2019-12

USER IDENTIFICATION THROUGH KEYSTROKE BIOMETRICS AT AN INTERNET SCALE

Veazey, Mark W.

Monterey, CA; Naval Postgraduate School

NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

USER IDENTIFICATION THROUGH KEYSTROKE BIOMETRICS AT AN INTERNET SCALE

by

Mark W. Veazey

December 2019

Thesis Advisor:	Vinnie Monaco
Second Reader:	John D. Fulp

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE	*Form Approved OMB No. 0704-0188*

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington, DC 20503.

1. AGENCY USE ONLY ***(Leave blank)***	**2. REPORT DATE** December 2019	**3. REPORT TYPE AND DATES COVERED** Master's thesis

4. TITLE AND SUBTITLE USER IDENTIFICATION THROUGH KEYSTROKE BIOMETRICS AT AN INTERNET SCALE	**5. FUNDING NUMBERS**
6. AUTHOR(S) Mark W. Veazey	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000	**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A	**10. SPONSORING / MONITORING AGENCY REPORT NUMBER**

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release. Distribution is unlimited.	**12b. DISTRIBUTION CODE** A

13. ABSTRACT (maximum 200 words)

Identification of users on the internet has broad-reaching implications in the computer science discipline regarding cyber security and privacy. Keystroke biometrics leverages the unique dynamics of how a user types to perform identification; however, current methods of authentication and identification using keystroke dynamics do not scale well beyond a few hundred users. This thesis investigates the feasibility of using conventional machine learning and deep learning techniques to identify users at an internet scale. By analyzing free-text keystroke information from a collection of over 100,000 users, several methods to perform user identification and profiling are identified, with a focus on determining how the size of the dataset affects identification accuracy. This thesis includes a novel method of representing keystroke data in a two-dimensional format suitable for a convolutional neural network, and it examines to what extent keystroke biometrics has implications for privacy on the internet.

14. SUBJECT TERMS artificial intelligence, machine learning, neural networks, keystroke biometrics, keystroke dynamics, authentication, identification, cyber security, fingerprinting	**15. NUMBER OF PAGES** 85
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UU

THIS PAGE INTENTIONALLY LEFT BLANK

USER IDENTIFICATION THROUGH KEYSTROKE BIOMETRICS AT AN INTERNET SCALE

Mark W. Veazey
Lieutenant, United States Navy
BS, U.S. Naval Academy, 2011

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
December 2019

Approved by: Vinnie Monaco
Advisor

John D. Fulp
Second Reader

Peter J. Denning
Chair, Department of Computer Science

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

Identification of users on the internet has broad-reaching implications in the computer science discipline regarding cyber security and privacy. Keystroke biometrics leverages the unique dynamics of how a user types to perform identification; however, current methods of authentication and identification using keystroke dynamics do not scale well beyond a few hundred users. This thesis investigates the feasibility of using conventional machine learning and deep learning techniques to identify users at an internet scale. By analyzing free-text keystroke information from a collection of over 100,000 users, several methods to perform user identification and profiling are identified, with a focus on determining how the size of the dataset affects identification accuracy. This thesis includes a novel method of representing keystroke data in a two-dimensional format suitable for a convolutional neural network, and it examines to what extent keystroke biometrics has implications for privacy on the internet.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

I. INTRODUCTION....1

- A. PROBLEM STATEMENT....1
- B. RESEARCH QUESTIONS....2
- C. SCALE....2
- D. BENEFITS OF RESEARCH....3
- E. ORGANIZATION....4

II. BACKGROUND....5

- A. BIOMETRICS....5
  - 1. Biometric Types....5
  - 2. Limitations....6
  - 3. Biometric Menagerie....7
  - 4. Performance Measures....7
- B. KEYSTROKE DYNAMICS....9
  - 1. Static versus Dynamic Typing....9
  - 2. Mechanics of a Keystroke....9
  - 3. Computing Features....10
  - 4. Typing Performance Factors....11
- C. INTERNET PRIVACY....13
  - 1. General Authorship....14
  - 2. Fingerprinting....14
  - 3. Methods of Anonymity....15
- D. KEYSTROKE DYNAMICS AND INTERNET PRIVACY....16
  - 1. Keystroke Dynamics versus Browser Fingerprinting....16
  - 2. Role of Keystroke Dynamics....18

III. THE KEYSTROKE DATASET....19

- A. DATA COLLECTION....19
  - 1. How Participants Were Chosen....19
  - 2. Demographics....19
  - 3. How Data Was Collected....20
  - 4. How Data Was Presented....21
  - 5. Keystroke Data Visualization....22
- B. LIMITATIONS OF THIS DATASET....25

IV. BASELINE APPROACH....27

- A. METHODOLOGY....28

1. Feature Extraction 28
2. Preprocessing 31
3. Classification 33
4. Accuracy Metrics 33
5. Profiling 34

B. RESULTS 34

1. Accuracy 34
2. Time 36
3. CDFs 37
4. Profiling Users 40

V. NEURAL NETWORK APPROACH 43

A. METHODOLOGY 44

1. Keyboard Grid 44
2. Preprocessing 45
3. Adding Features 46
4. Shaping the Data 46
5. Feature Normalization 49

B. NEURAL NETWORK 50

1. Types of Neural Networks 50
2. Network Architecture 51
3. Hyperparameter Tuning 52

C. RESULTS 55

1. Accuracy 55
2. Profiling 58

VI. CONCLUSION 61

A. DISCUSSION 61

B. INTERNET PRIVACY IMPLICATIONS 62

C. FUTURE WORK 63

LIST OF REFERENCES 65

INITIAL DISTRIBUTION LIST 69

LIST OF FIGURES

Figure 1. Varying Scales of Biometrics 3

Figure 2. Dynamics of a Key Press. Adapted from [9]. 10

Figure 3. Dataset Demographics. Source: [10]. 20

Figure 4. Example Sentences from the Typing Test. Source: [10]. 21

Figure 5. Histogram of Words per Minute of Users in the Keystroke Dataset 23

Figure 6. WPM versus ROR....................24

Figure 7. Error Rate versus Typing Speed 24

Figure 8. Number of Keystrokes per Session....................29

Figure 9. Format of Data Before and After Feature Creation 30

Figure 10. Top 1, 10, 100 and 1000 Identification Accuracies for Varying Numbers of Users 35

Figure 11. Time in Hours to Calculate Accuracy as Number of Users Increases....................37

Figure 12. Cumulative Distribution Function 100 Users 38

Figure 13. Cumulative Distribution Function 1000 Users 38

Figure 14. Cumulative Distribution Function 10,000 Users 39

Figure 15. Cumulative Distribution Function 50,000 Users 39

Figure 16. Cumulative Distribution Function 100,000 Users 40

Figure 17. Classification Accuracy of Users Based on Profile Information....................41

Figure 18. Keyboard Layout 45

Figure 19. Preprocessing Steps 47

Figure 20. Duration and Latencies per Session....................47

Figure 21. Differences in Typing Between Users 48

Figure 22. Two Methods of Filling Zero Values....................50

Figure 23. CNN Model for 100 Users 52

Figure 24. Accuracy and Loss Graphs Depicting Overfitting 53

Figure 25. Accuracy and Loss Graphs Depicting a Situation to Keep Training for More Epochs 54

Figure 26. Identification Accuracy for KNN and NN Methods 56

Figure 27. Top N Accuracies at 2000 Users 57

Figure 28. Top N Accuracies at 50000 Users 57

Figure 29. Profiling Accuracies Using the Neural Network 59

LIST OF TABLES

Table 1. Keystroke Dynamics versus Browser Fingerprinting 17

Table 2. Keystroke Data Example 22

Table 3. Metadata Example 22

Table 4. List of Features. Adapted from [25]. 31

Table 5. Achieved Accuracy versus Random Chance Accuracy Using Entire Dataset...36

Table 6. Number of Filters and Epochs Used to Train the Neural Network............55

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF ACRONYMS AND ABBREVIATIONS

ANSI	American National Standards Institute
CDF	Cumulative Distribution Function
CNN	Convolutional Neural Network
FMR	False Matching Rate
FNMR	False Non-Matching Rate
HPC	High Performance Computing
HTTP	Hypertext Transfer Protocol
IKI	Inter-Key Interval
KNN	K-Nearest Neighbor
PP	Press to Press Latency
PR	Press to Release Latency (Duration)
QWERTY	Type of keyboard layout
RNN	Recurrent Neural Network
ROC	Required Operating Curve
ROR	Roll Over Rate
RP	Release to Press Latency
RR	Release to Release Latency
TCP	Transfer Control Protocol
WPM	Words Per Minute

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

Numerous forms of biometrics are used to identify people based on how they look, talk, walk and act. Verification and identification of individuals has been imperative throughout the history of humans. In the computer age, identifying a user using biometrics has gained significant importance, complexity, accuracy and degree of difficulty as technology improves. Current forms of biometrics include fingerprints, retina scans, facial recognition, voice recognition and keystroke dynamics.

