokes: What you type and how you type". In: *2015 IEEE International Conference on Research in Computational Intelligence and Communication Networks (ICRCICN)*. 2015, pp. 257–261. DOI: 10.1109/ICRCICN.2015.7434246.

[50] Dwijen Rudrapal, Smita Das, and Swapan Debbarma. "Improvisation of Biometrics Authentication and Identification through Keystrokes Pattern Analysis". In: *Distributed Computing and Internet Technology*. Ed. by Raja Natarajan. Cham: Springer International Publishing, 2014, pp. 287–292. ISBN: 978-3-319-04483-5.

[51] Dharmendra Singh. "Presskey- a keystrokes dynamics based authentication system". In: *International Journal of Advanced Research in Computer Science* 8.5 (2017), pp. 2665–2670. ISSN: 0976-5697. DOI: 10.26483/ijarcs.v8i5.3956. URL: http://www.ijarcs.info/index.php/Ijarcs/article/view/3956.

[52] A. Peacock, Xian Ke, and M. Wilkerson. "Typing patterns: a key to user identification". In: *IEEE Security Privacy* 2.5 (2004), pp. 40–47. DOI: 10.1109/MSP.2004.89.

[53] M. Rybnik, M. Tabedzki, and K. Saeed. "A Keystroke Dynamics Based System for User Identification". In: *2008 7th Computer Information Systems and Industrial Management Applications*. 2008, pp. 225–230. DOI: 10.1109/CISIM.2008.8.

[54] J.L. Holland. *Making Vocational Choices: A Theory of Vocational Personalities and Work Environments*. Prentice-Hall series in counseling and human development. Prentice-Hall, 1985. ISBN: 9780135475973. URL: https://books.google.com/books?id=8QxBAAAAMAAJ.

[55] Lewis R Goldberg. "The development of markers for the Big-Five factor structure." In: *Psychological assessment* 4.1 (1992), p. 26.

[56] *What are Holland Codes? - Learn More About Career Test Theory*. 2021. URL: https://www.123test.com/holland-codes-career-tests/.

[57] Annabelle G.Y. Lim. *The big five personality traits*. June 2020. URL: https://www.simplypsychology.org/big-five-personality.html.

[58] A.F.M. Nazmul Haque Nahin et al. "Identifying emotion by keystroke dynamics and text pattern analysis". In: *Behaviour & Information Technology* 33.9 (2014), pp. 987–996. DOI: 10.1080/0144929X.2014.907343. eprint: https://doi.org/10.1080/0144929X.2014.907343. URL: https://doi.org/10.1080/0144929X.2014.907343.

[59] Rinky Solanki and P. Shukla. "Estimation of the User's Emotional State by Keystroke Dynamics". In: *International Journal of Computer Applications* 94 (2014), pp. 21–23.

[60] Ihor Tereikovskyi et al. "User Keystroke Authentication and Recognition of Emotions Based on Convolutional Neural Network". In: *Advances in Artificial Systems for Medicine and Education III*. Ed. by Zhengbing Hu, Sergey Petoukhov, and Matthew He. Cham: Springer International Publishing, 2020, pp. 283–292. ISBN: 978-3-030-39162-1.

[61] Abeer Buker, Alessandro Vinciarelli, and Giorgio Roffo. "Type Like a Man! Inferring Gender From Keystroke Dynamics in Live-Chats". In: *IEEE Intelligent Systems* 34 (Dec. 2019). DOI: 10.1109/MIS.2019.2948514.

[62] Avar Pentel. "Predicting Age and Gender by Keystroke Dynamics and Mouse Patterns". In: *Adjunct Publication of the 25th Conference on User Modeling, Adaptation and Personalization*. UMAP '17. Bratislava, Slovakia: Association for Computing Machinery, 2017, pp. 381–385. ISBN: 9781450350679. DOI: 10.1145/3099023.3099105. URL: https://doi.org/10.1145/3099023.3099105.

[63] Avar Pentel. "Predicting User Age by Keystroke Dynamics". In: *Artificial Intelligence and Algorithms in Intelligent Systems*. Ed. by Radek Silhavy. Cham: Springer International Publishing, 2019, pp. 336–343. ISBN: 978-3-319-91189-2.

[64] A. Pentel. "High precision handedness detection based on short input keystroke dynamics". In: *2017 8th International Conference on Information, Intelligence, Systems Applications (IISA)*. 2017, pp. 1–5. DOI: 10.1109/IISA.2017.8316380.

[65] M. Da Costa-Abreu. "Using keystroke dynamics for gender identification in social network environment". English. In: *IET Conference Proceedings* (Jan. 2011), 27–27(1). URL: https://digital-library.theiet.org/content/conferences/10.1049/ic.2011.0124.

[66] Soumen Roy, Utpal Roy, and D. Sinha. "The probability of predicting personality traits by the way user types on touch screen". In: *Innovations in Systems and Software Engineering* 15 (Mar. 2019). DOI: 10.1007/s11334-018-0317-6.

[67] Tzafilkou Katerina and Protogeros Nicolaos. "Mouse behavioral patterns and keystroke dynamics in End-User Development: What can they tell us about users' behavioral attributes?" In: *Computers in Human Behavior* 83 (2018), pp. 288–305. ISSN: 0747-5632. DOI: https://doi.org/10.1016/j.chb.2018.02.012. URL: https://www.sciencedirect.com/science/article/pii/S0747563218300700.

[68] Robert F. Tate. "Correlation Between a Discrete and a Continuous Variable. Point-Biserial Correlation". In: *The Annals of Mathematical Statistics* 25.3 (1954), pp. 603–607. ISSN: 00034851. URL: http://www.jstor.org/stable/2236844.

VITA

Amin Fallahi is a Ph.D. candidate in Computer & Information Science and Engineering (CISE) at Syracuse University. His current research at Prof. Vir Phoha's lab focuses on human biometrics and its security applications.

He earned his B.Sc. in Software Engineering at Iran University of Science and Technology under the supervision of Prof. Mohsen Sharifi at IUST Distributed Systems Lab where he did research in cloud computing, virtualization, and distributed systems.