Keystroke Dynamics for User Authentication Using Deep Multilayer Perceptron

Alvin Andrean

A capstone project submitted in fulfilment of the
requirements for the award of the degree of
Master of Science in Data Science and Business Analytics

Asia Pacific University of Technology & Innovation (APU)
School of Computing and Technology

OCTOBER 2017

ABSTRACT

User authentication is an important factor to protect digital service and prevent malicious users from gaining access to system. As Single Factor Authentication (SFA) is less secure, organizations started to utilize Multi Factor Authentication (MFA) to provide more reliable protection by using two or more identification measures. Keystroke dynamics is a behavioural biometric, which analyses users typing rhythm to identify the legitimacy of the subject accessing the system. Keystroke dynamics that has a low implementation cost, does not require additional hardware in the authentication process, since the collection of typing data is relatively simple as it does not require additional effort from the user. Based on the survey, deep learning implementation in keystroke dynamics are limited while compared with other classification methods. In addendum, essential elements such as features, datasets, and evaluation criteria were investigated. The most significant feature that affected keystroke dynamics performance had yet to be explored and CMU benchmark dataset was used more frequently while compared with other openly available dataset. Equal-error rate (EER) is the most widely used error metric to assess the performance of keystroke dynamics classifier. This study aimed to propose deep learning model using Multilayer Perceptron (MLP) in keystroke dynamics for user authentication on CMU benchmark dataset. The user typing rhythm from 51 subjects collected based on the static password (.tie5Roanl) typed 400 times over 8 sessions and 50 repetitions per session. The MLP achieved optimum EER of 4.45% compared to original benchmark classifiers such as 9.6% (scaled Manhattan), 9.96% (Mahalanobis Nearest Neighbor), 10.22% (Outlier Count), 10.25%, and 16.14% (Neural Network Auto-Assoc). Further testing were performed on other published datasets to evaluate the performance of the classifier and EER of 5.5%, 10%, and 5% were achieved.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ANN	Artificial Neural Network
ATM	Automatic Teller Machine
CMU	Carnegie Mellon University
DBN	Deep Belief Nets
DD	Down-Down
EDA	Exploratory Data Analysis
EER	Equal Error Rate
FAR	False Acceptance Rate
FNR	False Negative Rate
FPR	False Positive Rate
FRR	False Rejection Rate
GMM	Gaussian Mixture Model
GRU	Gated Recurrent Unit
GUI	Graphical User Interface
H	Hold
ID	Identifier
KDD	Knowledge Discovery in Databases
k-NN	k-Nearest Neighbor
LSTM	Long Short-Term Memory
MAE	Mean Absolute Error
MCC	Matthews Correlation Coefficient
MFA	Multi-Factor Authentication
MLP	Multi-Layer Perceptron
MSE	Mean Squared Error
PIN	Personal Identification Number
PRC	Precision-Recall Curve
ReLU	Rectified Linear Unit
RMSE	Root Mean Squared Error
ROC	Receiver Operating Characteristic
SFA	Single-Factor Authentication
SGD	Stochastic Gradient Descent

SVM	Support Vector Machine
UD	Up-Down
UU	Up-Up

CHAPTER 1

INTRODUCTION

1.1. Introduction

In this chapter, the background of the study for keystroke dynamics will be introduced. This includes general information about user authentication and keystroke dynamics. Next, the problem statement which will be solved in the study will be elaborated. Based on the problem formulation, the aim, objectives, and research questions of the study will be designed as a basis of what needs to be done and achieved by the end of the study. In addition, the significance of the study will be explained in order to understand the importance and relevance of keystroke dynamics research for user authentication. Furthermore, the scope of the study needs to be considered as constraints and limitations of the study. It shows how the study is narrowed down into a list of criteria which need to be focused in the study. Besides, the research methodology overview of the study will be illustrated in order to develop ideas of how the study will be conducted from the start until the end. Finally, the structure of the study will discuss the key topics covered in the following chapters in the study.

1.2. Background of the Study

In the information technology era where the amount of data keeps on increasing, data privacy is a concern for every individual. It causes the necessity to prevent the disclosure of private information which exists in electronic devices. User authentication is one of the available methods to protect user's data in a way that only responsible individual is able to access the information. Nowadays, using password alone or Single Factor Authentication (SFA) to authenticate a user is getting less secure as attackers are able to decrypt the password using brute force technique or password cracker. Thus, an efficient way to improve user authentication is needed to prevent data breaches.

There have been various measures established to improve the account security. Firstly, a knowledge-based authentication which uses personalised question in addition to a password. This method is useful unless the attacker knows enough information about the target user. Secondly, Multi-Factor Authentication (MFA) which uses two or more identification measures from the user. For instance, ATM (Automated Teller Machine) requires a user to insert the bank card to the machine followed by Personal Identification Number (PIN) to perform a transaction. Another instance of MFA is biometrics authentication which implements user's physiological or behavioural biometrics to replace or enhance the user authentication process. For instance, fingerprint and retina scan are categorized under physiological biometrics whereas keystroke dynamics is categorized under behavioural biometrics.

Keystroke dynamics is a user authentication method which uses user's typing rhythm to allow access into the system. It is an emerging field of interest for security especially in user authentication due to its advantages. Firstly, keystroke dynamics has a low implementation cost as no additional hardware is required in the authentication process. Secondly, it has easier implementation compared to other biometric authentication methods because the collection of typing data is relatively easy as it does not require special permission from the user. Application of keystroke dynamics has become popular due to the help of various classification algorithms such as statistical and machine learning.

Machine learning has been implemented in the cybersecurity field, including network anomaly detection, malware analysis, insider threat, and user authentication. Machine learning methods can increase a system's classification capability to detect outlier efficiently. For instance, machine learning algorithm can be applied in user authentication system for impostor detection. By learning the characteristic belonged of a genuine user, it will be able to classify different users and determine whether access to the account can be granted or not. Another branch of machine learning is deep learning, which represents hierarchical learning of non-linear features with the purpose of extracting dependencies between them. As these features may be complex and difficult to learn with normal machine learning methods, deep learning can help to learn high-level abstract ideas from low-level ones. Then, these abstractions can be separated in order to find features which can be used to improve the classification performance (Bengio, Courville & Vincent, 2012). In this study, evaluation of recent classification approaches in keystroke dynamics will be discussed. Also, a classification model based on deep learning will be implemented.

1.3. Problem Statement

Integrating enhanced security measures into an existing user authentication system is necessary to increase the level of security in user accounts while maintaining the user experience (Zurkus, 2016). Most of the authentication system is still using SFA (e.g. password) to provide access to user account due to its simplicity in the authentication process. As Single Factor Authentication (SFA) is less secure, organizations started to utilize Multi Factor Authentication (MFA) to provide more reliable protection by using two or more identification measures such as a combination of knowledge-based, object-based, and biometric-based authentication properties. Keystroke dynamics is a behavioural biometric, which analyses users typing rhythm to identify the legitimacy of the subject accessing the system.

Previous studies in keystroke dynamics involve building a classifier based on multiple approaches such as statistical approach (Monaco & Tappert, 2016; Roy et al., 2016; Killourhy, 2012; Messerman et al., 2011; and Teh et al., 2007) but had problems in handling non-linear keystroke data; machine learning approach (Ho & Kang, 2017; Maheshwary & Pudi, 2017; Darabseh & Namin, 2016; Pisani et al., 2015; and Giot, Abed & Rosenberger, 2009) but might require higher memory size and more expensive computational cost; and hybrid models (Mohabeer & Soyjaudah, 2015; Nisha & Kumar, 2014; Monaco et al., 2013; Bharadi. Shah & Ambardekar, 2011; and Azevedo, Cavalcanti & Filbo, 2007) with increasing complexity in model development. Also, there is limited research relating to the application of deep learning in keystroke dynamics.

The main challenge to be addressed before implementing keystroke dynamics in user authentication is to develop a classification model with high accuracy to differentiate typing pattern between genuine user and impostor. Various classification models have been developed however maximum rate of accuracy has not been achieved (Ali et al., 2017). (Zhong & Deng, 2015) suggested that more research needs to be done to develop new classification methods. Thus, further research is required to find out the best model to be used in keystroke dynamics for user authentication.

1.4. Aim and Objectives

The aim of this study is to propose deep learning model in keystroke dynamics for user authentication.

In order to achieve the aim, the objectives of the research are as follows:

1. To investigate the areas of keystroke dynamics specific to keystroke dataset, features, and classification approaches.
2. To propose an approach for keystroke dynamics user authentication using static password text.
3. To establish a classification model for keystroke dynamics by using deep learning approach.
4. To compare and evaluate the performance of the model in classifying genuine user and impostor.

1.5. Research Question

For the purpose of this study, the following questions are addressed:

1. What are the commonly used dataset, features, and classifiers in keystroke dynamics? (Objective 1)

   The commonly used dataset, features, and classifiers in keystroke dynamics will be investigated from previous studies in the domain. This information will support the researcher decision in choosing the dataset and features in this study.
2. What are the steps required in user authentication by using keystroke dynamics with static password text? (Objective 2)

   The researcher will learn the general user authentication process and the important elements in keystroke dynamics in order to propose the approach for user authentication by using keystroke dynamics.
3. How can the deep learning model be configured to fit keystroke dynamics problem? (Objective 3)

   The researcher will conduct a literature review on deep learning and perform trial and error in configuring the model to find out the best parameter for keystroke dynamics problem.

4. What are the criteria used to evaluate the performance keystroke dynamics classifier? (Objective 4)

   Previous related works will be reviewed in order to learn which criteria are frequently used to evaluate the performance of the classifier in keystroke dynamics.
5. What is the classifier performance achieved by previous studies in keystroke dynamics? (Objective 4)

   Besides the performance criteria, the current best performance classifier will be noted and compared with the classifier implemented in this study.

1.6. Significance of the Study

The project to research deep learning application in keystroke dynamics areas is important to provide a better understanding of the area and to solve the problems listed in Section 1.3. The outcome of the project is to successfully deliver a deep learning model in keystroke dynamics for user authentication.

As keystroke dynamics is relatively new and still involves many challenges (Zhong & Deng, 2015), the research in keystroke dynamics area can contribute to the improvement of knowledge since deep learning application in keystroke dynamics is limited. Besides, it can potentially increase the user awareness and understanding regarding the biometrics authentication and tackle security issues specifically in access control and data privacy.

Keystroke dynamics opens the opportunity for individual and organization to cost-effectively adopt the biometrics authentication method to protect their data. Besides, it also provides better security compared to traditional single-factor authentication as the authentication process requires the typing pattern of the user, which is unique for each individual. Next, Shanmugapriya & Padmavathi (2009) discussed the importance of accuracy in biometrics authentication acceptance and implementation in the organization. Thus, the research in keystroke dynamics is necessary in order to come up with a model which is able to verify the genuine user and to detect impostor with as high accuracy as possible.

1.7. Scope of the Study

The scope of the research in keystroke dynamics is limited to a certain extent. Firstly, the model will be built based on deep learning approach. Despite other classifier types such as statistical approach and machine learning approach can also be further explored to improve the existing classifiers, it requires more research allocation and is not supported by the duration given to complete the project. Secondly, the implementation of the research will focus only on one benchmark dataset. Using several datasets in training and validation may improve the observation in terms of consistency of the model but it will lead to a longer duration for training and validation processes depending on the classification method used. This may not be possible considering the time constraint of the project. However, a testing for the developed model can be performed on another dataset. Thirdly, due to the restriction mentioned previously, the research will not address the external factors affecting the keystroke timing such as emotion, age, handedness, health and others.

1.8. Research Methodology Overview

Research methodology shows how the study will be carried out phase by phase. The research methodology used in this study consists of six phases: a preliminary study, literature review, research methods, implementation, discussion and conclusion, and presentation. The research methodology overview of this project is illustrated in Figure 1.1 below.

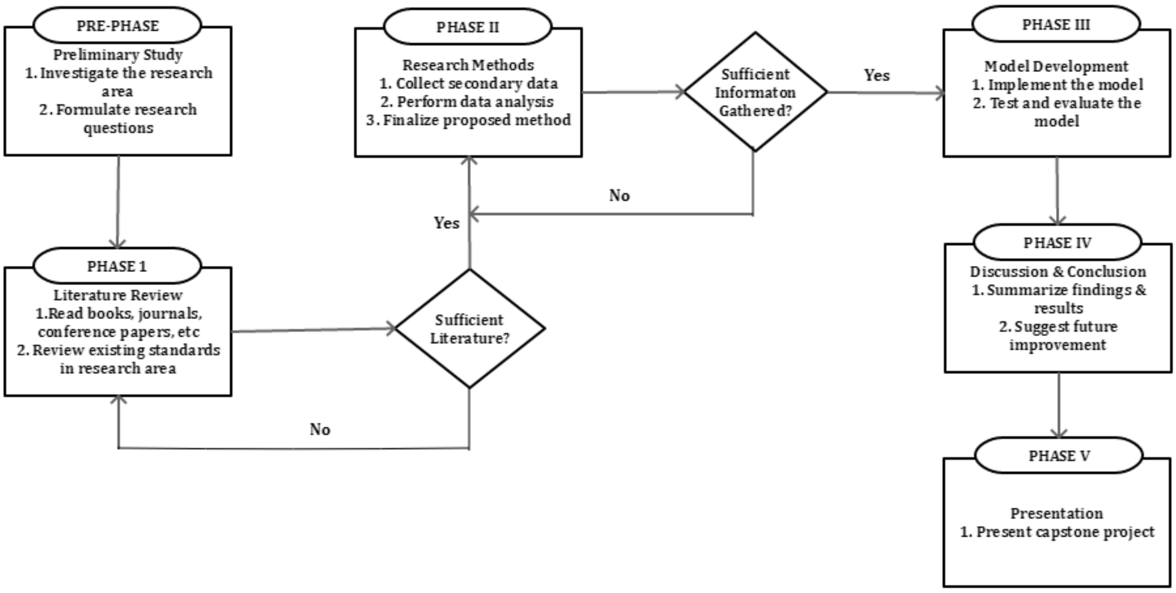

PRE-PHASE
Preliminary Study
1. Investigate the research area
2. Formulate research questions
PHASE 1
Literature Review
1.Read books, journals, conference papers, etc
2. Review existing standards in research area
Sufficient Literature?
No
Yes
PHASE II
Research Methods
1. Collect secondary data
2. Perform data analysis
3. Finalize proposed method
Sufficient Informaton Gathered?
Yes
No
PHASE III
Model Development
1. Implement the model
2. Test and evaluate the model
PHASE IV
Discussion & Conclusion
1. Summarize findings & results
2. Suggest future improvement
PHASE V
Presentation
1. Present capstone project

Figure 1.1: Research Methodology Overview.

In the beginning of this study, an investigation towards keystroke dynamics area is conducted and basic research questions are formulated in order to gain an initial understanding of the topic. Next, in the first phase of the study literature review for keystroke dynamics is performed by reviewing books, journals, conference papers, and other secondary sources. The purpose of this activity is to gain a better understanding of what has been achieved in the research area and to learn the existing standards in keystroke dynamics research. If sufficient numbers of literature have been reviewed, the next phase is to design research methodology for the study. This can be done by gathering secondary data which will be used in the study and by performing data exploration towards the collected data. In the methodology phase of the study, a deep learning method will also be elaborated according to the gathered knowledge and dataset. If the deep learning method has been clearly designed to fit the research aim and objectives, implementation of the method can be conducted. In this phase, the deep learning model is designed, tested, and evaluated on the dataset. Afterwards, the result of the implementation is discussed and analysed before drawing the conclusion of the study. At last but not least, this study as part of the capstone project is presented.

1.9. Structure of the Study

This study is organized as follows:

Chapter 2 / Literature Review

This chapter describes the research background of keystroke dynamics as a user authentication method. Key topics discussed in the chapter include biometric user authentication, history of keystroke dynamics, keystroke dynamics features, dataset, evaluation criteria, classification methods for keystroke dynamics such as statistical, machine learning, deep learning, and hybrid model, external factors affecting the performance of keystroke dynamics, and privacy in keystroke data acquisition.

Chapter 3 / Methodology

This chapter introduces the methodology used in this study and detailed explanation about the deep learning method. Key topics discussed in this chapter include research design used in the study, keystroke dynamics process for user authentication, deep learning architectures, areas in deep neural networks such as neural network concept, activation function, multilayer perceptron, weight initialization, gradient descent, error backpropagation, output and error function, and regularization.

Chapter 4 / Model Development

This chapter explains the model development processes for the proposed deep learning algorithm. Key topics discussed in this chapter include an explanation of the dataset, data exploration, data visualization, data selection, feature selection, development tool, and phases in model implementation.

Chapter 5 / Results and Analysis

This chapter reports the results and analysis of the deep learning implementation performed in Chapter 4. Key topics discussed in this chapter include evaluation criteria used in this study, results of the deep learning model implementation, evaluation of the deep learning model, and model testing on other datasets.

Chapter 6 / Discussion and Conclusion

This chapter summarises the findings of the study and draws a conclusion. Key topics discussed in this chapter include discussion and conclusion of the study, contribution and importance of the study, and future recommendation for research in keystroke dynamics.

1.10. Summary

As a summary, keystroke dynamics is a user authentication method which uses user's typing rhythm to allow access into the system. It has two advantages compared to other biometric authentication methods such as lower implementation cost as no additional hardware is required in the authentication process; and easier implementation because the collection of typing data is relatively easy as it does not require special permission from the user. Advances in keystroke dynamics have produced multiple classifiers such as statistical and machine learning in order to perform classification for genuine user and impostor, however, the maximum rate of accuracy has not been achieved. In order to solve the problem, the aim of this study is to propose deep learning model in keystroke dynamics for user authentication. This study is important because it can potentially increase the user awareness and understanding regarding the biometrics authentication, and tackle security issues specifically in access control and data privacy as can provide better authentication measure compared to SFA. The scope of this study is limited to the implementation of deep learning, training model with one dataset, and does not cover external factors affecting keystroke dynamics performance. The research methodology used in this study consists of six phases: a preliminary study, literature review, research methods, implementation, discussion and conclusion, and presentation. Finally, this chapter suggests brief overview on the structure of the study.

CHAPTER 2

LITERATURE REVIEW

2.1. Introduction

This chapter discusses the literature review in keystroke dynamics area. First of all, a brief introduction to user authentication including its types will be discussed. Next, keystroke dynamics will be explored as part of biometric user authentication. Topics covered in keystroke dynamics include its history in user authentication, features, dataset, evaluation criteria, and different classification approaches such as statistical, machine learning, deep learning, and hybrid model. Besides, factors which affect the performance of keystroke dynamics will also be discussed. Furthermore, privacy concern in keystroke data collection for dynamic text is elaborated. Finally, the researcher's discussions towards the topic including the summary and challenges of the study are presented.

2.2. User Authentication

User authentication refers to a process of verifying whether a user is eligible to access a system by using certain credentials. Implementing user authentication to information systems is important as they contain confidential information which belongs to a particular user. User authentication would be useful to protect the data and prevent possible harmful effect concerning the unauthorized access of the data. There are three types of user authentication such as knowledge-based authentication, object-based authentication and biometric-based authentication. Knowledge-based authentication uses something which has been learned and can be recalled by a user, for instance, password and PIN. Object-based authentication uses something (physical) which belongs to a user, such as a door key and access card. Biometric-based authentication works by using physiological or behavioural property which belongs to a user, for example, a fingerprint and keystroke dynamics. According to Vielhauer (2006), using biometric properties of a user for authentication is preferable as it tackles the issue of transferability of credentials which refers to a condition where imposter can obtain genuine user's credential easily. Unlike knowledge-based or object-based authentication, biometric-based authentication takes advantage of the fact that physiological and behavioural properties of an individual are unique from one another. Thus, it is more difficult to duplicate the user's biometric properties. There are two types of biometric properties of a user: a) physiological property which covers visible part of the human body; and b) behavioural property which covers behaviour pattern of a human. The Figure 2.1 below illustrates the types of user authentication along with examples.

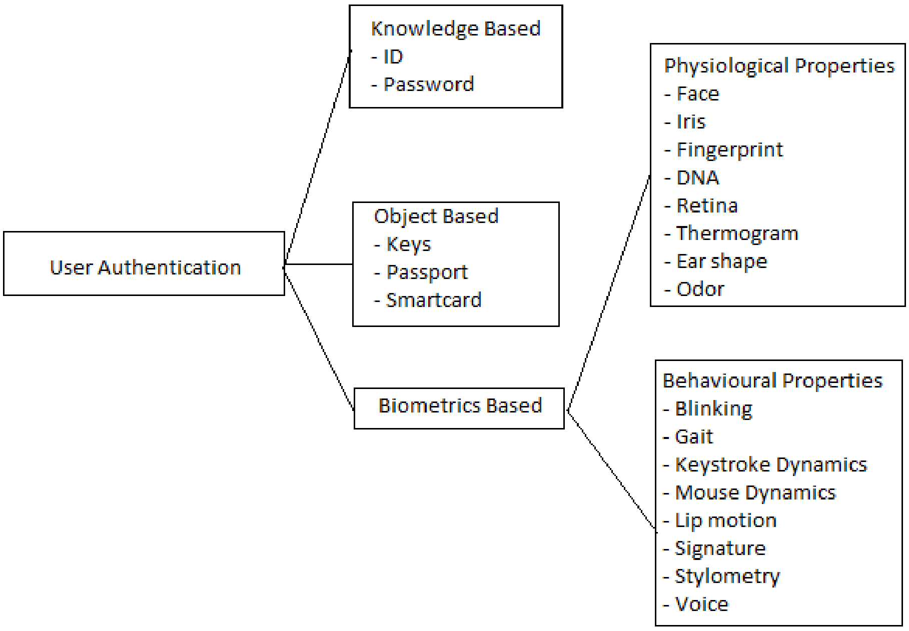


Figure 2.1: Types of User Authentication.

There are several behavioural properties of a user which have been used for user profiling and verification, as reviewed by Yampolskiy & Govindaraju (2008), such as blinking behaviour, calling behaviour, driving style, command line lexicon, credit card usage, dynamic facial features, e-mail behaviour, gait, game strategy, GUI (Graphical User Interface) interaction, handgrip, haptic, keystroke dynamics, lip motion, mouse dynamics, paint and sketch style, programming style, signature, stylometry, tapping and voice. These unique behaviour properties can be used to enhance user verification process and develop multi-modal user authentication system. For instance, by implementing keystroke dynamics alongside with password-based authentication system, the impostor will not only need to obtain the knowledge of the password but also the knowledge of how the password is typed. Thus, better security is provided by using multi-modal user authentication. Some studies on multi-modal biometrics authentication are: Fridman et al. (2015) who combined keystroke dynamics, mouse dynamics and stylometry in active authentication, Venugopalan et al. (2015) who applied electromyograph signals into keystroke dynamics for biometric authentication, and Kasprowski & Harezlak (2016) who fused the application of eye movement and mouse dynamics in biometrics authentication.

2.3. Keystroke Dynamics

Killourhy (2012) discussed the term keystroke dynamics as a method to differentiate users by using their typing rhythms. Typing pattern of a user is assumed to be unique from another user, which could be affected by factors such as age, gender, occupation, handedness and others.

2.3.1. History of Keystroke Dynamics

Bryan & Harter (1897) discussed the early applications of keystroke dynamics in middle 19th century used by telegraph operators and also in World War II where a methodology called ‘Fist of the Sender’ was used to identify the sender of a message by using typing rhythm, pace and syncopation of keys in the telegraph. Spillane (1975) initiated a concept of implementing keystroke dynamics to distinguish different users using the keyboard. Forsen. Nelson & Staron (1997) conducted an experiment where users were asked to type each other names using a keyboard in order to check whether the difference in typing behaviour existed among them. Next, Gaines et al. (1980) performed typing experiment using timing features towards seven typists and analysed the results by using statistical analyses. Moving to early 2000s, hundreds of keystroke dynamics studies using different classifiers had been proposed (Killourhy, 2012). These classifiers are categorized into different approaches such as statistical, machine learning, hybrid and neural network.

2.3.2. Features

Keystroke dynamics involves several timing features which are recorded (often in second) when a particular key is pressed on a keyboard. According to Maheshwary & Pudi (2017), there are three commonly used timing features for keystroke dynamics studies, such as digraph time, dwell time and flight time. Digraph time (or DD/down-down time) is the duration required between pressing a key and pressing the next key. Dwell time (or H/hold time) is the duration required between pressing a key and releasing that key. Flight time (or UD/up-down time) is the duration required between releasing a key and pressing the next key. Thus, digraph time is the sum of dwell time and flight time.

The Figure 2.2 below illustrates the keystroke timing features.

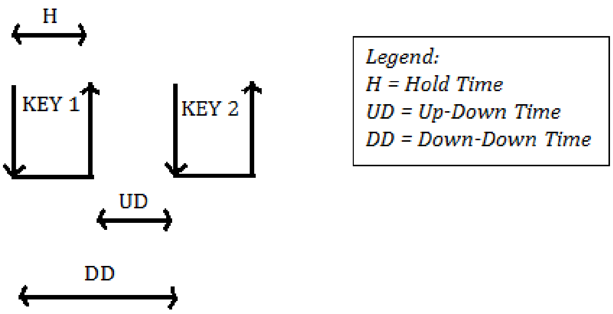


Figure 2.2: Keystroke Dynamics Timing Feature.

Alsultan, Warwick & Wai (2017) discussed that overlapping between keystroke events could lead to negative UD time. This situation occurs when the second key is pressed before the first key is released. The Figure 2.3 below illustrates the case of keystroke overlapping.

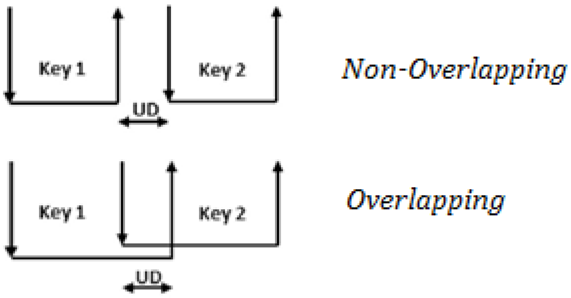


Figure 2.3: Negative UD Time (adapted from Alsultan, Warwick & Wei (2017), p.55).

Apart from the three features above, there have been researchers who used different features such as UU/up-up time, bigram time, inversion ratio time, and trigraph (using the duration of pressing/releasing three consecutive keys) timing features. Besides, some researchers also incorporated keystroke pressure feature in their analysis.

2.3.3. Dataset

Collection of timing features and user details are stored into a keystroke dynamics dataset. This dataset will be used to perform analysis and to build a classification model for user authentication system. Banerjee & Woodard (2012) described two types of keystroke dynamics data, which are static/fixed text and dynamic/free text. Static text is text which has been predetermined by the researcher whereas dynamic text is one that originated from the typing participants directly. There are twenty publicly available datasets for keystroke dynamics as provided by (Monaco (n.d.); Roth et al. (2013); and Teh et al. (2010)). The list of the dataset is shown in Table 2.1 below.

Table 2.1: List of Publicly Available Dataset for Keystroke Dynamics

No.	Dataset Name	Text	Participants
1.	CMU Keystroke Dynamics – Benchmark Data Set	Static	51
2.	CMU Free vs. Transcribed Text	Dynamic	20
3.	Keystroke Dynamics – Android Platform	Static	42
4.	BeiHang Keystroke Dynamics Database	Dynamic	209
5.	BioChaves Keystroke Databases	Static	47
6.	Keystroke100 Dataset	Static	100
7.	GREYC Keystroke Datasets	Static; Dynamic	113; 118; 110
8.	RHU KeyStroke Dynamics Benchmark Dataset	Static	51
9.	CITEFA Keystroke Dynamics Dataset	Static	63
10.	Clarkson University Keystroke Dataset	Static	39
11.	ATVS-Keystroke DB	Dynamic	63
12.	LSIA Keystroke Dynamics keypress latency dataset	Static	-
13.	Stonybrook Keystroke Patterns as Prosody in Digital Writings	Dynamic	196
14.	Keystroke acoustics, video and timings	Static; Dynamic	81
15.	Detecting Age Groups using Keystroke Dynamics	Static	100
16.	How We Type	Dynamic	30
17.	MOBIKEY Keystroke Dynamics Password Database	Static	54
18.	Statistical Analysis of Personality and Identity in Chats	Dynamic	50
19.	Strengthening Password Authentication using Keystroke Dynamics	Static	100
20.	Acoustic and Visual Typing Behaviour Dataset	Static, Dynamic	56; 30

Although keystroke datasets have been made publicly available, there are some researchers on keystroke dynamics who generated and analysed their own data by using keylogger software. These datasets are collected mainly using keyboard and smartphones. Analysis of keystroke data on the keyboard is different from a smartphone because the way a user type on the keyboard will be different with the one on a smartphone due to size and pressure of the key. Besides, there are two types of environment used for keystroke data acquisition such as controlled environment and uncontrolled environment. In a controlled environment, participants are asked to perform typing task on a given machine. By doing so, each participant will have an equal condition in performing the task. On the other hand, in the uncontrolled environment, the participants are requested to download software necessary to record the typing operation on their personal machines. Then, the participant will send the collected data to the researcher. While performing typing operation in an uncontrolled environment is easier for the participants, analysing the data will be more challenging for the researcher as there are many factors to be considered. For instance, the key's size and sensitivity in a keyboard could be different among computer brands.

