Naïve Bayes and similarity based methods for identifying computer users using keystroke patterns

Shrijit S. Joshi

Follow this and additional works at: https://digitalcommons.latech.edu/dissertations

Part of the Computer Sciences Commons
NOTE TO USERS

Page(s) missing in number only; text follows. The manuscript was microfilmed as received.

62

This reproduction is the best copy available.

UMI
NAÏVE BAYES AND SIMILARITY BASED METHODS FOR IDENTIFYING COMPUTER USERS USING KEYSTROKE PATTERNS

by

Shrijit S. Joshi, M. S.

A Dissertation Presented in Partial Fulfillment of the Requirements for the Degree Doctor of Philosophy

COLLEGE OF ENGINEERING AND SCIENCE LOUISIANA TECH UNIVERSITY

March 2009
LOUISIANA TECH UNIVERSITY
THE GRADUATE SCHOOL

October 27, 2008

We hereby recommend that the dissertation prepared under our supervision by
SHRIJIT SUDHAKAR JOSHI
entitled
NAIVE BAYES AND SIMILARITY BASED METHODS FOR IDENTIFYING COMPUTER USERS
USING KEYSTROKE PATTERNS

be accepted in partial fulfillment of the requirements for the Degree of
PhD IN COMPUTATIONAL ANALYSIS AND MODELING

Superintendent of Dissertation Research
Head of Department

Recommendation concurred in:
Advisory Committee

Approved:
Director of Graduate Studies
Dean of the College

Approved:
Dean of the Graduate School

GS Form 13a
(6/07)
In this dissertation, we present two methods for identifying computer users using keystroke patterns. In the first method “Competition between naïve Bayes models for user identification,” a naïve Bayes model is created for each user. In the training phase of this method, the model of a user is trained using maximum likelihood estimation on the key press latency values extracted from the texts typed by the user. In the user identification phase of this method, for each user we determine the probabilistic likelihood that the typed text belongs to a user. Finally, the typed text is assigned to the user with the highest likelihood value. In the second method “Similarity based user identification,” each user is represented by a distinct model. In the training phase of this method, the model parameters of a user are estimated using the extracted key press latency values from the texts typed by the user. In the user identification phase of this method, we assign a similarity score to each user given a typed text. The similarity score of a user is determined by finding the ratio between (1) the number of key press latency values extracted from the typed text similar to the estimated model parameters of the user and (2) the total number of key press latency values extracted from the typed text. Finally, the typed text is assigned to the user with the highest similarity score.

We also present a novel application of distance based outlier detection method for discarding outliers in the extracted key press latency values from a users’ typed text. Outliers are detected using the following three-step procedure: (1) for each extracted...
latency value $x$, a neighborhood region using a distance threshold is created, (2) a latency value $x_j$ is considered as a neighbor of $x_j$ if $x_j$ falls in the neighborhood region of $x$, and (3) the latency value $x_i$ is considered as an outlying value if the number of neighbors determined for $x_i$ are less than a pre-set threshold.

To empirically evaluate the performance of our proposed work, a keystroke data set was collected from ten users, where each user provided 15 typing samples. From the provided typing samples, six distinct datasets were created in which the number of user identification attempts varied from 150 to 54600. Results on the datasets indicate that the identification accuracy of the "Competition between naïve Bayes models for user identification method" ranges from 89.62% to 99.65% and the identification accuracy of the "Similarity based user identification method" ranges from 96.33% to 100%. Further, the performance of our proposed two user identification methods is compared with the performance of two user identification methods reported in the recent literature.

To further improve the performance of the user identification methods, we theoretically analyze Majority Voting Rule (MVR) based fusion of two or more user identification methods. We formulate a procedure for theoretically estimating the identification accuracy of the MVR based fusion of user identification methods. Our proposed procedure, unlike the procedure presented in the literature of MVR based fusion, does not assume that the methods to be fused have the identical identification accuracy. The theoretically estimated identification accuracy of the MVR based fusion of user identification methods is analyzed in the light of empirical results.
APPROVAL FOR SCHOLARLY DISSEMINATION

The author grants to the Prescott Memorial Library of Louisiana Tech University the right to reproduce, by appropriate methods, upon request, any or all portions of this Dissertation. It is understood that "proper request" consists of the agreement, on the part of the requesting party, that said reproduction is for his personal use and that subsequent reproduction will not occur without written approval of the author of this Dissertation. Further, any portions of the Dissertation used in books, papers, and other works must be appropriately referenced to this Dissertation.

Finally, the author of this Dissertation reserves the right to publish freely, in the literature, at any time, any or all portions of this Dissertation.

Author SHRIJIT S. JOSHI

Date 11/26/2008
# TABLE OF CONTENTS

ABSTRACT .................................................................................................................. iii
LIST OF TABLES ............................................................................................................. ix
LIST OF FIGURES .......................................................................................................... xi
ACKNOWLEDGMENTS ................................................................................................ xii

CHAPTER 1  INTRODUCTION ...................................................................................... 1
  1.1 Knowledge Based User Authentication ................................................................. 2
  1.2 Object Based User Authentication ........................................................................ 3
  1.3 Biometrics Based User Authentication .................................................................. 5
  1.4 Motivation for Using Keystroke Patterns ............................................................... 6
  1.5 Introduction to the Proposed Work ....................................................................... 9
    1.5.1 Keystroke Dataset ......................................................................................... 9
    1.5.2 Distance Based Outlier Detection Method ...................................................... 10
    1.5.3 CNBM ......................................................................................................... 12
    1.5.4 Similarity Based Method .............................................................................. 14

CHAPTER 2  LITERATURE REVIEW .......................................................................... 17
  2.1 User Authentication Using Structured Text .......................................................... 18
  2.2 User Authentication Using Unstructured Text ..................................................... 25

CHAPTER 3  TRAINING PHASE .................................................................................... 28
  3.1 Keystroke Data Extraction ................................................................................... 28
  3.2 Outlier Detection .................................................................................................. 31
CHAPTER 8 MAJORITY VOTING RULE BASED FUSION

8.1 Introduction to the Majority Voting Rule Based Fusion

8.2 Recognition Rate Estimation

8.2.1 Fusion of Two Classifiers

8.2.2 Fusion of Three Classifiers

8.2.3 Fusion of Four Classifiers

8.2.4 Fusion of ‘n’ Classifiers

8.3 Empirical Results

CHAPTER 9 CONCLUSIONS

REFERENCES
LIST OF TABLES

Table 3.1 Detecting outlying values of a vector using the distance based outlier detection method ............................................ 37
Table 4.1 Extraction of feature values when a text “purpose” is typed .................. 47
Table 4.2 Estimated mean and standard deviation values in the models of two users ................................................................. 48
Table 4.3 Estimated interval for the bins in the trained models of two users ........ 49
Table 4.4 Determining matching feature value and similar feature value .......... 54
Table 6.1 Description of the six datasets ........................................................ 64
Table 6.2 Empirical results of the CNBM method when the outliers are not detected .................................................................. 68
Table 6.3 Empirical results of the CNBM method on each set created for the six datasets when the outliers are not detected .......... 69
Table 6.4 Outlier detection using the distance based outlier detection method in the six datasets ...................................................... 71
Table 6.5 Empirical results of the CNBM method when the detected outliers are discarded ........................................................ 73
Table 6.6 Empirical results of the CNBM method on each set created for the six datasets when the detected outliers are discarded .... 75
Table 6.7 Empirical results of the Similarity based user identification method when the outliers are not detected ........................................... 78
Table 6.8 Empirical results of the Similarity based user identification method on each set created for the six datasets when the outliers are not detected ...... 79
Table 6.9 Empirical results of the Similarity based user identification method when the detected outliers are discarded ................. 81
Table 6.10  Empirical results of the Similarity based user identification method on each set created for the six datasets when the detected outliers are discarded. ................................................................. 82

Table 7.1  Empirical results of the Relative Measure based user identification method proposed by Bergadano et al. ........................................................................................................ 92

Table 7.2  Determining number of matching letter pairs are considered as similar letter pairs in the Illustration 7.1 ........................................................................................................ 95

Table 7.3  Empirical results of the Absolute Measure based user identification method proposed by Gunetti and Picardi over six datasets. ................................. 96

Table 7.4  Comparison between the identification accuracies obtained by four user identification methods on each of the six datasets ........................................................................ 98

Table 8.1  When two classifiers are fused using MVR, four decision vectors are possible given an input pattern. ................................................................. 104

Table 8.2  When three classifiers are fused using MVR, eight decision vectors are possible given an input pattern. ................................................................. 107

Table 8.3  The five decision vectors on which the MVR based fusion of four classifiers will make correct decision. ................................................................. 109

Table 8.4  Theoretical estimation of the identification accuracies when the user identification methods are fused using majority voting rule. ......................... 114

Table 8.5  Obtained identification accuracies on the six datasets when the user identification methods are fused using majority voting rule. ......................... 115
## LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Extraction of key press latency value when a string &quot;AB&quot; is typed. ..........30</td>
</tr>
<tr>
<td>3.2</td>
<td>Recording key press latency values of each letter pair in a distinct vector. .................................................................32</td>
</tr>
<tr>
<td>3.3</td>
<td>Recording key press latency values when text &quot;aaa ab ab aa&quot; is typed. ......33</td>
</tr>
<tr>
<td>3.4</td>
<td>Plotting the values present in a vector $X_{171}$........................................34</td>
</tr>
<tr>
<td>3.5</td>
<td>Pseudo-code of the proposed distance based outlier detection method.......36</td>
</tr>
<tr>
<td>3.6</td>
<td>Naïve Bayes model of a user. ...............................................................39</td>
</tr>
<tr>
<td>5.1</td>
<td>User typed in the provided text area. ..................................................56</td>
</tr>
<tr>
<td>5.2</td>
<td>Length of the samples typed by the users. .............................................59</td>
</tr>
<tr>
<td>5.3</td>
<td>Average elapsed time between two samples provided by the users. ..........60</td>
</tr>
<tr>
<td>6.1</td>
<td>Comparison between the identification accuracy obtained by the CNBM imethod - (1) when the outliers are not discarded and (2) when the outliers are discarded from each of six datasets. .........................74</td>
</tr>
<tr>
<td>6.2</td>
<td>Comparison between the identification accuracy obtained by the Similarity based user identification method - (1) when the outliers are not discarded and (2) when the outliers are discarded from each of six datasets. .......................84</td>
</tr>
<tr>
<td>7.1</td>
<td>Extracted key press latency values from two typed text &quot;purpose&quot; and &quot;propose&quot;. .................................................................89</td>
</tr>
<tr>
<td>7.2</td>
<td>Assigning ranks to the matching letter pairs observed in the training and testing typing samples.............................................90</td>
</tr>
<tr>
<td>7.3</td>
<td>Comparison between the identification accuracies obtained by four user identification methods on each of the six datasets...............98</td>
</tr>
</tbody>
</table>
ACKNOWLEDGMENTS

I sincerely thank Dr. Vir Phoha for providing me the opportunity to work in an exciting and challenging field of biometrics research. His invaluable guidance and generous support have often pushed me to put in my best possible efforts. His preference for perfection has inspired a great deal of my critical thinking. I am also grateful to Dr. Weizhong Dai, Dr. Dexter Cahoy, Dr. Jean Gourd, and Dr. Jinko Kanno for spending their time in reviewing my dissertation.

I thank all the members of the ADAM LAB and the SOFTMAC LAB for their guidance and knowledgeable discussions. I would especially like to thank Kiran Balagani and Sampath Vuyyuru. During my stay at Louisiana Tech, I came across with many people other than the ADAM LAB and the SOFTMAC LAB who constantly provided me great company: Sachin Jamthe, Shivashankar Suryanarayanan, Vikrant Patil, Ninad Mahajan, Vikram Darbhe, Kunal Tipnis, Vishnu Sharma, and Krithi Shetty. I am also grateful to my friends Abhijit Gadge, Suhas Kumbhalkar, Anil Barai, Sunil Chikate, Hitendra Sahare, Dhananjay Anwale, and Swati Aawale for their conversations over the phone that helped me feel at home.

Most importantly of all, I thank my mother Smita Joshi, my father Sudhakar Joshi, my sisters Sphurti (Gayatri Vaidya) and Siddhi (Aarati Pathak), and my brothers-in-law Prasanna Vaidya and Deepak Pathak for their unconditional love and support. My
family has been the pillar of my strength in all my endeavors and I dedicate this
dissertation to my family.
CHAPTER 1

INTRODUCTION

Computer systems are now widely used for performing a variety of tasks. However, there are various threats [1-6] to the security of these systems which can adversely affect their use for efficiently performing a task. Therefore, to prevent the computer systems from being affected by the security threats, computer scientists have proposed various security techniques. User authentication is one such security technique which aims at verifying the identity of a user [7]. If a user is positively authenticated by a user authentication technique, then the user will be allowed to access the computer system; otherwise, access will be denied.

User authentication techniques can be classified into three domains [8, 9]: (1) knowledge based user authentication, (2) object based user authentication, and (3) biometrics based user authentication. In the first domain, user authentication is performed by verifying whether a user remembers secret knowledge. For example, a password based user authentication system is from this domain. In the second domain of user authentication techniques, user authentication is performed by verifying whether the user has an object. In the third domain, user authentication is performed using a user's physiological characteristic such as fingerprint or behavioral characteristic such as keystroke pattern.
1.1 Knowledge Based User Authentication

In a knowledge based user authentication system, a secret word or phrase is shared between the user and the system. When a user is enrolled with the system, a shared secret is supplied by the user or by the system. The user has to provide the precise shared secret for positive authentication; if the user makes any error in providing the secret, the authentication fails. Password based user authentication systems and personal identification number based user authentication systems are examples of knowledge based user authentication systems [10].

Password based user authentication system is the most widely used system for user authentication. The following reasons can be attributed for the widespread use of this system for user authentication: (1) the system does not require any additional device for authentication; thus, it provides an inexpensive method for user authentication; (2) the system is easy to implement and easy to use [11]; and (3) in the case of remote user authentication, the strength of the system against security threats can be tuned up by employing cryptographic algorithm.

However, many password based user authentication systems are designed in such a way that the security of the authentication system relies entirely on a secret password. Previous studies on user authentication, such as [12-21], have shown that if the security of a user authentication system is entirely based on a secret password, then the system is prone to impostor attacks. Furthermore, in the password based user authentication system, a user has to provide his (or her) password precisely for positive authentication. Therefore, users generally use simple or meaningful passwords which are easy to remember [22]. Cheswick and Bellovin in [23] pointed out that these passwords are the
most common cause for impostors breaking into the system [24]. Impostors can crack such passwords by simple dictionary attacks or by an exhaustive search to compromise the system [25-27]. In addition, many users keep the same password across different applications, meaning an imposter who guesses a single password can break into the user authentication system of multiple applications [28].

Another domain of user authentication systems an object based user authentication system does not provide an economical solution for the user authentication problem when compared with that provided by a password based user authentication system. But object based user authentication system provides a stronger defense against impostors breaking into the system than that provided by password based user authentication system. (Note that it is difficult to measure the security defense provided by a user authentication system in absolute terms; therefore, we will relatively measure the security defense provided by two or more user authentication systems on specific imposter break-in scenarios.)

1.2 Object Based User Authentication

In object based user authentication systems, the user is authenticated using something he (or she) has, such as a smart card. An object used in this kind of authentication is usually a portable storage device [15]. During authentication, the object is typically read at the client end using an object reader to obtain a passcode. ('Passcode' is like a password, except it is machine generated, which may change with time.) The passcode is then transmitted to the host for user authentication.

Object based user authentication systems offer a stronger defense against imposter attacks than that offered by password based user authentication systems on
some impostor break-in scenarios. Some of those scenarios are (1) in a password based user authentication system, an impostor can guess the password of an authorized user. Similarly, in an object based user authentication system, an impostor can fraudulently acquire the object of an authorized user. But in the case of the password based user authentication system, the user whose password has been cracked may not know that his (or her) password has been compromised with someone. However in the case of the object based user authentication system, the user whose object has been lost or stolen may notice that his (or her) object has been acquired by someone and can act accordingly to stop the impostor breaking-in the system; (2) in object based user authentication system, typically a password is required to enable the object before using it for authentication. Thus, an impostor who wants to break into the system has to first acquire the object from an authorized user and then crack the password. However, in the case of password based user authentication system, an impostor can break into the system only by guessing the password of an authorized user.

In object based user authentication systems, an object reader is required for verifying the identity of a user. Both object and object reader costs money, and object reader should be available at every point where users are being authenticated [27]. Because of these reasons, these systems do not provide an inexpensive solution for the user authentication problem. In addition to this, an impostor can steal and forge the object of an authorized user for breaking into the system. Furthermore, an authorized user cannot get access to the computer system which the user is entitled to, if the user forgets the object or if the object malfunctions.
As discussed in Section 1.1 and in Section 1.2, both knowledge based and object based user authentication systems provide a solution for the user authentication problem. However, a major concern with these two traditional user authentication systems is that they do not directly verify the identity of a person; rather they verify whether the user remembers a secret word, as in the case of knowledge based user authentication systems, or whether the user has a unique object, as in the case of object based user authentication systems. If an impostor acquires the secret word or the object, then the impostor can break into the security of these traditional user authentication systems with ease. In other words, remembering a secret word or possessing a unique object does not necessarily prove the identity of a person. The other domain of user authentication systems biometrics based user authentication system, authenticates the identity of a user by neither verifying what the user remembers nor what the user possesses, but using his (or her) intrinsic characteristics, such as fingerprint.

1.3 Biometrics Based User Authentication

The word biometrics is derived from the two Greek words (1) bios meaning ‘life’ and (2) metricos meaning ‘measure’ [29]. In biometrics based user authentication system, user authentication is performed using something he (or she) ‘is’ or ‘produces’ [30]. Something he (or she) ‘is’ refers to a physiological characteristic of a user, and something he (or she) ‘produces’ refers to a behavioral characteristic of a user. The physiological characteristic of an individual is inherently associated with him (or her); while the behavioral characteristic of an individual is a trained act that he (or she) does unconsciously [30].
Biometric systems (biometrics based user authentication systems) offer several advantages over the traditional user authentication systems. Some of those advantages are (1) Access to an authorized user of the traditional user authentication systems is denied if the user forgets his (or her) password or if the user forgets his (or her) object. However, in the biometric systems, users do not have to remember any secret password and also do not have to carry any object for authentication; and (2) password can be guessed by an imposter and object can be fraudulently acquired or forged by an imposter, but it is theoretically impossible for an imposter to duplicate the biometric characteristic of any authorized user.

1.4 Motivation for Using Keystroke Patterns

Among many biometric identifiers (physiological and behavioral characteristics that are used for user authentication are referred to as biometric identifiers) proposed in the literature, keystroke patterns based user authentication systems provide a cost-effective solution for the user authentication problem. This could be because of the following two reasons: (1) a keystroke patterns based user authentication system does not require a separate device for extracting the measurements from a users' typing to form a keystroke pattern. The system only needs a keyboard to record the timings when a key was pressed and when a key was released to create a keystroke pattern. However, many biometric identifiers require a separate device to extract the measurements from a users' biometric identifier. For example, a fingerprint based user authentication system requires a separate scanning device to extract the measurements from a user's fingerprint. A device costs money and the device should be available at every point where users are being authenticated. Also, it is not obvious how to use biometric identifiers requiring an
additional device on the Internet. Therefore, a keystroke patterns based user authentication system provides a less costly solution for the user authentication problem when compared with the user authentication systems based on many other biometric identifiers; and (2) keystroke patterns based user authentication systems can be used for both static user authentication i.e., authenticating user before giving him (or her) access to the computer system and for continuous user authentication i.e., after the user has been given access to the computer system, it authenticates the identity of the user continuously. In the case of user authentication systems based on many other biometric identifiers, continuous user authentication could be intrusive for the users. However, in the case of keystroke patterns based user authentication systems, continuous user authentication would not be intrusive for the users because the system could run in the background while the user is typing at a keyboard.

Although keystroke patterns based user authentication system offers a less costly solution than many other biometric identifiers, user authentication using a physiological biometric identifier is considered to be more successful than user authentication using keystroke patterns. Various reasons can be credited for the lack of success of user authentication system using keystroke patterns. However, one prime reason may be keystroke pattern is from the domain of behavioral biometric identifiers. Being a behavioral biometric identifier, keystroke patterns of a user may change between two provided typing samples because of the change in the psychological or physiological condition of the user [31, 32].

Therefore, to minimize the effects of variability in the keystroke patterns on the performance of the user authentication system, most of the previous studies, such as [10,
have reported the performance of their proposed user authentication methods using the following experimental settings: (1) each user provided more than one typing sample of a text to create his (or her) typing profile; (2) users provided all the typing samples in one session to supply keystroke data for creating their typing profiles; and (3) the typing sample was discarded, if the user made any typing error while providing a sample. From these experimental settings, we may conclude that these authentication methods created a typing profile of a user using a structured text (i.e., the arrangement of words in the provided typing samples is fixed), which was provided more than one time consecutively. The typing profile of a user which was created on consecutively provided typing samples may not be perfectly representative of the user’s typing at a keyboard. This is because keystroke patterns of a user can change with the psychological condition of the user; and while providing consecutive samples, the psychological condition of the user could be most probably the same. Also, we may conclude that these authentication methods may not be applicable for the problem of identifying a user given arbitrary text, i.e., a text whose structure is unfamiliar to the user authentication method.

Furthermore, the presence of outliers in the data can adversely affect the performance of a keystroke patterns based user authentication system, if the outliers have not been detected effectively. This is because if observations that are deviating too much from other observations (i.e., outliers) are used for creating a typing profile of a user, then the created typing profile may not be perfectly representative of the user’s typing at a keyboard. However, very few studies on the keystroke patterns based user authentication system, such as [10, 39], have detected outliers in the keystroke data. These studies have detected outliers using some standard statistical distribution techniques.
1.5 Introduction to the Proposed Work

In this dissertation, we address the problem of the presence of outliers in keystroke data using a distance based outlier detection method, a novel approach in the field of user authentication using keystroke patterns. We also propose (1) Competition between naïve Bayes models for user identification (CNBM) method and (2) Similarity based user identification method for identifying a user given arbitrary text.

