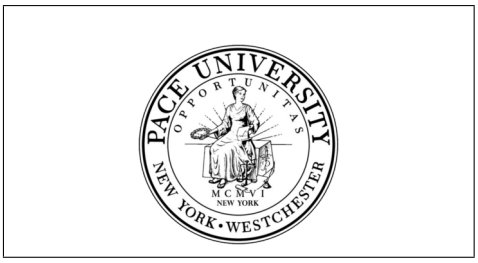
Thesis to get the degree of a master of computer science

Behavioral Biometric Authentication in Human-Computer Interaction

John Vincent Monaco

January 2014



Pace University
Seidenberg School of Computer Science and Information Systems

Dean
Dr. Amar Gupta

Referees
Dr. Charles Tappert

Contents

Abstract

The goal of this thesis is to present a model for authenticating users via interaction with a computer. In particular, the sequence of events generated when a user moves a mouse or types on a keyboard are considered. Such sequences can be observed when a user interacts with a standard desktop or laptop computer. Being able to authenticate a user by the interaction that occurs with an application can allow for a robust user-friendly identity management solution.

1. Introduction

As humans and machines become more tightly integrated it is important for the machine to be aware of who it is interacting with. Acknowledging the identity of the operator allows the machine to act in a more intelligent way and is the first step to responding appropriately to the user's actions. It may grant or deny access to certain portions of a program, adapt to the user's behavior, or account for and correct user errors.

The problem of confirming the identity of a person is not so simple. Humans perform this function effortlessly, making judgments based on inputs to nearly every sense. A person may be recognized by a combination of their appearance, body language, voice, and smell. It is even possible to recognize someone after their appearance has changed drastically or years later after their voice and body have aged. The temporal aspect of confirming someone's identity is just another challenge that a machine must account for.

1.1. Authentication

The need for a consistent and accurate means of authentication is apparent as people place more trust in machines. Authentication should be measurably resistant to threats [42]. Biometrics offers a way of authentication in which a confidence can be placed, since the expected number of false acceptances and false rejects can be estimated. Behavioral biometrics are harder to capture and reproduce than physiological biometrics although they are also more difficult quantify.

1.1.1. Types of threats

A biometric authentication system may be interacted with in many different ways[36].

Positive claim of identity The user claims to be known by the system

Negative claim of identity The user claims to not be known by the system

Genuine claim of identity The claimed identity is authentic

Impostor claim of identity The user makes a false claim of identity, pretending to be someone else

We are only interested in positive claims of identity scenarios, where a user makes either a genuine or impostor claim of identity. In our system, a negative claim of identity would deny access to the user by default. All simulated attacks are *zero-effort attacks*, in contrast to a *statistical attack*, which exploits the probable range of feature values to gain access to a BAS.

1.2. Behavioral biometrics

Biometric authentication has now become mainstream as the techniques have matured over the past several decades. It is used in law enforcement, border control, and personal device access. Simply put, it is the recognition of a human by some characteristic. There are many different biometrics that can be measured, and new ones are still being proposed. Various biometric measures may be considered either physiological or behavioral. This classification is not strict and some biometrics may fall into either category. A better ontology would be to consider a biometric as being either static or dynamic. A static biometric would be a fingerprint or a picture of someone's face. Dynamic biometrics change over time and include voice, This thesis deal with biometrics which are dynamic in nature.

Within dynamic biometrics, another important distinction is made. Some biometrics involve human-computer interaction (HCI), such as typing or operating a mouse, while others do not (non-HCI), such as gait recognition and voice. This thesis is mainly concerned with dynamic biometrics which involve HCI, although a few non-HCI biometrics are investigated.

1.2.1. Human-computer interaction

Some behavioral biometrics require the interaction between a human and computer interface. Behavioral biometrics may be placed into two categories: those that modify the state of the environment (*active*) and those that leave the environment unchanged from the user's perspective (*passive*). Gait would be an example of behavior in a static environment, while typing and moving a mouse require interaction and produce feedback for the user.

1.2.2. Cognition

Various behavioral biometrics rely more or less heavily on cognition. In Figure 1.1, the keystroke, mouse, and stylometry biometrics are placed in a hierarchy based intuitively on the cognitive load each one induces. It will be shown that the biometrics operating at the top of the hierarchy are generally more difficult to quantify.

According to Newell's time scale in Figure 1.2, human behavior may fall into any one of 4 categories. Quantifying behavior in each category comes with a varying degree

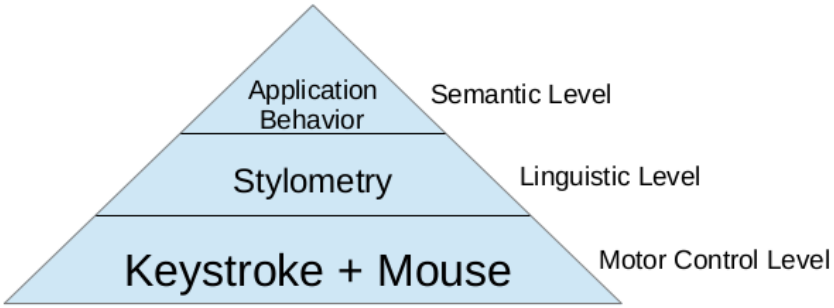


Figure 1.1.: Behavioral biometric hierarchy

of difficulty, and understanding behavior across all categories will involve analyzing sequences of events with different levels of granularity. A working model will likely require a hierarchical solution, with the understanding that behaviors at the lower levels form the base of a pyramid of behavioral biometrics. Human cognition lies somewhere in the middle of the pyramid, with physiological traits at the bottom and long term trends at the top. This suggests that better results may be achieved by focusing on the bottom on top of the pyramid, since action requiring a high level of cognition are difficult to predict.

Scale (sec)	Time Units	System	World (theory)
10^7	Months		SOCIAL BAND
10^6	Weeks		
10^5	Days		
10^4	Hours	Task	RATIONAL BAND
10^3	10 min	Task	
10^2	Minutes	Task	
10^1	10 sec	Unit task	COGNITIVE BAND
10^0	1 sec	Operations	
10^{-1}	100 ms	Deliberate act	
10^{-2}	10 ms	Neural circuit	BIOLOGICAL BAND
10^{-3}	1 ms	Neuron	
10^{-4}	100 µs	Organelle	

Figure 1.2.: Newell's Time Scale [41]

1.2.3. Sequence Classification

Behavioral biometrics makes heavy use of sequence classification techniques from machine learning. There are primarily 3 ways in which sequences are classified [55]:

- Feature-based (extract features, followed by conventional pattern recognition methods)
- Distance-based (using a distance function between sequences)
- Model-based (statistical models, such as hidden markov model)

This document is mostly concerned with the feature-based approach, and introduces a hybrid of feature and distance-based approaches.

1.2.4. Sequence source identification

A sequence S is comprised of events E which occur over time. Each event is instantaneous and occurs at time t belongs to a class C as defined by an identity function f which maps the events to the class space. An event may also have a vector of attributes $\mathbf{A}$ which describe the state of the system at the time of the event.

There are two types of features which may be used to identify the source of a sequence:

1. State features, which describe the attributes of an event class
2. Transition features, which are time dependent features that describe the transitioning from state $\{S1 \rightarrow S2 \rightarrow S3...\}$ in the sequence. Such features might include the velocity, acceleration, jerk, or higher order derivatives of the sequence.

The goal of authentication and identification in behavioral biometrics is identifying the source of the observed sequence. This is a subtle difference from many sequence classification tasks, since only the source of the sequence is of interest. The source is the user and the sequence consists of the events that occur during human-computer interaction. On a standard desktop, sequences will be made up of keystroke and mouse events. On a touchscreen mobile device, the events depend on the sensors available, and usually consist of touch, acceleration, sound, pressure, etc.

1.2.5. Authentication model evaluation

There are many facets to evaluating a biometric model. While we are interested mostly in the technical performance,[36] define other aspects which should be considered:

- Reliability, availability and maintainability
- Vulnerability
- Security

- User acceptance
- Human factors
- Cost/benefit
- Privacy regulation compliance

In several experiments, the individuality of users will be examined. It will be show that for some biometrics, the error rate for individual users is vastly different. This leads to the conclusion that a system which authenticates every user under the same parameters may not be optimal.

1.3. Structure of this document

With the exception of chapter 2, each chapter in the document is mostly self contained. The authentication model used in chapters 4, 5, and 6 is described in chapter 2. Chapter 3 covers the experimental data used in the experiments with some notes on the implementation of data collection software. Chapters 4 and 5 describe several studies and keystroke and stylometry biometrics, while chapter 6 include mouse and multimodal experimental results.

2. Experimental Data

2.1. Overview

A framework for data collection was developed and used to collect data from participants. The framework consists of three applications which can be used to collect data. A cross-platform application registers native operating system hooks and is used to log keyboard and mouse events in a system-wide context on standard desktop and laptop computers. This is able to capture input in any scenario on a desktop or laptop computer. A more practical web-based application was also developed. Implemented in JavaScript and run in a browser, the application only collects keyboard and mouse events which occur within the context of the page. This library can be initialized on any web page and is capable of recording any type of event the document might generate, including keystroke, mouse, touchscreen, and accelerometer events when the hardware is available. Finally, a mobile application was developed which replaced the soft keyboard on a mobile device and captures information in a system-wide context. Both applications send the data to a server where the user is authenticated by a session key to ensure data integrity during the collection phase.

Data from all applications was compared in order to determine whether there is any difference in the timing information of keystroke and mouse events and whether the two sources could be used in the same population.

The native Behavioral Biometrics Logger (native-BBL) is desktop application was developed with the purpose of collecting data from HCI on a standard desktop computer. The application records mouse movement, keystrokes, and textual information in a system-wide context. This application was used to collect several different sets of data.

(mobile-BBL)

A web-based behavioral Biometrics Logger (web-BBL) was developed to capture ...

A summary of each biometric dataset can be found in Table 2.3. An label is assigned to each database for future reference.

2.2. Keystroke and mouse

Most of the experimental keystroke and mouse data was collected on standard desktop computers running either Windows, OS X, or Linux with commodity hardware.

In this case, there are 5 different types of events which are considered. These include keystroke, stylometry, mouse motion, mouse click, and mouse scroll events. Unless otherwise stated, keystroke and mouse events are defined in Table 2.1:

Event	Attribute	Description
Keystroke	Time press	Timestamp of the key press
	Time release	Timestamp of the key release
	Key code	The key code (implementation-dependent)
	Key name	The name of the key (universal).
Stylometry	Time start	Beginning of the stylometry event
	Time end	End of the stylometry event
	Text	Text which was entered during the event
Mouse motion	Time	Instantaneous timestamp of the event
	Coordinates	Screen coordinates of the pointer
	Button	Which button (if any) was being held
Mouse click	Time press	Timestamp of the button press
	Time release	Timestamp of the button release
	Coordinates press	Screen coordinates at the time of the press
	Coordinates release	Screen coordinates at the time of the release
	Button	The button which was pressed
Mouse scroll	Time	Instantaneous time
	Coordinates	Screen coordinates at the time of the scroll
	Direction	The direction of the scroll apparatus
	Amount	The amount of scrolling which resulted (system dependent

Table 2.1.: Keystroke and mouse events

Timestamps for each event are usually provided with millisecond precision with an accuracy of .. depending on the system details. Mouse motion and scroll events are instantaneous, while keystroke, stylometry, and mouse click have a duration. This is merely a matter of convenience, since the keystroke, stylometry, and mouse click events may be represented as sequences of instantaneous events by “unstacking” the action associated with each timestamp. For example, the class of a keystroke event could be changed to a key-action pair, which occurs instantaneously.

The keycode of a keystroke event is not enough to identify the key when collecting data from heterogeneous systems. A key manager must be implemented (See Appendix) which is able to consistently classify key events from various systems to the corresponding physical key on the keyboard. The variation of keyboards in addition to operating systems and applications makes this task especially complex. In addition to this, it is known that variations in keyboard model, environmental conditions, and types of input may affect the accuracy of a biometric system which utilizes keystrokes[22, 53].

Stylometry events begin and end within a context in which the user's keystrokes result in text appearing on the screen. The context is bound by any other event which might alter the location of text being entered. This includes: mouse clicks, the occurrence of any non-printable key, changes in window focus caused by either a sequence of keystrokes or operating system command, and so on. The segmentation of stylometry events is still experimental. Its purpose is to segment stylometry events in a way which reflects the user's intentions when writing or modifying a document on a computer. These events may be considered to be operating at a higher cognitive level than keystroke or mouse.

2.2.1. Long free-text and fixed-text

In free-text keystroke entry, the sequence of keystrokes is unknown, as opposed to fixed-text entry where a known sequence is expected. In free-text, the sequence of keystrokes is not necessary arbitrary depending on the context and expected frequency of characters in the user's native language. Authentication via free-text generally requires a different approach than fixed text. Data were collected for both long free-text and long fixed-text, as well and short fixed-text described in the following section.

Free-text data was collected from 43 students, predominantly juniors and seniors, in two sections of a spreadsheet modeling course in the business school of a four-year liberal arts college. The classes met in a 20-seat desktop computer laboratory where the exams were administered. The 40 students took four online short-answer tests of 10 questions each, and the tests took place at approximately two week intervals. The students were unaware that their data were being captured for experimental analysis. Data from students not completing all four tests or having problems with the input system were removed, resulting in complete data sets from 30 students, 17 male and 13 female. The text lengths of the answers to a test ranged from 433 to 1831 words per test, with a mean of 966 and a median of 915 words. An average word length of five characters (six with spaces between words) yields roughly 6000 keystrokes per test. All the tests were taken on classroom Dell desktop computers with associated Dell keyboards.

In addition to this, an existing dataset containing both long fixed and free-text is used [53]. The data was collected from a total of 144 users primarily on Dell desktop

and laptop computers in a classroom setting. The fixed-text data was collected from users who were instructed to copy a 652-character paragraph. The free-text was collected from users who typed arbitrary emails of at least 650 keystrokes. Users were instructed to correct error before ending each session. Several smaller datasets were constructed to create users with equal number of free-text samples, resulting in population sizes of 14, 30, and 119. The average sample length for the 119 dataset is 755 keystrokes.

2.2.2. Short Fixed keystroke

2.2.2.1. Keypad

The numeric keypad samples were captured using an open-source third-party keylogger designed by Fimbel . This keylogger was originally developed for testing purposes and serves no malicious intent. It runs in the background of a computer system thus being unobtrusive to the user and allowing for the capture of keyboard and mouse input regardless of the application(s) running on a system. The numeric keypad data were collected from 30 subjects over a four-day period with no more than 60 samples collected per subject per day. Each subject first practiced keying the input string several times before the samples were recorded. Each sample consisted of the numeric sequence 914 193 7761 (shown here in telephone number format) followed by the Enter key to provide a total of 11 keystrokes per sample. All samples were entered with the right hand as if entering a phone number on a digital phone or entering an ATM pin, and only correct numeric-sequence samples were accepted.

2.2.2.2. Password

In addition to the keypad data collected, several publicly available password entry datasets are used.

The Keystroke Dynamics benchmark data set [31] contains the press and release times of keystrokes captured during password entry. A total 51 users were instructed to enter a password `(.tie5Roanl)` 50 times over 8 sessions, for a total of 400 entries. The password was selected to be representative of a string 10-charactor password, and only correct entries were kept. The keylogging application ran on a laptop with Windows XP and the timestamps are accurate to within $\pm 200\mu s$.

The GREYC-Keystroke benchmark dataset [20] contains password entry from 133 users, typing the password `(greyc laboratory)` over several months. Users were instructed to record 1 or 2 sessions per week and to practice typing the password before enrolling data in the system. Users were also instructed to data entry between 4 different keyboards.

The Keystroke 100 dataset [35, 34] contains password entry collected on a pressure-sensitive keyboard from 100 users. Ten samples were collected from each user entering the password `(try4-mbs)`, having practiced typing it beforehand. For each sample, the latency between key press timestamps was recorded. The raw press and release timestamps of each key are not available. A time series of the pressure exerted on each key was also recorded with a pressure-sensitive keyboard.

2.2.3. Multimodal tasks

Four different scenarios were developed to collect multimodal data from 50 users. Keystrokes, mouse motion, click, and scroll events were recorded. Six edit scenarios, six browser scenarios, 2 game scenarios, and 4 online quizzes were created for users to complete. Graduate students from two computer science and information technology courses were instructed to complete the scenarios, although not every student completed every task. For every scenario except the online quizzes, students were required to start the native-BBL to capture data in a system-wide context. The steps to start the logger are:

1. Login (create account if needed) to the native-BBL launch web page
2. Select the type of scenario (edit paragraph/web Search/game) and click on Launch. This will launch logger window.
3. Minimize the logger window.
4. Follow the instruction for the scenario, provided on a different web page
5. After completing the experiment open the logger window and click on exit to save the log, ending the session.

For the online quiz scenarios, the web-based BBL was used and students were not aware of their data being captured.

2.2.3.1. Edit Tasks

Edit or Modify tasks are the typical activities performed by computer users. The tasks were designed to induce a significant cognitive load, as a section of text must be carefully read and modified to match the given text. This requires hand-eye coordination and manipulation of the mouse and/or the keyboard. Six edit scenarios were prepared. Students were presented with a portion of text which they must edit to match another non-editable portion of text on the screen. A typical sample edit scenario is listed below. The fixed text is what the student must modify the given text to. The after text highlights the changes that must be made:

Fixed Koobface is a multi-platform computer worm that spreads primarily through social networking sites. Its name is an anagram of Facebook. The worm targets Web users with invitations to watch a video. Those curious enough to

click the link get a message to update their computer's software, which begins the download of the malware. Victims' computers are drafted into a peer-to-peer botnet and are sent official -looking advertisements of fake antivirus software. Further, their Web searches are hijacked and the clicks delivered to unscrupulous marketers. The "Koobface gang" made money from people who bought the bogus software and from unsuspecting advertisers

Given Koobface is a computer worm that spreads through social networking sites. Its name is an anagram for Facebook. The worm aims at web users with invitations to watch a video. Those who click the link get a message to update their computer's software, which begins the download of the Koobface malware. Victim's computers are drafted into a botnet and are sent official looking advertisements of fake antivirus software. Further, their Web searches are hijacked and the clicks delivered to marketers. The group made money from people who bought the bogus software and from advertisers.

After Koobface is a multi-platform computer worm that spreads primarily through social networking sites. Its name is an anagram for Facebook. The worm ~~aims~~ targets ~~at~~ ~~w~~Web users with invitations to watch a video. Those curious enough to ~~who~~ click the link get a message to update their computer's software, which begins the download of the ~~Koobface~~ malware. Victim's computers are drafted into a peer-to-peer botnet and are sent official looking advertisements of fake antivirus software. Further, their Web searches are hijacked and the clicks delivered to unscrupulous marketers. The "Koobface gang" group made money from people who bought the bogus software and from unsuspecting advertisers.

The six edit task were designed according to the following ontology:

Light copy editing In this type, there will be very minimal changes to the document (Before and After). These are generally performed at an editorial or review level where there's more focus to check on grammar rules.

Minor-Minor editing Minor editing involves changes, which do not alter/modify the meaning of the document. It may vary from merely spell checks to correcting typos. Again, they also contribute to minimal changes in the document but it's slightly heavier that a light copy edit.

Moderate copy editing A moderate copy edit can lead to substantial changes in a document. Typical components would involve table of contents entries, text to diagram relationships and structural (heading) reorganizations. It will also involve add/modify/replace of words or small paragraphs in a document.

Heavy copy editing Comparing heavy to medium copy editing the primary difference would be the rewriting or changes in the document. Heavy copy editing would involve changes to the base meaning of the document like active-passive voice conversion, introducing a formal writing approach etc., This will bring in major changes in the document and can sometimes lead to rewrite even though it's pretty rare.

Major changes In this scenario, there can be major changes in the document primarily leading to alter the meaning of the same. Majority of such scenarios will involve complete document rewrite or starting from scratch approach. This method is a rarely used edit scenario unless the prepared document is almost unfit to publish.

2.2.3.2. Browser Tasks

Browsing scenarios were designed to induce a "typical" web browsing session. Users were given specific instructions on how to interact with the application and web pages. Six scenarios were prepared. A typical sample browser scenario is listed below:

1. Login to the native-BBL launch web page
2. Select the type of scenario (edit paragraph/web Search/game) and click on Launch. This will launch logger window.
3. Go to Yahoo: http://www.yahoo.com/
4. Click on Sports (left menu) MLB (top menu) Teams (top sub-menu) Boston Red Sox Team Report for the Boston Red Sox
5. Go back two pages
6. Click on New York Yankees Depth Chart for the New York Yankees
7. Click on Roster for the New York Yankees (next to "Depth Chart")
8. Click on Sabathia, CC (scroll if necessary) and in the search field above, type "New York Yankees Captain" and click Sport Search
9. Exit tab or browser
10. Open the logger window and click on exit to save the log, ending the session

2.2.3.3. Gaming Scenarios

Two gaming scenarios were implemented. The game scenarios were observed to require heavy interaction with the computer, compared to the edit, browser, and quiz tasks. Both games are primarily operated by the mouse, so little or no keystroke information was recorded during these sessions. The games used were Spider Solitaire and Star Bubbles . Both games were web based and run in a typical browser. Students were directed to follow these instructions:

1. Login to the native-BBL launch web page
2. Select "Game: Spider Solitaire" from the task menu and then click on "Launch"
3. Once the application is running, minimize the application window so it will not interfere with you playing the game

4. Click on the Game: Spider Solitaire link at the bottom of the page to access the game
5. Click on How to Play to review the rules of the game
6. When done, click on Back
7. Click on Play One Suit
8. Attempt to finish the game
9. Exit the BioLogger when you finish playing
10. Open the logger window and click on exit to save the log, ending the session

2.2.3.4. Online quiz

Four quizzes were prepared for students taking an online course. Three of the quizzes were peer evaluations and one was an introductory quiz. All students in the course were required to take the quizzes, although they were only graded for completion.

2.3. Mobile

As the hardware for mobile devices varies greatly, it is important to consider OS and device-specific capabilities when collecting data. The information available depends on the device sensors, operating system, and permissions of the application. This can lead to a rich set of attributes. Android allows to build custom text input services, including keyboards, which enables us to write a custom keyboard to collect the necessary user data. Besides these technical considerations, Android has 79% market share which makes it by far the most popular mobile platform. Therefore the decision was made to select Android as the main OS to implement the mobile-BBL.

The mobile-BBL replaces the system keyboard on devices running the Android operating system. In this way, it is able to capture events in a system wide context in any application. The soft keyboard is itself an application and is used heavily for interacting with most applications. Limiting events to interaction with on-screen buttons, the attributes for mobile touchscreen events are summarized in Table 2.2.