2.3.4. Evaluation Criteria

As part of biometric authentication, evaluation criteria are important for keystroke dynamics in order to measure the performance of the classifier in classifying genuine user and impostor. According to Pisani & Lorena (2013), there are commonly used evaluation criteria for keystroke dynamics as stated below:

- False Acceptance Rate (FAR) measures how often a classifier falsely identifies impostor as a genuine user by calculating false matches over total impostor match attempts. FAR can also be referred as False Negative Rate (FNR).
- False Rejection Rate (FRR) measures how often a classifier falsely identifies a genuine user as an impostor by calculating false rejection over total genuine match attempts. FRR can also be referred as False Positive Rate (FPR).
- Equal Error Rate (EER) measures threshold point between FAR and FRR; and
- Accuracy rate measures correct classification obtained by the classifier in percentage.

The Figure 2.4 below illustrates the relationship between FAR, FRR, and EER plots.

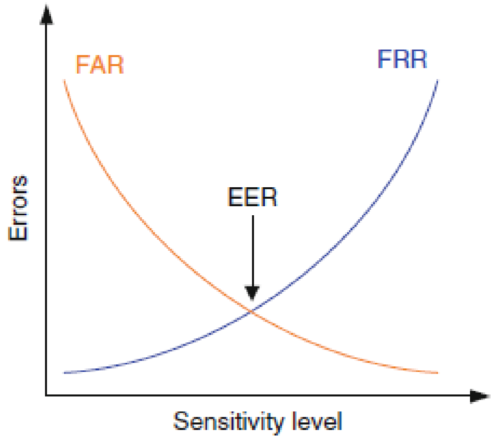


Figure 2 4: Performance Evaluation for Keystroke Dynamics Model (adapted from Pisani & Lorena (2013), p.583).

2.3.5. Classification Methods

Once the keystroke data has been collected and keystroke features have been selected, the next step is to choose a classification method which will be used to classify whether a user is genuine or an impostor. This classification is done by identifying similarities and differences in users' keystroke pattern. In the previous studies of keystroke dynamics, several approaches such as statistical, machine learning, neural network and hybrid models have been explored. These classification techniques are categorized based on their usage popularity in keystroke dynamics studies.

2.3.5.1. Statistical Approaches

Generally, statistical algorithm deals with calculation of mean and standard deviation of the features. It also calculates distance measures such as Euclidean, Manhattan, Mahalanobis and so on in order to classify the users. These calculations can then be used to generate inferences by using hypothesis testing and other statistical tests. Eltahir et al. (2008) conducted an experiment on keystroke based authentication system by using latency classifier which measured the latency score of user login attempt using standard deviation. Furthermore, they also implemented autoregressive classifier to compare the users' typing pressure and achieved FAR of 3.75% and FRR of 3.04%.

Teh et al. (2007) performed statistical fusion approach to classify 50 users' keystroke data by using direction similarity measure with Gaussian probability density function. It calculates mean and standard deviation of the keystroke data and applies weighted sum rule to improve the result. Their experiment achieved EER of 6.36%. Messerman et al. (2011) evaluated keystroke data obtained from 55 users by improving Spearman's Foot-rule distance with R-distance and A-distance which measured the relationship and mean distance between two users. Their experiment achieved FRR of 1.84%. Killourhy (2012) performed testing in Carnegie Mellon University (CMU) dataset using several statistical classifiers such as scaled Manhattan, Mahalanobis, Outlier Count and Euclidean with EER of 9.6%, 11%, 10.2% and 37.2% respectively. By using the same CMU dataset, Al-Jarrah (2012) applied median vector proximity technique for keystroke classification instead of mean to avoid outliers due to extreme value in the data and achieved EER of 8%.

Xuan, Fangxia & Jian-Feng (2012) explored a method to calculate the difference between users' feature vector by using slope correlation degree and achieved FAR of 47%. Giot, Abed & Rosenberger (2012) implemented Gaussian distribution in their experiment to calculate the mean and standard deviation distances between 48 samples and achieved EER of 13.15%. Deng & Zhong (2013) also performed testing using Gaussian Mixture Model (GMM) on CMU dataset and achieved EER of 5.5%. Bakelman et al. (2013) conducted an experiment on two keystroke datasets (51 users and 30 users) using Pace classifier which separates a user into the authenticated group and non-authenticated group based on the differences between feature vectors. Their experiment achieved EER of 8.7% for the former dataset and 6.1% for the latter dataset. Al-Rahmani (2014) proposed an enhancement model towards previous median vector proximity technique by using a Distance-to-Median metric which refers median as centre-point of classification. The experiment achieved EER of 7%. Morales & Fierrez (2014) conducted an experiment on keystroke data collected from 64 users by using normalized Manhattan distance and achieved an accuracy of 90%. Monaco & Tappert (2016) developed new model named Partially Observable Hidden Markov Model and performed testing on CMU dataset and achieved EER of 4.2%. Roy, Roy & Sinha (2016) used outlier count algorithm on knowledge-based user authentication system using keystroke dynamics and achieved EER of 2.4%. He, Li & Shen (2017) conducted an experiment on the normalized dataset using several algorithms and calculated the F1-measure score for them. The best performing statistical classifier on their experiments were Euclidean, Manhattan and Mahalanobis with F1 score of 5.6949, 7.1062 and 3.9274 respectively.

Statistical approaches have been popular in keystroke dynamics area and have obtained promising results as discussed above. However, using a statistical approach for keystroke classification could potentially overlook the existence of patterns in the data due to limited training stage. Besides, a statistical approach might not be suitable for handling non-linear keystroke data.

2.3.5.2. Machine Learning Approaches

Machine learning algorithm deals with pattern discovery and classification in order to form categories. The implementation of machine learning algorithms differs based on their complexity. For instance, nearest neighbour algorithm is simpler than Support Vector Machine (SVM). Gingrich & Sentosa (2008) used k-Nearest Neighbour (k-NN) as part of their clustering based keystroke authentication algorithm to solve scalability problem in classification and achieved FAR of 0.045%. Gio, Abed & Rosenberger (2009) performed classification using SVM towards keystroke data collected from 100 users and achieved EER of 6%. Killourhy (2012) also conducted an experiment on keystroke dynamics using SVM in 51 users' keystroke data and achieved EER of 10.2%. Wangsuk & Anusas (2013) proposed trajectory dissimilarity technique as an additional token for user verification by comparing the distance between genuine user/master trajectory profile and user attempts using Euclidean distance metric. Their experiment achieved EER of 4%.

Traore et al. (2014) applied Bayesian network algorithm in keystroke dynamics to classify 24 users' keystroke data and achieved EER of 24.78%. Pisani, Lorena & Carvalho (2015) proposed adaptive positive selection algorithms to tackle intra-class variation problem. The experiment was conducted on three datasets and it was found that Sliding method performed the best with a balanced accuracy of 77.5% up to 92.5%. Pisani & Lorena (2015) also used Self-Detector algorithm with rank transformation (immune algorithm) in order to measure the correlation between vectors by cosine distance. Their experiment was carried out on two datasets (100 users and 51 users) and achieved 80% accuracy and 70% accuracy respectively. Next, Pisani et al. (2016) proposed Enhanced Template Update to reduce chances of getting an error (FRR) by using Positive Gallery Protection.

Darabseh & Namin (2016) compared two classifiers: SVM and k-NN for their 28 users' keystroke dataset for active authentication process using 4 keystroke features. Their experiment found that hold time offered the most promising result as both classifiers achieved 84% accuracy. Maheshwary & Pudi (2017) engineered two new features named bigram time and inversion ratio time from CMU dataset and performed nearest neighbour regression on the data with and without the new features. Their experiment achieved EER of 6.98% with new features and 7.39% without the new features. Ho & Kang (2017) performed attribute ranking one-class Naïve Bayes classification to two datasets: 51 users and 118 users, and achieved EER of 6.6% and 4.7% respectively. Alsultan, Warwick & Wai (2017) implemented non-conventional features such as typing speed, error rate and shift key usage in keystroke dynamics classification using decision tree and SVM. Their experiment found that SVM performed better with FAR of 1.1% and FRR of 28%.

In keystroke dynamics studies, neural network has been used more frequently than other machine learning algorithms. Neural network or also known as Artificial Neural Network (ANN) is an adaptive learning algorithm developed based on how human brains learn something new. It consists of three layers: input, hidden and output, which transfer information across the network. Revett et al. (2007) performed classification on keystroke data from 50 users and achieved EER of 3.9% using a neural network. Loy, Lai & Lim (2007) used ARTMAP-FD neural network in order to identify keystroke latency and pressure patterns from inner and outside of boundaries, as an extension of adaptive resonance neural network. Their experiment achieved EER of 11.78%. Awad, Traore & Almulhem (2008) performed free-text detection using keystroke data collected from 22 users by using neural network and achieved FAR of 0.0152% and FRR of 4.82%. Pisani & Pereira (2010) applied Lamarckian Evolutionary Neural Network which is similar to backpropagation neural network into keystroke dynamics and achieved FAR of 0.22% and FRR of 8%. Killourhy (2012) performed classification on keystroke dynamics by using neural network and auto-association neural network and achieved EER of 82.8% and 16.1% respectively. Uzun & Bicakci (2012) proposed backpropagation neural network algorithm in keystroke dynamics and achieved EER of 7.73% by also feeding impostor attempts on the network. Alpar (2014) implemented ANN with RGB histogram in order to colorize user's keystroke timing for better security measure. The experiment achieved a classification accuracy of 90%.

Machine learning approach has an advantage over statistical approach in the sense that it is able to handle non-linear data and extracts patterns which might not be visible from linear data assumption. Besides, machine learning algorithm is able to provide and rank confidence value for each decision. By doing so, it can eliminate decision with low confidence value and increase the classification performance. Furthermore, implementation of neural network algorithms has contributed promising classification accuracy for keystroke dynamics authentication. One advantage of neural network is the ability to handle many parameters, which can be suitable for keystroke classification using a lot of features. However, the training time of neural network could take longer time compared to other algorithms. Also, neural network has a black box characteristic in which it is difficult to identify which keystroke feature is the most significant to the study.

2.3.5.3. Deep Learning

Deep learning is an advanced version of machine learning in a sense that it achieves higher abstraction and better flexibility in the representation. The use of deep learning has become popular in image recognition and speech recognition tasks. Deng & Zhong (2013) conducted an experiment using Deep Belief Network (DBN) which captured more complex non-linear features from the keystroke data. Their experiment on CMU dataset achieved EER of 3.5%. Kobojek & Saeed (2016) applied two recurrent neural networks model such as Long-Short Term Memory (LSTM) and Gated Recurrent Unit (GRU) on CMU dataset. Their experiment achieved EER of 13.6% for LSTM 2 cells, 16.5% for LSTM 3 cells, and 22.4% for GRU.

Implementation of deep learning model might require a higher cost in terms of hardware as it performs the best on Graphics Processing Unit (GPU), which supports higher computation capability than Central Processing Unit (CPU). Besides, the training time for deep learning model could take longer duration as it contains many parameters. However, deep learning algorithm could produce better performance and support a larger range of data compared to machine learning algorithm.

2.3.5.4. Hybrid Models

Hybrid model is proposed in order to increase classifier performance. It can be a fusion of neural network with other machine learning algorithm, statistical algorithm and vice versa. Azevedo, Cavalcanti & Filho (2007) conducted an experiment on 24 users' keystroke data by using hybrid model consisting of Support Vector Machine and two optimization techniques: genetic algorithms and particle swarm optimization, and achieved FAR and FRR of 2.5%, 0.37% and 1.18%, 0.41% respectively. Bharadi, Shah & Ambardekar (2011) discussed the application of relative entropy and Euclidean distance in k-NN algorithms and achieved a classification accuracy of 55% and 77% respectively. Monaco et al. (2013) performed classification using Euclidean distance in the k-NN algorithm and achieved EER of 4%. Nisha & Kumar (2014) combined backpropagation neural network with Artificial Bee Colony Optimization technique for classifying users based on their keystroke digraph and latency features. Their experiment achieved EER of 4.5%. Mohabeer & Soyjaudah (2015) applied genetic algorithms for neural network (Neuroevolution of Augmented Topologies) into keystroke dynamics. The experiment was conducted on 100 users and achieved Mean Squared Error (MSE) of 4%.

Previous studies have proven that hybrid model can also be used in keystroke dynamics and achieve promising results. Although it increases the complexity of the classification process, the hybrid model could tackle disadvantage that a single classifier has and improve the classification accuracy.

The Table 2.2 below summarizes the classification approaches used by previous studies in keystroke dynamics from the year 2007 until 2017. The table consists of research reference, list of features, classification algorithm, type of text (TT), environment (Env.), number of subjects, and evaluation of the classifier (Eval.). The type of text is divided into two categories: static (S) and dynamic (D). The environment is also divided into two categories: controlled (C) and uncontrolled (UC).

Table 2.2: Classification Approaches Summary

Reference	Features	Classification	TT	Env.	Subject	Eval. (%)
Revett et al. (2007)	Digraph, Trigraph	Neural Network	S, D	C	50	EER: 3.9
Azevedo, Cavalcanti & Filho (2007)	DD, H, UD	SVM with: Genetic Algorithms (1) Particle Swarm Optimization (2)	S	C	24	FAR(1): 2.5 FRR(1): 0.37 FAR(2): 1.18 FRR(2): 0.41
Loy, Lai & Lim (2007)	Timing, Pressure	ARTMAP-FD Neural Network	S	UC	100	EER: 11.78
Teh et al. (2007)	DD, H, UD	Direction Similarity Measure with Gaussian Probability Density Function	S, D	C	50	EER: 6.36
Eltahir et al. (2008)	Pressure	Autoregressive and Latency Classifiers	S	UC	23	FAR: 3.75 FRR: 3.04
Awad, Traore & Almulhem (2008)	Digraph	Neural Network	D	UC	22	FAR: 0.0152 FRR: 4.82
Gingrich & Sentosa (2008)	Digraph, Trigraph	k-Nearest Neighbor	S, D	UC	19	FAR: 0.045 FRR: 0
Gio, Abed & Rosenberger (2009)	DD, H, UD	Support Vector Machines	S	C	100	EER: 6
Pisani & Pereira (2010)	DD, H, UD	Neural Network	S	C	10	FAR: 0.22 FRR: 8
Messerman et al. (2011)	Digraph	Statistical	D	C	55	FRR: 1.84
Bharadi, Shah & Ambardekar (2011)	DD, H, UD	k-NN with: Euclidean(1) Relative Entropy(2)	S	C	33	EER(1): 77 EER(2): 55

Table 2.2: continued

Reference	Features	Classification	TT	Env.	Subject	Eval. (%)
Al-Jarrah (2012)	DD, H, UD, UU, DU	Statistical	S	C	51	EER: 8
Killourhy (2012)	DD, H, UD	Scaled Manhattan (1) Nearest Neighbor Mahalanobis (2) Outlier Count (3) SVM (4) Mahalanobis (5) Normed Mahalanobis (6) Filtered Manhattan (7) Manhattan (8) Auto-Assoc Neural Network (9) Euclidean (10) Normed Euclidean (11) Fuzzy Logic (12) k-Means (13) Neural Network (14)	S	C	51	EER(1): 9.6 EER(2): 9.9 EER(3): 10.2 EER(4): 10.2 EER(5): 11.0 EER(6): 11.0 EER(7): 13.6 EER(8): 15.2 EER(9): 16.1 EER(10): 17.0 EER(11): 21.5 EER(12): 22.1 EER(13): 37.2 EER(14): 82.8
Xuan, Fangxia & Jian-Feng (2012)	Digraph	Statistical	D	UC	20	FAR: 0.47 FRR: 0
Uzun & Bicakci (2012)	DD, H, UD	Neural Network	S	C	51	EER: 7.73
Giot, Abed & Rosenberger (2012)	DD, H, UD	Statistical	D	UC	48	EER: 13.15
Monaco et al. (2013)	Digraph	k-Nearest Neighbor with Euclidean Distance	D	C	30	EER: 4

Table 2.2: continued

Reference	Features	Classification	TT	Env.	Subject	Eval. (%)
Bakelman et al. (2013)	DD, H, UD	Pace Classifier	S	C, UC	51, 30	EER: 8.7 EER: 6.1
Deng & Zhong (2013)	DD, H, UD	Gaussian Mixture Model (1), Deep Belief Nets (2)	S	C	51	EER(1): 5.5 EER(2): 3.5
Wangsuk & Anusas (2013)	H, Latency, Interkey	Trajectory Dissimilarity	S	C	23	EER: 4
Al-Rahmani (2014)	DD, H, UD	Statistical	S	C	51	EER: 7
Alpar (2014)	Digraph	Artifical Neural Network with RGB Histogram	S	C	10	Accuracy: 90
Morales & Fierrez (2014)	Digraph, Trigraph	Normalized Manhattan Distance	D	C	64	Accuracy: 90
Traore et al. (2014)	DD, UD	Bayesian Network	D	UC	24	EER: 24.78
Nisha & Kumar (2014)	Digraph, Latency	Artificial Bee Colony Optimization with Back Propagation Neural Network	D	C	-	EER: 4.5
Pisani, Lorena & Carvalho (2015)	UD	Adaptive Positive Selection	S	C	100 (1) 51 (2) 118 (3)	FAR(1): 8.3 FRR(1): 8.6 FAR(2): 6.6 FRR(2): 12.2 FAR(3): 6.2 FRR(3): 6.0
Pisani & Lorena (2015)	UD	Immune Algorithms	S	C	100 (1) 51 (2)	Accuracy(1):80 Accuracy(2):70
Mohabeer & Soyjaudah (2015)	DD	Neuroevolution of Augmented Topologies	S	C	100	MSE: 4
Darabseh & Namin (2016)	DD, H, UD, Total	Support Vector Machines, k-Nearest Neighbor	S	C	28	Accuracy: 84

Table 2.2: continued

Reference	Features	Classification	TT	Env.	Subject	Eval. (%)
Pisani et al. (2016)	UD	Enhanced Template Update	S	C	51(1), 118(2)	FAR(1): 6.7 FRR(1): 15.1 FAR(2): 13.4 FRR(2): 26.8
Monaco & Tappert (2016)	DD, H, UD	Partially Observable Hidden Markov Model	S	C	51	EER: 4.2
Roy, Roy & Sinha (2016)	Digraph, Trigraph	Outlier Count	S	C	12	EER: 2.4
Kobojek & Saeed (2016)	DD, H, UD	Long Short Term Memory (1), Gated Recurrent Unit (2)	S	C	51	EER(1): 13.6 EER(2): 22.4
Alsultan, Warwick & Wai (2017)	DD, UU, H, UD, NC	Decision Tree & Support Vector Machines	D	C	30	FAR: 1.1 FRR: 28
Maheshwary & Pudi (2017)	DD, H, UD, Bigram, Ratio	Nearest Neighbor Regression with new features (1); without new features (2)	S	C	51	EER(1): 6.98 EER(2): 7.39
He, Li & Shen (2017)	H, UD	Normalized Dataset: Euclidean(1), Manhattan(2), Mahalanobis(3), Fisher LDA(4), GPC(5), Linear SVM(6), Polynomial SVM(7), RBF SVM(8), Sigmoid SVM(9), Euclidean k-NN(10), Manhattan k-NN(11), Mahalanobis k-NN(12), Random Forest(13), Random Guessing(14)	S	C	51	F1(1): 5.6949 F1(2): 7.1062 F1(3): 3.9274 F1(4): 38.1318 F1(5): 39.1513 F1(6): 6.0461 F1(7): 5.9127 F1(8): 0 F1(9): 1.9608 F1(10): 57.3094 F1(11): 57.3178 F1(12): 0.0754 F1(13): 69.019 F1(14): 1.9608
Ho & Kang (2017)	DD, H, UD, UU, DU	Attribute Ranking One-Class Naïve Bayes	S	C	51(1), 118(2)	EER(1): 6.6 EER(2): 4.7

2.3.6. External Factors

There have been researches which focused on the implication of external factors towards typing pattern and timing. A study performed by Lee, Tsui & Hsiao (2015) indicated that changes in emotion caused by experiencing an error in the system could result in unstable typing behaviour. Idrus et al. (2014) have also shown that soft biometrics such as gender, age, handedness and number of hand used to type could affect the performance of keystroke dynamics. Syed, Banerjee & Cukic (2014) introduced event sequences in keystroke dynamics while taking keyboard's flexibility into account to distinguish users based on their typing proficiency and effect of habituation. Basically, event sequence notes all possible combination that a user could use to fully type a phrase. For instance, the user may use left or right shift key, or even caps lock key to type the letter 'E' in 'tEsting' word. Their experiment showed that different event sequence made each user unique, and also user habituation towards a phrase led to less variation in the event sequence. Montalvao et al. (2015) conducted a test to check how users developed stable rhythmic signature while typing a new password and the influence of symbols in the password towards error rates. Their experiment showed that users required a number of repetitions before their typing rhythm became stable and error rates while typing tended to decrease accordingly to the increase of password length. Kang (2015) proved that considering user's typing proficiency and length of text could improve the keystroke authentication performance. Brizan et al. (2015) explored the effect of stylometry and language production with keystroke dynamics. Goodkind, Brizan & Rosenberg (2017) found out that overt and latent linguistic could affect the typing performance of the users in keystroke-based authentication. In addition, non-conventional factors such as typing speed, usage of shift and caps lock keys could also affect the classification performance (Alsultan, Warwick & Wei, 2017).

2.3.7. Privacy in Keystroke Data Acquisition

Data privacy concerns the usage of collected data from individuals in a legal manner. The data should not be used without the owner's consent. In keystroke dynamics studies, data privacy issue might appear on free-text data collection, in which the participants are asked to type dynamic text (usually password). This issue does not exist in fixed-text data collection because all participants are asked to type a specified text given by the researcher. However, regardless of the type of collected data, there is still a privacy concern for participants' typing rhythm. As previously discussed, typing rhythm is similar to another biometric measure. It is unique for each individual and this information should be protected to avoid the risk of account breach – if the participants use keystroke dynamics as their authentication measure. Because of these issues, privacy-preserving act on keystroke dynamics is needed.

Several privacy-preserving measures for keystroke dynamics studies to protect user's data have been proposed. Spantzel et al. (2006) proposed a privacy-preserving methodology by using a vector-space model in user authentication system to generate cryptographic biometric keys which would be secured by using zero-knowledge proof of knowledge. It provides an eligibility of a user to prove biometric possession without sharing it, thus the information would be protected. Sedenka et al. (2014) designed a protocol called Privacy-Preserving Population-Enhanced Biometric Key Generation which utilized key randomness and biometric privacy concept to replace keystroke data. Basically, it generates biometric signals for each user based on population data which would then be secured using homomorphic encryption. Liu, Uluagac & Beyah (2014) developed a privacy-preserving multi-factor authentication system called MACA to protect sensitive users' data from third party and to generate users' hybrid profiles by using fuzzy hashing and fully homomorphic encryption. Abidin, Rua & Peeters (2017) proposed a two-factor authentication protocol (enrolment and authentication) to achieve two things: (1) unlinkability – allow generation of the similar biometric template from original one but disallow cross-matching between them and (2) irreversibility – prevent biometric template reconstruction of collected keystroke templates. The protocol works by generating binary metadata from user credentials and protecting it by using symmetric key encryption. The implementation of this protocol is said to be simpler than using homomorphic encryption.

2.4. Discussion

In this literature review, relevant information and studies in keystroke dynamics field over past decade have been discussed. Despite extensive researches have been done in this area, there are still challenges which need to be addressed in order for keystroke dynamics to become an effective biometric measure for user authentication.

Firstly, the dataset for keystroke dynamics research should be standardized. From the past literature, it could be seen that there are some researchers who utilized the same dataset, but there are also researchers who used their own dataset to perform classification and analysis. Although there might be no rule specifying the selection and usage of data in the research, it would be good if researchers who are working on the same field use the same dataset as previous researchers did, especially if same keystroke features are used. The reason of doing so is to make result comparison easier and to see which classifier performs the best for keystroke dynamics. Secondly, based on the literature it could be seen that some researchers used different features in their analysis. Choosing the proper feature in the analysis is a challenging task because some feature might be useful than others depending on the focus of the study and type of classifier used. Thus it would be helpful if a common agreement in feature selection is established. Thirdly, it is also important to note that external factors such as emotions, language, soft biometrics, stylometry, and others could also affect the classification accuracy in keystroke dynamics. Whilst these factors could make keystroke dynamics a less rigid authentication measure compared to other behavioural biometrics, further research could be done to explore more about motor behaviour. By doing so, it might give future researchers a better understanding of factors consideration while developing the classifier and improves the accuracy. At last but not least, there seems to be lack of visualization for the keystroke data in past studies. Visualization is an important process to find a pattern which might not be visible only by looking at the raw data. Although keystroke data mainly consists of numerical data, the relationship between variables could still be explored and possibly lead to a new insight of the study.

2.5. Summary

This study has presented a literature review on keystroke dynamics studies. In addendum, essential elements such as features, datasets, and evaluation criteria were investigated. Next, classification approaches including statistical, machine learning, deep learning and hybrid models are explained in chronological order. Besides, external factors affecting the classifier accuracy and data privacy issues are also discussed in this study. The most significant feature that affected keystroke dynamics performance had yet to be explored and CMU benchmark dataset was used more frequently while compared with other openly available dataset. EER is the most widely used error metric to assess the performance of keystroke dynamics classifier. Based on the survey, deep learning implementation in keystroke dynamics are limited while compared with other classification methods. It is noted that external factors such as emotions, language, soft biometrics, stylometry, and others could also affect the classification accuracy in keystroke dynamics. This work can be used as a reference for future researchers who are interested in engaging in keystroke dynamics field.

Keystroke dynamics is an interesting field to explore as one type of biometric authentication measure despite it has lower classification accuracy and a limited amount of studies compared to other biometric modalities. Although the field of study is still open to challenges and improvement, it has a potential to become an effective, strong and low-cost biometric user authentication.

CHAPTER 3

RESEARCH METHODOLOGY

3.1. Introduction

Research philosophy is defined as a preference of assumption towards data and knowledge. In this study, positivism research philosophy will be used because of two reasons. Firstly, the research in keystroke dynamics deals with quantitative data which will be used to perform classification for genuine user and impostor. Secondly, it is categorized as an objective research as the result of this research is verifiable by using mathematical calculations. Thus, the output of this research can only be decided after the test is conducted. Next, this study will use deductive research approach in which hypothesis is formulated for each testing. In keystroke dynamics research, hypothesis testing can be used to determine whether the input data is categorized as genuine user or impostor.

In this study, a classification model to differentiate genuine user and impostor will be developed by using deep learning approach. The model will be evaluated by using keystroke dataset which has been available publicly. Thus, this study will only utilize secondary data and not primary data. After the performance of the model has been evaluated, the next step will be to test the model with another available keystroke dataset. The purpose of doing this is to determine whether the model can fit into multiple keystroke dynamics cases.

In this section, research design methodology used in this study, which is adapted from Knowledge Discovery in Database (KDD) will be elaborated. The sub-section will discuss data selection, data pre-processing, and tools required for achieving the outcome of this study. The following sub-section will discuss deep learning model for the classification task, including user authentication process for keystroke dynamics.