Keystroke patterns based user authentication systems can be employed for both user verification and user identification. In a user verification system, the user makes a claim and the system performs a one-to-one search to verify the claim. In a user identification system, the system performs a one-to-many search to identify a user from a set of users. A user verification system typically makes a decision on the authenticity of the claimed user using some user-defined-threshold(s). In other words, the performance of a user verification system can change with a change in the value of the user-defined-threshold. Therefore, to evaluate the performance of our methods with the methods proposed recently in [31, 32] without any bias, we identify a user from a set of users given a typed text.

Next, we briefly introduce a keystroke dataset that was used for (1) evaluating the performance of our proposed outlier detection method; and (2) comparing the performance of our proposed user identification methods with the user identification methods proposed recently in [31, 32].

1.5.1 Keystroke Dataset

An experiment was conducted for a month to collect the keystroke data set. In the experiment, a total of 15 typing samples from each of the ten users were collected.
During the collection of typing samples, the following experimental settings were used: (1) samples were created in such a way that entire text of any sample did not exactly match with that of any other sample, i.e., the structure (arrangement of words) of each sample was different from other samples; (2) the samples had lengths varying between 850 and 972 characters; (3) users were instructed to provide samples at different times of the day, such as one sample in the morning and next sample in the afternoon; and (4) users were allowed to make typing error(s), and no samples provided by the users for any reason have been discarded. When a user provided a typing sample, we recorded (1) the ASCII code of each pressed key, (2) the time when the keys were pressed, (3) the ASCII code of each released key, and (4) the time when the keys were released. From these recordings, we extracted the key press latency values between two successively pressed keys and recorded all the key press latencies from each English letter to each English letter. As a consequence, for each typing sample, we filled a total of $26 \times 26 = 676$ vectors with the recorded key press latency values. For example, the 1st vector was used to record the key press latency values for the letter pair "aa" and the 676th vector was used to record the key press latency values for the letter pair "zz".

Next, we present a brief description of the proposed distance based outlier detection method.

### 1.5.2 Distance Based Outlier Detection Method

We propose a distance based outlier detection method to detect outliers that may be present in the ‘training data’ of a user. (‘Training data’ of a user represents the extracted data from the typing samples that are selected for creating a typing profile of the user.) The proposed distance based outlier detection method is based on the study
performed by Knorr and Ng in [40]. Let us suppose, training data is extracted from $N$ typing samples of a user. To record the extracted key press latency values of each possible letter pair from the $N$ typing samples of the user, a total of $26 \times 26 = 676$ vectors are created. Later, the outliers that may be present in each vector are detected separately using the following three step procedure:

1. Determining the neighborhood of each key press latency value (data point) present in the vector;
2. Determining the number of neighbors of each data point; and
3. Flagging the data point as an “outlier”, if the number of neighbors determined for the data point is less than a user-defined value.

Note: (1) “neighborhood” of a data point represents a region around the data point, which is determined using a distance threshold and (2) “number of neighbors” of a data point represents the total number of data points of the vector are falling within the neighborhood of the selected data point.

To demonstrate the performance of the proposed distance based outlier detection method, user identification experiments were performed on six datasets. From the six datasets, a total of 1150 sets were created in such a way that the training data present in one set do not exactly match with that present in another set. For determining the effectiveness of the proposed distance based outlier detection method, two distinct user identification experiments were performed. In the first user identification experiment, the detected outliers by the outlier detection method are discarded from the training data and in the second user identification experiment, the detected outliers by the outlier detection method are not discarded from the training data. Empirical results of these user
identification experiments show that the average improvement in the identification accuracy of the CNBM method and the average improvement in the identification accuracy of the Similarity based user identification method is 42.16% and 42.53%, respectively, over six datasets when the detected outliers by the outlier detection method are discarded from the training data.

1.5.3 CNBM

Method “Competition between naïve Bayes models” (CNBM) for user identification is developed using the naïve Bayes classification technique. Our motivation for using the naïve Bayes classification technique is based on the following findings of various studies: (1) the naïve Bayes classifier is considered as a popular classification tool for solving various pattern classification problems, such as text classification, document categorization, speaker identification, and spam filtering, because of its robust performance and simple implementation [41]; (2) the computational complexity of the classifier in the classification phase is $O(n)$, where $n$ represents the number of features in the naïve Bayes model; and (3) naïve Bayes classifier assumes that the features of the model are independent of each other given the class. This independence assumption aids in estimating the parameters of the features separately, and this simplifies learning, especially when the number of features in the model is large [42].

In our naïve Bayes modeling, a letter pair represents a feature and extracted key press latency value of a letter pair represents a feature value. As there are possible $26 \times 26 = 676$ letter pairs, the naïve Bayes model of each user contains 676 features. In the training phase of this method, a total of 676 vectors are created for each user. These vectors are used to record the extracted feature values (i.e., key press latency values)
from the selected training typing samples of the user. The feature values present in these vectors represent the training data of the user. Before learning the naive Bayes model of a user, the outliers that may be present in the training data are detected using the proposed distance based outlier detection method. The detected outliers are then discarded from the training data, and the remaining training data is used to learn the parameters of the naive Bayes model using maximum likelihood estimation [41]. In the naive Bayes model of a user, we create two bins for each feature. An interval for the first bin of an $i^{th}$ feature is from $\mu_i - 2\sigma_i$ to $\mu_i + 2\sigma_i$, where $\mu_i$ represents the mean value and $\sigma_i$ represents the standard deviation value determined from the recorded feature values for the $i^{th}$ feature in the training data. Any feature value that does not fall into the interval created for the first bin of the feature is discretized into the second bin.

In the user identification phase of this method, a total of $26 \times 26 = 676$ vectors are created to record the extracted feature values from a given test typing sample. The feature values present in these vectors represent the test data. If an interval for the first bin of a feature is not determined in the naive Bayes models of all the users, then the recorded feature values for the feature are discarded from the test data. The remaining test data is then used to determine the probability of each user that the test typing sample is originated from his (or her) naive Bayes model. The test typing sample is then classified to the user whose model yielded the highest probability value.

To demonstrate the performance of the CNBM method, user identification experiments were performed on six datasets. From the six datasets, a total of 1150 sets were created in such a way that the training data present in one set do not exactly match with that present in another set. Empirical results show that (1) when 14 typing samples
are selected in the training data of each user, the identification accuracy of the method is 99.33%, and (2) the identification accuracy of the method was found to be decreasing with a decrease in the number of typing samples selected in the training data of each user. The obtained identification accuracies of this method are further compared with that of the user identification methods proposed recently in [31, 32].

1.5.4 Similarity Based Method

As discussed in the Paragraph 3 of the Section 1.5.3, if an interval for the first bin of a feature is not determined in the naïve Bayes models of all the users, then the recorded feature values for the feature are discarded from the test data. In other words, in our first user identification method, if an interval for the first bin of a feature is determined in the naïve Bayes model of a user, but not in the naïve Bayes models of all the users, then the recorded feature values for the feature in the test data are discarded. But if we can use the test data to determine a probability value (or a score) for each user independently, then we can use more, or at least the same, amount of information than our first method to make an identification decision. On the basis of this motivation, we formulated our second method “Similarity based user identification method” for identifying a user given arbitrary text.

In this method, a letter pair represents a feature and an extracted key press latency value of a letter pair represents a feature value. As there are possible $26 \times 26 = 676$ letter pairs, the model of each user contains 676 features. In the training phase of this method, a total of 676 vectors are created for each user. These vectors are used to record the extracted feature values (i.e., key press latency values) from the selected training typing samples of the user. The feature values present in these vectors represent the training data
of the user. Before learning the model of the user, the outliers that may be present in the
training data are detected using the proposed distance based outlier detection method.
The detected outliers are then discarded from the training data, and the remaining training
data is used to train the model. A trained model of a user is represented by two vectors:
(1) a vector containing the determined mean value for each of the \( n \) features, and (2) a
vector containing the determined standard deviation value for each of the \( n \) features.

In the user identification phase of this method, a total of \( 26 \times 26 = 676 \) vectors are
created to record the extracted feature values from a given test typing sample. The feature
values present in these vectors represent the test data. Next, a similarity score is assigned
to each of the users. Similarity score assigned to a user is determined using two measures:
(1) "matching feature values" – a feature value present in the test data is said to be a
'matching feature value', if the mean and standard deviation values are determined for
the feature in the user's trained model; and (2) "similar feature values" – a feature value
present in the test data is considered as 'similar feature value' to the model of the user, if
the feature value falls within \( \mu - 2 \cdot \sigma \) and \( \mu + 2 \cdot \sigma \) (\( \mu \) and \( \sigma \), respectively, represent the
mean and standard deviation value determined for the feature in the trained model of the
user). Assigned similarity score to the user is the ratio between the observed '# similar
feature values' and the '# matching feature values'. The test typing sample is then
classified to the user whose model yielded the highest similarity score.

To demonstrate the performance of the proposed Similarity based user
identification method, user identification experiments were performed on six datasets.
From the six datasets, total 1150 sets were created in such a way that the training data
present in one set do not exactly match with that present in another set. Empirical
evaluations show that when 13 or 14 typing samples are selected in the training data of each user, the identification accuracy of the method is 100%. However, the identification accuracy of the method was found to be decreasing with a decrease in the number of typing samples selected in the training data of each user. The obtained identification accuracies of the method are further compared with that of the user identification methods proposed recently in [31, 32].

The rest of the dissertation is organized as follows. Chapter 2 discusses related research in the field of user authentication using keystroke patterns. Chapter 3 details the proposed outlier detection method and the training phases of the proposed user identification methods. Chapter 4 details the user identification phases of the proposed two user identification methods. Chapter 5 presents the experimental settings that were used while collecting the keystroke dataset. Chapter 6 presents the identification results obtained by the proposed two user identification methods. Chapter 7 compares the performance of our proposed user identification methods with that of the user identification methods proposed recently in [31, 32]. Chapter 8 presents our work on theoretical estimation of the identification accuracy when user identification methods are fused using majority voting rule. We conclude the dissertation in Chapter 9.
In the literature, most of the studies have proposed user authentication methods using keystroke patterns for verifying the identity of a user given a predefined text. In these studies, a typing profile of a user is typically created on a text provided by the user more than one time. In addition, user authentication is also performed on the same predefined text. These kinds of user authentication methods can be employed for login time authentication. This is because a text provided by a user for creating his (or her) typing profile and a text provided for user authentication have the same structure, i.e., the arrangement of words in the two texts is fixed. These kinds of user authentication methods are referred to as “user authentication using structured text.”

Some studies have proposed user authentication methods that do not fall in the domain of “user authentication using structured text.” In these studies, users had not typed the same text more than one time. Therefore, a typing profile of a user is created using various structured texts. In addition, user authentication is performed on the text whose structure has not been seen by the authentication method earlier. This type of user authentication methods is referred to as “user authentication using unstructured text.” These methods can be employed for monitoring users, either continuously or periodically, during login session.
In Section 2.1, we present a brief description of various studies proposed in the domain of "user authentication using structured text." Some of the methods proposed in the domain of "user authentication using structured text" can be incorporated for "user authentication using unstructured text" with some modifications. A brief description of various studies proposed in the domain of "user authentication using unstructured text" is given in Section 2.2.

2.1 User Authentication Using Structured Text

Gaines et al. in [43] investigated the possibility of using keystroke patterns for user authentication. Seven users participated in their keystroke data set collection experiment. All the users were professional typists at Rand Corporation. To provide keystroke data for creating his (or her) typing profile, each user typed a text consisting of three paragraphs. To provide keystroke data for performing user verification, participating users were asked to type the same text after four months. However, only six users provided the keystroke data for user verification. For each typed text, the experiment recorded (1) the ASCII code of each pressed key and (2) the time (in milliseconds) when the keys were pressed. From these recordings, the key press latency values between two successively pressed keys were extracted. From these extracted values, the key press latency values from each English letter to each English letter and the key press latency values from each English letter to the space character were recorded. During verification of the identity of a claimed user given a typed text, a test of statistical independence was performed using a T-Test under the assumption that the mean latency values at both the sessions were the same. They reported that their verification system had zero false accept rate with 4% false reject rate on 55 test typing samples.
Leggett et al. in [44] proposed a user verification system. They conducted two experiments for empirically evaluating the performance of their user verification system. In the first experiment, 17 users participated and in the second experiment 36 users participated. In the first experiment, each user typed a text of about 1400 characters to provide keystroke data for creating his (or her) typing profile and each user typed a text of about 300 characters to provide keystroke data for performing user verification. In the second experiment, each user typed a text of about 537 characters at two different times separated by over a month. In both experiments, they extracted the key press latency values (in milliseconds) between all possible letter pairs. A typing profile of each user is represented by two vectors: (1) a vector containing the mean latency value of each possible letter pair and (2) a vector containing the standard deviation value determined from the observed latency values for each possible letter pair. During user verification, the mean key press latency values between all observed letter pairs were extracted from a test typing sample. A mean latency value of a letter pair observed in the test sample is considered as valid if it fell within 0.5 standard deviation of the mean latency value of the letter pair in the claimed user’s typing profile. Finally, a claimed user is positively verified if more than 60% of the observed letter pairs were considered as valid.

Joyce and Gupta in [10] proposed a user verification system in which each participating user was asked to type his (or her) username, password, first name, and last name eight times. While users were typing each time, key press latency values for each observed letter pairs were extracted. A typing profile of each user is represented by a vector containing the mean latency value of each observed letter pair. In their study, a latency value of a letter pair is considered as inlier if the latency value is falling within
the three standard deviations of the mean latency value determined for the letter pair. Remaining latency values were discarded before creating a typing profile of a user. This resulted in discarding about 0.85% of the latencies from the keystroke data set. During user verification, the user was asked to type a test signature (i.e., the username, password, first name, and last name) of any authorized user. From the typed test signature, the key press latency values were extracted for each observed letter pair. For user verification, a magnitude of difference was determined by finding the $L_1$ norm between the extracted latency values from the test signature and the mean latency values in the claimed user's typing profile. The claimed user was positively authenticated if the magnitude of difference was less than some user-defined-threshold.

Obaidat and Macchiarolo in [33] proposed a user identification system using the concept of Artificial Neural Networks (ANN). In their study, an experiment was conducted to collect keystroke data set. In their experiment, total six users participated, and each user typed a text consisting of 30 letters 40 times over a period of six weeks. To create a typing profile of each of the six users, a neural network was created and the network was then trained using back-propagation learning rule and sum-of-products network rule. They reported that the maximum authentication accuracy of their system on the keystroke data set was 97.50%. Other studies, such as [34-38], have also proposed a user authentication system using the concept of ANN with back-propagation learning rule. A major limitation of the user authentication system based on back-propagation learning rule is that the system has to be entirely retrained when a new user is registered to the system or when an existing user is removed from the system.
Robinson et al. in [45] proposed a user verification system. In their study, an experiment was conducted to collect keystroke dataset. In their experiment, ten users provided a password string twenty times (average string length of the passwords was about 6 characters). In addition, ten more people typed the password of each participating user ten times. In their experiment, any provided password containing a typing error was discarded from the keystroke dataset. For each typed password, they extracted the key hold times for each observed letter and the key interval latency values between all observed letter pairs. They proposed the user verification system using three classifiers: (1) minimum intra-class distance classifier, (2) nonlinear classifier, and (3) inductive learning classifier. They evaluated the user verification system on their keystroke dataset using key hold times and key interval latencies. They reported that the authentication accuracy of the verification system using key hold times was higher than that obtained using key interval latencies. They also reported that their verification system had the best false accept rate of 10%, with a false reject rate of 9%, using inductive learning classifier, when both key hold times and key interval latency values were used. However, in their study they do not mention (1) whether the decision obtained using key hold times and the decision obtained by key interval latencies were fused to make a verification decision or (2) a keystroke pattern was created using both key hold times and key interval latency values.

Haider et al. in [46] proposed a user verification system using ANN with back-propagation learning rule, fuzzy logic, statistical method, and a combination of the ANN, fuzzy logic, and statistical method. In their study, an experiment was conducted to collect a keystroke dataset. Each participated user provided 15 typing samples of his (or her)
selected password (password length on average was about 7 characters). In their experiment, any provided password containing a typing error was discarded from the keystroke dataset. To create a keystroke pattern of each typed password, key press latency values between the observed letter pairs were extracted. They reported the best false accept rate of 21% with the false reject rate of 2% when the verification system made a decision on the authenticity of the claimed user using a combination of ANN, fuzzy, and statistical method. However, they do not report (1) how the fusion of these three distinct classifiers was performed to make an authentication decision and (2) how many typing samples were used for empirically evaluating the performance of the user verification system.

Bergadano et al. in [31] proposed a user authentication system. In their study, a keystroke dataset gathering experiment was conducted to collect keystroke data from 44 users. Each user typed a sample text of 683 characters five times. In addition, 110 more users typed the sample text once. Therefore, a total of 330 typing samples were collected in their experiment. From each typing sample, the key press latency values between all observed letter pairs were extracted. They proposed an authentication method based on assigning ranks to each possible letter pair. During the training phase of their method, a rank was assigned to each of the observed letter pairs in each of the training samples. The ranks were assigned based on the mean latency value of the letter pairs. For example, the letter pair with the lowest mean latency value was assigned the rank '1' and the letter pair with the highest mean latency value was assigned the lowest possible rank. While processing a test typing sample, they extracted each latency value observed in the typed sample and determined the mean latency value for each observed letter pair. And the
letter pairs observed in the test sample were assigned ranks according to the technique mentioned earlier. During identifying a user for the test sample, each user was assigned a distance value. The distance value for a user was determined by finding the degree of disorder between the ranks for the letter pairs in the test sample and the assigned ranks for the letter pairs in each of the training samples of the user. The user with the least distance value was considered as the identified user for the given test sample. During authenticating the claimed user for a given test sample, they determined the distance value for each of the users according to the technique mentioned earlier. And a decision on the authenticity of the claimed user was made using some threshold criteria.

Determining a distance value by transforming the mean latency values of the letter pairs into ranks leads to various limitations of the techniques. For example, when the mean latency value of each possible letter pair of a user is exactly thrice that of other user. The typing profile of both users will assign the identical ranks to each letter pair. Therefore, the same distance value will be assigned to both the users for a given test sample; and hence, result in making a decision error by the authentication system.

Hosseinzadeh et al. in [47] proposed a user verification system using Gaussian mixture model (GMM). In their study, eight participating users provided keystroke data for empirically evaluating their proposed user verification system. Each user typed his (or her) first name and last name 10 times. For each typed text, two keystroke patterns were created: (1) a keystroke pattern with extracted key hold times and (2) a keystroke pattern with extracted key press latency values. Two models using GMM were created for each user: one model for key hold times and the other model for key press latency values. While verifying the identity of the claimed user given a test sample, keystroke patterns
were created by extracting key hold times and by extracting key press latency values from the typed test sample. Finally, a decision on the authenticity of the claimed user was made using some threshold criteria. They reported the following results: (1) when keystroke patterns were created using key hold times, the verification system had a false accept rate of 5.5% with the false reject rate of 1.4%; (2) when keystroke patterns were created using key press latencies, the verification system had a false accept rate of 9.2% with the false reject rate of 1.4%; and (3) when keystroke patterns were created using both key hold times and key press latency values, the verification system had false accept rate of 2.1% with the false reject rate of 2.4%. In their study, they do not report how many test samples were evaluated for determining the error rates of the verification system.

Joshi and Phoha recently proposed a user authentication system using the concept of self organizing map in [48]. In their study, a keystroke dataset was collected from 43 users. Each user typed “master of science in computer science” nine times. From each typed text, key hold times for each observed letter were extracted to create a keystroke pattern. Each user was represented by a distinct self organizing map, which was created by randomly selecting six patterns out of nine patterns. While authenticating a claimed user, a two-step procedure was adopted. In the first step, a one-to-many search was performed to identity a set of users for the typed text. In the second step, a one-to-one search was performed to determine a degree of similarity in terms of Euclidean distance between the keystroke pattern and the self organizing map of the claimed user. Finally, a decision on the authenticity of the claimed user was made using threshold criteria. Empirical evaluation of their authentication system on 873 test samples resulted in the
best false accept rate of 0.88% with the false reject rate of 3.55%. Like most of the studies presented in this section, this study also discarded a typing sample containing any typing errors and did not remove any outliers that may be present in the keystroke dataset.

2.2 User Authentication Using Unstructured Text

Studies proposed in the domain of “user authentication using unstructured text” are very limited. A brief overview of some of those studies is given below.

Monrose and Rubin in [39] proposed a user identification system. In their study, a keystroke dataset gathering experiment was conducted for a period of seven weeks. In their experiment, a total of 42 users participated; however, keystroke data of only 31 users was used due to erroneous timing results. Users were asked to type a few sentences from a list of given sentences and to type a few sentences of their own. From each typed text, the experiment extracted key press latency values and key hold times for the “features”. They do not report which features were used in the study, such as letters and space character or only letters. While creating a typing profile of a user, outliers present in the keystroke dataset were discarded using standard statistical distribution technique. Outlier detection was performed as follows: (1) first, a typing profile of a user was determined by finding the mean and the standard deviation value for each of the features; (2) next, each feature value was compared with its mean and standard deviation value, and any value greater than $T$ standard deviation away from its mean was considered as outlier and the value was discarded from the data set; and (3) the typing profile of a user was then updated. They reported that at $T = 0.5$, on average, more than 50% of the keystroke data was discarded. They proposed three distinct measures: (1) Euclidean
distance measure, (2) non-weighted probability measure, and (3) weighted probability measure for identifying a user given a typed text. They reported that the best identification accuracy of 23% was achieved using "weighted probability measure."

Dowland et al. in [49] proposed a user identification system. In their experiment, four users participated to provide keystroke data. The keystroke data was gathered while these users were doing normal activity on a computer. A typing profile of a user was determined by finding the mean and the standard deviation for the key press latency values of each of the "features." Note: (1) they do not mention which features were used in the study, such as letters and space character or only letters; (2) the mean and the standard deviation value were not determined for each feature, except for those that appeared some minimum number of times. While identifying a user given a test sample, mean key press latency values were determined for each of the observed features in the test sample. An observed feature was considered as valid if its feature value falls within a standard deviation of the feature's mean value in the typing profile of the user. The user having the highest number of valid features was considered as the identified user for the given test sample. They reported that the best identification accuracy of their identification system on the keystroke dataset is 50%.