Attribute	Description
Time	Timestamp of the event
Action	Type of action (press, move, release)
Entity	The ID of the button or entity associated with this event
Keyboard	Keyboard layout in use (Latin, symbol, etc.)
Orientation	The screen orientation (portrait or landscape)
Coordinates	Screen coordinates where the event occurred (pixels)
Pressure	Pressure exerted on the screen (normalized)
Touch major/minor	Length of major/minor axis of touch as an ellipse (pixels)
Rotation	Rotation rate of the device (rad/s)
Acceleration	Acceleration of the devices (m/s^2)
Screen density	Density of the screen (dpi)
Screen size	Screen size (pixels)

Table 2.2.: Mobile event attributes

The class of each touchscreen events is uniquely defined by action and entity attributes and each event occurs instantaneously. The screen density and size can be used to get the physical location of the event on the device. The rotation and acceleration are capture from the gyroscope and accelerometer if they are available on the device. The touch major and touch minor values give the length of the major and minor axes of an ellipse that describes the touch area at the point of contact. Also note that since a pointing device may slide after it has generated a press event, the release event may not necessarily be generated with the same entity. This is especially the case with swype keyboards, although users were not allowed to use this method of input for data collection.

Another important consideration is that software keyboards usually never display all the possible symbols on a single screen due to limited available area size. This is remedied by presenting the user with different ways to input additional symbols, such as alternative layouts (for example, for numbers and symbols), and by using long key presses on some of the buttons. Therefore, the keyboard layout must be recorded, as different layouts contain different entities.

Data was collected from 10 users on a Nexus 4 device in both portrait and landscape

mode. No instructions were given other than to use the device normally and type on the soft keyboard for several minutes.

In addition to the data collection, a publicly available mobile dataset is used. The Touchanalytics database contains touchscreen data from 41 users [19]. There were 5 different phones and 7 different tasks involved. In 4 of the tasks, users were instructed to read a document followed by answering a questionnaire. The other 3 tasks involved spotting the difference between two images. Similar attributes to those in Table 2.2 were recorded, less the accelerometer and gyroscopic data.

2.4. Additional datasets

2.4.1. Keystroke RTI

A keystroke random time-interval (RTI) dataset contains a sequence of instantaneous, single-class events from 60 users. The only information available is the time at which each event occurs. This presents a challenge to traditional feature extraction techniques for keystroke biometrics, and a new dynamical systems approach is introduced. The dataset was collected by asking users to repeatedly hit a single key on a keyboard, with the hypothesis that the rhythm of individual users will be unique. It is part of the benchmark dataset for the 2014 Competition on Biometric identification based on user-generated RTI.

2.4.2. Web search history

A web search history database is provided by [27]. The database contains anonymized browsing history for 452 users. Each record contains the timestamp of the visit, a unique ID for the visit place, the type of request, and the place ID of where the user came from. The database contains over 4 million page visits.

2.4.3. Eye movement

The Second Eye Movements Verification and Identification Competition (EMVIC) provides a database of eye movements recorded from 34 users. Each user was instructed to look at several different photographs and decide if they know anyone in each photograph. Samples were recorded at a rate of 1kHz and vary in length from 891ms to 22,012ms.

ID	Task	Users	Sessions	Source	Biometric (event freq.)
A	Online quiz	43	1.2k	[51]	Keystroke ($2.5Hz$) Stylometry
B	Mixed	57	738	New	Keystroke ($0.2Hz$) Mouse click ($0.1Hz$) Mouse motion ($12Hz$) Mouse scroll ($0.05Hz$)
B1	Edit paragraph	45	270		
B2	Web browsing	40	240		
B3	Game 1 (Solitaire)	18	108		
B4	Game 2 (Star Bubbles)	20	120		
B5	Online quiz	18	198		
C	Fixed motion tasks	10	60	[48]	Mouse motion ($9Hz$)
D	Mixed	144	1.7k	[52]	Keystroke ($3.2Hz$)
D1	Essay questions	14	210		
D2	Essay questions	30	400		
D3	Essay questions	119	845		
D4	Copy task	39	393		
D5	Essay questions	142	1.3k		
E	Keypad	30	600	[5]	Keystroke ($2.7Hz$)
F	Password	51	20k	[31]	Keystroke ($4Hz$)
G	Typing	10	52	New	Mobile touch ($4Hz$)
H	Trigger actions	41	232	[19]	Mobile touch ($1.6Hz$)
I	Web browsing	452	500k	[27]	Web search history ($3.5\mu Hz$)
J	Picture gazing	34	837	IJCB 2014: EMVIC	Eye movement ($1kHz$)
K	RTI	60	420	IJCB 2014: RTI	Keystroke RTI ($3.4Hz$)
L	Password	100	1k	[35, 34]	Keystroke with pressure ($3.7Hz$)
M	Password	133	7.5k	[20]	Keystroke ($4.4Hz$)

Table 2.3.: Dataset summary

3. Authentication Model

3.1. Introduction

The classification procedure is based on a vector-difference authentication model which transforms a multi-class problem into a two-class problem. The resulting two classes are within-person ("you are authenticated") and between-person ("you are not authenticated"). The dichotomy model is a strong inferential statistics method found to be particularly effective in large open biometric systems where it is not possible to train the system on all individuals in the population. The applications of interest here, however, involve a closed population where it is possible to train the system on all of the authorized users. Therefore, a more accurate "engineering" closed-system procedure was developed for these and similar applications.

In the simulated authentication process, a claimed user's keystroke sample requiring authentication is first converted into a feature vector. The differences between this feature vector and all the earlier-obtained enrollment feature vectors from this user are computed. The resulting query difference vectors are then classified as within-person (authentication) or between-person (non-authentication) by comparing them to the previously computed difference vectors for the claimed user. A k-nearest-neighbor algorithm with Euclidean distance is used to classify the unknown difference vectors, with a reference set composed of the differences between all combinations of the claimed user's enrolled vectors (within-person) and the differences between the claimed user and every other user (between- person). Thus, differences of difference vectors are being calculated.

3.2. Dichotomy model

The classification procedure uses a vector-difference authentication model which transforms a multi-class problem into a two-class problem [13]. The resulting two classes are within-person ("you are authenticated") and between-person ("you are not authenticated"). To explain the dichotomy transformation process, consider a small example of three people {P1, P2, P3} where each person supplies four biometric samples. Figure 3.1 plots the biometric sample data for these three people in two-dimensional feature space. This feature space is transformed into a feature-difference space by calculating vector distances between pairs of samples of the

same person (within-person distances, denoted by $x_{\oplus}$) and distances between pairs of samples of different people (between-person distances, denoted by $x_{\oslash}$). Let d_{ij} represent the individual feature vector of the i^{th} person's j^{th} biometric sample, then the sets $x_{\oplus}$ and $x_{\oslash}$ of vector differences are calculated as follows:

$$x_{\oplus} = \{|d_{ij} - d_{ik}| \, where \, i = [1 \ldots n], \, j, k = [1 \ldots m], \, j \neq k\} \tag{3.1}$$

$$x_{\oslash} = \{|d_{ij} - d_{kl}| \, where \, i, k = [1 \ldots n], \, i \neq k, \, j, l = [1 \ldots m]\} \tag{3.2}$$

where n is the number of people, m is the number of samples per person, and the absolute value is of the elements of these vectors. If n people provide m biometric samples each, the numbers of within-person $n_{\oplus}$ and between-person $n_{\oslash}$ distance samples, respectively, are:

$$n_{\oplus} = \frac{m \times (m-1) \times n}{2} \tag{3.3}$$

$$n_{\oslash} = m \times m \times \frac{n \times (n-1)}{2} \tag{3.4}$$

Figure 3.1 shows the transformed feature difference space for the example, $n = 3$ and $m = 4$ yields $n = 18$ and $n = 48$ for a total of 66 vector differences. The two highlighted difference samples come from the two lines in Figure 3.1.

Two performance enhancing modifications were made in converting the open to the closed-system procedure. First, the new procedure matches the claimed user's sample against all the enrollment samples from that user for authentication rather than just one as in the open system. Second, the new procedure is user-focused in that only the claimed user's enrollment samples and their relationships to the other users' enrollment samples are utilized in the classification process.

Because most pattern recognition systems calculate difference vectors in the matching/classification process, the fact that the dichotomy model takes differences of difference vectors is often not understood as different and unique. The training space of difference vectors grows rapidly as the population increases, particularly the number of between-person distance samples – for example, 200 users each producing 10 enrollment samples generates 1,990,000 between-person distance samples. Thus, it is necessary to reduce the number of training difference vectors, and previously a random sampling was performed. Now, however, an improved reduction method has been discovered that has led to significantly higher performance. For efficiency and performance the improved user-focused reduction method retains only training difference vectors that include the claimed user's test samples. Thus, the numbers of within-person n and between-person n distance samples, respectively, become:

$$n_{\oplus} = m \times (m-1)/2 \tag{3.5}$$
$$n_{\oslash} = m \times m \times (n-1) \tag{3.6}$$

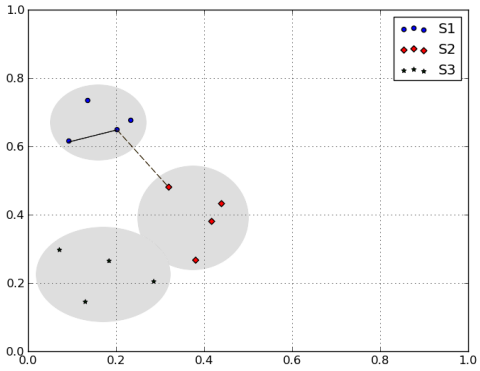


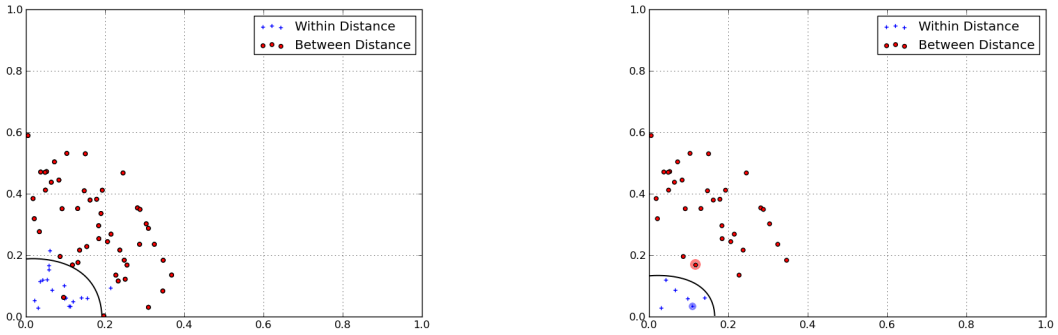
Figure 3.1.: Feature space

For the small example illustrated above, 3.2a shows the corresponding feature-difference spaces for user S1, yielding n = 6 and n = 32.

For large n the number of vector difference samples, especially the between-person differences, is greatly reduced. For example, in the above mentioned example of 200 users each producing 10 enrollment samples, the number of between-person distance samples is reduced from 1,990,000 to 19,900. More importantly, this user-focused approach improves performance by taking into account the clustering of the individual user's samples. Figure 3.1 shows three clusters of samples from the three users. 3.2b shows the feature difference space for only user S1 where the within-class feature-difference samples are clustered rather tightly, corresponding to the tight cluster for user S1 in feature space Figure 3.1. In contrast, the within-class feature difference samples in 3.2a are less tightly clustered because they represent the feature-difference samples from all three users. Now, realizing that 3.2b characterizes this phenomenon for only two features to permit a two-dimensional drawing, consider the greater overall tighter clustering effect of this user-oriented approach in a higher dimensional pattern feature space.

3.3. Model validation

The methods of evaluation in this section are all considered technical evaluations of a BAS.



(a) Original feature difference space **(b)** User-only feature-difference space

Figure 3.2.: Feature difference space

3.3.1. Receiver operating characteristic (ROC) curve

The *false accept rate* (FAR) and *false reject rate* (FRR) are traditionally used to measure the performance of a binary decision classifier, where data may be labeled as either positive or negative.

Receiver operating characteristic (ROC) curves characterize the performance of a biometric system and show the trade-off between the False Accept Rate (FAR) and the False Reject Rate (FRR). In this study, the ROC curves were obtained using a linear-weighted decision procedure of the k nearest neighbors with k=21 unless otherwise noted. Each neighbor is assigned a weight, from k to 1, with the closest neighbor weighted by k, the second by k-1, ..., and the farthest by 1. With k fixed, another parameter, l, is varied from 0 to k(k+1)/2, resulting in 232 points on the ROC curve. At each point, the query sample is accepted as within if the weighted sum is greater than or equal to l and between otherwise. The error rates are then calculated as $FAR = FP/(FP + CN)$ and $FRR = FN/(FN + CP)$, where FP = # false positives, FN = # false negatives, CP = # correct positives, and CN = # correct negatives.

3.3.1.1. Leave-one-out cross-validation (LOOCV)

The *leave-one-out cross-validation* (LOOCV) procedure simulates many true users trying to get authenticated and many impostors trying to get authenticated as other users. For n users each supplying m samples, m n positive (one for each sample) and m 1 negative (each sample versus the other users) tests can be performed. This is particularly effective for smaller populations and limited data, since it utilizes all of the enrolled samples in a BAS.

The procedure is implemented as follows. For each question (“left-out”) test sample to be authenticated, the within and between-person difference vectors are computed

without that sample. This creates the reference set, which consists of the entire population less the sample to be authenticated. The query vectors are then computed by taking the difference vectors between the question test sample and enrollment samples belonging to the claimed user. The results of a nearest neighbor classification for each of the test difference vectors are grouped together to make the decision by considering the nearest neighbors from all the resulting vector differences. The FAR and FRR can be calculated when the procedure is complete by counting the number of FP, FN, CP, and CN.

3.3.1.2. Repeated random sub-sampling

In cases where a population is very large, or each user has many enrolled samples, LOOCV is not possible. Instead, a repeated random sub-sampling is used to validate the model. For each repeated experiment, the database is split into a reference set and a query set. All of the samples in the query set are authenticated as each user is the reference set. The FAR and FRR are then calculated similarly to the LOOCV.

3.3.2. Biometric menagerie

The equal error rate (EER) acts as a single measure of the performance of a biometric authentication system. We have generally found that a small number of samples with larger amounts of data result in lower error rates. In order to evaluate the performance of a system in practice, it is important to account for the variability in types of users. In several experiments, user error histograms are introduced in order to determine the distribution of individual error at the population EER. It has been pointed out that users may often fall into different categories in a BAS [56]. The following ontology is proposed:

Sheep Easily identifiable users

Goat Users which are difficult to identify

Lamb Users which are easily imitated

Wolf Users which are good at imitating others

The presence of both lambs and wolves in the population of a BAS presents a potential security threat. It should also be noted that all of experiments in this thesis involve zero-effort attacks in order to evaluate performance.

4. Keystroke

4.1. Introduction

Keystroke dynamics refers to the way a person types on a keyboard. Individual typing characteristics allow for the identification and authentication of users.

Keystroke dynamics can be used in both online and offline applications. In online applications, it is desirable to verify a user as soon as possible. Since longer samples give a better estimate of a user's typing characteristics, there is a tradeoff between time and accuracy.

There primarily two categories of keystroke biometrics: fixed text and free text. Fixed text applications deal with input where the ordering of events is fixed, such as password or PIN entry. Sequences which are observed and do not match the expected order of events are generally discarded. This requires that a user inputs the correct sequence before the keystroke dynamics of that sequence are even considered for authentication. Free text applications deal with sequences where the ordering of events is unknown beforehand. Features may be optimized depending on the type of input.

The keystroke biometric is a behavioral biometric that has gained contemporary popularity as the keyboard provides a vital input device. Keystroke biometric systems utilize the keystroke dynamics as features or measurements [12]. The usual keystroke dynamic measurements are the dwell (key press duration) and flight (transition between two keys) times.

The goals of this chapter are to further develop and quantify the keystroke biometric by evaluating the technical performance of a system which authenticates users by keystroke dynamics. Environmental variables considered are population size, input length, and input type. Several feature sets are evaluated and a novel fallback mechanism is proposed for free-text applications.

4.1.1. Related work

A comprehensive evaluation of long text keystroke biometrics can be found in [52]. Villani et al. collected both free and fixed-text long samples from over 100 participants. Factors such as keyboard type, fixed vs free-text, and template degradation

over time were evaluated to determine the performance of a long-text keystroke authentication system.

Killourhy et al. evaluate the performance of several classifiers on short fixed-text input.[31]. Using a database of 400 password-typing samples from 51 participants, they found the best EER to be 0.096% with a scaled Manhattan detector described in [4]. In [30], the authors determined a 1% EER for 200 PIN entries from each of 28 participants. In this case, 2 out of 3 repetitions of the PIN are considered, requiring that 2 match the user's template. A $\pm 200\mu s$ resolution was obtained with a special clock.

With special hardware, Loy et al. evaluate the performance of an authentication system using pressure-based typing biometrics[35]. A keyboard which records the pressure exerted on each key was created. They combine latency times and the typing pressure biometric to achieve a FAR of 0.87% and FRR of 4.4% on 100 samples made up of 10 participants typing an 8-character password 10 times each.

Roth et al. make use of only keystroke sound, picked up by a low-cost microphone[46]. They evaluate the possibility of authenticating an individual from typing sound alone, with an experimental result of 25% EER for 5-8 minute sessions recorded from 45 participants.

In [47], Serwadda et al. examine the possibility of statistical attacks against keystroke biometric systems. As the technology becomes more prevalent, attackers are more likely to find innovative means of defeating the system. Zero-effort attacks do not fully capture the system performance in the presence of statistical attacks performed by bots, as shown by the authors. They concluded with an increase of the EER of three high performance systems by 28.6% to 84.4%. In order for a bot to perform a statistical attack, the variation in typing characteristics of a representative population must be known. With public keystroke databases, it is possible for an attacker to obtain this information and enumerate a user's keystroke dynamics.

4.1.2. Other factors to consider

Using the same database as [31], Killourhy et al. determine the effect of the system clock on keystroke biometric authentication systems in [29], and determine an increase of 4.2% when using a standard low-resolution clock of $\pm 10 - 15ms$ compared to a high resolution clock with an accuracy of $\pm 200\mu s$. They hypothesize that other factors, such as bus contention, system load, and networking delays, may have an effect on a user's keystroke dynamics. Any environmental factor which interferes with the HIC has the potential to disrupt the dynamics of a user's keystrokes.

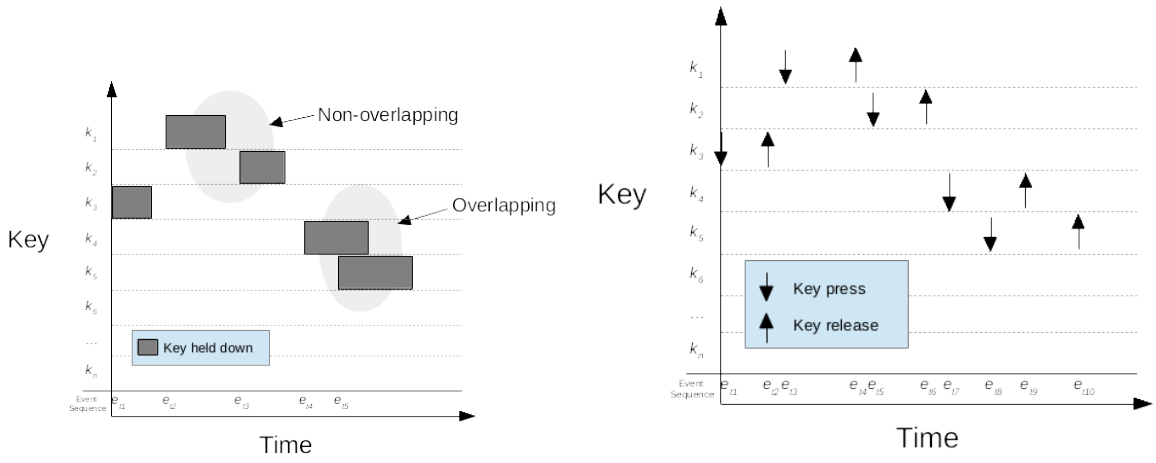
User emotion was determined to have an effect on keystroke dynamics in [28]. Khanna et al. found that a positive state generally led to an increase in typing speed, while a negative state led to a decrease. Recognition rates of negative vs neutral and positive vs neutral user states were found to be anywhere from 62-89% using various classifiers.

4.1.3. Placement on Newell's Time Scale

The keystroke biometrics operates primarily in the Cognitive Band of Newell's Time Scale [41]. Using dataset D2 to justify this claim, the rate at which a key is pressed for the typical user is $295ms$ with a standard deviation of $126ms$. The formation of words also lies in the Cognitive band, as the word rate of the typical user was found to be $1.6s$. The boundary between the Cognitive and Rational Bands lies somewhere between the formation of words and sentences, as the sentence rate of the typical user is $21.3s$. This implies that the individual keystrokes should provide a more reliable biometric than measurements taken on words and sentences, which operate at a higher cognitive level.

4.1.4. Keystroke event models

There are primarily two different event models for keystroke event sequences. In the first, events are order by the timestamp of the press event. The event classes correspond to the location of the physical key on the keyboard. Each event contains a single attribute: the timestamp of the key release (or alternatively, the duration the key was pressed for). This model is depicted in 4.1a. For each event, $e_t = (key, time_{release})$, where t is the key press timestamp, c is the identity of the physical key, and$time_{release}$ is the timestamp of the key release. The key release timestamp may be replaced by the duration the key was held down for.



(a) Duration model **(b)** Instantaneous model

Figure 4.1.: Keystroke event models

In the other event model, keystroke events contain no attributes and the class of the event is uniquely determined by the action (press or release) and the identity of the physical key. In this model, events occur instantaneously and press/release action

may occur in any order. This is depicted in 4.1b. For each event, $e_t = (key, action)$, where t is the event timestamp, *action* is either **press** or **release**, and *key is* defined similarly as above.

It is trivial to convert a keystroke sequence from one model to the other. However, different features may be extracted for each type of model, as will be seen in the following sections.

4.2. Statistical features

Using the duration model, a number of statistical features can be defined which capture a user's typing keystroke dynamics. The differences between event timestamps and durations are considered. The various time measurements that can be made are in Figure 4.2. Note than for overlapping keystrokes, the type 1 transition time will be negative.