3.2. Research Design

The main research methodology for the project is quantitative research, which means it will deal with objective measurements and apply numerical analysis or computational techniques to the collected data. To fulfil the quantitative research requirements, KDD will be used. Fayyad, Shapiro & Smyth (1996) stated that KDD is useful to identify patterns in data.

KDD process consists of five steps as shown in Figure 3.1 below.

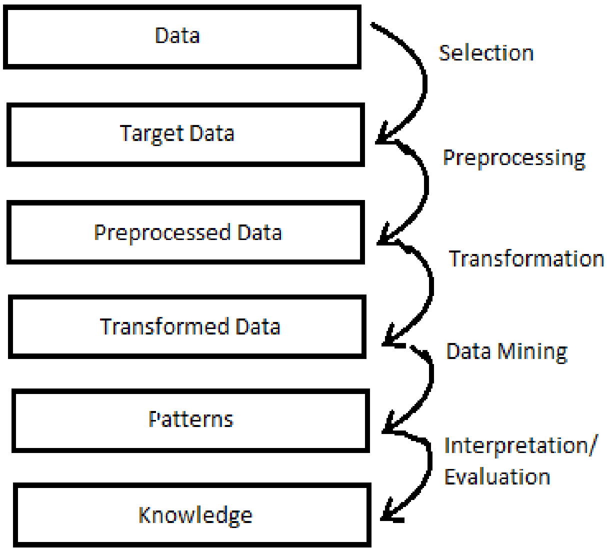


Figure 3.1: Knowledge Discovery in Database (Fayyad, Shapiro & Smyth (1996), p. 41)

3.2.1. Selection

Selection is the process of selecting target dataset which fulfils the domain, customer and objective requirements. In this research, a benchmark dataset for keystroke dynamics named CMU will be selected. Killourhy (2012) published the dataset for the purpose of aiding future research concerning keystroke dynamics area. CMU dataset is chosen because it has been used frequently in past studies. Thus, to ensure consistency among the studies and to compare the output between them easily, Maheshwary & Pudi (2017) suggested the use of the same standardized dataset for future keystroke dynamics research. Besides, it will also ease the process to improve the previous classifier which uses the same dataset.

CMU dataset itself consists of 20400 observations collected from 51 users performing typing task with the password (".tie5Roanl") for 8 different sessions at 50 typing repetitions for each session. The dataset also contains 34 columns which record the dwell, diagraph and flight time for each of the letter in the password, with measurement unit of second. Thus, the dataset consists of 31 numerical attributes for timing feature and 3 categorical attributes for the subject number, session index and repetition.

3.2.2. Pre-processing

Pre-processing is the process of cleaning the selected data in order to eliminate noise and outliers. In this step, the CMU dataset will be observed in order to find out if there is any missing data, noise or outlier. If any missing data exists, measures of central tendency can be used to fill in the missing value. There are three types of measures of central tendency such as mean, median and mode. Since the dataset consists of timing information with a close range of deviation and falls under quantitative data, mean is more appropriate to be used as the measure of central tendency (Laerd, 2013). Next, if any noisy data or outlier exists, the binning method can be used to smooth the data. It works by partitioning a set of data into equal bins and the mean of each bin is calculated to replace the previous value. By performing binning, duplicate records, incomplete or inconsistent data can be removed. Another way to handle outlier is by performing normalization in order to scale the input data to value between 0 and 1. The purpose of normalization is to allow the model to treat all instances similarly during the training phase. After the data has been cleaned successfully, it is required to ensure all the data is in the proper format. For instance, to ensure all the numerical attributes are recorded in measurement unit of second. If another measurement unit such as minute or hour is found, it needs to be converted into second. This will ease the data analysis process and ensure consistency. The result of this step is called pre-processed data.

3.2.3. Transformation

Transformation is the process of gathering summary or aggregation operations from the cleaned data. The main purpose of this stage is to find useful feature in keystroke dynamics data to be analysed. Currently, as CMU dataset contains 34 attributes, it is unclear which attribute is more significant against the others. By determining attributes which are significant, less significant attributes can be ignored in order to reduce the time taken for analysis. This can be done by using Exploratory Data Analysis (EDA). Microsoft Power BI (Microsoft, 2017) allows data to be transformed into insights easily with interactive graphics visualization in order to possibly identify previously unseen information about the data. Data sampling will be performed based on the result of the EDA. Besides, the file format needs to be changed to Comma Separated Values (CSV) to make it readable by the system in analysis step. The result of this step is called transformed data.

3.2.4. Data Mining

Data mining involves pattern extraction from the data by using intelligent methods. In this step, the deep learning model will be developed and tested using the transformed data. The purpose of the model is to perform classification towards the test data and to determine whether it belongs to genuine user or impostor. By the end of this step, a model should have been built and tested to identify new patterns.

The tool used in to develop the model is Weka Explorer 3.8.1 (Frank, Hall & Witten, 2016). It supports multiple machine learning algorithms which can be used to perform data mining activities. A package called Dl4jMlpClassifier will be used as a wrapper for an open-source deep learning programming library called Deeplearning4j which works by using Java virtual machine. This package allows the user to build a multi-layer perceptron deep learning model.

3.2.5. Interpretation/Evaluation

Evaluation is the process of generating new knowledge by unique patterns identification. In this step, the output produced by the model is interpreted and transformed into knowledge. One possible way to interpret the result is by using statistical inference (Killourhy, 2012). It enables the esearcher to understand whether the output has significant effect to the study or not, by formulating a hypothesis. Besides, statistical metrics such as kappa, Matthews correlation, and others can also be used to interpret the result. In addition, the model is evaluated whether it has fulfiled the expected requirements and is compared among existing model to determine which one is better and more applicable for biometrics user authentication. Comparison between these models is conducted using performance evaluation criteria. Future recommendation and challenges will also be discussed. By the end of this step, all the research questions should have been answered in order to fulfil objectives of the research.

3.3. Proposed Method (Deep Learning)

3.3.1. Keystroke Dynamics Authentication Process

Conceptual framework refers to a tool which illustrates the abstract representation of a study. It is useful to organize ideas concerning the problem to be addressed in a research. The Figure 3.2 below shows the conceptual framework for this study of keystroke dynamics as biometric user authentication.

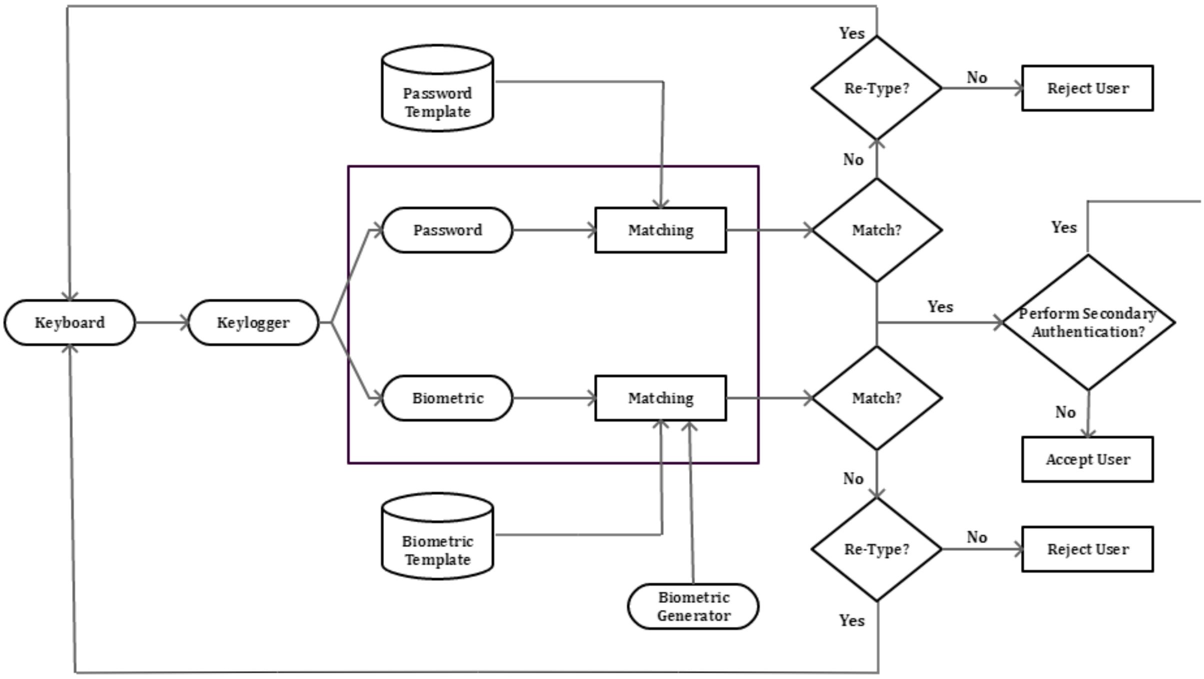


Figure 3.2: Keystroke Dynamics Authentication Process.

The process flow of biometric authentication using keystroke dynamics can be defined in several phases. First of all, in order to access a user account, the user will need to follow authentication procedure which is implemented in the system (in this case, using the keyboard). A passphrase inputted from the user will be captured along with its keystroke dynamics pattern information. This capturing process is done by using a keylogger, a software which is capable of recording activities related to the keyboard. By this step, the system will obtain both passphrase information and keystroke dynamics timing features. Next, this information will be compared separately with the defined templates (one for a passphrase, and one for biometric). These templates store the genuine user information such as the correct passphrase and typing rhythm. If one of the matching processes fails, the system will ask the user to perform the typing task again or to stop the authentication procedure. Otherwise, if both matching processes succeed, the system will recognize the user as a genuine user and accept the access into the system, or to perform a secondary authentication procedure. These processes summarise the biometric authentication procedure by using keystroke dynamics.

3.3.2. Deep Learning Architectures

Machine learning has been used to solve problems in several disciplines such as business, medical science, computer science, artificial intelligence and so on. It has the capability to recognize a pattern in the data and develop intelligent decision. Shallow learning techniques such as SVM, logistic regression, Multi-Layer Perceptron (MLP) with a single hidden layer, GMM, and others have been used to solve a lot of simple problems. However, it has been proven to be less effective on a complex problem which requires a higher level of modelling and representational capabilities (Deng & Yu, 2014). Deep learning has received more attention recently as it is able to overcome shallow learning's problem, which can be done by using multiple layers for feature learning. An instance of deep learning method is a deep neural network (MLP with more than one hidden layer). There are three categories of a deep network according to its architecture and implementation such as:

- Supervised learning deep networks, which capable of differentiating posterior distributions of classes in the data for pattern classification tasks.
- Unsupervised learning deep networks, which capable of discover high-order correlation in the data without specified class labels for pattern analysis tasks.
- Hybrid deep networks, which uses optimization and/or regularization to improve the discrimination performance in unsupervised deep networks, or to estimate unsupervised learning parameters by using discrimination features from supervised learning.

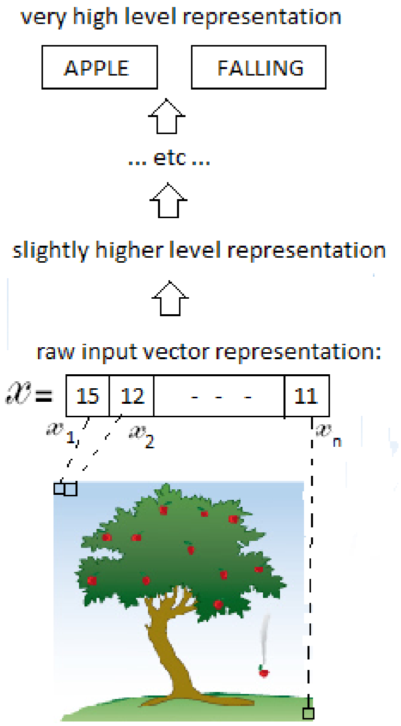

very high level representation
APPLE
FALLING
... etc ...
slightly higher level representation
raw input vector representation:
x =
15
12
- - -
11
x_1
x_2
x_n

Figure 3.3: Example of deep learning architectures representation.

The Figure 3.3 above illustrates the level of abstraction and representation in deep architectures. The main task is to convert the raw input image into very high-level representation "APPLE" by going through several layers of representation. Initially, the model recognizes the input image as a collection of object and arranges them in vector representation. Then, it gradually improves the abstraction by developing functions such as edges, shapes, and so on. The main purpose of deep learning is to automate the process of finding high-level representation from low-level features (Bengio, 2009). Deep learning offers several benefits compared to shallow learning. Firstly, deep learning allows selection and learning of all features in depth *k* architecture which may be too complex to be represented by shallow learning (Bengio, 2009). Secondly, deep learning architecture is able to perform multi-task learning in which features and functions can be re-used by multiple tasks in the learning process. This is possible due to its multi-level structure, and sparsity characteristic of the architecture which increases the representation efficiency by only utilizing up to 4% of the neurons (Deng & Yu, 2014). Thus, deep learning is able to optimize the parameters used in a study to improve its representation.

3.3.3. Deep Neural Networks

Generally, the difference between deep neural network and the shallow neural network can be seen based on the number of layers (input, hidden, and output) within the architectures. The deep neural network has more than one hidden layer, and this increases the network capability to handle complex and non-linear functions which are difficult to handle by traditional shallow machine learning algorithms. In order to understand how deep neural network works, basic concepts about the traditional neural network are discussed in this section.

3.3.3.1. Basic Concepts of Neural Network

Neural network is a machine learning model which performs similarly to human brain and consists of three layers (input, hidden and output). An illustration of the neural network can be seen in Figure 3.4. According to Kumar (2004), there are two types of neural network architecture: feed-forward network and feedback network. The former one sends information from one layer to another layer without looping back to the previous layer whereas the latter one performs the same activities as feed-forward but allows loop between the layers. Neural network learns by using neurons, which are connected with these layers by weighted signal links in order to process information across the network. The information is collected from input layers and processed by using mathematical calculation until the output is found. According to Heaton (2015), for each i number of x inputs, the w weights of each input are multiplied and fed into ϕ activation function to decide the output. The formula for the computation can be seen as follows:

$$f(x_i, w_i) = \phi(\sum_i (w_i \cdot x_i) \qquad (3.1)$$

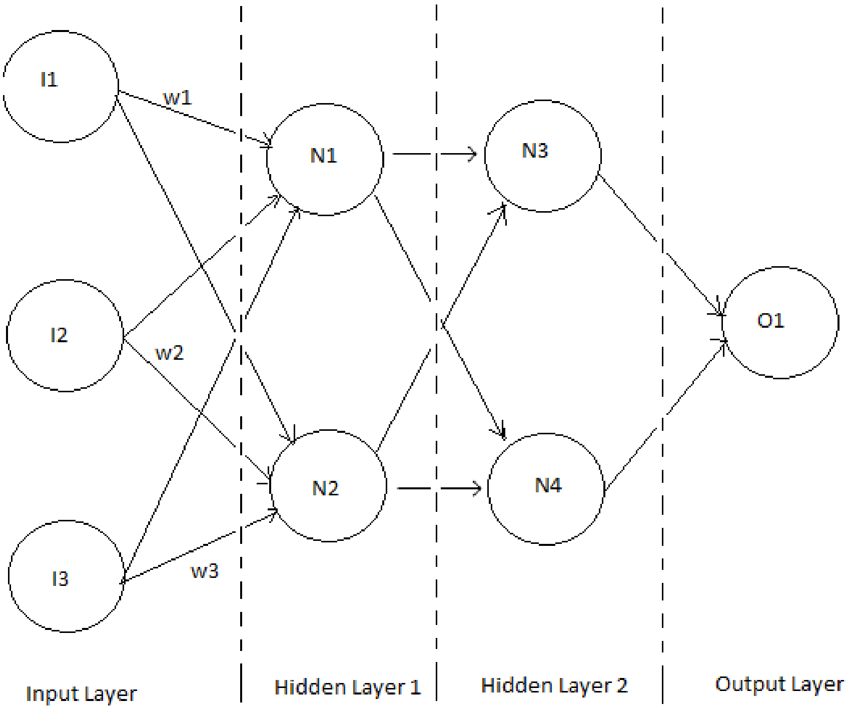


Figure 3.4: Illustration of a neural network with input layer, hidden layers, and output layer (adapted from Heaton, 2015).

3.3.3.2. Activation Function

Activation function refers to a rule to specify the output of the input calculations, and is located between the final layer and output layer. It uses a threshold value in order to classify the calculation result and to determine which class the input belongs to. Some neural network includes bias neurons (input neuron with a value of 1) in order to improve the pattern learning process and to obtain expected output by altering the activation function to be an upper threshold or lower threshold. Some of the frequently used activation functions are stated in the following formula and Figure 3.5 (Heaton, 2015):

1. Sigmoid Activation Function

$$\Phi(x) = \frac{1}{1+e^{-x}} \tag{3.2}$$

2. Hyperbolic Tangent Activation Function

$$\Phi(x) = \tanh(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}} \tag{3.3}$$

3. Rectified Linear Units (ReLU)

$$\phi(x) = \max(0, x) \tag{3.4}$$

4. Softmax Activation Function

$$\phi_i = \frac{e^{z_i}}{\sum_{j\in group} e^{z_i}} \tag{3.5}$$

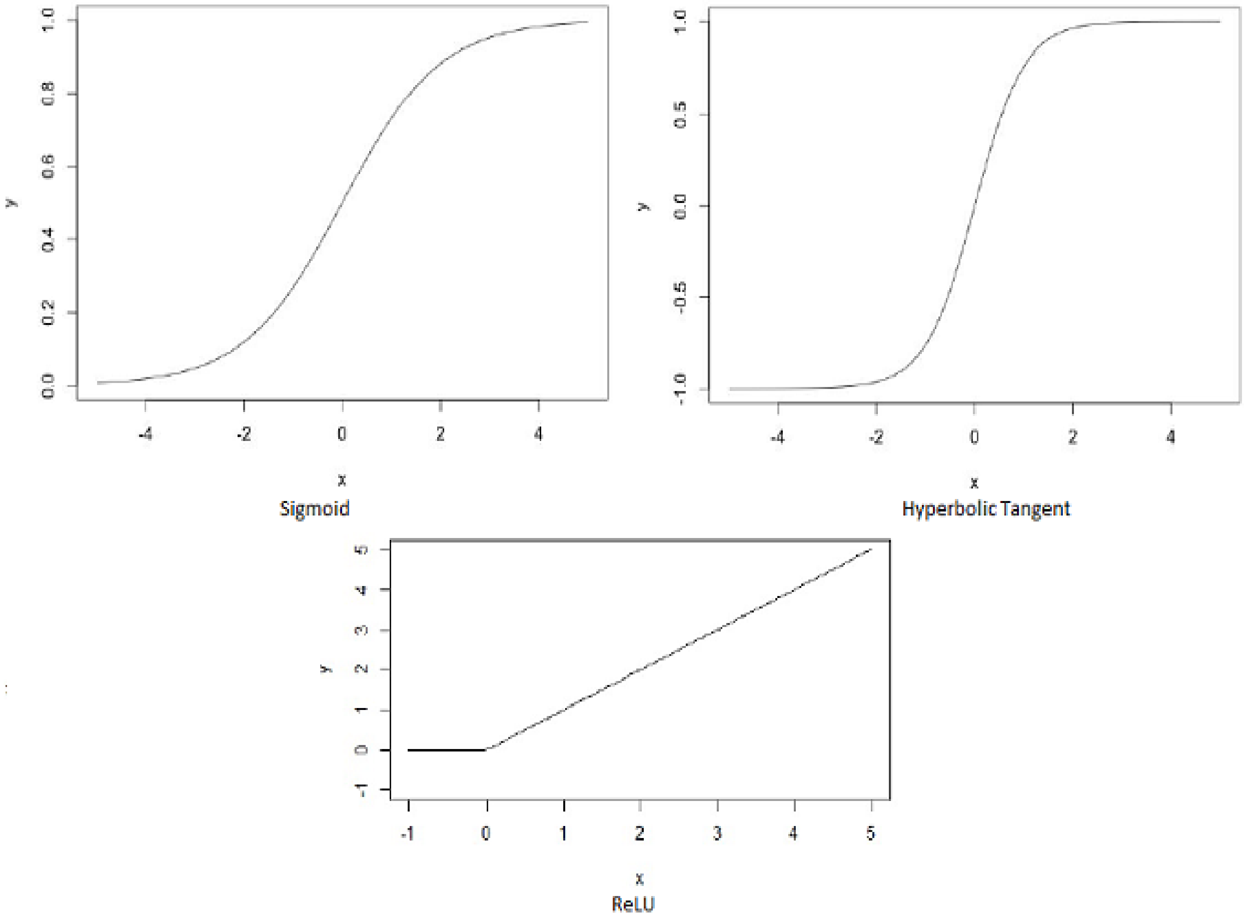


Figure 3.5: Plots of Activation Functions

3.3.3.3. Multilayer Perceptron

Multilayer Perceptron, or also known as a multilayer feed-forward neural network refers to a network model in which each neuron in a layer is connected with neurons from another layer without cycling back to previous layer (Kumar, 2004). It consists of an input layer, one or more hidden layer, and an output layer. The neural network illustration in Figure 3.4 is also considered as MLP, or three-layer neural network because it consists of two hidden layers. Input layer consists of neuron which receives the input values (either numerical or binary) from training tuple. These inputs contain a weight assigned to each of them, which will be carried on to the next layer called hidden layer. It receives the input values from input layer, performs the mathematical calculation, and generates a temporary output for each training tuple which has entered the network. Next, these outputs are sent to an output layer where the predicted value for each training tuple will be assigned accordingly to the type of embedded activation function.

Although usually only one hidden layer is used in neural network, its number can increase arbitrarily. As previously discussed, when a network contains more than one hidden layer, it can be called as deep network. (Bishop, 2006) suggested the use of non-linear activation function in deep learning in order to handle the composition of continuous linear transformation. By using matrix multiplication, the non-linear function can reproduce numbers of linear transformations in a single layer.

In this study, the multilayer perceptron model using deep learning is built by using one input layer, two hidden layers, and one output layer. For input layer, the number of neurons is set to thirty-one units as corresponding to the number of input features in the dataset. For hidden layers, the number of neurons is set to twenty-three units. This number is selected based on trial and error in finding the optimal accuracy for the classifier. However, Blum (1992) explained that the rule of thumb in deciding the number of units in a hidden layer is to choose a number between units in the input layer and the output layer. Thus, the median number between thirty-one and fifteen is selected. For output layer, the amount of neuron is set to fifteen units because the classifier returns genuine user or impostor information for fifteen users.

3.3.3.4. Weight Initialization

A weight in a neural network refers to the relevance of a feature in the model. It carries a value for each neuron in the network. There is two weight initialization methods used in this study such as Xavier and ReLU. Xavier is used for input and hidden layers, whereas relu is used for the output layer. Xavier initialization refers to a technique that assigns weight by considering the learning effect of the neurons in order to maintain equal distribution of activations (Glorot & Bengio, 2010). The formula for Xavier weight initialization is described as follows:

$$Var(W_i) = \frac{2}{n_{in} + n_{out}} \tag{3.6}$$

3.3.3.5. Gradient Descent

Gradient descent or also known as steepest descent refers to an optimization where the best weight value is used to optimize the network output. The algorithm works by initially assign random probability values into the weights in order to compute the gradients. Next, the function is moved according to the learning rate into the direction of the negative gradient slowly and iteratively until the best function is found. Finally, the weights are normalized in order to achieve total probability value equals to 1 (Han, Kamber & Pei, 2012). With w weights and η learning rate > 0, the formula for gradient descent can be represented as follows:

$$w^{(\tau+1)} = w^{(\tau)} - \eta \nabla E(w^{(\tau)}) \tag{3.7}$$

The goal of gradient descent algorithm is to find the local minimum and global minimum of the function in order to minimize the error and the cost of a function (Bishop, 2006). This can be done by performing multiple gradient descent algorithms using different starting point and by finding the best performance of the algorithm towards the function.

Stochastic gradient descent (SGD) is similar to gradient descent, however only one training sample is used in the parameter update thus leads to faster implementation. Besides, momentum can also be used to increase the learning speed of the network. Momentum refers to an acceleration technique for gradient descent in accumulating convergence towards local minimum (Sutskever et al., 2013). The formula for momentum can be described as follows:

$$v_{t+1} = \mu v_t - \eta \nabla f(\theta_t) \tag{3.8}$$

$$\theta_{t+1} = \theta_t + v_{t+1} \tag{3.9}$$

where *f(θ$_t$)* is minimized objective function, μ ϵ [0, 1] is momentum coefficient, *∇ f(θ$_t$)* is gradient at *θ$_t$*, and *η* is learning rate > 0.

3.3.3.6. Error Backpropagation

The error of a function can be found by using error's derivatives evaluation technique called backpropagation. Backpropagation works by iteratively processing training tuples and comparing it with each other to obtain the best target value (class label for classification problems or continuous value for numeric prediction), by modifying the weight for each training so that the minimum mean-squared error is achieved (Han, Kamber & Pei, 2012). The idea of backpropagation is to make the weight modification in a backward direction from output layer to hidden layer. The error calculation of backpropagation algorithm can be seen in Figure 3.6 below. By knowing O_j actual output of unit *j* and T_j known target value for given training tuple, the error of each network prediction can be calculated by using the following formula:

$$Err_j = O_j(1 - O_j)\left(T_j - O_j\right) \tag{3.10}$$

Also, in order to calculate the error in the hidden layer of unit *j*, the weighted sum of errors of w_{jk} connection weight is calculated for each unit connected to unit *j* to unit *k*, as shown in the following formula:

$$Err_j = O_j(1 - O_j)\sum_k Err_k w_{jk} \tag{3.11}$$

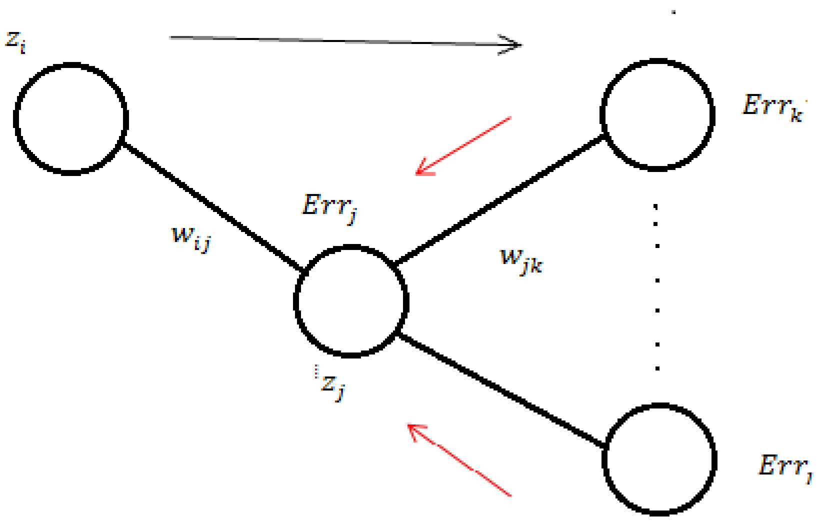


Figure 3.6: Illustration of backpropagation error calculation, where z_i and z_j is the activation function for unit *i* and unit *j* (adapted from Bishop (2006)).