The focus of this thesis will be keystroke dynamics. This form of biometrics is important because it can be passive, meaning it can be obtained without the knowledge of the user. It can also be collected continuously and unobtrusively as a user goes about his everyday interactions with a computer system.

Significant developments have been made through combining biometrics with machine learning techniques so as to enhance the ability of a computer to perform identification of users. This thesis investigated whether machine-learning techniques can be successful in determining the identification of users on a large scale based on the uniqueness of their keystroke dynamics.

A. PROBLEM STATEMENT

Substantial research has been conducted regarding keystroke dynamics and identification methods using this form of biometrics. There are also commercial applications that use keystroke dynamics to enhance security of systems. However, there is an absence of research concerned with keystroke biometric applications on an Internet scale. It is unknown whether an Internet user could be identified or even significantly narrowed-down based on the way he interacts with a computer keyboard. It would be extremely useful to be able to compare keystroke patterns and present results with likely matches out of a population of users. In addition, keystroke patterns can be leveraged to profile a user into categories such as age, gender, nationality, etc., based on his typing patterns.

B. RESEARCH QUESTIONS

There are two questions that this research sought to address. First, we aimed to determine to what extent can keystroke biometrics be performed at an Internet scale, using a large dataset of keystroke logs collected from over 160,000 users. This thesis investigated whether machine-learning techniques and neural networks are able to perform identification and profiling on such a large collection of users.

Our second question aims to determine the privacy implications of using keystroke dynamics to perform identification at an Internet scale. For example, what are the privacy consequences when keystroke dynamics can accurately de-anonymize users on the Web? Such implications could extend to, and adversely affect, users employing the Tor network—or other such mechanisms—to help achieve anonymity for protection against political targeting.

C. SCALE

Keystroke biometrics can be performed at various scales, determined by the application setting. Figure 1 demonstrates three scale categories—two extreme and one intermediate—that we considered. At one end of the spectrum, a one-to-one comparison is made between a user-provided sample and that user's pre-registered/collected template. This scenario encompasses authentication, which is useful for adding security to digitally enabled transactions. For example, comparing the way a user enters a password to ensure the user is who he says he is. The system can also continuously check that the person behind the keyboard remains the same user that originally authenticated at the time of log-on. In this way, the system compares the sequence of keystrokes to the "registered" example of the user's typing characteristics.

For the "intermediate" category, typing behavior could be collected on the users of an organization's intranet, so as to perform identification on a "medium" sized set of persons. Though the size of intranets can vary widely, in the context of our research, it refers to the nominal size of an average corporate network: a few thousand or less. Identification on this scale consists of comparing keystroke dynamics to a fixed number of already known users on the intranet of interest.

The most difficult scale category we chose is identification on an Internet-scale. At this scale, we are comparing the keystroke dynamics of users to a much larger and ever-increasing collection of potential matches. In this research, this is the scale category of most interest, because this thesis introduces new techniques to successfully identify users in a dataset of over 160,000 users.

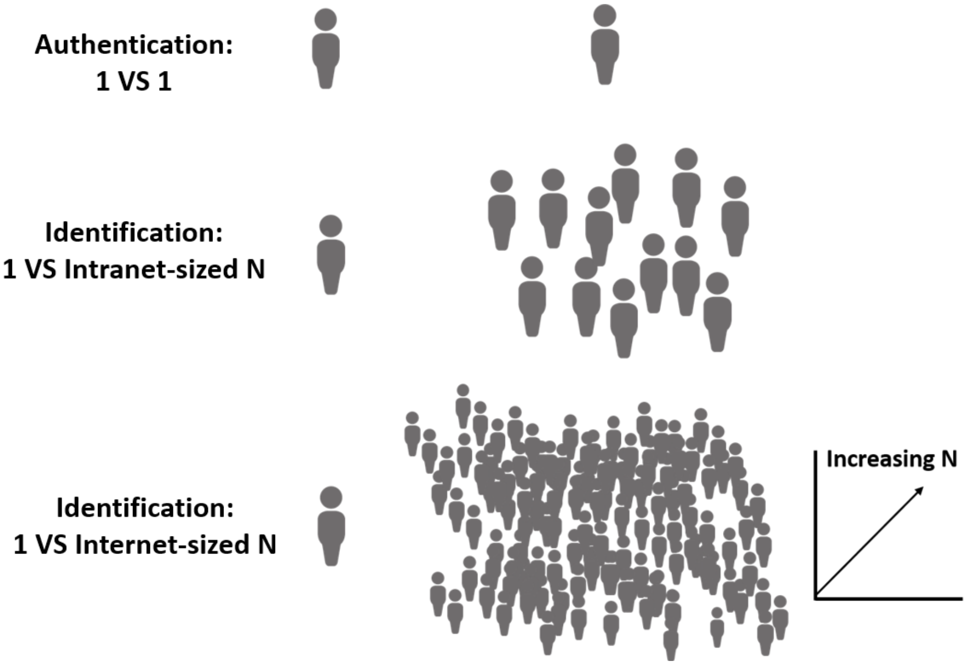


Figure 1. Varying Scales of Biometrics

D. BENEFITS OF RESEARCH

One contribution of this study is in the field of cyber security. If keystroke biometrics becomes a viable option of authentication, it would add a much-needed security feature in digital computing environments. Keystroke biometrics is a relatively low-cost measure that uses commodity hardware to monitor keystrokes to identify a user or verify a user's claimed identity. Continuous authentication could even be used to monitor a user after entering a password to guarantee that the user's session has not been hijacked.

Another benefit this study may bring to the cyber security community is improved identification abilities. Identifying a user based on his keystrokes could be extremely valuable to intelligence agencies, and both the defensive and offensive cyber communities. Patterns could be defined based on keystroke information in order to allow malicious actors to be more easily identified. Offensive cyber applications involve using these techniques to identify targets as well as learn ways for offensive actors to mask their keystroke data in order to avoid detection.

The results of this thesis are of interest to both the Department of Defense and the U.S. government. DARPA's Active Authentication program is evidence of this. DARPA is seeking "to develop novel ways of validating the identity of the person at the console that focus on the unique aspects of the individual through the use of software based biometrics" [1]. This thesis seeks to help with that development. Cyber security is an area where research is needed and new capabilities are constantly in demand. The prospect of identifying users in extremely large-scale datasets has broad-reaching implications to the United States' cyber workforce in both the offensive and defensive realms.

E. ORGANIZATION

Chapter II of this thesis explores the background of biometrics, keystroke dynamics, and Internet privacy to ensure the reader has an appropriate knowledge base regarding the concepts contained in this thesis. Chapter III investigates the keystroke dataset that was used in this research. The scale, statistics and other factors of the data will be laid out. Chapter IV provides a baseline approach to identification of users in this dataset and the results obtained from this approach. Chapter V introduces a method of identification that relies on a neural network and presents the results of this method. Finally, Chapter VI provides a conclusion and future work that may build off of this thesis.