Gunetti and Picardi in [32] proposed a user authentication system. In their keystroke dataset gathering experiment, each of the participating 40 users provided 15 typing samples. The text typed in each of the samples was spontaneously provided by the users. In addition, 165 more users provided a typing sample in the experiment. They provided two techniques for performing user authentication. The first technique was the same as that was presented in Bergadano et al. in [31] (this technique is briefly described
in Paragraph 7 of Section 2.1). In the training phase of the second technique, for each sample the mean key press latency value for each of the observed letter pairs was determined. While identifying a user given a test sample, Bergadano et al. assigned a score to each of the users. The score for a user was determined by finding the number of 'valid' letter pairs in the test sample with respect to each of the training sample of the user. A letter pair in the test sample was considered as 'valid' with respect to a training sample of a user if the mean latency value of the letter pair in the test sample was within 1.25 times the mean latency value of the selected training sample of the user. The user with the highest score was considered as the identified user for the test sample. Similarly, while authenticating a user, a score was determined for each of the users and a decision on the authenticity of the claimed user was made using some user-defined threshold.
CHAPTER 3

TRAINING PHASE

In this dissertation we propose two user identification methods: (1) Competition between naïve Bayes models (CNBM) for user identification and (2) Similarity based user identification method. The objective of the training phase in each method is to create a typing profile of a user using the provided training typing samples. Before a typing profile of a user is created, the following two data preprocessing operations are performed: Keystroke data extraction and Outlier detection. In the “Keystroke data extraction” operation, key press latency values are extracted from the provided training typing samples. And in the “Outlier detection” operation, outlying values that may be present in the extracted key press latency values are detected using the proposed distance based outlier detection method.

A description of the keystroke data extraction operation, the proposed distance based outlier detection method, and the training phases of our proposed two user identification methods follows. Next, we present our first data preprocessing operation “Keystroke data Extraction.”

3.1 Keystroke Data Extraction

When a user is providing a typing sample at a computer keyboard, timing information of the typing sample is typically extracted by recording two keystroke
events: (1) key press event and (2) key release event. These recordings can be used to ascertain the time when a key was pressed and when a key was released. These recordings are also helpful for determining the ASCII value of a pressed key and the ASCII value of a released key. The ascertained timing information aids in determining the following: (1) latency between consecutive key press and key release events i.e., the amount of time a key was held down and (2) latency between consecutive key release and key press events i.e., the flight time from a released key to a pressed key. In the literature, the amount of time a key was held down is referred as “key hold time” and the flight time from a released key to a pressed key is referred as “key interval latency.”

In our user identification experiments, the recordings of key press events and key release events were used to determine the latency between two successively pressed keys (referred as “key press latency”). “Key press latency” is determined by adding the latency between consecutive key release and key press events (i.e., key interval latency) to the latency between consecutive key press and key release events (i.e., key hold time).

To demonstrate the procedure for extracting key press latency values from a typed text, Figure 3.1 illustrates the extraction of the key press latency value when a string “AB” is typed. In the Figure 3.1, the key press time of letter ‘A’ and the key press time of letter ‘B’ are represented by \( KP_A \) and \( KP_B \), respectively. In the figure, the key release time of letter ‘A’ is represented by \( KR_A \). As shown in the Figure 3.1, the key hold time of letter ‘A’ (represented as \( KHT_A \)) is \( KHT_A = KR_A - KP_A \). Also we can see in the Figure 3.1 that the key interval latency between the letter pair “AB” (represented as \( KIL_{AB} \)) is \( KIL_{AB} = KP_B - KR_A \). From the key hold time of letter ‘A’ and the key interval latency
between the letter pair "AB", the key press latency between the letter pair "AB" (represented as $KPL_{AB}$) is determined as $KPL_{AB} = KHT_A + KIL_{AB}$.

![Diagram showing key press latency](image)

**Figure 3.1** Extraction of key press latency value when a string "AB" is typed.

In our user identification experiments, we extracted the key press latency value(s) for each observed letter pair from a provided typing sample. A letter pair may be repeated in a provided typing sample i.e., one or more key press latency values can be extracted for a letter pair from the provided typing sample. To record all the extracted key press latency values for each possible letter pair, a total of $26 \times 26 = 676$ vectors were created. Each of these 676 vectors was used to record the key press latency values for a particular letter pair. For example, the 1st vector was used to record the extracted key press latency values for the letter pair "aa" and the 676th vector was used to record the extracted key press latency values for the letter pair "zz". Note some vectors were empty, because some letter pairs had not been observed in a typing sample.

As discussed in the preceding paragraph, we record the extracted key press latency values (feature values) from a typing sample in the 676 vectors. If two or more typing samples are selected as the training typing samples of a user, then we create
another 676 vectors for the user to record all the extracted feature values from the selected typing samples. The feature values containing in the 676 vectors of the user constitute his (or her) training data. To detect whether any outlying value is present in a user’s training data, a distance based outlier detection method is proposed.

3.2 Outlier Detection

As discussed in the earlier section, we create 676 vectors for a user to record the extracted feature values from his (or her) selected training typing samples. Mathematically, these vectors of a user can be represented as

\[ X = \{ X_1, X_2, \ldots, X_{n-1}, X_n \}, \]

Equation 3.1

where subscript \( n \) represents the total number of vectors (in our case, the value of \( n \) is set to 676), \( X_1 \) represents a vector containing the extracted feature values for the first feature i.e., for letter pair “aa”, \( X_2 \) represents a vector containing the extracted feature values for the second feature i.e., for letter pair “ab”, \( X_{n-1} \) (in our case \( X_{675} \)) represents a vector containing the extracted feature values for the \((n-1)\)th feature i.e., for letter pair “zy”, and \( X_n \) (in our case \( X_{676} \)) represents a vector containing the extracted feature values for the \(n\)th feature i.e., for letter pair “zz”. A pictorial representation of these \( X_1 \) through \( X_{676} \) vectors is given in Figure 3.2. We can see in Figure 3.2 that (1) a vector \( X_i \) is created for each letter pair and (2) total 676 vectors are created as 676 possible letter pairs are possible. For example, with ‘a’ as the first letter in a letter pair we can have possible 26 letter pairs, from “aa” through “az”.
Furthermore, each vector $X_i$ can be represented as

$$X_i = \{x_{i1}^*, x_{i2}^*, \ldots, x_{im}^*\},$$  

Equation 3.2

where subscript $i$ refers to an $i$th feature, superscript $m_i$ represents the total number of times feature value for the $i$th feature is recorded, and $x_{ij}^*$ represents the recorded feature value for the $i$th feature at the $j$th component position in the $X_i$ vector. For instance — let us suppose, a user typed a text “aaa ab ab aa” using a keyboard. As shown in Figure 3.3, from the typed text “aaa ab ab aa”, we can extract three key press latency values for the letter pair “aa” and two key press latency values for the letter pair “ab”. The extracted key press latency values from this typed text are given in Figure 3.3. We can see that (1) the extracted key press latency values for the letter pair “aa” are 110, 90, and 100; therefore, in the figure $X_1 = \{110, 90, 100\}$ and (2) the extracted key press latency values for the letter pair “ab” are 170 and 160; therefore, in the figure $X_2 = \{170, 160\}$.
The following example, Example 3.1, will be used in the remainder of this subsection to illustrate three points: (1) how the extracted feature values are recorded in the vectors; (2) one or more values of a vector do vary or may vary so much from the remaining values of the vector as to arouse suspicion that these values are not generated by the same mechanism as that of the remaining values of the vector. In other words, outlying values may exist in the training data of a user; and (3) how the proposed outlier detection method detects outliers.

Example 3.1: The following seven feature values are extracted in our user identification experiment for the 171th feature i.e., for letter pair ‘go’ from the 14 typing samples of a user: 234, 516, 281, 250, 281, 265, and 1500.

Because a total of seven feature values are extracted for the 171th feature from the selected training typing samples of a user. The vector \(X_{mn}\), which is created to record the extracted feature values for the 171th feature, is given as:

\[
X_{mn} = \{ x_{1mn} = 234, x_{2mn} = 516, x_{3mn} = 281, x_{4mn} = 250, x_{5mn} = 281, x_{6mn} = 265, x_{7mn} = 1500 \}.
\]
To see whether any value of the vector is outlying with respect to the remaining values of the vector, the seven values present in the vector $X_{71}$ are plotted in Figure 3.4.

![Figure 3.4 Plotting the values present in a vector $X_{71}$.](image)

We can see in Figure 3.4 that the five values: $x_{1}^{1}$, $x_{1}^{7}$, $x_{1}^{4}$, $x_{2}^{7}$, and $x_{5}^{7}$ are nicely grouped together with the values ranging from 234 to 281. However, the two values of the vector $x_{1}^{1}$ and $x_{7}^{7}$ shows a variability with the remaining values of the vector $X_{71}$. These two values can be classified as the candidate outliers of this vector.

To detect such outlying values in a vector, the following three definitions are incorporated in the proposed outlier detection method. Note our proposed work on outlier detection is based on [40].

**Definition 3.1:** Neighborhood of a feature value $x_{i}^{j}$ is defined as a region around the value $x_{i}^{j}$ which ranges from $x_{i}^{j} - r$ to $x_{i}^{j} + r$. 

**Definition 3.2:** Feature value $x_i^t$ is considered as a neighbor of a feature value $x_j^t$, if the value $x_i^t$ falls within the neighborhood of $x_j^t$.

**Definition 3.3:** Feature value $x_i^t$ of vector $X_i$ is considered as an outlying value with respect to the remaining feature values of the vector $X_i$, if less than $\beta$ feature values of the vector are falling within the neighborhood of $x_i^t$.

A pseudo-code of the proposed distance based outlier detection method is given in Figure 3.5. The input to this method is the vectors $X = \{X_1, X_2, \ldots, X_n\}$ of a user. As shown in Step 1 of Figure 3.5, a vector $X_i = \{x_i^1, x_i^2, \ldots, x_i^n\}$ is created for each feature; where superscript $m_i$ represents the total number of times feature value for the $i^{th}$ feature is recorded in the $X_i$ vector and $x_i^t$ represents the recorded feature value for the $i^{th}$ feature at the $j^{th}$ component position in the $X_i$ vector. As shown in Step 2 of Figure 3.5, first, a neighborhood for each $x_i^t$ value is determined using Definition 3.1. Then, the total number of feature values of the $X_i$ vector falling within the neighborhood of a feature value $x_i^t$ is counted. In Figure 3.5, the total number of neighbors determined for an $x_i^t$ value is denoted by $NN(x_i^t)$. Finally, as shown in Step 3 of Figure 3.5, each feature value is either detected as outlier or inlier. If the total number of neighbors determined for a feature value is less than some $\beta$ percentage of the total number of values present in the vector, then the feature value is detected as an outlier. Otherwise, the feature value is considered as an inlier.
Input: Vectors $X = \{X_1, X_2, \ldots, X_{m+1}, X_n\}$ of a user.

Step 1: For $i = 1$ to $n$

Creating a vector for an $i^{th}$ feature $X_i = \{x_1^i, x_2^i, \ldots, x_m^i\}$

Step 2:

For $j = 1$ to $m$

Step 2.1: Determine the neighborhood of a feature value $x_j^i$.

Step 2.2: Initialize the number of neighbors determined for $x_j^i$ (i.e., $NN(x_j^i)$) to zero.

Step 2.3:

For $k = 1$ to $m$

- If the value $x_j^i$ falls within the neighborhood of $x_j^i$, then increment the number of neighbors determined for $x_j^i$ by 1 (i.e., $NN(x_j^i) = NN(x_j^i) + 1$).

End of $k$

End of $j$

Step 3:

For $j = 1$ to $m$

Step 3.1: If $NN(x_j^i)$ is less than some $\beta$ percentage of the total number of values present in the $X_j$ vector, then the feature value $x_j^i$ is detected as an outlier.

End of $j$

End of $i$

Figure 3.5 Pseudo-code of the proposed distance based outlier detection method.

Note parameter $r$, as given in Definition 3.1, is useful for determining the neighborhood of a feature value and the parameter $\beta$, as given in Definition 3.3, aids in setting a criterion for detecting outliers in a vector. Both these parameters must be set to some user-defined values in order to detect outliers using the proposed outlier detection method. In such a case, a well known problem of overfitting may arise i.e., selecting parameter values in such a way that method attains the best results on a particular dataset, but fails to attain the same kind of results on another dataset. To limit the problem, we set...
the value of $r$ to 100 and the value of $\beta$ to 68\% throughout our user identification experiments that are performed on total 1150 sets.

To describe how the proposed outlier detection method is implemented in our user identification experiments, Table 3.1 illustrates the obtained results of the outlier detection method on Example 3.1.

Table 3.1 Detecting outlying values of a vector using the distance based outlier detection method.

<table>
<thead>
<tr>
<th>Component Position in the vector</th>
<th>Feature Value</th>
<th>Neighborhood region</th>
<th>Neighbors</th>
<th>Number of Neighbors</th>
<th>Outlier decision</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>234</td>
<td>[134, 334]</td>
<td>234, 281, 250, 281, 265</td>
<td>5</td>
<td>No</td>
</tr>
<tr>
<td>2</td>
<td>516</td>
<td>[416, 616]</td>
<td>516</td>
<td>1</td>
<td>Yes</td>
</tr>
<tr>
<td>3</td>
<td>281</td>
<td>[181, 381]</td>
<td>234, 281, 250, 281, 265</td>
<td>5</td>
<td>No</td>
</tr>
<tr>
<td>4</td>
<td>250</td>
<td>[150, 350]</td>
<td>234, 281, 250, 281, 265</td>
<td>5</td>
<td>No</td>
</tr>
<tr>
<td>5</td>
<td>281</td>
<td>[181, 381]</td>
<td>234, 281, 250, 281, 265</td>
<td>5</td>
<td>No</td>
</tr>
<tr>
<td>6</td>
<td>265</td>
<td>[165, 365]</td>
<td>234, 281, 250, 281, 265</td>
<td>5</td>
<td>No</td>
</tr>
<tr>
<td>7</td>
<td>1500</td>
<td>[1400, 1600]</td>
<td>1500</td>
<td>1</td>
<td>Yes</td>
</tr>
</tbody>
</table>

In Table 3.1: (1) the first column “Component Position in the vector” gives a sequential ordering of the seven components of the $X_m$ vector, (2) the second column “Feature Value” shows the feature values present in the $X_m$ vector, (3) the third column
displays the “Neighborhood region” determined for each of the components of the vector, (4) the fourth column lists the “Neighbors” of each of the components of the vector, (5) the fifth column determines the “Number of Neighbors” of each of the seven components of the vector, and (6) the sixth column makes an outlier decision on each component based on the number of neighbors determined for a component. Note in this example, the value of $\beta = 0.68 \times 7 = 4.76$; therefore, feature values 516 and 1600 are considered as outliers.

The detected outliers by the proposed distance based outlier detection method are discarded from the vectors before creating a typing profile of a user using either CNBM method or Similarity based user identification method.

### 3.3 CNBM

As discussed in Section 3.1, first the feature values are extracted from the selected training typing samples of a user. The extracted feature values are then recorded in the 676 vectors created for the user which constitutes the training data of the user. Also as discussed in Section 3.2, the outlying values that may be present in the training data are detected using the distance based outlier detection method. Later in the training phase of the “CNBM” method, the detected outlying values are discarded from the training data and the remaining training data is used to create a typing profile of the user.

In the training phase, a naïve Bayes model is created for each user. The model has a total of 676 features, where each feature corresponds to a particular letter pair. Figure 3.6 gives a pictorial representation of the model. Under the naïve Bayes assumption, features of the model are independent of each other, given the class label [42, 50-53].
Therefore, in Figure 3.6, all the features $X = \{X_1, X_2, ..., X_n\}$ are shown as independent of each other, given the user $U$.

![Naive Bayes model of a user](image)

Each feature $X_i$ of the model is a discrete variable. However, the feature values that are extracted from a typing sample can range from 0 to $\infty$. Therefore, the values contained in the training data of a user must be discretized into some pre-decided $k$ bins of each feature.

The parameters of each feature $X_i$ of the model, with $k$ bins, can be represented as:

$$\Theta_i = \{\theta_1^i, \theta_2^i, ..., \theta_k^i\},$$  \hspace{1cm} \text{Equation 3.3}

such that $\sum_{j=1}^{k} \theta_j^i = 1$. Each $\Theta_i$ is assumed to follow a Dirichlet distribution [51] with parameters $\alpha, \alpha_2, ..., \alpha_k$ as the prior for each $\Theta_i$. The parameters of each $\theta_j^i$ are estimated using the following equation:

$$\theta_j^i = \frac{\alpha_j + y_j^i}{\alpha + m_i^j},$$  \hspace{1cm} \text{Equation 3.4}
where (1) \( m'_i \) represents the total number of times the feature values for the \( i^{th} \) feature is recorded in the training data of the user and (2) \( y'_j \) represents the total number of times the feature values for the \( i^{th} \) feature are discretized into the \( j^{th} \) bin. Typically in the Equation 3.4, \( \alpha_1, \alpha_2, \ldots, \alpha_n \) are estimated using Jaynes prior \([51, 54]\) i.e., setting the value of each \( \alpha_j \) to zero. However, if the value of any \( y'_j \) is zero then zero probability value will be estimated for the parameter \( \theta'_j \). To avoid this problem, Laplace’s estimate \([41, 51]\) was used for setting the value of each \( \alpha_j \), i.e., setting the value of each \( \alpha_j \) to 1.

In our user identification experiments, we have created two bins for each feature. An interval for the first bin of each feature is estimated by determining the mean and standard deviation values from the recorded feature values in the user’s training data. More specifically, if \( \mu_i \) and \( \sigma_i \), respectively, represent the mean and standard deviation value of the recorded feature values for an \( i^{th} \) feature, then the interval for the first bin of the feature will be \([ -\infty, \mu_i - \omega \sigma_i ] \). A feature value is discretized into the second bin of a feature, if the value does not fall into the estimated interval for the first bin of the feature.

Note parameter \( \omega \) is useful for determining an interval for the first bin of each feature. A numerical value must be set to \( \omega \) for estimating the parameters of the naïve Bayes model of each user. As noted earlier, in such a case a well known problem of overfitting may arise i.e., selecting a parameter value in such a way that method attains the best results on a particular dataset, but fails to attain the same kind of results on another dataset. To limit the problem, we set the value of \( \omega \) to 2 throughout our user identification experiments that are performed on total 1150 sets.
Next, a description of the training phase of the Similarity based user identification method follows.

### 3.4 Similarity Based User Identification Method

Before creating a typing profile of a user in the training phase of the “Similarity based user identification method,” feature values are extracted from the selected training typing samples of the user. The extracted feature values are then recorded in the 676 vectors created for the user which constitutes the training data of the user. Next, outliers that may be present in the training data are detected using the distance based outlier detection method. Later in the training phase of the Similarity based user identification method, the detected outlying values are discarded from the training data of the user and the remaining training data is used to create a typing profile of the user.

In the training phase of this method, a model is created for each user. The model has a total of 676 features, where each feature corresponds to a particular letter pair. The parameters of a user’s model are estimated by determining the mean and standard deviation values from the recorded feature values of each feature in the user’s training data. A trained model of a user is represented by two vectors: (1) a vector containing the determined mean values for each of the $n$ features i.e., $\mu = \{\mu_1, \mu_2, ..., \mu_n\}$, and (2) a vector containing the determined standard deviation values for each of the $n$ features i.e., $\sigma = \{\sigma_1, \sigma_2, ..., \sigma_n\}$. 
CHAPTER 4

USER IDENTIFICATION PHASE

We propose two user identification methods: (1) Competition between naïve Bayes models (CNBM) for user identification and (2) Similarity based user identification method. The objective of the user identification phase in the methods is to identify a user given a test typing sample.

Before searching the identity of the user, feature values (key press latency values) are extracted from the provided test typing sample. A letter pair may be repeated in a provided test typing sample, i.e., one or more key press latency values can be extracted for a letter pair. Therefore to record all the extracted key press latency values for each possible letter pair, a total of $26 \times 26 = 676$ vectors are created. Each of these 676 vectors is used to record the key press latency values for a particular letter pair. For example, the $1^{st}$ vector is used to record the extracted key press latency values for the letter pair “aa” and the $676^{th}$ vector is used to record the extracted key press latency values for the letter pair “zz”. The feature values (key press latency values) contained in these 676 vectors constitute the test data. Mathematically, these vectors can be represented as

$$Z = \{Z_1, Z_2, \ldots, Z_{n-1}, Z_n\}, \quad \text{Equation 4.1}$$
where subscript $n$ represents the total number of vectors (in our case, the value of $n$ is set to 676). $Z_i$ represents a vector containing the extracted feature values for the first feature i.e., for letter pair “aa”; $Z_{i+1}$ represents a vector containing the extracted feature values for the second feature i.e., for letter pair “ab”; $Z_{i+n-1}$ (in our case $Z_{675}$) represents a vector containing the extracted feature values for the $(n-1)^{th}$ feature i.e., for letter pair “zy”, and $Z_n$ (in our case $Z_{676}$) represents a vector containing the extracted feature values for the $n^{th}$ feature, i.e., for letter pair “zz”. Furthermore, each vector $Z_i$ can be represented as

$$Z_i = \{z'_i, z''_i, \ldots, z^n_i\},$$  

Equation 4.2

where subscript $i$ refers to an $i^{th}$ feature, superscript $m_i$ represents the total number of times feature value for the $i^{th}$ feature is recorded, and $z'_i$ represents the recorded feature value for the $i^{th}$ feature at the $j^{th}$ component position in the $Z_i$ vector.

In the user identification phase of the proposed two methods, these 676 vectors containing the test data are used to ascertain the identity of the user for a given test typing sample. A description of the user identification phases of the two user identification methods follows.