3.3.3.7. Output and Error Function

Gradient descent and backpropagation algorithm can be used to minimize the error of a function so that it will lead to more accurate prediction. However, the error calculation depends on the type of activation function used in the layer and also on the type of problem to be addressed (Bishop, 2006). There are three general problems which can be solved by using neural network:

- Regression problem covers the calculation of linear combination of inputs by using linear activation function which is suitable for the problem as it does not have specific output range constraints. Based on $h_{w,b(x)}$ network output, the sum-of-squares error function for regression problem can be seen in the formula below:

$$SSE(\theta) = \frac{1}{2}\sum_{i=1}^{n}\left|h_{w,b}(x_i) - y_i\right|^2 \qquad (3.12)$$

- Binary classification problem covers the classification of two classes where the output unit is represented as a binary class label. For binary classification, the general activation function which can be used is a logistic sigmoid function or hyperbolic tangent function. The cross-entropy error function for binary classification problem can be seen in the formula below:

$$CE(\theta) = -\sum_{i=1}^{n}\{y_i \ln h(x_i) + (1 - y_i)\ln(1 - h(x_i))\} \qquad (3.13)$$

- Multi-class classification problem covers the classification of k classes (more than two) where the network has k output units. Softmax activation function can be used in the last network layer to solve multi-class classification problem. The multi-class cross-entropy error function for multi-class classification problem can be seen in the formula below:

$$MCE(\theta) = \sum_{t=1}^{n} \sum_{j=1}^{k} \delta_{y_i j} \ln \sigma_j \tag{3.14}$$

where δ_{yij} refers to Kronecker delta:

$$\delta_{y_i j} \begin{cases} 0 & if\ y_i \neq j \\ 1 & if\ y_i = j \end{cases} \tag{3.15}$$

3.3.3.8. Regularization

Regularization refers to the process of modifying algorithm in order to minimize generalization error of a model. Regularization can be done by adding restrictions on parameter values of a model, or by using additional rules on the objective function (Goodfellow, Bengio & Courville, 2016). For instance, determining an optimal number of hidden units for a network will improve the parameter adjustment and model capacity to avoid overfitting. Overfitting is a situation where a model learns too much detail on training data and unable to perform good prediction on new data. Bengio (2012) suggested an implementation of regularization term (also known as weight decay) to error function to avoid overfitting as shown in the following formula:

$$\bar{E}(\theta) = E(\theta) + \lambda \sum_t (\theta_t)^2 \tag{3.16}$$

where λ refers to regularization coefficient. Besides regularization term, early stopping algorithm is also frequently used to avoid overfitting. It works by configuring parameter to obtain the lowest validation set error iteratively and the algorithm stops when the parameter could not be improved after certain iterations or when the performance starts to decrease (Goodfellow, Bengio & Courville, 2016). Early stopping can avoid damaging the learning dynamics of a model as it only makes minimum changes in the training procedure, unlike in regularization term where overusing it may generate bad local minimum in the model.

3.4. Summary

As a summary, the methodology for the implementation of this study is discussed. As part of the quantitative research, KDD will be used in the research design in order to perform numerical analysis or computational techniques towards the collected data. Next, a keystroke dynamics authentication process is developed by using deep learning. In addition, the classification method for keystroke dynamics by using deep learning is elaborated. The network model used in the deep learning is multilayer perceptron with two hidden layers. Stochastic gradient descent algorithm will be used as the optimization technique as it is able to minimize the error and the cost of a function. An acceleration technique for gradient descent called momentum will be used to increase the learning speed of the network. To calculate the error of the function, backpropagation algorithm will be used. For weight initialization, Xavier initialization will be used. It is a technique that assigns weight by considering the learning effect of the neurons in order to maintain equal distribution of activations. Next, there are two activation functions used in the network: relu for hidden layers and softmax for the output layer. The model will use multi-class cross-entropy as its output function.

CHAPTER 4

MODEL DEVELOPMENT

4.1. Introduction

In this chapter, the processes involved in the model development for keystroke dynamics study will be discussed. First of all, the variables in the original dataset used in this study will be explained. Next, data exploration and data visualization will be conducted towards the data in order to identify interesting patterns from the dataset. Based on these processes, the original dataset will be reduced. In addition, the features used in the model development will also be discussed. This is important because the literature review has shown that there is no standard for the type of feature which needs to be used. Furthermore, the tool required for model development will be introduced along with the configuration required to fulfil the requirements of the deep learning method.

4.2. Explanation of the Dataset

The dataset used in this study is Keystroke Dynamics Benchmark Dataset (CMU) which is provided by Killourhy (2012) in his research on comparing anomaly detectors in keystroke dynamics. It consists of a subject identifier (ID) variable, session number, repetition number and 31 keystroke timing features (H, DD, and UD) collected from 51 users who were asked to type a strong password (‘.tie5Roanl) for 8 sessions with 50 typing repetitions for each session, which lead to a total of 34 variables and 20400 observations. These timing features are recorded in measurement unit of second. The brief explanation of the dataset can be seen in Table 4.1 below.

Table 4.1: Summary of the Dataset

No.	Variables	Details
1	Subject	Subject ID or class label for 51 users involved in typing task.
2	sessionIndex	Number of session in the typing task; consists of 8 sessions in total.
3	Rep	Number of repetition in the typing task; consists of 50 repetitions for each session.
4	H.period	Duration between pressing and releasing '.' key.
5	DD.period.t	Duration between pressing '.' key and pressing 't' key.
6	UD.period.t	Duration between releasing '.' key and pressing 't' key.
7	H.t	Duration between pressing and releasing 't' key.
8	DD.t.i	Duration between pressing 't' key and pressing 'i' key.
9	UD.t.i	Duration between releasing 't' key and pressing 'i' key.
10	H.i	Duration between pressing and releasing 'i' key.
11	DD.i.e	Duration between pressing 'i' key and pressing 'e' key.
12	UD.i.e	Duration between releasing 'i' key and pressing 'e' key.
13	H.e	Duration between pressing and releasing 'e' key.
14	DD.e.five	Duration between pressing 'e' key and pressing 'five' key.
15	UD.e.five	Duration between releasing 'e' key and pressing 'five' key.
16	H.five	Duration between pressing and releasing '.' key.
17	DD.five.shift.r	Duration between pressing 'five' key and pressing 'shift.r' key.
18	UD.five.shift.r	Duration between releasing 'five' key and pressing 'shift.r' key.
19	H.shift.r	Duration between pressing and releasing 'r' key.
20	DD.shift.r.o	Duration between pressing 'shift.r' key and pressing 'o' key.
21	UD.shift.r.o	Duration between releasing 'shift.r' key and pressing 'o' key.
22	H.o	Duration between pressing and releasing 'o' key.
23	DD.o.a	Duration between pressing 'o' key and pressing 'a' key.
24	UD.o.a	Duration between releasing 'o' key and pressing 'a' key.
25	H.a	Duration between pressing and releasing 'a' key.
26	DD.a.n	Duration between pressing 'a' key and pressing 'n' key.
27	UD.a.n	Duration between releasing 'a' key and pressing 'n' key.
28	H.n	Duration between pressing and releasing 'n' key.
29	DD.n.l	Duration between pressing 'n' key and pressing 'l' key.
30	UD.n.l	Duration between releasing 'n' key and pressing 'l' key.
31	H.l	Duration between pressing and releasing 'l' key.
32	DD.l.return	Duration between pressing 'l' key and pressing 'return' key.
33	UD.l.return	Duration between releasing 'l' key and pressing 'return' key.
34	H.return	Duration between pressing and releasing 'return' key.

4.3. Data Preparation

The CMU dataset does not have any missing value, but some outliers could be found for several timing features. These outliers might occur because each participant has a different style and efficiency of typing a keyboard. For instance, a participant who has a job or experience related to typing task should be able to type quicker than those who do not have one. Unfortunately, CMU does not provide information on the typing efficiency for the participants.

In order to learn more about the data, and to find new insight related to the data, two data analysis techniques will be used in this study. The first one is to generate the descriptive statistics of the data in order to see the pattern for each keystroke timing feature. The second one is to generate visual representation of the dataset to gain new insights related to the correlation and outlier, as presented in section 4.4.

4.3.1. Descriptive Statistics

In information collection, descriptive statistics or also known as summary statistics refers to a method to summarize or describe features in the dataset in a quantitative manner. Generally, descriptive statistics consists of measures of central tendency (mean, median, and mode) and measures of dispersion (standard deviation, variance, minimum value and maximum value). The descriptive statistics of CMU dataset is represented in Table 4.2 below.

Several interesting observations are highlighted in the table. First of all, it can be seen that DD.five.shift.r and UD.a.n have the highest and the lowest mean among all features respectively. This means most of the participants have difficulty in typing number and uppercase letter consecutively. The inference is also supported by the values of median and mode of DD.five.shift.r, which is the highest among all other features. Whereas for the feature with the lowest mean, it could be resulted by the position of the keyboard keys which made it easier to type key ‘a’ and key ‘n’ consecutively while typing with both hands. This is supported by the lowest mode belonged of the feature. Next, all the features have considerably low standard errors, standard deviation, and variance, which means the values tend to close to the mean of the dataset. Furthermore, it can be seen that some features have negative median and minimum values, which indicate overlapping in typing task. At last but not least, the maximum values of DD.i.e and UD.i.e are significantly higher compared to other features. This might occur because of the user being idle (taking a break) during typing task.

Table 4.2: Descriptive Statistics for CMU Dataset Keystroke Timing Features (Overall)

Features	Mean	Std. Error	Median	Mode	Std. Deviation	Variance	Min	Max
H.period	0.0934	0.0002	0.0895	0.0834	0.0296	0.0009	0.0014	0.3761
DD.period.t	0.2641	0.0015	0.2059	0.1184	0.2205	0.0486	0.0187	12.5061
UD.period.t	0.1707	0.0016	0.1087	0.0013	0.2268	0.0515	**-0.2358**	12.4517
H.t	0.0857	0.0002	0.0081	0.076	0.0274	0.0008	0.0093	0.2411
DD.t.i	0.1691	0.0009	0.1404	0.1175	0.1235	0.0153	0.0011	4.9197
UD.t.i	0.0834	0.0009	0.0578	0.0011	0.1258	0.0158	**-0.1621**	4.7999
H.i	0.0816	0.0002	0.0771	0.0681	0.0269	0.0007	0.0032	0.3312
DD.i.e	0.1594	0.0016	0.1209	0.0795	0.2269	0.0515	0.0014	**25.9873**
UD.i.e	0.0778	0.0016	0.0412	0.0014	0.2285	0.0522	**-0.16**	**25.9158**
H.e	0.0891	0.0002	0.0834	0.0723	0.0306	0.0009	0.0021	0.3254
DD.e.five	0.3774	0.0019	0.289	0.2027	0.2653	0.0704	0.0013	4.9618
UD.e.five	0.2283	0.0019	0.2004	0.129	0.2667	0.0711	**-0.1505**	4.8827
H.five	0.0769	0.0002	0.0742	0.0697	0.0217	0.0005	0.0014	0.1989
DD.five.shift.r	**0.4389**	0.0018	**0.3775**	**0.3688**	0.2603	0.0678	0.1694	8.3702
UD.five.shift.r	0.3620	0.0018	0.302	0.3197	0.2609	0.0681	0.0856	8.2908
H.shift.r	0.0959	0.0002	0.0935	0.0512	0.0339	0.0011	0.0014	0.2817
DD.shift.r.o	0.2509	0.0012	0.2014	0.146	0.1745	0.0305	0.0494	4.1523
UD.shift.r.o	0.1550	0.0013	0.1022	**-0.0014**	0.1816	0.033	**-0.0865**	4.012
H.o	0.0884	0.0002	0.0863	0.0855	0.0264	0.0007	0.0069	0.6872
DD.o.a	0.1569	0.0007	0.1316	0.1096	0.1066	0.0114	0.0012	2.8567
UD.o.a	0.0686	0.0008	0.0444	**-0.0013**	0.1085	0.0118	**-0.2287**	2.8152
H.a	0.1063	0.0003	0.1019	0.09	0.0388	0.0015	0.004	2.0353
DD.a.n	0.1507	0.0008	0.125	0.099	0.1074	0.012	0.0011	3.3278
UD.a.n	**0.0444**	0.0008	0.0227	**0.0011**	0.1052	0.0111	**-0.2355**	2.5242
H.n	0.0899	0.0002	0.0853	0.0813	0.0307	0.0009	0.0037	0.3577
DD.n.l	0.2026	0.0011	0.1725	0.1689	0.1502	0.0226	0.0013	4.0252
UD.n.l	0.1127	0.0011	0.0955	**-0.0011**	0.1596	0.0255	**-0.1758**	3.9782
H.l	0.0956	0.0002	0.0937	0.0942	0.0283	0.0008	0.0037	0.3407
DD.l.return	0.3218	0.0016	0.263	0.241	0.2254	0.0508	0.0083	5.8836
UD.l.return	0.2263	0.0016	0.1603	0.1128	0.2308	0.0533	**-0.1245**	5.8364
H.return	0.0883	0.0002	0.0855	0.0871	0.0275	0.0008	0.0029	0.2651

4.4. Exploratory Data Analysis

Exploratory Data Analysis (EDA) refers to a technique that is used to gain better understanding of the data, to explore and discover patterns which may be unseen previously, and also to find new insights related to the data. Usually, EDA is conducted by generating a visual representation of the data. There are two types of visual representation such as: explanatory and exploratory. Explanatory aims to communicate insights and messages existed in the data to the viewers, whereas exploratory aims to discover hidden patterns in the data. In this study, the latter one will be used to compare the feature patterns between 51 users. The visualization is done by using Microsoft Power BI tool which provides an interactive view of the data and delivers insight related to the data (Microsoft, 2017). The result of this visualization is grouped into several categories such as correlation, outlier, uppercase timing, and symbols timing.

4.4.1. Correlation

Correlation represents a linear relationship between two variables. It is useful to understand which variable is strongly related to another variable. In this study, a total of 55 correlations is found from the insights generated by the tool. The three correlations with the highest frequency are illustrated in the following figures:

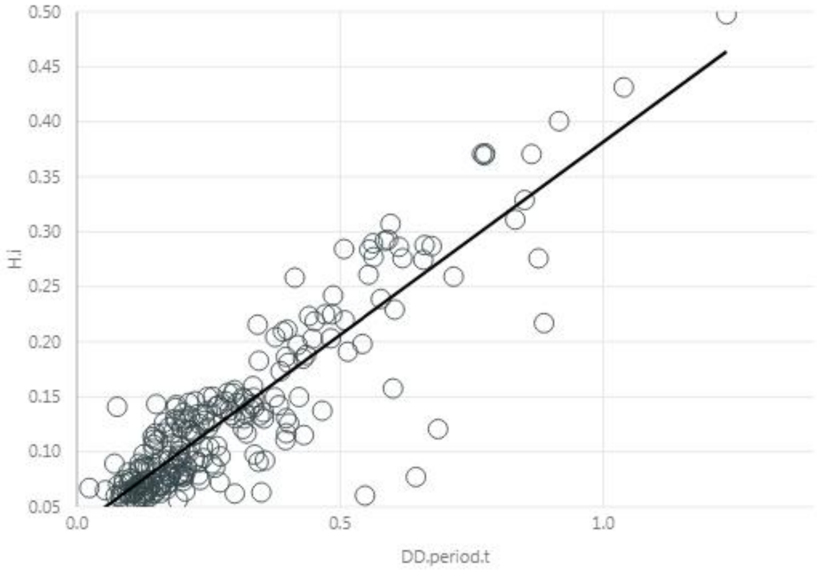


Figure 4.1: Correlation between DD.period.t & H.i.

Figure 4.1 above illustrates the relationship between the duration of pressing key '.' to key 'T' and holding key 'I' can appear because of key 'I' is pressed after key 'T'. This correlation appears five times in the dataset for subject S002, S004, S010, S029, and S055.

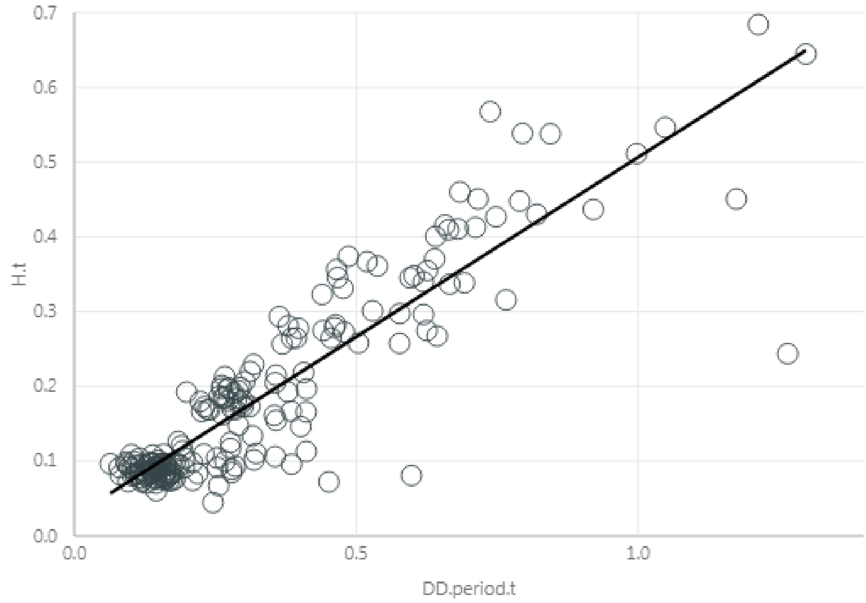


Figure 4.2: Correlation between DD.period.t & H.t.

Figure 4.2 above illustrates the relationship between the duration of pressing key '.' to key 'T' and holding key 'T' can appear because of key 'T' is pressed after key '.'.This correlation appears three times in the dataset for subject S018, S052 and S056.

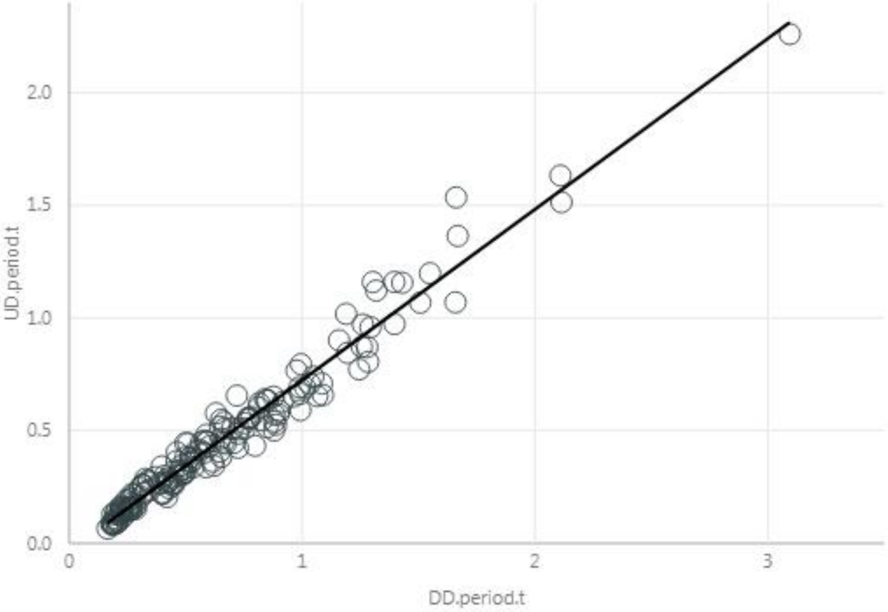


Figure 4.3: Correlation between DD.period.t & UD.period.t.

Figure 4.3 above illustrates the relationship appears because DD.period.t is a sum of UD.period.t and H.period. This correlation appears thirty-three times in the dataset for subject S005, S007, S008, S013, S015, S016, S017, S019, S020, S021, S024, S025, S026, S027, S028, S030, S031, S033, S035, S037, S038, S039, S040, S041, S042, S043, S044, S046, S047, S049, S050, S053 and S054.

The rest of correlations in the dataset are summarized in Table 4.3 below.

Table 4.3: Summary of Correlation by Keystroke Features

Correlation	Frequency	Subject
DD.e.five & UD.e.five	2	S003, S038
DD.period.t & DD.i.e	2	S034, S057
DD.period.t & DD.t.i	2	S032, S048
DD.period.t & H.five	2	S011, S036
DD.i.e & UD.i.e	1	S003
DD.period.t & DD.five.shift.r	1	S051
DD.period.t & UD.t.i	1	S022
DD.t.i & UD.t.i	1	S031
H.t & H.e	1	S012
H.t & H.i	1	S003

4.4.2. Outlier

Outlier represents an observation which is far away from the other observations or clusters. There are two general causes of an outlier. The first one is variability in measurement, such as extreme values and high range values (difference between maximum value and minimum value) in the dataset. The second one is experimental error such as mistake in recording the data or wrong target participant in data collection process. In this study, the outlier obtained from the visualization process seems to follow the first cause of outlier. A total of 436 outliers are observed using the tool, and the five outliers with the highest frequency are illustrated in the following figures:

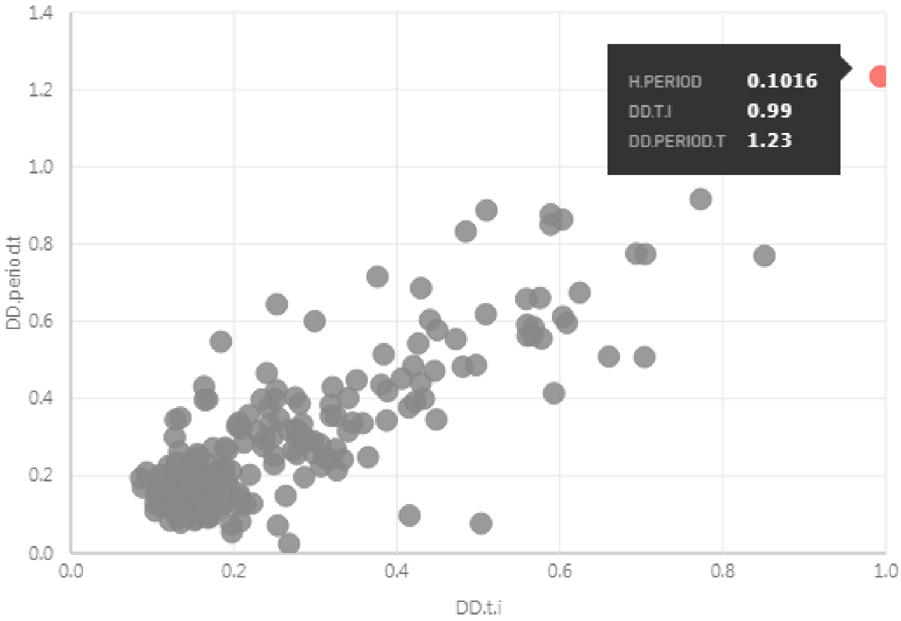


Figure 4.4: DD.t.i and DD.period.t have an outlier at H.period 0.1016.

From Figure 4.4 above, it can be seen that the value of DD.t.i 0.99 and DD.period.t 1.23 is higher than the boundary of the cluster. This outlier appears sixty-five times in the dataset.

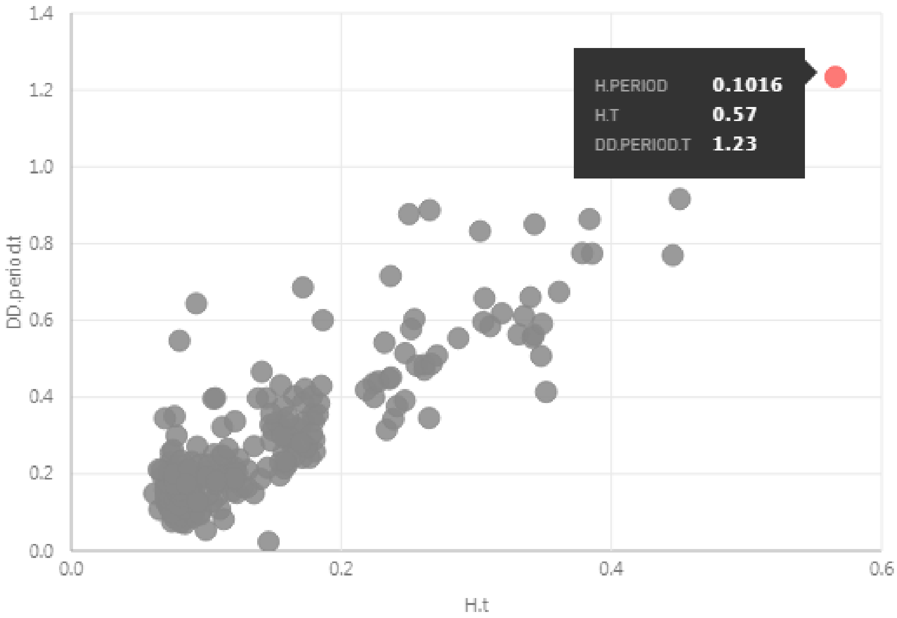


Figure 4.5: H.t and DD.period.t have an outlier at H.period 0.1016.

From Figure 4.5 above, it can be seen that the value of H.t 0.57 and DD.period.t 1.23 is higher than the boundary of the cluster. This outlier appears seventy-five times in the dataset.

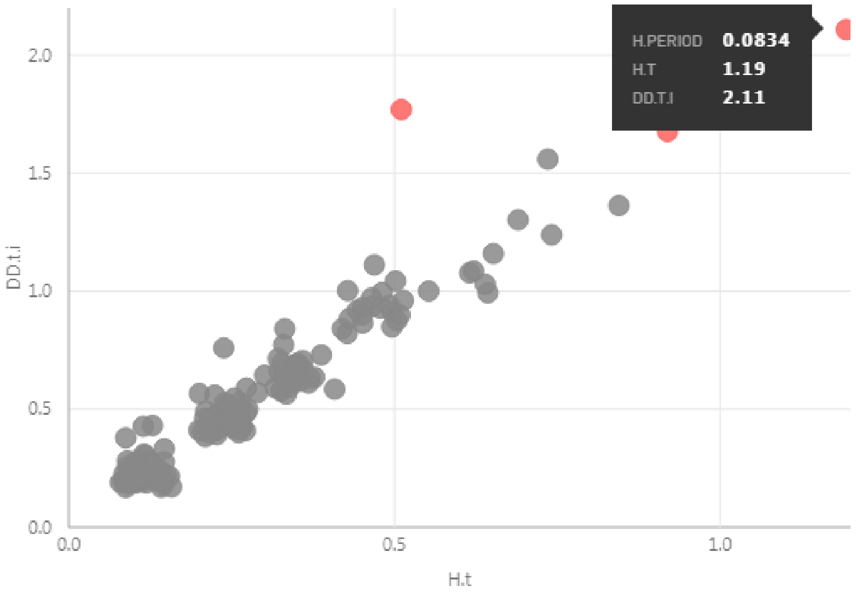


Figure 4.6: H.t and DD.t.i have an outlier at H.period 0.0834.

From Figure 4.6 above, it can be seen that the value of H.t 1.19 and DD.t.i 2.11 is higher than the boundary of the cluster. This outlier appears fifty-four times in the dataset.

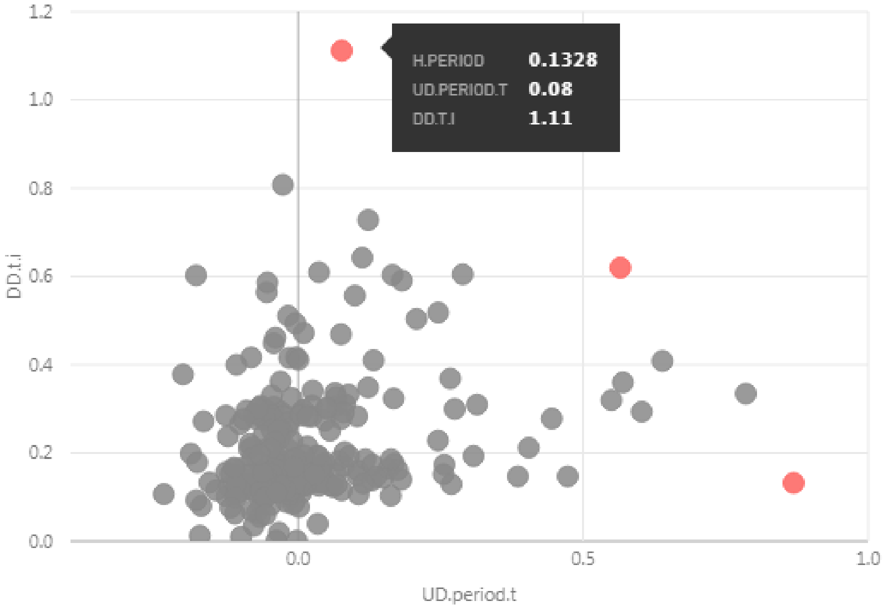


Figure 4.7: UD.period.t and DD.t.i have an outlier at H.period 0.1328.

From Figure 4.7 above, it can be seen that the value of UD.period.t 0.08 and DD.t.i 1.11 is higher than the boundary of the cluster. This outlier appears fifty-five times in the dataset.