II. BACKGROUND

This section will focus on providing a background of knowledge regarding keystroke dynamics and the associated data that can be obtained from this timing information. Background information regarding Internet privacy, including browser and device fingerprinting, is explored, as well as general information about user identification and computer biometrics.

A. BIOMETRICS

Biometrics are metrics related to human characteristics, and a biometric system is any system that uses these characteristics to perform authentication or identification. Biometrics present an added security measure to any system by increasing the factors used for identification. Adding a biometric to a password creates a two-factor authentication system, thus increasing the amount of security beyond a single-factor system. Biometrics have also been used throughout history to aid in identification. One of the earliest forms of biometrics performed by humans is simple facial recognition. DNA and fingerprints, some of the first widely used biometrics, have been successfully used for decades in law enforcement. More recently, new forms of biometrics based on human-computer interaction have entered the computer security realm. Biometrics and the various processing systems introduce technical complexities that must be understood in order to fully appreciate the research findings presented in this thesis. This section presents the types of biometric systems, the limitations of biometrics, the concept of the biometric menagerie, and the types of error rates and measures of success of biometric systems.

1. Biometric Types

A biometric system can be used in two basic ways. The type of biometric used depends on a multitude of factors, including the reason for use, number of participants, and whether or not enrollment has been performed, among others. The motive for using biometrics is broken down into either verification or identification. This thesis is concerned with the identification mode of biometrics due to the large-scale dataset that is being used, as well as the reason for conducting keystroke biometrics in order to identify users on the Internet.

Verification mode refers to the use case wherein a system is used to *confirm* the identity of a user based on the user’s template that was collected and saved to the system database at the time of that user's enrollment [2]. This mode begins when a user claims to be a valid user on a system, which then performs a biometric check in order to compare the user with his registered identity. The reason for this is to ensure only one person can use each identity and eliminate imposter users. Verification mode represents a one-to-one authentication of a user and the corresponding template.

Identification is the alternate mode of biometrics. A one-to-many search is performed in order to determine if there is a match on the system [2]. This mode is used if the system has information on a particular user and wants to search the database to see if the user corresponds to a previously defined identity. Identification is key to this thesis as this research evaluated the viability of identification on the Internet scale.

2. Limitations

There are a few glaring limitations to biometric systems. These limitations are exaggerated when dealing with a large dataset. The five limitations of unimodal biometric systems are presented by Jain et al. in the paper: “An introduction to biometric recognition” [2]. Noisy data makes biometric identification difficult and may lead to false positives or false negatives in the results. Noise in keystroke dynamics equates to a user who is not inputting keystroke data in his normal/usual way, due to a factor such as typing position or mood. Intra-class variations refer to differences in data from template creation to when the identification is occurring. Variations in keystroke data could occur in many ways, including changing keyboard type. Another limitation is the distinctiveness of the biometric. Distinctiveness must be taken into consideration because in order for verification or identification to occur, the data between users must be sufficiently distinct. This is especially a problem with keystroke biometrics because it is possible for multiple users’ keystrokes to not be unique enough to make a correct match. Non-universality is a consideration because it is possible that not all users have a certain biometric trait. This is not so much a limitation in the keystroke biometrics realm because it is extremely rare to use a computer without interacting with a keyboard, but it is possible to not provide enough biometric data to make a match. Lastly, spoof attacks could be used to impersonate another user in the system, or to deliberately change the data enough so it is not recognizable. It is entirely possible to disrupt a keystroke biometric system by typing differently than usual. Each of the described limitations presents unique challenges to a biometric system and needs to be taken into account.

3. Biometric Menagerie

The biometric menagerie refers to the groups into which users are categorized based on ease of detection when dealing with a biometric system. Doddington et al. [3] first described the animal names that are assigned to different types of users in a system. Sheep are the most abundant user type in a system. They are defined as the model user for which the system performs well. Goats are users who are not easily recognizable. They may have noisy data or may not be very distinct. Goats can disproportionately lead to false negatives in a biometric system. Lambs are those users who are easily imitated. This type of user negatively affects the performance of a biometric system because lambs result in a majority of the false positives that occur. Wolves are the users who are able to imitate others and be falsely identified.

All users in a biometric system may display a continuum of menagerie traits. Just because a user is falsely identified as an imposter or missed when he should have been identified, does not ensure he is labeled as a goat, lamb or wolf. Biometric system designers are interested in finding users who continuously exhibit these characteristics. Yager et al. states, “The reasons that a particular animal group exist are complex and varied. They depend on a number of factors, including enrollment procedures, feature extraction and matching algorithms, data quality, and intrinsic properties of the user population” [4].

4. Performance Measures

A biometric system never works 100% of the time. The goal is to get the results as close as possible to perfect matching (i.e., no false positives or false negatives), but there will always be some errors and mistakes. There must be a way to characterize these systems and determine how well a given biometric system is performing. Authentication and Identification are fundamentally different, and thus require different metrics to measure how well they are working. The next two sections will describe the methods of quantifying errors and accuracy in a biometric system.

a. Authentication Errors

There are two types of errors used to describe biometric systems. These errors are called false match and false non-match. False match occurs when a system incorrectly accepts a biometric sample as a match. False match is also called a false positive. False non-match, also called false negative, occurs when valid biometric data is incorrectly rejected by the system. The False Match Rate (FMR) can be calculated by taking the number of false matches divided by the total number of false match attempts. The False Non-Match Rate (FNMR) is calculated by the number of false non-matches over the total number of genuine match attempts [5]. Both of these error rates can be displayed together in a Receiver Operating Characteristic (ROC) curve. A ROC curve plots the FMR as it relates to the FNMR. Every biometric system displays an inverse relationship between the FMR and FNMR. As the FMR increases, the FNMR decreases, and vice versa. It is imperative to evaluate the ROC curve when deciding how a system needs to operate. High security applications require the lowest possible FMR, but may sacrifice usability if the FNMR becomes too high. Likewise, in other applications, a low FNMR may be required but may come with a high FMR that leads to more false positives [2].

b. Accuracy

The accuracy of a biometric system is normally reported as a percentage of queries that were classified correctly. However, when operating at a large scale, different metrics of merit may need to be considered. This is of particular concern when a biometric system is operating in one-to-many identification mode, as the simple accuracy percentage may not tell the whole story. Li, Guo and Hopper [6] argue that validating by accuracy alone is flawed. In their paper regarding website fingerprinting on the Tor browser, they describe how low classification accuracy may not equate to low information leakage. Accuracy is a measure of all or nothing classification, but large amounts of information could still be gleaned while reporting a low accuracy score. Another metric that can be used is a gap statistic that outputs a similarity score between the user being tested and each class [7]. Other statistics; such as percentage of successful classifications within the top N choices, may present a better representation of the usefulness of biometric experiments. This metric is referred to as top-N accuracy and was used extensively in this research.

B. KEYSTROKE DYNAMICS

Many factors influence how a user types on a keyboard. These individual factors provide a way of identifying a person based on how he types. Keystroke dynamics depend on the type of keyboard he is using, how his hands are positioned, the exact timing of each key press and the content (thus key patterns) that he is inputting. These aspects have been thoroughly covered in past research, and are used as vital underlying information throughout my thesis.

1. Static versus Dynamic Typing

An extremely important factor when collecting data on keystroke dynamics is whether a user is typing a fixed entry or typing freely while interacting with the computer system. Monitoring the keystroke dynamics of a fixed entry is considered static typing. This category of typing most often occurs when a user is logging on and the typing dynamics of a password or phrase are measured. The system can then compare the typing behavior of this common phrase or unique user ID and password in order to authenticate a user. Dynamic typing is captured as a user continuously interacts with a computer [5]. For example, keystroke information may be collected as a user types an email or searches in a browser. Dynamic text entry can be first checked during logon and continuously monitored afterward. The research in this thesis is based on dynamic typing.

