FRAUD DETECTION THROUGH KEYSTROKE DYNAMICS BASED ON TIME, LOCATION CHARACTERISTICS AND KEYBOARD PRESSURE

Project Work Report

Submitted in partial fulfillment of the requirements for the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER ENGINEERING

by

SHIVA SWAROOP V (16CO252)

SRI CHARAN REDDY M (16CO228)

UMESH SAI R (16CO240)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

NATIONAL INSTITUTE OF TECHNOLOGY KARNATAKA,

SURATHKAL, MANGALORE - 575025

June, 2020

DECLARATION

We hereby declare that the project Work Report entitled **FRAUD DETECTION THROUGH KEYSTROKE DYNAMICS BASED ON TIME, LOCATION CHARACTERISTICS AND KEYBOARD PRESSURE** which is being submitted to the **National Institute of Technology Karnataka, Surathkal** in partial fulfilment of the requirements for the award of the Degree of **BACHELOR OF TECHNOLOGY** in **Computer Engineering** is a *bonafide report of the work carried out by us.* The material contained in this report has not been submitted to any University or Institution for the award of any degree.

Shiva Swaroop V (16CO252)
Department of Computer Science and Engineering

Sri Charan Reddy M (16CO228)
Department of Computer Science and Engineering

Umesh Sai R (16CO240)
Department of Computer Science and Engineering

Place: NITK, Surathkal.
Date: 15-06-2020

CERTIFICATE

This is to *certify* that the B.Tech Project Work Report entitled **FRAUD DETECTION THROUGH KEYSTROKE DYNAMICS BASED ON TIME, LOCATION CHARACTERISTICS AND KEYBOARD PRESSURE** submitted by **SHIVA SWAROOP V** (Register Number: 16CO252), **SREE CHARAN M** (Register Number: 16CO228) and **UMESH SAI R** (Register Number: 16CO240) as the record of the work carried out by them, is accepted as the *B.Tech. Project Work Report submission* in partial fulfilment of the requirements for the award of degree of **Bachelor of Technology** in **Computer Engineering**.

Prof. Shashidhar G. Koolagudi
Project Guide

Prof. Alwyn Roshan Pais
Chairman - DUGC

Acknowledgment

We would like to thank the institution, National Institute of Technology Karnataka, for providing the necessary infrastructure and support needed to conduct this project. We also thank the Department of Computer Science and its faculty for providing insight and expertise that greatly assisted the research.

We thank Prof. Shashidhar G. Koolagudi for his valuable guidance throughout this project, and our mentor ms. Nagaratna for her comments that greatly improved the quality of this work. We are thankful to them for the encouragement they have given us in completing the project.

Lastly, we would like to express our deep appreciation towards our classmates and our indebtedness to our parents for providing us the moral support and encouragement we needed during the course of this project.

Place: Surathkal

Date: 15/06/2020

Abstract

Keystroke dynamics, keystroke biometrics, typing dynamics and lately typing biometrics, is the detailed timing information which describes exactly when each key was pressed and when it was released as a person is typing at a computer keyboard. It is the study of whether people can be distinguished by their typing rhythms, much like handwriting is used to identify the author of a written text. Possible applications include acting as an electronic fingerprint, or in an access-control mechanism. A digital fingerprint would tie a person to a computer-based crime in the same manner that a physical fingerprint ties a person to the scene of a physical crime. Access control could incorporate keystroke dynamics both by requiring a legitimate user to type a password with the correct rhythm, and by continually authenticating that user while they type on the keyboard. Our project considers an issue of protecting data from unauthorized access by users' authentication through keystroke dynamics. It proposes to use keyboard pressure parameters in combination with time characteristics of keystrokes and location parameters to distinguish between the genuine user and fraud. We are to design a keyboard with special sensors that allow recording complementary parameters. We also present an estimation of the information value for these new characteristics and error probabilities of users' identification. Our work also proves that keystroke dynamics accompanied with time and location parameters can help in fraud detection better than that using Keystroke dynamics alone.

Keywords: Keystroke Dynamics, Biometrics, Fraud Detection, PyXHook, Arduino.

Chapter 1

Introduction

Nowadays IT penetration into society is an active process. A growing number of web-services appears. Many countries tend to build an electronic government to provide services to their citizens. The level of confidentiality to such services must be the highest. But the majority of confidential information leaks, and cyber attacks happen due to the web services. The world's number of leakages annually grows. The available estimations of the world financial losses due to these incidents impress, they account for $375-575 billion annually. Organizations of different levels deploy biometric security systems to decrease these losses. As of October 2016 biometric authentication is used by 57% of enterprises. Statistic biometric images (a fingerprint or an iris) are not private, so they can be copied by making a material or digital model (for remote authentication). Private biometric images contain a secret (a password) so they can potentially provide a higher level of protection. They include personal keystroke dynamics disclosed while typing a password phrase. A weak point of the method of authentication through keystroke dynamics is relatively low reliability of decisions made as the probability of false rejection error (FRR) and false access rate error are too significant to use this method in practice. This work investigates the issue of increasing the reliability of personal recognition through keystroke dynamics by using additional characteristics that describe the dynamics of text typing: key pressure parameters (pressure force) and Location and Time parameters.

Keystroke dynamics is part of a larger class of biometrics known as behavioral biometrics; a field in which observed patterns are statistical in nature. Because of this inherent uncertainty, a commonly held belief is that behavioral biometrics are not as reliable as biometrics used for authentication based on physically observable

characteristics such as fingerprints or retinal scans or DNA. The reality here is that behavioral biometrics use a confidence measurement instead of the traditional pass/-fail measurements. One of the major problems that keystroke dynamics runs into is that a person's typing varies substantially during the day and between different days, and may be affected by any number of external factors. Because of these variations, any system will make false-positive and false-negative errors. Some of the successful commercial products have strategies to handle these issues and have proven effective in large-scale use (thousands of users) in real-world settings and applications.

The combination of the users keystrokes patterns as well as the password used in password enabled applications results in a hardened password. This newly generated hardened password is very difficult for an impostor to crack. But this technique has a risk associated with it. i.e.., if there is a slight change in the users typing pattern, then the user may be unable to login. So Users have to use the same keyboard and certain recovery methods have to be used. Keystroke dynamics is the easiest method of behavioural biometrics as no additional hardware is required. But with the invent and use of keyloggers, methods had to be found out to increase the durability of authentication. So Hidetoshi Nonaka et al experimented with using pressure sensing along with keystroke analysis. The system was tested with few users and now work is under way to involve more diverse users to test.

1.1 Issues and Challenges

The keystroke rhythms of a user are measured to develop a unique bio metric template of the user's typing pattern for future authentication.Data needed to analyze keystroke dynamics is obtained by keystroke logging.for example, When reading email, the receiver cannot tell from reading the phrase "I saw 3 zebras!" whether:

- That was typed rapidly or slowly.
- The sender used the left shift key, the right shift key, or the caps-lock key to make the "i" turn into a capitalized letter "I".
- The letters were all typed at the same pace, or if there was a long pause before any characters while looking for that key.

- The sender typed any letters wrong initially and then went back and corrected them, or if they got them right the first time.

Similarly Common endings, such as "ing", may be entered far faster than, say, the same letters in different order to a degree that varies consistently by person. This consistency may hold and may reveal the person's native language's common sequences even when they are writing entirely in a different language, just as revealing as an accent might in spoken English. So this differs from person to person, which can be the basis of our assumption of detecting the fraud using their keystroke biometrics with extra features included.

So, The issues are to find the best methods to obtain the information regarding the above mentioned points. which can help us in knowing more about the user. The further challenge after learning about the user is to distinguish the user from others by using the data obtained from him.

1.2 Outline of the report

In this report, we explain about our work in more detail. This report further consists of following sections:

1. **Literature Survey**: In this section, We explain about the main ideas of the works previously done in this field of keystroke dynamics.

2. **Implementation Methodology**: In this section, We present our approach or idea of implementation. We further explain about the implementation of the hardware and software which is required to obtain the keystroke dynamics. The obtained data is also analysed to make sure that this information can be useful to proceed further.

3. **Fraud Detection**: In this section of the report, We deal with data. We try different algorithms on the data previously extracted which helps us in achieving our task.

4. **Results**: In this section of the report, we evaluate and compare the models with different metrics in-order to select the best model which helps us in detecting the intruders effectively.

5. **Conclusion**: In this section of the report, we conclude our project by briefing about the work we done, shortcomings of our work and the future work to be done which can help us in overcoming those shortcomings.

Further, we have mentioned all the publications at the end of the report which we have referred throughout our work.

Chapter 2

Literature Survey

In this section, we explain about the outline of different existing technologies and works done previously by different authors in this area of keystroke dynamics. Keystroke dynamics has become an active research area due to the increasing importance of cyber security and computer or network access control. There are generally two types of authentications using these keystroke dynamics. Biometrics [1][2][3][4][5][6] based on “who” is the person or “how” the person behaves present a significant security advancement to meet many new challenges. Among them, keystroke dynamics [7][8][9] provides a natural choice for secure “password-free” computer access. Keystroke dynamics refers to the habitual patterns or rhythms an individual exhibits while typing on a keyboard input device.As early as in the 19th century, telegraph operators used to recognize one other based on one’s specific tapping style[10].This suggests that keystroke dynamics contain sufficient information to serve as a potential biometric identifier to ascertain a specific keyboard user. Many of the research studies focus on the static verification of the password (i.e., a pre-enrolled string of characters chosen by the user) along with these keystroke dynamics[12]. Here these keystroke biometrics captured are used for the authentication purposes. The other type is the most challenging problem of keystroke biometrics using "free text"[7][13][14],where there is no pre-enrolled string to be typed but the user is to type any arbitrary string. Only few researchers address the above point.For our work we concentrate on the static verification as our work is fraud detection deals with the former type.

In their study on keystroke analysis using free text, Sim and Janakiraman [15] studied about the digraphs and their properties.Also they have studied general n-graphs for free text keystroke biometrics in which there is no password.Keystroke biometrics

are the only way of authentication. They concluded that the above studies resulted as discriminative only when the biometrics are word-specific.In their work,Gaines et al. [16] did their study on keystroke dynamics based authentication using T-test on digraph features. Monrose and Rubin [7] later obtained the features(biometrics) using the mean and variance of the n-graphs obtained through the above studies.They have obtained 92% accuracy in identifying the correct cases from their dataset using 63 users.

Later In thier work, Bergadano et al.[17] and later Gunetti and Picardi[18] also studied these n-graphs.They have extracted the features using the relative order of times of duration for different n-graphs. They proved with their work that the above relative feature, when combined with the existing features using absolute timing,improved the authentication performance using free text. In many works, keystroke biometrics research has utilized many existing machine learning and classification techniques. Both classical and advanced classifiers have been used, including K-Nearest Neighbor (KNN) classifiers [23], K-means methods [24], Bayesian classifiers [7], Fuzzy logic [25], neural networks [25][26], and support vector machines (SVMs)[22]. Different distance metrics, such as the Euclidean distance [19][7] and the Manhattan distance [20][21] have also been explored in these machine learning algorithms. Since the password varies for each subject, There will be different datasets and different evaluation metrics used different algorithms for each subject. So, it is difficult to compare results of different algorithms. This issue has been addressed in the papers [26][27], where a standard experimental platform for progress assessment has been provided.

2.1 Problem Statement

Keystroke biometrics can become useless in distinguishing the genuine and fraud user when the features are very less or more. So, We need to figure out different problems like What are the different features to be extracted, How to extract them efficiently and what are the different algorithms to be used on the extracted features in order to obtain better fraud detection.

2.2 Objectives

The primary objectives of our work are as follows:

1. To figure out new biometrics which can help us in fraud detection.
2. To implement the software or hardware required to collect those biometrics.
3. To find out the best model on which the data can fit perfectly.i.e., to use the obtained data in a right way so that it performs well in fraud detection.

Chapter 3

Implementation Methodology

For our approach, the general idea is to extract the desired features by building a pressure sensitive keyboard and apply different algorithms on the extracted features in order to obtain better fraud detection.

Whereas in our model we use machine learning algorithms for the threshold calculation and decision making as there are many ways for such calculation like SVM, KNN classifications etc.. . The whole model can be divided into four distinct steps. These are listed as follows:

- The individual user will be given 10 sessions in order to maintain the diversity in data where he types his password for 10 times. This results in the creation of different datasets for each user which consists of the time,location characteristics.
- A fraud dataset is also created for the individual user for the machine to be trained in order to classify whether the given details represent a fraud or a genuine user.
- The genuine data is labelled as 1 and the fraud data is being labelled as 0 and the whole data is shuffled and split to 80% train data and 20% test data.
- A classification model is selected and the train data is fitted into that model and this model is used for the evaluation or prediction of the nature of the user whether he is genuine or fraud

So, this model re-authenticates the users using his/her biometrics even after the authentication is done using username, password which helps in providing extra se-

curity. The flowchart of this model when being used in other applications will be of 2 modes (i.e., new user, old user) . It will be as shown in Fig.3.1.