4.1 CNBM

The objective of the user identification phase in the “CNBM” method is to identify a user from a set of users $U=\{U_1, U_2, \ldots, U_{N-1}, U_N\}$ using Bayes’ theorem, where $N$ represents the total number of users registered with the user identification system. A probability value for each user $U_i$ that a set of vectors
$Z = \{ Z_1, Z_2, \ldots, Z_n \}$ is generated from his (or her) naïve Bayes model is determined using the Bayes’ theorem [55, 56] as follows:

$$P(U_i | Z) = \frac{P(Z | U_i) \cdot P(U_i)}{P(Z)}.$$  \hspace{1cm} \text{Equation 4.3}

In the above equation, the first term $P(U_i | Z)$ is the posterior probability of user $U_i$ given a set of vectors $Z$. As set $Z$ consists of vectors $Z_1, Z_2, \ldots, Z_n$, Equation 4.3 can be written as

$$P(U_i | Z) = P(U_i | Z_1, Z_2, \ldots, Z_n) = \frac{P(Z_1, Z_2, \ldots, Z_n | U_i) \cdot P(U_i)}{P(Z_1, Z_2, \ldots, Z_n)}.$$  \hspace{1cm} \text{Equation 4.4}

Under the naïve Bayes assumption, all the features are independent of each other given a user $U_i$’s model. Therefore, Equation 4.4 can be written as

$$P(U_i | Z) = P(U_i | Z_1, Z_2, \ldots, Z_n) = \frac{P(U_i)}{P(Z_1, Z_2, \ldots, Z_n)} \prod_{j=1}^{n} P(Z_j | U_i).$$  \hspace{1cm} \text{Equation 4.5}

Furthermore, each $Z_j$ vector consists of feature values $z_{1j}, z_{2j}, \ldots, z_{mj}$, where

1. subscript $j$ refers to a $j$th feature,
2. superscript $m_j$ represents the total number of times feature value for the $j$th feature is recorded in the vector $Z_j$, and
3. $z_{ij}$ represents the recorded feature value for the $j$th feature at the $i$th component position in the $Z_j$ vector. Therefore, Equation 4.5 can be written as

$$P(U_i | Z) = P(U_i | Z_1, Z_2, \ldots, Z_n) = \frac{P(U_i)}{P(Z_1, Z_2, \ldots, Z_n)} \prod_{j=1}^{n} P(z_{ij} | U_i).$$  \hspace{1cm} \text{Equation 4.6}

In the above equation, the numerator term $P(U_i)$ refers to the probability of observing user $U_i$. Numerical value for the term $P(U_i)$ is determined using the following equation:
\[ p(U_i) = \frac{\eta_i}{\sum \eta_i}, \quad \text{Equation 4.7} \]

where (1) \( \eta_i \) is the total number of training typing samples provided by the user \( U_i \), (2) \( \eta_i \) is the total number of training typing samples provided by a user \( U_i \), and (3) \( N \) is the total number of users provided training typing samples.

In Equation 4.6, the term \( P(z'_{j}/U_i) \) refers to the probability of observing a feature value \( z'_{j} \) for the \( j^{\text{th}} \) feature in the model of user \( U_i \). A numerical value for the term \( P(z'_{j}/U_i) \) is determined using the following two step procedure: (1) first, the feature value \( z'_{j} \) is discretized into one of the two bins created for the \( j^{\text{th}} \) feature in the model of user \( U_i \), and (2) then based on the bin to which the feature value \( z'_{j} \) is discretized, the estimated probability value of the bin is assigned to the term \( P(z'_{j}/U_i) \). In other words, a numerical value for the term \( P(z'_{j}/U_i) \) can be determined, if the feature value \( z'_{j} \) can be discretized into either of the bins created for the \( j^{\text{th}} \) feature in the model of user \( U_i \). However, note that (1) determining a bin to which a feature value \( z'_{j} \) can be discretized is based on the estimated interval for the first bin of the \( j^{\text{th}} \) feature in the model of user \( U_i \); and (2) an interval for the first bin of the \( j^{\text{th}} \) feature can be estimated, if two or more feature values for the \( j^{\text{th}} \) feature are observed in the user \( U_i \)'s training data. This means that a numerical value for the term \( P(z'_{j}/U_i) \) cannot be determined, if one or less than one time feature value for the \( j^{\text{th}} \) feature is observed in the user \( U_i \)'s training data.
Because of this, we may come across a situation where for one or more than one
$z'_j\,\text{ values a numerical value for the term } P(z'_j/U_i)\text{ can be determined for some users, but}
not for all the $N$ users. If this is the case, then we cannot make a comparison between the
obtained probability values for each of the $N$ registered users to make a user
identification decision. This is because Bayes' theorem can be used for comparing two or
more posterior probabilities, if all the obtained posterior probabilities are determined
using the same evidence. Therefore, to determine the posterior probability value for each
user using the same evidence, we select features from $X_1, X_2, \ldots, X_n$ such that an
interval for the first bin of each selected feature is estimated in the naïve Bayes model of
each of the $N$ users. In other words, we select the features from $X_1, X_2, \ldots, X_n$ such
that $P(z'_j/U_i)$ can be determined in each user's model, and thereby use the same
evidence to determine posterior probability value for each user. (Note that the term
$P(Z) = P(Z_1, Z_2, \ldots, Z_n)$ in Equation 4.6 is considered as constant, because it does not
provide any discrimination between the users.)

Finally, an identified user for a given test typing sample is the one whose
posterior probability is the highest among all other users. Mathematically, identifying
user $U_i$ for a given test typing sample is given as:

$$\text{Assign } Z \text{ to } U_i \text{ if } P(U_i/Z) = \max_{i=1:N} P(U_i/Z).$$  \hspace{1cm} \text{Equation 4.8}

Example 4.1, as given below, will be used in the remainder of this chapter to
illustrate the user identification phase of the "CNBM" method. (We will use the same
example in Section 4.2 to illustrate the user identification phase of the Similarity based
user identification method.)
Example 4.1: Let us suppose, a user identification system has two registered users: $U_1$ and $U_2$. Let the model of each user be trained on the same number of training typing samples. Let a test typing sample with text "purpose" be provided by a user. In Table 4.1, each extracted feature value (key press latency value) from the typed text "purpose" is given, along with the corresponding feature (letter pair) and its corresponding feature number. We can see in Table 4.1, the following six letter pairs are observed in the test typing sample: (1) "pu", (2) "ur", (3) "rp", (4) "po", (5) "os", and (6) "se".

Table 4.1 Extraction of feature values when a text "purpose" is typed.

<table>
<thead>
<tr>
<th>Feature (letter pair)</th>
<th>Feature number</th>
<th>Feature value (in milliseconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>pu</td>
<td>$Z_{pu}$</td>
<td>135</td>
</tr>
<tr>
<td>ur</td>
<td>$Z_{ur}$</td>
<td>105</td>
</tr>
<tr>
<td>rp</td>
<td>$Z_{rp}$</td>
<td>95</td>
</tr>
<tr>
<td>po</td>
<td>$Z_{po}$</td>
<td>90</td>
</tr>
<tr>
<td>os</td>
<td>$Z_{os}$</td>
<td>107</td>
</tr>
<tr>
<td>se</td>
<td>$Z_{se}$</td>
<td>74</td>
</tr>
</tbody>
</table>

Let us suppose, the letter pair "po" and the letter pair "ur" are not observed in the training data of user $U_1$ and in the training data of user $U_2$, respectively. Therefore, as shown in Table 4.2, the mean and the standard deviation values for the letter pair "po" and the letter pair "ur" has not been determined in the models of user $U_1$ and $U_2$, respectively.
Table 4.2 Estimated mean and standard deviation values in the models of two users.

<table>
<thead>
<tr>
<th>Feature (letter pair)</th>
<th>Feature number</th>
<th>User U₁</th>
<th></th>
<th>User U₂</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Mean</td>
<td>Standard deviation</td>
<td>Mean</td>
<td>Standard deviation</td>
</tr>
<tr>
<td>pu</td>
<td>Zₚᵤ</td>
<td>120</td>
<td>3</td>
<td>130</td>
<td>10</td>
</tr>
<tr>
<td>ur</td>
<td>Zₑᵤ</td>
<td>135</td>
<td>10</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>rp</td>
<td>Zₑᵦ</td>
<td>90</td>
<td>12</td>
<td>125</td>
<td>5</td>
</tr>
<tr>
<td>po</td>
<td>Zₑᵦ</td>
<td>-</td>
<td>-</td>
<td>100</td>
<td>8</td>
</tr>
<tr>
<td>os</td>
<td>Zₑₑ</td>
<td>130</td>
<td>10</td>
<td>95</td>
<td>10</td>
</tr>
<tr>
<td>se</td>
<td>Zₑₑ</td>
<td>140</td>
<td>20</td>
<td>75</td>
<td>10</td>
</tr>
</tbody>
</table>

Consequently, we can see in Table 4.3, an interval for the first bin of the feature corresponding to the letter pair “po” is not determined in the model of user U₁, and an interval for the first bin of the feature corresponding to the letter pair “ur” is not determined in the model of user U₂. To ascertain the identity of the user who might have typed the text “purpose” in example 4.1, we first select the features from Z₁, Z₂, ..., Zₙ such that an interval for the first bin of each selected feature is estimated in the trained models of user U₁ and U₂. We can see in Table 4.3, an interval for the first bin of feature Z₁₉₆ and that of feature Zₑₑ are not determined in the models of both the users. Therefore, these features are not selected, and the remaining extracted features from the test typing sample i.e., Zₚᵤ, Zₑₑ, Zₑₑ, and Zₑₑ are selected to make a user identification decision.
Table 4.3 Estimated interval for the bins in the trained models of two users.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Feature number</th>
<th>User U₁</th>
<th>User U₂</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Interval from</td>
<td>Interval to</td>
</tr>
<tr>
<td>pu</td>
<td>Z₁₁₁</td>
<td>110</td>
<td>130</td>
</tr>
<tr>
<td>ur</td>
<td>Z₁₈₈</td>
<td>115</td>
<td>155</td>
</tr>
<tr>
<td>rp</td>
<td>Z₄₄₈</td>
<td>61</td>
<td>114</td>
</tr>
<tr>
<td>po</td>
<td>Z₄₄₅</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>os</td>
<td>Z₅₅₀</td>
<td>110</td>
<td>150</td>
</tr>
<tr>
<td>se</td>
<td>Z₄₇₅</td>
<td>100</td>
<td>180</td>
</tr>
</tbody>
</table>

The posterior probability of user $U_i$ is determined as follows. First, the extracted feature values from the test typing sample are discretized. We can see from Table 4.1, Table 4.2, and Table 4.3: (1) the extracted feature value for the feature $Z_{u₁₁}$, i.e., $z_{u₁₁}$, does not fall in the estimated interval for the first bin of the feature; and hence, the feature will be discretized into the second bin; and (2) the extracted feature value for the feature $Z_{u₇₈}$, i.e., $z_{u₇₈}$, falls in the estimated interval for the first bin of the feature; and hence, the feature will be discretized into the first bin. Similarly, the extracted feature values for the feature $Z_{₂₃₅}$ and $Z_{₇₈₅}$ will be discretized into the second bin of the respective features.

Next, the probability value for user $U_i$ that test typing sample is generated from his (or her) model is determined as follows:
\[
P(U_i|Z) = \frac{P(U_i)}{P(Z)} \prod_{j=1}^{n} \left( \prod_{x} P(z_{x}/U_i) \right) \\
= \frac{0.5}{P(Z)} \left\{ P(z_{a1}/U_i) \cdot P(z_{e1}/U_i) \cdot P(z_{a2}/U_i) \cdot P(z_{e2}/U_i) \right\} \\
= \frac{0.5}{P(Z)} \cdot \left[ 0.2 \times 0.95 \times 0.15 \times 0.30 \right] \\
= \frac{0.004275}{P(Z)}.
\]

Note in the above calculation, the \( P(U_i) \) is 0.5 because both the users provided the same number of training typing samples. The posterior probability of user \( U_i \) is determined as follows. First, the extracted feature values from the test typing sample are discretized based on the estimated intervals in the model of user \( U_i \). We can see from the Table 4.1, Table 4.2, and Table 4.3: (1) the extracted feature values for the feature \( Z_{a1}, \ Z_{e1}, \) and \( Z_{e2} \) falls in the estimated interval for the first bin of the respective features; and hence, these values will be discretized into the first bin; and (2) as the extracted feature value \( Z_{a2} \) i.e., \( z_{a2} \) does not fall in the estimated interval for the feature, the feature value will be discretized into the second bin. Next, the probability value for user \( U_i \) that test typing sample is generated from his (or her) model is determined as follows:

\[
P(U_i|Z) = \frac{P(U_i)}{P(Z)} \prod_{j=1}^{n} \left( \prod_{x} P(z_{x}/U_i) \right) \\
= \frac{0.5}{P(Z)} \left\{ P(z_{a1}/U_i) \cdot P(z_{e1}/U_i) \cdot P(z_{a2}/U_i) \cdot P(z_{e2}/U_i) \right\} \\
= \frac{0.5}{P(Z)} \cdot \left[ 0.8 \times 0.05 \times 0.95 \times 0.80 \right] \\
= 0.0152 \\
= \frac{P(Z)}{P(Z)}.
\]
Among the two users, user $U_2$ has the highest posterior probability. Therefore, the identified user for the typed text "purpose" is user $U_2$.

### 4.2 Similarity Based User Identification Method

The objective of the user identification phase in this method is to identify a user from a set of users $U = \{U_1, U_2, ..., U_{k-1}, U_N\}$ by assigning a similarity score to each user present in the set $U$. A similarity score is assigned to a user, given a set of vectors $Z = \{Z_1, Z_2, ..., Z_{n-1}, Z_n\}$, where: (1) subscript $n$ represents the total number of vectors, in our case, the value of $n$ is set to 676; (2) $Z_1$ represents a vector containing the extracted feature values for the first feature i.e., for letter pair "aa" from a test typing sample; (3) $Z_2$ represents a vector containing the extracted feature values for the second feature i.e., for letter pair "ab" from a test typing sample; (4) $Z_{n-1}$, in our case $Z_{675}$, represents a vector containing the extracted feature values for the (n-1)$^{th}$ feature i.e., for letter pair "zy" from a test typing sample; and (5) $Z_n$, in our case $Z_{676}$, represents a vector containing the extracted feature values for the $n^{th}$ feature i.e., for letter pair "zz" from a test typing sample. Furthermore, each vector $Z_i = \{z_1^i, z_2^i, ..., z_m^i\}$, where: (1) subscript $i$ refers to an $i^{th}$ feature; (2) superscript $m_i$ represents the total number of times feature value for the $i^{th}$ feature is extracted from a test typing sample; and (3) $z_t^i$ represents the recorded feature value for the $i^{th}$ feature at the $t^{th}$ component position in the $Z_i$ vector.

To determine a similarity score for a user $U_i$, each feature value observed in $Z$ is compared with the trained model of $U_i$. As discussed in Section 3.4, a trained model of
a user is represented by two vectors: (1) a vector containing the mean value for each of the \( n \) features i.e., \( \mu = \{ \mu_1, \mu_2, \ldots, \mu_n \} \); and (2) a vector containing the standard deviation value for each of the \( n \) features i.e., \( \sigma = \{ \sigma_1, \sigma_2, \ldots, \sigma_n \} \). The following two measures are used for assigning a similarity score to user \( U_i \): (1) "matching feature values" - a feature value \( z'_j \) is said to be a "matching feature value", if the mean and standard deviation values are determined for the \( j^{th} \) feature in the trained model of \( U_j \); and (2) "similar feature values" - a feature value \( z'_j \) is considered as "similar feature value" to the model of a user \( U_j \), if feature value \( z'_j \) falls within \( \mu_j - 2\cdot\sigma_j \) and \( \mu_j + 2\cdot\sigma_j \), where \( \mu_j \) and \( \sigma_j \) respectively represent the mean and standard deviation value for the \( j^{th} \) feature in the model of user \( U_j \). Using these two measures, a similarity score \( SS_{U_i} \) is assigned to user \( U_i \), using the following equation:

\[
SS_{U_i} = \frac{\text{Number of Similar Feature values}}{\text{Number of Matching Feature values}}.
\]  

Equation 4.9

From the above equation, we can conclude that the highest similarity score that can be assigned to any user is 1 and the lowest similarity score that can be assigned to any user is 0. The user with the highest similarity score is considered as the identified user for a given test typing sample.

As discussed in Section 4.1, in the CNBM method we select features from \( Z_1, Z_2, \ldots, Z_n \) such that an interval for the first bin of each selected feature is estimated in the naïve Bayes models of all the users to make an identification decision. That is, in the CNBM method, we select extracted feature value for making an identification decision if, and only if, it is considered as a matching feature value in the models of all
the users. However, in the user identification phase of the Similarity based user identification method, a similarity score is assigned to a user by finding how many matching feature values are observed in the model of a particular user only. This helps us in using more, or at least the same, number of feature values in the Similarity based user identification method for making an identification decision.

To illustrate the user identification phase of this method, we solve Example 4.1 given in Section 4.1 using the user identification procedure described above. In Table 4.4: (1) the first column and the second column, respectively, give the extracted feature and its corresponding feature value when the text “purpose” is typed; (2) the third column and the fifth column, respectively, determine whether an extracted feature value can be considered as a “matching feature value” in the model of user $U_1$ and in the model of user $U_2$; and (3) the fourth column and the sixth column, respectively, determine whether an extracted feature value can be considered as a “similar feature value” in the model of user $U_1$ and user $U_2$. (Note if an extracted feature value from the test typing sample is not considered as a “matching feature value” in the model of a user, then the feature value cannot be used for making a decision on whether the feature value can be considered as a “similar feature value” to the model of the user. Therefore, for such cases '-' is shown in the column corresponding to the “similar feature value” in Table 4.4.)
Table 4.4 Determining matching feature value and similar feature value.

<table>
<thead>
<tr>
<th>Extracted Feature (letter pair)</th>
<th>Extracted Feature Value</th>
<th>User U₁</th>
<th>User U₂</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Matching feature value</td>
<td>Similar feature value</td>
</tr>
<tr>
<td>pu</td>
<td>135</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>ur</td>
<td>105</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>rp</td>
<td>95</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>po</td>
<td>90</td>
<td>No</td>
<td>-</td>
</tr>
<tr>
<td>os</td>
<td>107</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>se</td>
<td>74</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>

We can see in Table 4.4, feature values corresponding to letter pairs “pu”, “ur”, “rp”, “os”, and “se” are considered as the “matching feature values” to the model of the user $U₁$. Out of these five matching feature values, only one feature value corresponding to the letter pair “rp” is considered as “similar feature value” to the model of $U₁$. Therefore, the similarity score assigned to $U₁$ is $SS_{U₁} = \frac{1}{5} = 0.2$. Similarly, we can see in Table 4.4, feature values corresponding to letter pairs “pu”, “rp”, “po”, “os”, and “se” are considered as the “matching feature values” to the model of user $U₂$. Out of these five matching feature values, only one feature value corresponding to the letter pair “rp” is considered as not a “similar feature value” to the model of $U₂$. Therefore, the similarity score assigned to user $U₂$ is $SS_{U₂} = \frac{4}{5} = 0.8$. As $SS_{U₂} > SS_{U₁}$, the identified user for the typed text “purpose” is user $U₂$. 

CHAPTER 5

KEYSTROKE DATASET

A keystroke dataset gathering experiment was conducted from December 2007 through January 2008. For the experiment, a computer program using Microsoft Foundation Classes (MFC) was developed in Microsoft Visual C++ 6.0. The program provided a graphical user interface through which users were prompted to type their user IDs. In the experiment when a user was providing a typing sample for the first time, he (or she) was asked to provide a desired user ID. The provided user IDs were then considered as their unique identifiers. After a user provided his (or her) user ID, a new dialog containing a text area was provided to type a sample text. The text area was 72 characters wide and 15 lines long, to accommodate a total of 1080 characters.

When a user was typing in the text area, our program captured two Windows Messages: WM_KEYDOWN and WM_KEYUP. Windows Message WM_KEYDOWN was used for (1) extracting the ASCII value of a pressed key and (2) recording the time when the key was pressed. Similarly, Windows Message WM_KEYUP was used for extracting the ASCII value of a released key and recording the time when the key was released. These key press and key release timings were recorded in milliseconds using SYSTEMTIME structure, which has a timing resolution with an order of thousandth of a second. We recorded the timings in milliseconds because previous studies, such as [31,
32, 37, 38, 43, 44, 48, 57-63], have empirically shown that authentication systems perform well when the timings are recorded with a resolution of an order of thousandth of a second. Furthermore, Bechtel and Serpen in [64] empirically evaluated their authentication system when the timings are recorded with (1) a resolution of an order of hundredth of a second and (2) a resolution of an order of thousandth of a second. They found that the error rates of their authentication system were higher when the timings were recorded with a resolution of an order of hundredth of a second than that obtained with a resolution of an order of thousandth of a second (see page 7 in Section 2.1 of [64]).

The dialog containing the text area is depicted in Figure 5.1. After the user typed the required text in the text area, he (or she) clicked on the “Submit” button provided in the dialog, as shown in Figure 5.1.

![Figure 5.1 User typed in the provided text area.](image-url)
The “Submit” button as shown in Figure 5.1 was used to store the sampling data in our keystroke dataset. The sampling data consisted of following elements:

1. ASCII code of each pressed key;
2. the time when the keys were pressed;
3. ASCII code of each released key; and
4. the time when the keys were released.

The remainder of this chapter is organized as follows. First, a description of the user population of the experiment is given. Next, we answer some questions regarding the samples provided by the participated users, such as how many samples were provided by each participated user; how the samples were created; and what was the length of each sample. We will conclude this chapter by describing the experimental settings that were used while collecting the samples from each participating user.