2. Mechanics of a Keystroke

Understanding the dynamics of a key press is essential to this thesis. A computer records the time a certain key was pressed as well as when it was released. From these times, metrics can be extracted from this raw keystroke data. The time during which a key is pressed until it is released is called the duration. Other features can be determined by taking the press and release times of different keys. These features are called latencies. The time between a key being pressed until the next key is pressed is called the press-to-press latency (PP). PP is also called a digraph or, if extended to the next press, a trigraph. Release-to-press (RP) and release-to-release (RR) can also be calculated [5]. As shown in Figure 2, RP latencies can be negative. This characteristic of typing is called rollover and occurs when a user begins pressing the next key before releasing the previous key [8]. These features are the basic timing events for individual keystrokes and are used to compute more complex features.

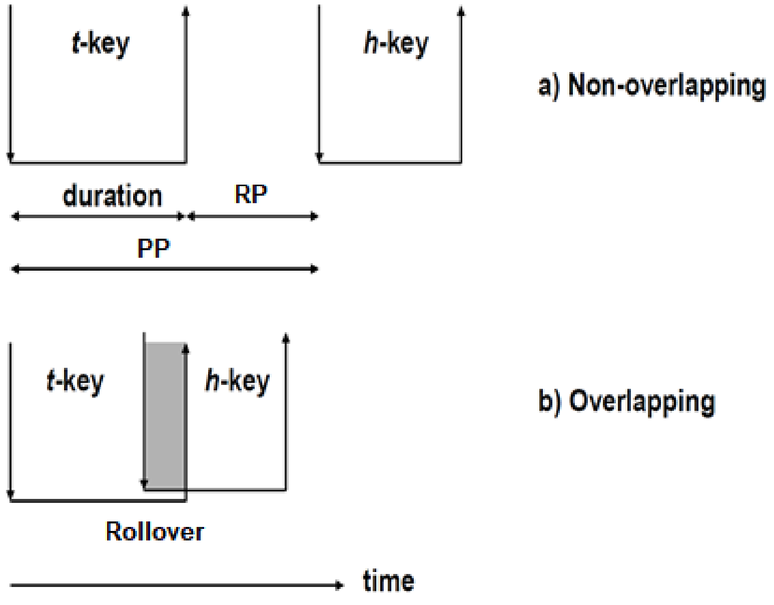


Figure 2. Dynamics of a Key Press. Adapted from [9].

3. Computing Features

Many more features can be computed using the keystroke timing data discussed in the previous section. Durations of individual keys and groups of keys can be assessed as well as the mean and standard deviation of these durations. Features can be added to represent PP, RP and RR latencies between letters and non-letters in addition to the mean and standard deviation of these latencies. Latencies and durations dependent on keys struck with the left or right hand can be included [8]. These features can then be repeated on specific digraphs and trigraphs. Calculating each of these times can lead to an increased feature space and improve accuracy of user identification. The research conducted by Tappert et al. [8] on long text input used 239 feature measurements making use of the letter and digraph frequencies in the English language.

Overall performance measures can be added as well. When conducting the research on the large dataset that is used in this thesis, Dhakal et al. [10] computed more features that helped to sum up the key press data. In addition to the durations and latencies previously described, words per minute (WPM) was calculated by counting the number of words and dividing by minutes. Features were also computed based on errors made and errors corrected. This was displayed as the percentage of uncorrected errors and corrected errors. An additional feature was the rollover ratio, which is a percentage of times the RP latency is negative.

Even more features can be extracted from dynamic keystroke data than has been previously described. Exponentially more features can be developed when taking into account new digraphs and trigraphs and the relationships within, between, and among them. Further, considering the context with which a certain letter is used may lead to the introduction of even more features [11]. As an example, the letter “a” may have a different duration when it is typed in the word “ant,” then when typed in the word “marker.”

4. Typing Performance Factors

There are many factors that contribute to the way that a person types. The amount of formal training, depth of experience, age, handedness and gender all have an effect on typing speed and dynamics. This section will describe a few factors that played a role in this research and elaborate on how they affect the typing behavior of a user.

a. Keyboard Types

Not all keyboards are created equal. Many English-speaking users type on an externally connected QWERTY keyboard with large, deep keys. Laptop keyboards are usually much shallower and sometimes smaller compared to desktop keyboards. Most prior research makes a distinction between these two types of keyboards because a user’s typing on a desktop keyboard could be very different from how that same user types on a laptop keyboard. Villani et al. [9] concludes that keystroke biometrics can only be effectively used if the user maintains the same type of keyboard.

Other keyboard factors may contribute to variations that could affect performance. Some keyboards may not have the same keys or may combine keys on the keyboards to save space. Some laptops make use of a function key to access lesser-used keys on the keyboard. Certain keyboard brands may be more or less sensitive to the touch than others. Lastly, laptops are mobile and allow a user to type in different positions like on the lap, on a desk, or even laying down on a bed [11]. Typing in different positions will undeniably make the task of keystroke biometric recognition even harder.

Mobile devices and tablets are becoming increasingly popular and make up a larger percentage of devices each year. Some of these devices use a soft, flexible keyboard, while others use multitouch functionality. Measuring keystroke dynamics on a mobile device will be completely different from a conventional keyboard. According to Varcholik et al. [12], typing on multitouch devices cuts performance in half, from an average of 60 words per minute to an average of 30 words per minute. The research of this thesis deals solely on keystroke data taken from non-multitouch devices.

b. Types of Typists

Of course, not all typists type in the same manner. The most common method of typing is where the typist rests his hands on the keyboard in a position wherein the left hand rests on the keys A, S, D, F, and the right hand rests on the keys J, K, L, ;.. This provides a referential starting location for the fingers to reach all the keys on the keyboard.

Another way of typing that is used by less skilled typists is the hunt and peck method. This method uses either one or two fingers to press the keys and is usually a much slower means of typing. This method requires the typist to look down and find a key before pressing it, and since it only uses one finger on each hand, it reduces the chance of rollover.

These two types of typists present an important distinction for use in this research because they are two extremely different methods that lead to unique results in keystroke timing. Some features are based on digraphs of a certain hand or timings between switching from the left hand to the right hand. If a user is using the hunt and peck method of typing, these features will not be useful, and in some cases, they may adversely affect the accuracy of the results. The features chosen that rely on which hand is pressing the key, are justified by the "How-we-Type" study by Feit et al. [10], [13]. Although some keys may be struck by either hand, this study examined which hand most frequently presses each key.

c. Other Factors

Other factors also have the potential to affect how a user types. Whether the user is transcribing visually from text, or if the user is thinking about what to write as they type introduces significant delay. Also, the mood that a user is in has been shown to affect performance. Khanna and Sasikumar [14] determined that a negative emotional state is more visible and recognized better using keyboard dynamics. Another factor that could have a significant impact on keyboard performance is the language of the typist. Is this person a primary English speaker, or is he typing in a different language on a non-English keyboard? These factors introduce more layers that could be used to help narrow down the identity of a user.

One study even proved it was possible to use such factors that affect typing in order to detect early motor impairment caused by Parkinson's disease [15]. The subtle changes in motor control caused by the disease were enough for properly instrumented measurements to detect the onset of Parkinson's. This study highlights the effect that age and disease can have on a person's typing behavior.

C. INTERNET PRIVACY

Identifying a user on the Internet can be an extremely difficult task. The more potential candidates that are added to the pool, the harder it becomes. Using certain characteristics to identify or narrow down the identity of an Internet user has wide reaching implications with respect to privacy. A working keystroke biometric system that could correctly identify a user on an Internet scale, in concert with other techniques, is an important development with respect to free speech and remaining anonymous on the Internet.