3.1 Feature Extraction

This is the hectic step among the whole series of steps. In this section, we are to obtain different keystroke biometrics, which are unique for different individuals and help in improving the fraud detection. Such unique Keystroke biometrics involve timing parameters and Pressure parameters. This stage involves building the pressure sensitive keyboard which is used to capture the pressure characteristics of keystrokes. Keystroke dynamics features are usually extracted using the timing information of the key down/hold/up events and the pressure information is also to be extracted as they play an important role in classification since timing patterns become similar among many individuals but this feature helps when the former is compromised. The hold time or dwell time of individual keys, and the latency between two keys, i.e., the time interval between the release of a key and the pressing of the next key are typically exploited.

3.1.1 Extraction of Timing Parameters

In this step, The hold time or dwell time of individual keys, and the latency between two keys, i.e., the time interval between the release of a key and the pressing of the next key are typically exploited. Timing parameters of keystrokes are as shown in Fig.3.2:

The timing parameters represented in Fig.3.2 are obtained between any two consecutive keystrokes. The representations of above parameters are as follows:

- Hold Time : key delay between pressed and key released.
- Press-Press Time: time in between two consecutive presses.
- Release-Release Time : time between two successive releases.
- Release-Press Time : time in between the current key release and the next key press.

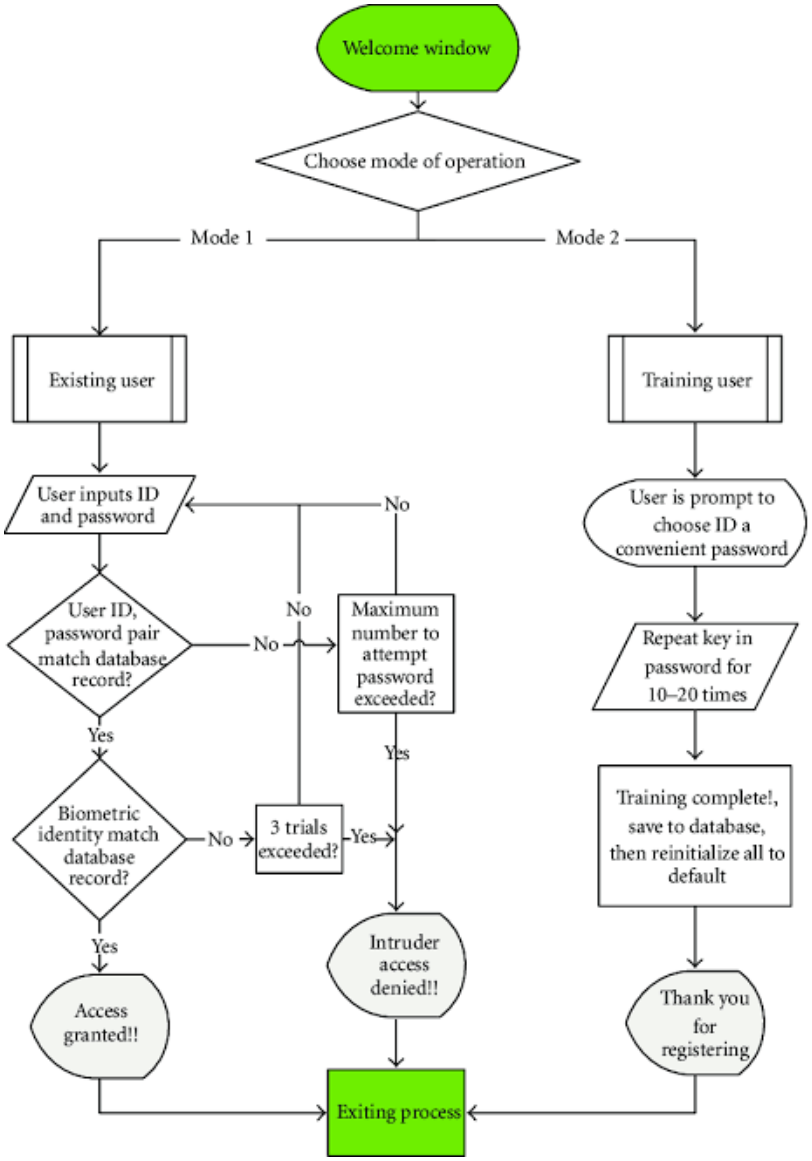


Figure 3.1: General Flowchart of our methodology

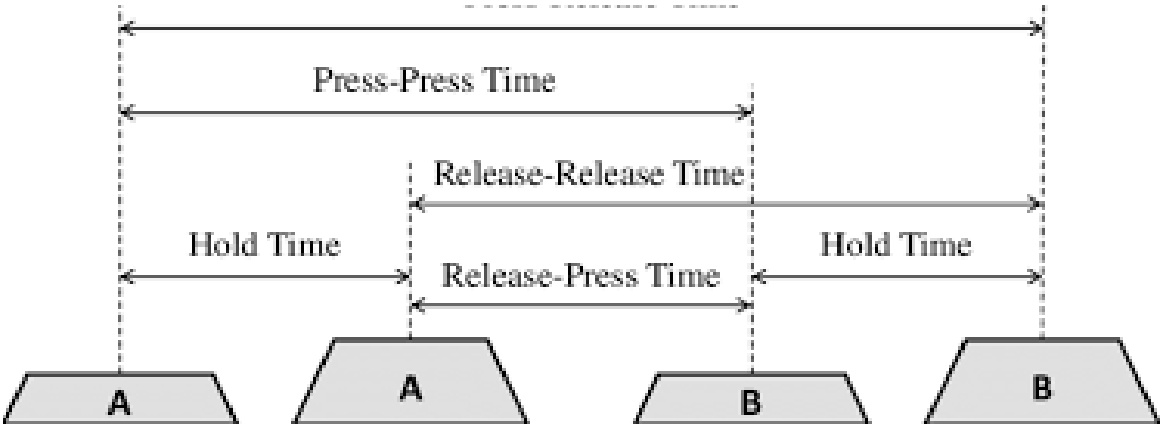


Figure 3.2: Different Timing Parameters between consecutive keys A,B

- Press-Release Time : time between the current key press and the next key release.

Now, we are to obtain these timings information and store them. For this task, a python package named **PYXHOOK** is available for capturing the keyboard events in linux operating system. The basic usage of the package is as shown in Fig.3.3.

The Fig 3.3 explains the general concept to capture key press in linux using python. We have gone through the documentations of the above mentioned library to understand the functionalities that this library can perform and their internal workings. Since pyx hook works only on linux, we worked on the oracle virtual machine. For the ease of the recording, we have considered the passwords to be the names of the subjects itself. Since a person's typing pattern of his own name may show the clear difference from the other's typing pattern. So, this consists of two python files namely timing.py, append_in_dataset.py.

The pyHook library wraps the low-level mouse and keyboard hooks in the Windows Hooking API for use in Python applications. Since we need only the keyboard events, We need to hook a keyboard to the hookmanager object as follows:

```
# Create hookmanager
hookman = pyxhook.HookManager()
# Define our callback to fire when a key is pressed down
hookman.KeyDown = kb_down_event
# Define our callback to fire when a key is pressed down
hookman.KeyUp = kb_up_event
# Hook the keyboard
hookman.HookKeyboard()
# Start our listener. Threads can only be started once.
hookman.start()
```

The below mentioned snippets are the functions responsible for noting the timestamp of the keystroke whenever pressed and released.

```
# This function is called every time a key is pressed down
def kb_down_event(event):
    try:
        key_timings[event.Key]["keyUp"] = time.time()
         if event.Key == "Return":
             print("key_timings[{}][\"keyUp\"]::{}".format(event.Key,
             key_timings[event.Key]["keyUp"]))
         print("key_timings[{}][\"keyUp\"]::{}".format(event.Key,
         key_timings[event.Key]["keyUp"]))
    except KeyError:
         print("This key is not to be recorded : ", event.Key)
        pass


# This function is called every time a keypress is released
def kb_up_event(event):
    try:
        key_timings[event.Key]["keyDown"] = time.time()
         if event.Key == "Return":
             print("key_timings[{}][\"keyDown\"]::{}".format(event.Key,
             key_timings[event.Key]["keyDown"]))
    except KeyError:
         print("This key is not to be recorded : ", event.Key)
        Pass
```

So, in order to obtain the time for different events for example, to find the time for which a key is being pressed, we are to subtract the timestamp at the press of the key(`key_timings['keyDown']`) from the timestamp at the release of the key(`key_timings['keyUp']`). The above part in the code is as mentioned below:

```
    dataset_based_timings["hold_time"]["Return"] =
    key_timings["Return"]["keyDown"] - key_timings["Return"]["keyUp"]

dataset_based_timings["ud_key1_key2"]["UD." + list(password)[-1] + ".Return"] =
key_timings["Return"]["keyDown"] - key_timings[list(password)[-1]]["keyUp"]

dataset_based_timings["dd_key1_key2"]["DD." + list(password)[-1] + ".Return"] =
key_timings["Return"]["keyDown"] - key_timings[list(password)[-1]]["keyDown"]
```

All the timing information is saved into a json file and all these json objects are then easily inserted into a dataset. Since, each person's name may differ from others with respect to the number of characteristics in a person's name. At the end, Each subject will be having a dataset which consists of the genuine timing information (where he/she types his/her password) and fraud timing information (where his/her password is typed by others). The attributes of the dataset are as follows:

```
    column_names = []
    for i in range(len(password)-1):
        column_names.append("H."+password[i])
        column_names.append("DD."+password[i]+"."+password[i+1])
        column_names.append("UD."+password[i]+"."+password[i+1])
    column_names.append("H."+password[i+1])
    column_names.append("DD."+password[i+1]+".Return")
    column_names.append("UD."+password[i+1]+".Return")
    column_names.append("H.Return")
```

Where "H.s" denotes hold time of key s, "DD.s.h" denotes Press-Press Time of keys s,h and etc.. Return indicates the end of the password.It is treated as a character although nothing exists at the end. The same process is repeated for n times to collect the timing information of the subject. The process is repeated at different times of

the day to consider the fact that these characteristics may differ during different times of a day. The below mentioned snippet meant the following:

```
while password_entry_count <= frequency_password_entry:
print("enter {} times more!".format(1+frequency_password_entry-password_
entry_count))
input_pwd = input("Enter \'{}\' : ".format(password))
is_pwd_correct = False
if input_pwd == password:
    print("pwd correct!")
    is_pwd_correct = True
    ...
    ...
```

These Key Latencies are calculated using the pyxhook package in python. The time characteristics mentioned above are stored to a json file and they are appended to a dataset after the 10 sessions of the user are taken. The data was not being collected at the same part of the day. Each session is recorded at different times to maintain that diversity in our user's typing patterns.

3.1.2 Extraction of Pressure Parameters

This step involves building the pressure sensitive keyboard. This involves in working with the hardware components.The hardware components required for building such system are given below in the Fig 3.4.