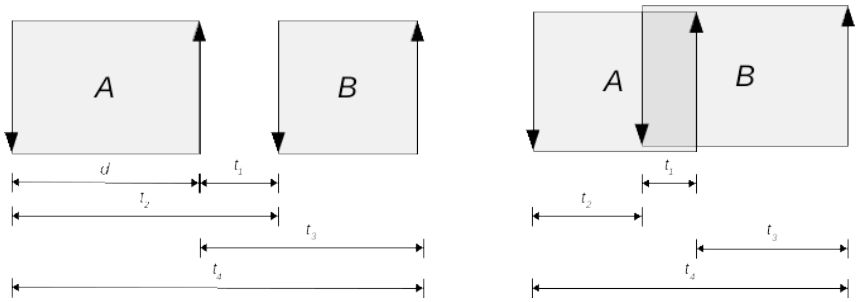


Figure 4.2.: Keystroke latency times for non-overlapping and overlapping keys

The feature set used to obtain experimental results on long free-text samples consists of:

- 78 duration features (39 means and 39 standard deviations) of individual letter and non-letter keys, and of groups of letter and non-letter keys
- 70 key-release-to-key-press transition features (35 means and 35 standard deviations) of the transitions between letters or groups of letters, between letters and non-letters or groups thereof, between non-letters and letters or groups thereof, and between non-letters and non-letters or groups thereof
- 70 key-press-to-key-press transition features (35 means and 35 standard deviations) identical to the above features except for the method of measurement
- 19 percentage features that measure the percentage of use of the non-letter keys and mouse clicks
- 2 keystroke input rates: the unadjusted input rate (total time to enter the text / total number of keystrokes and mouse events) and the adjusted input rate (total time to enter the text minus pauses greater than 1/2 second / total number of keystrokes and mouse events)

The full set of keystroke features can be found in Appendix A.

4.2.1. Features for fixed-text

For fixed-text, it is useful to consider the position of each key in the event sequence as the event identity, instead of the physical location of the key. For example, consider the expected text to be "biometric" without the quotes. The duration of the first key, rather than the duration of *b*, would be taken as a feature. Additionally, the duration of the 2nd and 8th keys would be taken as separate features, rather than the duration of *i*.

4.3. Fallback

A number of applications need methods for handling missing or insufficient data. In speech and language processing, several methods of handling missing or sparse data are described in [25]. The N-gram model of missing or infrequent data is estimated based on the (N1)-gram model of sufficient data recursively in the back-off [26] and deleted interpolation [23]. Although both models fail if the unigram is missing, this occurs rarely. In this section, a correlation technique is proposed to handle insufficient keystroke data.

For arbitrary text input and a fixed set of features, there may be insufficient observations to compute some features. The missing observations must be compensated for in some way. A fallback procedure places the features into a hierarchy so that features with insufficient observations are able to fall back to features with more observations. The observations in parent features are either a superset of the child features or contain more commonly occurring observations believed to be correlated with child features. Three different fallback hierarchies were evaluated. The first is a hierarchy based on the frequency of observations (characters) in the English language. Keys are grouped by character type, with the root of the hierarchy being all possible keys. The second hierarchy is a physiological model based on the layout of the keyboard and typing characteristics of touch-typists. The third fallback model is a regression model, where features with insufficient observations fall back to closely correlated features. The same model is used for the entire population, so correlation between features within an individual are not taken into account.

In contrast to password input, which is fixed, long-text input samples can consist of several hundred keystrokes of varying frequency. Therefore, keystroke biometric systems operating on such input can use statistical features such as the mean and standard deviation of the dwell and flight times [53]. These measurements, however, suffer from poor estimates of those keys where the number of samples during an acquisition session is missing or insufficient.

Inspired from the language processing "back-off" models, two hierarchical fallback tree models were evaluated for use in a keystroke biometric system. These hierarchical tree models served two functions. First, they provided fallback to additional data when insufficient keystroke instances were available to compute the statistical features. Second, they provided a granularity of features, where the granularity increases from gross features at the top of the tree to fine features at the bottom. The first hierarchical model, called the 'linguistic' model, organizes keys based primarily on frequency of use [53, 54]. The second model, called the 'touch-type' model, groups keys based on the fingers used to strike keys by touch typists [53]. These linguistic and touch-type models are depicted in Figure 4.3 and Figure 4.4, respectively.

An early analysis of the fallback aspect of the two models found the linguistic model to be slightly better than the touch-type model [53], although different features were used in the comparison since the models served two different functions. Compared to a default model of simply falling back to the top node of the hierarchy tree, the linguistic model reduced the error rate by 26% and 53%, respectively on two datasets, showing the utility of the hierarchy tree for the fallback function. These datasets contained samples of 500 or more keystrokes. Fallback never occurred more than one level up from the leaf nodes and most of the one-level-up nodes were never used (vowel, frequent consonant, all letters, non-letters) because their leaf nodes were sufficiently frequent to not require fallback.

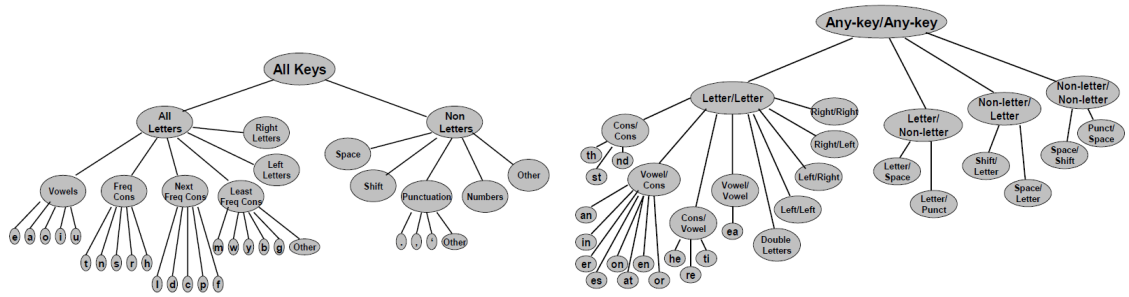
In this study, the two functions of the hierarchy tree – feature granularity and fallback – are separated. The hierarchy trees are retained for feature granularity and a sounder statistical model, called the correlation-based fallback table model, is proposed for fallback.

Two large independent long-text keystroke databases, **A** and **D2**, are used in this study. The first database is used to construct the correlation-based fallback table model. The second is used to evaluate system performance as a function of sample length. The new correlation fallback model should show improvement over the earlier models, especially as the data becomes sparser (fewer keystrokes per sample).

The concept of correlation plays important roles in many aspects of pattern recognition [32]; it can be modeled as an ultimate goal to optimize while it can be a serious problem to mitigate. In keystroke biometric, the problematic side of correlation between feature variables was addressed in [11, 14] to justify their choice of Mahalanobis distance over Euclidean distance. Here we focus on the useful side of correlation for estimating better feature values of sparse keystroke dwell data.

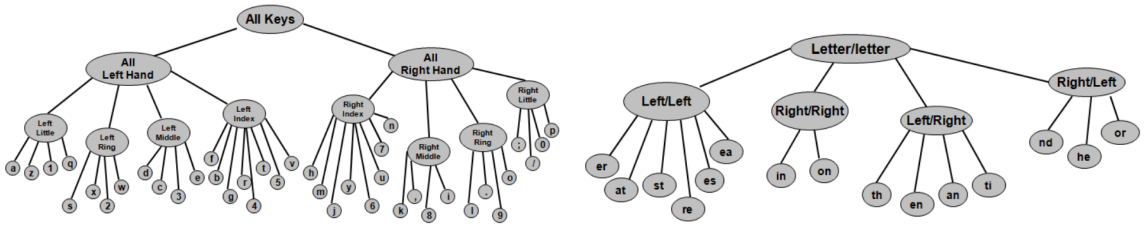
4.3.1. Feature correlation

The i^{th} keystroke of a user is denoted as A_i and consists of three tuples: key value, pressed time, and released time: $A_i = (k_i, p_i, r_i)$. The *dwell* is the duration of a key



(a) Linguistic duration fallback **(b)** Linguistic transition fallback

Figure 4.3.: Linguistic fallback model



(a) Touch-type duration fallback **(b)** Touch-type transition fallback

Figure 4.4.: Touch-type fallback model

pressed and a set of samples of dwell of a certain key x is defined in Equation 4.1.

$$S_x = \{r_i - p_i \,|\, (r_i,\, p_i,\, k_i) \in A \,\wedge\, k_i = x\} \tag{4.1}$$

Assuming that samples in S_x follow the normal distribution, the mean μ_x and standard deviation σ_x of S_x are often used as dwell features to represent a keystroke biometric sample [54, 53]. According to the fundamental theorem in probability called '*law of large numbers*' [2], the larger number of dwell information for a certain key, the closer the mean feature to the user's habitual expected value. However, the estimated μ_x and σ_x may be unreliable if the size of S_x, $|S_x|$ is too small.

Hence, linguistic [54] and physiological [53] hierarchical fallback models were used to mitigate this data insufficiency problem. If a certain key x occurs infrequently, i.e., the set size $|S_x|$ is less than t, the user defined threshold, the *fallback* procedure as defined recursively in Equation 4.2 was used to increase the number of samples; $S_x = fallb(x)$.

$$fallb(x) = \begin{cases} \{r_i - p_i \,|\, k_i \in leaf(x)\} & if\ |S_x| > t \\ fallb(parent(x)) & otherwise \end{cases} \tag{4.2}$$

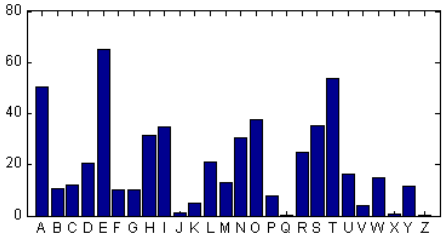


Figure 4.5.: The average frequency of alphabet keys

Poor correlations between parent and children nodes in the previous hierarchical models and the most correlated keys to each key are revealed from a keystroke database. The sessions in database **A** were limited to 500 keystrokes since many of the sessions vary significantly in length. The new database, **A'**, contains 1666 sessions from 43 users with an average of 38 sessions each and the session length is 495 on average. The data was collected from university students who took 4 online exams over a semester. Each session is approximately one question, with a total of 10 questions in each exam. Not all of the students completed the exam successfully, resulting in some missing sessions. Average frequency of each alphabet key in the database is given in Figure 4.5, which is astoundingly similar to that in common English published in [7]. This similarity justifies the representativeness of the keystroke database.

The database is represented as a table R as given in Figure 4.6 where each row represents a session and column represents the average key dwell and its frequency value, which is parenthesized. Prior to the correlation analysis, the preprocessing consists of extracting a sufficient co-exist table of two keys with a user defined threshold, t_1, is defined in Equation 4.3 and illustrated in Figure 4.6.

$$D_{x,y} = \{(\mu_{i,x}, \mu_{i,y}) \,|\, s_{i,x} > t_1 \wedge s_{i,y} > t_1 \tag{4.3}$$

The table $D_{x,y}$, not R, is used to derive the correlation between x and y. Let $n_{x,y} = |D_{x,y}|$ be the size of instances and the *Pearson product-moment correlation coefficient*, $\rho_{x,y}$ is defined in Equation 4.4. A value of $\rho_{x,y}$ closer to 1 indicates better correlation between two keys.

$$\rho_{x,y} = \frac{\sum_{i=1}^{n_{x,y}} (\mu_{i,x} - \bar{\mu}_x)(\mu_{i,y} - \bar{\mu}_y)}{\sqrt{\sum_{i=1}^{n_{x,y}} (\mu_{i,x} - \bar{\mu}_x)^2} \sqrt{\sum_{i=1}^{n_{x,y}} (\mu_{i,y} - \bar{\mu}_y)^2}} \tag{4.4}$$

4.7a and 4.7b are examples of good and bad correlated cases, respectively. 4.7c shows correlations of all pairs of vowels {a, e, i, o, u}. Upper right and lower left triangles contain the mean and standard deviation value plots for each pair of vowels.

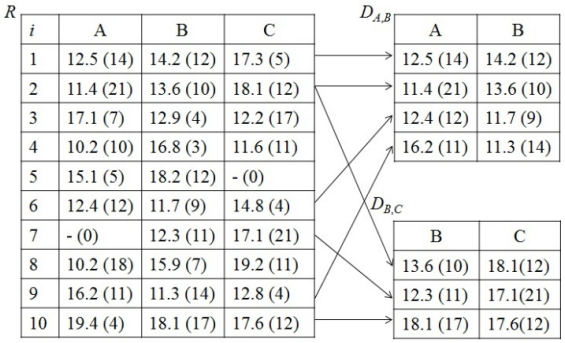
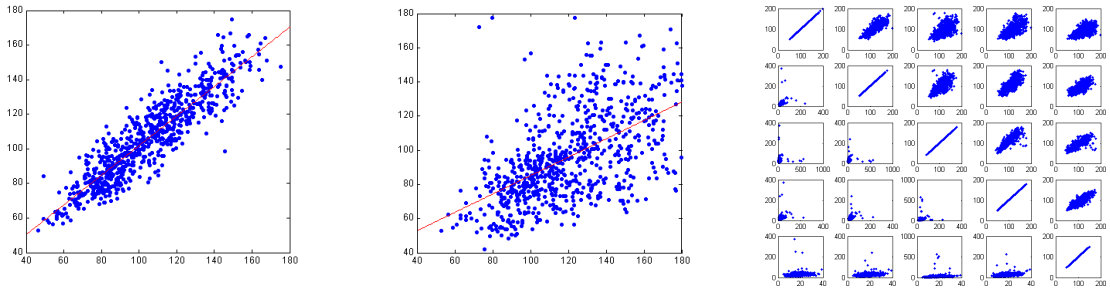


Figure 4.6.: Extracting sufficient co-exist table with t1 = 7



(a) Good correlation (0.908) b/w D and E

(b) Bad correlation (0.561) b/w A and I

(c) Correlations among vowels

Figure 4.7.: Correlation Plots between pairs of key dwells

Even after outlier remover, the standard deviation distribution did not form a linear correlation.

Table 4.1 shows the best four keys that correlate to each key. It should be noted that even though the key x correlates best with a key y, x is not necessarily the best correlating key for y. In all, it can be observed that majorities of keys correlate highly with other keys. Keys 'E' and 'S' have the highest correlation coefficient value and are one of the only two keys which are symmetrically correlated (the other being 'N' and 'O'). Letters like {'Q', 'X', 'Z'} are not frequently used and thus do not correlate well with others

Figure 4.8 shows a plot of key frequency and the first choice correlation coefficient. The correlation coefficient decreases for infrequently used keys. This suggests a limiting factor in the effectiveness of the regression model, since correlations with other keys are low and it is the infrequently used keys that usually must be accounted for small sample sizes.

There are two other important correlation parameters that can be useful in the later fallback model. They are the slope, $\alpha_{x,y}$ Equation 4.6 and the intercept, $\beta_{x,y}$ Equation 4.7 of simple linear regression line Equation 4.5 that fits two correlating keystroke dwell variables.

	1st Choice		2nd Choice		3rd Choice		4th Choice	
A	S	0.847	T	0.818	E	0.801	R	0.76
B	T	0.555	H	0.527	D	0.52	S	0.518
C	T	0.797	S	0.772	E	0.763	R	0.754
D	E	0.773	T	0.759	S	0.75	C	0.715
E	S	0.876	T	0.835	R	0.826	A	0.801
F	T	0.74	E	0.708	R	0.692	S	0.69
G	T	0.707	E	0.649	R	0.641	S	0.629
H	N	0.803	I	0.76	U	0.748	T	0.744
I	N	0.809	O	0.804	T	0.771	H	0.76
J	U	0.407	I	0.4	O	0.386	P	0.374
K	O	0.577	L	0.558	I	0.546	N	0.544
L	O	0.775	T	0.72	S	0.719	I	0.716
M	N	0.789	O	0.729	U	0.724	I	0.723
N	O	0.833	I	0.809	H	0.803	U	0.79
O	N	0.833	I	0.804	U	0.777	L	0.775
P	H	0.6	I	0.599	O	0.595	U	0.587
Q	E	0.596	S	0.594	T	0.592	A	0.574
R	T	0.849	E	0.826	S	0.796	A	0.76
S	E	0.876	T	0.86	A	0.847	R	0.796
T	S	0.86	R	0.849	E	0.835	A	0.818
U	N	0.79	O	0.777	I	0.759	H	0.748
V	E	0.581	T	0.562	S	0.559	R	0.541
W	E	0.775	S	0.771	T	0.742	A	0.71
X	E	0.538	T	0.529	R	0.51	S	0.504
Y	T	0.595	E	0.569	S	0.569	R	0.551
Z	E	0.46	A	0.445	T	0.443	S	0.433

Table 4.1.: Correlation coefficient-based fallback table

$$f_{x,y}(x) = \alpha_{x,y}x + \beta_{x,y} \tag{4.5}$$

$$\alpha_{x,y} = \rho_{x,y}\frac{\sigma_y}{\sigma_x} \tag{4.6}$$

$$\beta_{x,y} = \mu_y - \alpha_{x,y}\mu_x \tag{4.7}$$

These parameter values are given in Table C.1 in Appendix C.

4.3.2. Correlation based fallback table

In the previous section, it was claimed and discovered empirically that most key dwell mean values having high correlation with other keys. This section focuses

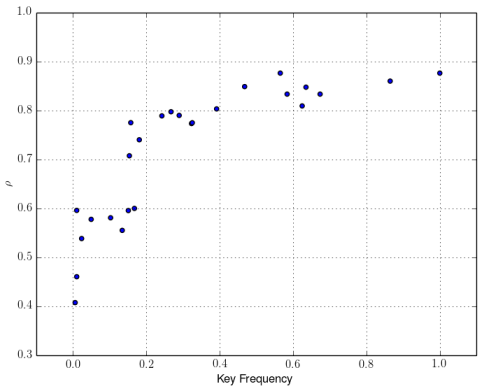
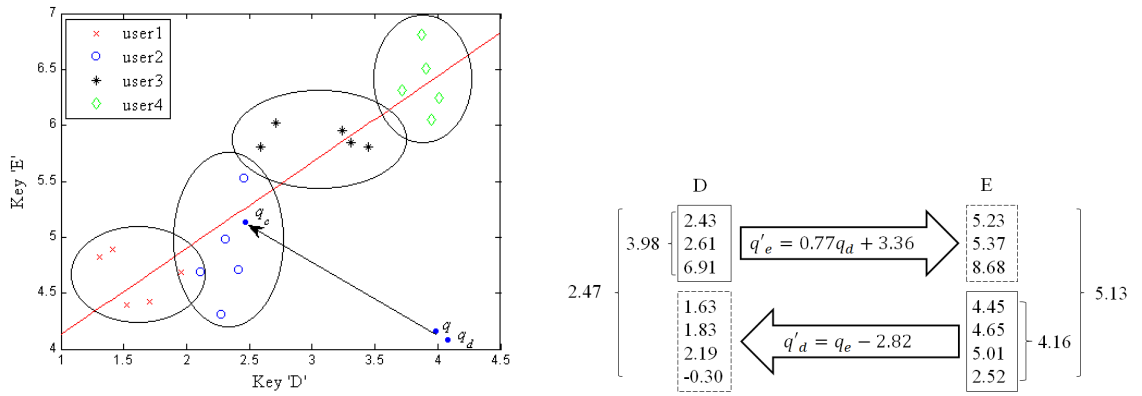


Figure 4.8.: Max correlation vs. key frequency

on how to utilize these discovered correlation parameters to better estimate the features.

Figure 4.9 illustrates the essence of our claims. Suppose that there are four users and each user provided five mean feature values as plotted with its linear regression line in 4.9a. Consider a query session, q, which claims to be a user 2. Only three and four samples appeared for the keys 'D' and 'E', respectively. The computed mean values $q = (3.98, 4.16)$ are poor estimates.



(a) 2-Feature space of keys D and E **(b)** Shifting from q (3.98, 4.16) to qc (2.47, 5.13)

Figure 4.9.: Proposed scenario

In this infrequent case, the most correlating key dwell values may be summed to compute the new mean value. As illustrated in 4.9b, the linear regression line can be utilized to convert the value. This new mean value augmented with linearly transformed highly correlating key dwell values is denoted as $q_c = (2.47, 5.13)$ and we claim that this is a much better estimate as depicted in 4.9a.

If the linear regression line is not used but the other key dwell values are directly augmented, this will also result in a poor estimate which is denoted as $q_d = (4.08, 4.08)$.

Previous hierarchical fallback models used direct values rather than linear transformed values.

Figure 4.10 shows the flow chart of the proposed model that utilizes the correlation information R discovered in the previous section.

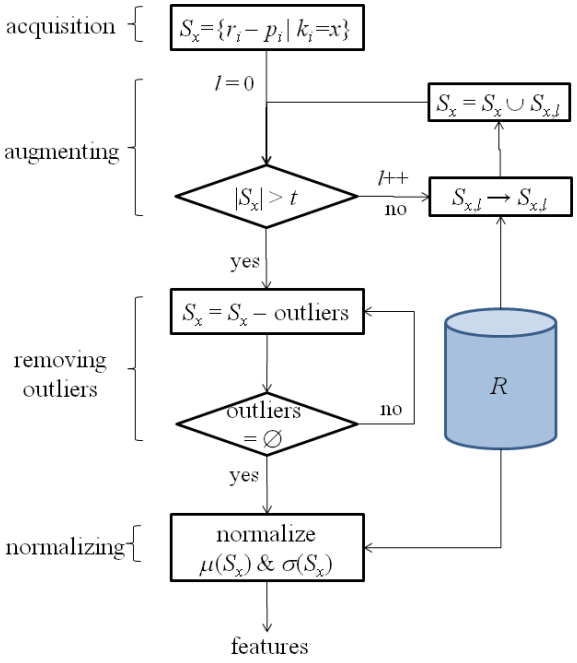


Figure 4.10.: Flow chart of proposed correlation-based fallback table model

$$S_{x,l} = \{\alpha_{x,l}(r_i - p_i) + \beta_{x,l} \mid (r_i, p_i, k_i) \in A \wedge k_i = k_{x,l}\} \tag{4.8}$$

The rows of R are alphabet keys containing the sorted other keys in descending order of correlation coefficient where each key has four tuples, $(k_{x,l}, \alpha_{x,l}, \beta_{x,l}, \gamma_{x,l})$ denoting the l^{th} rank key, slope, intersect, and correlation coefficient for the key x, respectively. Let $S_{x,l}$ defined in Equation 4.8 denote the set of linearly transformed values, for example in 4.9b, $S\prime_{E',1} = 5.23, 5.37, 8.68$ is transformed from $S\prime_{E'} = 2.43, 2.61, 6.91$.

The proposed correlation based fallback table model is defined in Equation 4.9 as a recursive function, cft. It is called initially, $S_x = cft(S_x, 0)$ with the user defined threshold for the minimum number of observations.

$$cft(S_x, l) = \begin{cases} S_x & if\ |S_x| > t \\ cft(S_x \cup S_{x,l}, l+1) & otherwise \end{cases} \tag{4.9}$$

It should be noted that Equation 4.9 is imperfect. It can be infinite when the size is never greater than t or undefined if there are no more keys available. Yet it gives the succinct definition of the proposed system.