1. General Authorship

An Internet user who wishes to remain anonymous may be much more easily identifiable than he realizes. Although user identification on such a large scale becomes more difficult as the size of the dataset increases, there are many techniques that can be used to gain data on a user and attempt to identify who he or she is. Whether aggregating public data or using biometrics to evaluate writing style and keystrokes, this represents a serious problem for Internet users who value their anonymity.

Maintaining privacy on the Internet has been explored in many different contexts. One framework was examined by Narayanan et al. [7] which explores blog authorship on the scale of 100,000 users. This paper noticed that as the number of users increases, the classification task becomes increasingly difficult. Their analysis states: "An immediate consequence of having more classes is that they become more densely distributed…the decision boundary that separates each class now has to accurately distinguish it from a much larger number of close alternatives." Despite the challenges associated with identification on such a large number of classes, the study was able to use machine-learning techniques to correctly identify an anonymous author in over 20% of cases. In 35% of the blogs, the correct author is within the top 20 guesses.

Public data on the Internet can lead to the de-anonymization of users and disrupt Internet privacy. An example is the NETFLIX Prize dataset that was released which included over 500,000 users and the ratings that they gave to movies [16]. This seemingly harmless dataset was cross-referenced with other publicly available data to de-anonymize the users. With the increase in big data and advanced machine learning techniques, Internet privacy is rapidly becoming less assured.

2. Fingerprinting

Another way a user may be identified on the Internet is through fingerprinting of his or her browser or device. There are multiple ways that fingerprinting can occur, and this data just adds to the available information that can be used to perform identification on an Internet scale.

a. Browser Fingerprinting

The ability to track a web browser is a threat to Internet privacy. Most Internet users are aware of HTTP cookies and how they are used to track a user. Internet users wishing to maintain privacy on the web can follow certain steps to turn off cookies and not allow websites to monitor their actions. A far less recognized way of tracking a user on the Internet is through browser fingerprinting. Peter Eckersley's research [17] was the first study that performed fingerprinting on browsers based on the unique collection of versions and updates attributed to a user's browser. He concluded that in a selection of 470,000 browsers, 83.6% had a completely unique fingerprint. This percentage is even higher when the browsers being analyzed are limited to those that are running Adobe Flash or Java Virtual Machine. Even rapidly changing fingerprints were able to be tracked and guessed correctly most of the time as well. This information is all collected from available data that is voluntarily given to websites from a user's browser, and can reveal operating system and hardware data [18]. Browser fingerprinting is an Internet privacy problem that significantly increases the difficulty of a user maintaining anonymity.

b. Device Fingerprinting

Device fingerprinting is yet another way information can be determined about an Internet user without his knowledge. Each physical device has small but detectable clock skews that can be observed and used to uniquely identify a physical device. Khono et al. [19] used this information, as well as TCP timestamps and different operating system clock data, to fingerprint devices and eliminate the anonymity of users in the study. This adds to the amount of useful information that is available on the Internet to perform identification on such a large scale.

3. Methods of Anonymity

The average Internet user does not take much action to hide his identity online, and most likely does not even think twice about how his privacy is affected while using a computer. For the user that does understand what information can be collected, and wants to mitigate unwanted, anonymity-defeating, data leakage; there are a few security control options available.

One option is to use multiple different browsers and settings. By continuously switching the type of browser used, one can make it more difficult to develop a fingerprint for him. However, this method can be time consuming and result in unnecessary extra work for the user. A browser that has similar settings to many other browsers being used on the Internet is another way to lower the entropy of a browser and make it harder to uniquely identify. A downside of these methods is they require continued diligence with updating and upkeep in order to remain anonymous.

Lastly, a browser named Tor is popular for browser anonymity. Tor browser relies on the Tor network to provide free software and a worldwide network to help facilitate anonymous web browsing [20].

D. KEYSTROKE DYNAMICS AND INTERNET PRIVACY

Maintaining privacy on the Internet is an extremely sensitive and important topic within many areas. Law enforcement, cyber security and freedom of speech concerns are among the concentrations that have a stake in the Internet privacy discussion.

1. Keystroke Dynamics versus Browser Fingerprinting

Using browser fingerprinting as a method of identification can be a very successful technique. However, there are disadvantages to browser fingerprinting as compared to keystroke dynamics. Table 1 displays the characteristics of browser fingerprinting and keystroke dynamics with respect to five desirable biometric traits [2].

Table 1. Keystroke Dynamics versus Browser Fingerprinting

	Browser Fingerprinting	**Keystroke Dynamics**
Robust Over Time	No	Yes
Across Devices	No	Yes
User Dependent	No	Yes
Universality	No	Yes
Accuracy at Large Scale	High	Low

Browser fingerprinting has limitations that make it a less than ideal means of identification at an Internet scale. Browsers are not robust over time. According to Vastel et al [18], browser fingerprints tend to change frequently. They may change every few hours or every few days because of software updates or configuration changes. This presents a problem when attempting to identify a user based on his browser configuration. Keystroke dynamics of a user; on the other hand, rarely change over time. If a user's typing pattern *does* change, it is usually a result of a long-term factor such as taking a typing class or losing speed due to a medical condition, such as arthritis. Abrupt changes are possible for various reasons, such as an injury to a typist's arm, but these reasons are relatively rare. Concerning identification across devices, we see the same result. Browser fingerprinting is useless on different devices that use their own version of browser. Keystroke dynamics of a user; on the other hand, can be traced from device to device with ease. Another factor that affects browser fingerprinting, but not keystroke dynamics, is whether the form of identification is user dependent. Multiple users can use the same browser on the same computer, yet there would be no way to determine that the identity of the user had changed. Keystroke dynamics is user dependent, and if the user changes, the keystroke profile will change as well. Universality is the last factor that keystroke dynamics possesses, but which browser fingerprinting does not. Universality means that every person using the system shares a given trait. For example, it is possible and normal for a user to interact with a computer without ever accessing a browser, but it is *unlikely* that a user will interact with a computer without performing any typing. Browser fingerprinting does hold an advantage over keystroke dynamics in that it has been proven to be rather effective in determining the identity of a user on a large scale, whereas keystroke dynamics has—so far—enjoyed the same success.

2. Role of Keystroke Dynamics

Methods of Internet identification have already been covered in this research, and so have methods of anonymity. Not mentioned so far is how keystroke dynamics fit into this narrative. As shown in the last section, keystroke dynamics hold quite a few important advantages over fingerprinting within the framework of identification and Internet privacy. If the keystrokes of a user can be collected and utilized to reveal the identity of that user, that would be a significant advance in the realm of identification, but a regression in the world of Internet privacy.

III. THE KEYSTROKE DATASET

The most critical factor in this research is the size of the dataset. Determining whether keystroke biometric identification can be performed on a large scale such as the Internet requires an extremely large keystroke dataset. The dataset that this research uses originates from the paper, “Observations on Typing from 136 Million Keystrokes” by Dhakal et al. [10]. This paper reports on the findings from an online study that collected millions of keystrokes from 168,000 volunteers.

A. DATA COLLECTION

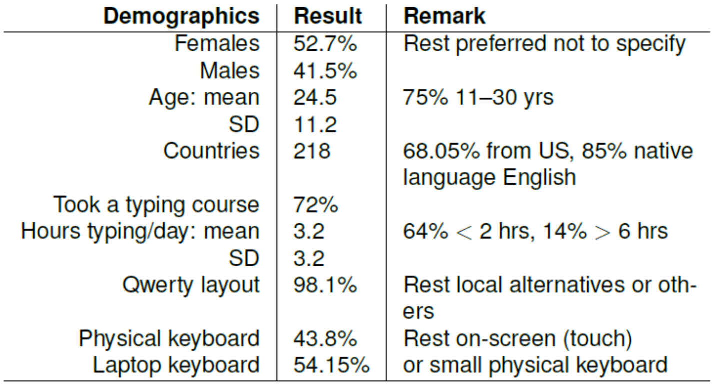
It is important for this research to take a detailed look into the data and methods that the original researchers used to compile the keystroke dataset. This section will focus on how participants were chosen, who the participants were and how the data was collected.