The hardware components represented in Fig 3.4 are the primary and essential components required to construct the pressure sensitive keyboard. The components in the above figure are listed in further detail below:

- Arduino UNO R3 : A device which converts the analogue pressure signals to digital
- 3 force-measuring resistors(Interlink 408 FSR) : A sensor to measure pressure

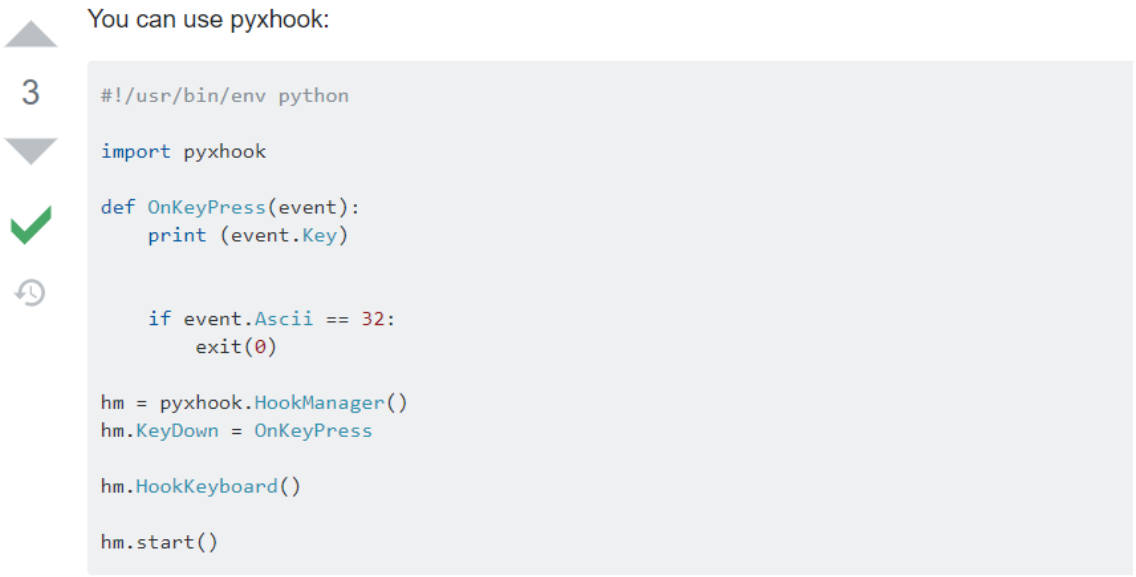


Figure 3.3: Basic keystroke event on pyxhook

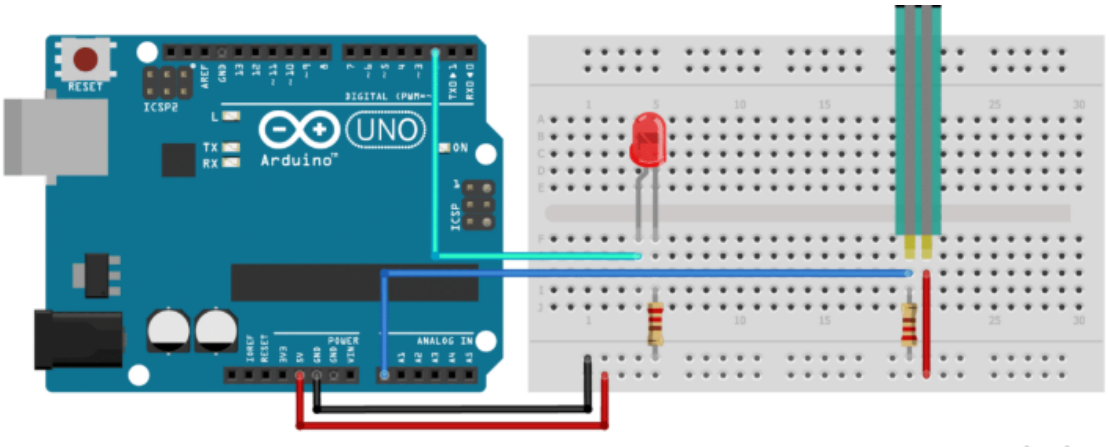


Figure 3.4: Basic setting for pressure capture

- Wires, breadboard and resistors : for connection and well functioning of sensors.

There are many kinds of force-measuring resistors available in the market. Since we are to calculate the pressure applied under each key when pressed, We need a long sensor strip which is to be placed under the keys of the same row inside the keyboard. Such a sensor is FSR Interlink-408 as shown in Fig 3.5. Using this sensor we can calibrate pressure along the length of the strip. It can detect pressure throughout its length. Now 5 of such pressure sensors are placed under the keyboard right below the keys as shown in the figure below. The keyboard used is Logitech K120 as the interior design of this keyboard is well suited for our sensor (It's edges prevent exerting unnecessary pressure on our sensors). The conducting ends of these sensors are connected to the arduino using breadboard and copper wires. The Fig 3.6 shows the setup clearly. This completes the construction of a pressure sensitive keyboard. In order to obtain the pressure characteristics we are to connect this arduino with the computer. Pressure values can be obtained using an Arduino IDE application through the port at which our arduino is connected. The code written on Arduino IDE doesn't do any calculations, it just prints out what it interprets as the amount of pressure in a qualitative manner. Since we have taken FSR Interlink-408, whenever there is pressure applied on any part of the strip, The circuit generates analog signals which depends on the pressure being applied on the strip. The Arduino device then converts these analog signals to digital readings and supply them to the computer along the port at which the device was connected to the computer. This way we can get our pressure readings.

The below snippet is to be uploaded on the arduino device through the Arduino IDE to obtain the amount of pressure being applied on the sensor strip.The output is then displayed on the separate window for every 300 ms (as mentioned as delay parameter in below snippet).

```
int pressureAnalogPin = 0; //pin where our pressure pad is located.
int pressureReading; //variable for storing our reading

//Adjust these if required.
```

Figure 3.5: FSR Interlink-408 (Pressure sensor)

Figure 3.6: Hardware setup for pressure caputre

```
int noPressure = 5; //max value for no pressure on the pad
int lightPressure = 400; //max value for light pressure on the pad
int mediumPressure = 800; //max value for medium pressure on the pad
int incomingByte = 0;

void setup(void) {
    Serial.begin(9600);
    }

void loop(void) {
    pressureReading = analogRead(pressureAnalogPin);
    incomingByte = Serial.read();

    Serial.println(pressureReading);

    delay(300);
    }
```

Now that the pressure is being displayed on the arduino window which is being received from the connected port. We have to record this pressure information. This recording of pressure values from the arduino connected port can be done using another python library. Connect the arduino usb and enable the port with UNO input in order to obtain the digital signals (i.e., pressure readings). Fig 3.7 explains this. An arduino is a device used to convert the analog signals to digital signals. When our pressure sensors are placed under the keyboard, the pressure signals (analog signals) are sensed by the arduino and it converts them to digital signals and displays it on the command window of arduino IDE according to the code uploaded on it as shown in Fig 3.8. In Fig 3.8, the values of pressure were printed on the window. Now, we are to record the pressure readings and store them in a dataset. There is a python library "Serial" through which we can get this work done. Through this library we can listen to the desired port and store the information coming through that port. In

the below snippet 'COM8' port (it may also be different) is used by the arduino. The data is recorded and saved in `raw_pressure_data.csv`.

```
import serial
import time
import csv

ser = serial.Serial(port='COM8')
ser.flushInput()

while(True):
        ser_bytes = ser.readline()
        print(ser_bytes)
        decoded_bytes = ser_bytes[len(ser_bytes)-5:len(ser_bytes)-2].decode('
        print(decoded_bytes)
        with open("raw_pressure_data.csv","a") as f:
            writer = csv.writer(f,delimiter=",")
            writer.writerow([decoded_bytes])
```

Since there is no use in storing 0s in the csv file we are to clean it. Also the arduino displays the pressure information for every certain interval (i.e., Delay) as mentioned above. There will be a situation where a person may press the key for a bit longer. In that case we may obtain more than one pressure reading for that key. To avoid such cases, we calculate the average of non-zero adjacent pressure readings and store them in such a way that total pressure readings obtained are the same as that of total characters present in the password. In the below snippet variable 'n' keeps count of non-zero adjacent pressure readings. Through this below snippet, we are obtaining the final pressure characteristics.

```
for i in raw_pressure_data:
```

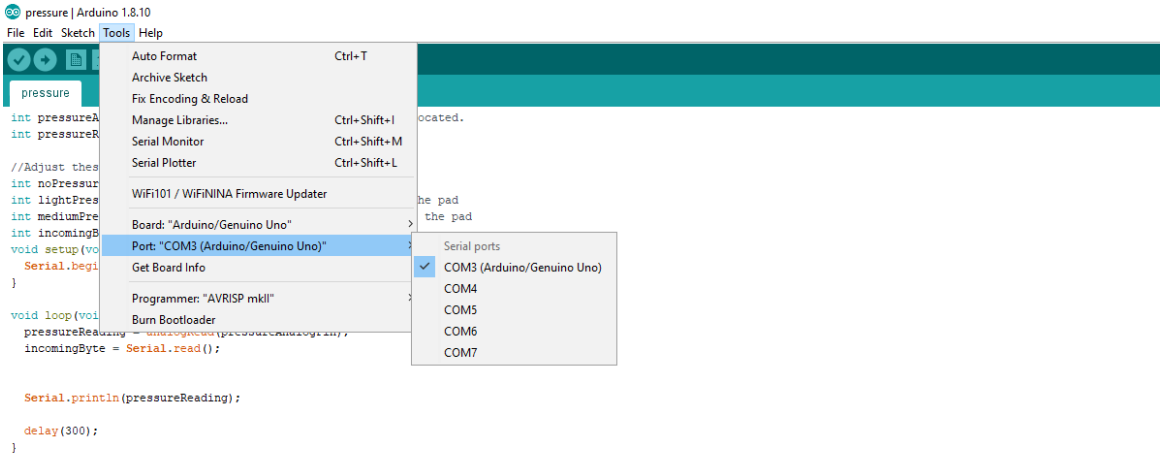


Figure 3.7: Setting the arduino port to recieve pressure data

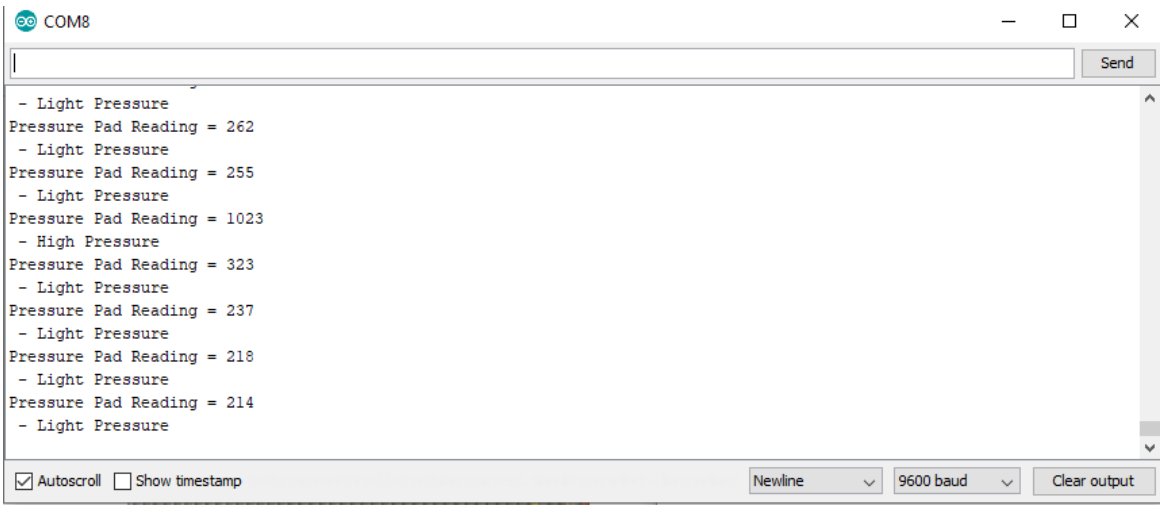


Figure 3.8: obtaining pressure data through the port

```
        if(i!=0):
            n = n + 1
            count = count + i
            zcount = 0
        else:
            if(zcount <= 1):
                zcount = zcount + 1
            if(count!=0 and n!=0 and zcount>1):
                arr2.append(count/n)
                n=0
                count=0
                zcount=0
    if(count!=0 and n!=0):
        arr2.append(count/n)
        n=0
        count=0
    print(arr2)
    with open('pressure2.csv','a') as d:
        writer = csv.writer(d,delimiter=",")
        writer.writerow(arr2)
```

So, by typing the password using this pressure sensitive keyboard both the timing and pressure characteristics can be captured simultaneously. Both genuine and fraud datasets of the different subjects containing timing and pressure parameters can be obtained through these codes and hardware implemented.

3.1.3 Extraction of location Parameters

Since our effort is to distinguish between the genuine and the fraud user through these behavioural biometrics, A fraud user could access this system from different places.If he could somehow match the typing sequence then he may not be identified as a fraud.

So through our approach we could be able to add another secure feature i.e., the location parameter. We further use the recent time of authentication and recent location of authentication through which we can identify if the authentication is by fraud or not, by defining the relation between the time difference of successive authentications and distance between successive locations where this system is accessed.

So, This location parameter is the new parameter being added to increase the performance of fraud detection.The Time Location parameters are as follows:

- Time difference between present and previous authentications.
- Distance between the present and previous locations of authentication.

Time differences between successive authentications are also recorded per every row and also the distance between the successive locations are also noted and they are differentiated as genuine and fraud by defining the relation between those two parameters through GoogleMaps API. This API tells us the average time taken to travel between the given locations(i.e., latitudes and longitudes of 2 locations given). The below snippet helps us to obtain distance parameter using a key which has to be purchased online provided by Google.

```
import googlemaps
gmaps = googlemaps.Client(key='*****************************')
# Requires cities name
my_dist = gmaps.distance_matrix('Delhi','Mumbai')['rows'][0]['elements'][0]
# Printing the result
print(my_dist)
```

If the time difference between successive locations is lesser than the average time given by API,we can get to know he is a fraud. Since this GoogleMapsAPI shows some payment errors,for the time being we have filled the dataset manually with some genuine and fraud instances with respect to the time difference between successive logins and distance between the locations of both logins. Since these applications can be used in many authentication systems in order to give extra support to the

authentication even the password is compromised. For example, If this system is being used in an ATM machine and a fraud user managed to match your keystroke bio metrics i.e., timing and pressure characteristics, Then these parameters helps in identifying fraud user in some cases based on his history i.e., previous time and location of authentication.The features that are obtained in this stage are as displayed in Fig 3.9.