If the alphabet 'A' appears sufficiently enough, i.e., above the user defined threshold t, we just use the feature set. If not, use the linear regression function in the first choice, i.e., 'E' for 'A'. If $|S_{'A'} \cup S_{'A',1}|$ are sufficient, use them for the feature value for $S_{'A'}$.

$$S_{x,l} = \{r_i - p_i | (r_i, p_i, k_i) \in A \wedge k_i = x \wedge r_i < tc\} \tag{4.10}$$

The system was tested on dataset **D2**, which contains a population of 30 users with a total of 400 sessions. Each user recorded between 10 and 25 sessions and each session contained between 500 and 1000 keystrokes. The samples were recorded over a semester and the population consisted mainly of university students. For each session, users were instructed to respond freely to essay-type questions.

Four different fallback models were evaluated with the same feature set. The features consist of the mean and standard deviation of each letter key, for a total of 52 features. Using only the 26 letter keys, a linguistic, physiological, regression, and default one-level model were used. The linguist and physiological models are subsets of the trees in 4.3a and 4.4a, containing only the relevant letter nodes, and the default model falls back to all letter keys when there are insufficient samples.

Each session was truncated at various intervals from 50 to 500 keystrokes to get the EER as a function of input length for each model. Figure 4.11 shows the equal error rate (EER) for each type of fallback model as a function of input length. Table 4.2 contains the EER values of each model at each input length.

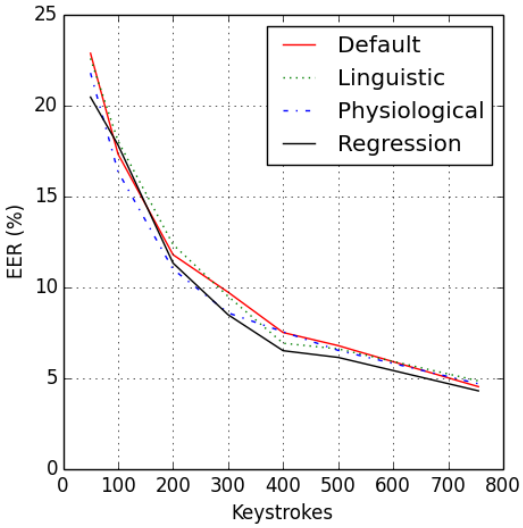


Figure 4.11.: EER of each fallback model as a function of $|S_x|$

The linear regression model offers modest improvements over the linguistic and physiological model with the exception of two different input sizes. This may be a result

$\|S_x\|$	Default	Linguistic	Physiologic	Regression
50	22.88	22.60	21.80	20.47
100	17.34	18.06	16.37	17.84
200	11.80	12.36	11.00	11.34
300	9.74	9.51	8.60	8.50
400	7.52	3.93	7.56	6.52
500	6.80	6.62	6.54	6.15
Max	4.54	4.86	4.70	4.31

Table 4.2.: Fallback model EER values

of the regression model having a higher number of fallbacks for the shorter input lengths than the hierarchical models, which rarely fall back beyond the first level. Features for infrequently occurring keys are more likely to "run out of data" quicker in the regression model, since each level contains only a single key. Calculating an accurate and useful linear regression model requires large amounts of data since its purpose is to effectively handle low occurring keystrokes. In general, the keys with the lowest frequency also have the weakest correlation with other keys, which limit the effectiveness of any fallback model.

Parent nodes in previous fallback models in[25, 26] played not only the substitute role for insufficient keys, but also global feature roles. Separating these two functions allows for greater flexibility in choosing features and a particular fallback model.

Correlation attributes (alpha beta gamma) among different key dwells are used to estimate any infrequent key dwell mean value. These attributes within a user may be utilized as distinctive features for verification purpose and it remains as a future work. The correlation between groups of keys and flight times were also not considered in this study and may play a role in a complete regression fallback method.

4.4. Bigram frequency

Using the instantaneous event model for keystroke sequences, the frequency of n-grams becomes of interest when trying to determine the source of the sequence. In sequence classification, n-gram features are often used as input to traditional classifiers, such as SVM or KNN[55]. The difficulty in using n-gram features to classify the user of a keystroke sequence is that the features quickly become dependent on the context as n increases. For this reason, only bigrams are considered. Intuitively, increasing n will capture character pair and work usage and lead to a weaker set of stylometric features.

To compute the bigram features, first the K most frequent bigrams of each session are calculated. These are each normalized to the sum of the K most frequent bi-

grams only, ignoring the other bigrams. This ensures that the frequencies are not affected by outliers or sample length, as well as choosing bigrams that are most likely to be approaching their true distribution. The union of all the K most frequent bigrams in each session is then calculated. With a diverse population, this is likely to be much larger than each K due to the occurrence of different bigrams in each session. For feature extraction, only the maximum L normalized frequencies in B are used. Increasing L has the effect of capturing population variability. Increasing and decreasing K has the effect of individual variability, eventually decreasing performance. Note that some of the M bigrams will not appear in any of a user's sessions, and have a frequency of 0.

Since the selection of features is dependent on the frequency of each user, key usage and some stylometric information is preserved (ie. the frequently occurring event bigrams are correlated with both frequent characters, when a release event immediately follows a press event for that key, and with character bigrams whether the keys overlap or not). The presence of overlap between keystrokes is also captured, or alternatively whether t2 is positive or negative between the most frequently used key pairs. Note that only the order of events is considered and timing information is not used. The procedure is summarized in Algorithm 4.1.

Algorithm 4.1 Relative n-gram frequency feature extraction

Let N be the set of users in a closed system
Let M_i be the set of sessions for user i
Let $f(s)$ = the frequency of every unique bigram in sample s
Let $\hat{f}(b, s)$ = the frequency of bigram b in sample s
$b_{ij} = \{\text{K most frequent bigrams} \in f(s_{ij}), i \in N, j \in M_i\}$
$B = \{\text{L most frequent bigrams} \in \bigcup_{i \in N, j \in M_i} b_{ij}\ \}$
Use B as the set of bigram features
Then calculate the feature vector x_{ij} for each sample as $x_{ij} = \{f(b, s_{ij}) / \sum_{b \in B} f(b)\}$

The frequency and Fisher score (ration of between-class variance to within-class variance) of keystroke digrams for dataset D1 is shown in Figure 4.12. Using the set of 10 features (for the frequency of 10 bigrams), the EER of dataset **D1** was found to be 18%.

4.5. Experimental Results

4.5.1. Input type

An application of keystroke dynamics can fall into two different categories: fixed text vs. arbitrary input. For a copy task, the user is copying some known text, such as a password. Features may be used which exploit some structure in the fixed text,

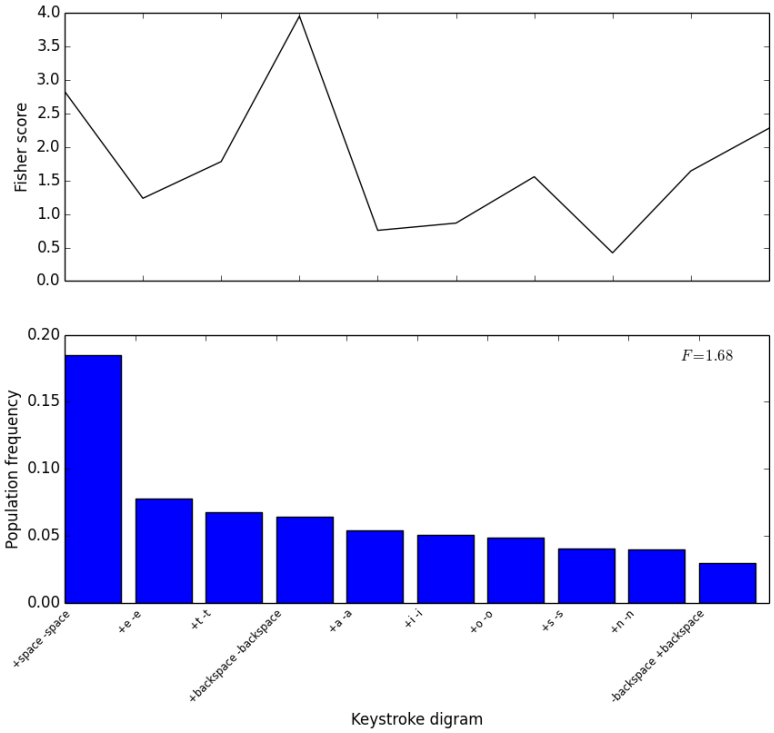


Figure 4.12.: Keystroke digram frequency

since the input will not change and is considered authentic if it does not match the original text. In arbitrary input applications, the length and sequence of characters are both unknown. Most keyboard input falls into this category. The features used for arbitrary input must be comprehensive and able to compensate for missing or low-frequency observations.

The cognitive load during keystroke input may have an effect on typing rhythm, causing delays during keystrokes or different sequences where the text entered will be evaluated at some point, perhaps as part of an online course.

Most fixed input occur as passwords or pin codes on a keypad. Longer fixed input applications are less common, although it may be useful to determine the accuracy as a function of input length for fixed input applications only.

4.5.2. Free text population size and input length

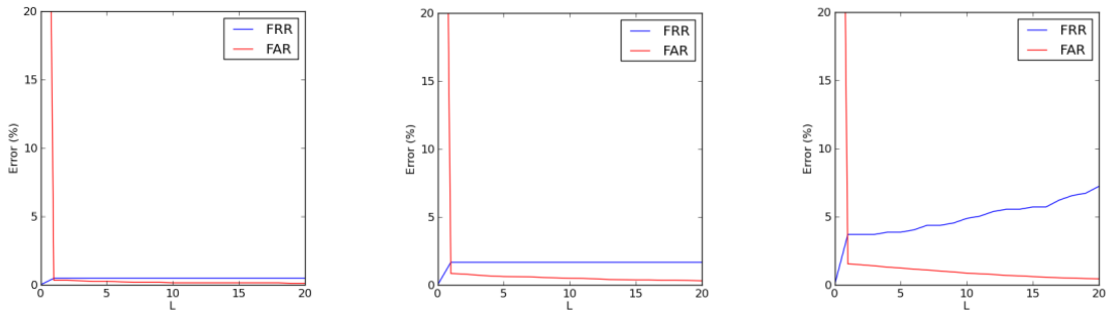
These experiments employed free-text (arbitrary input) keystroke data samples from dataset **D**. All the data samples contained over 500 keystrokes, averaged 755

keystrokes, and were input on Dell desktop PCs and laptop PCs (almost exclusively Dell machines). The data samples were collected in sets of five, the sets recorded at two-week intervals, and the five samples of a set usually recorded in a single day's session. For their five samples in a set, the participants were instructed to enter emails on five different topics from a given list of topics. Three experiments were conducted to analyze performance as a function of the number of keystrokes per sample and the user population size Table 4.3.

Exp	Number of Participants	Number of Samples Each	Total Number of Samples	EER(%)
1	14	15	210	0.4
2	30	10	300	1.7
3	119	5	595	3.7

Table 4.3.: Summary of experimental data and EER for the 755-keystroke samples

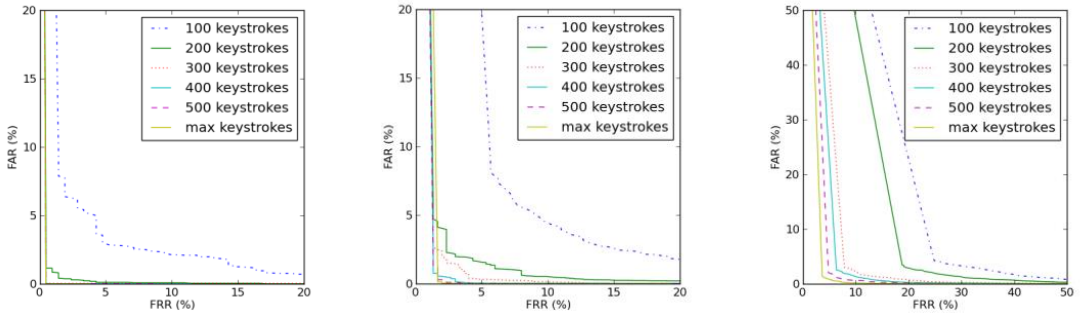
Each of the experiments involved positive and negative authentication tests. The number of positive tests = *number-of-samples* and the number of negative tests = *number-of-samples times (n-1)*. For example, for the 119 participant experiments, the 595 keystroke samples allowed for the evaluation of 595 positive and 70,210 (595x118) negative tests. The negative tests were zero-effort imitations by other subjects in the database. Figure 4.16 shows the ROC curves for the 14, 30, and 119 user experiments. Figure 4.13 shows the FAR/FRR versus the l (L) parameter for the 755-keystroke samples on the right. Although the EER can be approximated from the ROC curve, it can be more accurately determined form the crossover point on the FAR/FRR versus L curve (note that because the lowercase l can be confused with the digit 1 we sometimes use the uppercase L to represent the parameter). Although L goes from 0-231, expanded FAR/FRR plots at low L values are shown here because the crossover points on the FAR/FRR versus L curves occur in that region.



(a) 14-user FAR and FRR **(b)** 30-user FAR and FRR **(c)** 119-user FAR and FRR

Figure 4.13.: FAR and FRR for 14, 30, and 119 users at original input lengths

To determine how fast an unauthorized user could be detected, the EER rate was determined as a function of sample length (number of input keystrokes) for the 14, 30, and 119 user populations Figure 4.15.



(a) 14-user ROC curves **(b)** 30-user ROC curves **(c)** 119-user ROC curves

Figure 4.14.: ROC curves for 14, 30, and 119 users at various input lengths

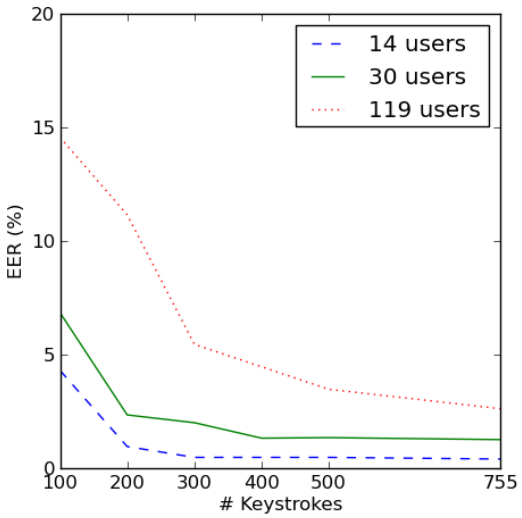


Figure 4.15.: EER vs. input length

For the maximum length keystroke samples, Figure 4.16 shows the ROC curves. The EERs were 0.4%, 1.7%, and 3.7% for the 14, 30, and 119 participant populations, respectively.

Because the mean population performance does not give the complete picture, the varied performance over the population of users was analyzed and described using the animal designations of [16]. Figure 4.17 shows three histograms analyzing the populations of 30 and 119 users for the maximum keystroke samples when operating at the EER point on the ROC curve. For each population size the three histograms show the FRR (potentially identifying those easily verified, sheep, and those difficult to verify, goats), the FAR of how easily the true authors were imitated (potentially identifying those easily attacked, lambs), and the FAR of how easily imitators can attack the true users (potentially identifying the strong attackers, wolves).

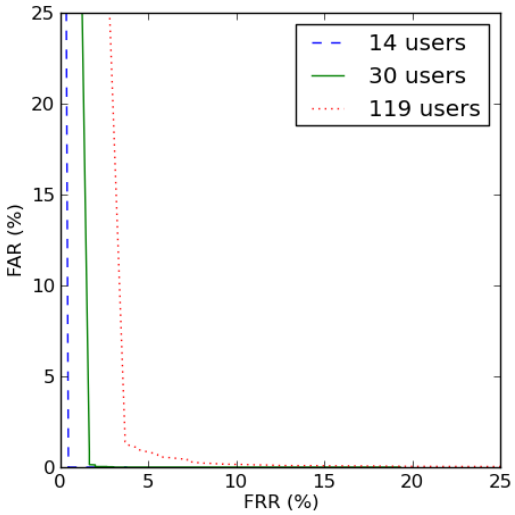
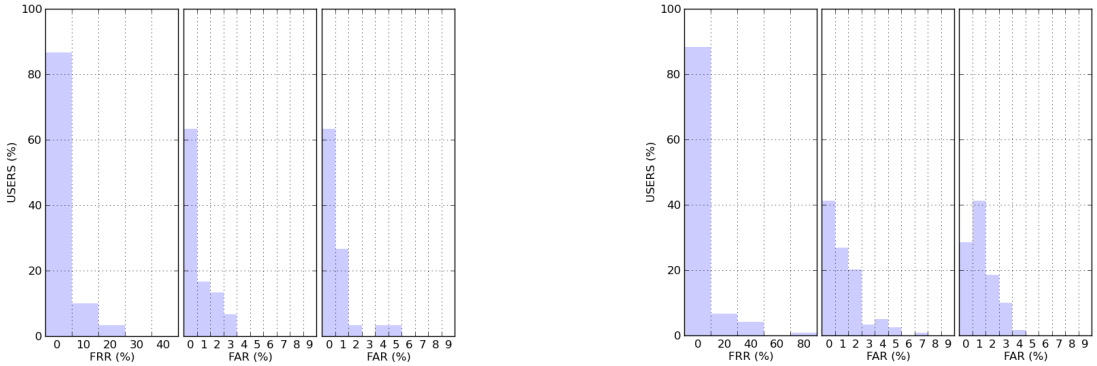


Figure 4.16.: ROC curves for 14, 30, and 119 users for the maximum keystroke samples

An examination of these histograms found only one significant outlier and that occurred in the FRR histogram of the 119 user population. This FRR histogram displays the percent of the 119 users (5 samples each) having 0% (0 of 5), 20% (1 of 5), 40% (2 of 5), 60% (3 of 5), 80% (4 of 5), and 100% (5 of 5) false rejects. Because one user had 80% (4 of 5) samples rejected while all others had 40% or fewer, this user is an outlier and could be considered a goat (a user difficult to authenticate).

To obtain system performance in this study we simulated the authentication process of many true users trying to get authenticated and of many zero-effort impostors trying to get authenticated as other users. An important advantage of this vector-difference model is that it provides relatively large numbers of between- and within-person distance samples for analysis and ROC curve generation. Furthermore, the leave-one-out method allowed for the closed-system evaluation of a considerably larger population size than had been evaluated previously.

As in a study by [6], the approach taken in this study was to train on as much enrollment data as possible while authenticating users on smaller quantities of data as appropriate for the application. For the test taker application there is usually no hurry to authenticate the user and all of the keystroke data for an online test could be used for authentication. However, for detecting unauthorized users in security sensitive applications the quantity of keystroke data must be limited to a minute or so in order to detect the intruder before significant harm is committed. Because large quantities of training data from authentic users – perhaps over many days, weeks, and even months – are available for some applications, such as the intruder detection application, elaborate procedures for training the system on significant quantities of data should be investigated.



(a) 30-user population **(b)** 119-user population

Figure 4.17.: Histograms of FRR (left), FAR of attack receivers (middle), and FAR of attackers (right)

For using such a continual authentication system on government or private company machines, keylogger software could be installed to transparently capture user input on all monitored PCs and the authentication processing performed on servers. However, because many employees like to use their PC for occasional personal use – email, banking, stock market transactions, etc. – there are obvious privacy concerns with a keylogger capturing all input, including account numbers and passwords. And, although the organizations might say they can monitor their machines as they like, the employees could have strong objections. To increase user acceptance and ameliorate privacy concerns, monitored machines should be clearly marked as such and unmonitored machines could be made available for employee personal use during lunch and break times.

Although privacy concerns remain, for authenticating test takers this should not be a problem because the students would be using a test-taking application. For the intruder detection application, it is important to relate typing speed to the number of keystrokes entered per minute, and it will be done in this discussion for the English language. The average word length is five, plus a space, or six characters per word. For average computer users, the average typing speed is about 33 words per minute, while a professional typist's speed is about twice that of the average user, and keyboard key spacing has an effect on the typing speed [43]. Since the number of keystrokes is usually only slightly more than the number of characters, the average computer user generates about 200 keystrokes per minute, while a professional typist generates about 400 keystrokes per minute. Assuming an average typing speed, 1-2 minutes of a potential intruder's input would likely be in the 200-400 keystroke range. In examining the error rate as a function of the number of keystrokes Figure 4.15, a big drop occurs in going from 100 to 200 keystrokes for 14 and 30 users, and in going from 200 to 300 keystrokes for 119 users, which implies that at least 200-300 keystrokes, or 1 to $1\frac{1}{2}$ minutes, is necessary to detect an intruder, which is hopefully fast enough to stop the intruder from causing damage.

The main contributions of this study were the development of the improved classification system and its performance evaluation. On samples of 300 or more keystrokes ($1\frac{1}{2}$ minutes or more at average typing speed) performance was over 98% on 30 users and over 94% on 119 users. On the large 755-keystroke samples performance reached 98.3% on 30 users and 96.3% on 119 users, indicating the potential of this approach.

In this study the EER was used for simplicity as a single value of performance to show the trends of performance as a function of the number of keystrokes per sample and the population size. However, in setting up the procedure for authenticating a user in a deployed system, the operating point on the ROC curve would be chosen appropriately, usually with a considerably lower FAR than FRR. For example, a good operating point on the ROC curves for 14 and 30 users in Figure 4.14 might be FRR = 2% and FAR ~ 0%. Although a low FAR operating point would incur more false rejections, several authentication failures could be required before making an unauthorized-user decision. How easily users accept such a system and at what value of FRR it becomes too intrusive are problems for future work concerning system deployment, maintenance, and user acceptance.

4.5.3. Short fixed text

Biometric analysis of short keystroke input has high security importance in two common applications – password input and number pad input. The password application is important because computer users frequently use passwords to log on to computers and to access email, bank, brokerage, and online store accounts. The number pad input application is important for similar reasons because digit only passcodes are used for access on automated teller machines (ATMs), on electronic security keypads for building and room access, and on mobile/digital phones. Passwords and digit-only passcodes are currently the only security measures employed in these access applications and the password/passcode information that we are required to remember is easily compromised. These security threat situations could be improved considerably if it were possible to authenticate the user more precisely, distinguishing between the genuine user and an impostor.

Two studies were conducted to measure the performance of the keystroke biometric authentication system in the password and numeric keypad input applications. Carnegie Mellon University (CMU) conducted similar studies [31, 37]. Using the passwords data from CMU [31], the first study consisted of experiments using two different feature sets – the features used in the CMU study and a new feature set created for this study. The second study on short numeric input used two different feature sets – the CMU features and a new feature sets created for the study.

Dataset **F** is used for this study, which was made available by Killourhy and Maxion[31]. Two experiments were conducted on the CMU 51 subject data using the closed-system dichotomy model described in section 3.2.

Due to the large number of samples per user, a repeated random sub-sampling validation method was used to derive the ROC curve, described in subsubsection 3.3.1.2. Each experiment was repeated three times, each time randomly splitting the feature space into 380 reference samples and 20 query samples per user. The reference samples are further refined by k-means clustering in order to reduce the size of both the within and between spaces. For each user, this would be equivalent to an authentication attempt after having recorded 380 samples. The key independent variable was the feature sets.