5.1 Participated Users

In the experiment, total ten volunteers (seven male and three female) participated as legal users of a hypothetical computer user identification system. All the users were graduate students of Louisiana Tech University; eight users were pursuing graduate degrees in Computer Science, and the remaining two were pursuing graduate degrees in Civil Engineering. These users were neither hired nor in any way paid for their assistance, and not all the users knew the purpose of the experiment. Typing proficiency of the volunteers was not a requirement in the experiment; however, most of the users have been using computers for at least four years and some over ten years.
5.2 Typing Samples

In the keystroke dataset gathering experiment, each participated user provided 15 typing samples. These samples were created in such a way that the entire text of any sample did not exactly match with that of any other sample. In other words, the structure, i.e., arrangement of words of each sample, was different from that of the remaining samples. If the structure of all the samples is the same, then the empirical evaluation of a user identification system will be performed on a fixed text. However, the objective of this dissertation is to develop a user identification method which can identify the user given arbitrary text, i.e., a text whose structure is unfamiliar to a user identification method. By not having the text of each sample exactly matching with that of any other sample, we are able to simulate conditions which is more like identifying the user given arbitrary text. In the context of identifying the user given arbitrary text, Gunetti and Picardi in [32] state that keystroke analysis can be useful tool for user identification if the identification system can deal with arbitrary text.

The text used to create the typing samples was taken from various sources, such as books and Wikipedia. The text of each sample was based on different topics. Some of the topics were writing skills, literature, geography, artificial intelligence, dynamic programming, microeconomics, broadcast programming, web browser, production scheduling, etc. Furthermore, all the samples did not have the same length; by the length of a sample we mean total number of characters in the sample. Figure 5.2 shows the length of each of the 15 samples. In Figure 5.2, we can see that: (i) the samples have lengths varying between 850 and 972 characters; (ii) most of the samples have lengths between 850 and 950 characters (seven samples have lengths between 850 and 900
characters and six samples have lengths between 900 and 950 characters), and only two samples have lengths greater than 950 characters; and (iii) on average the sample length is about 899 characters.

Figure 5.2 Length of the samples typed by the users.

5.3 Sample Collection

In the experiment, each participated user provided 15 typing samples. While these samples were collected, a specific timing schedule, such as collecting samples everyday from 1 P.M. to 2 P.M., was not adopted. Because user identification using keystroke patterns is from the domain of behavioral biometrics, the users' typing patterns can be influenced by transient situations. Therefore, we instructed users to provide typing samples at different times of the day, such as type one sample in the morning, type the next sample in the afternoon, and type after performing different activities, such as after attending a class.

Participating users provided these samples on the basis of their availability. Some of the users provided the samples on a regular basis, like typing one sample per day, so
they completed their typing task in about 15-20 days. However, most of the users provided the samples on irregular basis; these users provided many samples where the elapsed time between two successive samples is more than two or three days. These users completed their typing task in about a month. In Figure 5.3, the average elapsed time (in hours) between two successively provided samples by each of the users is plotted.

In Figure 5.3, we can see that: (1) the average elapsed time of most of the participating users falls between one and two days, i.e., between 24 hours and 48 hours; (2) “user1” has the lowest average elapsed time of approximately 18 hours and “user6” has the highest average elapsed time of approximately 48 hours; and (3) three users: “user1”, “user2”, and “user5” have completed their typing task in about 20 days, and the remaining users completed their task in about a month.

This experiment was conducted using a Dell® keyboard (layout of the keyboard was QWERTY) on a desktop computer Dell® Dimension DIM4600 in our laboratory. Some of the previous studies on user authentication using keystroke patterns, such as [65-
have also used a keyboard throughout their keystroke data set collection experiment. Before typing a sample, users were unaware of the text they were going to type. While typing a sample, a printout of the text was given to them and also, the text was displayed on the desktop screen; the screen was shared by the text to be typed and the text area of our program where the user was typing. Most of the samples were provided by looking at the text displayed on the screen; some samples were provided by looking at the printout or by using both the options.

Users were instructed to provide samples as if they were writing a report on a topic. They were allowed to take pauses while typing the samples. While typing the samples, users made some typing errors. Various reasons can be attributed for the typing errors, such as the complexity of the word to be typed. If a user wished to correct them, then they had the option of correcting them. We have collected all the samples provided by the users irrespective of whether a provided sample contains any typing error (some previous studies, such as Gunetti and Picardi in [32] and Rao in [69], have also collected samples irrespective of whether they contain any typing errors), which is not the case in most of the previous studies on user authentication using keystroke patterns, such as [37, 45, 46, 48, 57, 61-63, 65, 67].

In the experiment, a total of 150 typing samples were provided by ten users, as described above. In the next chapter, we will use these typing samples to empirically evaluate the performance of the proposed two user identification methods.
CHAPTER 6

EXPERIMENTAL RESULTS

In this chapter we present a description of the six datasets that are created using the typing samples collected in our keystroke dataset collection experiment. Each dataset consists of training data and testing data. The datasets are created in such a way that the number of typing samples selected in the training data of a dataset does not exactly match with that in the training data of any other dataset. Creating datasets in such a manner aids in analyzing whether the number of typing samples selected in the training data has any effect on the performance of the two user identification methods proposed in this dissertation: (1) Competition between naïve Bayes models for user identification (CNBM) and (2) Similarity based user identification method. These datasets are also used for evaluating the performance of our proposed distance based outlier detection method in the light of the two user identification methods. A description of these six datasets follows.

6.1 Description of the Datasets

Table 6.1 gives a description of the six datasets created from the typing samples collected in our keystroke dataset collection experiment. The first column of this table illustrates the number of typing samples are selected in the training data of each user in a dataset. Based on the number of typing samples selected in the training data of each user
in a dataset, a label (dataset identifier) is assigned to the dataset. The label assigned to the dataset is given in the second column of the Table 6.1. For instance, if one typing sample is selected in the training data of each user in a dataset, then the label Dataset, is assigned to the dataset.

Table 6.1 Description of the six datasets.

<table>
<thead>
<tr>
<th># Training samples per user</th>
<th>Dataset label</th>
<th># Testing samples per user</th>
<th># Testing samples per set</th>
<th>Total number of testing samples</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Dataset₁</td>
<td>14</td>
<td>15</td>
<td>140 x 15 = 2100</td>
</tr>
<tr>
<td>2</td>
<td>Dataset₂</td>
<td>13</td>
<td>105</td>
<td>130 x 105 = 13650</td>
</tr>
<tr>
<td>3</td>
<td>Dataset₃</td>
<td>12</td>
<td>455</td>
<td>120 x 455 = 54600</td>
</tr>
<tr>
<td>12</td>
<td>Dataset₁₂</td>
<td>3</td>
<td>455</td>
<td>30 x 455 = 13650</td>
</tr>
<tr>
<td>13</td>
<td>Dataset₁₃</td>
<td>2</td>
<td>105</td>
<td>20 x 105 = 2100</td>
</tr>
<tr>
<td>14</td>
<td>Dataset₁₄</td>
<td>1</td>
<td>15</td>
<td>10 x 15 = 150</td>
</tr>
</tbody>
</table>

The third column of Table 6.1 illustrates the number of typing samples selected in the testing data of each user in a dataset. Each typing sample provided by a user is present in either training data or testing data of the user. In our keystroke dataset collection experiment, each user provided 15 typing samples. Therefore, the number of typing samples present in the testing data of a user can be determined by subtracting the number of typing samples selected in the training data of the user from the provided 15 typing
samples by the user. For instance, if one typing sample is selected in the training data of a user in a dataset, then the remaining 14 typing samples of the user are selected in his (or her) testing data.

The selected typing samples in the training data of a user are useful for creating a typing profile of the user. The created typing profile of each of the users is then employed to identify a user given a test typing sample. In other words, the selected typing samples in the training data forms the basis of user identification performance of a method. Therefore, to see whether any change in the selected typing samples in the training data has any effect on the user identification performance of the method, we create various sets for each dataset. The fourth column of Table 6.1 illustrates the number of sets are created for each dataset. These sets are created in such a way that each possible combination of the typing samples is selected in the training data of a dataset. For instance, if two typing samples are selected in the training data of a user in a dataset, then selecting any two typing samples out of 15 typing samples can be performed in

\[ \binom{15}{2} = \frac{15!}{2!(15-2)!} = 105 \]

possible ways. Therefore, when two typing samples are selected in the training data of a user, a total of 105 sets are created for Dataset 1.

The fifth column of Table 6.1 gives the number of typing samples selected as the training data in each set of a dataset. The number of typing samples selected as the training data in each set of a dataset is determined by multiplying the number of typing samples selected in the training data of a user i.e., the value given in the first column of the table with the number of users participating in the keystroke dataset collection experiment. For instance, if one typing sample is selected in the training data of a user in
a dataset, then the number of typing samples selected as the training data in a set of the
dataset is $1 \times 10 = 10$.

The sixth column of Table 6.1 gives the number of typing samples selected as the
testing data in each set of a dataset. The seventh column of Table 6.1 gives the total
number of typing samples selected as the testing data in a dataset. Each set of a dataset
has the same number of typing samples as the testing data. Therefore, the total number of
typing samples selected as the testing data in a dataset is determined by multiplying the
total number of testing samples selected as the testing data in each set of a dataset with
the number of sets created for the dataset. For instance, if two typing samples are selected
as the training data in each set of a dataset, then the number of typing samples selected as
the testing data in the Dataset 2 is $130 \times 105 = 13650$.

Next, a description on the measures that are used for evaluating the performance
of our proposed user identification methods on these created six datasets follows.

### 6.2 Performance Measures

User identification system makes either correct user identification or an incorrect
user identification given a test typing sample. When user identification system makes
correct user identification, the decision obtained by the system is referred to as “True
Positive” (TP); or otherwise, the decision is referred to as “False Negative” (FN). The
method with a higher value of “TP”, or the method with a lower value of “FN”,
represents a more accurate user identification method. However, the measures TP and FN
can be used for comparing the performance of the user identification methods when the
methods are evaluated on the same number of test typing samples. Therefore, to compare
the evaluation results of the user identification methods on a varying number of test
typing samples, a normalized measure "Identification Accuracy" is used. An identification accuracy of a method is calculated as: \( \frac{\#TP}{\#P} \times 100\% \), where \#TP represents the total number of typing samples for which the method made a correct user identification decision and \#P represents the total number of test typing samples evaluated by the method. The method with higher identification accuracy represents a more secure user identification method than the other methods.

6.3 Evaluation Results of the CNBM Method

In this section we present the empirical evaluation results of the CNBM method over the six datasets on two distinct user identification experiments. In the first user identification experiment, the outliers that may be present in the training data are not detected, and in the second user identification experiment, the detected outliers by the outlier detection method are discarded from the training data.

6.3.1 Before Detecting Outliers

Table 6.2 illustrates the empirical evaluation results of the CNBM method over the six datasets. During this evaluation, outliers that may be present in the training data are not detected. In Table 6.2, we present the total number of true positives (#TP) and the total number of false negatives (#FN) determined by the method on each dataset. From the value of #TP, the identification accuracy of the method is obtained; and the obtained identification accuracy is given in the last column of Table 6.2. We can see in the table that the highest identification accuracy of 60.70% is observed on Dataset_1, and the lowest identification accuracy of 50.00% is observed on Dataset_6.
Table 6.2 Empirical results of the CNBM method when the outliers are not detected.

<table>
<thead>
<tr>
<th>Dataset label</th>
<th>Total number of testing samples</th>
<th>Identification label</th>
<th>#TP</th>
<th>#FN</th>
<th>Identification Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dataset$_1$</td>
<td>2100</td>
<td></td>
<td>1238</td>
<td>862</td>
<td>58.95%</td>
</tr>
<tr>
<td>Dataset$_2$</td>
<td>13650</td>
<td></td>
<td>8286</td>
<td>5364</td>
<td>60.70%</td>
</tr>
<tr>
<td>Dataset$_3$</td>
<td>54600</td>
<td></td>
<td>32692</td>
<td>21908</td>
<td>59.88%</td>
</tr>
<tr>
<td>Dataset$_{12}$</td>
<td>13650</td>
<td></td>
<td>7091</td>
<td>6559</td>
<td>51.95%</td>
</tr>
<tr>
<td>Dataset$_{13}$</td>
<td>2100</td>
<td></td>
<td>1076</td>
<td>1024</td>
<td>51.24%</td>
</tr>
<tr>
<td>Dataset$_{14}$</td>
<td>150</td>
<td></td>
<td>75</td>
<td>75</td>
<td>50.00%</td>
</tr>
</tbody>
</table>

We can see in Table 6.2 that the identification accuracy of the CNBM method on Dataset$_1$ is 58.95%. In other words, the performance of the method has shown improvement of 1.75% when the typing samples in the training data of each user are increased from one to two (improvement is from 58.95% on Dataset$_1$ to 60.70% on Dataset$_2$). However, the performance of the method does not show any improvement when the typing samples are increased from two to three, twelve, thirteen, or fourteen in the training data. Rather, the performance of the method is reduced. For example: (1) the identification accuracy of the method is reduced by 0.82% when the typing samples in the training data of each user are increased from two to three, and (2) the identification accuracy of the method is reduced by 8.75% when the typing samples in the training data of each user are increased from two to twelve. From these results of the first user
identification experiment, we may conclude that the performance of the CNBM method does not show improvement with an increase in the number of typing samples in the training data when the outliers that may present in the training data are not detected.

To evaluate whether the performance of the CNBM method on a dataset is affected by the change in the typing samples selected in the training data, the obtained empirical results of the method on the six datasets are summarized in Table 6.3.

Table 6.3 Empirical results of the CNBM method on each set created for the six datasets when the outliers are not detected.

<table>
<thead>
<tr>
<th>Dataset label</th>
<th># Sets</th>
<th># Testing samples per set</th>
<th>Range in number of correct identifications (#TP)</th>
<th>Measuring spread in the #TP values (standard deviation)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dataset&lt;sub&gt;1&lt;/sub&gt;</td>
<td>15</td>
<td>140</td>
<td>66 – 121</td>
<td>13.21</td>
</tr>
<tr>
<td>Dataset&lt;sub&gt;2&lt;/sub&gt;</td>
<td>105</td>
<td>130</td>
<td>58 – 105</td>
<td>9.20</td>
</tr>
<tr>
<td>Dataset&lt;sub&gt;3&lt;/sub&gt;</td>
<td>455</td>
<td>120</td>
<td>54 – 95</td>
<td>7.41</td>
</tr>
<tr>
<td>Dataset&lt;sub&gt;4&lt;/sub&gt;</td>
<td>455</td>
<td>30</td>
<td>11 – 21</td>
<td>1.81</td>
</tr>
<tr>
<td>Dataset&lt;sub&gt;5&lt;/sub&gt;</td>
<td>105</td>
<td>20</td>
<td>7 – 14</td>
<td>1.30</td>
</tr>
<tr>
<td>Dataset&lt;sub&gt;6&lt;/sub&gt;</td>
<td>15</td>
<td>10</td>
<td>3 – 6</td>
<td>0.84</td>
</tr>
</tbody>
</table>

The number of sets determined for each dataset and the number of testing samples per set of a dataset is given, respectively, in the second and the third column of Table 6.3. While evaluating the performance of the method on a dataset, we record the total number of true positives determined in each set of the dataset. From these determined values, we
report its minimum and maximum value in Table 6.3. The spread in the determined values for the dataset is measured in their standard deviation value, as given in the last column of Table 6.3.

We can see in Table 6.3 that the difference between the maximum and minimum value determined for the total number of correct identifications (#TP) on (1) Dataset, is 55 ("minimum" is 66 and "maximum" is 121); (2) Dataset, is 47; (3) Dataset, is 41; (4) Dataset, is 10; (5) Dataset, is 7; and (6) Dataset, is 3. Note the difference between the "maximum" and "minimum" value determined for the datasets is expected to decrease with a decrease in the number of testing samples present in each set. This is because a range of values that are possible for the total number of true positives in a set will decrease with a decrease in the number of testing samples available for evaluation. On Dataset, the total number of true positives determined for its sets varies from 66 to 121 with a standard deviation of 13.21; in other words, the performance of the method has shown some variation with a change in the typing samples selected in the training data of the Dataset. A similar conclusion can be drawn for the remaining five datasets. From these results of the first user identification experiment, we may conclude that the performance of the CNBM method has shown some variation with a change in the typing samples selected in the training data.

Next, we present empirical evaluation results of the proposed distance based outlier detection method.
6.3.2 Outlier Detection

Table 6.4 illustrates the amount of data discarded from the training data of each
dataset using the proposed distance based outlier detection method. The average amount
of available training data for each user in a dataset is given in the column “Before Outlier
Detection” of Table 6.4. After the detected outliers are discarded, the average amount of
training data used for each user in a dataset is given in the column “After Outlier
Detection” of Table 6.4. The column “Absolute amount” of the table presents the average
amount of data detected as outliers in the training data of each user. The column “In
percentage” of the table presents the percentage of the data, out of available training data
for each user, detected as outliers by the distance based outlier detection method.

Table 6.4 Outlier detection using the distance based outlier detection method in the six
datasets.

<table>
<thead>
<tr>
<th>Dataset label</th>
<th># Sets</th>
<th>Average amount of data used in the training data of a user per set</th>
<th>Amount of data discarded from the training data of a user per set</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Before Outlier Detection</td>
<td>After Outlier Detection</td>
</tr>
<tr>
<td>Dataset1</td>
<td>15</td>
<td>607</td>
<td>479</td>
</tr>
<tr>
<td>Dataset2</td>
<td>105</td>
<td>1213</td>
<td>939</td>
</tr>
<tr>
<td>Dataset3</td>
<td>455</td>
<td>1820</td>
<td>1397</td>
</tr>
<tr>
<td>Dataset4</td>
<td>455</td>
<td>7280</td>
<td>5474</td>
</tr>
<tr>
<td>Dataset5</td>
<td>105</td>
<td>7886</td>
<td>5904</td>
</tr>
<tr>
<td>Dataset6</td>
<td>15</td>
<td>8493</td>
<td>6317</td>
</tr>
</tbody>
</table>
In Table 6.4, we can see that: (1) many outliers, ranging from 128 to 2176, were detected in the training data of each dataset using the distance based outlier detection method; (2) the minimum percentage of data was discarded on the Dataset1, and the maximum percentage of data was discarded on the Dataset4; and (3) in all the datasets more than 20% of the available data was detected as outliers. In the case of Dataset12, Dataset14, and Dataset16, almost 1/4th of the available data was detected as outliers.

Next, we empirically evaluate the performance of the CNBM method when the detected outliers by the distance based outlier detection method are discarded.

### 6.3.3 After Detecting Outliers

Table 6.5 illustrates the empirical results of the CNBM method. During this evaluation, detected outliers by the distance based outlier detection method are discarded. The total number of true positives (#TP), total number of false negatives (#FN), and the identification accuracy obtained by the CNBM method on each of the six datasets are given in Table 6.5. We can see in the table that the highest identification accuracy of the method of 99.65% is observed on Dataset12 and the lowest identification of 89.62% is observed on Dataset1.

We can also see in Table 6.5 that the identification accuracy of the method is increasing with an increase in the number of typing samples in the training data of each user. For example: (1) the identification accuracy of the method is improved by 8.55% when the typing samples in the training data of each user are increased from one to two (improvement is from 89.62% to 98.17%); (2) the identification accuracy of the method is improved by 9.67% when the typing samples in the training data of each user are
increased from one to three (improvement is from 89.62% to 99.29%); and (3) the identification of the method is improved by 10.03% when the typing samples in the training data of each user are increased from one to twelve (improvement is from 89.62% to 99.65%). One reason for such an improvement in the identification accuracy of the method could be the amount of data available for creating a typing profile of each user is increasing with an increase in the number of typing samples in the training data of the users. We note that the identification accuracies of the method on Dataset$_1$ and Dataset$_{14}$ are marginally decreased by 0.03% and by 0.32% with respect to that obtained by the method on Dataset$_{12}$.

Table 6.5 Empirical results of the CNBM method when the detected outliers are discarded.

<table>
<thead>
<tr>
<th>Dataset label</th>
<th>Total number of testing samples</th>
<th>#TP</th>
<th>#FN</th>
<th>Identification Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dataset$_1$</td>
<td>2100</td>
<td>1882</td>
<td>218</td>
<td>89.62%</td>
</tr>
<tr>
<td>Dataset$_2$</td>
<td>13650</td>
<td>13400</td>
<td>250</td>
<td>98.17%</td>
</tr>
<tr>
<td>Dataset$_3$</td>
<td>54600</td>
<td>54213</td>
<td>387</td>
<td>99.29%</td>
</tr>
<tr>
<td>Dataset$_{12}$</td>
<td>13650</td>
<td>13602</td>
<td>48</td>
<td>99.65%</td>
</tr>
<tr>
<td>Dataset$_3$</td>
<td>2100</td>
<td>2092</td>
<td>8</td>
<td>99.62%</td>
</tr>
<tr>
<td>Dataset$_{14}$</td>
<td>150</td>
<td>149</td>
<td>1</td>
<td>99.33%</td>
</tr>
</tbody>
</table>
To compare the performance of the CNBM method – (1) when the outliers are not discarded and (2) when the outliers are discarded, the obtained identification accuracies on each of the six datasets are plotted in Figure 6.1.

![Figure 6.1 Comparison between the identification accuracy obtained by the CNBM method - (1) when the outliers are not discarded and (2) when the outliers are discarded from each of six datasets.](image)

We can see in Figure 6.1 that the identification accuracy of the CNBM method when the detected outliers by the outlier detection method are discarded is higher on each of the six datasets than that of the same method when the outliers are not discarded. Improvement in the identification accuracies on each dataset is as follows: (1) on Dataset1, the identification accuracy is increased by 30.67% (from 58.95% to 89.62%); (2) on Dataset2, the identification accuracy is increased by 37.47% (from 60.70% to 98.17%); (3) on Dataset3, the identification accuracy is increased by 39.41% (from
59.88% to 99.29%); (4) on Dataset_{12}, the identification accuracy is improved by 47.70% (from 51.95% to 99.65%); (5) on Dataset_{13}, the identification accuracy is improved by 48.38% (from 51.24% to 99.62%); and (6) on Dataset_{14}, the identification accuracy is improved by 49.33% (from 50.00% to 99.33%).

To evaluate whether the performance of the CNBM method on a dataset when the outliers are discarded is affected by the change in the typing samples selected in the training data, the obtained empirical results of the method on the six datasets are summarized in Table 6.6.