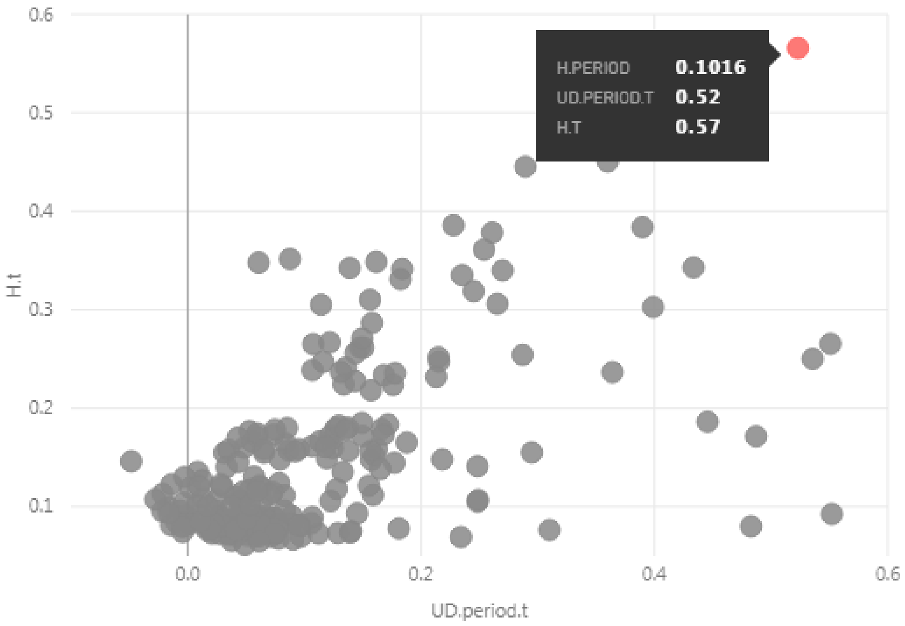


Figure 4.8: UD.period.t and H.t have an outlier at H.period 0.1016.

From the Figure 4.8 above, it can be seen that the value of UD.period.t 0.52 and H.t 0.57 is higher than the boundary of the cluster. This outlier appears seventy-four times in the dataset. The rest of the outliers are summarised in Table 4.4 below.

Table 4.4: Summary of Outliers by Keystroke Features

Outlier	**Frequency**	**Subject**
UD.t.i & UD.period.t	17	S011, S012, S033, S038, S048, S055
UD.i.e & DD.period.t	14	S015, S027, S041, S052, S054
UD.period.t & UD.i.e	11	S015, S027, S041
UD.t.i & DD.period.t	11	S031, S033, S038, S048, S055
UD.i.e & H.t	8	S027, S041, S052, S054
UD.t.i & H.t	7	S003, S033, S048
UD.period.t & DD.i.e	5	S015, S052
UD.t.i & DD.e.five	5	S003, S038
DD.t.i & DD.e.five	4	S003
UD.e.five & DD.e.five	4	S038
UD.t.i & DD.i.e	4	S003
UD.t.i & DD.t.i	4	S003, S031
H.t & DD.e.five	3	S003

Table 4.4: continued

Outlier	Frequency	Subject
UD.i.e & DD.i.e	3	S015
UD.period.t & UD.e.five	3	S004, S038
DD.period.t & DD.e.five	2	S003, S038
DD.period.t & DD.i.e	2	S015
UD.e.five & DD.period.t	2	S038
UD.t.i & UD.e.five	2	S038
UD.period.t & DD.e.five	1	S038
UD.t.i & UD.i.e	1	S055

4.4.3. Uppercase Timing

In uppercase timing, the target focus is keystroke features in the dataset which contains the duration of when key 'shift' is pressed and released. There are five features which are related to uppercase timing such as H.shift.r, UD.five.shift.r, UD.shift.r.o, DD.five.shift.r, and DD.shift.r.o. From the Figure 4.9 below it can be seen that subjects with ID number 49, 16, 22, 36, and 41 have the longest uppercase timing. Meanwhile, the subjects who have the shortest uppercase timing are recognized with ID number 55, 10, 7, 11, and 13. Besides, the outliers which are related to uppercase timing can be seen in Table 4.5 below. Since the number of outliers is considerably high, it can be concluded that uppercase letter significantly affects the typing performance of a user.

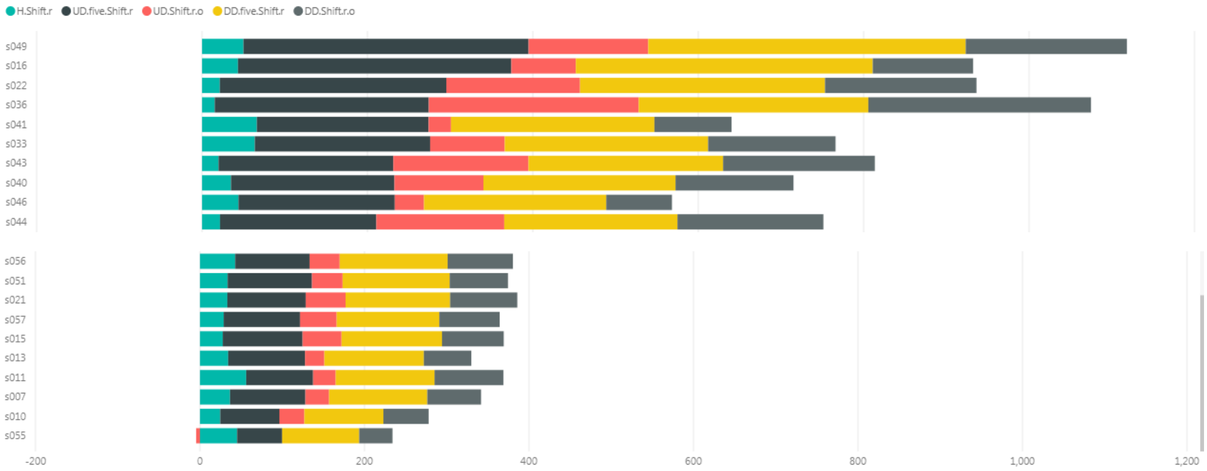


Figure 4.9: 10 subjects with highest and lowest uppercase timing.

Table 4.5: Summary of Outliers related to Uppercase Timing

Features	No. of Outliers
UD.shift.r.o and DD.shift.r.o	1
UD.five.shift.r and DD.five.shift.r	4
H.shift.r and DD.shift.r.o	8
DD.shift.r.o and DD.five.shift.r	6
UD.shift.r.o and DD.five.shift.r	3
UD.five.shift.r and DD.shift.r.o	1
UD.shift.r.o and H.shift.r	4

4.4.4. Symbol Timing

In symbol timing, the target focus is keystroke features in the dataset which contains the duration of when key ‘.’ is pressed and released. There are four features which are related to symbol timing such as H.period, UD.period.t, DD.period.t, and H.t. From the Figure 4.10 below it can be seen that the subjects with ID number 36, 52, 49, 22, and 16 have the longest symbol timing. Meanwhile, the subjects who have the lowest symbol timing are recognized with ID number 37, 10, 57, 55, and 13. Besides, from the analysis output of the tool it can be concluded that subject with ID number 52 has noticeably less UD.t.i duration compared to others. Besides, the outliers which are related to symbol timing can be seen in Table 4.6 below. Since all outlier is related to both ‘.’ and ‘shift’ key, this indicates the occurrence of the symbol in typing task can affect the uppercase timing feature.

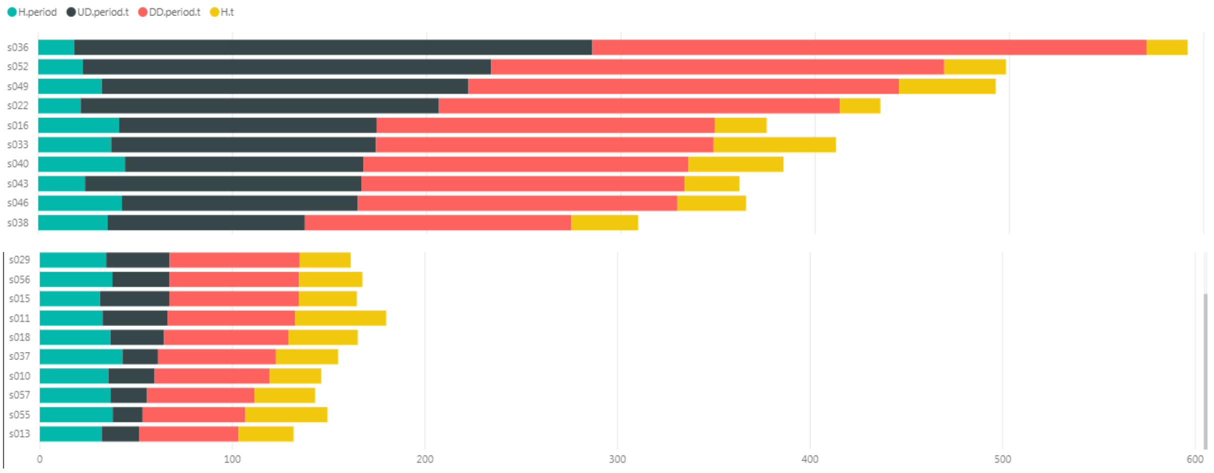


Figure 4.10: 10 subjects with highest and lowest symbol timing.

Table 4.6: Summary of Outliers related to Symbol Timing

Features	No. of Outliers
DD.period.t and DD.five.shift.r	1
DD.shift.r.o and DD.period.t	2
H.shift.r and DD.period.t	1

4.5. Data Sampling

The final dataset used in the model development consists of 32 columns and 6000 rows. In the previous chapter, data exploration and visualization has been performed to the original CMU dataset and only fifteen out of fifty-one participant data are selected. The selection process of the data is divided into three criteria: (i) five user data which have the most outliers; (ii) five user data which have the least outliers; and (iii) five user data which have the median amount of outliers. Besides, sessionIndex and repetition are removed from the dataset.

4.6. Feature Selection

The keystroke timing features used in the model development are H, UD, and DD. These features are extracted from the CMU dataset and feature reduction could not be performed as the number of the feature is considered small. Thus, these features which hold the timing for each key-press of ten characters password are kept only for fifteen selected participants.

4.7. Model Implementation

The keystroke dynamics model implementation was carried out using Walkato Environment for Knowledge Analysis (WEKA), which supports machine learning algorithms (Frank, Hall & Witten, 2016). In order to fulfil the deep learning technique as elaborated in Chapter 3, a package called Dl4jMlpClassifier will be used. It is a wrapper for an open-source deep learning programming library called Deeplearning4j which works by using Java virtual machine. This package allows the user to build a multi-layer perceptron deep learning model.

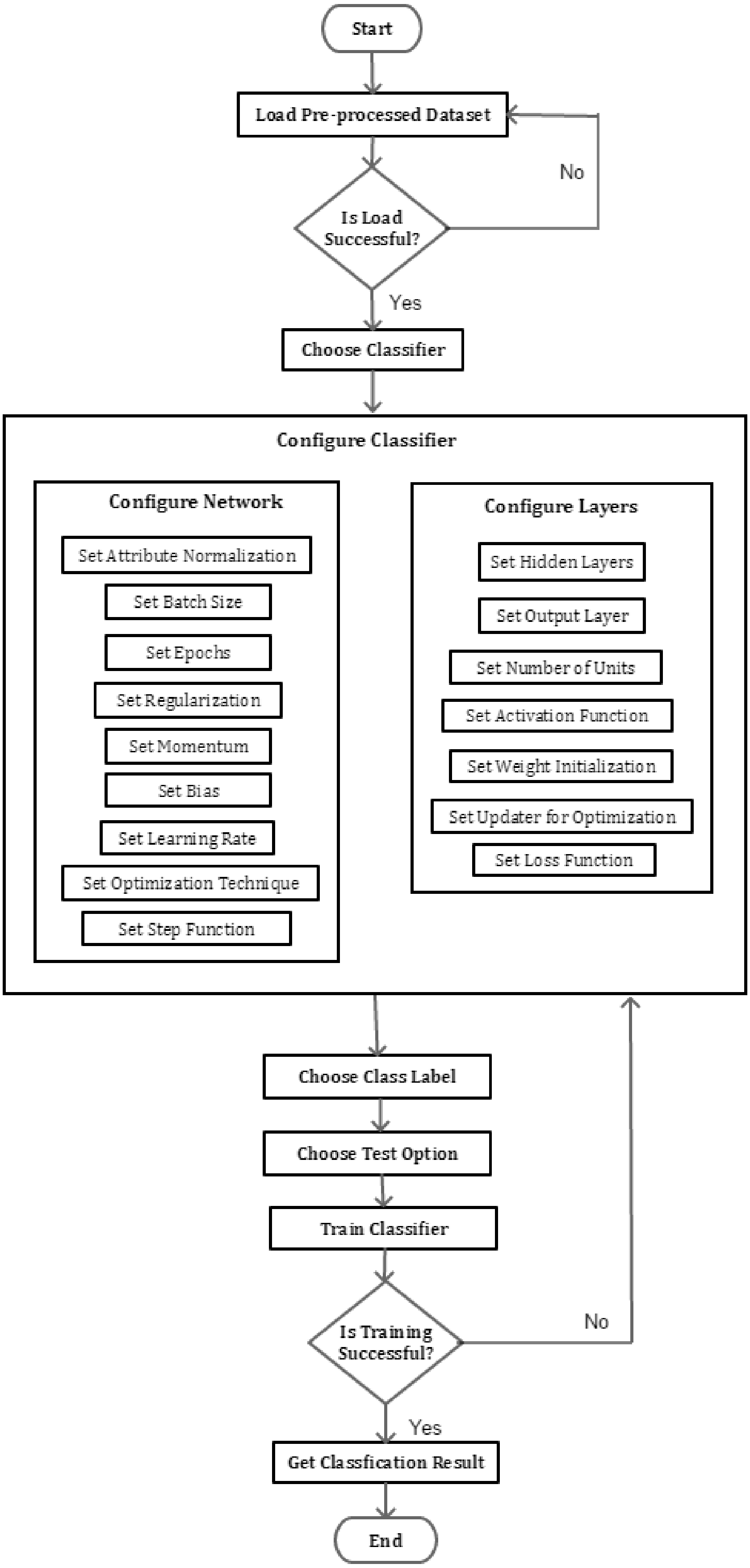


Figure 4.11: Deep Learning Model Implementation for Keystroke Dynamics.

The deep learning model implementation can be divided into seven phases such as load pre-processed dataset, choose classifier, configure classifier, choose class label, choose test option, train classifier, and get classification result. The detailed implementation of the model is illustrated in the Figure 4.11 above.

Load Pre-processed Dataset

The first phase of the model implementation is to load the dataset into the development tool. After the data has been successfully loaded, the details of the dataset will be displayed. The current dataset contains 6000 instances and 32 attributes, which is corresponding to the amount of features and subjects used in this study. The number of instances each subject is 400, which is corresponding to the amount of password typing repetitions. Although the tool allows the usage of filtering technique such as normalization, the process will not be done in this phase. Instead, it will be done in the next phases as the classifier provides option for regularization in its configuration.

Choose Classifier

The next phase after loading the dataset is to select the appropriate classifier for the deep learning implementation. As discussed in the previous section, the development tool provides a package called Dl4jMlpClassifier which is able to perform classification using deep MLP.

Configure Classifier

After the classifier has been chosen, there are some changes that need to be made in order to match the model configuration discussed in previous section. The development tool allows the user to configure the model easily through its interactive user interface. The deep network model configuration is divided into two steps: configure network and configure layers.

The changes for the network configuration are as follows:

- The number of epochs is set to ten by default. This number is changed to fifty.
- Attribute normalization is set to standardize by default. This option is changed to normalize.

- The batch size is set to one hundred by default. This number is changed to fifty. This means the network training iterates up to fifty times and uses fifty training samples per iteration.
- The optimization algorithm is set as SGD.
- The step function for optimization algorithm is set as NegativeGradientStepFunction.
- The regularization option is set as true.
- The remaining configuration remains as default.

The changes for layers configuration are as follows:

- The layer specification of the model is set to one (output layer) by default. Additional two dense (hidden) layers are added to the model.
- The number of units for the hidden layers is set to twenty-three.
- The activation function for hidden layers is set as ActivationRelu.
- The weight initialization method for hidden layers is set as XAVIER.
- The updater for Stochastic Gradient Descent for hidden layers is set as SGD.
- The remaining configuration for hidden layers remains as default.
- The loss function for output layer is set as LossMXCENT. It is a multi-class cross-entropy error function for the classifier.
- The activation function for output layer is set as ActivationSoftmax.
- The weight initialization method for output layer is set as RELU.
- The updater for Stochastic Gradient Descent for output layer is set as SGD.
- The remaining configuration for output layer remains as default.

Choose Class Label

After the model configuration has been completed, the next phase is to choose the class label which is important to train the classifier. Subject ID is chosen as the class label because it is the only nominal attribute which has the capability to differentiate the instances in the dataset. Thus, the classifier is trained to perform classification towards fifteen classes of users.

Choose Test Option

The last phase before the classifier is trained using the dataset is to choose the test option for model evaluation. The development tool allows four types of test option such as training set, the test set from another dataset, cross-validation, and train-test splitting. For this study, cross-validation with ten folds is chosen because it is more suitable for fewer amounts of data due to lesser variance output.

Train Classifier

By this phase, all configurations should have been completed. Thus, the training of the model can be started. The training of the model could take longer time depending on the configuration. For this implementation, the duration taken to complete the training is approximately thirty minutes.

Get Classification Result

After the model has been trained, a classification result will be given as the output. The interpretation and analysis of the output will be discussed in Chapter 5.

4.8. Summary

As a summary, the dataset preparation and model development processes for keystroke dynamics are discussed in this chapter. First of all, the keystroke dynamics dataset is briefly explained which includes the data collection process and the definition of the variables. Next, descriptive statistics and EDA of the dataset are generated in order to explore previously unseen pattern and to explore correlation and outlier that exists in the dataset. Based on these observations, the dataset is reduced by selecting only fifteen users based on the number of outliers existed for each user (Five users who have the lowest number of outliers, five users who have medium number of outlier, and five users who have the highest number of outliers). SessionIndex and repetition variables are removed from the dataset. All the features are selected from the dataset as feature reduction could not be performed because the number of the feature is considered small. Thus, the final dataset used for model development consists of 6000 instances and 32 variables. After the dataset has been prepared, the seven phases of model development is implemented. The development tool used in this study is Weka Explorer which contains a package called Dl4jMlpClassifier as a wrapper for an open-source deep learning programming library called Deeplearning4j which works by using Java virtual machine. The classifier is configured in accordance with the deep network model explained in Chapter 3.

CHAPTER 5

RESULTS AND ANALYSIS

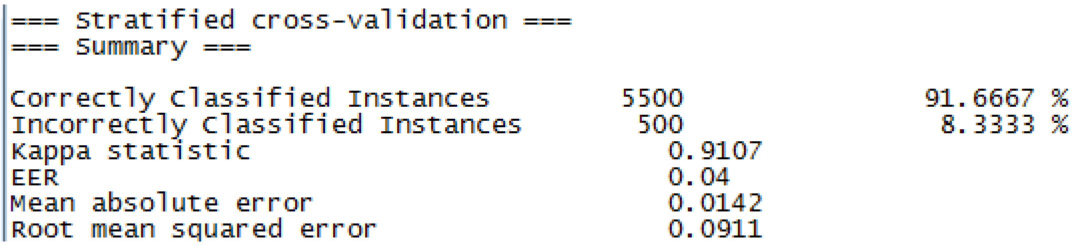
5.1. Introduction

In this chapter, the evaluation criteria selected for this study will be discussed. Next, the result of classifier training as discussed in Chapter 4 will be elaborated in order to get better understanding of the classifier performance. Although the performance metrics which are commonly used in keystroke dynamics only involves FAR, FRR, and EER, other types of performance metrics for the classifier such as accuracy, kappa statistic, MAE, RMSE, precision, recall, F-measure, MCC, ROC, PRC, and confusion matrix will also be investigated in this study. In addition, the performance of the classifier will be evaluated and compared with previous related works. The comparison will be done by using EER evaluation metric for keystroke dynamics studies which used the same dataset. Furthermore, the classifier will be tested on other datasets to measure its applicability in keystroke dynamics user authentication.

5.2. Evaluation Criteria and Result

The performance of a classifier needs to be evaluated in order to understand how well it performs in the study, in this case, keystroke dynamics. As a biometric user authentication technique, keystroke dynamics requires high accuracy in classifying genuine user and impostor. To evaluate the performance of deep learning model implemented in this study, FAR, FRR and EER will be prioritised as these are commonly used evaluation metrics in keystroke dynamics studies (as discussed in Chapter 2). However, the output of classifier training discussed in Chapter 4 offers different types of evaluation criteria such as accuracy, kappa statistic, RMSE, precision, recall, F-measure, MCC, ROC, PRC and confusion matrix. These evaluation metrics are also elaborated to get a better understanding of the performance of the classifier.

The results of the classification performed by Dl4jMlpClassifier are illustrated in the figures and tables below. Figure 5.1 shows the summary output for the classifier, Figure 5.2 and Figure 5.3 shows the confusion matrix for each class, and Table 5.2 shows the detailed accuracy for each class.



```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances        5500               91.6667 %
Incorrectly Classified Instances       500                8.3333 %
Kappa statistic                          0.9107
EER                                      0.04
Mean absolute error                      0.0142
Root mean squared error                  0.0911
```

Figure 5.1: Summary output for Dl4jMlpClassifier.

The overall accuracy of the classifier is 91.67% as a result of being able to identify 5500 out of 6000 instances. This means the classifier is able to correctly identify nine out of ten classification tasks (either identifying genuine user or impostor). Viera & Garrett (2005) discussed the use of kappa statistics in calculating the agreement level between observers towards the case studies. The Kappa statistic of the classifier is 0.9107 which means it has almost perfect and positive inter-observer agreement. This means the deep learning classifier is statistically significant to be used for keystroke dynamics studies. EER or a measure of accuracy for classification will be discussed in Table 5.2 along with FAR and FRR. Chai & Draxler (2014) compared the use of Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) in measuring the performance of a model. MAE measures the absolute distance between predicted value and observed value, whereas RMSE measures the squared distance between predicted value and observed value. The calculation for MAE and RMSE are represented in the Formula 5.1 and 5.2 respectively as follows:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|e_i| \qquad (5.1)$$

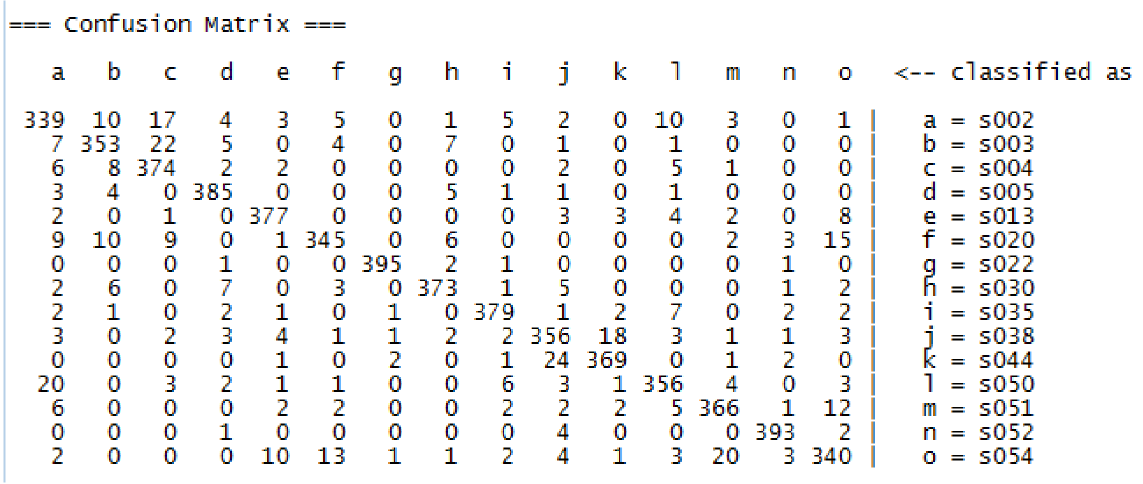
$$\text{RMSE}= \sqrt{\frac{1}{n}\sum_{i=1}^{n} e_i^2} \qquad (5.2)$$

The MAE value for the classifier is 0.0142. This indicates the classifier has approximately an average absolute error of 0.0142 in identifying genuine user and impostor for fifteen users. The RMSE value for the classifier is 0.0911. This indicates the classifier gives approximately a squared error of 0.0911 in identifying genuine user and impostor for fifteen users.

In solving the classification problem for keystroke dynamics, simpler representation of the users (subject IDs) is implemented. User with subject ID ‘s002’ is represented as ‘a’, user with subject ID ‘s003’ is represented as ‘b’, and so on. The full representation list is shown in Table 5.1 below.

Table 5.1: Representation of Users in Classification Task

Subject ID	Represented As
s002	a
s003	b
s004	c
s005	d
s013	e
s020	f
s022	g
s030	h
s035	i
s038	j
s044	k
s050	l
s051	m
s052	n
s054	o



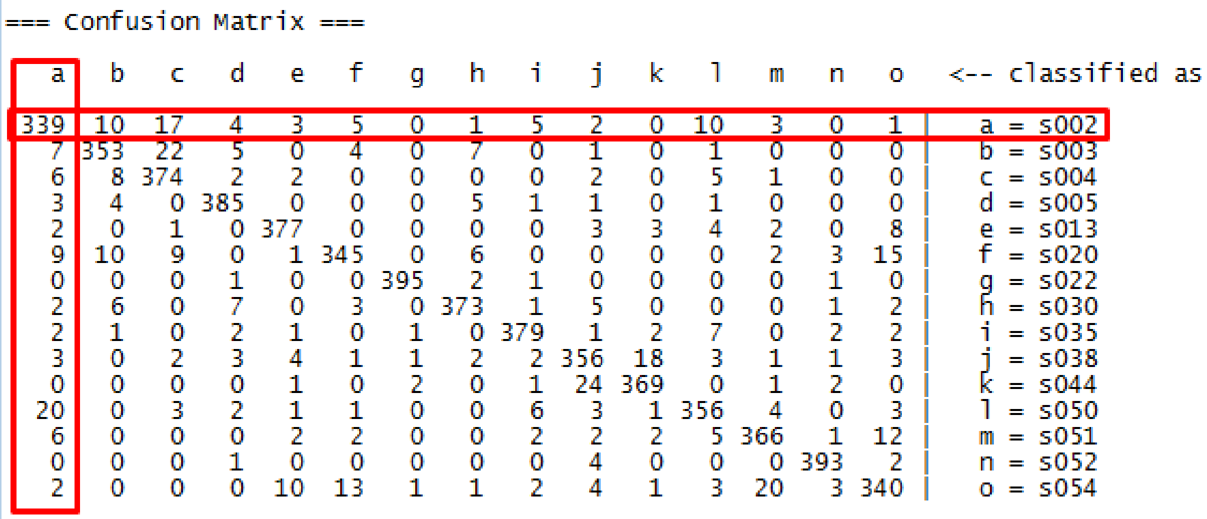
```
=== Confusion Matrix ===

   a   b   c   d   e   f   g   h   i   j   k   l   m   n   o   <-- classified as

 339  10  17   4   3   5   0   1   5   2   0  10   3   0   1 |   a = s002
   7 353  22   5   0   4   0   7   0   1   0   1   0   0   0 |   b = s003
   6   8 374   2   2   0   0   0   0   2   0   5   1   0   0 |   c = s004
   3   4   0 385   0   0   0   5   1   1   0   1   0   0   0 |   d = s005
   2   0   1   0 377   0   0   0   0   3   3   4   2   0   8 |   e = s013
   9  10   9   0   1 345   0   6   0   0   0   0   2   3  15 |   f = s020
   0   0   0   1   0   0 395   2   1   0   0   0   0   1   0 |   g = s022
   2   6   0   7   0   3   0 373   1   5   0   0   0   1   2 |   h = s030
   2   1   0   2   1   0   1   0 379   1   2   7   0   2   2 |   i = s035
   3   0   2   3   4   1   1   2   2 356  18   3   1   1   3 |   j = s038
   0   0   0   0   1   0   2   0   1  24 369   0   1   2   0 |   k = s044
  20   0   3   2   1   1   0   0   6   3   1 356   4   0   3 |   l = s050
   6   0   0   0   2   2   0   0   2   2   2   5 366   1  12 |   m = s051
   0   0   0   1   0   0   0   0   0   4   0   0   0 393   2 |   n = s052
   2   0   0   0  10  13   1   1   2   4   1   3  20   3 340 |   o = s054
```

Figure 5.2: Confusion matrix for each class.