The first experiment used the CMU 51 subject data and re-implemented the 31 CMU features from [31]. From the 10-character password ".tie5Roanl" + Enter, the 31 CMU features were the 11 hold (dwell) times plus the 10 keydown-keydown and 10 keyup-keydown transition times.

The second experiment was similar to the first but used a set of 75 features designed for this study. The 75 features were differences between event timestamps using the instantaneous event model, where each event is the action-key combination occurring instantaneously at time t. For example, 'press e' and 'release e' are considered as two separate events and are not necessary consecutive. The set of bigrams over the entire population was used, producing the set of 75 bigrams. The time difference of each bigram was taken for each user. So although each feature vector contains 75 features, only 21 of these are non-zero, since there are only 22 events (21 pairs). The non-zero features are usually different for each user because some users consistently overlap certain keys while others do not, as show in section 4.4. Thus, there are 75 features, but it depends on the user which features are non-zero.

A summary of the results is shown in Table 4.4, the ROC curves in Figure 4.18, and the FAR/FRR versus parameter L curves in Figure 4.19. Although the EER can be approximated from the ROC curve, it can be more accurately determined from the crossover point on the FAR/FRR versus L curve (L is used in place of l to avoid confusion with 1). Although L goes from 0-231, expanded FAR/FRR plots at low L values are shown here because the crossover points on the FAR/FRR versus L curves occur in that region.

Experiment	Number of Subjects	Centroids per Subject	Number of Features	EER (%)
1	51	20	31	8.7
2	51	20	75 (21)	8.7

Table 4.4.: Experimental Results on CMU Password Data

Two experiments were conducted on newly collected numeric keypad input to compare against the password performance above and the CMU number-pad study [37]. In this study, although it was not possible to compare the algorithms on the same input data as in the password study above because the CMU data were not available. Instead, dataset **E** is used, which consist of new numeric keypad data were obtained in a manner similar to how the CMU data were collected.

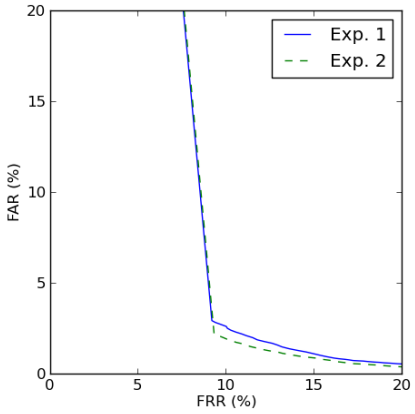
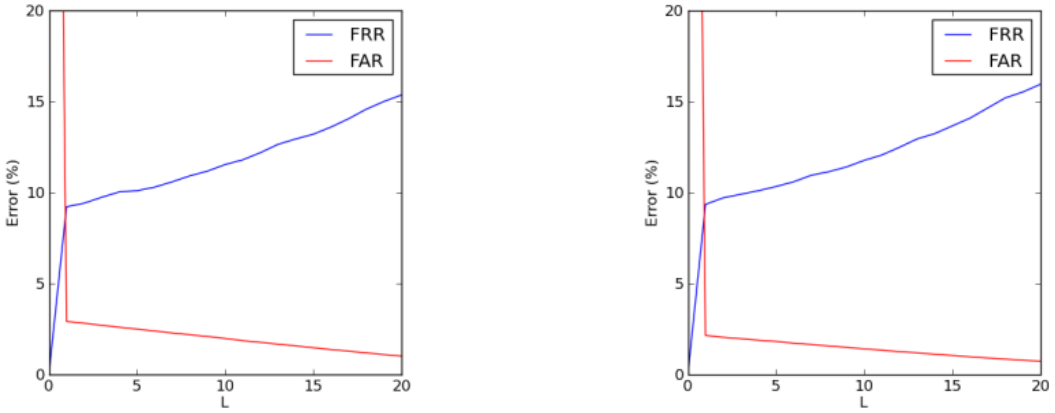


Figure 4.18.: ROC curves for the CMU data experiments



(a) Exp. 1 **(b)** Exp. 2

Figure 4.19.: FAR/FRR versus L for experiments 1 and 2

The third short text experiment used the same 31 CMU features as experiment 1 and the fourth experiment used similar features to experiment 2. The results are shown in Table 4.5, the ROC curves in Figure 4.20, and the FAR/FRR versus parameter L curves in Figure 4.21.

4.6. Conclusion

The performance of a behavioral biometric authentication system was evaluated on two different types of short input in which the sequence of keystrokes is fixed, password and numeric keypad sequences. In contrast to long text input applications, it is more feasible to obtain large numbers of enrollment samples for short input applications from the population of participants.

Experiment	Number of Subjects	Samples per Subject	Number of Features	EER (%)
Numeric Keypad 1	30	20	31	10.5
Numeric Keypad 2	30	20	44	6.1

Table 4.5.: Experimental Results on CMU Password Data

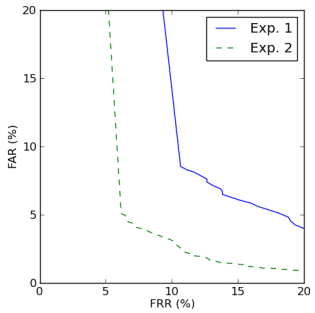


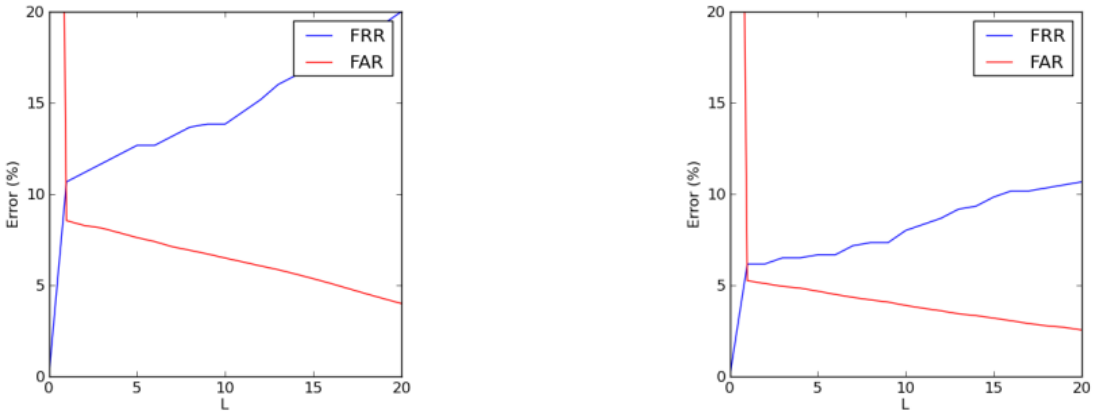
Figure 4.20.: ROC curves for the CMU data experiments

An advantage of the vector-difference model is that it operates efficiently with a small number of enrollment samples. Due to the large reference set in the first set of experiments, enrollment samples are reduced by k-means clustering, creating a smaller "ideal" reference set for each user. This model was then validated by a repeated-random sub-sampling procedure. With a smaller number of samples per subject in the second set of experiments, a leave-one-out validation method was used to obtain system performance.

Privacy is less of a concern for applications of interest which collect fixed-text short input, as opposed to those which require long-text arbitrary input. Many authorizing entities (ATM, security keypad, mobile phone) already have knowledge of the password or numeric sequence which must be entered. Others, such as a cash register, do not obtain any personal information from the user. Obtaining a metric on the keystroke dynamics of the fixed input does not require monitoring a session and possibly collecting personal information, as is the case in arbitrary input.

The main contribution of this study was the development of an improved classification system and its performance evaluation. In the password study, on the same CMU input data, the Pace Classifier was compared against 14 other systems analyzed in a CMU study [31]. The three top performing systems in that study had EERs between 9.6% and 10.2%, while the EER achieved in this study was 8.7% on both the features from the previous study and on a new set of features.

In the numeric keypad study, the EERs achieved were 10.5% and 6.1%, which are comparable to those obtained in the password study described here and to the basic EER of 8.6% obtained in the CMU numeric keypad study [37].



(a) Exp. 1 **(b)** Exp. 2

Figure 4.21.: FAR/FRR versus L for experiments 1 and 2

5. Stylometry

5.1. Introduction

Text authorship is becoming an increasingly important task as information is transmitted in an environment where anonymity is not only possible, but easy to obtain for the average user. One way of attributing an author to a piece of text is through a stylometry analysis. This technique involves characterizing the writing style of a person through a number of syntactic features. Combined with keystroke analysis, this is particularly effective on long text input. In a situation where keystroke information is not available (a keylogger cannot be legally install on the client machine) or not reliable (the keystroke events within a web browser may not correlate to the system key events), then authorship must be obtained through a textual analysis alone. Syntactic features have proven to be reliable for long text input, but as the length or quality of the text is decreased, then the ability to discriminate based solely on these features also decreases. The goal of this study is to perform a deeper level semantic analysis in order to attribute an author to a piece of text with a higher confidence. This is done with both short and long text samples, and compared with result previously obtained using only stylometry features.

The main application of interest in this study is verifying the identity of students in online examination environments, an application that is becoming more important with the student enrollment of online classes increasing, and instructors and administrations becoming concerned about evaluation security and academic integrity. The 2008 federal Higher Education Opportunity Act (HEOA) requires institutions of higher learning to make greater access control efforts for the purposes of assuring that students of record are those actually accessing the systems and taking online exams by adopting identification technologies as they become more ubiquitous [1]. To meet the needs of this act, the keystroke biometric seems appropriate for the student authentication process. Stylometry appears to be a useful addition to the process because the correct student may be keying in the test answers while a coach provides the answers with the student merely typing the coach's words without bothering to convert the linguistic style into his own.

Keystroke biometric systems measure typing characteristics believed to be unique to an individual and difficult to duplicate [12, 24]. The keystroke biometric is a behavioral biometric, and most of the systems developed previously have been experimental in nature. Nevertheless, there has been a long history of commercially unsuccessful implementations aimed at continuous recognition of a typist.

While most previous work dealt with short input (passwords or short name strings) [8, 21, 40, 44, 45], some used long free (arbitrary) text input [9, 22, 33, 39, 51, 53]. Free-text input as the user continues typing allows for continuous authentication [17, 38, 39, 49] which can be important in online exam applications [18, 51].

Stylometry is the study of determining authorship from the authors' linguistic styles. Traditionally, it has been used to attribute authorship to anonymous or disputed literary documents. More recently, computer-based communication and digital documents have been the focus of research, sometimes with the goal of identifying perpetrators or other malicious behavior. Recent computer studies have used stylometry to determine authorship of emails, tweets, and instant messaging, in an effort to authenticate users of the more commonly used digital media. A few studies have applied stylometry to the detection of intentional obfuscation or deceptive writing style, and others to the detection of the author's demographics [10]. Appendix D summarizes the prior authorship attribution stylometry studies and lists the associated references.

There are several reasons keystroke and stylometry biometric applications are appealing. First, they are not intrusive to computer users. Second, they are inexpensive since the only hardware required is a computer with keyboard. Third, text continues to be entered for potential repeated checking after an initial authentication phase, and this continuing verification throughout a computer session is referred to as dynamic verification [33].

A number of measurements or features are generally used to characterize an individual. For the keystroke biometric these measurements are typically key press duration (dwell) times, transition (latency) times, and the identity of the keys pressed. Stylometry typically uses statistical linguistic features at the word and syntax level.

The current work addresses some of the limitations of prior work on free-text biometric systems [53]. The current system has several unique aspects. First, it can collect raw keystroke data over the Internet as well as from a key logger IEEE 6th International Conference on Biometrics, BTAS 2013. on an individual machine. Second, it focuses on free-text input where sufficient keystroke data are available to permit the use of powerful statistical feature measurements – and the number, variety, and strength of the measurements used in the system are much greater than those used by earlier systems reported in the literature. Third, it focuses on applications using arbitrary text input because copy texts are unacceptable for most applications of interest. And, fourth, because of the statistical nature of the features and the use of arbitrary text input, special statistical procedures are incorporated into the system to handle the paucity of data from infrequently used keyboard keys.

Using an open biometric system approach, an earlier student authentication study was conducted on data obtained from students taking actual tests in a university course [51]. In contrast, this paper presents a closed biometric system approach to classification that significantly increases the performance reported in the earlier study. Also, to further analyze the stylometry component of the system, a separate

study on 30 book authors was undertaken to evaluate the stylometry performance on text lengths ranging from 250 to 10000 words. Additionally, because the mean population performance does not give the complete picture, the varied performance over the population of users was analyzed on the book-author study.

5.2. Features

The stylometry system uses a set of 228 linguistic features – 49 character-based, 13 word-based, and 166 syntax-based features, summarized in Appendix B. The features were normalized to be relatively independent of the text length – for example, the number of different words (vocabulary) / total number of words was used rather than simply the number of different words. The features were also chosen to show reasonable variation over a population of users – for example, some students use a large vocabulary and others a small one. As in the keystroke system, the features are standardized into the range 0-1.

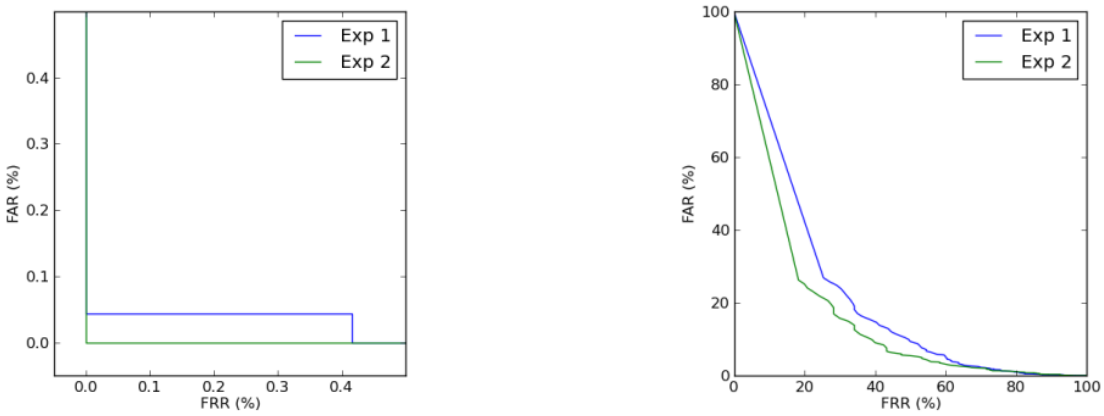
5.3. Experimental results: online test-takers

Two closed-system experiments were conducted on the data from the 30 students on each of the keystroke and stylometry systems using the leave-one-out procedure. Because the answers to the test questions could be short, several answers were combined to obtain reasonably sized biometric samples. In the first experiment, five test answers (half the test answers) were combined to obtain each sample, resulting in eight samples per student since each of the four tests contained ten questions for a total of 40 questions. In the second experiment, ten answers (all the answers of a test) were combined to obtain each sample, resulting in four samples per student. The experimental design and results are summarized in Table 5.1.

Experiment	Data Samples	Keystr EER	Stylo EER
1	8 samples/student 5 answers combined	0.04%	~26%
2	4 samples/student 10 answers combined	0.00%	~22%

Table 5.1.: Summary of experimental results

Figure 5.1 presents the ROC curves for the keystroke and stylometry systems for the two experiments. For both the keystroke and stylometry systems, performance improved in going from experiment 1 to experiment 2 with the doubling of the data sample size.



(a) Keystroke **(b)** Stylometry

Figure 5.1.: ROC curves for dataset A

5.4. Literature authorship

The stylometry results on the student tests were considered weak and the combined keystroke-stylometry system did not result in increased performance over that of the keystroke system alone. Therefore, considering that stylometry could require considerably more text input than keystroke analysis, a more extensive stylometry study was performed on short novels to determine system performance as a function of text length.

5.4.1. Data collection

Text samples, 10 from each of 30 authors for a total of 300 samples, were retrieved from Project Gutenberg [3]. The text samples were taken from books published between 1880 and 1930. This period was chosen based on the availability of books with expired copyrights and the period was restricted to fifty years to ensure that linguistic differences between authors would be more related to personal style than to the time of writing. The samples were not restricted geographically – authors were included from Great Britain, Ireland, and the United States. The samples from each author also span a variety of text types. For example, Oscar Wilde's samples include an essay, De Profundis, a novel, The Picture of Dorian Gray, and a play, The Importance of Being Earnest. All texts were longer than 5,000 words and originally written in English. The thirty authors wrote in various genres – fiction (8), action/adventure fiction (3), science fiction (1), British literature (6), mystery and thriller (3), classical literature (7), and horror (2), shown in Table 5.2.

The 300 text samples were cut into files of eleven different sizes (250, 500, 750, 1000, 1500, 2000, 2500, 3000, 4000, 5000, and 10000 words) in order to obtain

Author	Genre
Arnold Bennett, Thomas A. Janvier, Andrew Lang, L.M. Montgomery, Pelham Grenville Wodehouse, Annie Fellows Johnston, Louis Tracy, W.W. Jacobs	Fiction
Algernon Blackwood, Vernon Lee	Horror
John Buchan, H. Rider Haggard, Jack London	Action/Adventure
Edgar Rice Burroughs	Science Fiction
G.K. Chesterton, Joseph Conrad, Rudyard Kipling, George Bernard Shaw, Robert Louis Stevenson, Oscar Wilde	British Literature
Arthur Conan Doyle, Anna Katharine Green, Sax Rohmer	Mystery/Thriller
Bret Harte, Henry James, Frank R. Stockton, Mark Twain, H.G. Wells, Edith Wharton, Agnes C. Laut	Classic Literature

Table 5.2.: Overview of the 30 Authors

system performance as a function of text length. Of the 10000 word samples, 8 had slightly less than 10000 words due to the size of the original text file.

5.4.2. Experimental results

Figure 5.2 presents the ROC curves for the various sample lengths and Figure 5.3 shows the ERR as a function of the sample sizes in words. The EER was 8.5% for the 10K and 11.8% for the 5K word samples. As expected, performance gradually increased (lower EER) with increasing text length.

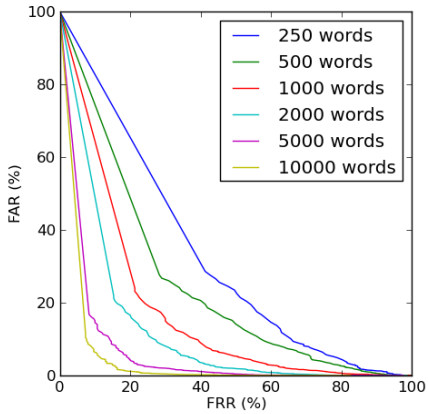


Figure 5.2.: Book stylometry ROC Curves, 30 authors

Because the mean population performance does not give the complete picture, the varied performance at the EER over the population of authors was analyzed and described using the biometric animal designations. The FRR of each individual user was analyzed in order to find users which have trouble authenticated as themselves (goats). A distinction between two different types of FAR must be made though.

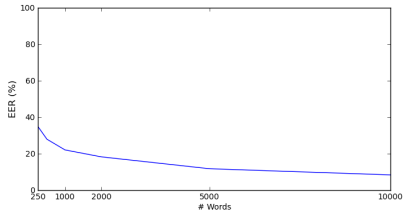


Figure 5.3.: EER as a function of sample sizes in words

When the true identity of the query sample is different from what is claimed during authentication, and a decision has been made to accept the query, then a false acceptance occurs. This false acceptance may contribute to either a weak template or a strong impostor. A distinction is made between the rate at which a template falsely accepts query samples and the rate at which an attacking query sample is falsely accepted. This distinction allows weak templates in the model to be identified (lambs), as well as attackers who may be skilled at imitating the identity of others (wolves). Over the author population, Figure 5.4 shows histograms of:

- FRR to identify those easily verified, sheep, and those difficult to verify, goats
- $FAR_{template}$ of how easily the true authors were imitated – identifying those easily attacked, lambs.
- $FAR_{attacker}$ of how easily imitators attacked true authors – identifying the strong attackers, wolves.

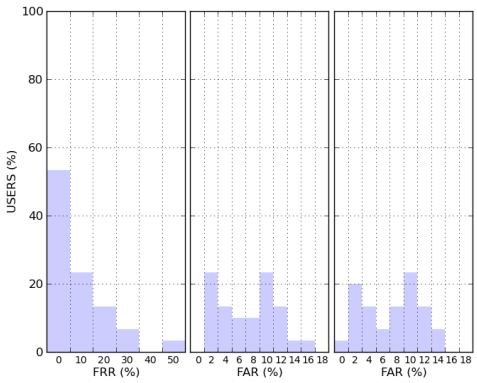


Figure 5.4.: Histograms: FRR (left), FAR of receivers (middle), FAR of attackers (right), over the 30 author 10000 word samples

Significant variation in performance over the population was demonstrated. For example, one author had 50% FRR (5 of 10 samples rejected) while all others had 30% or less (Figure 6 left). This author, Oscar Wilde, can be considered difficult to verify, a goat. Oscar Wilde's samples – which included an essay, a novel, and a play, as noted earlier – were not as homogeneous as those of the other authors.

5.5. Conclusion

The keystroke system performance results on the student test data were 100% on the 6000-keystroke full-test and 99.96% on the 3000-keystroke half-test samples. Although the results were obtained on a relatively small database, 30 students is a reasonable class size. These results were an improvement over the 99.45% performance on the 3000-keystroke half-test samples previously reported on the same data [51] Note that the leave-one-out procedure used in this study permitted the full test evaluations which were not possible using the procedure of the earlier study. High keystroke performance was anticipated in this study for such large volume keystroke input because high performance was also achieved in the earlier study on the same data [51] and a 98.3% performance was achieved on a 30-user, 750- keystroke-sample experiment in a recent study [39].

The performance of the keystroke biometric system is far superior to that of the stylometry one. While the keystroke and stylometry biometrics are both behavioral biometrics, they operate at different cognitive levels. The keystroke biometric operates at essentially an automatic motor control level. Stylometry, however, operates at a higher cognitive level, and because it primarily involves word and syntax-level units, much longer text passages are required relative to those required by the keystroke biometric.

To obtain system performance in this study we simulated the authentication process of many true users trying to get authenticated and of many zero-effort impostors trying to get authenticated as other users. Although authentication of online examination participants in real time would not be possible with the described technique due to the significant amount of input required (half or full test), delayed authentication with batch processing should be sufficient for university and HEOA requirements.

Important parameters in authorship attribution methods are the length and number of training and testing texts, and the number of potential authors [50]. Another important factor discovered in this stylometry study was the relationship between the texts under study and how the texts are produced. For example, in an earlier study it was discovered that a relatively strong correlation existed between the test answers and the test questions producing the answers [51]. Content-specific terminology inherent to the course subject matter, and used by a majority of the participants, confounded the results. Therefore, better performance results would likely be obtained from student essays on a variety of topics, as might be obtained from students in an English class, although two students who happen to choose the same or similar topic may present a problem.