Table 6.6 Empirical results of the CNBM method on each set created for the six datasets when the detected outliers are discarded.

<table>
<thead>
<tr>
<th>Dataset label</th>
<th># Sets</th>
<th># Testing samples per set</th>
<th>Range in number of correct identifications (#TP)</th>
<th>Measuring spread in the #TP values (standard deviation)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dataset_{1}</td>
<td>15</td>
<td>140</td>
<td>107 - 136</td>
<td>7.26</td>
</tr>
<tr>
<td>Dataset_{2}</td>
<td>105</td>
<td>130</td>
<td>122 - 130</td>
<td>1.83</td>
</tr>
<tr>
<td>Dataset_{3}</td>
<td>455</td>
<td>120</td>
<td>113 - 120</td>
<td>0.97</td>
</tr>
<tr>
<td>Dataset_{4}</td>
<td>455</td>
<td>30</td>
<td>29 - 30</td>
<td>0.31</td>
</tr>
<tr>
<td>Dataset_{5}</td>
<td>105</td>
<td>20</td>
<td>19 - 20</td>
<td>0.27</td>
</tr>
<tr>
<td>Dataset_{6}</td>
<td>15</td>
<td>10</td>
<td>9 - 10</td>
<td>0.26</td>
</tr>
</tbody>
</table>
The number of sets determined for each dataset and the number of testing samples per set of a dataset is given, respectively, in the second and the third column of Table 6.6. While evaluating the performance of the method on a dataset, we record the total number of true positives determined in each set of the dataset. From these determined values, we report its "minimum" and "maximum" value in Table 6.6. The spread in the determined values for the dataset is measured in their standard deviation value, as given in the last column of Table 6.6.

We can see in Table 6.6 that the difference between the maximum and minimum value determined for the total number of correct identifications (#TP) on (1) Dataset, is 29; (2) Dataset, is 8; (3) Dataset, is 7; (4) Dataset, is 1; (5) Dataset, is 1; and (6) Dataset, is 1. We can see in Table 6.6 that: (1) on the datasets Dataset, and Dataset, the total number of true positives determined for their sets varies by only 1 with a standard deviation value of less than 1; and (2) on the datasets Dataset, and Dataset, the total number of true positives determined for its sets varies by 8 and 7, respectively, with standard deviation values of around 2. From these results, we may conclude that the performance of the CNBM method observed on these datasets is almost constant with a change in the typing samples selected in the training data of a dataset. We note that on Dataset, the total number of true positives determined for its sets vary by 29 with a standard deviation of 7.26.

Next, we present the empirical evaluation results of the Similarity based user identification method on the six datasets.
6.4 Evaluation Results of the Similarity Based User Identification Method

In this section, we present the empirical evaluation results of the Similarity based user identification method over the six datasets on two distinct user identification experiments. In the first user identification experiment, the outliers that may be present in the training data are not detected, and in the second user identification experiment the detected outliers by the outlier detection method are discarded from the training data.

6.4.1 Before Detecting Outliers

Table 6.7 illustrates the empirical evaluation results of the Similarity based user identification method over the six datasets. During this evaluation, outliers that may be present in the training data are not detected. In Table 6.7, we present the total number of true positives (#TP) and the total number of false negatives (#FN) determined by the method on each dataset. From the value of #TP, the identification accuracy of the method is obtained; and the obtained identification accuracy is given in the last column of Table 6.7.

We can see in Table 6.7 that the highest identification accuracy of 61.08% is observed on Dataset₂ and the lowest identification accuracy of 52% is observed on Dataset₁. We can also see in the table that the identification accuracy of the method on Dataset₃ is 60.67%. The performance of the method has shown improvement of 0.41% when the typing samples in the training data of each user are increased from one to two (improvement is from 60.67% on Dataset, to 61.08% on Dataset₂). However, the identification accuracy of the method does not show any improvement when the typing samples are increased from one to three, twelve, thirteen, or fourteen. Rather, the
identification accuracy of the method is reduced when the typing samples in the training data of each user are increased from one to three, twelve, thirteen, or fourteen in the training data of each user. For example: (1) the identification accuracy of the method is reduced by 0.26% when the typing samples in the training data of each user are increased from one to three, and (2) the identification accuracy of the method is reduced by 7.52% when the typing samples in the training data of each user are increased from one to twelve. From these results of the first user identification experiment, we may conclude that the performance of the Similarity based user identification method does not show improvement with an increase in the number of typing samples in the training data when the outliers that may present in the training data are not detected.

Table 6.7 Empirical results of the Similarity based user identification method when the outliers are not detected.

<table>
<thead>
<tr>
<th>Dataset label</th>
<th>Total number of testing samples</th>
<th>Identifications</th>
<th>Identification Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dataset,</td>
<td>2100</td>
<td>1274</td>
<td>826</td>
</tr>
<tr>
<td>Dataset,</td>
<td>13650</td>
<td>8338</td>
<td>5312</td>
</tr>
<tr>
<td>Dataset,</td>
<td>54600</td>
<td>32985</td>
<td>21615</td>
</tr>
<tr>
<td>Dataset,</td>
<td>13650</td>
<td>7255</td>
<td>6395</td>
</tr>
<tr>
<td>Dataset,</td>
<td>2100</td>
<td>1108</td>
<td>992</td>
</tr>
<tr>
<td>Dataset,</td>
<td>150</td>
<td>78</td>
<td>72</td>
</tr>
</tbody>
</table>
To evaluate whether the performance of the Similarity based user identification method on a dataset is affected by the change in the typing samples selected in the training data, the obtained empirical results of the method on the six datasets are summarized in Table 6.8. The number of sets determined for each dataset and the number of testing samples per set of a dataset is given, respectively, in the second and the third column of Table 6.8. While evaluating the performance of the method on a dataset, we record the total number of true positives determined in each set of the dataset. From these determined values, we report its minimum and maximum value in the Table 6.8. The spread in the determined values for the dataset is measured in their standard deviation value, as given in the last column of Table 6.8.

Table 6.8 Empirical results of the Similarity based user identification method on each set created for the six datasets when the outliers are not detected.

<table>
<thead>
<tr>
<th>Dataset label</th>
<th># Sets</th>
<th># Testing samples per set</th>
<th>Range in number of correct identifications (#TP)</th>
<th>Measuring spread in the #TP values (standard deviation)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dataset 1</td>
<td>15</td>
<td>140</td>
<td>68</td>
<td>113</td>
</tr>
<tr>
<td>Dataset 2</td>
<td>105</td>
<td>130</td>
<td>61</td>
<td>100</td>
</tr>
<tr>
<td>Dataset 3</td>
<td>455</td>
<td>120</td>
<td>51</td>
<td>101</td>
</tr>
<tr>
<td>Dataset 4</td>
<td>455</td>
<td>30</td>
<td>12</td>
<td>21</td>
</tr>
<tr>
<td>Dataset 5</td>
<td>105</td>
<td>20</td>
<td>8</td>
<td>15</td>
</tr>
<tr>
<td>Dataset 6</td>
<td>15</td>
<td>10</td>
<td>4</td>
<td>7</td>
</tr>
</tbody>
</table>
We can see in Table 6.8 that the difference between the maximum and minimum value determined for the total number of correct identifications (#TP) on (1) Dataset₁ is 45; (2) Dataset₂ is 39; (3) Dataset₃ is 50; (4) Dataset₄ is 9; (5) Dataset₅ is 7; and (6) Dataset₆ is 3. On Dataset₁, the total number of true positives determined for its sets varies from 68 to 113 with a standard deviation of 11.55; this means the performance of the method has shown some variation with a change in the typing samples selected in the training data of the Dataset₁. Also on Dataset₁, the total number of true positives determined for its sets varies from 51 to 101 with a standard deviation of 7.49; this means the performance of the method has shown some variation with a change in the typing samples selected in the training data of the Dataset₁. Similar conclusion can be drawn for the remaining four datasets as well. From these results of the first user identification experiment, we may conclude that the performance of the Similarity based user identification method has shown some variation with a change in the typing samples selected in the training data.

Next, we empirically evaluate the performance of the Similarity based user identification method when the detected outliers are discarded.

6.4.2 After Detecting Outliers

Table 6.9 illustrates the empirical results of the Similarity based user identification method. During this evaluation, outliers detected by the distance based outlier detection method are discarded. The total number of true positives (#TP), total number of false negatives (#FN), and the identification accuracy obtained by the CNBM method on each of the six datasets are given in Table 6.9. We can see in the table that the
highest identification accuracy of the method of 100% is observed on Dataset, and Dataset,, and the lowest identification of 96.33% is observed on Dataset. ,.

Table 6.9 Empirical results of the Similarity based user identification method when the detected outliers are discarded.

<table>
<thead>
<tr>
<th>Dataset label</th>
<th>Total number of testing samples</th>
<th>Identifications</th>
<th>Identification Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>#TP</td>
<td>#FN</td>
</tr>
<tr>
<td>Dataset,</td>
<td>2100</td>
<td>2023</td>
<td>77</td>
</tr>
<tr>
<td>Dataset,</td>
<td>13650</td>
<td>13549</td>
<td>101</td>
</tr>
<tr>
<td>Dataset,</td>
<td>54600</td>
<td>54420</td>
<td>180</td>
</tr>
<tr>
<td>Dataset,,</td>
<td>13650</td>
<td>13649</td>
<td>1</td>
</tr>
<tr>
<td>Dataset,,</td>
<td>2100</td>
<td>2100</td>
<td>0</td>
</tr>
<tr>
<td>Dataset,,</td>
<td>150</td>
<td>150</td>
<td>0</td>
</tr>
</tbody>
</table>

We can also see in Table 6.9 that the identification accuracy of the method is increasing with an increase in the number of typing samples in the training data of each user. For example: (1) the identification accuracy of the method is improved by 2.93% when the typing samples in the training data of each user are increased from one to two; (2) the identification accuracy of the method is improved by 3.34% when the typing samples in the training data of each user are increased from one to three; (3) the identification accuracy of the method is improved by 3.66% when the typing samples in the training data of each user are increased from one to twelve; and (4) the identification...
accuracy of the method is improved by 3.67% when the typing samples in the training data of each user are increased from one to thirteen and fourteen. As discussed earlier in Section 6.3.3, one reason for such an improvement in the identification accuracy of the method could be the amount of data available for creating a typing profile of a user is increasing when there is an increase in the number of typing samples in the training data of the users.

To evaluate whether the performance of the Similarity based user identification method on a dataset is affected by the change in the typing samples selected in the training data, the obtained empirical results of the method on the six datasets are summarized in Table 6.10.

Table 6.10 Empirical results of the Similarity based user identification method on each set created for the six datasets when the detected outliers are discarded.

<table>
<thead>
<tr>
<th>Dataset label</th>
<th># Sets</th>
<th># Testing samples per set</th>
<th>Range in number of correct identifications (#TP)</th>
<th>Measuring spread in the #TP values (standard deviation)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dataset_1</td>
<td>15</td>
<td>140</td>
<td>129-139</td>
<td>2.77</td>
</tr>
<tr>
<td>Dataset_2</td>
<td>105</td>
<td>130</td>
<td>125-130</td>
<td>1.07</td>
</tr>
<tr>
<td>Dataset_3</td>
<td>455</td>
<td>120</td>
<td>114-120</td>
<td>0.74</td>
</tr>
<tr>
<td>Dataset_17</td>
<td>455</td>
<td>30</td>
<td>29-30</td>
<td>0.05</td>
</tr>
<tr>
<td>Dataset_13</td>
<td>105</td>
<td>20</td>
<td>20-20</td>
<td>0.00</td>
</tr>
<tr>
<td>Dataset_14</td>
<td>15</td>
<td>10</td>
<td>10-10</td>
<td>0.00</td>
</tr>
</tbody>
</table>
The number of sets determined for each dataset and the number of testing samples per set of a dataset is given, respectively, in the second and the third column of Table 6.10. While evaluating the performance of the method on a dataset, we record the total number of true positives determined in each set of the dataset. From these determined values, we report its “minimum” and “maximum” value in Table 6.10. The spread in the determined values for the dataset is measured in their standard deviation value, as given in the last column of Table 6.10.

We can see in Table 6.10 that the difference between the maximum and minimum value determined for the total number of correct identifications on (1) Dataset, is 10; (2) Dataset₂ is 5; (3) Dataset₃ is 6; (4) Dataset₄ is 1; (5) Dataset₅ is 0; and (6) Dataset₆ is 0. We can see in Table 6.10 that (1) on the datasets Dataset₆ and Dataset₇, the total number of true positives determined for all the sets are the same; (2) on Dataset₆, the total number of true positives determined for its sets varies by only 1 with a standard deviation value of 0.05; and (3) on Dataset₂ and Dataset₃, the total number of true positives determined for its sets varies by 5 and 6, respectively, with standard deviation values of around 1. From these results, we may conclude that the performance of the method observed on these datasets is almost constant with a change in the typing samples selected in the training data. We note that on Dataset, the total number of true positives determined for its sets vary by 10 with a standard deviation of 2.77.
To compare the performance of the Similarity based user identification method –
(1) when the outliers are not discarded and (2) when the outliers are discarded, the
obtained identification accuracies on each of the six datasets are plotted in Figure 6.2.

We can see in Figure 6.2 that the identification accuracy of the Similarity based
user identification method when the detected outliers by the outlier detection method are
discarded is higher on each of the six datasets than that of the same method when the
outliers are not discarded. Improvement in the identification accuracies on each dataset is
as follows: (1) on Dataset 1, the identification accuracy is increased by 35.66% (from
60.67% to 96.33%); (2) on Dataset 2, the identification accuracy is improved by 38.18%
(from 61.08% to 99.26%); (3) on Dataset 3, the identification accuracy is improved by

![Figure 6.2](image-url)

Figure 6.2 Comparison between the identification accuracy obtained by the Similarity
based user identification method - (1) when the outliers are not discarded and
(2) when the outliers are discarded from each of six datasets.
39.26% (from 60.41% to 99.67%); (4) on Dataset_1, the identification accuracy is improved by 46.84% (from 53.15% to 99.99%); (5) on Dataset_3, the identification accuracy is improved by 47.24% (from 52.76% to 100%); and (6) on Dataset_4, the identification accuracy is improved by 48% (from 52% to 100%).

In the next chapter, we compare the evaluation results obtained by our proposed two user identification methods with that obtained by the user identification methods proposed recently in [31, 32].
CHAPTER 7

COMPARISON WITH OTHER STUDIES

In this chapter, we compare the identification accuracies obtained by our proposed two user identification methods with that obtained by the user identification methods proposed recently by Bergadano et al. in [31] and by Gunetti and Picardi in [32]. Note that a direct comparison of the identification accuracy obtained by our methods over the six datasets with the reported identification accuracy in [31, 32] cannot be justified. This is because the keystroke datasets employed to report the identification accuracy in [31, 32] were not the same as used in our dissertation. Therefore, to compare the identification accuracies of the methods on the same keystroke dataset, we empirically evaluated the performance of the methods proposed by Bergadano et al. in [31] and by Gunetti and Picardi in [32] over our six datasets. Other studies on user identification using unstructured text do exist, such as [39, 49]. But in [39], Monrose and Rubin reported that the identification accuracy of their proposed methods is as high as 23%. In [49], Dowland et al. reported that the highest identification accuracy achieved by their method is 50%. These reported identification accuracies are far from being acceptable; hence, the methods proposed in [39, 49] are excluded from the comparison.

In [31], Bergadano et al. proposed a Relative Measure based user identification method. Gunetti and Picardi in [32] evaluated the performance of the Relative Measure
based user identification method introduced in [31] and proposed a Absolute Measure
based user identification method for performing user identification. A description of these
two user identification methods follows.

7.1 Relative Measure Based User Identification Method

In the Relative Measure based user identification method, each observed letter
pair in a typing sample is assigned a rank. The ranks are assigned based on the observed
mean key press latency value for the letter pairs. The letter pair which has the lowest
mean key press latency among other observed letter pairs is assigned the first rank, the
letter pair with the next lowest mean key press latency is assigned the second rank, and
accordingly other observed letter pairs in the sample are assigned ranks.

Let us assume, a test typing sample $Y = \{y_1, y_2, ..., y_n\}$ be provided by a
user, where $n$ represents the total number of possible letter pairs and each $y_i$ represents
the mean key press latency observed for an $i^{th}$ letter pair in the test typing sample. For
identifying a user from a set of users $U = \{U_1, U_2, ..., U_n\}$, each user $U_i$ is
assigned a distance score. The distance score is determined by finding the distance
between the test typing sample and each training sample provided by the user. For
example, the distance between the test typing sample $Y = \{y_1, y_2, ..., y_n\}$ and an $i^{th}$
training sample $X_i = \{x_1, x_2, ..., x_n\}$ of a user $U_i$ is determined using the following
four step procedure:

**Step 1:** First, the letter pairs for which the mean key press latency value is
determined in both $Y$ and $X_i$ vectors are determined. The determined
letter pairs are referred to as matching letter pairs.
Step 2: In vector $X$, ranks are assigned to the determined matching letters pairs based on their observed mean key press latency in the vector. Similarly in vector $Y$, ranks are assigned to the determined matching letters pairs based on their observed mean key press latency in the vector. (A procedure for assigning ranks to the letter pairs in a sample is given in the first paragraph of this section.)

Step 3: For each of the matching letter pairs, absolute difference between the ranks assigned in the vector $Y$ and in the vector $X$, is determined.

Step 4: All the determined absolute differences are then added together to calculate the distance between the vectors $Y$ and $X$. Let the calculated distance between $Y$ and $X$, be termed as $V_i$. The distance $V_i$ is then used to determine a normalized distance $d(U_i, X)$ between $Y$ and $X$, based on whether the number of matching letter pairs is even or odd. If the number of matching letter pairs, say $\kappa$, are even, then the distance $d(U_i, X)$ between $Y$ and $X$, is $d(U_i, X) = \frac{V_i}{(\kappa^2/2)}$. If the value $\kappa$ is odd, then the distance $d(U_i, X)$ between $Y$ and $X$, is $d(U_i, X) = \frac{V_i}{((\kappa^2-1)/2)}$.

Finally, the assigned distance score $D(U_i, Y)$ to a user $U_i$ is given by:

$$D(U_i, Y) = \sum_{i=1}^{M} \frac{d(U_i, X)}{M}$$
where $M$ represents the total number of training samples provided by $U_k$. The user with the least distance score is the identified user for the given test typing sample.

The following illustration will be used in the remainder of this section to demonstrate how the distance between the two typing samples is determined.

**Illustration 7.1**: Let us suppose, a test typing sample of text "purpose" is provided by a user. In this test typing sample, the following six letter pairs are observed: (1) letter pair “pu”, (2) letter pair “ur”, (3) letter pair “rp”, (4) letter pair “po”, (5) letter pair “os”, and (6) letter pair “se”. In part (A) of Figure 7.1, the extracted key press latency values for each of these letter pairs from the test typing sample are given.

![Figure 7.1 Extracted key press latency values from two typed text "purpose" and "propose".](image)

Let a user provided a training typing sample of text "propose". In this training typing sample, the following six letter pairs are observed: (1) letter pair “pr”, (2) letter pair “ro", (3) letter pair “op", (4) letter pair “po", (5) letter pair “os", and (6) letter pair
In part (B) of Figure 7.1, the extracted key press latency values for each of these letter pairs from the training typing sample are shown.

The distance between the test typing sample "purpose" and the training typing sample "propose" using the Relative Measure based user identification method is determined as follows:

**Step 1:** First, the letter pairs for which the mean key press latency value is determined in both the test typing sample and the training typing sample are determined. The determined letter pairs are referred to as matching letter pairs. In Illustration 7.1, there are three matching letter pairs: (1) letter pair "po", (2) letter pair "os", and (3) letter pair "se".

**Step 2:** As shown in Figure 7.2, the three matching letter pairs are assigned ranks in both the testing typing sample and the training typing sample. Part (A) of Figure 7.2 represents the assigned ranks to the matching letter pairs in the testing typing sample and part (B) of Figure 7.2 represents the assigned ranks to the matching letter pairs in the training typing sample.

![Figure 7.2](image.png)

Figure 7.2 Assigning ranks to the matching letter pairs observed in the training and testing typing samples.
Step 3: For each of the three matching letter pairs, an absolute difference between the ranks assigned in the test typing sample and the training typing sample is determined. We can deduce from Figure 7.2 that the absolute difference between the ranks assigned for (1) the letter pair "po" is \(|2 - 1| = 1\); (2) the letter pair "os" is \(|3 - 3| = 0\); and (3) the letter pair "se" is \(|1 - 2| = 1\).

Step 4: All the absolute differences determined in Step 3 are then added together to calculate the distance between the test typing sample and the training typing sample. Therefore, the distance between the test typing sample and the training typing sample in Illustration 7.1 is \(1 + 0 + 1 = 2\). As the number of matching letter pairs are 3, i.e., odd, the normalized distance between these two typing samples is \(\frac{2}{(3^2 - 1)/2} = 0.5\).

Next, we present the identification accuracies obtained by the Relative Measure based user identification method on our six datasets.

7.1.1 Empirical Evaluation

Table 7.1 illustrates the performance evaluation results of the Relative Measure based user identification method proposed in [31] on our six datasets. In Table 7.1, we present the total number of true positives (#TP) and the total number of false negatives (#FN) determined by the method on each dataset. From the value of #TP, the identification accuracy of the method is obtained; and the obtained identification accuracy is given in the last column of Table 7.1.
Table 7.1 Empirical results of the Relative Measure based user identification method proposed by Bergadano et al.