The Figure 5.2 above shows the confusion matrix for each class. It shows the actual number of instances in a class (horizontal), the total correct classification as a class (vertical) and the total number of correct classification for all classes (which is 5500 as shown in Figure 5.1). The Figure 5.3 below shows the actual number of instances in class a (addition of values in horizontal line) and the total correct classification as class a (addition of values in vertical line). Thus, the precision and recall of class a can be found by calculating 339/401 and 339/400 respectively.



```
=== Confusion Matrix ===

   a   b   c   d   e   f   g   h   i   j   k   l   m   n   o   <-- classified as
 339  10  17   4   3   5   0   1   5   2   0  10   3   0   1 |   a = s002
   7 353  22   5   0   4   0   7   0   1   0   1   0   0   0 |   b = s003
   6   8 374   2   2   0   0   0   0   2   0   5   1   0   0 |   c = s004
   3   4   0 385   0   0   0   5   1   1   0   1   0   0   0 |   d = s005
   2   0   1   0 377   0   0   0   0   3   3   4   2   0   8 |   e = s013
   9  10   9   0   1 345   0   6   0   0   0   0   2   3  15 |   f = s020
   0   0   0   1   0   0 395   2   1   0   0   0   0   1   0 |   g = s022
   2   6   0   7   0   3   0 373   1   5   0   0   0   1   2 |   h = s030
   2   1   0   2   1   0   1   0 379   1   2   7   0   2   2 |   i = s035
   3   0   2   3   4   1   1   2   2 356  18   3   1   1   3 |   j = s038
   0   0   0   0   1   0   2   0   1  24 369   0   1   2   0 |   k = s044
  20   0   3   2   1   1   0   0   6   3   1 356   4   0   3 |   l = s050
   6   0   0   0   2   2   0   0   2   2   2   5 366   1  12 |   m = s051
   0   0   0   1   0   0   0   0   0   4   0   0   0 393   2 |   n = s052
   2   0   0   0  10  13   1   1   2   4   1   3  20   3 340 |   o = s054
```

Figure 5.3: Confusion matrix interpretation for s002.

From the confusion matrix above, evaluation metrics such as precision, recall, and others can be calculated for each class, as shown in Table 5.2 below. The weighted average value for all classes is highlighted in the last row of the table.

Table 5.2: Detailed Accuracy by Class

TP Rate	**FP Rate**	**Precision**	**Recall**	**F-Measure**	**MCC**	**ROC Area**	**PRC Area**	**Class**
0.848	0.011	0.845	0.848	0.846	0.835	0.991	0.932	s002
0.883	0.007	0.901	0.883	0.891	0.884	0.996	0.964	s003
0.935	0.010	0.874	0.935	0.903	0.897	0.995	0.964	s004
0.963	0.005	0.934	0.963	0.948	0.945	0.998	0.987	s005
0.943	0.004	0.938	0.943	0.940	0.936	0.997	0.982	s013
0.863	0.005	0.922	0.863	0.891	0.885	0.993	0.956	s020
0.988	0.001	0.988	0.988	0.988	0.987	1.000	0.999	s022
0.933	0.004	0.940	0.933	0.936	0.931	0.995	0.976	s030
0.948	0.004	0.948	0.948	0.948	0.944	0.997	0.976	s035
0.890	0.009	0.873	0.890	0.881	0.873	0.996	0.957	s038
0.923	0.005	0.932	0.923	0.927	0.922	0.998	0.972	s044
0.890	0.007	0.901	0.890	0.896	0.888	0.994	0.945	s050
0.915	0.006	0.915	0.915	0.915	0.909	0.995	0.968	s051
0.983	0.003	0.966	0.983	0.974	0.972	0.999	0.993	s052
0.850	0.009	0.876	0.850	0.863	0.853	0.992	0.933	s054
0.917	**0.006**	**0.917**	**0.917**	**0.917**	**0.911**	**0.996**	**0.997**	

The Table 5.2 above shows the individual accuracy for each class and average accuracy by class for the classifier. From the result above, it can be inferred that the classifier achieves average 0.083 FAR (1 – TPR) and average 0.006 FRR. This indicates that the classifier has 8.3% rate in false identification of impostor as genuine user and 0.6% rate in false identification of the genuine user as an impostor. After obtaining the value of FAR and FRR, Equal Error Rate (EER) can be calculated by using the formula ((FAR+FRR)/2, which gives 0.0445 as a result. Thus, 0.0445 is the threshold value for FAR and FRR in the classifier. This value is similar to the result given by the classifier as illustrated in Figure 5.1, which is 0.04.

Next, the classifier achieves an average precision, recall, and F-measure of 0.917. The precision indicates that the classifier is able to identify 91.7% of the impostor correctly and the recall indicates that the classifier is able to recognize 91.7% of all impostor cases in the dataset. The F-measure calculates the harmonic mean of precision and recall of the classifier, thus also achieves 91.7% rate. Matthews (1975) discussed the use of Matthews Correlation Coefficient (MCC) in measuring the quality of classification of a machine learning model by comparing the predicted value and actual value. The classifier achieves average 0.911 MCC which means it has an almost perfect prediction for the genuine user and impostor as the value is close to 1. Davis & Goadrich (2006) discussed the importance of Receiver Operator Characteristics (ROC) curve and Precision-Recall Curve (PRC) for machine learning classification. The classifier achieves 0.996 average area under ROC curve and 0.967 area under PRC curve. This indicates that the classifier achieves almost excellent discrimination (area > 0.9) in correct classification of genuine user and impostor.

5.3. Comparison of the MLP with Other Works

After discussing different types of evaluation metrics for the deep learning classifier, the result is compared with previous related works in keystroke dynamics. In order to gain better interpretability, the same evaluation metric (EER) is used for the comparison. Also, in order to achieve a better understanding, the comparison is only performed on the researchers who used the same dataset (CMU). The performance comparison can be seen in Table 5.3 below.

Table 5.3: Comparison with Related Works (for CMU dataset)

Reference	Classifier	EER
Deng & Zhong (2013)	Deep belief nets	0.0350
Current Work	**Deep Multilayer Perceptron**	**0.0445**
Deng & Zhong (2013)	Gaussian Mixture Model with Universal Background Model	0.0550
Maheshwary & Pudi (2017)	Nearest Neighbor Regression	0.0698
Al-Jarrah (2012)	Median Proximity	0.0800
Bakelman et al. (2013)	Pace classifier	0.0870
Killourhy (2012)	Scaled Manhattan	0.0962
Killourhy (2012)	Support Vector Machine	0.1025

The deep learning method is able to achieve better EER compared to other classifiers in keystroke dynamics. It can be seen that the best performing classifier in the comparison table is also another deep learning algorithm named deep belief nets with EER of 0.035. This indicates deep learning classifier has not only been performing well in other fields but also in keystroke dynamics field to identify genuine user and impostor in user authentication system. The achieved performance of 4.45% by using deep learning classifier in this study opens further improvement in this particular area as this implementation is only conducted by using one word (a ten characters password). For future research on keystroke dynamics as continuous user authentication, more data will be used as the authentication factors and better performance by deep learning classifier can be achieved.

5.4. Testing on Other Datasets

Further evaluation was performed on three publicly available datasets. The first dataset is BioChaves dataset provided by Montalvao, Almeida & Freire (2006). In the data acquisition, a total of 47 participants were asked to type five fixed texts (‘chocolate’, ‘zebra’, ‘banana’, ‘taxi’, and ’computador calcula’) consecutively for ten repetitions. This data was stored in four separate databases (‘A’, ‘B’, ‘C’, and ‘D’). In this testing, database A of BioChaves dataset was evaluated. The result of the test can be seen in Table 5.4 below. The second dataset is Keystroke100 dataset provided by Loy, Lai & Lim (2007). The data was collected from 100 participants typing ‘try4-mbs’ password. The third dataset is GREYC-NISLAB dataset provided by Idrus et al. (2014) in their work to explore soft biometric effects on keystroke dynamics. The dataset was collected from 110 participants.

Table 5.4: Testing Classifier with BioChaves dataset

Reference	**Classifier**	**EER**
Current work	**Deep Multilayer Perceptron**	**0.055**
Montalvao, Almeida & Freire (2006)	Statistical with Equalization	0.100

Table 5.5: Testing Classifier with Keystroke100 dataset

Reference	**Classifier**	**EER**
Current work	**Deep Multilayer Perceptron**	**0.1**
Loy, Lai & Lim (2007)	ARTMAP-FD Neural Network	11.78

Table 5.6: Testing Classifier with GREYC-NISLAB dataset

Reference	**Classifier**	**Accuracy**
Current work	Deep Multilayer Perceptron	90.67%
Idrus et al. (2014)	**Majority Voting and Score Fusion**	**91.67%**

The sampling size for all test datasets are limited to fifteen users. For BioChaves dataset, the deep multilayer perceptron is able to provide better classification accuracy compared to the baseline model in previous study. Although BioChaves dataset has smaller size compared to CMU dataset, the deep learning classifier is able to perform classification task in differentiating ten users with 90% accuracy. For both remaining datasets, a small change is made in the classifier configuration. The dataset is standardized instead of normalized because all the features have value above one (unlike CMU dataset). For the Keystroke100 dataset, the deep multilayer perceptron is able to achieve better accuracy than the baseline model. It achieves 89.33% (EER: 0.05) accuracy in classifying the genuine users and impostor. For the GREYC-NISLAB dataset, the classifier achieves lower accuracy by one percentage than the baseline model. Overall, the classifier was able to achieve an average of 90% accuracy in all datasets. These indicate that it could be a suitable classifier to be used in keystroke dynamics for user authentication.

5.5. Summary

As a summary, three evaluation metrics such as FAR, FRR, and EER are selected and prioritised in this study to evaluate the performance of the deep learning classifier. Based on the training result, the classifier has achieved 0.083 FAR, 0.006 FRR, and 0.0445 EER in classifying genuine user and impostor based on fifteen users data. However, there are also other types of performance metrics which can also be used to evaluate the classifier such as accuracy, kappa statistic, MAE, RMSE, precision, recall, F-measure, MCC, ROC, PRC, and confusion matrix. The accuracy of the classifier shows that it is able to identify 91.67% of the instances correctly. The Kappa statistic of the classifier shows that it has almost perfect and positive inter-observer agreement with a coefficient of 0.9107. The MAE and RMSE have indicated that the classifier suffers differences between predicted value and actual value with error of 0.0142 and 0.0911 respectively. The MCC statistic of the classifier shows that it has close to perfect prediction for genuine user and impostor with a coefficient of 0.911. The classifier also achieved 91.7% precision, recall, and F-measure in the classification task. The ROC area and PRC area of the classifier indicates that the classifier achieves almost excellent discrimination in correct classification of genuine user and impostor with the value of 0.996 and 0.967 respectively. After conducting a comparison with related works on the same dataset, the deep learning classifier is able to achieve better performance compared to other classifiers in keystroke dynamics with EER of 4.45%. At last but not least, the classifier is also able to achieve an average of 90% classification accuracy and EER of 5.5%, 10%, and 5% using three different datasets.

CHAPTER 6

DISCUSSION AND CONCLUSION

6.1. Introduction

In this chapter, the discussion and conclusion of the study will be summarised based on important points from each chapter. Next, the contribution achieved in this study will be discussed, especially in literature review, keystroke dynamics authentication process, and deep learning model. Besides, the importance of the study will also be discussed in a sense that how this study will provide impact towards the field of keystroke dynamics. In addition, future recommendation in keystroke dynamics will be suggested in order to explore more knowledge related to keystroke dynamics and to improve the performance of the existing models.

6.2. Discussion and Conclusions

Keystroke dynamics is a user authentication method which uses user's typing rhythm to allow access into the system. It has two advantages compared to other biometric authentication methods such as: lower implementation cost as no additional hardware is required in the authentication process; and easier implementation because the collection of typing data is relatively easy as it does not require special permission from the user. Advances in keystroke dynamics have produced multiple classifiers such as statistical and machine learning in order to perform classification for genuine user and impostor, however, maximum rate of accuracy has not been achieved. In order to solve the problem, the aim of this study is to propose deep learning model in keystroke dynamics for user authentication. This study is important because it can potentially increase the user awareness and understanding regarding the biometrics authentication, and tackle security issues specifically in access control and data privacy as can provide better authentication measure compared to SFA. The scope of this study is limited to the implementation of deep learning, training model with one dataset, and does not cover external factors affecting keystroke dynamics performance.

The literature review conducted in this study indicated that there are still challenges which need to be addressed in order for keystroke dynamics to become an effective biometric measure for user authentication. Firstly, the dataset for keystroke dynamics research should be standardized in order to make result comparison easier. Secondly, because there is no standard rule to decide which keystroke features should be used in keystroke dynamics many researchers have used a different type of features in their analysis. Thirdly, whilst external factors such as emotions, language, soft biometrics, stylometry, and others could make keystroke dynamics a less rigid authentication measure compared to other behavioural biometrics, further research could be done to explore more about another factor such as motor behaviour in order to provide better understanding in factors consideration while developing the classifier. At last but not least, the lack of visualization of keystroke data in past researches opens the possibility that hidden pattern might exist in the data. Although keystroke data mainly consists of numerical data, the relationship between variables could still be explored and possibly lead to a new insight of the study.

This study to propose a deep learning model in keystroke dynamics for user authentication is conducted by using KDD methodology. A keystroke dynamics authentication process is developed by using deep learning. In addition, the classification method for keystroke dynamics by using deep learning is elaborated. The network model used in the deep learning is multilayer perceptron with two hidden layers. Stochastic gradient descent algorithm will be used as the optimization technique as it is able to minimize the error and the cost of a function. An acceleration technique for gradient descent called momentum will be used to increase the learning speed of the network. To calculate the error of the function, backpropagation algorithm will be used. For weight initialization, Xavier initialization will be used. It is a technique that assigns weight by considering the learning effect of the neurons in order to maintain equal distribution of activations. Next, there are two activation functions used in the network: relu for hidden layers and softmax for the output layer. The model will use multi-class cross-entropy as its output function.

The dataset preparation and model development processes for keystroke dynamics are discussed in this chapter. First of all, the keystroke dynamics dataset is briefly explained which includes the data collection process and the definition of the variables. Next, descriptive statistics and EDA of the dataset are generated in order to explore previously unseen pattern and to explore correlation and outlier that exists in the dataset. Based on these observations, the dataset is reduced by selecting only fifteen users based on the number of outliers existed for each user (Five users who have the lowest number of outliers, five users who have medium number of outlier, and five users who have the highest number of outliers). SessionIndex and repetition variables are removed from the dataset. All the features are selected from the dataset as feature reduction could not be performed because the number of the feature is considered small. Thus, the final dataset used for model development consists of 6000 instances and 32 variables. After the dataset has been prepared, the seven phases of model development is implemented. The development tool used in this study is Weka Explorer which contains a package called Dl4jMlpClassifier as a wrapper for an open-source deep learning programming library called Deeplearning4j which works by using Java virtual machine. The classifier is configured in accordance with the deep network model explained in Chapter 3.

Three evaluation metrics such as FAR, FRR, and EER are selected and prioritised in this study to evaluate the performance of the deep learning classifier. Based on the training result, the classifier has achieved 0.083 FAR, 0.006 FRR, and 0.0445 EER in classifying genuine user and impostor based on fifteen users data. However, there are also other types of performance metrics which can also be used to evaluate the classifier such as accuracy, kappa statistic, MAE, RMSE, precision, recall, F-measure, MCC, ROC, PRC, and confusion matrix. The accuracy of the classifier shows that it is able to identify 91.67% of the instances correctly. The Kappa statistic of the classifier shows that it has almost perfect and positive inter-observer agreement with a coefficient of 0.9107. The MAE and RMSE have indicated that the classifier suffers differences between predicted value and actual value with error of 0.0142 and 0.0911 respectively. The MCC statistic of the classifier shows that it has close to perfect prediction for genuine user and impostor with a coefficient of 0.911. The classifier also achieved 91.7% precision, recall, and F-measure in the classification task. The ROC area and PRC area of the classifier indicates that the classifier achieves almost excellent discrimination in correct classification of genuine user and impostor with the value of 0.996 and 0.967 respectively. After conducting a comparison with related works on the same dataset, the deep learning classifier is able to achieve better performance compared to other classifiers in keystroke dynamics with EER of 4.45%. At last but not least, the classifier is also able to achieve an average of 90% classification accuracy and EER of 5.5%, 10%, and 5% using three different datasets.

Keystroke dynamics is an interesting field to explore as one type of biometric authentication measure despite it has lower classification accuracy and a limited amount of studies compared to other biometric modalities. Although the field of study is still open to challenges and improvement, it has a potential to become an effective, strong and low-cost biometric user authentication.

6.3. Contribution and Importance of the Study

The contributions achieved from this study are as follows:

Investigation of recent keystroke dynamics studies

This study has presented a literature review on keystroke dynamics studies. In addendum, essential elements such as features, datasets, and evaluation criteria were investigated. Next, classification approaches including statistical, machine learning, deep learning and hybrid models are explained in chronological order. Besides, external factors affecting the classifier accuracy and data privacy issues are also discussed in this study. The most significant feature that affected keystroke dynamics performance had yet to be explored and CMU benchmark dataset was used more frequently while compared with other openly available dataset. EER is the most widely used error metric to assess the performance of keystroke dynamics classifier. Based on the survey, deep learning implementation in keystroke dynamics are limited while compared with other classification methods. This work can be used as a reference for future researchers who are interested in engaging in keystroke dynamics field.

Implementation of deep learning multilayer perceptron model for keystroke dynamics user authentication using static password text

This study has presented a process flow for keystroke dynamics user authentication by using deep learning. It illustrates the keystroke dynamics authentication procedures required to be taken by user when accessing an account. The brief explanation of the framework is as follows: a keylogger software will record password typed by the user (e.g. during login) and pass both password information and keystroke information to the system for matching. The password will be matched with the password template and the keystroke information will be matched with the biometric template using deep learning. If both matching processes succeed, the system will accept the user as a genuine user or reject the user otherwise. Besides, this study has also presented a deep learning model by using multilayer perceptron to perform classification task for keystroke dynamics. The network configuration and layer specification of the model are done by using Weka Explorer. The result of the classification is analysed by using multiple evaluation metrics and the deep learning classifier achieved EER of 4.45%. At last but not least, the classifier is also able to achieve an average of 90% classification accuracy and EER of 5.5%, 10%, and 5% using three different datasets.

The project to research deep learning application in keystroke dynamics areas is important to provide a better understanding of the area. As keystroke dynamics is relatively new and still involves many challenges (Zhong & Deng, 2015), the research in keystroke dynamics area can contribute to the improvement of knowledge since deep learning application in keystroke dynamics is limited. The results of this study will provide some insights and information regarding the effectiveness of deep learning implementation in keystroke dynamics. This indicates that deep learning is not only effective in sound and image recognition, but also in identifying pattern for user authentication measures such as keystroke dynamics. Despite having not achieved maximum accuracy in the classification task, the study will encourage future researchers to explore more possibilities for the implementation of deep learning in keystroke dynamics. Also, this study can help boost the awareness of users to use multi-factor authentication measure such as keystroke dynamics, since keystroke dynamics has lower implementation cost compared to other biometrics authentication.

6.4. Future Recommendations

The study on deep learning model development for keystroke dynamics has achieved a promising result. However, there are several limitations which could not be addressed by the completion of the study. Firstly, the study only uses single model (multilayer perceptron) for the deep learning implementation. Secondly, the study only uses a single dataset to perform model training. Although these limitations did not affect the achievement of aim and objectives of the study, better performance could be achieved. Hence, the future research can compare more complex deep learning models such as Convolutional Neural Networks (CNN), autoencoders, recurrent neural networks and others for keystroke dynamics. Another future research in keystroke dynamics field is to build deep learning model for the mobile platform.

6.5. Summary

As a summary, the discussion and conclusion for each chapter are discussed. By the end of this study, there are three contributions achieved such as: (i) conduct literature review in keystroke dynamics for past decade; (ii) design a conceptual framework for keystroke dynamics; and (iii) develop a deep learning multilayer perceptron model for keystroke dynamics. This study is important because it can help to boost the awareness of users to use multi-factor authentication measure such as keystroke dynamics, The study also indicates that deep learning is not only effective in sound and image recognition, but also in identifying the pattern for user authentication measures such as keystroke dynamics. The future recommendations for keystroke dynamics studies are to compare more complex deep learning models, to test classifier on more datasets in order to verify its performance and readiness in handling user authentication with keystroke dynamics and to build deep learning model for the mobile platform.

REFERENCES

Abidin, A., Argones Rúa, E. & Peeters, R. (2017). *Uncoupling Biometrics from Templates for Secure and Privacy-Preserving Authentication*. In Proceedings of the 22nd ACM on Symposium on Access Control Models and Technologies - SACMAT '17. [Online]. p. 21–29.

Al-Jarrah, M.M. (2012). An Anomaly Detector for Keystroke Dynamics Based on Medians Vector Proximity. *Journal of Emerging Trends in Computing and Information Sciences*. [Online]. 3(6). p. 988–993. Available from: http://cisjournal.org/journalofcomputing/archive/vol3no6/vol3no6_20.pdf [Accessed: 3/7/2017]

Al-Rahmani, A.O. (2014). *An Enhanced Classifier for Authentication in Keystroke Dynamics Using Experimental Data*. Middle East University.

Ali, M.L., Monaco, J. V., Tappert, C.C. & Qiu, M. (2017). Keystroke Biometric Systems for User Authentication. *Journal of Signal Processing Systems*. [Online]. 86 (2–3). p. 175–190. Available from: http://dx.doi.org/10.1007/s11265-016-1114-9 [Accessed: 8/7/2017]

Alpar, O. (2014). Keystroke recognition in user authentication using ANN based RGB histogram technique. *Journal of Engineering Applications of Artificial Intelligence*. [Online]. 32. p. 213–217. Available from: http://dx.doi.org/10.1016/j.engappai.2013.11.009 [Accessed: 9/7/2017]

Alsultan, A., Warwick, K. & Wei, H. (2017). Non-conventional keystroke dynamics for user authentication. *Journal of Pattern Recognition Letters*. [Online]. 89. p. 53–59. Available from: http://dx.doi.org/10.1016/j.patrec.2017.02.010 [Accessed: 8/7/2017]

Awad, A., Traore, I. & Almulhem, A. (2008). *Digital Fingerprinting Based on Keystroke Dynamics*. In Second International Symposium on Human Aspects of Information Security & Assurance (HAISA 2008) Digital. (Haisa). p. 94–104.

Azevedo, G.L.F., Cavalcanti, G.D.C. & Carvalho Filho, E.C.B. (2007). An approach to feature selection for keystroke dynamics systems based on PSO and feature weighting. *Journal of Evolutionary Computation*. [Online]. p. 3577–3584. Available from: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4424936 [Accessed: 9/7/2017]

Bakelman, N., Monaco, J. V., Cha, S.H. & Tappert, C.C. (2013). *Keystroke biometric studies on password and numeric keypad input*. In Proceedings - 2013 European Intelligence and Security Informatics Conference, EISIC 2013. p. 204–207.

Banerjee, S.P. & Woodard, D. (2012). Biometric Authentication and Identification Using

Keystroke Dynamics: A Survey. *Journal of Pattern Recognition Research*. [Online]. 7 (1). p. 116–139. Available from: http://jprr.org/index.php/jprr/article/view/427%5Cnhttp://www.jprr.org/index.php/jprr/article/view/427/167 [Accessed: 9/7/2017]

Bengio, Y. (2009). Learning Deep Architectures for AI. *Foundations and Trends® in Machine Learning*. [Online]. 2 (1). p. 1–127. Available from: http://www.nowpublishers.com/article/Details/MAL-006 [Accessed: 15/7/2017]

Bengio, Y. (2012). Practical recommendations for gradient-based training of deep architectures. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. 7700 LECTU. p. 437–478.

Bengio, Y., Courville, A. & Vincent, P. (2012). *Representation Learning: A Review and New Perspectives*. [Online]. (1993). p. 1–30. Available from: http://arxiv.org/abs/1206.5538 [Accessed: 11/7/2017]

Bharadi, H.B.K., Shah, P.S. & Ambardekar, A.. (2011). *Keystroke dynamic analysis using relative entropy & timing sequence euclidian distance*. In Proceedings of the International Conference & Workshop on Emerging Trends in Technology - ICWET '11. [Online]. (Icwet). p.p. 220. Available from: http://dl.acm.org/citation.cfm?id=1980072.

Bishop, C.M. (2006). *Pattern Recognition and Machine Learning*. p. 1-749. Singapore: Springer.

Blum, A. (1992). *Neural Networks in C++: an Object-Oriented Framework for Building Connectionist System*. p. 1-207. New York: John Wiley & Sons.

Brizan, D.G., Goodkind, A., Koch, P., Balagani, K., Phoha, V. V. & Rosenberg, A. (2015). Utilizing linguistically enhanced keystroke dynamics to predict typist cognition and demographics. *International Journal of Human Computer Studies*. [Online]. 82. p. 57–68. Available from: http://dx.doi.org/10.1016/j.ijhcs.2015.04.005 [Accessed: 19/7/2017].

Bryan, W.L. & Harter, N. (1897). *Studies in the physiology and psychology of the telegraphic language.* [Online]. Available from: http://content.apa.org/journals/rev/4/1/27 [Accessed: 6/7/2017]

Chai, T. & Draxler, R.R. (2014). Root mean square error (RMSE) or mean absolute error (MAE)-Arguments against avoiding RMSE in the literature. *Journal in Geoscientific Model Development*. [Online]. 7 (3). p. 1247–1250. Available from: https://www.geosci-model-dev.net/7/1247/2014/gmd-7-1247-2014.pdf [Accessed: 16/7/2017]

Darabseh, A. & Namin, A.S. (2016). *On Accuracy of Classification-Based Keystroke Dynamics for Continuous User Authentication*. In Proceedings - 2015 International

Conference on Cyberworlds, CW 2015. p. 321–324.

Davis, J. & Goadrich, M. (2006). *The relationship between Precision-Recall and ROC curves.* In Proceedings of the 23rd international conference on Machine learning - ICML ’06. [Online]. 2006, New York, New York, USA: ACM Press, p. 233–240.

Deng, L. & Yu, D. (2014). Deep Learning: Methods and Applications. *Journal of Foundations and Trends® in Signal Processing*. [Online]. 7 (3–4). p. 197–387. Available from: http://nowpublishers.com/articles/foundations-and-trends-in-signal-processing/SIG-039 [Accessed: 18/7/2017]

Deng, Y. & Zhong, Y. (2013). Keystroke Dynamics User Authentication Based on Gaussian Mixture Model and Deep Belief Nets. *Journal of Signal Processing*. [Online]. 2013. p. 1–7. Available from: http://www.hindawi.com/journals/isrn/2013/565183/ [Accessed: 16/7/2017]

Eltahir, W.E., Salami, M.J.E., Ismail, A.F. & Lai, W.K. (2008). Design and evaluation of a pressure-based typing biometric authentication system. *Eurasip Journal on Information Security*. [Online]. p. 1-14. Available from: https://link.springer.com/article/10.1155/2008/345047 [Accessed: 12/7/2017]

Fayyad, U., Piatetsky-Shapiro, G. & Smyth, P. (1996). From data mining to knowledge discovery in databases. *AI magazine*. [Online]. p. 37–54. Available from: http://www.aaai.org/ojs/index.php/aimagazine/article/viewArticle/1230 [Accessed: 12/7/2017]

Forsen, G.E., Nelson, M.R. & Staron, R.J. (1997). *Personal Attributes Authentication Techniques*. [Online]. Available from: http://www.dtic.mil/cgi-bin/GetTRDoc?Location=U2&doc=GetTRDoc.pdf&AD=ADA047645 [Accessed: 12/7/2017]

Frank, E., Hall, M.A. & Witten, I.H. (2016). The WEKA Workbench. *Morgan Kaufmann, Fourth Edition*. [Online]. p. 553–571. Available from: http://www.cs.waikato.ac.nz/ml/weka/Witten_et_al_2016_appendix.pdf [Accessed: 29/7/2017]

Fridman, L., Stolerman, A., Acharya, S., Brennan, P., Juola, P., Greenstadt, R., Kam, M. & Gomez, F. (2015). Multi-modal decision fusion for continuous authentication. *Computers and Electrical Engineering*. [Online]. 41 (C). p. 142–156. Available from: http://dx.doi.org/10.1016/j.compeleceng.2014.10.018 [Accessed: 14/7/2017]

Gaines, R.S., Lisowski, W., Press, S.J. & Shapiro, N. (1980). *Authentication by Keystroke Timing: Some Preliminary Results*. [Online]. California: The Rand Corporation.