Future work on improving stylometry in student examination applications might investigate the use of idiosyncratic features like the fraction of misspelled words, typing speed, and sequences of characters such that would be found in short words like “the” [18]. The use of longer text passages and those on different topics, such

as essays in English classes, might also be explored, as well as different ways of fusing the keystroke and stylometry results. Finally, while the student examination experiments reported here used actual test data, the authentication process itself was simulated, so future work might explore an actual authentication process in a student assessment environment.

6. Mouse

6.1. Overview

Current researches in the area of mouse biometrics include different approaches to obtaining data. These include using grid systems with predefined buttons that subjects would need to click, moving the mouse to follow a sequence of dots presented on a screen with feature extraction occurring from multiple iterations, and capturing mouse movements when users play specific video games like Solitaire and StarCraft. All these examples revolve around using specific applications or software to capture mouse movements. Additionally the feature extraction and authentication process is restricted to basic trajectory information like size, length, time, velocity and acceleration.

These earlier studies have been based on fixed patterns with data acquisition occurring on system-controlled actions performed by the user. The drawback of such systems is that the user cannot make natural mouse movements at his or her convenience.

This chapter will explore mouse biometrics typical of a user using a computer. No special hardware is used. The software that captures natural mouse movements runs in the background without interfering with the user experience. This should allow observation of mouse actions that occur in a natural unpredictable manner. Feature extraction is not limited to trajectories, but includes more generic actions such as mouse clicks, wheel spins, drag and drop movements, and highlighting text behavior. We hope these effects lead to a strong mouse biometric system.

6.2. Preprocessing

The frequency at which the position of the pointer is captured depends on the velocity of the pointer. When the pointer is not moving, no motion events will be generated. Thus, the sequence must be re-sampled at evenly spaced intervals in order to calculate certain key features, such as velocity and acceleration. Because of this, the coordinates in the motion sequence for each session are first re-sampled at a constant $100Hz$. A parameter, τ, sets a threshold on the amount of time between events which may be considered to have no motion. For example, consider two consecutive events, e_1 and e_2, which occur at time t_1 and t_2, respectively. Depending

on the time difference between e_1 and e_2, a new event or sequence of events is calculated according to Equation 6.1. In cases where the If $|t_1 - t_2|$ is above the threshold τ, the the coordinates of e_1 are padded up until e_2, creating a sequence of length $floor(|t_1 - t_2|/T) + 1$. If $|t_1 - t_2|$ is less than τ and still above the period of the sampling rate, T, then the points between e_1 and e_2 are linearly interpolated to create a new sequence. In cases where $|t_1 - t_2|$ is less than T, the mean point is used.

$$[e_n, e_{n+1}, ..., e_{n+floor(|t_1-t_2|/T)}] = \begin{cases} [e_1, e_1, ..., e_1] & if\ |t_1 - t_2| > \tau \\ interpolate(e_1, e_2) & if\ |t_1 - t_2| <= \tau\ \ and\ |t_1 - t_2| > T \\ [mean(e_1, e_2)] & if\ |t_1 - t_2| < T \end{cases} \tag{6.1}$$

The sampling rate was empirically chosen to be $100Hz$ to be great enough to minimize information loss from taking the mean location of consecutive events in the case of high sampling rate. Most of the experimental data was collected on standard desktops, with a clock resolution of approximately $\pm 10 - 15ms$. It was also chosen to be low enough that the samples don't become unnecessarily large.

The threshold τ was chosen so that consecutive events are not mistaken to be part of the same motion track. Since events are only generated when the mouse is moved, τ acts as a minimum velocity threshold.

After the motion sequence is re-sampled, it is passed through a low-pass filter, removing noise and smoothing the signal. A Hanning window of length 5 is used to calculate the new (x, y) coordinates of each event in the sequence, where the window function is defined by Equation 6.2.

$$w(n) = 0.5\left(1 - \cos\left(\frac{2\pi n}{N-1}\right)\right) \tag{6.2}$$

The point-to-point velocity is calculated by Equation 6.3 , where *dist* is a function which gives the Euclidean distance between two points and T is the period ($10ms$ for a $100Hz$ sampling rate).

$$v_i = dist(e_i, e_{i-1})/T \tag{6.3}$$

6.3. Sequence segmentation

The motion sequence by itself is relatively unstructured. In the behavioral biometric event model, each event has an (x, y) attribute which occurred at timestamp t, and

all event belong to a single class. In order to obtain a richer sequence of events, the motion event sequence is segmented so that each event approximately represents a single objective motion taken by the user. The purpose is to obtain a mapping from the motion sequence to a class of actions, where each event corresponds to a goal in the cognitive band of Newell's Time Scale of human action [41].

The segmentation is performed with a threshold technique similar to the eye movement segmentation in [15]. Events are considered to be part of a segment if the point-to-point velocity at the event is a threshold, according to Equation 6.4. The segments are then filtered to remove any segments which last less than a duration threshold. The segments removed are labeled as *noise*, usually occurring when a user is touching the mouse without moving it to another location on the screen. This is similar to a *gaze* in eye movement, versus a *saccade*, which occurs in between gazes. The algorithm to compute segments is found in Algorithm 6.1.

$$segment(e_n) = \begin{cases} true & if\ v_n > \alpha \\ false & otherwise \end{cases} \tag{6.4}$$

Algorithm 6.1 Sequence segmentation

Label each event with velocity above α as part of a segment.
Label each unique segment as consecutive segment events, separated by non-segments.
Keep segments which last longer than *dur*, where *dur* is a minimum time threshold.

The segmentation of a motion track is show in Figure 6.1. The shaded areas show movement (saccades) while the non-shaded areas show fixations.

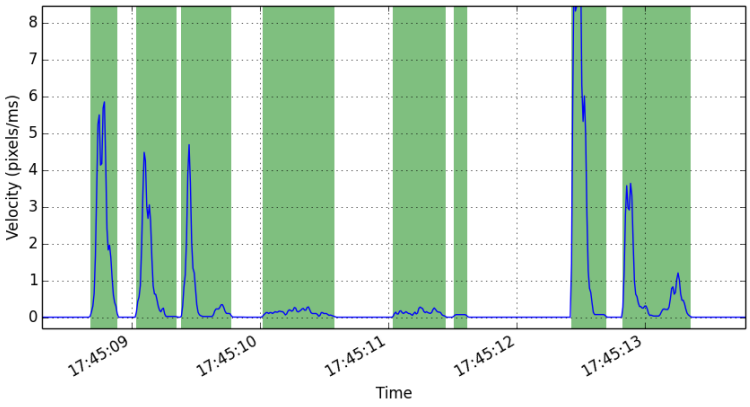


Figure 6.1.: Velocity segmentation

The segments can now be discretized in order to determine the event class. A class label is assigned to each segment according to its distance and direction. Similarly to

[48], the combination of distance and direction labels uniquely identify the segment, depending on which bin the segment falls into. Let e_n now be a segment event in a sequence of segments. The function $dist(e)$ gives the pixel distance from the first point in the segment to the last point. The function $dire(e)$ gives the direction of the segment, computed as an angle in $[0°, 360°)$. The number of distance and direction bins are both parameters to the segment class labeling algorithm, given as $N_{distance\,classes}$ in Equation 6.5 and $N_{direction\,classes}$ in Equation 6.6 respectively. The maximum distance, δ, is a normalizing parameter. Since the display and resolution may vary across different machines, δ should be chosen large enough to cover most trajectories. Any segments with a distance above δ are labeled with the largest distance class.

$$e_{distance\,class} = \begin{cases} N_{distance\,classes} - 1 & if\ dist(e) > \delta \\ floor(N_{distance\,classes} \times dist(e)/\delta & otherwise \end{cases} \tag{6.5}$$

$$e_{direction\,class} = floor(N_{direction\,classes} \times dire(e)/\phi \tag{6.6}$$

Each event in the sequence now contains:

- A unique class given by the distance and direction of the segment
- A sequence of timestamp-coordinates for each point in the segment, $c_i = \{(t_i, x_i, y_i)\, where\, i \in [1...n]\}$

Segments class frequencies determined by domain to a certain extent.

Figure 6.2 shows the segments from two different users performing the same task, with 4 direction classes and 8 distance classes.

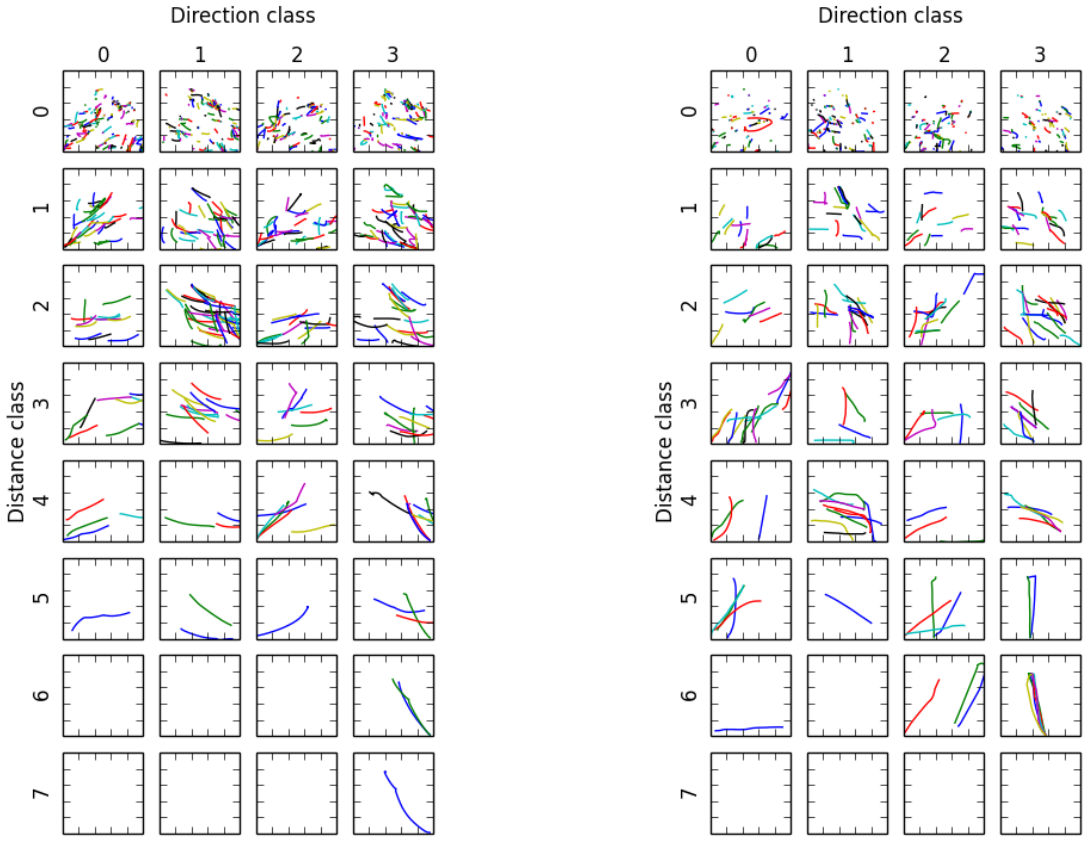
6.4. Features

6.4.1. Motion

A set of motion features was developed with the intention of being representative of a user's mouse movement over the course of a session, with as little dependency on the context of the session as possible. These include:

Duration The time of the trajectory is the total time taken to complete the trajectory from start to end point. This is the difference in the timestamp obtained at the start x-y coordinate and the timestamp obtained at the end x-y coordinate of the curve.

$$dur = t_n - t_1$$



(a) User A in dataset B2

(b) User B in dataset B2

Figure 6.2.: Segments from different users performing the same task

Distance The distance is the Euclidean distance between the start x-y coordinate and the end x-y coordinate.

$$dist = \|(x_n - x_1), (y_n - y_1)\|$$

Velocity The velocity of the segment is the ratio of the distance traveled to the time taken, given in pixels/ms.

$$v = dist/dur$$

Length The length of the segment is the total distance traveled, or the sum of distances between every x-y coordinate.

$$length = \sum_{i=2}^{n} dist_i$$

Curviness The curviness of the trajectory is the length of the trajectory divided by the distance between the first and last points. However, to avoid division by zero if the curve ends where it starts (like in drawing an 'O'), the divisor is limited to a small number, ie. 1/40 of a typical screen width.

$$cur = length/distance$$

Energy The most efficient path to get from the location of the first point in the sequence to the last point is a straight line which passes through both points. In order to quantify the shape of the segment, the area between the line and the segment is calculated. At each point on the segment, the distance between the point and the intersecting point of the normal line and perfect line is calculated. These distances are summed to approximate an integral, or the total area covered between the segment and the perfect line.

Inflection points The number of inflection points in the trajectory is the number of changes between curvatures (clockwise to counterclockwise or vice versa). For example, a typical drawing of the letter O, either clockwise or counterclockwise, would have no inflection points, whereas a typical drawing of the letter S would have one (from counterclockwise to clockwise). This can be found by detecting changes in sign in the angular velocity at each point.

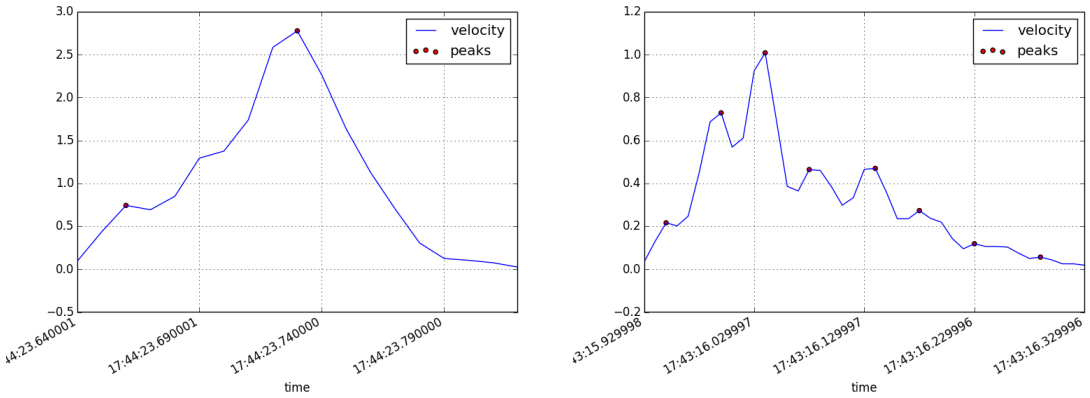
$$infl = |\{dir_i \times dir_{i-1} < 0 \, where \, i \in [2...n]\}|$$

Smoothness The smoothness can be determined by number of local peaks in the point-to-point velocity of the segment. Segments which represent a very "smooth" motion will have a small number of peaks, as the initial acceleration and deceleration are applied. Segments with a "rough" motion may have many velocity peaks, as the velocity continues to increase and decrease several times over a segment. The velocity of a smooth segment and peaks can be seen in 6.3a, while the velocity of a rough segment is show in 6.3b.

$$vp = |\{v_{i-1} < v_i > v_{i+1} \, where \, i \in [2...n-1]\}|$$

Radius In physics, the radius of the path formed by a moving object can be determined by the velocity and the angular velocity of the object. To approximate the radius about which the pointer is revolving, the ratio of the mean point-to-point velocity and mean point-to-point angular velocity is taken.

$$r = \frac{\overline{v}_i}{\overline{av}_i}$$



(a) Smooth segment **(b)** Rough segment

Figure 6.3.: Velocity peaks in smooth and rough segments

The distribution of several point-to-point measurements are also considered. Each measure is taken at every point in the segment. Several descriptive statistics are taken on the distribution, creating an addition 3 features for each function: mean, median, and standard deviation.

Point-to-point distance The point-to-point distances are the distance between consecutive pair of points in the sequence

$$dist_i = \{\|(x_i - x_{i-1}), (y_i - y_{i-1})\| \, where \, i \in [2...n]\}$$

Point-to-point direction The direction at each point is calculated by taking the arctangent of the slope of each consecutive point-pair

$$dir_i = \{arctan((y_i - y_{i-1})/(x_i - x_{i-1})) \, where \, i \in [2...n]\}$$

Point-to-point velocity The point-to-point velocity is given in pixels/ms. Note that $(t_i - t_{i-1}) = T$ since each point is sampled at regular intervals.

$$v_i = \{dist_i/(t_i - t_{i-1}) \, where \, i \in [2...n]\}$$

Point-to-point acceleration The acceleration is the change in velocity between consecutive point-pairs.

$$a_i = \{(v_i - v_{i-1})/(t_i - t_{i-1}) \, where \, i \in [2...n-1]\}$$

Point-to-point angular velocity Angular velocity between each consecutive point-pair is the change in direction, given in rad/ms.

$$av_i = \{(dir_i - dir_{i-1})/(t_i - t_{i-1}) \, where \, i \in [2...n]\}$$

Point-to-point angular acceleration The angular acceleration is calculated similarly to acceleration, by taking the change in angular velocity between point-pairs.

$$aa_i = \{(av_i - av_{i-1})/(t_i - t_{i-1})\, where\, i \in [2...n-1]\}$$

Each of the features above are taken for each event class, where the number of event classes is given by $N_{distance\,classes} \times N_{direction\,classes}$. The frequencies of individual event are also be used as features. Since these are strictly holistic features, it is possible that they may be dependent on the context of the user application (ie. editing a paragraph may invoke different user mouse behavior than playing an online game).

6.4.2. Click

Detecting double click events to avoid being system-dependent.

Mouse click features

Each click event is generated when the user presses the left or right buttons on the mouse. Along with the button and the timestamps of the press and release, the event contains the pointer coordinates at both the press and release of the button.

The events in the click event sequence are placed into one of three different categories: single click, double click, or drag-and-drop. This corresponds to three commonly occurring interactions involving the mouse buttons.

Double clicks are first found by taking the transition between press times of each event. The default timing threshold between click events on Windows is 500ms (ie. click events which occur within 500ms of each other will generate a double click system event). In order to capture all double click events, event pairs where the transition between press timestamps is less than 1000ms are considered to be double clicks.

Drag-and-drop events are found by taking the distance between the press and release coordinates of the click event. Events where the distance between the press and release coordinates is greater than 5 pixels are considered to be drag and drops. This value was determined to avoid including single clicks which contain small amounts of motion between the press and release. This type of motion is called *jitter*.

Events which have not been labeled as either double click or drag-and-drop actions are assumed to be single clicks.

The following features are taken on the action-labeled click events:

Single click and drag features

Duration The time between the button press and release

Transition The time between consecutive event presses

Distance The Euclidean distance between the press coordinate and release coordinate

Velocity The velocity of the action, given in pixels/ms. Note that for single clicks, the velocity normally 0 since no motion is observed.

Rate The number of actions normalized to the session duration

Double clicks

First duration The duration of the first click in the double click action

Second duration The duration of the second click in the double click action

Transitions Type 1, 2, 3, and 4 transition times of the double click (see keystroke transition times)

Jitter Distance between the press and release coordinates within each click

Motion Distance between the press of the first click and release of the second click

Rate The number of double click events normalized to the session duration

The ratios of double clicks to single clicks, drags to single clicks, and drags to double clicks are also taken, for a total of 38 features for each button. The features are taken for the left and right mouse button, and for both buttons, creating a total of 102 features altogether. Note that for users who rarely use the right mouse button, many of the right mouse button features will be zero. It is also important to consider that many of these features may become system-dependent in the case of different hardware. The experimental datasets used mostly homogeneous hardware to minimize environmental effects.

6.4.3. Scroll

System dependent

Mouse wheel scroll features

Scroll events are generated when a user moves the wheel of a mouse past a threshold. Each event contains a timestamp, the direction and amount of rotation, and the location of the pointer on screen.

The scroll event sequence is first segmented similar to the motion event sequence. The velocity of the wheel is calculated by Equation 6.7, where $scroll\,distance(\cdot)$ is a function which gives the scroll distance between two events by multiplying $direction = \{1, -1\}$ and $amount$, which is generally in the range $(-3, 3)$.

$$v_i = scroll\,distance(e_i, e_{i-1})/(t_i - t_{i-1}) \tag{6.7}$$

Boundaries of scroll segments are found where the velocity falls below a threshold, experimentally determined to be $v_{thresh} = 0.01$. Changes in direction also mark

segment boundaries. The value was determined to create segments which were believed to originate from one finger motion (ie. moving the mouse wheel without lifting the finger up or changing directions).

Three sets of segments are considered: scroll-up segments, scroll-down segments, and the set of all segments. For each set, the mean and standard deviations of the following features are taken:

Velocity The mean velocity of each segment is found by taking the mean point-to-point velocity at each event in the segment.

Duration The duration of the segment is the time from the first event to the last event.

Pointer motion The pointer motion is calculated by taking the point-to-point Euclidean distance between mouse pointer coordinates. The total distance traveled during the scroll segment is the sum of each point-to-point motion distance.

Scroll distance The scroll distance is the sum of scroll amounts in each event.

Time between segments The transition time between segments is taken as the time between the first event in each segment (similar to a type 2 keystroke transition).

Inter-segment pointer motion The mouse pointer distance is calculated between segments by taking the Euclidean distances between the first event of each segment.

Number of segments The number of unique segments found in the session.

Number of long segments The number of segments containing at least 2 events.

Number of short segments The number of segments containing only 1 event.

The resulting feature vectors contain 45 values, from the 15 features above, (mean and std for 6 of the above features), and 3 sets of segments. Some of the features may become system dependent, especially for data collected within a web browser. There is no industry standard for the scroll amount and direction, so care must be taken to avoid environmental dependencies. The experimental datasets used in this chapter were collected with homogenous systems to determine the performance of the model.

6.5. Experimental results

6.5.1. Fixed motion tasks

Dataset **C** was used to get baseline results from a dataset with minimal environmental factors. Since each user was directed to perform a certain task for data

collection, the results of the motion feature set on this data are not ideal (since some of the features are based on the presence or absence of certain motions a user might perform). Figure 6.4 shows the ROC curve for the motion-only feature set on **C**. The EER was found to be 11.6% and the error rate of each user was roughly the same at the EER.

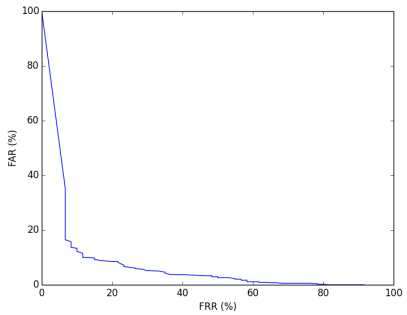


Figure 6.4.: ROC curve for motion features on **C**

6.5.2. Unrestrained motion in a single domain

The performance of the motion features described in this chapter were tested on datasets **B1**, **B2**, **B3**, and **B4**, which consist of unrestrained motion for 4 different tasks. Table 6.1 shows the EER for a reduced set of 10 users in each dataset to compare to the results of the baseline experiment. The EER was also found with the maximum number of users available, to give an indication of how well the motions features scale with population size.