1. How Participants Were Chosen

How participants were chosen to participate in the study is extremely relevant. The study was hosted on a commercial site that measured typing speed. This means that the participants for the study were interested in learning how fast they type. Another pertinent fact was that there was a rather high dropout rate. Approximately 406,000 participants started the study and only 193,000 finished. After this, 12% were excluded based on too high error rates and too slow WPM (Words Per Minute) that indicated the participant was distracted at some point during the test [10]. Determining how the set of participants was chosen is important for understanding profiling implications and possible bias in the results. Even though bias may be present, overall, the study does provide a useful database of typing information for a large number of people.

2. Demographics

One of the main goals of this thesis is to determine if profiling is possible using keystroke dynamics. The demographics of the participants are of extreme importance when it comes to potentially profiling a user. It is also very important to know what kind of users participated in the study. Figure 3 displays the demographic data of the participants in the study. The chart shows the breakdown of the different demographics.



Demographics	Result	Remark
Females	52.7%	Rest preferred not to specify
Males	41.5%	
Age: mean	24.5	75% 11–30 yrs
SD	11.2	
Countries	218	68.05% from US, 85% native language English
Took a typing course	72%	
Hours typing/day: mean	3.2	64% < 2 hrs, 14% > 6 hrs
SD	3.2	
Qwerty layout	98.1%	Rest local alternatives or others
Physical keyboard	43.8%	Rest on-screen (touch)
Laptop keyboard	54.15%	or small physical keyboard

Figure 3. Dataset Demographics. Source: [10].

It is important to note that while a majority of the participants were from the US, participants hailed from countries all over the world. Most spoke English as their native language, but 15% claimed a language other than English as their native tongue. Other important facts about the participants are that they were predominately young and experienced at typing with 72% having taken a typing course and typing an average of 3 hours per day. It is also interesting to note that various keyboards were used with laptop being the most common followed by a physical keyboard.

3. How Data Was Collected

Each volunteer was instructed to type 15 English sentences and the timing data was collected from the keystrokes. The sentences were selected randomly from a set of 1,525 sentences that were taken from the Enron email corpus as well as a few other sources representing simple, common typing tasks [10]. Figure 4 shows a few examples of sentences taken from the online test. A sentence was displayed to the user, who was instructed to read it fully and then type it as fast as possible. At the end of the test, the participants were asked to give demographic information about themselves.

Foreign insurance firms can file applications to the ministry.
He said the operating theatres were barely functioning.
I am meeting with my direct reports at 1 to work on recommendations.
She probably won't come next year either.
Song was vice minister of the geology and mineral resources minister.

Figure 4. Example Sentences from the Typing Test. Source: [10].

4. How Data Was Presented

The data was collected and put into multiple files representing each distinct keystroke’s press and release timing as well as the participant and session it was associated with. This led to a total dataset of over 136 million entries of keystroke data. Another file contained the metadata relating to the participant, e.g. gender, nationality, age and other demographic information previously described. Tables 2 and 3 represent examples of the datasets described. Table 2 shows five keystrokes in one user’s session. The key code corresponds to the JavaScript Event Keycode of the key that was pressed. In this example, the user pressed Shift, W, A, S, Spacebar (“Was ”). Table 3 depicts what the metadata file looks like. It contains the user along with the corresponding demographic information. Not all columns could be displayed in this form and other statistics were present in this dataset such as average WPM, IKI and ROR.

Table 2. Keystroke Data Example

User	Session	Press Time	Release Time	Key Code
100001	1090979	1473275372512	1473275372663	16
100001	1090979	1473275372583	1473275372703	87
100001	1090979	1473275372759	1473275372903	65
100001	1090979	1473275372831	1473275372975	83
100001	1090979	1473275372943	1473275373079	32

Table 3. Metadata Example

User	Age	Gender	Typing Course	Country	Layout	Native Language	Fingers	Hours Per Day	Keyboard
3	30	none	0	US	qwerty	English	1-2	8	full
5	27	female	0	MY	qwerty	English	7-8	6	laptop
7	13	female	0	AU	qwerty	English	7-8	0	laptop
23	21	female	0	IN	qwerty	English	3-4	0	full
24	21	female	0	PH	qwerty	Tagaleg	7-8	1	laptop

5. Keystroke Data Visualization

An abundance of interesting statistics can be uncovered while investigating the large amounts of data provided by this study. Inferences regarding typing speed and the way a user's WPM correlates to the error rate and the roll over rate in his typing are valuable results of this study. The study by Dhakal et al. had valuable contributions. It was determined that there is a large variance in typing speed with the average speed being 52 WPM. Roll over key pressing is very prevalent and has a strong correlation with typing speed. Also, users with a faster typing speed make less errors than slow typists. Figure 5 shows a histogram of typing speeds and the number of users exhibiting each speed. Most users type in the 30 to 60 WPM range. Figure 6 clearly shows a positive relationship between ROR and typing speed. The fastest typists routinely have RORs well above 50 percent. Figure 7 represents the relationship between typing speed and error rate. There is a clear negative trend and as the WPM increases, the error rate decreases.