Available from: http://www.dtic.mil/dtic/tr/fulltext/u2/a484022.pdf [Accessed: 8/7/2017]

Gingrich, J.H.D. & Sentosa, A. (2008). *Authentication through Biometric Keystroke Dynamics*. In Proceedings of IEEE International Conference on Communications, ICC 2008, 19-23 May 2008. China: Beijing. p. 1556–1560.

Giot, R., El-Abed, M. & Rosenberger, C. (2009). *Keystroke dynamics with low constraints SVM based passphrase enrollment*. In IEEE 3rd International Conference on Biometrics: Theory, Applications and Systems, BTAS 2009 France: Caen Cedex. p. 1-6.

Giot, R., El-Abed, M. & Rosenberger, C. (2012). *Web-Based Benchmark for Keystroke Dynamics Biometric Systems: A Statistical Analysis*. In The Eighth International Conference on Intelligenct Information Hiding and Multimedia Signal Processing, Jul 2012. Greece: Piraeus. p. 1-5.

Glorot, X. & Bengio, Y. (2010). *Understanding the difficulty of training deep feedforward neural networks*. In Proceedings of the 13th International Conference on Artificial Intelligence and Statistics (AISTATS). vol. 9. Italy: Sardinia. p. 249–256.

Goodfellow, I., Bengio, Y. & Courville, A. (2016). Deep Learning. *MIT press*. [Online]. p. 1-800. Available from: http://files.sig2d.org/sig2d14.pdf#page=5 [Accessed: 15/7/2017]

Goodkind, A., Brizan, D.G. & Rosenberg, A. (2017). Utilizing overt and latent linguistic structure to improve keystroke-based authentication. *Image and Vision Computing*. [Online]. 58. p.pp. 230–238. Available from: http://dx.doi.org/10.1016/j.imavis.2016.06.003.

Han, J., Kamber, M. & Pei, J. (2012). *Data Mining: Concepts and Techniques*. 3rd Ed. United States of America: Morgan Kaufmann.

He, L., Li, Z. & Shen, C. (2017). *Performance Evaluation of Anomaly-Detection Algorithm for Keystroke-Typing based Insider Detection*. In Proceedings of ACM TUR-C '17. May 12-14, 2017. China: Shanghai. p. 1-7.

Heaton, J. (2015). *AIFH, Volume 3: Deep Learning and Neural Networks*. 1st Ed. T. Heaton (ed.). Heaton Research, Inc.

Ho, J. & Kang, D.K. (2017). Mini-batch bagging and attribute ranking for accurate user authentication in keystroke dynamics. *Journal of Pattern Recognition*. [Online]. 70. p. 139–151. Available from: http://dx.doi.org/10.1016/j.patcog.2017.05.002 [Accessed: 19/7/2017]

Idrus, S.Z.S., Cherrier, E., Rosenberger, C. & Bours, P. (2014). Soft biometrics for keystroke dynamics: Profiling individuals while typing passwords. *Journal of Computers and Security*. [Online]. 45. p. 147–155. Available from:

http://dx.doi.org/10.1016/j.cose.2014.05.008 [Accessed: 20/7/2017]

Kang, P. (2015). The effects of different alphabets on free text keystroke authentication: A case study on the Korean-English users. *Journal of Systems and Software*. [Online]. 102. p. 1–11. Available from: http://dx.doi.org/10.1016/j.jss.2014.12.017 [Accessed: 17/7/2017]

Kasprowski, P. & Harezlak, K. (2016). Fusion of eye movement and mouse dynamics for reliable behavioral biometrics. *Journal of Pattern Analysis and Applications*. [Online]. p. 1–13 [Accessed: 20/7/2017]

Killourhy, K.S. (2012). *A Scientific Understanding of Keystroke Dynamics*. A Doctoral Dissertation Submitted in partial fulfilment of the requirements of Carnegie Mellon University for the degree of Doctor of Philosophy. USA: Carnegie Mellon University.

Kobojek, P. & Saeed, K. (2016). Application of recurrent neural networks for user verification based on keystroke dynamics. *Journal of Telecommunications and Information Technology*. 2016 (3). p.pp. 80–90.

Kumar, S. (2004). *Neural Networks: A Classroom Approach*. 3rd Ed. McGraw-Hill (ed.). New Delhi: McGraw-Hill.

Laerd (2013). *Measures of Central Tendency*. [Online]. Available from: https://statistics.laerd.com/statistical-guides/measures-central-tendency-mean-mode-median.php. [Accessed: 15/7/2017]

Lee, P.M., Tsui, W.H. & Hsiao, T.C. (2015). The Influence of Emotion on Keyboard Typing: An Experimental Study Using Visual Stimuli. *Journal of BioMedical Engineering Online*. [Online] 13(6). p. 1-12. Available from: https://ir.nctu.edu.tw/bitstream/11536/24927/1/000338784200001.pdf [Accessed: 11/7/2017]

Liu, W., Uluagac, A.S. & Beyah, R. (2014). *MACA: A privacy-preserving multi-factor cloud authentication system utilizing big data*. In Proceedings - IEEE INFOCOM. p. 518–523.

Loy, C.C., Lai, W.K. & Lim, C.P. (2007). *Keystroke Patterns Classification Using the ARTMAP-FD Neural Network.* In Third International Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP 2007). November 2007, IEEE, p. 61–64..

Maheshwary, S. & Pudi, V. (2017). *Mining Keystroke Timing Pattern for User Authentication*. In NFMCP: European Conference on Machine Learning and Practice of Knowledge Discovery in Databases. Italy, Monday 19th September to Friday 23rd September 2016. India: IIIT. pp. 1-12.

Matthews, B.W. (1975). Comparison of the predicted and observed secondary structure of T4 phage lysozyme. *Journal of Biochimica et Biophysica Acta (BBA) - Protein Structure*. [Online]. 405 (2). p. 442–451. Available from: http://linkinghub.elsevier.com/retrieve/pii/0005279575901099 [Accessed: 21/8/2017]

Messerman, A., Mustafić, T., Camtepe, S.A. & Albayrak, S. (2011). *Continuous and non-intrusive identity verification in real-time environments based on free-text keystroke dynamic*s. In International Joint Conference on Biometrics, IJCB 2011. p. 1-8.

Microsoft (2017). *Microsoft Power BI*. [Online]. Available from: https://powerbi.microsoft.com/en-us/. [Accessed: 20/7/2017]

Mohabeer, H. & Soyjaudah, S.K.M. (2015). Application of Predictive Coding in Neuroevolution. *International Journal of Computer Applications*. [Online]. 114 (2). p. 41–47. Available from: http://research.ijcaonline.org/volume114/number2/pxc3901782.pdf [Accessed: 30/7/2017]

Monaco, J. V., Stewart, J.C., Cha, S.H. & Tappert, C.C. (2013). *Behavioral biometric verification of student identity in online course assessment and authentication of authors in literary works*. In IEEE 6th International Conference on Biometrics: Theory, Applications and Systems, BTAS 2013. p. 1-8.

Monaco, J. V. & Tappert, C.C. (2016). *The Partially Observable Hidden Markov Model and its Application to Keystroke Dynamics*. [Online] USA: arXiv. Available from: https://arxiv.org/pdf/1607.03854.pdf [Accessed: 11/7/2017]

Monaco, V. (n.d.). *Keystroke Dynamics Datasets*. [Online]. Available from: http://www.vmonaco.com/keystroke-datasets. [Accessed: 10/8/2017]

Montalvao, J., Almeida, C.A.S. & Freire, E.O. (2006). *Equalization of keystroke timing histograms for improved identification performance*. In 2006 International Telecommunications Symposium. September 2006, IEEE, p. 560–565.

Montalvao, J., Freire, E.O., Bezerra, M.A. & Garcia, R. (2015). Contributions to empirical analysis of keystroke dynamics in passwords. *Journal of Pattern Recognition Letters.* [Online] 52(1). p. 80-86. Available from: http://www.sciencedirect.com/science/article/pii/S0167865514003092 [Accessed: 11/7/2017]

Morales, A. & Fierrez, J. (2014). *Keystroke Biometrics for Student Authentication : A Case Study*. In Proceedings of the 2015 ACM Conference, 2015. [Online]. p. 28-49.

Nisha, J.. & Kumar, R.P.. (2014). User authentication based on keystroke dynamics analysis.

International Journal of Engineering Research and Applications. [Online] 4(3). p. 345-349. Available from: http://www.ijera.com/papers/Vol4_issue3/Version%201/BK4301345349.pdf [Accessed: 11/7/2017]

Pisani, P.H., Giot, R., De Carvalho, A.C.P.L.F. & Lorena, A.C. (2016). Enhanced template update: Application to keystroke dynamics. *Journal of Computers and Security*. [Online]. 60. p. 134–153. Available from: http://dx.doi.org/10.1016/j.cose.2016.04.004 [Accessed: 29/7/2017]

Pisani, P.H. & Lorena, A.C. (2013). A systematic review on keystroke dynamics. *Journal of the Brazilian Computer Society*. [Online] 19 (4). p. 573–587. Available from: http://dx.doi.org/10.1007/s13173-013-0117-7 [Accessed: 7/7/2017]

Pisani, P.H. & Lorena, A.C. (2015). Emphasizing typing signature in keystroke dynamics using immune algorithms. *Applied Soft Computing Journal*. [Online]. 34. p.pp. 178–193. Available from: http://dx.doi.org/10.1016/j.asoc.2015.05.008.

Pisani, P.H., Lorena, A.C. & de Carvalho, A.C.P.L.F. (2015). Adaptive Positive Selection for Keystroke Dynamics. *Journal of Intelligent and Robotic Systems*. [Online] 80(1). p. 277-293. Available from: https://link.springer.com/article/10.1007/s10846-014-0148-0?no-access=true [Accessed: 12/8/2017]

Pisani, P.H. & Pereira, S. do L. (2010). *Lamarckian Evolution of Neural Networks Applied To Keystroke Dynamics*. In Proceedings of the International Conference on Evolutionary Computation, 2010. p. 357-364.

Revett, K., Gorunescu, F., Gorunescu, M., Ene, M., Tenreiro De Magalhaes, S., Santos, D., Henrique, M., Magalhaes, S. & Santos, H. (2007). A machine learning approach to keystroke dynamics based user authentication. *International Journal of Electronic Security and Digital Forensics*. [Online]. 1 (1). p. 55. Available from: http://dx.doi.org/10.1504/IJESDF.2007.013592 [Accessed: 19/7/2017]

Roth, J., Liu, X., Ross, A. & Metaxas, D. (2013). *Acoustic and Visual Typing Behaviour Dataset*. [Online]. Available from: http://cvlab.cse.msu.edu/typing-behavior-dataset.html. [Accessed: 7/8/2017]

Roy, S., Roy, U. & Sinha, D.D. (2016). *Security Enhancement of Knowledge-based User Authentication through Keystroke Dynamics*. In MATEC ’16: International Conference on Advancements in Engineering and Technology (ICAET). India: Friday 18th March to Saturday 19th March 2016. India: EDP Sciences. pp. 1-9.

Sedenka, J., Balagani, K., Phoha, V. & Gasti, P. (2014). *Privacy-Preserving Population-*

Enhanced Biometric Key Generation from Free-Text Keystroke Dynamics. [Online]. Available from: http://arxiv.org/abs/1407.4179 [Accessed: 20/7/2017]

Shanmugapriya, D. & Padmavathi, G. (2009). A Survey of Biometric keystroke Dynamics: Approaches, Security and Challenges. *International Journal of Computer Science and Information Security*. [Online] 5(1). p. 115-119. Available from: https://arxiv.org/ftp/arxiv/papers/0910/0910.0817.pdf [Accessed: 25/7/2017]

Spantzel, A.B., Squicciarini, A.C., Modi, S., Young, M., Bertino, E. & Elliott, S.J. (2006). *Privacy preserving multi-factor authentication with biometrics.* In Proceedings of the second ACM workshop on Digital identity management. p. 63–72.

Spillane, R. (1975). *Keyboard Apparatus for Personal Identification*. United States.

Sutskever, I., Martens, J., Dahl, G. & Hinton, G. (2013). *On the importance of initialization and momentum in deep learning*. In, IEEE International Conference on Acoustics, Speech and Signal Processing - ICASSP, 2010. p. 13.

Syed, Z., Banerjee, S. & Cukic, B. (2014). Leveraging variations in event sequences in keystroke-dynamics authentication systems. In HASE '14: Proceedings of the 2014 IEEE 15th International Symposium on High-Assurance Systems Engineering. USA, Thursday 9th January to Saturday 11st January 2014. USA: IEEE. pp. 9-16.

Teh, P.S., Andrew Teoh, B.J., Ong, T.S. & Neo, H.F. (2007). *Statistical fusion approach on keystroke dynamics*. In Proceedings - International Conference on Signal Image Technologies and Internet Based Systems, SITIS 2007. January. p. 918–923.

Teh, P.S., Andrew Teoh, B.J., Tee, C. & Ong, T.S. (2010). *Keystroke Dynamics Benchmark Dataset*. [Online]. 2010. Available from: https://sites.google.com/site/keystrokedynamic/. [Accessed: 5/8/2017]

Traore, I., Woungang, I., Obaidat, M.S., Nakkabi, Y. & Lai, I. (2014). Online risk-based authentication using behavioral biometrics. *Journal of Multimedia Tools and Applications*. [Online] 71 (2). p. 575–605. Available from: http://dx.doi.org/10.1007/s11042-013-1518-5 [Accessed: 24/7/2017]

Uzun, Y. & Bicakci, K. (2012). A second look at the performance of neural networks for keystroke dynamics using a publicly available dataset. *Journal of Computers and Security*. [Online]. 31 (5). p. 717–726. Available from: http://dx.doi.org/10.1016/j.cose.2012.04.002 [Accessed: 17/7/2017]

Venugopalan, S., Juefei-Xu, F., Cowley, B. & Savvides, M. (2015). *Electromyograph and keystroke dynamics for spoof-resistant biometric authenticatio*n. In IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshop*s*. 2015–

Octob. p. 109–118.

Vielhauer, C. (2006). *Biometric User Authentication for IT Security: From Fundamentals to Handwriting*. [Online] New York: Springer. Available from: http://libgen.me/view.php?id=67251 [Accessed: 5/8/2017]

Viera, A.J. & Garrett, J.M. (2005). Understanding interobserver agreement: The Kappa Statistic. *Journal of Family Medicine*. [Online] 37 (5). p. 360–363. Available from: http://www1.cs.columbia.edu/~julia/courses/CS6998/Interrater_agreement.Kappa_statistic.pdf [Accessed: 29/8/2017]

Wangsuk, K. & Anusas, T.A. (2013). Trajectory mining for keystroke dynamics authentication. *17th Asia Pacific Symposium on Intelligent and Evolutionary Systems*. [Online] 24(11). p. 175-183. Available from: http://www.sciencedirect.com/science/article/pii/S1877050913011836 [Accessed: 12/8/2017]

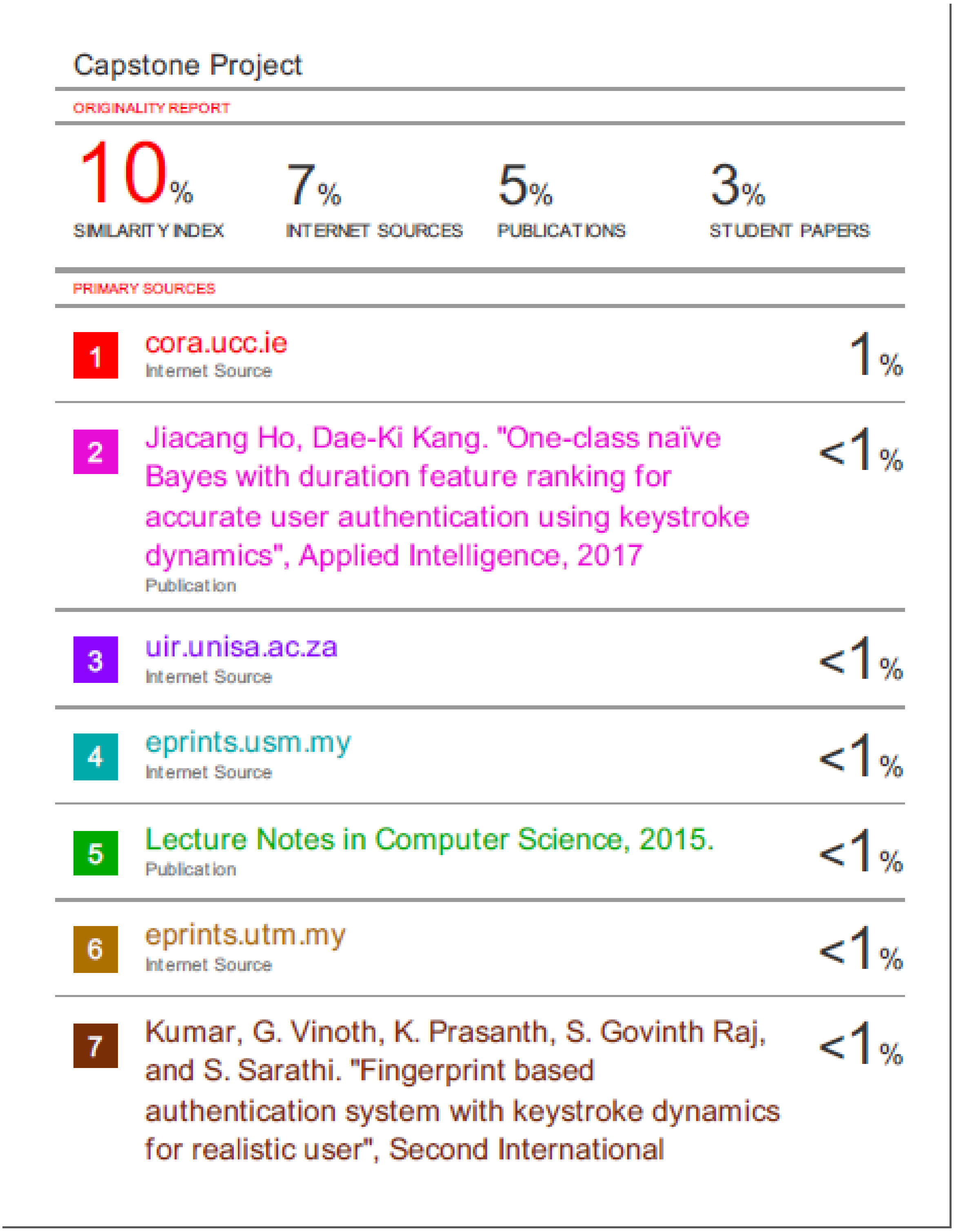
Xuan, W., Fangxia, G. & Jian-Feng, M. (2012). User authentication via keystroke dynamics based on difference subspace and slope correlation degree. *Digital Signal Processing: A Review Journal*. [Online]. 22 (5). p. 707–712. Available from: http://dx.doi.org/10.1016/j.dsp.2012.04.012 [Accessed: 28/7/2017]

Yampolskiy, R. V. & Govindaraju, V. (2008). Behavioural biometrics: a survey and classification. *International Journal of Biometrics*. [Online]. 1 (1). p. 81. Available from: http://www.inderscience.com/link.php?id=18665 [Accessed: 29/7/2017]

Zhong, Y. & Deng, Y. (2015). *A Survey on Keystroke Dynamics Biometrics: Approaches, Advances, and Evaluations*. [Online]. USA: Science Gate Publishing. p. 1-22. Available from: http://sciencegatepub.com/books/gcsr/gcsr_vol2/GCSR_Vol2_Ch1.pdf [Accessed: 28/7/2017]

Zurkus, K. (2016). *The future of passwords is no more passwords*. [Online]. Available from: https://www.csoonline.com/article/3120382/security/the-future-of-passwords-is-no-more-passwords.html. [Accessed: 1/7/2017]

APPENDIX A: TURNITIN REPORT



Capstone Project

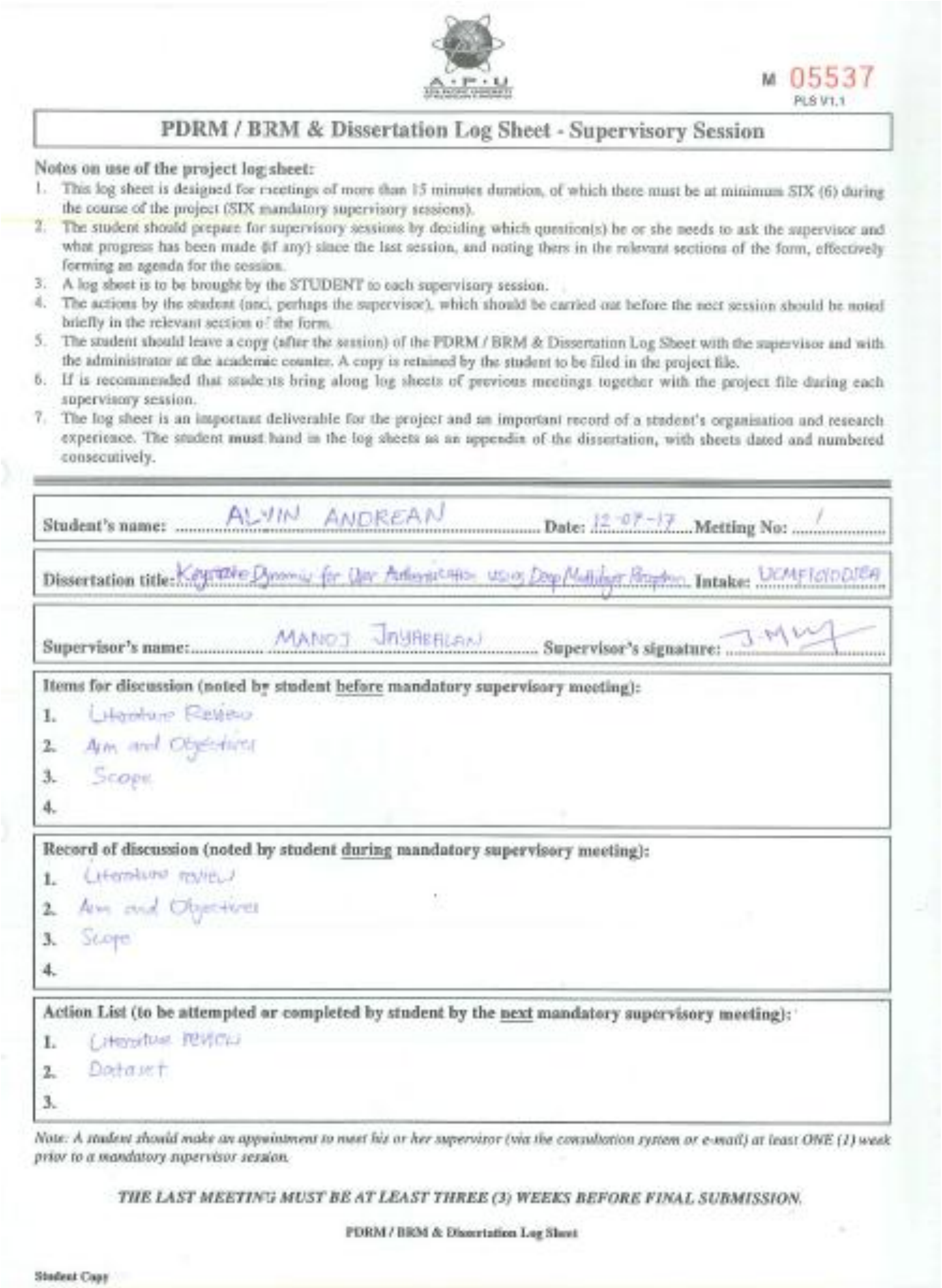
ORIGINALITY REPORT

10%	7%	5%	3%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	cora.ucc.ie Internet Source	1%
2	Jiacang Ho, Dae-Ki Kang. "One-class naïve Bayes with duration feature ranking for accurate user authentication using keystroke dynamics", Applied Intelligence, 2017 Publication	<1%
3	uir.unisa.ac.za Internet Source	<1%
4	eprints.usm.my Internet Source	<1%
5	Lecture Notes in Computer Science, 2015. Publication	<1%
6	eprints.utm.my Internet Source	<1%
7	Kumar, G. Vinoth, K. Prasanth, S. Govinth Raj, and S. Sarathi. "Fingerprint based authentication system with keystroke dynamics for realistic user", Second International	<1%

APPENDIX B: LOG SHEETS



A·P·U

M 05537

PLS V1.1

PDRM / BRM & Dissertation Log Sheet - Supervisory Session

Notes on use of the project log sheet:

1. This log sheet is designed for meetings of more than 15 minutes duration, of which there must be at minimum SIX (6) during the course of the project (SIX mandatory supervisory sessions).
2. The student should prepare for supervisory sessions by deciding which question(s) he or she needs to ask the supervisor and what progress has been made (if any) since the last session, and noting them in the relevant sections of the form, effectively forming an agenda for the session.
3. A log sheet is to be brought by the STUDENT to each supervisory session.
4. The actions by the student (and, perhaps the supervisor), which should be carried out before the next session should be noted briefly in the relevant section of the form.
5. The student should leave a copy (after the session) of the PDRM / BRM & Dissertation Log Sheet with the supervisor and with the administrator at the academic counter. A copy is retained by the student to be filed in the project file.
6. It is recommended that students bring along log sheets of previous meetings together with the project file during each supervisory session.
7. The log sheet is an important deliverable for the project and an important record of a student's organisation and research experience. The student **must** hand in the log sheets as an appendix of the dissertation, with sheets dated and numbered consecutively.

Student's name: ALVIN ANDREAN **Date:** 12-07-17 **Metting No:** 1

Dissertation title: Keystroke Dynamics for User Authentication using Deep Multilayer Perceptron **Intake:** UCMF1610DIEA

Supervisor's name: MANOJ JAYABALAN **Supervisor's signature:**

Items for discussion (noted by student before mandatory supervisory meeting):

1. Literature Review
2. Aim and Objectives
3. Scope
4.

Record of discussion (noted by student during mandatory supervisory meeting):

1. Literature review
2. Aim and Objectives
3. Scope
4.

Action List (to be attempted or completed by student by the next mandatory supervisory meeting):

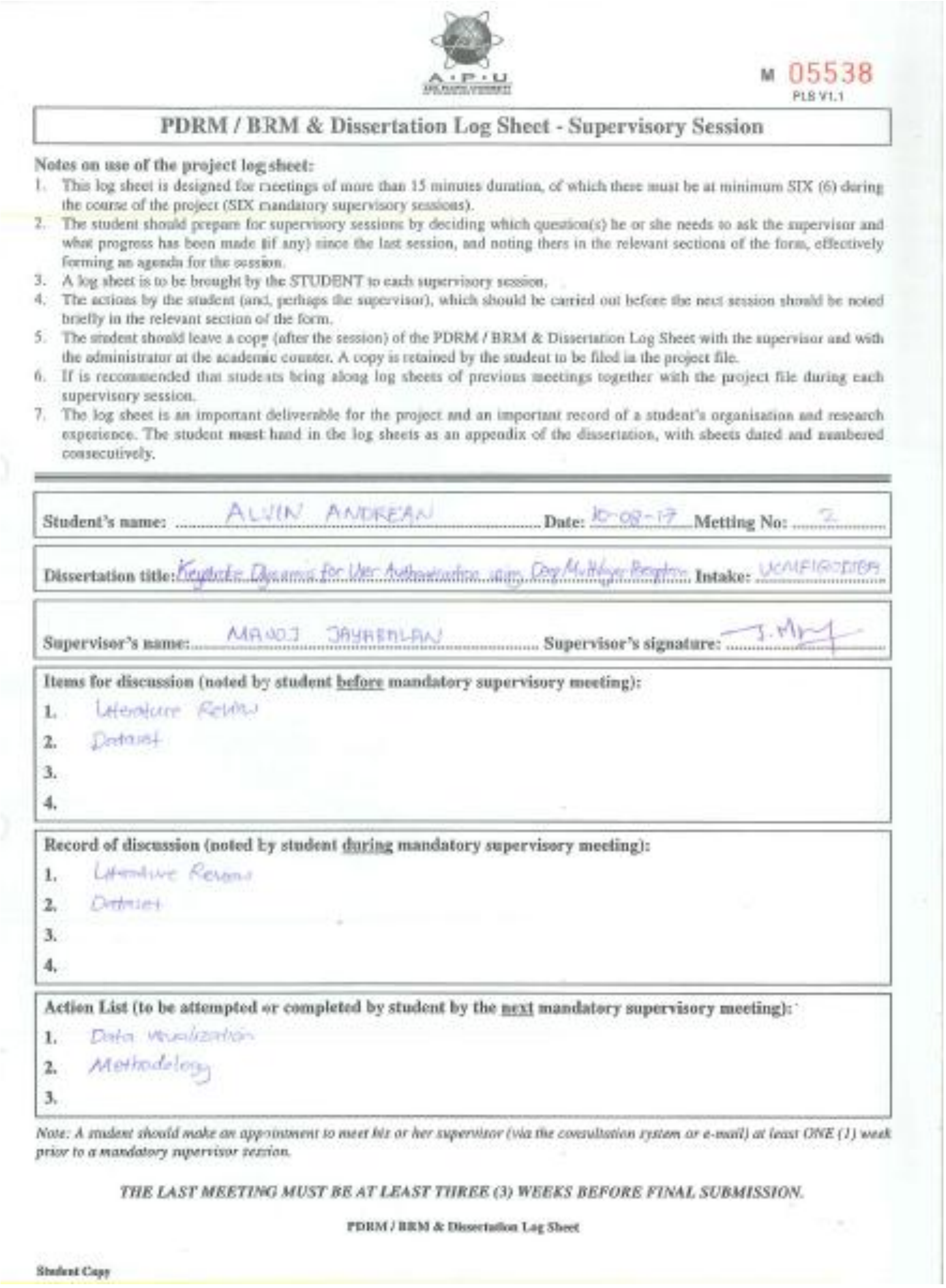
1. Literature review
2. Dataset
3.