By the end of this step, we have obtained all the timing, pressure and required location parameters. So, the next step is to use this data for the Fraud detection.

3.2 Data Analysis

3.2.1 Comparison of Timing Parameters between genuine and fraud users

In this section, Analysis is to be done in order to observe whether there is any difference between the timing patterns. After the dataset is prepared, we have analysed the dataset by visualizing the various parameters for 5 rows of each genuine and fraud sets of data. We can clearly observe the difference between the typing patterns of genuine and fraud through this plotting.As the fraud users take different times to figure out the next key.There are cases when genuine users take more time whereas fraud users take lesser time for typing the whole passkey. Also there are cases when the genuine users type their passkey faster than that of passkey. If both the users take same time then pressure parameters helps us in determining the fraud.The plotting of single genuine user vs single fraud user is shown in Fig 3.10.

Whereas the plotting of the various time characteristics of few genuine users vs few fraud users are shown in Fig 3.11. The plotting is done in order to understand how the data of the genuine and fraud users is distributed with respect to others. If presented in visual form, information is often much easier to digest, especially if it makes use of patterns and structures that humans can interpret intuitively.

From the visualization we can distinguish that the behavioural bio metrics can be used to differentiate the users. This can be a motivation to proceed further in this project. Now that we have completed the construction of the pressure sensitive keyboard, we can also analyse further whether or not we can differentiate the fraud

Time_from_lastchec	Distance_from_lastc	Result	Place	Last_Checkin_time	Current_checkin_tim	Previous_latitude	Previous_longitude	Previous_latitude	Present_longitude
1	0	1	mangalore	10/10/2019 10:20	10/11/2019 10:20	12.914142	74.855957	12.914142	74.855957
1	358	1	mangalore	10/11/2019 10:20	10/12/2019 10:20	12.914142	74.855957	12.971599	77.594566
1.000000116	813	1	mangalore	10/12/2019 10:20	10/13/2019 10:20	12.914142	74.855957	17.385044	78.486671
1.000000174	13462	1	mangalore	10/13/2019 10:20	10/14/2019 10:20	12.914142	74.855957	42.331429	-83.045753
1.000000231	14282	1	mangalore	10/14/2019 10:20	10/15/2019 10:20	12.914142	74.855957	36.778259	-119.417931
1.000000289	2190	1	mangalore	10/15/2019 10:20	10/16/2019 10:20	12.914142	74.855957	28.70406	77.102493
1.000000347	704	1	mangalore	10/16/2019 10:20	10/17/2019 10:20	12.914142	74.855957	13.08268	80.270721
1.000000405	892	1	mangalore	10/17/2019 10:20	10/18/2019 10:20	12.914142	74.855957	19.075983	72.877655
1.000000463	2190	1	mangalore	10/18/2019 10:20	10/19/2019 10:20	12.971599	77.594566	34.083672	74.797279
1.000000521	9578	1	mangalore	10/19/2019 10:20	10/20/2019 10:20	12.971599	77.594566	-33.86882	151.20929
1.000000579	358	1	Bangalore	10/20/2019 10:20	10/21/2019 10:20	12.971599	77.594566	12.914142	74.855957
1.000000637	0	1	Bangalore	10/21/2019 10:20	10/22/2019 10:20	12.971599	77.594566	12.971599	77.594566
1.000000694	569	1	Bangalore	10/22/2019 10:20	10/23/2019 10:20	12.971599	77.594566	17.385044	78.486671
1.000000752	13457	1	Bangalore	10/23/2019 10:20	10/24/2019 10:20	12.971599	77.594566	42.331429	-83.045753
1.00000081	14194	1	Bangalore	10/24/2019 10:20	10/25/2019 10:20	12.971599	77.594566	36.778259	-119.417931
1.000000868	2197	1	Bangalore	10/25/2019 10:20	10/26/2019 10:20	12.971599	77.594566	28.70406	77.102493
1.000000926	347	1	Bangalore	10/26/2019 10:20	10/27/2019 10:20	12.971599	77.594566	13.08268	80.270721
1.000000984	985	1	Bangalore	10/27/2019 10:20	10/28/2019 10:20	12.971599	77.594566	19.075983	72.877655
1.000001042	2987	1	Bangalore	10/28/2019 10:20	10/29/2019 10:20	12.971599	77.594566	34.083672	74.797279
1.0000011	9343	1	Bangalore	10/29/2019 10:20	10/30/2019 10:20	12.971599	77.594566	-33.86882	151.20929
1.000001157	813	1	Hyderabad	10/30/2019 10:20	10/31/2019 10:20	17.385044	78.486671	12.914142	74.855957
1.000001215	569	1	Hyderabad	10/31/2019 10:20	11/1/2019 10:20	17.385044	78.486671	12.971599	77.594566
1.000001273	0	1	Hyderabad	11/1/2019 10:20	11/2/2019 10:20	17.385044	78.486671	17.385044	78.486671
1.000001331	13102	1	Hyderabad	11/2/2019 10:20	11/3/2019 10:20	17.385044	78.486671	42.331429	-83.045753
1.000001389	13679	1	Hyderabad	11/3/2019 10:20	11/4/2019 10:20	17.385044	78.486671	36.778259	-119.417931
1.000001447	1578	1	Hyderabad	11/4/2019 10:20	11/5/2019 10:20	17.385044	78.486671	28.70406	77.102493

Figure 3.9: Location and distance features

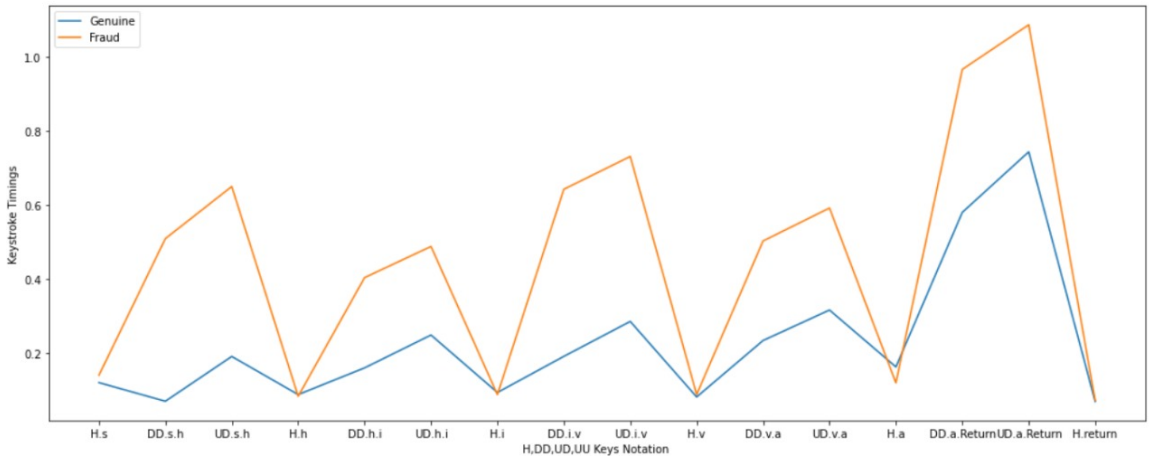


Figure 3.10: Plotting single genuine vs single fraud user timing data

or genuine using their pressure information.

3.2.2 Comparison of Pressure, additional Parameters between genuine and fraud users

In the below graph pressure recordings are along y axis and 10 repetitions along x axis. For each repetition the pressure recordings are compared between Genuine(Red) and Fraud(green) users.Fig.3.12 is the plotted distribution of pressure characteristics of 5 repetitions from each genuine and fraud category. As we can see we can not observe any pattern in order to differentiate one from another.

In the Fig 3.13 graph pressure recordings are along y axis and 50 repetitions along x axis.we are able to observe a rough pattern i.e., we were able to identify a range in which most of the genuine users fall into with our naked eye for some extent.

In Fig.3.14 we mapped the user's time from last check-in on x-axis and distance between the present and last check-in location on y axis. Then we have marked the genuine user's observations as green and fraud's as red. From the figure we are roughly able to distinguish the genuine and fraud i.e., we are able to find a boundary between those 2 types.In order to make it even clearer we have to use different machine learning models. These machine learning models are capable of dealing with complex data distributions.

So, In the next step we use various machine learning algorithms with different parameters and choose the best model for our fraud detection.