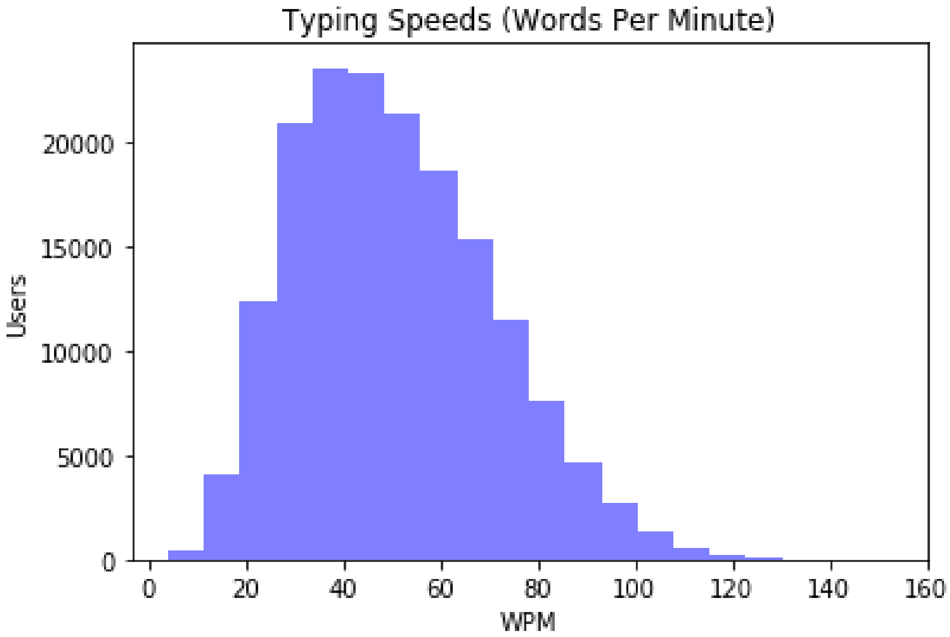


Figure 5. Histogram of Words per Minute of Users in the Keystroke Dataset

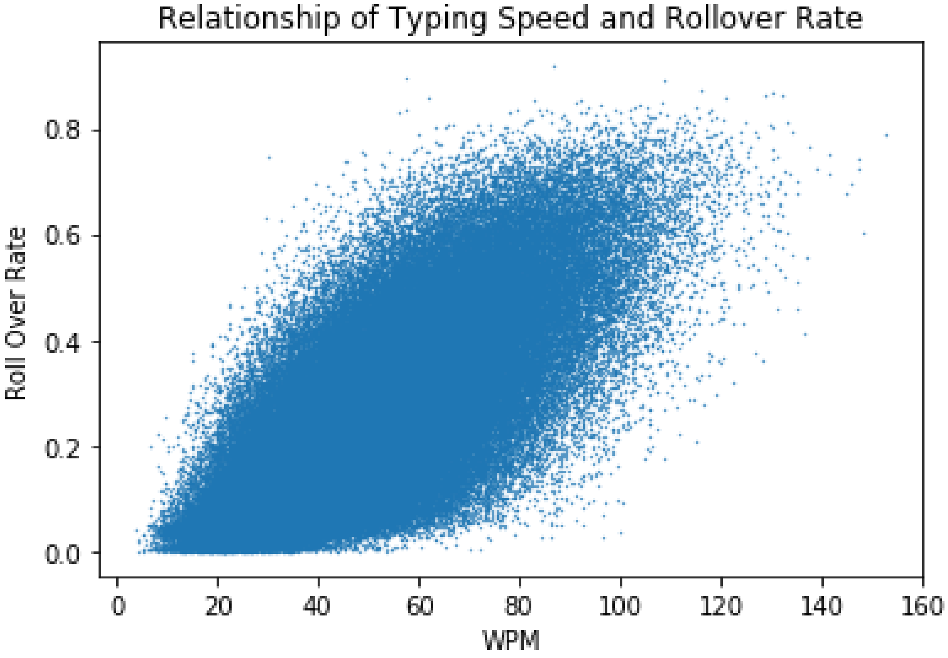


Figure 6. WPM versus ROR

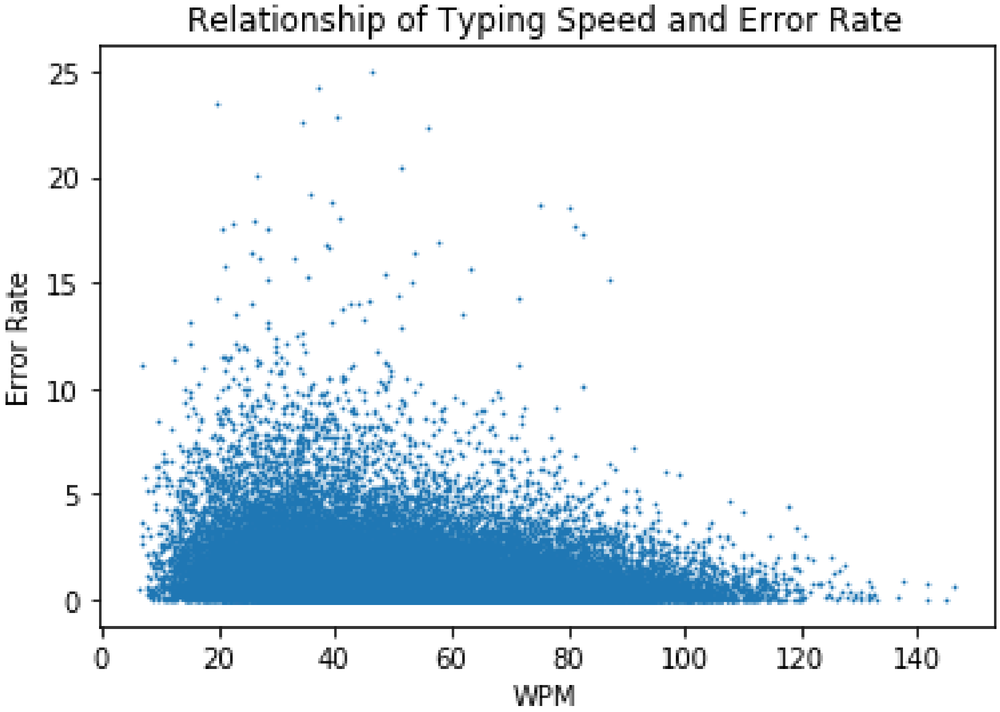


Figure 7. Error Rate versus Typing Speed

B. LIMITATIONS OF THIS DATASET

Limitations of using this dataset must be considered. The data that was collected was wide-ranging and extensive, making it ideal for the large-scale research conducted in this thesis. However, it may not accurately represent the population of users on the Internet for a few reasons. The participants were not chosen randomly. They visited the commercial typing site first and were asked to participate in the study. Since the website was related to typing, the visitors to the site were most likely overwhelmingly interested in how they type and becoming better typists. Also, the participants in the study presumably do not accurately compare to a cross-section of Internet users. With 75% of the participants between the ages of 11 and 30, as well as most of them being from the United States with the primary language of English, this could introduce some bias into the research. Lastly, the participants most likely performed the test by transcribing the sentences and attempting to type as fast as possible which may not represent how most computer users perform such tasks as email, web browsing, or document creation. While the data may not correctly characterize the average Internet user, it is still the largest set of keystroke data that has been collected to date, and is extremely useful for research efforts directed at improving the performance of identification on large-scale user groups.

THIS PAGE INTENTIONALLY LEFT BLANK

IV. BASELINE APPROACH

The first part of the experimentation in this thesis is related to the steps and results that were achieved while establishing an identification accuracy baseline from this dataset using handcrafted features and conventional machine learning techniques. Establishing a baseline is necessary because the second part of the experiment involves improving on that accuracy using a different approach: a neural network. Obtaining this baseline measurement required addressing several challenges due to the size of the dataset and number of classes.

It was necessary to learn how to work with such a large dataset with so many classes to identify. Many classification problems involve binary classification, such as determining whether or not a patient has a disease, or in quality control, does the object in question meet the criteria or not. Other problems attempt to identify based on a small number of classes such as the Iris plant dataset [21], an important machine learning dataset that seeks to classify plants into one of three species of Iris flowers based on the features of the flower. It is much less common to attempt to identify based on over 160,000 classes using hundreds of features, as was the effort in this thesis. Some recent research has been conducted using extremely high numbers of classes. In “What Does Classifying More Than 10,000 Image Categories Tell Us?” [22], the authors describe that size of the dataset is an extremely important factor. They concluded that some classifiers that might work well on small amounts of the data might not fare well when using the same methods on high numbers of classes. In the paper, “Training Highly Multiclass Classifiers” [23], the authors also discuss the difficulty of increasing the number of classes and attempting to perform identification. The authors state, “In practice, the more classes considered, the greater the chance that some classes will be easy to separate, but that some classes will be highly confusable.” These are problems that we encountered as well when attempting to classify using such high numbers of classes.

Another question that needed to be addressed was determining which machine-learning algorithm would be the best to use in this scenario. Quite a few approaches can be taken. In the study by Killourhy and Maxion [24], “Comparing Anomaly Detection

Algorithms for Keystroke Dynamics," the authors evaluated a few different classification algorithms that they gathered from keystroke dynamics literature. Because a simple Manhattan distance detector performs well, we used that approach in obtaining our baseline results.

Lastly, before getting to the results of the baseline performance, we needed to address the computational complexity of the preprocessing, feature extraction, and classification algorithms. Even simple tasks, such as reading the dataset from disk and manipulating the millions of lines of data, took too much time for a personal computer to handle.

This chapter will describe the steps that were taken to combat these problems and answer the questions above. It will address failures that we experienced and successes that were encountered. Lastly, results of the baseline experiments will be presented.

A. METHODOLOGY

Establishing a baseline for identification accuracy using this dataset turned out to be a more difficult task than expected but it provided some very interesting and useful results. It was necessary to adjust for a few roadblocks that we encountered while running tests. The design of the baseline experiments evolved over time. The following sections describe the steps taken to eventually obtain results.