Note: A student should make an appointment to meet his or her supervisor (via the consultation system or e-mail) at least ONE (1) week prior to a mandatory supervisor session.

THE LAST MEETING MUST BE AT LEAST THREE (3) WEEKS BEFORE FINAL SUBMISSION.

PDRM / BRM & Dissertation Log Sheet

Student Copy



A·P·U

M 05538

PLS V1.1

PDRM / BRM & Dissertation Log Sheet - Supervisory Session

Notes on use of the project log sheet:

1. This log sheet is designed for meetings of more than 15 minutes duration, of which there must be at minimum SIX (6) during the course of the project (SIX mandatory supervisory sessions).
2. The student should prepare for supervisory sessions by deciding which question(s) he or she needs to ask the supervisor and what progress has been made (if any) since the last session, and noting thers in the relevant sections of the form, effectively forming an agenda for the session.
3. A log sheet is to be brought by the STUDENT to each supervisory session.
4. The actions by the student (and, perhaps the supervisor), which should be carried out before the next session should be noted briefly in the relevant section of the form.
5. The student should leave a copy (after the session) of the PDRM / BRM & Dissertation Log Sheet with the supervisor and with the administrator at the academic counter. A copy is retained by the student to be filed in the project file.
6. If is recommended that students bring along log sheets of previous meetings together with the project file during each supervisory session.
7. The log sheet is an important deliverable for the project and an important record of a student's organisation and research experience. The student **must** hand in the log sheets as an appendix of the dissertation, with sheets dated and numbered consecutively.

Student's name: ALVIN ANDREAN **Date:** 10-08-17 **Metting No:** 2

Dissertation title: Keystroke Dynamics for User Authentication using Deep Multilayer Perceptron **Intake:** UCMF1607IT(IS)

Supervisor's name: MANOJ JAYABALAN **Supervisor's signature:**

Items for discussion (noted by student before mandatory supervisory meeting):

1. Literature Review
2. Dataset
3.
4.

Record of discussion (noted by student during mandatory supervisory meeting):

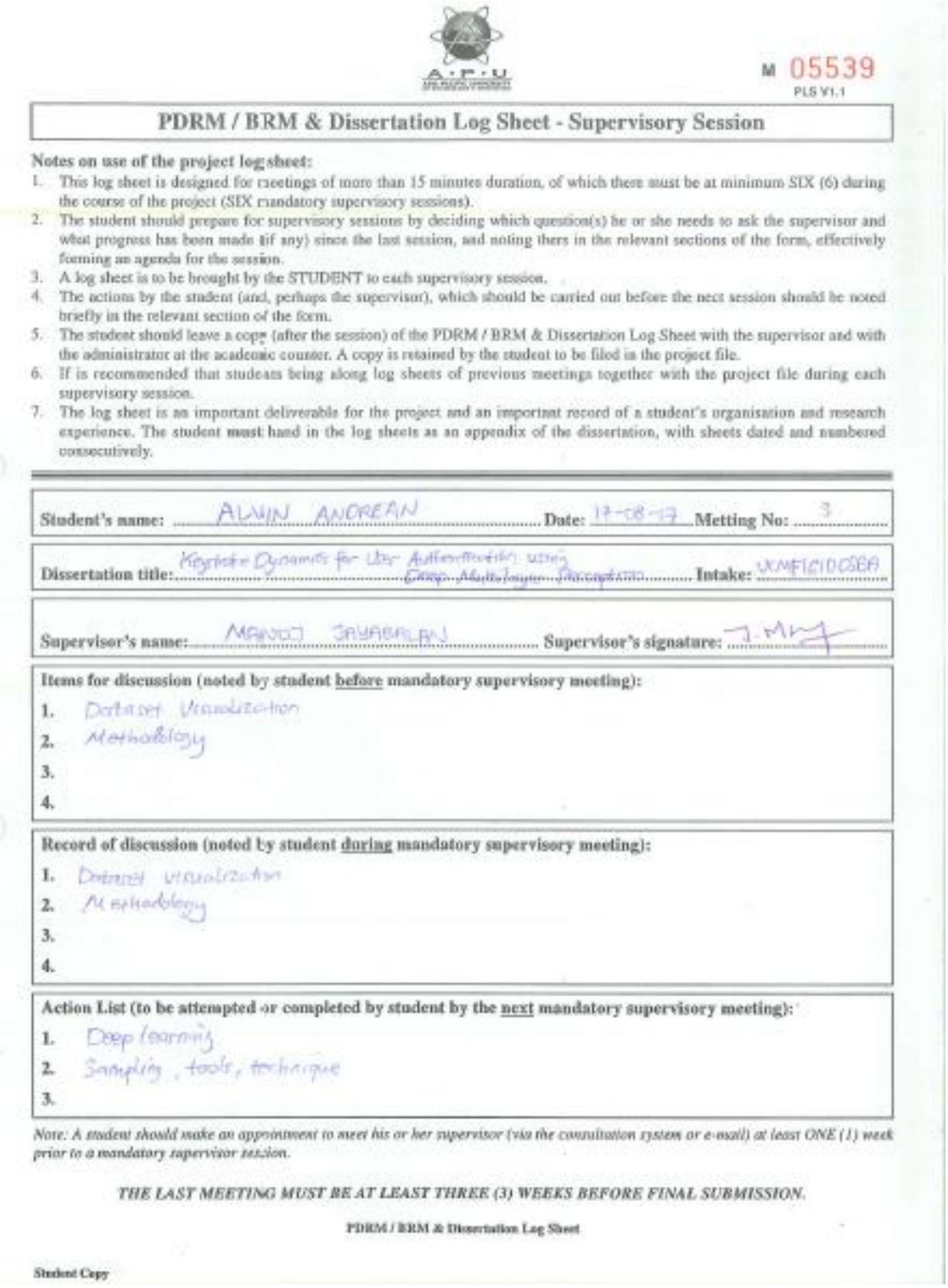
1. Literature Review
2. Dataset
3.
4.

Action List (to be attempted or completed by student by the next mandatory supervisory meeting):

1. Data visualization
2. Methodology
3.

Note: A student should make an appointment to meet his or her supervisor (via the consultation system or e-mail) at least ONE (1) week prior to a mandatory supervisor session.

THE LAST MEETING MUST BE AT LEAST THREE (3) WEEKS BEFORE FINAL SUBMISSION.



A·P·U

M 05539
PLS V1.1

PDRM / BRM & Dissertation Log Sheet - Supervisory Session

Notes on use of the project log sheet:

1. This log sheet is designed for meetings of more than 15 minutes duration, of which there must be at minimum SIX (6) during the course of the project (SIX mandatory supervisory sessions).
2. The student should prepare for supervisory sessions by deciding which question(s) he or she needs to ask the supervisor and what progress has been made (if any) since the last session, and noting thers in the relevant sections of the form, effectively forming an agenda for the session.
3. A log sheet is to be brought by the STUDENT to each supervisory session.
4. The actions by the student (and, perhaps the supervisor), which should be carried out before the nect session should be noted briefly in the relevant section of the form.
5. The student should leave a copy (after the session) of the PDRM / BRM & Dissertation Log Sheet with the supervisor and with the administrator at the academic counter. A copy is retained by the student to be filed in the project file.
6. If is recommended that students bring along log sheets of previous meetings together with the project file during each supervisory session.
7. The log sheet is an important deliverable for the project and an important record of a student's organisation and research experience. The student **must** hand in the log sheets as an appendix of the dissertation, with sheets dated and numbered consecutively.

Student's name: ALWIN ANDREAN **Date:** 17-08-17 **Metting No:** 3

Dissertation title: Keystroke Dynamics for User Authentication using Deep Multilayer Perceptron **Intake:** UCMF1610CSBA

Supervisor's name: MANOJ JAYABALAN **Supervisor's signature:** J.M

Items for discussion (noted by student <u>before</u> mandatory supervisory meeting):

1. Dataset Visualization
2. Methodology
3.
4.

Record of discussion (noted by student <u>during</u> mandatory supervisory meeting):

1. Dataset visualization
2. Methodology
3.
4.

Action List (to be attempted or completed by student by the <u>next</u> mandatory supervisory meeting):

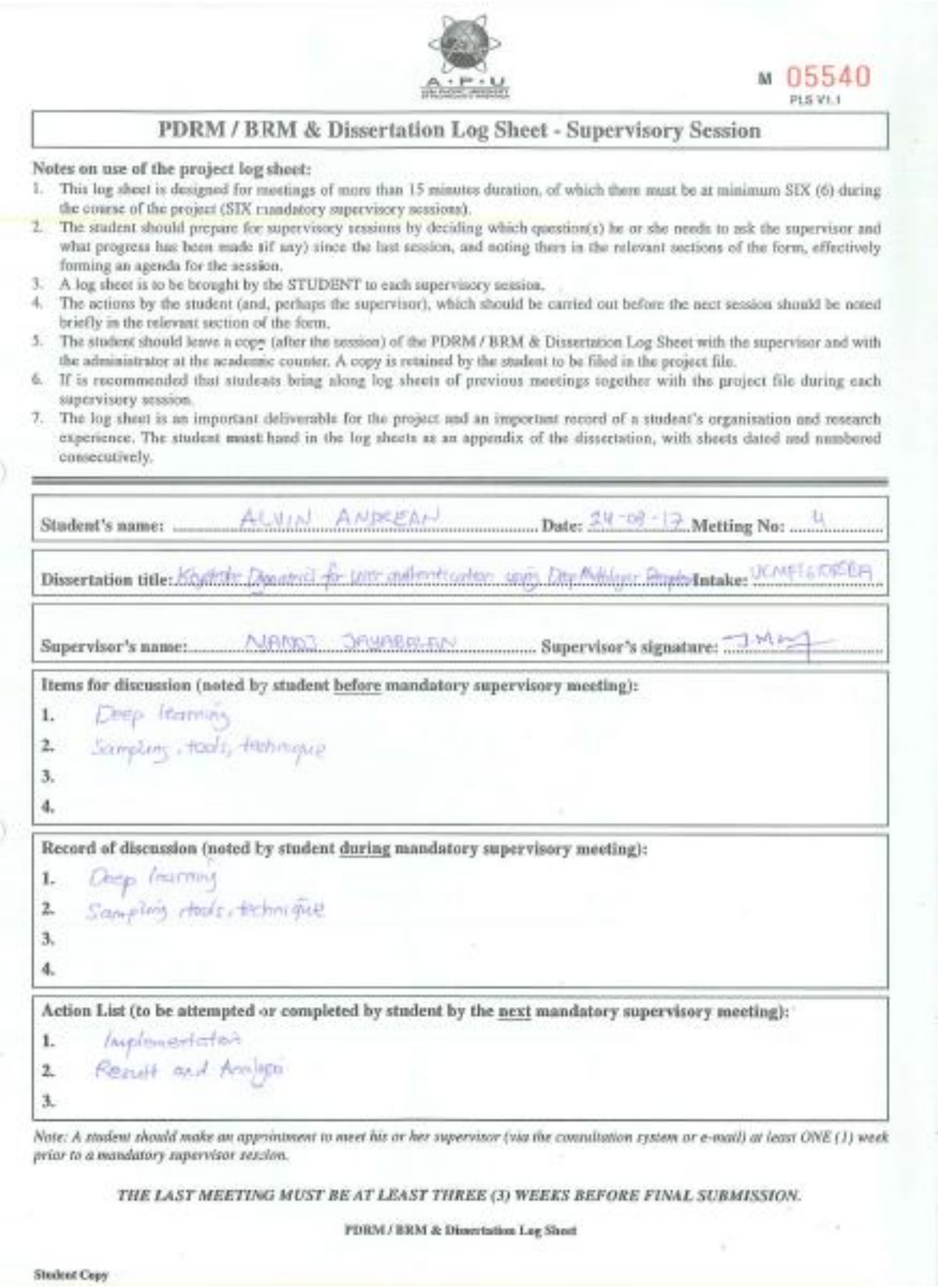
1. Deep learning
2. Sampling, tools, technique
3.

Note: A student should make an appointment to meet his or her supervisor (via the consultation system or e-mail) at least ONE (1) week prior to a mandatory supervisor session.

THE LAST MEETING MUST BE AT LEAST THREE (3) WEEKS BEFORE FINAL SUBMISSION.

PDRM / BRM & Dissertation Log Sheet

Student Copy



A·P·U

M 05540
PLS V1.1

PDRM / BRM & Dissertation Log Sheet - Supervisory Session

Notes on use of the project log sheet:

1. This log sheet is designed for meetings of more than 15 minutes duration, of which there must be at minimum SIX (6) during the course of the project (SIX mandatory supervisory sessions).
2. The student should prepare for supervisory sessions by deciding which question(s) he or she needs to ask the supervisor and what progress has been made (if any) since the last session, and noting them in the relevant sections of the form, effectively forming an agenda for the session.
3. A log sheet is to be brought by the STUDENT to each supervisory session.
4. The actions by the student (and, perhaps the supervisor), which should be carried out before the nect session should be noted briefly in the relevant section of the form.
5. The student should leave a copy (after the session) of the PDRM / BRM & Dissertation Log Sheet with the supervisor and with the administrator at the academic counter. A copy is retained by the student to be filed in the project file.
6. If is recommended that students bring along log sheets of previous meetings together with the project file during each supervisory session.
7. The log sheet is an important deliverable for the project and an important record of a student's organisation and research experience. The student **must** hand in the log sheets as an appendix of the dissertation, with sheets dated and numbered consecutively.

Student's name: ALVIN ANDREAN **Date:** 24-08-17 **Metting No:** 4

Dissertation title: Keystroke Dynamics for user authentication using Deep Multilayer Perceptron **Intake:** UCMF1610DBA

Supervisor's name: NAMOJ JAYABALAN **Supervisor's signature:**

Items for discussion (noted by student <u>before</u> mandatory supervisory meeting):

1. Deep learning
2. Sampling, tools, technique
3.
4.

Record of discussion (noted by student <u>during</u> mandatory supervisory meeting):

1. Deep learning
2. Sampling tools, technique
3.
4.

Action List (to be attempted or completed by student by the <u>next</u> mandatory supervisory meeting):

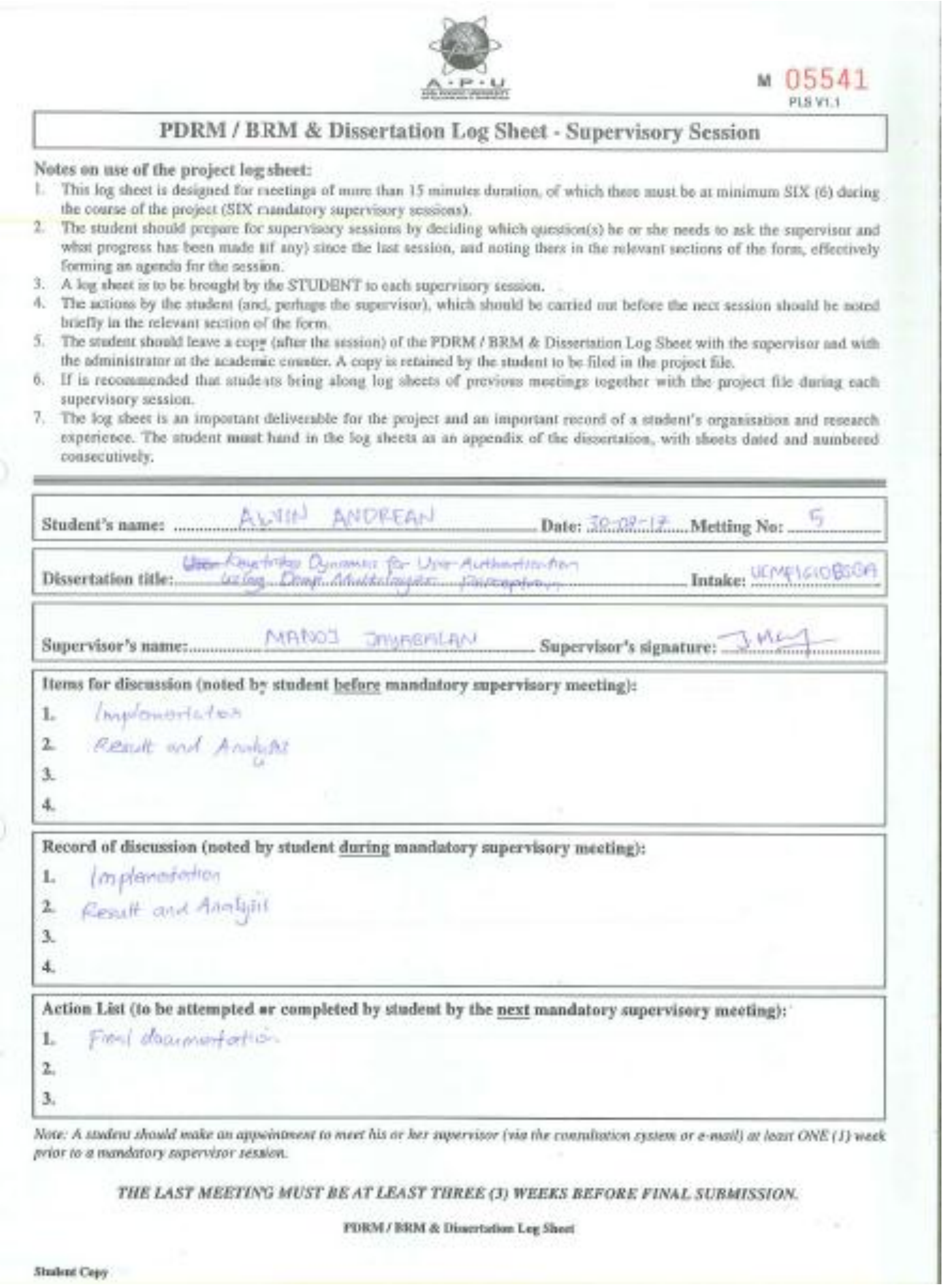
1. Implementation
2. Result and Analysis
3.

Note: A student should make an appointment to meet his or her supervisor (via the consultation system or e-mail) at least ONE (1) week prior to a mandatory supervisor session.

THE LAST MEETING MUST BE AT LEAST THREE (3) WEEKS BEFORE FINAL SUBMISSION.

PDRM / BRM & Dissertation Log Sheet

Student Copy



M 05541

PLS V1.1

PDRM / BRM & Dissertation Log Sheet - Supervisory Session

Notes on use of the project log sheet:

1. This log sheet is designed for meetings of more than 15 minutes duration, of which there must be at minimum SIX (6) during the course of the project (SIX mandatory supervisory sessions).
2. The student should prepare for supervisory sessions by deciding which question(s) he or she needs to ask the supervisor and what progress has been made (if any) since the last session, and noting them in the relevant sections of the form, effectively forming an agenda for the session.
3. A log sheet is to be brought by the STUDENT to each supervisory session.
4. The actions by the student (and, perhaps the supervisor), which should be carried out before the next session should be noted briefly in the relevant section of the form.
5. The student should leave a copy (after the session) of the PDRM / BRM & Dissertation Log Sheet with the supervisor and with the administrator at the academic counter. A copy is retained by the student to be filed in the project file.
6. If is recommended that students bring along log sheets of previous meetings together with the project file during each supervisory session.
7. The log sheet is an important deliverable for the project and an important record of a student's organisation and research experience. The student **must** hand in the log sheets as an appendix of the dissertation, with sheets dated and numbered consecutively.

Student's name: ALVIN ANDREAN **Date:** 30-08-17 **Metting No:** 5

Dissertation title: ~~User~~ Keystroke Dynamics for User Authentication using Deep Multilayer Perceptron **Intake:** UCMF1610BSCA

Supervisor's name: MANOJ JAYABALAN **Supervisor's signature:** J.Mey

Items for discussion (noted by student before mandatory supervisory meeting):

1. Implementation
2. Result and Analysis
3.
4.

Record of discussion (noted by student during mandatory supervisory meeting):

1. Implementation
2. Result and Analysis
3.
4.

Action List (to be attempted or completed by student by the next mandatory supervisory meeting):

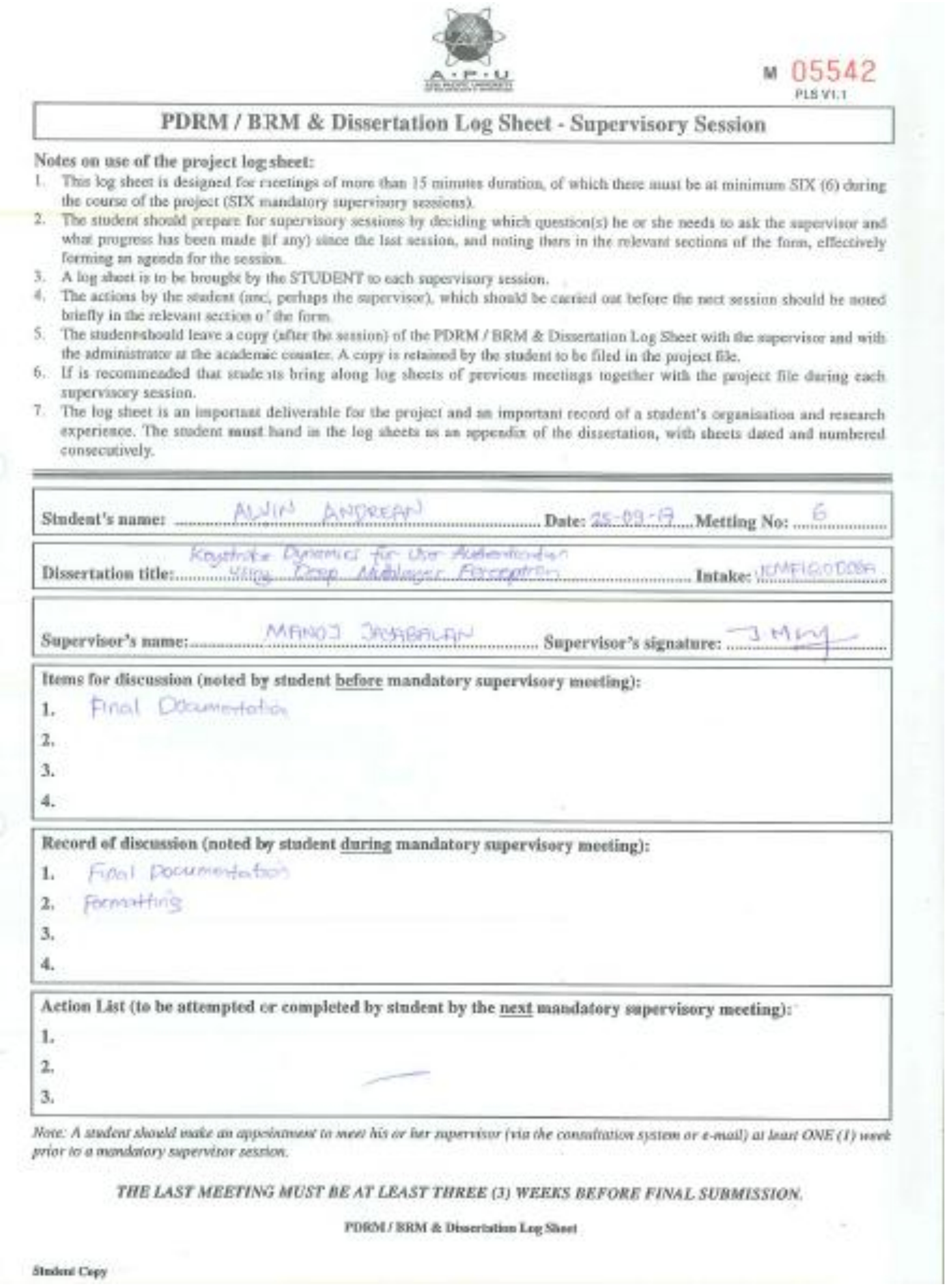
1. Final documentation
2.
3.

Note: A student should make an appointment to meet his or her supervisor (via the consultation system or e-mail) at least ONE (1) week prior to a mandatory supervisor session.

THE LAST MEETING MUST BE AT LEAST THREE (3) WEEKS BEFORE FINAL SUBMISSION.

PDRM / BRM & Dissertation Log Sheet

Student Copy



A·P·U

M 05542

PLS V1.1

PDRM / BRM & Dissertation Log Sheet - Supervisory Session

Notes on use of the project log sheet:

1. This log sheet is designed for meetings of more than 15 minutes duration, of which there must be at minimum SIX (6) during the course of the project (SIX mandatory supervisory sessions).
2. The student should prepare for supervisory sessions by deciding which question(s) he or she needs to ask the supervisor and what progress has been made (if any) since the last session, and noting them in the relevant sections of the form, effectively forming an agenda for the session.
3. A log sheet is to be brought by the STUDENT to each supervisory session.
4. The actions by the student (and, perhaps the supervisor), which should be carried out before the next session should be noted briefly in the relevant section o' the form.
5. The student should leave a copy (after the session) of the PDRM / BRM & Dissertation Log Sheet with the supervisor and with the administrator at the academic counter. A copy is retained by the student to be filed in the project file.
6. If is recommended that stude its bring along log sheets of previous meetings together with the project file during each supervisory session.
7. The log sheet is an important deliverable for the project and an important record of a student's organisation and research experience. The student **must** hand in the log sheets as an appendix of the dissertation, with sheets dated and numbered consecutively.

Student's name: ALVIN ANDREAN **Date:** 25-09-19 **Metting No:** 6

Dissertation title: Keystroke Dynamics for User Authentication Using Deep Multilayer Perceptron **Intake:** ICMF1601DDBA

Supervisor's name: MANOJ JAYABALAN **Supervisor's signature:** J M

Items for discussion (noted by student before mandatory supervisory meeting):

1. Final Documentation
2.
3.
4.

Record of discussion (noted by student during mandatory supervisory meeting):

1. Final Documentation
2. Formatting
3.
4.

Action List (to be attempted or completed by student by the next mandatory supervisory meeting):

1.
2.
3.

Note: A student should make an appointment to meet his or her supervisor (via the consultation system or e-mail) at least ONE (1) week prior to a mandatory supervisor session.

THE LAST MEETING MUST BE AT LEAST THREE (3) WEEKS BEFORE FINAL SUBMISSION.

PDRM / BRM & Dissertation Log Sheet

Student Copy