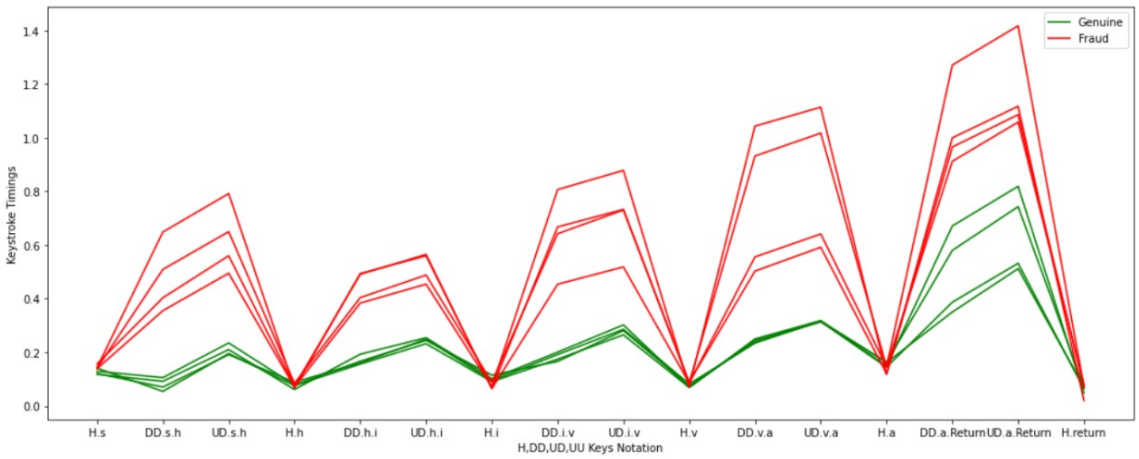


Figure 3.11: Distribution of few genuine vs fraud users timing data

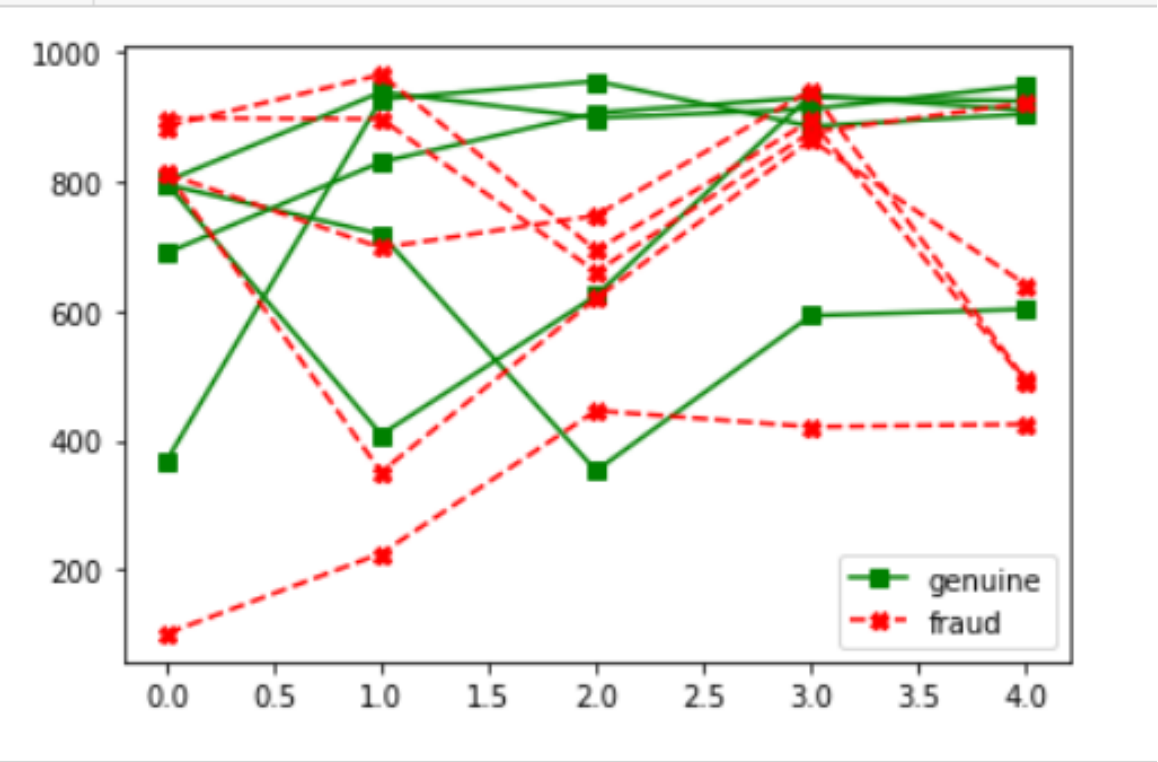


Figure 3.12: Distribution of few genuine vs fraud users pressure data

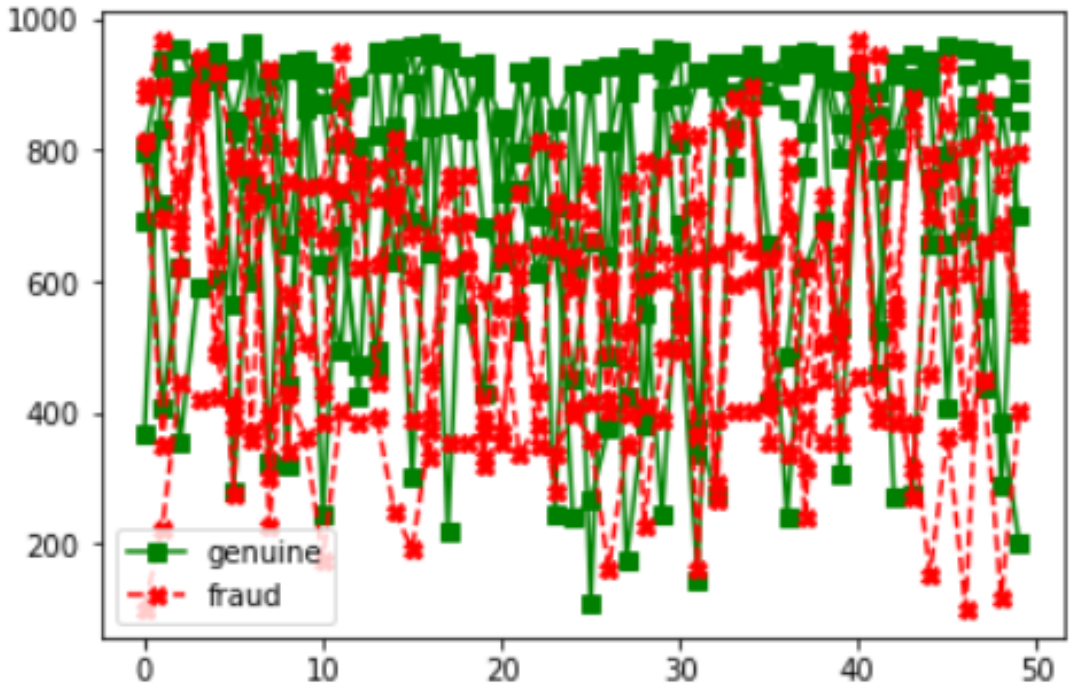


Figure 3.13: Distribution of 50 repetitions of genuine vs fraud users pressure data

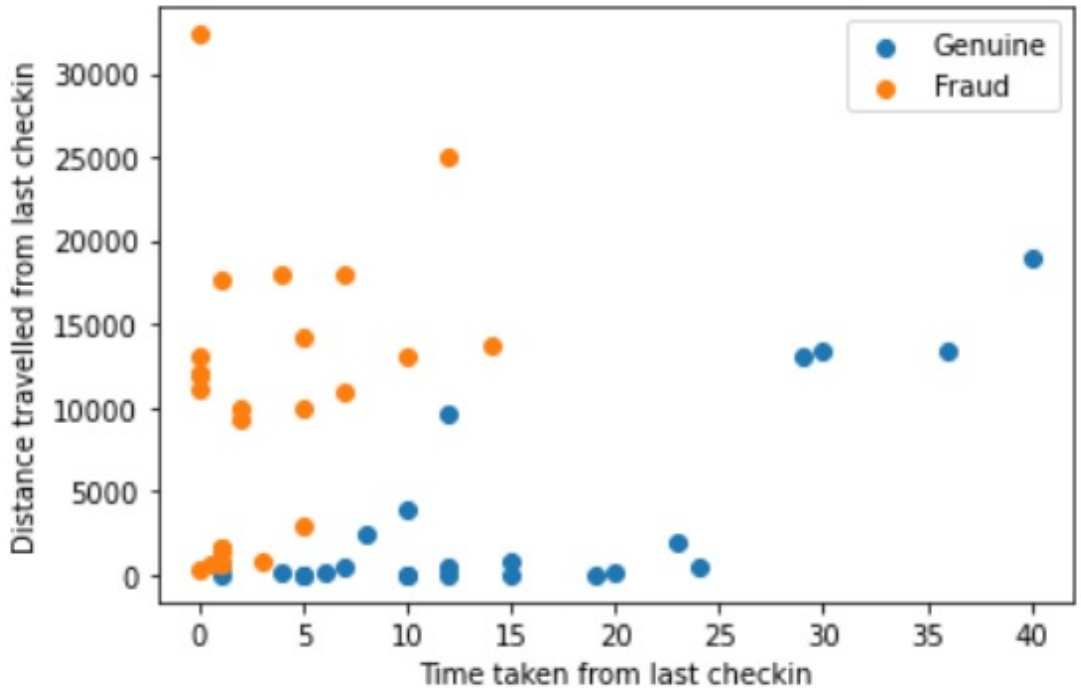


Figure 3.14: Distribution of few repetitions of genuine vs fraud users location and time data

Chapter 4

Fraud Detection

The dataset with time characteristics and location details are to be fed into a model. This step involves the following processes:

- The dataset of genuine users are labelled 1 and frauds with 0 and It was shuffled.
- Variable selection
- Normalization of data
- Splitting into 90% train data and 10% test data.
- Model fitting comparision.

4.1 Preprocessing

The first 4 steps mentioned above comes under Preprocessing.

4.1.1 Variable selection

A central problem in machine learning is identifying a representative set of features from which to construct a classification model for a particular task. Correlation is a statistical term which in common usage refers to how close two variables are to having a linear relationship with each other. Features with high correlation are more linearly dependent and hence have almost the same effect on the dependent variable. So, when two features have high correlation, we can drop one of the two features.
In our dataset as correlation is less between the variables all parameters are considered for the model training.

4.1.2 Min-Max Normalization

We used this method where the parameters are rescaled to the range of the data to [0,1]

$$\hat{X}[:,i] = \frac{X[:,i] - \min(X[:,i])}{\max(X[:,i]) - \min(X[:,i])}$$

For every feature, the minimum value of that feature gets transformed into a 0, the maximum value gets transformed into a 1, and every other value gets transformed into a decimal between 0 and 1.
The dataset is split into 90% train data and 10% test data.

4.2 Model fitting

We have used different machine learning classification algorithms like logistic regression,knn-classification and Support Vector Machines. When used only timing characteristics to distinguish between genuine and fraud users, SVM performed well among the above algorithms.The 80% data is used to train the model and the remaining for validation purposes which resulted in the accuracy of nearly 88%. Then, after the implementation of the pressure sensitive keyboard, we have collected all the timing, location and pressure characteristics. Also there is a problem of using such small datasets. Some times as we are using a small dataset, This model may become overfitted and may classify the genuine user as a fraud. To overcome this we can train the model dynamically.
Using the above collected characteristics, We have used different machine learning models for classification of our data on Genuine and Fraud Users. The below mentioned are those algorithms on which we have worked:

- Logistic Regression
- Bayesian Classification
- SVM (support vector machine)
- KNN (K Nearest Neighbours)

- MLP (Multi Layer Perceptron)
- RNN (LSTM)

In the following section we would be explaining why we have used a particular model for our dataset and the performance comparison of the model on keystroke data and Keystroke data with pressure recordings.

4.2.1 Logistic Regression

Logistic regression is an extension of simple linear regression.Where the dependent variable is binary in nature, we cannot use simple linear regression. Logistic regression is the statistical technique used to predict the relationship between predictors (our independent variables) and a predicted variable (the dependent variable) where the dependent variable is binary.
Binary Logistic Regression is a type of logistic Regression where The categorical response has only two 2 possible outcomes for example in this case genuine or not genuine(Fraud). As our dataset has statistical values and the result is binary this model helps in classification of our dataset used.
Dataset has following following categories of parameters

- Keystroke timings of user(UD,DD,DU,UU timings of each key)
- Distance and time from previous checking.
- Pressure recordings of each key of Genuine and Fraud users.

Classification without pressure data

```
logistic regression model accuracy: 0.9333333333333333
```

As divergent individuals differ in pressurising the keys of the keyboard, pressure recordings help in improving the classification accuracy of a model. With the concatenation of the data with pressure recordings of genuine and fraud users the accuracy of the model has increased.

Classification with pressure and additional data

```
logistic regression model accuracy: 0.9821428571428571
```

Accuracy has increased from 93 to 98 with inclusion of pressure data. Thus Logistic regression is used to describe data and to explain the relationship between one dependent binary variable and one or more statistical independent variables.

4.2.2 Bayesian Classification

Bayesian classification is based on Bayes' Theorem. Bayesian classifiers are the statistical classifiers. Naive Bayes (classifier) is a type of generative model that models each possible category based on training samples. We call it "Naive Bayes" because of the assumption that each attribute has conditional independence. This assumes that each attribute has an independent effect on the eventual classification result.
The following formula is the working rule of this classifier.

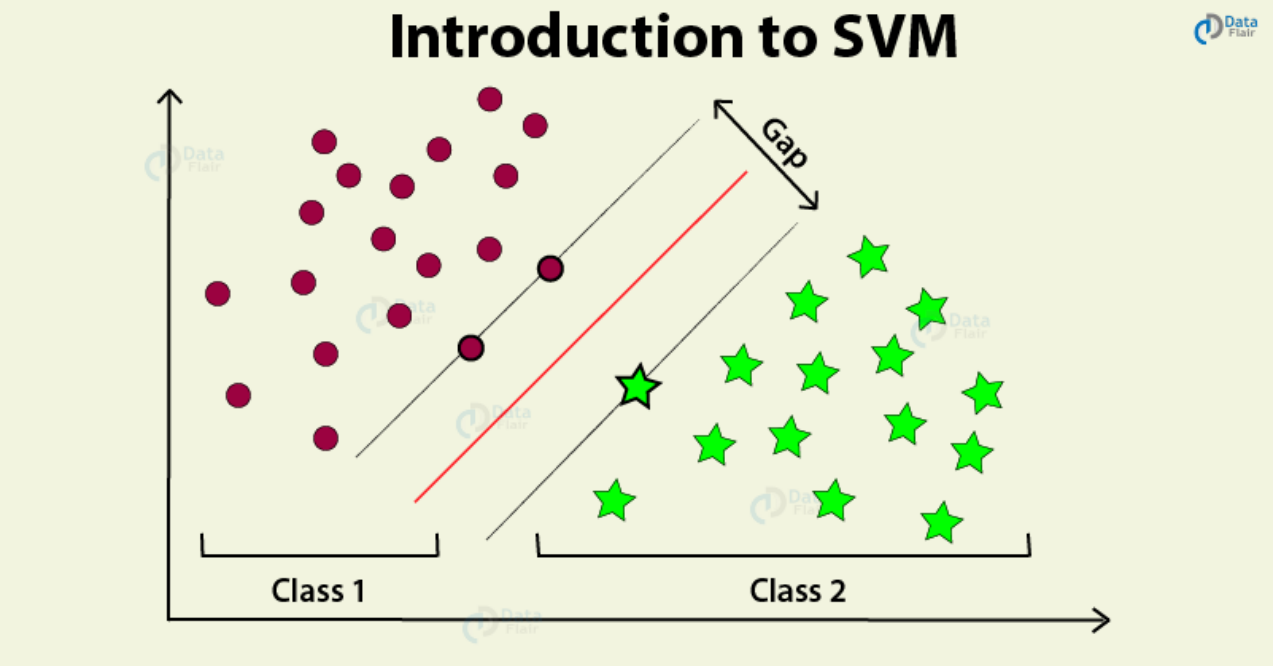
$$P(c|x) = \frac{P(c)P(x|c)}{P(x)} = \frac{P(c)}{P(x)} \prod_{i=1}^{d} P(x_i|c)$$

The main task is to determine the characteristics of the attributes according to the attributes' specific conditions and to perform the appropriate division of each feature attribute. Then, we select a portion of the classification for sampling and use it to form the training sample set. The input for the earlier stage of this model is all the classifiable data, and the output is the characteristic attributes and training samples. As the work we referred to, they have used the Bayesian classification as one of their models. We have used this same approach on our dataset and obtained the result as follows

Bayesian Classification model Accuracy obtained : 0.8714285714285714

4.2.3 Support Vector Machine (SVM):

SVM is a supervised machine learning algorithm which can be used for classification or regression problems. SVM is capable of doing both classification and regression.We have used SVM for classification. Non-linear SVM means that the boundary that the algorithm calculates doesn't have to be a straight line. The benefit is that you can capture much more complex relationships between your data points without having to perform difficult transformations on your own. The downside is that the training time is much longer as it's much more computationally intensive.
SVM is well suited for classification of complex but small or medium sized datasets. Our dataset is not substantial and is complex with 30 to 35 parameters so SVM would be a suitable model for classification of our dataset.