Dataset	Description	Users	EER
B1	Edit paragraph	10	23%
		46	32%
B2	Web browsing	10	18%
		40	31%
B3	Solitaire	10	17%
		19	16%
B4	Star bubbles	10	15%
		21	19%

Table 6.1.: Baseline motion EER

Since each of the 4 datasets contain different users, the intersection of users across the 4 datasets was taken to get users who completed 6 sessions for each task. This set of 16 common users are used in all of the following experiments.

The EER of motion-only features was found for the common users in each dataset. The scroll, click, and keystroke features were then introduced to create multimodal feature vectors. The performance of motion only vs. multimodal features is found in Table 6.2. For the 4 experiments, the leave-one-out cross validation method was used to validate the model, using feature vectors from a single dataset (or a single task).

Dataset	Description	Motion EER	Multimodal EER
B1	Edit paragraph	24%	14%
B2	Web browsing	20%	7%
B3	Solitaire	17%	6%
B4	Star bubbles	21%	7%
Average		20.5%	8.5%
Std dev		2.5%	3.2%

Table 6.2.: 16 common users motion and multimodal performance in a single task

6.5.3. Unrestrained motion in multiple domains

To test the robustness of the motion and multimodal features across multiple domains, the authentication process was simulated using two different tasks as reference sets and query sets, respectively. For each experiment, the feature vectors from the query dataset are used to authenticate users who have enrolled sessions in the reference dataset. The set of 16 common users was used so that each user had 6 sessions enrolled from each task. The EER of each of the 24 experiments are found in Table 6.3.

Finally, the system was tested using all of the enrolled sessions from each of the 16 common each user across 4 different tasks. This is the most realistic scenario, since the data from each task is used. For each authentication, a user's feature vector is left out and compared to the remaining feature vectors from that task, as well as the feature vectors enrolled from every other task. The ROC curve and error rates as a function of m are shown in Figure 6.5, with an EER of 3.9%.

6.6. Conclusion

Using only motion features on unrestrained input resulted in an error rate which was roughly twice as much as a fixed input task. This shows the difficulty in quantifying unrestrained mouse input, in contrast to the modest increases in error rate when dealing with unrestrained keystroke input. Though the input from the mouse is generally observed with e higher frequency than keystroke, an authentication system which relies on motion only will generally perform worse than keystroke. It may

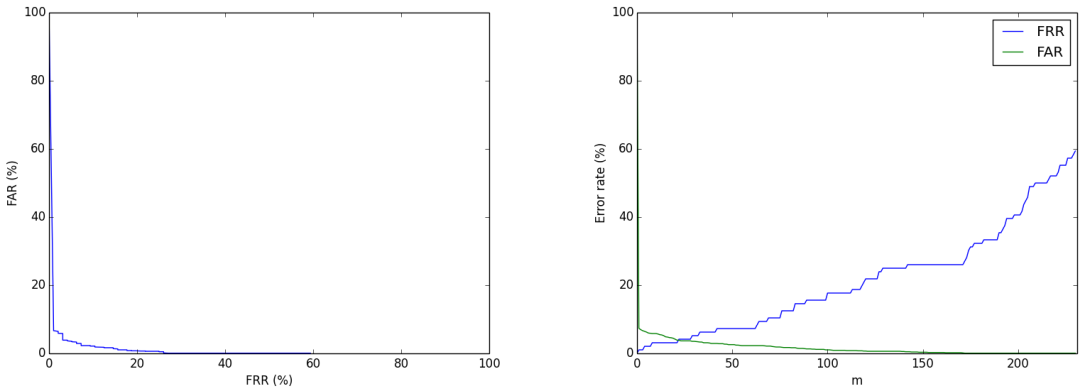
Reference	Query	Motion EER	Multimodal EER
B1	**B2**	39%	47%
B1	**B3**	43%	44%
B1	**B4**	43%	43%
B2	**B1**	37%	44%
B2	**B3**	36%	36%
B2	**B4**	41%	32%
B3	**B1**	42%	43%
B3	**B2**	41%	38%
B3	**B4**	22%	22%
B4	**B1**	38%	41%
B4	**B2**	39%	36%
B4	**B3**	32%	27%
Average		37.75%	37.75%
Std dev		5.6%	7.3%

Table 6.3.: Motion and multimodal performance across multiple domains

be the case that the atomic actions that a user intends when interacting with the application actually occur with a much lower frequency. In this case, events (actions) would be the movement from one location on the screen to another. In addition to this, the motion track contains more noise than the keystroke biometric.

Combining mouse and keystroke biometrics resulted in moderate performance on unrestrained input, with an EER of 3.9% when all of a user's sessions from multiple tasks are considered. Trying to authenticate a user with data enrolled from heterogenous tasks may not be possible with this approach, as the average EER in this case was found to be 37.75%. Similar tasks (using sessions from each of the two games, **B3** and **B4**) did perform better than average, with the EER ranging from 20-30% with motion and multimodal features. This might suggest that in certain scenarios, it may be possible to authenticate a user performing novel tasks with the mouse. The EER was also more consistent when considering only motion across heterogenous domains, with a standard deviation of 5.6% versus 7.3% for multimodal features. This is an indication that features besides motion may be less reliable when trying to authenticate a user performing novel tasks.

Future work should better identify the features which perform well across multiple domains. A more diverse dataset with a larger population is desirable, since the number of enrolled sessions from users performing various tasks was limited in this study. Simple concatetaning the feature vectors from multiple biometris is also not ideal. Combining multiple biometrics when authenticating users in different domains is also a topic of interest.



(a) ROC curve **(b)** FRR and FAR as a function of m

Figure 6.5.: Common user multimodal performance for all tasks

Acknowledgments

We thank the student teams in the masters level projects course that contributed to this effort over recent years

A. Keystroke features

Feature	Measure	Feature Measured	Feature	Measure	Feature Measured
1-2	μ & σ	dur all keystrokes	131-32	μ & σ	tran1 letter/non-letter
3-4	μ & σ	dur all alphabet letters	133-34	μ & σ	tran1 letter/space
5-6	μ & σ	dur vowels	135-36	μ & σ	tran1 letter/punct
7-8	μ & σ	dur vowels a	137-38	μ & σ	tran1 non-letter/letter
9-10	μ & σ	dur vowels e	139-40	μ & σ	tran1 shift/letter
11-12	μ & σ	dur vowels i	141-42	μ & σ	tran1 space/letter
13-14	μ & σ	dur vowels o	143-44	μ & σ	tran1 non-letter/non-letter
15-16	μ & σ	dur vowels u	145-46	μ & σ	tran1 space/shift
17-18	μ & σ	dur freq cons	147-48	μ & σ	tran1 punct/space
19-20	μ & σ	dur freq cons t	149-50	μ & σ	tran2 any-key/any-key
21-22	μ & σ	dur freq cons n	151-52	μ & σ	tran2 letter/letter
23-24	μ & σ	dur freq cons s	153-54	μ & σ	tran2 top cons pairs
25-26	μ & σ	dur freq cons r	155-56	μ & σ	tran2 top cons pairs th
27-28	μ & σ	dur freq cons h	157-58	μ & σ	tran2 top cons pairs st
29-30	μ & σ	dur next freq cons	159-60	μ & σ	tran2 top cons pairs nd
31-32	μ & σ	dur next freq cons l	161-62	μ & σ	tran2 vowel/cons
33-34	μ & σ	dur next freq cons d	163-64	μ & σ	tran2 vowel/cons an
35-36	μ & σ	dur next freq cons c	165-66	μ & σ	tran2 vowel/cons in
37-38	μ & σ	dur next freq cons p	167-68	μ & σ	tran2 vowel/cons er
39-40	μ & σ	dur next freq cons f	169-70	μ & σ	tran2 vowel/cons es
41-42	μ & σ	dur least freq cons	171-72	μ & σ	tran2 vowel/cons on
43-44	μ & σ	dur least freq cons m	173-74	μ & σ	tran2 vowel/cons at
45-46	μ & σ	dur least freq cons w	175-76	μ & σ	tran2 vowel/cons en
47-48	μ & σ	dur least freq cons y	177-78	μ & σ	tran2 vowel/cons or
49-50	μ & σ	dur least freq cons b	179-80	μ & σ	tran2 cons/vowel
51-52	μ & σ	dur least freq cons g	181-82	μ & σ	tran2 cons/vowel he
53-54	μ & σ	dur least freq cons other	183-84	μ & σ	tran2 cons/vowel re
55-56	μ & σ	dur all left hand letters	185-86	μ & σ	tran2 cons/vowel ti
57-58	μ & σ	dur all right hand letters	187-88	μ & σ	tran2 vowel/vowel
59-60	μ & σ	dur non-letters	189-90	μ & σ	tran2 vowel/vowel ea
61-62	μ & σ	dur space	191-92	μ & σ	tran2 double letters
63-64	μ & σ	dur shift	193-94	μ & σ	tran2 left/left
65-66	μ & σ	dur punctuation	195-96	μ & σ	tran2 left/right
67-68	μ & σ	dur punctuation period .	197-98	μ & σ	tran2 right/left
69-70	μ & σ	dur punctuation comma ,	199-200	μ & σ	tran right/right
71-72	μ & σ	dur punctuation apost '	201-02	μ & σ	tran2 letter/non-letter
73-74	μ & σ	dur punctuation other	203-04	μ & σ	tran2 letter/space
75-76	μ & σ	dur numbers	205-06	μ & σ	tran2 letter/punct
77-78	μ & σ	dur other	2070-8	μ & σ	tran2 non-letter/letter
79-80	μ & σ	tran1 any-key/any-key	209-10	μ & σ	tran2 shift/letter
81-82	μ & σ	tran1 letter/letter	211-12	μ & σ	tran2 space/letter
83-84	μ & σ	tran1 top cons pairs	213-14	μ & σ	tran2 non-letter/non-letter
85-86	μ & σ	tran1 top cons pairs th	215-16	μ & σ	tran2 space/shift
87-88	μ & σ	tran1 top cons pairs st	217-18	μ & σ	tran2 punct/space
89-90	μ & σ	tran1 top cons pairs nd	219	%	shift
91-92	μ & σ	tran1 vowel/cons	220	%	caps lock
93-94	μ & σ	tran1 vowel/cons an	221	%	space
95-96	μ & σ	tran1 vowel/cons in	222	%	backspace
97-98	μ & σ	tran1 vowel/cons er	223	%	delete
99-100	μ & σ	tran1 vowel/cons es	224	%	insert
101-02	μ & σ	tran1 vowel/cons on	225	%	home
103-04	μ & σ	tran1 vowel/cons at	226	%	end
105-06	μ & σ	tran1 vowel/cons en	227	%	enter
107-08	μ & σ	tran1 vowel/cons or	228	%	ctl
109-10	μ & σ	tran1 cons/vowel	229	%	four arrow keys combined
111-12	μ & σ	tran1 cons/vowel he	230	%	sentence ending punct .?!
113-14	μ & σ	tran1 cons/vowel re	231	%	other punct
115-16	μ & σ	tran1 cons/vowel ti	232	%	left shift
117-18	μ & σ	tran1 vowel/vowel	233	%	right shift
119-20	μ & σ	tran1 vowel/vowel ea	234	%	left mouse click
121-22	μ & σ	tran1 double letters	235	%	right mouse click
123-24	μ & σ	tran1 left/left	236	%	double left mouse click
125-26	μ & σ	tran1 left/right	237	%	left shift to right shift
127-28	μ & σ	tran1 right/left	238	rate	input rate with pauses
129-30	μ & σ	tran1 right/right	239	rate	input rate w/o pauses

Table A.1.: Keystroke features

B. Stylometry features

Character-based features:
1. number of alphabetic characters/number of characters
2. number of uppercase alphabetic characters/ number of alphabetic char
3. number of digit characters/number of characters
4. number of space characters/number of characters
5. number of vowel (a,e,i,o,u) characters/number of alphabetic characters
6. number of "a" (upper or lowercase) characters/number of vowel char
7. number of "e" characters/number of vowel characters
8. number of "i" characters/number of vowel characters
9. number of "o" characters/number of vowel characters
10. number of "u" characters/number of vowel characters
11. number of most frequent consonants (t,n,s,r,h)/number of alph char
12. number of "t" characters/number of (t,n,s,r,h)
13. number of "n" characters/number of (t,n,s,r,h)
14. number of "s" characters/number of (t,n,s,r,h)
15. number of "r" characters/number of (t,n,s,r,h)
16. number of "h" characters/number of (t,n,s,r,h)
17. number 2nd most frequent consonants (l,d,c,p,f)/number of alph char
18. number of "l" characters/number of (l,d,c,p,f)
19. number of "d" characters/number of (l,d,c,p,f)
20. number of "c" characters/number of (l,d,c,p,f)
21. number of "p" characters/number of (l,d,c,p,f)
22. number of "f" characters/number of (l,d,c,p,f)
23. number 3rd most frequent consonants (m,w,y,b,g)/number of alph char
24. number of "m" characters/number of (m,w,y,b,g)
25. number of "w" characters/number of (m,w,y,b,g)
26. number of "y" characters/number of (m,w,y,b,g)
27. number of "b" characters/number of (m,w,y,b,g)
28. number of "g" characters/number of (m,w,y,b,g)
29. number of least frequent consonants (j,k,q,v,x,z) / number of alph char
30. number of consonant-consonant digrams/number alph digrams
31. number of "th" digrams/consonant-consonant digrams
32. number of "st" digrams/number consonant-consonant digrams
33.number of "nd" digrams/number consonant-consonant digrams
34. number of vowel-consonant digrams/number alph digrams
35. number of "an" digrams/number of vowel-consonant digrams
36. number of "in" digrams/number of vowel-consonant digrams
37. number of "er" digrams/number of vowel-consonant digrams
38. number of "es" digrams/number of vowel-consonant digrams
39. number of "on" digrams/number of vowel-consonant digrams
40. number of "at" digrams/number of vowel-consonant digrams
41. number of "en" digrams/number of vowel-consonant digrams
42. number of "or" digrams/number of vowel-consonant digrams
43. number of consonant-vowel digrams/number of alphabet digrams
44. number of "he" digrams/number of consonant-vowel digrams
45. number of "re" digrams/number of consonant-vowel digrams
46. number of "ti" digrams/number of consonant-vowel digrams
47. number of vowel-vowel digrams/number of alphabet letter digrams
48. number of "ea" digrams/number of vowel-vowel digrams
49. number of double-letter digrams/number of alphabet letter digrams
Word-based features:
1. number of one-letter words/number of words
2. number of two-letter words/number of words
3. number of three-letter words/number of words
4. number of four-letter words/number of words
5. number of five-letter words/number of words
6. number of six-letter words/number of words
7. number of seven-letter words/number of words
8. number of long words (eight or more letters)/number of words
9. number of short words (one to three letters)/number of words
10. average word length = number letters in all words/number of words
11. number of different words (vocabulary)/number of words
12. number of words occurring once/number of words
13. number of words occurring twice/number of words
Syntax-based features:
1. number of the eight punctuation symbols (.,?!;:"')/number of char
2. number of periods (.)/number of the eight punctuation symbols
3. number of commas (,)/number of the eight punctuation symbols
4. number of "?" and "!"/number of the eight punctuation symbols
5. number of semicolons (;) and colons (:)/number punctuation symbols
6. number of single (') and double quotes (")/punctuation symbols
7. number of non-alphabetic, non-punctuation, and non-space characters (0,1,2,3,4,5,6,7,8,9,@,#,$,%,etc.)/number of characters

8. number of digit char/number of non-alph, non-punct, and non-space char
9. number of common conjunctions/number of words
10. number of common interrogatives/number of words
11. number of common prepositions/number of words
12. number of first-person personal pronouns/number of personal pronouns
13. number of 2nd-person personal pronouns/number personal pronouns
14. number of 3rd-person personal pronouns/number of personal pronouns
15. number of personal pronouns (from above)/number of words
16. number of common nouns number of words
17. number of common verbs/number of words
18. number of common auxiliary verbs/number of words
19. number of "can" words/number of common auxiliary verbs
20. number of "did", "do", "does" words/number of common auxiliary verbs
21. number of "had", "has", "have" words/number of common auxiliary verbs
22. number of could, should, would/number of common auxiliary verbs
23. number of "will" words/number of common auxiliary verbs
24. number of common auxiliary verbs/number of common verbs
25. number to-be verbs (am,are,be,been,being,is,was,were)/number words
26. number of to-be verbs/number of common verbs
27. number of "am" words/number of to-be verbs
28. number of "are" words/number of to-be verbs
29. number of "be", "been", and "being" words/number of to-be verbs
30. number of "is" words/number of to-be verbs
31. number of "was" words/number of to-be verbs
32. number of "were" words/number of to-be verbs
33. number of common adjectives/number of words
34. number of articles (a, an, the)/number of words
35. number of articles (a, an, the)/number of common adjectives
36. number of "the" articles/number of articles
37. number of "a" or "an" articles/number of articles
38. number of indefinite personal pronouns/number of words
39. number of determiners/number of words
40-64. number of each of 25 most common words/number of words
65-164. number of each of 100 most common words /number of most common words of that major category (e.g., number 'the'/num adjectives)
165. average number of characters per sentence
166. average number of words per sentence

Table B.1.: Stylometry features

C. Linear regression fallback functions

	First choice	Second choice	Third Choice	Fourth Choice
A	$f_{E,A}(x) = 0.96x + 18.20$	$f_{S,A}(x) = 0.90x + 19.71$	$f_{D,A}(x) = 0.85x + 30.26$	$f_{W,A}(x) = 0.86x + 26.19$
B	$f_{G,B}(x) = 0.78x + 17.02$	$f_{M,B}(x) = 0.79x + 13.89$	$f_{O,B}(x) = 0.65x + 22.56$	$f_{N,B}(x) = 0.73x + 17.91$
C	$f_{D,C}(x) = 0.91x + 8.73$	$f_{S,C}(x) = 0.84x + 10.28$	$f_{E,C}(x) = 0.89x + 9.42$	$f_{G,C}(x) = 0.91x + 19.26$
D	$f_{E,D}(x) = 0.96x + 2.71$	$f_{S,D}(x) = 0.89x + 5.51$	$f_{C,D}(x) = 0.80x + 21.19$	$f_{F,D}(x) = 1.01x + 6.01$
E	$f_{D,E}(x) = 0.86x + 16.18$	$f_{S,E}(x) = 0.84x + 11.95$	$f_{R,E}(x) = 0.85x + 16.45$	$f_{W,E}(x) = 0.82x + 16.31$
F	$f_{E,F}(x) = 0.74x + 19.36$	$f_{G,F}(x) = 0.79x + 24.30$	$f_{D,F}(x) = 0.68x + 26.19$	$f_{T,F}(x) = 0.67x + 31.34$
G	$f_{T,G}(x) = 0.75x + 18.40$	$f_{E,G}(x) = 0.79x + 9.81$	$f_{F,G}(x) = 0.88x + 6.94$	$f_{V,G}(x) = 0.77x + 19.49$
H	$f_{N,H}(x) = 0.92x + 4.07$	$f_{O,H}(x) = 0.81x + 11.14$	$f_{U,H}(x) = 1.01x + 1.58$	$f_{I,H}(x) = 0.78x + 18.95$
I	$f_{O,I}(x) = 0.87x + 6.74$	$f_{H,I}(x) = 0.92x + 9.45$	$f_{N,I}(x) = 0.95x + 2.87$	$f_{K,I}(x) = 0.91x + 8.93$
J	$f_{K,J}(x) = 0.47x + 39.99$	$f_{B,J}(x) = 0.63x + 31.69$	$f_{P,J}(x) = 0.53x + 36.09$	$f_{I,J}(x) = 0.39x + 46.33$
K	$f_{J,K}(x) = 1.52x + -33.27$	$f_{M,K}(x) = 0.87x + 10.48$	$f_{L,K}(x) = 0.84x + 13.11$	$f_{I,K}(x) = 0.74x + 23.22$
L	$f_{O,L}(x) = 0.76x + 18.97$	$f_{N,L}(x) = 0.83x + 16.24$	$f_{K,L}(x) = 0.80x + 19.74$	$f_{I,L}(x) = 0.71x + 28.52$
M	$f_{N,M}(x) = 0.83x + 15.18$	$f_{O,M}(x) = 0.71x + 23.58$	$f_{U,M}(x) = 0.88x + 16.13$	$f_{H,M}(x) = 0.76x + 25.10$
N	$f_{H,N}(x) = 0.86x + 17.12$	$f_{M,N}(x) = 0.93x + 7.66$	$f_{O,N}(x) = 0.76x + 19.90$	$f_{I,N}(x) = 0.75x + 25.33$
O	$f_{H,O}(x) = 0.96x + 12.77$	$f_{I,O}(x) = 0.87x + 19.01$	$f_{U,O}(x) = 1.10x + 2.37$	$f_{L,O}(x) = 0.96x + 8.91$
P	$f_{O,P}(x) = 0.77x + 18.86$	$f_{U,P}(x) = 0.92x + 13.95$	$f_{L,P}(x) = 0.79x + 21.20$	$f_{K,P}(x) = 0.83x + 19.46$
R	$f_{E,R}(x) = 0.88x + 11.41$	$f_{T,R}(x) = 0.81x + 24.46$	$f_{G,R}(x) = 0.87x + 23.42$	$f_{F,R}(x) = 0.92x + 14.45$
S	$f_{E,S}(x) = 0.96x + 10.20$	$f_{D,S}(x) = 0.90x + 17.53$	$f_{A,S}(x) = 0.78x + 18.61$	$f_{C,s}(x) = 0.82x + 26.58$
T	$f_{R,T}(x) = 0.90x + 5.21$	$f_{E,T}(x) = 0.91x + 2.79$	$f_{G,T}(x) = 0.96x + 9.74$	$f_{D,T}(x) = 0.82x + 13.69$
U	$f_{H,U}(x) = 0.74x + 21.41$	$f_{O,U}(x) = 0.68x + 21.27$	$f_{N,U}(x) = 0.74x + 18.33$	$f_{M,U}(x) = 0.78x + 15.74$
V	$f_{G,V}(x) = 0.86x + 14.84$	$f_{F,V}(x) = 0.91x + 6.72$	$f_{E,V}(x) = 0.80x + 11.53$	$f_{T,V}(x) = 0.80x + 18.15$
W	$f_{E,W}(x) = 0.89x + 14.72$	$f_{S,W}(x) = 0.81x + 18.68$	$f_{D,W}(x) = 0.80x + 25.74$	$f_{R,W}(x) = 0.83x + 22.07$
Y	$f_{H,Y}(x) = 0.79x + 17.78$	$f_{N,Y}(x) = 0.81x + 12.32$	$f_{I,Y}(x) = 0.72x + 23.10$	$f_{O,Y}(x) = 0.69x + 20.76$