<table>
<thead>
<tr>
<th>Dataset label</th>
<th>Total number of testing samples</th>
<th>Identifications</th>
<th>Identification Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>#TP</td>
<td>#FN</td>
</tr>
<tr>
<td>Dataset_1</td>
<td>2100</td>
<td>1801</td>
<td>299</td>
</tr>
<tr>
<td>Dataset_2</td>
<td>13650</td>
<td>12850</td>
<td>800</td>
</tr>
<tr>
<td>Dataset_3</td>
<td>54600</td>
<td>52796</td>
<td>1804</td>
</tr>
<tr>
<td>Dataset_4</td>
<td>13650</td>
<td>13614</td>
<td>36</td>
</tr>
<tr>
<td>Dataset_5</td>
<td>2100</td>
<td>2097</td>
<td>3</td>
</tr>
<tr>
<td>Dataset_6</td>
<td>150</td>
<td>150</td>
<td>0</td>
</tr>
</tbody>
</table>

We can see in Table 7.1 that the highest identification accuracy of the method proposed by Bergadano et al. in [31] is observed on Dataset_5, and the lowest identification of the method is observed on Dataset_1. Furthermore, the performance of the method has shown improvement, with an increase in the number of typing samples in the training data of a user. For example: (1) the identification accuracy of the method is improved by 8.38% when the typing samples in the training data of each user are increased from one to two; (2) the identification accuracy of the method is improved by 10.94% when the typing samples in the training data of each user are increased from one to three; (3) the identification accuracy of the method is improved by 13.98% when the typing samples in the training data of each user are increased from one to twelve; (4) the
identification accuracy of the method is improved by 14.10% when the typing samples in the training data of each user are increased from one to thirteen; and (5) the identification accuracy of the method is improved by 14.24% when the typing samples in the training data of each user are increased from one to fourteen.

7.2 Absolute Measure Based User Identification Method

Like the Relative Measure based user identification method introduced in [31], the Absolute Measure based user identification method [32] also assigns a distance score to each user given a test typing sample. However in this method, the distance score for a user is determined using the absolute key press latency values of each observed letter pair in the samples rather than assigning ranks to the letter pairs.

Let us assume, a test typing sample \( Y = \{y_1, y_2, ..., y_n\} \) be provided by a user, where \( n \) represents the total number of possible letter pairs and each \( y_i \) represents the mean key press latency observed for an \( i^{th} \) letter pair in the test typing sample. While identifying a user from a set of users \( U = \{U_1, U_2, ..., U_{N_k}, U_{N_u}\} \), each user \( U_i \) is assigned a distance score using his (or her) provided training typing samples. The assigned distance score to a user \( U_i \) is determined by finding the distance between the test typing sample and each training sample provided by the user. For example, the distance between the test typing sample \( Y = \{y_1, y_2, ..., y_{n_t}, y_{n_t}\} \) and an \( i^{th} \) training sample \( X_i = \{x'_1, x'_2, ..., x'_{n_t}, x'_{n_t}\} \) of a user \( U_i \) is determined using the following three step procedure:
Step 1: First, the letter pairs for which the mean key press latency value is determined in both \( Y \) and \( X \) vectors are determined. The determined letter pairs are referred to as matching letter pairs.

Step 2: For each matching letter pair, a ratio \( \frac{\text{max value}}{\text{min value}} \) is determined, where “max value” and “min value”, respectively, represent the maximum and the minimum value observed for the matching letter pair in the training and testing typing samples. If the ratio is less than 1.25, the matching letter pair is said to be “similar”.

Step 3: Finally, a distance between the test typing sample and the training typing sample \( d(U_k, X_f) \) is determined as:

\[
d(U_k, X_f) = 1 \frac{\# \text{Similar matching letter pairs}}{\# \text{Matching letter pairs}}.
\]

Finally, the assigned distance score \( D(U_k, Y) \) to user \( U_k \) is given by:

\[
D(U_k, Y) = \frac{\sum_{j=1}^{M} d(U_k, X_j)}{M}
\]

where \( M \) represents the total number of training samples provided by user \( U_k \). The user with the least distance score is the identified user for the given test typing sample.

To demonstrate how the distance between a testing typing sample and a training typing sample is determined in this method, we determine the distance between the test typing sample “purpose” and the training typing sample “propose” as given in Illustration 7.1. The distance is determined using the following three step procedure as mentioned above:
Step 1: First, the letter pairs for which the mean key press latency value is determined in both the test typing sample and the training typing sample are determined. The determined letter pairs are referred to as matching letter pairs. In Illustration 7.1, there are three matching letter pairs: (1) letter pair “po”, (2) letter pair “os”, and (3) letter pair “se”.

Step 2: As shown in Table 7.2, each of the three matching letter pairs are verified whether it is “similar” or not based on the determined ratio value. As we can see in the column 4 of Table 7.2, the ratio value determined for each of the matching letter pair is less than 1.25; hence, all the matching letter pairs are considered as “similar”.

<table>
<thead>
<tr>
<th>Matching letter pair</th>
<th>Max value</th>
<th>Min value</th>
<th>Ratio</th>
<th>Similar</th>
</tr>
</thead>
<tbody>
<tr>
<td>po</td>
<td>90</td>
<td>80</td>
<td>1.125</td>
<td>Yes</td>
</tr>
<tr>
<td>os</td>
<td>107</td>
<td>105</td>
<td>1.019</td>
<td>Yes</td>
</tr>
<tr>
<td>se</td>
<td>90</td>
<td>74</td>
<td>1.216</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Step 3: Therefore, the distance between the test typing sample and the training typing sample in the Illustration 7.1 is $1 - \frac{3}{3} = 0$.

Next, we present the identification accuracies obtained by the Absolute Measure based user identification method proposed by Gunetti and Picardi in [32] on our six datasets.
7.2.1 Empirical Evaluation

Table 7.3 illustrates the performance evaluation results of the Absolute Measure based user identification method proposed by Gunetti and Picardi in [32] on our six datasets. In Table 7.3, we present the total number of true positives (#TP) and the total number of false negatives (#FN) determined by the method on each dataset. From the value of #TP, the identification accuracy of the method is obtained; and the obtained identification accuracy is given in the last column of the Table 7.3.

Table 7.3 Empirical results of the Absolute Measure based user identification method proposed by Gunetti and Picardi over six datasets.

<table>
<thead>
<tr>
<th>Dataset label</th>
<th>Total number of testing samples</th>
<th>#TP</th>
<th>#FN</th>
<th>Identification Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dataset_1</td>
<td>2100</td>
<td>1695</td>
<td>405</td>
<td>80.71%</td>
</tr>
<tr>
<td>Dataset_2</td>
<td>13650</td>
<td>11855</td>
<td>1795</td>
<td>86.85%</td>
</tr>
<tr>
<td>Dataset_3</td>
<td>54600</td>
<td>48528</td>
<td>6072</td>
<td>88.88%</td>
</tr>
<tr>
<td>Dataset_4</td>
<td>13650</td>
<td>12464</td>
<td>1186</td>
<td>91.31%</td>
</tr>
<tr>
<td>Dataset_5</td>
<td>2100</td>
<td>1920</td>
<td>180</td>
<td>91.43%</td>
</tr>
<tr>
<td>Dataset_6</td>
<td>150</td>
<td>138</td>
<td>12</td>
<td>92.00%</td>
</tr>
</tbody>
</table>

We can see in Table 7.3 that the highest identification accuracy of the method of 92% is observed on Dataset_6, and the lowest identification of 80.71% is observed on
Furthermore, we can see in the Table 7.3 that the identification accuracy of the method has shown improvement with an increase in the number of typing samples selected in the training data of a user. For example: (1) the identification accuracy of the method is improved by 5.80% when the typing samples in the training data of each user are increased from one to two; (2) the identification accuracy of the method is improved by 8.17% when the typing samples in the training data of each user are increased from one to three; (3) the identification accuracy of the method is improved by 10.60% when the typing samples in the training data of each user are increased from one to twelve; (4) the identification accuracy of the method is improved by 10.72% when the typing samples in the training data of each user are increased from one to thirteen; and (5) the identification accuracy of the method is improved by 11.29% when the typing samples in the training data of each user are increased from one to fourteen.

Next, a comparison between the (1) the method proposed by Bergadano et al. in [31], (2) the method proposed by Gunetti and Picardi in [32], and (3) our proposed two methods follows.

### 7.3 Comparison Between the User Identification Methods

In this section, we compare the identification accuracies obtained by our proposed two user identification methods with that obtained by two recently proposed user identification methods in the literature. In Table 7.4, the identification accuracies obtained on each of the six datasets by our proposed two user identification methods: (1) CNBM and (2) Similarity based user identification method are given. Table 7.4 also gives the identification accuracies obtained by the user identification methods: (1) Relative Measure based user identification method proposed by Bergadano et al. in
[31] and (2) Absolute Measure based user identification method proposed by Gunetti and Picardi in [32] over the six datasets. For visual inspection, the obtained identification accuracies by each of the four user identification methods on each dataset are plotted in Figure 7.3. For example, a dataset with label Dataset, represents that one typing sample is selected in the training data of each user in the dataset.

Table 7.4 Comparison between the identification accuracies obtained by four user identification methods on each of the six datasets.

<table>
<thead>
<tr>
<th>Dataset label</th>
<th>CNBM method</th>
<th>Similarity based method</th>
<th>Relative measure based method</th>
<th>Absolute measure based method</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dataset1</td>
<td>89.62%</td>
<td>96.33%</td>
<td>85.76%</td>
<td>80.71%</td>
</tr>
<tr>
<td>Dataset2</td>
<td>98.17%</td>
<td>99.26%</td>
<td>94.14%</td>
<td>86.85%</td>
</tr>
<tr>
<td>Dataset3</td>
<td>99.29%</td>
<td>99.67%</td>
<td>96.70%</td>
<td>88.88%</td>
</tr>
<tr>
<td>Dataset12</td>
<td>99.65%</td>
<td>99.99%</td>
<td>99.74%</td>
<td>91.31%</td>
</tr>
<tr>
<td>Dataset13</td>
<td>99.62%</td>
<td>100.00%</td>
<td>99.86%</td>
<td>91.43%</td>
</tr>
<tr>
<td>Dataset14</td>
<td>99.33%</td>
<td>100.00%</td>
<td>100.00%</td>
<td>92.00%</td>
</tr>
</tbody>
</table>

Figure 7.3 Comparison between the identification accuracies obtained by four user identification methods on each of the six datasets.
The obtained identification accuracies by each of the four methods are compared on each dataset distinctly as follows.

We can see in Figure 7.3 that our proposed Similarity based user identification method has attained the highest identification accuracy on Dataset, i.e., when one typing sample is selected in the training data of each user among the methods under comparison. Also, we can see in Figure 7.4 that our proposed CNBM method has attained higher identification accuracy than that obtained by the Relative Measure based user identification method proposed by Bergadano et al. in [31] and by the Absolute Measure based user identification method proposed by Gunetti and Picardi in [32] on Dataset. More specifically, the identification accuracy obtained by the CNBM on Dataset, is 3.86% higher and 8.91% higher than that obtained by the Relative Measure based user identification method and the Absolute Measure based user identification method on Dataset, respectively. We can also see in Figure 7.4 and in Table 7.3 that both of our proposed two user identification methods have attained higher identification accuracy than that obtained by the other two user identification methods on Dataset, i.e., when two typing samples are selected in the training data of each user.

We can also see in Table 7.3 that Similarity based user identification method has attained the highest identification accuracy of 99.67% on Dataset, than that obtained by the other three methods. Absolute Measure based user identification method has attained the least identification accuracy of 88.88% on Dataset, which is 10.79% and 10.41% lower than that obtained by the Similarity based user identification and CNBM method, respectively.
On Dataset$_{12}$ and on Dataset$_{13}$, the identification accuracy obtained by the Similarity based user identification method is the highest as compared to the identification accuracy obtained by the other three methods. We note that the identification accuracy obtained by the CNBM method on Dataset$_{12}$ is 0.09% less and on Dataset$_{13}$ is 0.24% less than that obtained by the Relative Measure based user identification method on Dataset$_{12}$ and on Dataset$_{13}$. We can also see in Table 7.4 that both Similarity based user identification method and Relative Measure based user identification method has attained 100% identification accuracy on Dataset$_{14}$.

From this comparison, we may conclude that (1) our proposed Similarity based user identification method has attained the highest identification accuracy, among the other three methods in comparison, over all the six datasets and attained 100% identification accuracy when the number of typing samples selected in the training data of each user are thirteen or fourteen; (2) the Relative measure based user identification method proposed in [31] has attained the 100% identification accuracy only when the number of typing samples selected in the training data of each user are fourteen; and (3) the Absolute measure based user identification method proposed in [32] has attained the lowest identification accuracy among the other three methods over all the six datasets.

In the next chapter, we fuse the identification decision obtained by each of the four user identification methods presented in this section to further enhance the identification accuracy of the user identification system.
CHAPTER 8

MAJORITY VOTING RULE BASED FUSION

The primary objective of designing a user identification system is to achieve the best possible identification accuracy. Therefore, the selection of one method from many competing methods to build user identification system is performed through empirical evaluation, i.e., the method with the highest identification accuracy is typically selected. However in the pattern recognition literature, several studies have demonstrated that although one method (classifier) would present the best pattern recognition results, the misclassification of the patterns by the different classifiers would not necessarily match [70, 71]. In other words, fusing two or more classifiers may yield better recognition results than that obtained by single classifier.

In this chapter, we analyze Majority Voting Rule (MVR) for fusing two or more user identification methods. A brief description of the MVR follows.

8.1 Introduction to the Majority Voting Rule Based Fusion

Let us suppose, total $n$ classifiers are designed for solving a pattern recognition problem and each classifier given an input pattern produces a unique decision regarding the identity of the input pattern. MVR assigns the input pattern to the class when at least $k$ classifiers are agreed on the identity, where the value of $k$ is determined using the following equation [70, 72-75]:
The provided decision by each of the \( n \) classifiers can be either correct or wrong and the decision provided by the MVR is wrong, if at least \( k \) classifiers make a wrong decision regarding the identity of the pattern. If the recognition rate, i.e., the probability that the decision provided by the classifier might be correct is known, then it is easy to calculate the probability value that the decision provided by the classifier might be wrong. For example, if the recognition rate of a classifier is \( p \), then the probability that the decision provided by the classifier might be wrong will be \( 1 - p \).

The probability that the decision provided by the fusion of \( n \) classifiers using MVR, say \( P_{\text{MVR}}(n) \), is correct can be estimated using the recognition rate of each of the \( n \) classifiers. In many studies, such as [72, 75], the value of \( P_{\text{MVR}}(n) \) has been estimated under the following two assumptions: (1) the recognition rate of each of the \( n \) classifiers is the same and (2) all the \( n \) classifiers are independent of each other. If the above two assumptions are satisfied, then \( P_{\text{MVR}}(n) \) can be estimated using the following equation [72, 75]:

\[
P_{\text{MVR}}(n) = \sum_{m=k}^{n} \binom{n}{m} p^m (1-p)^{n-m},
\]

where \( p \) is the recognition rate of each of the \( n \) classifiers and the value of \( k \) is determined using the equation 8.1. The following example illustrates the procedure for estimating the recognition rate of the MVR based fusion of three classifiers using equation 8.2.

\[
k = \begin{cases} 
\frac{n+1}{2}, & \text{if } n \text{ is even} \\
\frac{n+1}{2}, & \text{if } n \text{ is odd}.
\end{cases}
\]
Example 8.1: Let us suppose, three independent classifiers i.e., $n = 3$ has the same recognition rate $p = 0.6$ are to be fused using MVR. In this case, the value of $k = \frac{n + 1}{2} = \frac{3 + 1}{2} = 2$, and substituting $n = 3$, $k = 2$, and $p = 0.6$ in the equation 8.2, we get

$P_{\text{MVR}}(3) = \sum_{m=1}^{3} \binom{3}{m} 0.6^m (1 - 0.6)^{3-m} = 0.648$. Therefore, we may conclude that the fusion of these three classifiers using MVR might be beneficial as the recognition rate of the fusion is estimated to be higher than that obtained using individual classifier.

The assumption of each classifier having the same recognition rate cannot be expected to be always true in practice. If all the $n$ classifiers do not have the same recognition rate, then the equation 8.2 cannot be used for estimating the recognition rate of the fusion of $n$ classifiers using MVR. According to our knowledge, not much research has been carried out in theoretically estimating the recognition rate of the fusion of $n$ classifiers using MVR when the classifiers to be fused do not have the identical recognition rate. (We note that Lam and Suen in [72] have given some conditions when it is useful to add one or two classifiers to the ensemble of classifiers for MVR based classifier fusion.)

The next section presents our work on theoretically estimating the recognition rate of the fusion of classifiers using MVR irrespective of whether the classifiers to be fused have the identical recognition rate or not.
8.2 Recognition Rate Estimation

In this section, first, we theoretically estimate the recognition rate of the MVR based fusion of two classifiers, three classifiers, and four classifiers. Then, based on the findings, we estimate the recognition rate of the MVR based fusion of \( n \) classifiers.

8.2.1 Fusion of Two Classifiers

Let us suppose two classifiers \( C_1 \) and \( C_2 \) are to be fused using MVR. Let the recognition rates of the two classifiers \( C_1 \) and \( C_2 \) be \( p_1 \) and \( p_2 \), respectively. Given an input pattern, each classifier can make either a correct or wrong decision regarding the identity of the pattern. Therefore, for an input pattern, the decisions provided by the classifiers \( C_1 \) and \( C_2 \) will fall into one of the possible \( 2^2 = 4 \) decision vectors. By “decision vector” we mean a vector consisting of two components: (1) a decision provided by \( C_1 \) and (2) a decision provided by \( C_2 \). The possible four decision vectors when two classifiers are fused are given in the third column of the Table 8.1.

<table>
<thead>
<tr>
<th>( C_1 )</th>
<th>( C_2 )</th>
<th>Decision vector</th>
<th>( MVR(C_1, C_2) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>&lt;1,1&gt;</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>&lt;1,0&gt;</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>&lt;0,1&gt;</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>&lt;0,0&gt;</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 8.1 When two classifiers are fused using MVR, four decision vectors are possible given an input pattern.
In Table 8.1, note: (1) number “1” represents that the correct decision is provided by the classifier; (2) number “0” represents that the wrong decision is provided by the classifier; and (3) $MVR(C_1, C_2)$ represents the decision provided by fusing the decisions received from two classifiers using MVR. We can see in Table 8.1 that the fusion of two classifiers using MVR makes correct decision only on 1 decision vector $<1,1>$ and on the remaining decision vectors, MVR based fusion makes the wrong decision. Hence, decision vector $<1,1>$ is the only decision vector which is useful for estimating the recognition rate of the fusion of two classifiers using MVR. Subsequently, the recognition rate of the fusion of two classifiers using MVR, say $P_{MVR}(C_1, C_2)$, can be estimated as follows:

$$P_{MVR}(C_1, C_2) = P(C_1 = 1, C_2 = 1)$$  \hspace{1cm} \text{Equation 8.3}

Under the assumption that both the classifiers are independent of each other, equation 8.3 can be written as:

$$P_{MVR}(C_1, C_2) = P(C_1 = 1, C_2 = 1) = P(C_1 = 1) \cdot P(C_2 = 1)$$  \hspace{1cm} \text{Equation 8.4}

$$= p_1 \cdot p_2$$

From the equation 8.4, we may conclude that the recognition rate of the fusion of two classifiers using MVR will not be higher than that of the recognition rate of the individual classifier. This is because, $P_{MVR}(C_1, C_2) = p_1 \cdot p_2$ and the recognition rates $p_1$ and $p_2$ lies between 0 and 1. Mathematically, this relationship can be given as:

$$P_{MVR}(C_1, C_2) = p_1 \cdot p_2 \leq p_j, \hspace{0.5cm} j = 1 \text{ or } 2$$  \hspace{1cm} \text{Equation 8.5}
The following example illustrates the procedure for estimating the recognition rate of the MVR based fusion of two classifiers using equation 8.4.

**Example 8.2:** Let us suppose, two independent classifiers $C_1$ and $C_2$ are to be fused using MVR and the recognition rates of $C_1$ and $C_2$ be 0.7 and 0.8, respectively. From equation 8.4, the recognition rate of the fusion of these two classifiers using MVR, $P_{\text{MVR}}(C_1, C_2)$, can be estimated as: $P_{\text{MVR}}(C_1, C_2) = 0.7 \times 0.8 = 0.56$. Therefore, we may conclude that the fusion of these two classifiers using MVR might not be beneficial, as the recognition rate of the fusion is estimated to be lower than that obtained by the individual classifiers $C_1$ and $C_2$.

### 8.2.2 Fusion of Three Classifiers

Let us suppose, three classifiers $C_1$, $C_2$, and $C_3$ are to be fused using MVR. Let the recognition rates of the three classifiers $C_1$, $C_2$, and $C_3$ be $p_1$, $p_2$, and $p_3$, respectively. Given an input pattern, each of these three classifiers can make either a correct or wrong decision regarding the identity of the pattern. Therefore, for an input pattern, the decisions provided by $C_1$, $C_2$, and $C_3$ will fall into one of the possible $2^3 = 8$ decision vectors. By “decision vector” we mean a vector consisting of three components: (1) a decision provided by $C_1$, (2) a decision provided by $C_2$, and (3) a decision provided by $C_3$. The possible 8 decision vectors when three classifiers are fused are given in the fourth column of Table 8.2. In the Table 8.2, note the following: (1) number “1” represents that the correct decision is provided by the classifier; (2) number “0” represents that the wrong decision is provided by the classifier; and (3) $MVR(C_1, C_2, C_3)$
represents the decision provided by fusing the decisions received from the three classifiers using MVR.

Table 8.2 When three classifiers are fused using MVR, eight decision vectors are possible given an input pattern.