Classification without pressure data

```
Accuracy: 0.86 (+/- 0.04)
```

As divergent individuals differ in pressurising the keys of the keyboard, pressure recordings help in improving the classification accuracy of a model. With the concatenation of the data with pressure recordings of genuine and fraud users the accuracy of the model has increased.

Classification with pressure and additional data

```
svm model accuracy: 0.9
```

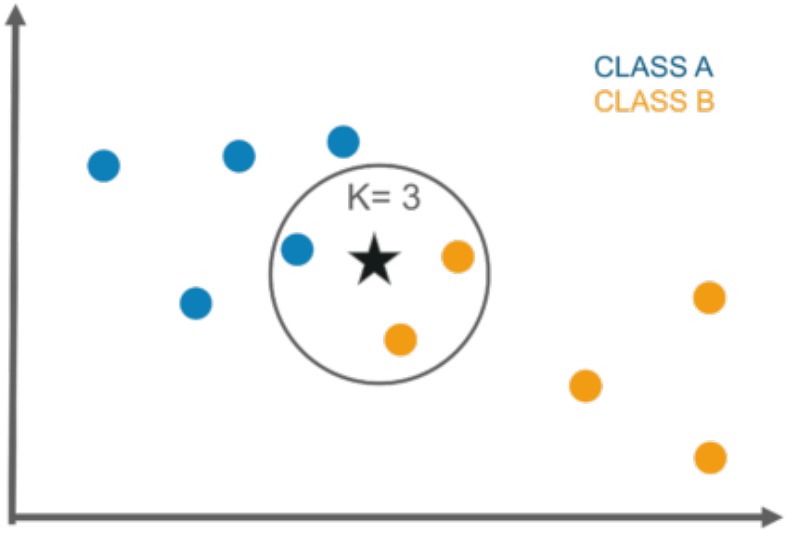
Accuracy has increased from 86 to 90 with inclusion of pressure data.

4.2.4 K Nearest Neighbours (KNN):

The KNN algorithm can compete with the most accurate models because it makes highly accurate predictions. Therefore, you can use the KNN algorithm for applications that require high accuracy but that do not require a human-readable model. The quality of the predictions depends on the distance measure.KNN tends to perform well when you have many instances (points) and few dimensions.

KNN-classifier can be used when your data set is small enough, so that KNN-Classifier completes running in a shorter time. In general, the complexity of KNN Classifier in Big Oh notation is where n is the number of data points. KNN works by finding the distances between a query and all the examples in the data, selecting the specified number examples (K) closest to the query, then votes for the most frequent label (in the case of classification) or averages the labels (in the case of regression).

We have tried for different K values with different metrics like euclidean distance, manhattan distance and etc.. Of these metrics , we have obtained better results with K as 3 for our KNN model.



Classification without pressure data

```
kNN model accuracy: 0.9
```

As different individuals differ in pressurising the keys of the keyboard, pressure recordings help in improving the classification accuracy of a model. With the concatenation of this data with pressure recordings of genuine and fraud users the accuracy of the model has increased.

Classification with pressure and additional data

```
kNN model accuracy: 0.9892857142857143
```

As observed in different algorithms, Accuracy has increased from 90 to 98.9 with inclusion of pressure data in this algorithm too.

4.2.5 Multi Layer Perceptron (MLP):

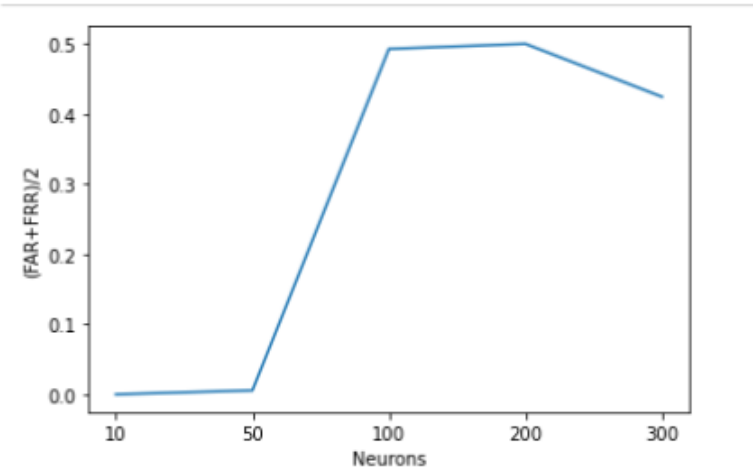
As the work we referred to, they created their dataset by building their own apparatus and used the Artificial neural networks as one of their models and experimented with different numbers of neurons, different number of input variables. So we have used our

data with the same approach as theirs. Our data is different from theirs with some additional features. So, Through this model we can have a chance to check whether these additional features help us by increasing the accuracy in fraud detection or decrease it's capacity by making the system even more overfitted.

The state standard GOST R 52633.5-2011 recommends using a single-layer or two-layer artificial neural network for biometric authentication. The first layer enhances data, the second layer acts as an error-correcting code. In our work we use a single-layer neural network. As the reference work had done the same.

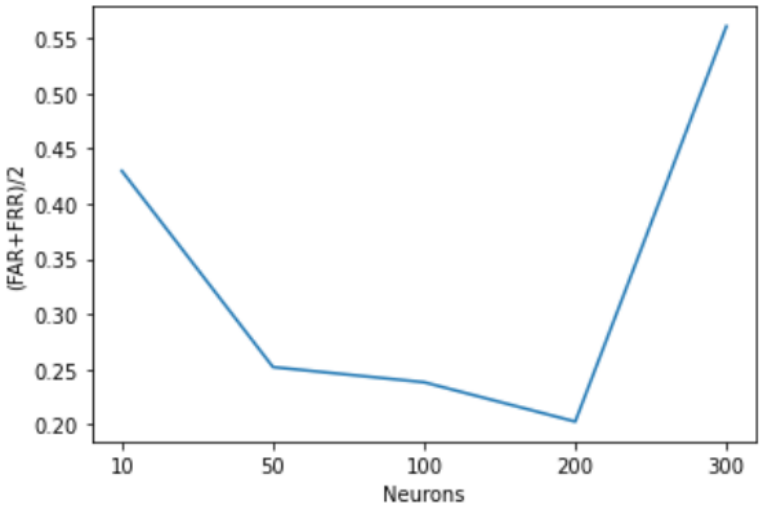
A number of neurons and their inputs in this work is a parameter that varied in the process of the computational experiment. The results of the experiment are shown in the below Figures. Error probabilities were computed as a ratio of a number of false code outputs to a number of experiments carried out. The average of false acceptance and false rejection error rates and the number of neurons are being mapped in the below figure.

Classification with Timing characteristics and location characteristics as input:



We have tried with different numbers of neurons in the hidden layer and obtained the above graph. From the graph, we have obtained that the model performed well when there are 50 neurons in the hidden layer.

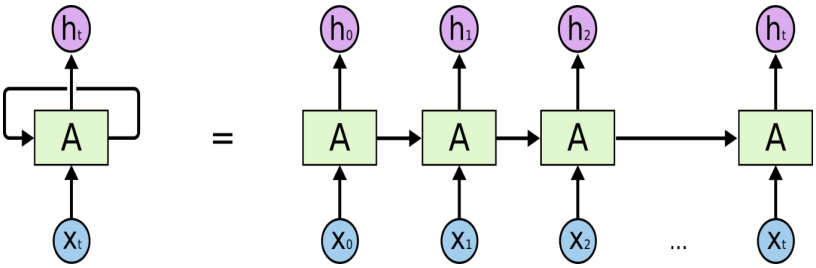
Classification with pressure and location characteristics as input



When we used the pressure characteristics with location characteristics, we have obtained the average error rates as shown above. By considering both the graphs we obtained better results when the number of neurons in the hidden layer is 50.

4.2.6 Long Short term memory (RNN):

We created a dataset of genuine users in which there are a total of 1000 entries labelled as 1. We also created a dataset of fraud users which consists of 500 entries and label as 0. We shuffle both datasets and the new dataset is our final one on which we apply LSTM. Firstly, we take the pressure and time parameters for the name "shiva" which consists of 23 columns i.e., 23 features of the given password.Then after data preprocessing (shuffling and labelling the data) we split the data to 90% train and 10% test and train the data in batch of 3 and add layers of LSTM and dense layers and Softmax function is used as activation function along with adam optimizer and mean squared error as our loss parameter as it is accurateHere there are 23 features and LSTM models take those features into its network and as per the given data, we can set the number of time steps in the hidden layer to 1 .



Train data is reshaped into 3D inorder to input into the Long Short Term Memory model.The model consists of LSTM layers and Dense layers, LSTM layer consists of 80 neurons whereas a dense layer has input data shape.Model trains the data in batch size of 3 of 20 epochs where trainloss,train error,validation loss, validation error are the parameters obtained at each epoch. **Model Summary**

```
model.summary()
```

```
Model: "sequential_26"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_27 (LSTM)               (None, 80)                32640
_________________________________________________________________
dense_49 (Dense)             (None, 10)                810
_________________________________________________________________
dense_50 (Dense)             (None, 1)                 11
=================================================================
Total params: 33,461
Trainable params: 33,461
Non-trainable params: 0
_________________________________________________________________
```

Classification without pressure data

```
model.fit(x_train, y_train, epochs=20, validation_data=(x_test,y_test), batch_size=3)

Train on 1260 samples, validate on 140 samples
Epoch 1/20
1260/1260 [==============================] - 2s 2ms/step - loss: 0.2640 - accuracy: 0.5937 - val_loss: 0.1526 - val_accurac
y: 0.7714
Epoch 2/20
1260/1260 [==============================] - 1s 1ms/step - loss: 0.1607 - accuracy: 0.7389 - val_loss: 0.1247 - val_accurac
y: 0.7714
Epoch 3/20
1260/1260 [==============================] - 1s 1ms/step - loss: 0.1351 - accuracy: 0.8317 - val_loss: 0.1044 - val_accurac
y: 0.8714
Epoch 4/20
1260/1260 [==============================] - 1s 1ms/step - loss: 0.1240 - accuracy: 0.8524 - val_loss: 0.0911 - val_accurac
y: 0.8857
Epoch 5/20
1260/1260 [==============================] - 1s 1ms/step - loss: 0.1109 - accuracy: 0.8611 - val_loss: 0.0872 - val_accurac
y: 0.8929
Epoch 6/20
1260/1260 [==============================] - 1s 981us/step - loss: 0.1053 - accuracy: 0.8635 - val_loss: 0.0806 - val_accura
cy: 0.8929
```

An average of 75% is the accuracy when we used just the timing characteristics alone without pressure recordings. As divergent individuals differ in pressurising the keys of the keyboard, pressure recordings help in improving the classification accuracy of a model. With the concatenation of the data with pressure recordings of genuine and fraud users the accuracy of the model has increased.

Classification with pressure and additional data

```
Epoch 6/10
270/270 [==============================] - 0s 218us/step - loss: 5.5492 - dense_5_loss: 0.6631 - dense_5_accuracy: 0.559
dense_5_accuracy_1: 0.6037 - dense_5_accuracy_2: 0.7333 - dense_5_accuracy_3: 0.9222 - dense_5_accuracy_4: 0.9630 - dens
accuracy_5: 0.5815 - dense_5_accuracy_6: 0.6630 - dense_5_accuracy_7: 0.6222 - dense_5_accuracy_8: 0.5741
Epoch 7/10
270/270 [==============================] - 0s 207us/step - loss: 4.8348 - dense_5_loss: 0.6035 - dense_5_accuracy: 0.577
dense_5_accuracy_1: 0.6407 - dense_5_accuracy_2: 0.7815 - dense_5_accuracy_3: 0.8407 - dense_5_accuracy_4: 0.9519 - dens
accuracy_5: 0.8593 - dense_5_accuracy_6: 0.8185 - dense_5_accuracy_7: 0.6815 - dense_5_accuracy_8: 0.6593
Epoch 8/10
270/270 [==============================] - 0s 207us/step - loss: 4.1479 - dense_5_loss: 0.5681 - dense_5_accuracy: 0.611
dense_5_accuracy_1: 0.7556 - dense_5_accuracy_2: 0.9037 - dense_5_accuracy_3: 0.9444 - dense_5_accuracy_4: 0.9519 - dens
accuracy_5: 0.9222 - dense_5_accuracy_6: 0.8296 - dense_5_accuracy_7: 0.7926 - dense_5_accuracy_8: 0.7333
Epoch 9/10
270/270 [==============================] - 0s 211us/step - loss: 3.4126 - dense_5_loss: 0.4738 - dense_5_accuracy: 0.670
dense_5_accuracy_1: 0.8222 - dense_5_accuracy_2: 0.9630 - dense_5_accuracy_3: 0.9704 - dense_5_accuracy_4: 0.9667 - dens
accuracy_5: 0.9593 - dense_5_accuracy_6: 0.9111 - dense_5_accuracy_7: 0.8852 - dense_5_accuracy_8: 0.7815
Epoch 10/10
```

As we can observe in the below image, the accuracy got increased as the epochs increased i.e, as the training proceeds. Average of 85% accuracy is recorded with pressure parameters.