Table C.1.: Linear regression functions for each key

D. Summary of Prior Authorship Attribution Stylometry Studies

Author - Year	#Subjects	Samples/Subj	Sample Size	Feature Types	#Features	Classification	Accuracy
Afroz, et. al. 2012	68	documents	500 words	Lexical, Syntactic, Content	707	SVM	96.6%
Alison & Guthrie 2008	9	174-706 Emails	~75 words	Bigrams, Trigrams Word frequency	Not given	Multimodal Hierarchical SVM	78.46% 87.05% 86.74%
Christani, et. al. 2012	77	60-100 IMs	615 words avg	Lex, Syntactic, Struct, Topic	Not given	Cumulative match	89.5%
Corney, et al. 2002	4	253 Emails	50-200 words	Stylistic, Struct, Func words	184	SVM	70.2%
de Vel 2000	5	18-87 Emails	3-680 words	Func words, Struct, Stylistic	38	SVM	85.7%
de Vel, et al. 2001	3	156 Docs	~12000 words	Stylistic	191	SVM	100%
Feiguina & Hirst 2007	11	4-10	2000 words	POS, Lexical, Syntactic	194	SVM	91.2% lex feat 88.7% all feat
Feng, et. al. 2012	10 5	8 sci papers 5 Novels	Variable 3000 sentences	Syntactic, Lexical,style	11 11	PCFG Trees & SVM	96% 95.2%
Gamon 2004	3	20 Sentences	Sentence	Func words, POS, Semantic	6018	SVM	85%
Goldman & Allison 2008	5	3 Novels	Variable length	POS, bigrams, Word freq	Not given	Chi Square	80%
Hirst & Feiguina 2007	2	250	1000 words	POS, Lexical, Vocab richness	194	SVM	99.2%
Hoover 2001	10	17 Novels	Variable length	Most frequent words	500	Cluster analysis	70%
Hoover 2003	8	16 Books	~24000 words	Vocabulary richness	10	Cluster analysis	37%
Iqbal, et al. 2008	6	20 Emails	Email	Lexical, Vocab richness	Not given	Data mining	86-90%
Iqbal, et al. 2010	158	200 Emails	Enron Corpus	Lex, Syntactic, Struct, Topic	292	SVM	82.9%
Kelselj, et al. 2003	8	Books	Variable length	Character 4-8-grams	Not given	Dissimilarity	100%
Koppel & Schler 2003	11	480 Emails	~200 words	Lexical, POS, Idiosyncrasies	358	C4.5 Trees, SVM	71.8%
Koppel & Schler 2004	10	21 Books	Variable length	Most frequent words	250	SVM	95.7%
Layton, et al. 2010	50	120 Tweets	=<140 char	Character n-grams	Not given	SCAP	70%
Li, et al. 2006	10	30-40	~169 words	Lexical, Structural, Syntactic	270	SVM	99.01%
Luyckx & Daelemans 2005	2	100 Articles	Variable length	n-grams, Syntactic, POS	91	ANN	71.3%
Luyckx & Daelemans 2008	145	Student Essays	~1400 words	Word, POS, n-grams, Lexical	91	k-NN, SVM	34%
Mustafa, et al. 2009	3	8 Books	Variable length	Frequent words, Word pairs	42	Correlation	0.99 correlation
Narayanan, et. al. 2012	100,000	24 blogs avg	Avg 305 words	Lex, Syntactic, Struct, Topic	1,188	kNN/RLSC	20%
Pavelec, et al. 2009	20	30 Articles	Variable length	Conjunctions (Portuguese) Adverbs (Portuguese)	77 94	Partial match SVM	84.3% 83.2%
Popescu & Dinu 2009	10	21 Books	Variable length	Function words	Not given	PCA Clustering	100%
Raghavan et al. 2010	6	14-28 Docs	7-24 k words	Syntactic	Not given	Naïve Bayes	95%
Stanczk & Cyan 2007	2	70 and 98	Short works	Function words, Punctuation	17	ANN	100%
Sun, et al. 2010 Sun, et al. 2010	20	30 Messages	~1383 char	Character n-grams	575 645	SVM GA	96.67% 93.67%
Tan & Tsai 2010	2	Novels	~60000 words	Syntactic	13	Bayesian	88.71%
Timboukakis & Tambouratzis 2009	5	1004	Variable length	Word freq, POS, Structural	85	MLP-NN SVM	89,71% 91.43%
Van Haltern 2004	8	9	628-1342words	Lexical and Syntactic	1050	Weighted voting	97%
Zheng, et al. 2003	9 3 3	153 70 70	News group Email Messages	Style Structural Content-specific	18	SVM	97% 91% 84%
Zheng, et al. 2006	20	30-92 Emails	84-346 words	Lexical, Syntactic, Structural	270	C4.5 Trees / SVM	93.36% / 97.69%

Table D.1.: Summary of Prior Authorship Attribution Stylometry Studies

D.1. Stylometry prior work references

1. S. Afroz, M. Brennan, and R. Greenstadt. Detecting hoaxes, frauds, and deception in writing style online. Proc. 2012 IEEE Sym. Security and Privacy. IEEE Computer Soc., Wash. DC, 461-475, 2012.
2. B. Allison and L. Guthrie. Authorship attribution of e-mail comparing classifiers cver a new corpus for evaluation, Proc. LREC'08, 2008.

3. M.Cristani, et al. Conversationally-inspired stylometric features for authorship attribution in instant messaging. Proc. 20th ACM Int. Conf. Multimedia. NY, 1121-1124, 2012.

4. M. Corney, O. de Vel, A. Anderson, and G. Mohay. Gender-preferential text mining of e-mail discourse. Proc. 18th Annual Computer Security App. Conf., Las Vegas, NV, Dec 2002.

5. O. de Vel. Mining e-mail authorship. Proc. KDD-2000 Workshop on Text Mining, Boston, Aug 2000.

6. O. de Vel, A. Anderson, M. Corney, and G. Mohay. Mining e-mail content for author identification forensics. ACM SIGMOD Record, 30(4):55, Dec 2001.

7. O.Feiguenia and G.Hirst. Authorship attribution for small texts: literary and forensic experiments. Proc. 30th Int.Conf. Special Int. Group. Info Retrieval (SIGIR), 2007.

8. S. Feng, R. Banerjee, and Y. Choi. Syntactic stylometry for deception detection. Proc. 50th Annual Meeting Assoc. Comp. Linguistics: Short Papers, Assoc. Comp. Ling., Stroudsburg, PA, 2: 171-175, 2012.

9. M. Gamon. Linguistic correlates of style: authorship classification with deep linguistic analysis features. Proc. 20th Int. Conf. Comp. Ling. (COLING '04). Assoc. Comp. Ling., Morristown, NJ, 611-617, 2004.

10. E. Goldman and A. Allison. Using grammatical markov modes for stylometric analysis. Stanford Univ. Tech. Report.

11. G. Hirst and O. Feiguina. Bigrams of syntactic labels for authorship discrimination of short texts. Literary and Linguistic Computing, 22(4): 405-417, 2007.

12. D. Hoover. Multivariate analysis and the study of style variation. Literary and Linguistic Computing, 18(4): 341-60, 2003.

13. D. Hoover. Statistical stylistics and authorship attribution: an empirical investigation. Literary and Linguistic Computing, 16: 421-44, 2001.

14. F. Iqbal, R. Hadjidj, B. Fung, and M. Debbabi. A novel approach of mining write-prints for authorship attribution in e-mail forensics. Digtal Investigation, 5: 42-51, 2008.

15. F. Iqbal, L. Khan, C. Benjamin, and M. Debbabi. E-mail authorship verification for forensic investigation. Proc. 2010 ACM Symposium Applied Computing (SAC '10). ACM, New York, NY, 1591-1598, 2010.

16. V. Keselj, F. Peng, N. Cerone, and C. Thomas. N-gram-based author profiles for authorship attribution. Proc. Conf. Pacific Assoc. Comp. Ling., PACLING'03, Nova Scotia, 255-264, 2003.

17. M. Koppel and J. Schler. Authorship verification as a one-class classification problem. ICML '04 Proc. .21st Int. Conf. Machine Learning, New York, 2004.

18. M. Koppel and J. Schler. Exploiting stylistic idiosyncrasies for authorship attribution. Proc. IJCAI'03 Workshop Comp. Approaches to Style Analysis and Synthesis, 69-72, 2003.

19. R. Layton, P. Watters, and R. Dazeley. Authorship attribution for twitter in 140 characters or less. Second Cybercrime and Trustworthy Comp. Workshop, 1-8, 2010

20. J. Li, R. Zheng, and H. Chen. From fingerprint to writeprint. Comm. ACM, 49(4): 76-82, 2006.

21. K. Luyckx and W. Daelemans. Authorship attribution and verification with many authors and limited data. Proc. 22nd Int. Conf. Comp. Ling., COLING '08, Assoc. Comp. Ling., NJ, 1: 513-520, 2008.

22. K. Luyckx and W. Daelemans. Shallow text analysis and machine learning for authorship attribution. Proc. 15th Meeting Comp. Ling. of the Netherlands, 2005.

23. T. Mustafa, N. Mustapha, M. Azmi, and N. Sulaiman. Computational stylometic approach based on frequent word and frequent pair in text mining authorship attrib. IJCSNS Int. J. Comp. Sci. Net. Sec., 9-3, 2009.

24. A. Narayanan, A. Paskov, H. Gong, N.Z. Bethencourt, J. Stefanov, E. Shin, E.C.R. Song, D. On the feasibility of internet-scale author identification. IEEE Symp. Security and. Privacy, 300-314, 2012.

25. D. Pavelec, L.S. Oliveira, E. Justino, F.D. Nobre Neto, and L.V. Batista. Compression and stylometry for author identification. Int. Joint Conf. Neural Networks, 2445-2450, 2009.

26. M. Popescu and L. Dinu. Comparing statistical similarity measures for stylistic multivariate analysis. Proc. RANLP 2009, Borovets, Bulgaria, 2009.

27. S. Raghavan, A. Kovashka, and R. Mooney. Authorship attribution using probabilistic context-free grammars. Proc. ACL 2010 Conf. Short Papers, CLShort '10, Assoc. Comp. Ling., PA, 38-42, 2010.

28. U. Stanczyk and K. Cyran. Machine learning approach to authorship attribution of literary texts. Int. J. Applied Mathematics and Informatics, 1(4), 2007.

29. J. Sun, Z. Yang, P. Wang, and S. Liu. Variable length character n-gram approach for online writeprint identification. Int.Conf. Multimedia Info. Netw. Sec. (MINES), Nanjing, Jiangsu, 486-490, 2010.

30. J. Sun, Z. Yang, P. Wang, L. Liu, and S. Liu. Feature selection for online writeprint identification using hybrid genetic algorithm. Int. Symp. Comp. Intel. and Design (ISCID), Hangzhou, 76-79, 2010.

31. F. Tan and R. Tsai. Authorship identification for online text. Proc. 2010 Int. Conf. Cyberworlds (CW '10). IEEE Computer Society, Washington, DC, 155-162, 2010.

32. N. Tsimboukakis and G. Tambouratzis. A comparative study on authorship attribution classification tasks using both neural network and statistical methods. Neural Comp. Appl., 19(4): 573-582, 2009.

33. H. Van Halteren. Linguistic profiling for author recognition and verification. Proc. 42nd Annual Meeting Assoc. Comp.Ling, Stroudsburg, PA, 199-206, 2004.

34. R. Zheng, J. Li, H. Chen, and Z. Huang. A framework for authorship identification of online messages: writing-style features and classification techniques. J. Am. Soc. Info. Science and Tech., Feb 2006.

35. R. Zheng, Y. Qin, Z. Huang, and H. Chen. Authorship Analysis in Cybercrime Investigation. Intel. Security Informatics, Hsinchun Chen et al., Eds., Springer Berlin Heidelberg, 2665: 59-73, 2003.

Bibliography

[1] Higher education opportunity act (heoa) of 2008. Accessed May 2012.

[2] Law of large numbers. (Accessed October 2013).

[3] Project gutenberg. (Accessed September 2012).

[4] Livia CF Araujo, Luiz HR Sucupira Jr, Miguel Gustavo Lizarraga, Lee Luan Ling, and João Baptista T Yabu-Uti. User authentication through typing biometrics features. *Signal Processing, IEEE Transactions on*, 53(2):851–855, 2005.

[5] Ned Bakelman, John V Monaco, Sung-Hyuk Cha, and Charles C Tappert. Keystroke biometric studies on password and numeric keypad input. 2013.

[6] Dieter Bartmann, Idir Bakdi, and Michael Achatz. On the design of an authentication system based on keystroke dynamics using a predefined input text. *International Journal of Information Security and Privacy (IJISP)*, 1(2):1–12, 2007.

[7] Henry Beker and Fred Piper. *Cipher systems: the protection of communications*. Northwood Books London, 1982.

[8] Steven S Bender and Howard J Postley. Key sequence rhythm recognition system and method, April 17 2007. US Patent 7,206,938.

[9] Francesco Bergadano, Daniele Gunetti, and Claudia Picardi. User authentication through keystroke dynamics. *ACM Transactions on Information and System Security (TISSEC)*, 5(4):367–397, 2002.

[10] Shane Bergsma, Matt Post, and David Yarowsky. Stylometric analysis of scientific articles. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 327–337. Association for Computational Linguistics, 2012.

[11] Saleh Bleha, Charles Slivinsky, and Bassam Hussien. Computer-access security systems using keystroke dynamics. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 12(12):1217–1222, 1990.

[12] Ruud M. Bolle, Jonathan Connell, Sharath Pankanti, Nalini K. Ratha, and Andrew W. Senior. *Guide to Biometrics (Springer Professional Computing)*. Springer, 2010.

[13] Sung-Hyuk Cha and Sargur N Srihari. Writer identification: statistical analysis and dichotomizer. In *Advances in Pattern Recognition*, pages 123–132. Springer, 2000.

[14] Sungzoon Cho, Chigeun Han, Dae Hee Han, and Hyung-Il Kim. Web-based keystroke dynamics identity verification using neural network. *Journal of organizational computing and electronic commerce*, 10(4):295–307, 2000.

[15] Ali Darwish and Michel Pasquier. Biometric identification using the dynamic features of the eyes. In *Biometrics: Theory, Applications and Systems*, 2013.

[16] George Doddington, Walter Liggett, Alvin Martin, Mark Przybocki, and Douglas Reynolds. Sheep, goats, lambs and wolves: A statistical analysis of speaker performance in the nist 1998 speaker recognition evaluation. Technical report, DTIC Document, 1998.

[17] Joao Ferreira and Henrique Santos. Keystroke dynamics for continuous access control enforcement. In *Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), 2012 International Conference on*, pages 216–223. IEEE, 2012.

[18] Eric Flior and Kazimierz Kowalski. Continuous biometric user authentication in online examinations. In *Information Technology: New Generations (ITNG), 2010 Seventh International Conference on*, pages 488–492. IEEE, 2010.

[19] Mario Frank, Ralf Biedert, Eugene Ma, Ivan Martinovic, and Dawn Song. Touchalytics: On the applicability of touchscreen input as a behavioral biometric for continuous authentication. 2012.

[20] Romain Giot, Mohamad El-Abed, and Christophe Rosenberger. Greyc keystroke: a benchmark for keystroke dynamics biometric systems. In *Biometrics: Theory, Applications, and Systems, 2009. BTAS'09. IEEE 3rd International Conference on*, pages 1–6. IEEE, 2009.

[21] Romain Giot, Mohamad El-Abed, and Christophe Rosenberger. Keystroke dynamics with low constraints svm based passphrase enrollment. In *Biometrics: Theory, Applications, and Systems, 2009. BTAS'09. IEEE 3rd International Conference on*, pages 1–6. IEEE, 2009.

[22] Daniele Gunetti and Claudia Picardi. Keystroke analysis of free text. *ACM Transactions on Information and System Security (TISSEC)*, 8(3):312–347, 2005.

[23] Frederick Jelinek. Interpolated estimation of markov source parameters from sparse data. *Pattern recognition in practice*, 1980.

[24] L Jin, X Ke, R Manuel, and M Wilkerson. Keystroke dynamics: A software based biometric solution. In *Proc. 13th USENIX Security Symposium*, 2004.

[25] Dan Jurafsky, James H Martin, Andrew Kehler, Keith Vander Linden, and Nigel Ward. *Speech and language processing: An introduction to natural lan-*

guage processing, computational linguistics, and speech recognition, volume 2. MIT Press, 2000.

[26] Slava Katz. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 35(3):400–401, 1987.

[27] Ricardo Kawase, George Papadakis, Eelco Herder, and Wolfgang Nejdl. Beyond the usual suspects: context-aware revisitation support. In *HT*, pages 27–36, 2011.

[28] Preeti Khanna and M Sasikumar. Recognising emotions from keyboard stroke pattern. *International Journal of Computer Applications*, 11(9), 2010.

[29] Kevin Killourhy and Roy Maxion. The effect of clock resolution on keystroke dynamics. In *Recent Advances in Intrusion Detection*, pages 331–350. Springer, 2008.

[30] Kevin S Killourhy. A scientific understanding of keystroke dynamics. Technical report, DTIC Document, 2012.

[31] Kevin S Killourhy and Roy A Maxion. Comparing anomaly-detection algorithms for keystroke dynamics. In *Dependable Systems & Networks, 2009. DSN'09. IEEE/IFIP International Conference on*, pages 125–134. IEEE, 2009.

[32] BVK Vijaya Kumar. *Correlation pattern recognition*. Cambridge University Press, 2005.

[33] John Leggett, Glen Williams, Mark Usnick, and Mike Longnecker. Dynamic identity verification via keystroke characteristics. *International Journal of Man-Machine Studies*, 35(6):859–870, 1991.

[34] Chen Change Loy, Weng Kin Lai, and Chee Peng Lim. Keystroke patterns classification using the artmap-fd neural network. In *Intelligent Information Hiding and Multimedia Signal Processing, 2007. IIHMSP 2007. Third International Conference on*, volume 1, pages 61–64. IEEE, 2007.

[35] Chen Change LOY, Chee Peng LIM, and Weng Kin LAI. Pressure-based typing biometrics user authentication using the fuzzy artmap neural network. In *Proceeding of the 12th International Conference on Neural Information Processing*, 2005.

[36] Anthony J Mansfield and James L Wayman. *Best practices in testing and reporting performance of biometric devices*. Centre for Mathematics and Scientific Computing, National Physical Laboratory Teddington, Middlesex, UK, 2002.

[37] Roy A Maxion and Kevin S Killourhy. Keystroke biometrics with number-pad input. In *Dependable Systems and Networks (DSN), 2010 IEEE/IFIP International Conference on*, pages 201–210. IEEE, 2010.

[38] Arik Messerman, Tarik Mustafic, Seyit Ahmet Camtepe, and Sahin Albayrak. Continuous and non-intrusive identity verification in real-time environments

based on free-text keystroke dynamics. In *Biometrics (IJCB), 2011 International Joint Conference on*, pages 1–8. IEEE, 2011.

[39] John V Monaco, Ned Bakelman, Sung-Hyuk Cha, and Charles C Tappert. Recent advances in the development of a long-text-input keystroke biometric authentication system for arbitrary text input. In *Intelligence and Security Informatics Conference (EISIC), 2013 European*, pages 60–66. IEEE, 2013.

[40] Fabian Monrose, Michael K Reiter, and Susanne Wetzel. Password hardening based on keystroke dynamics. *International Journal of Information Security*, 1(2):69–83, 2002.

[41] A. Newell. *Unified Theories of Cognition*. William James Lectures. Harvard University Press, 1994.

[42] Sean Peisert, Ed Talbot, and Tom Kroeger. Principles of authentication. In *Proceedings of the 2013 workshop on New security paradigms workshop*, pages 47–56. ACM, 2013.

[43] Anna Pereira, David L Lee, Harini Sadeeshkumar, Charles Laroche, Dan Odell, and David Rempel. The effect of keyboard key spacing on typing speed, error, usability, and biomechanics part 1. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 55(3):557–566, 2013.

[44] Kenneth Revett. *Behavioral biometrics: a remote access approach*. Wiley. com, 2008.

[45] Ricardo N Rodrigues, Glauco FG Yared, Carlos R do N Costa, João BT Yabu-Uti, Fábio Violaro, and Lee Luan Ling. Biometric access control through numerical keyboards based on keystroke dynamics. In *Advances in Biometrics*, pages 640–646. Springer, 2005.

[46] Joseph Roth, Xiaoming Liu, Arun Ross, and Dimitris Metaxas. Biometric authentication via keystroke sound.

[47] Abdul Serwadda and Vir V Phoha. Examining a large keystroke biometrics dataset for statistical-attack openings. *ACM Transactions on Information and System Security (TISSEC)*, 16(2):8, 2013.

[48] Chao Shen, Zhongmin Cai, Xiaohong Guan, Youtian Du, and R Maxion. User authentication through mouse dynamics. 2013.

[49] Tomer Shimshon, Robert Moskovitch, Lior Rokach, and Yuval Elovici. Continuous verification using keystroke dynamics. In *Computational Intelligence and Security (CIS), 2010 International Conference on*, pages 411–415. IEEE, 2010.

[50] Efstathios Stamatatos. A survey of modern authorship attribution methods. *Journal of the American Society for information Science and Technology*, 60(3):538–556, 2009.

[51] John C Stewart, John V Monaco, Sung-Hyuk Cha, and Charles C Tappert. An investigation of keystroke and stylometry traits for authenticating online test

takers. In *Biometrics (IJCB), 2011 International Joint Conference on*, pages 1–7. IEEE, 2011.

[52] C. C. Tappert, S. Cha, M. Villani, and R. S. Zack. Keystroke biometric identification and authentication on long-text input. *Int. Journal Information Security and Privacy (IJISP)*, 2010.

[53] Charles C Tappert, Sung-Hyuk Cha, Mary Villani, and Robert S Zack. A keystroke biometric system for long-text input. *International Journal of Information Security and Privacy (IJISP)*, 4(1):32–60, 2010.

[54] Mary Villani, Charles Tappert, Giang Ngo, Justin Simone, H St Fort, and Sung-Hyuk Cha. Keystroke biometric recognition studies on long-text input under ideal and application-oriented conditions. In *Computer Vision and Pattern Recognition Workshop, 2006. CVPRW'06. Conference on*, pages 39–39. IEEE, 2006.

[55] Zhengzheng Xing, Jian Pei, and Eamonn Keogh. A brief survey on sequence classification. *ACM SIGKDD Explorations Newsletter*, 12(1):40–48, 2010.

[56] Neil Yager and Ted Dunstone. The biometric menagerie. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(2):220–230, 2010.