<table>
<thead>
<tr>
<th>$C_1$</th>
<th>$C_2$</th>
<th>$C_3$</th>
<th>Decision vector</th>
<th>$MVR(C_1, C_2, C_3)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>&lt;1,1,1&gt;</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>&lt;1,1,0&gt;</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>&lt;1,0,1&gt;</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>&lt;0,1,1&gt;</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>&lt;0,0,0&gt;</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>&lt;0,0,1&gt;</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>&lt;0,1,0&gt;</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>&lt;1,0,0&gt;</td>
<td>0</td>
</tr>
</tbody>
</table>

We can see in Table 8.2 that fusion of three classifiers using MVR makes a correct decision on four decision vectors $<1,1,1>$, $<1,1,0>$, $<1,0,1>$, and $<0,1,1>$. On the remaining four decision vectors, MVR based fusion makes the wrong decision. Hence, the four decision vectors mentioned above are useful for estimating the recognition rate of the fusion of three classifiers using MVR. Subsequently, the recognition rate of the fusion of three classifiers using MVR, say $P_{MVR}(C_1, C_2, C_3)$, can be estimated as follows:

$$
P_{MVR}(C_1, C_2, C_3) = P(C_1 = 1, C_2 = 1, C_3 = 1) + P(C_1 = 1, C_2 = 1, C_3 = 0) + P(C_1 = 1, C_2 = 0, C_3 = 1) + P(C_1 = 0, C_2 = 1, C_3 = 1)
$$

Equation 8.6
Under the assumption that all the three classifiers are independent of each other, equation 8.6 can be written as:

\[ P_{\text{MVR}}(C_1 > C_2 > C_i) = P_1P_2P_1 + P_1P_2(l - P_1)P_3 + (l - P_1)P_2P_3 \]  

Equation 8.7

Simplifying the equation 8.7, we have

\[ P_{\text{MVR}}(C_1 > C_2 > C_i) = (P_1P_2) + (P_1P_3) + (P_2P_3) - (P_1P_1P_2P_3) \]  

Equation 8.8

The following example illustrates the procedure for estimating the recognition rate of the MVR based fusion of three classifiers using equation 8.8.

**Example 8.3:** Let us suppose, three independent classifiers \( C_1, C_2, \) and \( C_3 \) are to be fused using MVR and the recognition rates of \( C_1, C_2, \) and \( C_3 \) be 0.7, 0.8, and 0.75, respectively. From equation 8.8, the recognition rate of the fusion of the three classifiers using MVR, \( P_{\text{MVR}}(C_1, C_2, C_3) \), can be estimated as: \( P_{\text{MVR}}(C_1, C_2, C_3) = 0.845 \). Therefore, we may conclude that the fusion of these three classifiers using MVR could be beneficial as the recognition rate of the fusion is estimated to be higher than that obtained by the individual classifiers \( C_1, C_2, \) and \( C_3 \).

### 8.2.3 Fusion of Four Classifiers

Let us suppose, four classifiers \( C_1, C_2, C_3, \) and \( C_4 \) are to be fused using MVR. Let the recognition rates of the four classifiers \( C_1, C_2, C_3, \) and \( C_4 \) be \( p_1, p_2, p_3, \) and \( p_4 \), respectively. Given an input pattern, each of these four classifiers can make either a correct or wrong decision regarding the identity of the pattern. Therefore, for an input pattern, the decisions provided by \( C_1, C_2, C_3, \) and \( C_4 \) will fall into one of the possible \( 2^4 = 16 \) decision vectors. By “decision vector” we mean a vector consisting of four
components: (1) the decision provided by $C_1$, (2) the decision provided by $C_2$, (3) the decision provided by $C_3$, and (4) the decision provided by $C_4$. The MVR based fusion of these four classifiers, $MVR(C_1, C_2, C_3, C_4)$, will make the correct decision if at least three of the classifiers have made correct decision regarding the identity of the pattern. This is because, in the case of fusion of four classifiers, the value of $k = \frac{n+1}{2} = \frac{4+1}{2} = 3$.

Therefore, the total number of decision vectors consisting of three or four correct decisions will be $\sum_{m=\frac{4}{2}}^{4} \binom{4}{m} = \binom{4}{3} + \binom{4}{4} = 10$. These five decision vectors are given in the fifth column of Table 8.3.

Table 8.3 The five decision vectors on which the MVR based fusion of four classifiers will make correct decision.

<table>
<thead>
<tr>
<th>$C_1$</th>
<th>$C_2$</th>
<th>$C_3$</th>
<th>$C_4$</th>
<th>Decision vector</th>
<th>$MVR(C_1, C_2, C_3, C_4)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>&lt;1,1,1,1&gt;</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>&lt;1,1,1,0&gt;</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>&lt;1,1,0,1&gt;</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>&lt;1,0,1,1&gt;</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>&lt;0,1,1,1&gt;</td>
<td>1</td>
</tr>
</tbody>
</table>

In Table 8.3, note the following: (1) number "1" represents that the correct decision is provided by the classifier; (2) number "0" represents that the wrong decision
(3) $MVR(C_1, C_2, C_3, C_4)$ represents the decision provided by fusing the decisions received from the four classifiers using MVR.

We can see in Table 8.3 that fusion of four classifiers using MVR makes a correct decision on the following five decision vectors $<1,1,1,1>$, $<1,1,1,0>$, $<1,1,0,1>$, $<1,0,1,1>$, and $<0,1,1,1>$. On the remaining possible decision vectors, MVR based fusion makes the wrong decision. Therefore, the five decision vectors mentioned above are useful for estimating the recognition rate of the fusion of four classifiers using MVR. Subsequently, the recognition rate of the fusion of four classifiers using MVR, say $P_{MVR}(C_1, C_2, C_3, C_4)$, can be estimated as follows:

\[
P_{MVR}(C_1, C_2, C_3, C_4) = P(C_1 = 1, C_2 = 1, C_3 = 1, C_4 = 1) + P(C_1 = 1, C_2 = 1, C_3 = 1, C_4 = 0) + P(C_1 = 1, C_2 = 1, C_3 = 0, C_4 = 1) + P(C_1 = 1, C_2 = 0, C_3 = 1, C_4 = 1) + P(C_1 = 0, C_2 = 1, C_3 = 1, C_4 = 1)
\]

Under the assumption that all the four classifiers are independent of each other, $P_{MVR}(C_1, C_2, C_3, C_4)$ can be written as:

\[
P_{MVR}(C_1, C_2, C_3, C_4) = (p_1p_2p_3p_4) + (p_1p_2p_2(1-p_3)) + (p_1p_2(1-p_2)p_3) + (p_1(1-p_2)p_3p_4) + ((1-p_2)p_3p_4) + (p_1p_2((1-p_2)p_3p_4) + (1-p_2))(1-p_2)(p_3p_4))
\]

Equation 8.9

The following example illustrates the procedure for estimating the recognition rate of the MVR based fusion of four classifiers using equation 8.9.

**Example 8.4:** Let us suppose, four independent classifiers $C_1$, $C_2$, $C_3$, and $C_4$ are to be fused using MVR. Let the recognition rates of $C_1$, $C_2$, $C_3$, and $C_4$ be 0.7, 0.8, 0.75, and 0.85, respectively. From the equation 8.9, the recognition rate of the fusion of the four classifiers using MVR, $P_{MVR}(C_1, C_2, C_3, C_4)$, can be estimated as:
Therefore, we may conclude that the fusion of these four classifiers using MVR might not be beneficial as the recognition rate of the fusion is estimated to be lower than that obtained by the individual classifiers \( C_3 \) and \( C_4 \).

Now, we compare the theoretically estimated recognition rate for the MVR based fusion of three classifiers with that for the fusion of four classifiers. Let us assume that after the addition of the fourth classifier i.e., \( C_4 \) to the ensemble of three classifiers, the recognition rate of the MVR based fusion increases. Therefore, we have

\[
P_{\text{MVR}}(C_1, C_2, C_3) > P_{\text{MVR}}(C_1, C_2, C_3, C_4)
\]

Equation 8.10

From equations 8.7 and 8.9, we have

\[
P_{\text{MVR}}(C_1, C_2, C_3) = p_1 p_2 p_3 + p_1 (1-p_1) p_2 + (1-p_1) p_3
\]

\[
P_{\text{MVR}}(C_1, C_2, C_3, C_4) = p_1 p_2 p_3 + p_1 (1-p_1) p_2 + (1-p_1) p_3 + (1-p_1) p_4 p_5
\]

Subtracting \( p_1 p_2 p_3 \) from the above two equations, we have

\[
p_1 (p_1 (1-p_1) + p_1 (1-p_1) p_2 + (1-p_1) p_3, p_5) > p_1 p_2 (1-p_1) + (1-p_1) p_3 + (1-p_1) p_4 p_5
\]

Assume \( p_1, p_2, (1-p_1) + (1-p_1) p_2, (1-p_1) p_3 + (1-p_1) p_4, p_5 = 0 \) and dividing both the sides by this term, we have

\[
p_4 > 1
\]

Equation 8.11

Therefore, \( P_{\text{MVR}}(C_1, C_2, C_3, C_4) > P_{\text{MVR}}(C_1, C_2, C_3) \) will be satisfied if, and only if, \( p_4 > 1 \).

However, \( p_4 \), being the recognition rate, can have value ranging from 0 to 1, but not greater than 1. Therefore, we may conclude that the recognition rate obtained after adding one more classifier to the ensemble of three classifiers will not be higher than that of the ensemble of the same three classifiers.
8.2.4 Fusion of 'n' Classifiers

Let us suppose, $n$ classifiers $C_1, C_2, \ldots, C_n$ are to be fused using MVR. Let the recognition rates of the classifiers $C_1, C_2, \ldots, C_n$ be $p_1, p_2, \ldots, p_n$, respectively.

Given an input pattern, each of these $n$ classifiers can make either a correct or wrong decision regarding the identity of the pattern. Therefore for an input pattern, the decisions provided by each of the classifiers will fall into one of the possible $2^n$ decision vectors. The MVR based fusion of these $n$ classifiers, $MVR(C_1, C_2, \ldots, C_n)$, will make a correct decision if at least $k$ of the classifiers have made a correct decision regarding the identity of the pattern where the value of $k$ is determined using the equation 8.1. The total number of decision vectors consisting of at least $k$ correct decisions will be

$$\sum_{m=k}^{n} \binom{n}{m} + \binom{n}{k+1} + \cdots + \binom{n}{n}.$$  

Let a set of decision vectors be $D = \{D_1, D_2, \ldots, D_k\}$ where (1) $N$ represents the total number of decision vectors on which $MVR(C_1, C_2, \ldots, C_n)$ makes correct decision, i.e., $N = \sum_{m=k}^{n} \binom{n}{m}$; and (2) each $D_i = \{d'_1, d'_2, \ldots, d'_n\}$ where each $d'_j$ represents the decision provided by a $j^{th}$ classifier in the decision vector $D_i$. The decision provided by a classifier can be either correct or wrong. Hence, $d'_j$ can have value either “1” or “0”, where value “1” represents that the correct decision is provided by a $j^{th}$ classifier and value “0” represents that the wrong decision is provided by a $j^{th}$ classifier in the $i^{th}$ decision vector. Consequently, the recognition rate of the MVR based fusion of $n$ classifier, say $P_{\text{MVR}}(C_1, C_2, \ldots, C_n)$, can be given as:
In the next section, we analyze the estimated identification accuracy of the MVR based fusion of (1) the CNBM, (2) the similarity based user identification method, (3) the relative measure based user identification method proposed by Bergadano et al. in [31], and (4) the absolute measure based user identification method proposed by Gunetti and Picardi in [32] in the light of empirical results.

8.3 Empirical Results

In Table 8.4, we present the identification accuracies obtained on each of the six datasets by four methods: (1) CNBM (referred as "C"), (2) Similarity based user identification method (referred as "C_2"), (3) Relative measure based user identification method proposed by Bergadano et al. in [31] (referred as "C_3"), and (4) Absolute measure based user identification method proposed by Gunetti and Picardi in [32] (referred as "C_4"). Table 8.4 also gives the theoretically estimated identification accuracies when (1) the methods C_1, C_2, C_3, and C_4 are fused using MVR, (2) the methods C_1, C_2, and C_3 are fused using MVR, and (3) the methods C_1, C_2, and C_4 are fused using MVR. The identification accuracies of $MVR(C_1, C_2, C_3)$ and $MVR(C_1, C_2, C_4)$ are theoretically estimated using equation 8.8 given in Section 8.2.2. And, the identification accuracy of $MVR(C_1, C_2, C_3, C_4)$ is theoretically estimated using equation 8.9 given in Section 8.2.3.

\[ P_{mn}(C_1, C_2, \ldots, C_s) = \sum \prod_{j=1}^{s} \chi(d'_j), \]  
\[ \chi(d'_j) = \begin{cases} p_i, & d'_j = 1 \\ 1 - p_i, & d'_j = 0 \end{cases} \]  
Equation 8.12
Table 8.4 Theoretical estimation of the identification accuracies when the user identification methods are fused using majority voting rule.

<table>
<thead>
<tr>
<th>Method</th>
<th>Dataset_1</th>
<th>Dataset_2</th>
<th>Dataset_3</th>
<th>Dataset_4</th>
<th>Dataset_5</th>
<th>Dataset_6</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C_1$</td>
<td>89.62%</td>
<td>98.17%</td>
<td>99.29%</td>
<td>99.65%</td>
<td>99.62%</td>
<td>99.33%</td>
</tr>
<tr>
<td>$C_2$</td>
<td>96.33%</td>
<td>99.26%</td>
<td>99.67%</td>
<td>99.99%</td>
<td>100.00%</td>
<td>100.00%</td>
</tr>
<tr>
<td>$C_3$</td>
<td>85.76%</td>
<td>94.14%</td>
<td>96.70%</td>
<td>99.74%</td>
<td>99.86%</td>
<td>100.00%</td>
</tr>
<tr>
<td>$C_4$</td>
<td>80.71%</td>
<td>86.85%</td>
<td>88.88%</td>
<td>91.31%</td>
<td>91.43%</td>
<td>92.00%</td>
</tr>
<tr>
<td>$MVR(C_1, C_2, C_3)$</td>
<td>93.16%</td>
<td>98.77%</td>
<td>99.49%</td>
<td>99.94%</td>
<td>99.95%</td>
<td></td>
</tr>
<tr>
<td>$MVR(C_1, C_2, C_3, C_4)$</td>
<td>97.73%</td>
<td>99.84%</td>
<td>99.96%</td>
<td>100%</td>
<td>100%</td>
<td></td>
</tr>
<tr>
<td>$MVR(C_1, C_2, C_3)$</td>
<td>97.06%</td>
<td>99.65%</td>
<td>99.88%</td>
<td>99.97%</td>
<td>99.97%</td>
<td></td>
</tr>
<tr>
<td>$MVR(C_1, C_2, C_3)$</td>
<td>99.95%</td>
<td>99.95%</td>
<td>99.95%</td>
<td>99.95%</td>
<td>99.95%</td>
<td></td>
</tr>
</tbody>
</table>

$C_1$: CNBM method  
$C_2$: Similarity based user identification method  
$C_3$: Relative measure based user identification method  
$C_4$: Absolute measure based user identification method

To illustrate the procedure for estimating the identification accuracy of the MVR based fusion of user identification methods, we estimate the identification accuracy of $MVR(C_1, C_2, C_3)$ and $MVR(C_1, C_2, C_3, C_4)$ on a dataset in the following two examples.

**Example 8.5:** On “Dataset_1”, identification accuracy of $MVR(C_1, C_2, C_3)$ is estimated as follows:

\[
P_{\text{est}}(C_1, C_2, C_3) = p_{c1}p_{c2} + p_{c1}p_{c3} + p_{c2}p_{c3} - 2p_{c1}p_{c2}p_{c3}
\]

\[
= (0.8962 \times 0.9633) + (0.8962 \times 0.8576) + (0.9633 \times 0.8576) - 2 \times 0.8962 \times 0.9633 \times 0.8576
\]

\[
= 0.9773 = 97.73\%
\]
Example 8.6: On “Dataset1”, identification accuracy of $MVR(C_1, C_2, C_3, C_4)$ is estimated as follows:

$$P_{MVR}(C_1, C_2, C_3, C_4) = p_1 p_2 p_3 (1 - p_4) + p_1 (1 - p_3) p_4 + (1 - p_1) p_2 p_3$$

$$= 0.8962 \times 0.9633 \times 0.8576 + 0.8962 \times (1 - 0.9633) \times 0.8576$$

$$= 0.9316 = 93.16\%$$

Table 8.5 gives the obtained identification accuracy of the MVR based fusion of (1) $C_1$, $C_2$, $C_3$, and $C_4$; (2) $C_1$, $C_2$, and $C_4$; and (3) $C_1$, $C_2$, $C_3$, and $C_4$ on the six datasets.

Table 8.5 Obtained identification accuracies on the six datasets when the user identification methods are fused using majority voting rule.

<table>
<thead>
<tr>
<th>Method</th>
<th>Dataset1</th>
<th>Dataset2</th>
<th>Dataset3</th>
<th>Dataset12</th>
<th>Dataset13</th>
<th>Dataset14</th>
</tr>
</thead>
<tbody>
<tr>
<td>$MVR(C_1, C_2, C_3, C_4)$</td>
<td>90.09%</td>
<td>97.33%</td>
<td>98.81%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>$MVR(C_1, C_2, C_4)$</td>
<td>96.52%</td>
<td>99.40%</td>
<td>99.72%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>$MVR(C_1, C_2, C_3)$</td>
<td>95.62%</td>
<td>99.20%</td>
<td>99.66%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
</tr>
</tbody>
</table>

$C_1$: CNBM method  
$C_2$: Similarity based user identification method  
$C_3$: Relative measure based user identification method  
$C_4$: Absolute measure based user identification method

We can see in Table 8.4 and Table 8.5: (1) the identification accuracy of the MVR based fusion of three user identification methods i.e., $MVR(C_1, C_2, C_3)$ and $MVR(C_1, C_2, C_4)$ is higher than that of the MVR based fusion of four user identification
methods $MVR(C_1, C_2, C_3, C_4)$ on all the six datasets. In other words, the identification accuracy of the MVR based fusion of four user identification methods is less than that of the MVR based fusion of three user identification methods on all the six datasets, which is theoretically observed in Section 8.2.3 (refer equation 8.11 and the last paragraph of Section 8.2.3); (2) the identification accuracy of $MVR(C_1, C_2, C_3)$ is higher than, or at least the same, that of the other two fusions $MVR(C_1, C_2, C_4)$ and $MVR(C_1, C_3, C_4)$ on all six datasets; and (3) the identification accuracy of the $MVR(C_1, C_2, C_3)$ is higher than, or at least the same as, that of the individual user identification methods $C_1$, $C_2$, and $C_3$ on all the six datasets.
CHAPTER 9

CONCLUSIONS

In this dissertation, we proposed two methods for identifying computer users using keystroke patterns: (1) Competition between naïve Bayes models (CNBM) and (2) Similarity based user identification method. In the CNBM method, for each user we first determine the probabilistic likelihood that the typed text belongs to a user and then the typed text is assigned to the user with the highest likelihood value. In the Similarity based user identification method, we assign a similarity score to each user given a typed text. The assigned similarity score to a user is determined by finding the ratio between (1) the number of key press latency values extracted from the typed text similar to the estimated model parameters of the user and (2) the total number of key press latency values extracted from the typed text. Finally, the typed text is assigned to the user with the highest similarity score. We also present a novel application of distance based outlier detection method for detecting outlying values that may be present in the keystroke data. More specifically, outliers in the keystroke data are detected using the following three-step procedure: (1) for each extracted key press latency value $x_i$, a neighborhood region using a distance threshold is created; (2) a key press latency value $x_j$ is considered as a neighbor of $x_i$, if $x_j$ falls in the neighborhood region of $x_i$; and (3) finally, the latency
value \( x_i \) is considered as an outlying value if the number of neighbors determined for \( x_i \) are less than some pre-set threshold.

In the pattern recognition literature, several studies have demonstrated that fusing two or more classifiers may yield better recognition results than that obtained by single classifier. On the basis of this motivation to further improve the performance of the user identification system, we have theoretically analyzed Majority Voting Rule (MVR) based fusion of two or more user identification methods. We formulated a procedure for theoretically estimating the identification accuracy of the MVR based fusion of user identification methods. Our proposed procedure, unlike the procedure presented in the literature of MVR based fusion, does not assume that the methods to be fused have the identical identification accuracy.

To empirically evaluate the performance of our proposed work, a keystroke data set was collected from ten users, where each user provided 15 typing samples. While collecting typing samples, the following experimental settings were used: (1) samples were created in such a way that entire text of any sample did not exactly match with that of any other sample; (2) the samples had lengths varying between 850 and 972 characters; (3) users were instructed to provide samples at different times of the day (like one sample in the morning, the next sample in the afternoon); and (4) users were allowed to make typing error(s), and we have not discarded any sample provided by the users for any reason. From the provided typing samples, six distinct datasets were created. The datasets were created in such a way that the number of typing samples selected in the training data of a dataset does not exactly match with that in the training data of any other dataset. The performance of the CNBM method and the Similarity based user
identification method was demonstrated on these six datasets. Further, the performance of the CNBM method and the Similarity based user identification method was compared with the performance of the Relative Measure based user identification method proposed by Bergadano et al. in [31] and the Absolute Measure based user identification method proposed by Gunetti and Picardi in [32]. Our empirical evaluations showed that:

1) The distance based outlier detection method detected about 20% of keystroke data as outliers in each of the six datasets;

2) The identification accuracy of the CNBM method has improved by 42.16%, on average, on each dataset when the detected outliers by the proposed outlier detection method were discarded;

3) The identification accuracy of the CNBM method has shown improvement with an increase in the number of typing samples selected for training the model of a user. The highest identification accuracy of the CNBM method of 99.65% is observed on a dataset when twelve typing samples were selected for training the model of a user;

4) The identification accuracy of the Similarity based user identification method has improved by 42.53%, on average, on each dataset when the detected outliers by the proposed outlier detection method were discarded;

5) The identification accuracy of the Similarity based user identification method has shown improvement with an increase in the number of typing samples selected for training the model of a user. The highest identification accuracy of the method of 100% is observed on two datasets when thirteen and fourteen typing samples were selected for training the model of a user;
6) The Similarity based user identification method outperforms the CNBM method, Bergadano et al.’s Relative Measure based user identification method, and Gunneti and Picardi’s Absolute Measure based user identification method in terms of identification accuracy over all the six datasets;

7) The MVR based fusion of the Similarity based user identification method, the CNBM method, and the Relative Measure based user identification method outperforms (1) each of the individual four user identification methods, (2) the MVR based fusion of all the four user identification methods, and (3) the MVR based fusion of the Similarity based user identification method, the CNBM method, and the Absolute Measure based user identification method in terms of identification accuracy over all the six datasets.

Our future work could include evaluating the performance of the CNBM method and the Similarity based user identification method in the applications such as email and chat.
REFERENCES