Chapter 5

Results

We are done with data collection and modelling by now. In this section we present our results obtained on different algorithms and decide which model has outperformed the others in fraud detection. For this, We have taken 20 different subjects and created datasets containing 1000 genuine repetitions and 600 fraud repetitions of each password and applied the above mentioned algorithms on these datasets. We have used a few metrics to understand about the models.

5.1 Metrics of comparison

5.1.1 ROC Curve

AUC - ROC curve is a performance measurement for classification problems at various thresholds settings. ROC is a probability curve and AUC represents degree or measure of separability. It tells how much a model is capable of distinguishing between classes. Higher the AUC (Area Under Curve), better the model is at predicting 0s as 0s and 1s as 1s.

The ROC curve is plotted with TPR against the FPR where TPR is on the y-axis and FPR is on the x-axis. Where TPR, FPR are as follows:

TPR = Sensitivity = $\frac{TP}{\mathrm{TP + FN}}$

FPR = 1 - Specificity = $\frac{FP}{\mathrm{TN + FP}}$

The AUC - ROC curves are represented in the figures.

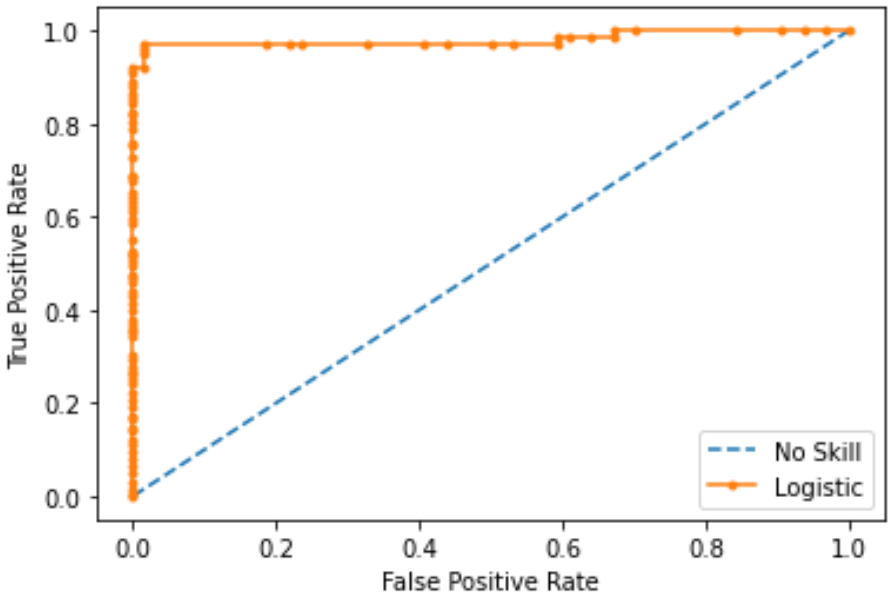


Figure 5.1: AUC - ROC curve for Logistic Regression

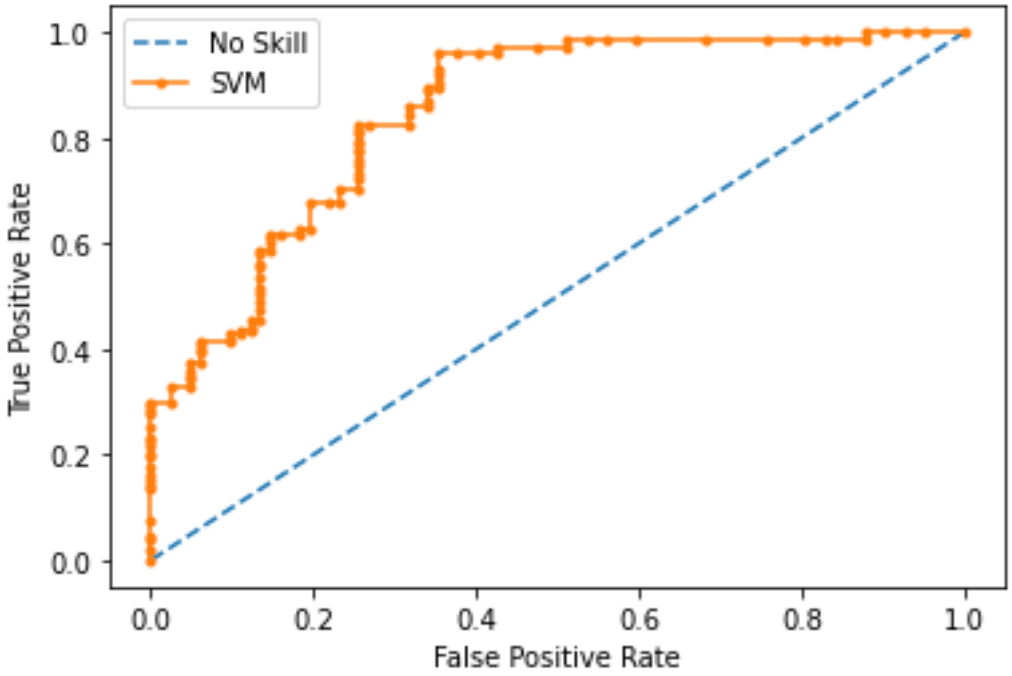


Figure 5.2: AUC - ROC curve for SVM classification

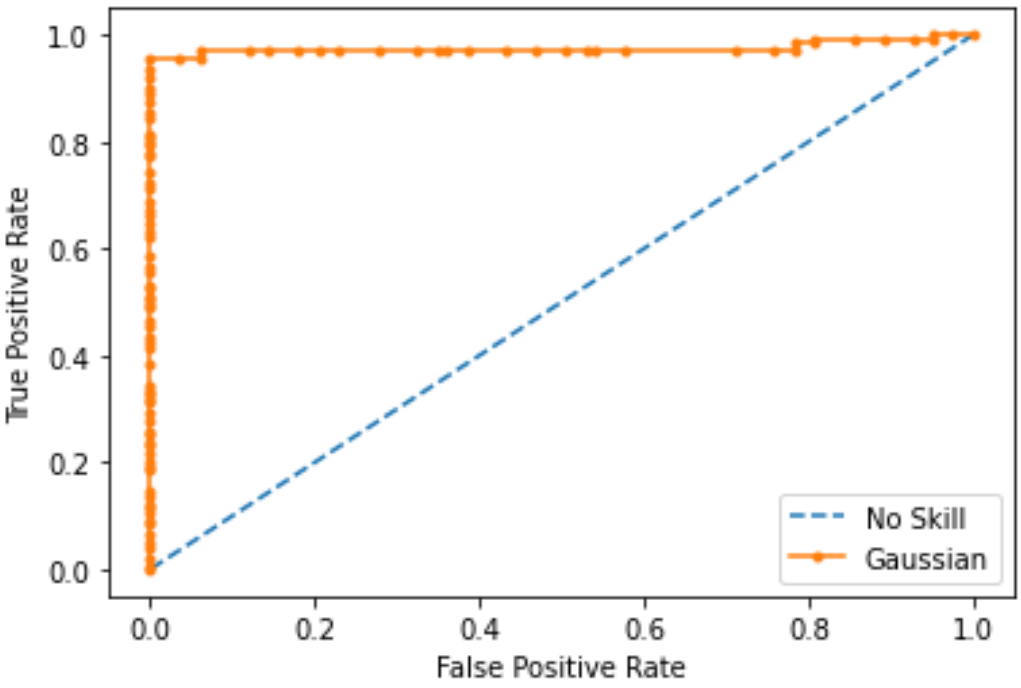


Figure 5.3: AUC - ROC curve for Bayesian Classification

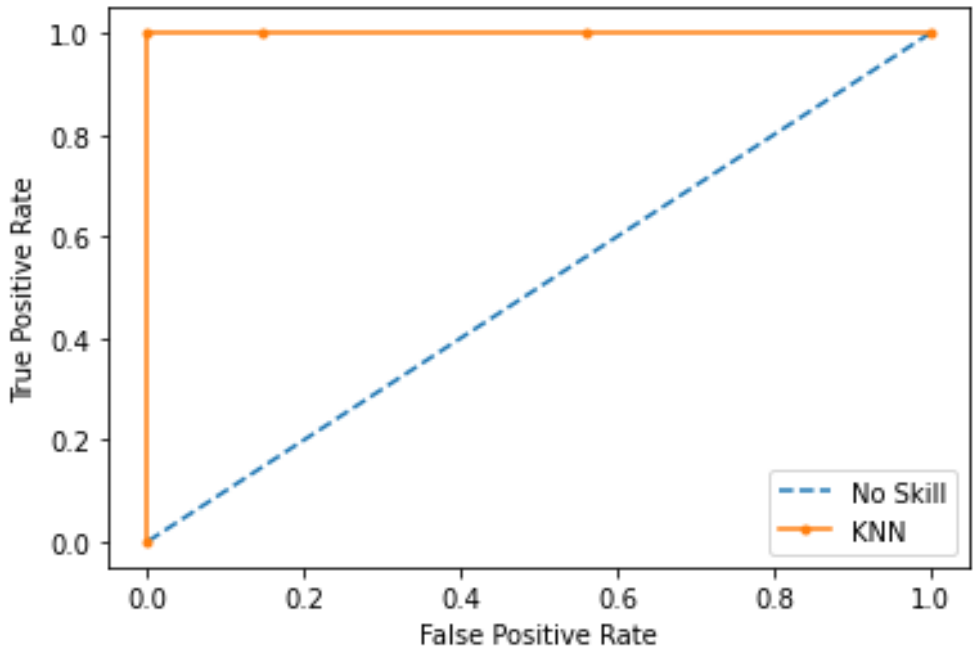


Figure 5.4: AUC - ROC curve for Knn classification

The AUC - ROC curves of logistic regression, SVM classification, Bayesian Classification and Knn classification are shown in Fig.5.1,Fig 5.2,Fig.5.3,Fig.5.4 respectively. The areas under curves are shown below in the table.

Classification Algorithm	**Area Under Curve**
Logistic Regression	0.982
SVM classification	0.848
Bayesian Classification	0.973
Knn classification	1

By Observing the above ROC- AUC(Area Under Curve) of all the algorithms it is evident that Knn classification has shown the better value of AUC among other models. As we know that higher the AUC value, Better the model. So, This metric suggests Knn classifier as the best model.

5.1.2 Accuracy

We have then taken the average of the accuracies of different user datasets for each algorithm and plotted then down in Fig.5.5.

From Fig.5.5, It is evident that inclusion of pressure and location characteristics have increased the accuracies.i.e., resulted in better fraud detection than the system with timings information as features alone. Their accuracy values are as follows.

Classification Algorithm	**Accuracy with our data**
Logistic Regression	0.982
SVM classification	0.901
Bayesian Classification	0.871
Knn classification	0.989

Since different data distributions do well on different machine learning algorithms. In these distributions, Knn algorithms averaged really high with the inclusion of pressure and other data,Which once again proves that Knn classifier is better suited for this data.

5.1.3 Average Error Rates

FAR and **FRR**. Anyone who wants to assess or compare the performance of biometric security systems cannot ignore these terms. Those terms are as follows:
False Acceptance Rate (FAR): the percentage of identification instances in which unauthorised persons are incorrectly accepted.

FAR = False Acceptance Rate = $\frac{FP}{\text{FP + TN}}$

False Rejection Rate (FRR): the percentage of identification instances in which authorised persons are incorrectly rejected.

FRR = False Rejection Rate = $\frac{FN}{\text{FN + TP}}$

So, The average error is the average of both FAR and FRR. The lesser the value of average error, the better is the system.The average error obtained by each algorithm are mapped in the Fig.5.6. In Fig.5.6 once again Knn classification has outperformed the other algorithms with lesser average error in classification.

Since different data distributions do well on different machine learning algorithms. In these distributions, Knn algorithms averaged really low in the error rates with the inclusion of pressure data.By the visualization of all the metrics, We have successfully proved that Knn classification is the best suitable algorithm.

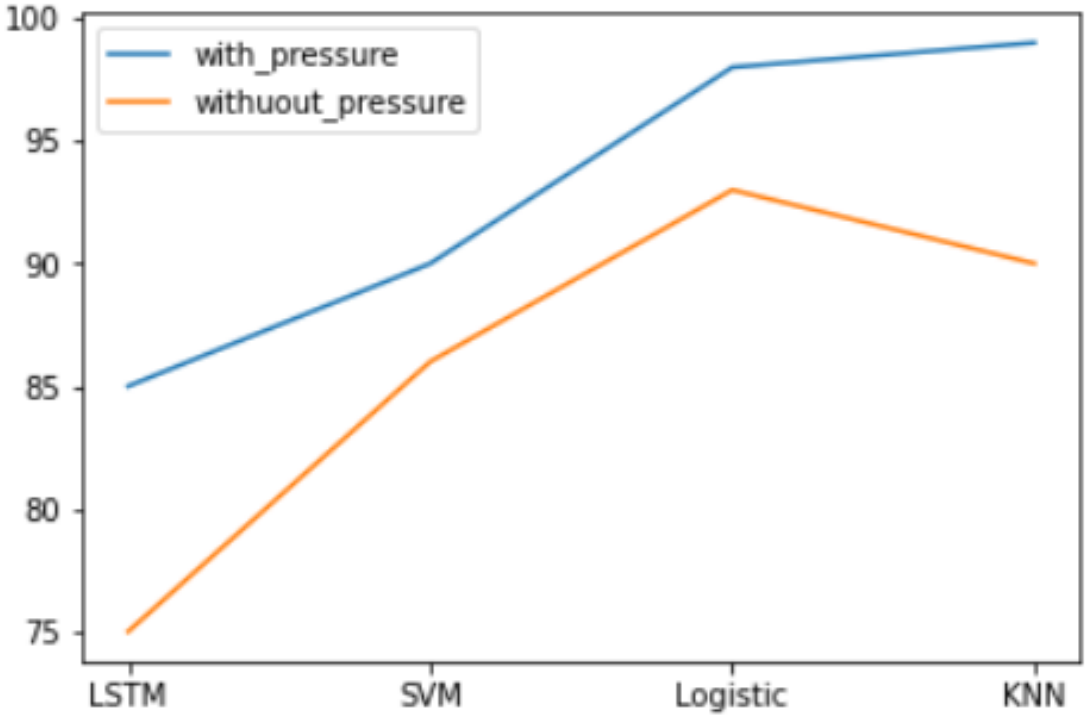


Figure 5.5: Comparison of average accuracies between Different Algorithms

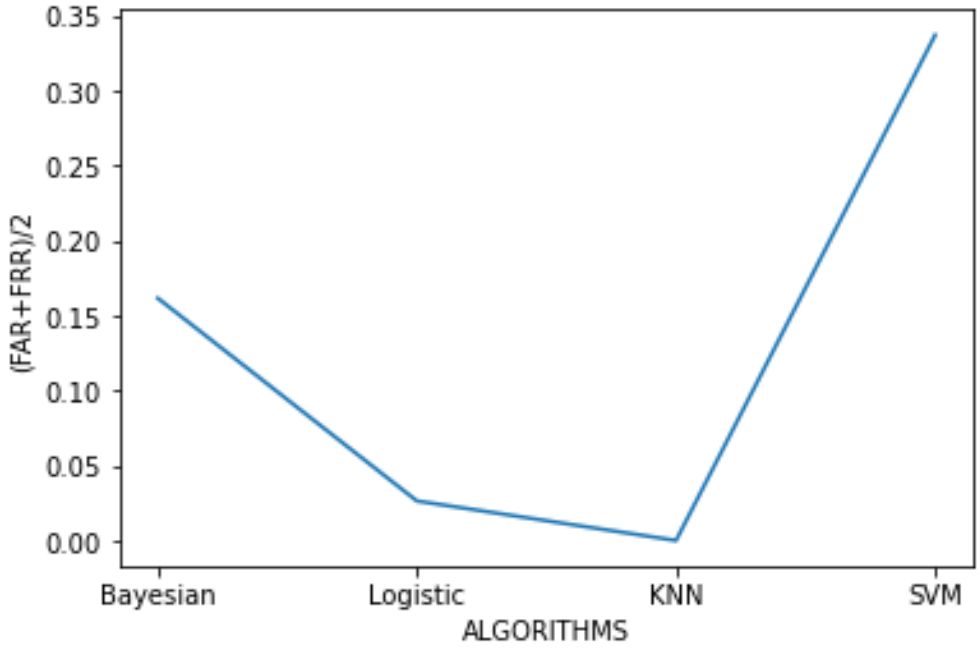


Figure 5.6: Comparison of error rates between Different Algorithms

Chapter 6

Conclusion

We have used different machine learning algorithms to classify the subject as genuine or fraud using their timing characteristics alone. Since, typing is used almost everywhere in our daily life, the case where genuine and fraud subject's timing characteristics match can be observed very often. In such cases we need some other parameters which can help us in distinguishing them. In this project, Those parameters were the pressure and other parameters. We have obtained pressure information along with the timing information followed by location information and used these parameters to classify the subjects based on them. Many algorithms performed well in this task. But KNN classification has outperformed the others.

Now for the future work, We have successfully completed the software and hardware implementation of the pressure sensitive keyboard with the addition of some more parameters which helps in increasing the security of the system. Although, the accuracy increased when the pressure and other data was included, The accuracy is being restricted to some value in between 98 to 99. Also the added parameters may not be useful in fraud detection always. Since we have taken history of check-in as one of the parameters i.e., Last check-in location and last check-in time. This is one of the many ways a fraud can be committed during system authentication. There can be a case where a fraud user may try to access others account within the same location and timings i.e., the case where location, timings information are compromised. In such situations we can not rely on a few parameters like timing and location parameters to detect the fraud and restrict him from authentication. So we need further more parameters which can help us in such situations. Also, In order to maintain this high accuracy in every case, we need to handle the cases where both the timing

and pressure parameters are being compromised as mentioned above. To rectify such cases we need some more new parameters which can help us in differentiating the genuine and fraud. So, we can add a new parameter of keystroke sounds. These are the sounds which are obtained on keystrokes. Since, there exists a pattern in typing, there may also exist a pattern in those keystroke sounds. This may help us in solving the problems in above mentioned cases. For this, we need to modify the pressure sensitive keyboard by involving a microphone in that system where keyboard sounds can also be collected simultaneously with pressure, timing etc.. There are many other such parameters which can be added to the existing model. We must also keep in mind that if we keep on adding more and more parameters the model may over fit and which results in a risky case where the system does not even allow the genuine user to authenticate properly. So, most important and reliable parameters must be taken into account while building the system. This mostly constitutes our future work in order to increase the stability of the system and accuracy of fraud detection.

Publications

1. R. Giot, B. Hemery and C. Rosenberger, "Low Cost and Usable Multimodal Biometric System Based on Keystroke Dynamics and 2D Face Recognition", Int'l Conf. on Pattern Recognition (ICPR), pp. 1128 -1131, 2010.

2. A. K. Jain, R. Bolle, and S. Pankanti (editors), "Biometrics: Personal Identification in Networked Society", Kluwer Academic Publishers, 1999.

3. A. K. Jain, S. Pankanti, S. Prabhakar, L. Hong, and A. Ross, "Biometrics: a grand challenge", Proc. Int'l Conf. on Pattern Recognition, vol. 2, pp. 935–942, August 2004.

4. A. K. Jain,A. Ross, and S. Prabhakar, "An introduction to biometric recognition", IEEE Trans. on Circuits and Systems for Video Technology, vol. 14, pp. 4–20, Jan 2004.

5. S. Prabhakar, S. Pankanti, and A. K. Jain, "Biometric Recognition: Security and Privacy Concerns", IEEE Security and Privacy Magazine, Vol. 1, No. 2, pp. 33-42, 2003.

6. D. Woodward, N. M. Orlans, and P. T. Higgins."Biometrics: Identity Assurance in the Information Age", McGraw-Hill, New York, USA, 2003

7. A. Messerman, T. Mustafic, S. Çamtepe and S. Albayrak, "Continuous and non-intrusive identity verification in real-time environments based on free-text keystroke dynamics' ', Int'l Joint Conf. on Biometrics (IJCB), 2011.

8. A. Peacock, X. Ke, and M. Wilkerson. "Typing patterns: A key to user identification", IEEE Security and Privacy, 2(5):40–47, 2004.

9. J.A. Robinson et al., "Computer User Verification Using Login String Keystroke Dynamics", IEEE Trans. Systems, Man and Cybernetics, Part A, vol. 28, pp. 236–241, Mar. 1998.

10. J. Leggett and G. Williams, "Verifying Identity via Keystroke Characteristics", Int'l J. Man-Machine Studies, vol. 28, no. 1, pp. 67–76, 1988.

11. Benjamin Ngugi, Beverly K. Kahn, and Marilyn Tremaine, "Typing Biometrics: Impact of Human Learning on Performance Quality", J. Data and Information Quality, Vol. 2, No. 2, p. 11, 2011.

12. D. Umphress and G. Williams, "Identity Verification through Keyboard Characteristics", Int'l J. Man-Machine Studies, Vol. 23, No. 3, pp. 263–273, 1985.

13. E. Al Solami, C. Boyd, A. Clark, and A. K. Islam, "Continuous Biometric Authentication: Can It Be More Practical?",IEEE Int'l Conf. on High Performance Computing and Communications (HPCC), pp. 647–652, 2010.

14. R. Zack, C. Tappert, and S. Cha, "Performance of a long-text-input keystroke biometric authentication system using an improved k-nearest-neighbor classification method", IEEE Int'l Conf. on Biometrics: Theory Applications and Systems (BTAS), pp. 1-6, 2010.

15. T. Sim and R. Janakiraman, "Are digraphs good for free-text keystroke dynamics? ", IEEE CVPR, pp. 17-22, 2007.

16. R. Gaines, W. Lisowski, S. Press, and N. Shapiro, "Authentication by keystroke timing: some preliminary results", Rand Rep. R-2560-NSF, Rand Corporation, 1980.

17. F. Bergadano, D. Gunetti, and C. Picardi, "User Authentication through Keystroke Dynamics", ACM Trans. Information and System Security, 5(4), pp. 367–397, 2002.

18. D. Gunetti and C. Picardi. "Keystroke analysis of free text", ACM Transactions on Information and System Security, 8(3):312–347, 2005.

19. S. Bleha, C. Slivinsky, and B. Hussein, "Computer-Access Security Systems using Keystroke Dynamics", IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 12, no. 12, 1990, pp. 1217–1222.

20. L. C. F. Ara ujo, L. H. R. Sucupira, M. G. Liz arraga, L. L. Ling, and J. B. T. Yabu-uti. "User authentication through typing biometrics features", In Proc.

1st Int'l Conf. on Biometric Authentication (ICBA), volume 3071 of Lecture Notes in Computer Science, pp. 694–700, 2004.

21. R. Joyce and G. Gupta. "Identity authentication based on keystroke latencies", Communications of the ACM, 33(2):168–176, 1990.

22. E. Yu and S. Cho. "GA-SVM wrapper approach for feature subset selection in keystroke dynamics identity verification", In Proc. Int'l Joint Conf. on Neural Networks (IJCNN), pp. 2253–2257, 2003.

23. S. Cho, C. Han, D. H. Han, and H. Kim. "Web-based keystroke dynamics identity verification using neural network", Journal of Organizational Computing and Electronic Commerce, 10(4):295–307, 2000.

24. P. Kang, S. Hwang, and S. Cho. "Continual retraining of keystroke dynamics based authenticator", In Proc. 2nd Int'l Conf. on Biometrics (ICB'07), pp. 1203–1211, 2007.

25. S. Haider, A. Abbas, and A. K. Zaidi. "A multi-technique approach for user identification through keystroke dynamics", IEEE Int'l Conf. on Systems, Man and Cybernetics, pp. 1336–1341, 2000.

26. Y. Li, B. Zhang, Y. Cao, S. Zhao, Y. Gao and J. Liu, "Study on the Beihang Keystroke Dynamics Database", Int'l Joint Conf. on Biometrics (IJCB), pp. 1-5, 2011.

27. K. S. Killourhy and R. A. Maxion, "Comparing Anomaly Detectors for Keystroke Dynamics", in Proc. 39th Annual Int'l Conf. on Dependable Systems and Networks (DSN2009), pp. 125-134, 2009.

28. F. Bergadano et al., "User Authentication through Keystroke Dynamics," ACM Trans. Information and System Security.

29. On Accuracy of Classification-based Keystroke Dynamics for Continuous User Authentication, Alaa Darabseh Department of Computer Science Texas Tech University Lubbock.

30. Authentication system using behavioral biometrics through keystroke dynamics,D. D. Alves ; G. Cruz ; C. Vinhal

31. Authentication through Keystrokes: What You Type and How You Type Md. Asraful Haque, Namra Zia Khan, Gulnar Khatoon

32. Effective User Authentication Using Keystroke Dynamics Based on Feature Selections Alaa Darabseh.