1. Feature Extraction

The first steps that needed to be taken in order to experiment with the data involved data manipulation. Each user typed 15 unique sentences (sessions), and each session had a range of 20 to 700 keystrokes. Figure 8 depicts a histogram of keystrokes per session. Most fall into the range of 0 to 120, but a very small number of outlier users performed larger numbers of keystrokes. The dataset presented in this paper recorded the press time and release time for every keystroke. Very generic descriptive features were calculated to sum up the typing of each participant as well, such as typing speed and rollover rate. This information describing an overview of the typing within this study was sufficient for the paper, but our research required many more features to be calculated. The individual features needed to be calculated from all the keystrokes of a session and then combined into a single row that corresponds to each user's session. Completing this step reduced the number of rows in the data frame from over 160 million to around 2.5 million while creating 218 feature columns. Figure 9 depicts the process of creating features from the keystroke data and converting it to the desired format for the machine learning algorithm. The following sections will describe how those features were calculated and what features were used to identify users in this study.

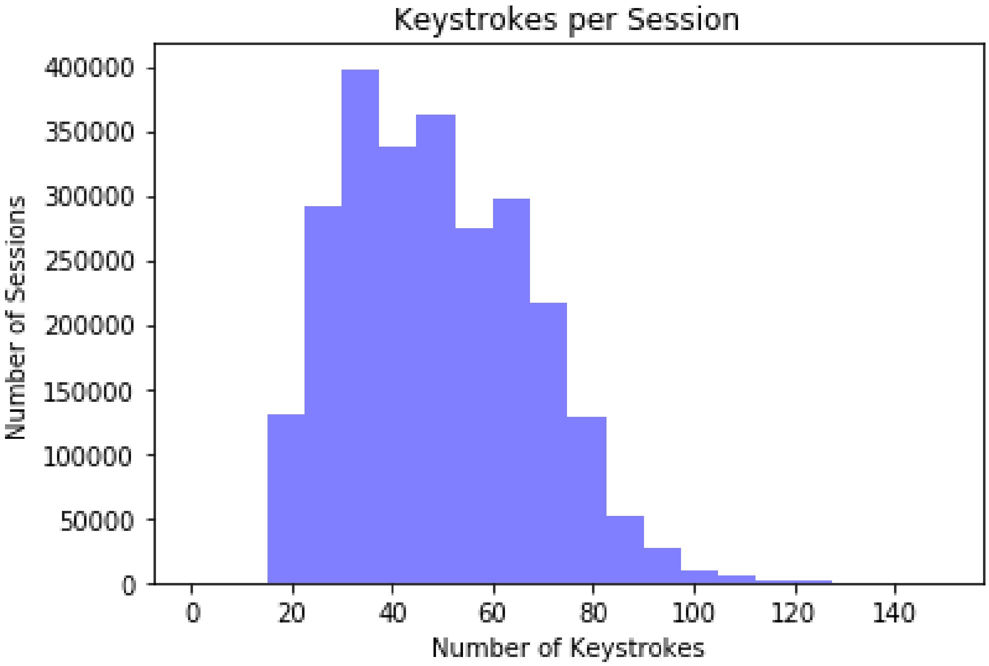
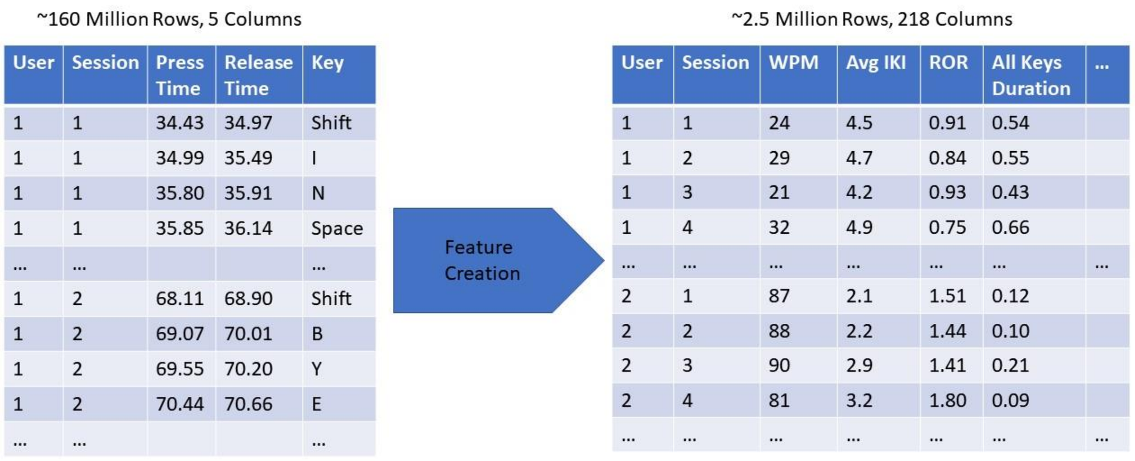


Figure 8. Number of Keystrokes per Session



~160 Million Rows, 5 Columns

User	Session	Press Time	Release Time	Key
1	1	34.43	34.97	Shift
1	1	34.99	35.49	I
1	1	35.80	35.91	N
1	1	35.85	36.14	Space
…	…			…
1	2	68.11	68.90	Shift
1	2	69.07	70.01	B
1	2	69.55	70.20	Y
1	2	70.44	70.66	E
…	…			…

Feature Creation

~2.5 Million Rows, 218 Columns

User	Session	WPM	Avg IKI	ROR	All Keys Duration	…
1	1	24	4.5	0.91	0.54	
1	2	29	4.7	0.84	0.55	
1	3	21	4.2	0.93	0.43	
1	4	32	4.9	0.75	0.66	
…	…	…	…	…	…	…
2	1	87	2.1	1.51	0.12	
2	2	88	2.2	1.44	0.10	
2	3	90	2.9	1.41	0.21	
2	4	81	3.2	1.80	0.09	
…	…	…	…	…	…	…

Figure 9. Format of Data Before and After Feature Creation

In order to gain the most information out of this dataset, it was necessary to calculate as many features as possible. Considering we were only provided with the press and release times for each key, plus a small number of general statistics such as WPM and IKI (Inter Key Interval), many features needed to be added to fully capture each user's keystroke dynamics. Keys were divided up into vowels / consonants, right hand keys / left hand keys and letters / non-letters. Durations were calculated based on which key or what type of key was being pressed. Latencies were also calculated between specific keys and between the hands of the typist. Means and standard deviations were then calculated based on these key sets.

This method of creating features led to a total of 218 features which needed to be calculated based on the press and release timings as described in the previous section. Table 4 describes some of the features broken down into the categories of general, duration and transition. It is important to note that for the duration features, each type of feature can be divided further into the mean and standard deviation of those calculations. For the transition features, each type of feature can be divided into mean and standard deviation, as well as press-to-press and release-to-press transitions. The list in the figure is not exhaustive and does not show every single combination of transition that can be calculated. It does, however, represent a majority of the features chosen and helps to explain how 218 features can be created from only the press and release times of a key by taking into account the category of key that is pressed. This list of features is based on those used in [25].

Table 4. List of Features. Adapted from [25].

General Features	Duration Features (Mean and SD)	Transition Features (Mean and SD), (PP and RP)	
Average WPM	All Keys	All Keys - All Keys	Individual Keys - Individual Keys (a - b, s - t, …)
Average IKI	Letters	Letters - Letters	Double Letters
Error Correction Per Char	Vowels	Consonants - Consonants	Letters - Space
Keystrokes Per Char	Consonants (different levels of frequency)	Vowels - Consonants	Shift - Letters
Roll Over Rate	Left Hand	Vowels - Vowels	Punctuation - Space
	Right Hand	Right Hand - Left Hand	digits - digits
	Non-Letters	Right Hand - Right Hand	Individual Keys - Individual Keys (a - b, s - t, …)
	Punctuation	Non-Letters - Non-Letters	
	Digits	Punctuation - Punctuation	
	Each Key on the Keyboard (a, b, c, …, space, shift)	digits - digits	

2. Preprocessing

Next, we needed to split the dataset into training and testing sets that we could feed into the classification algorithm of our choice. Splitting into a train and test set comes with decisions that need to be made about how to accomplish this task. For each user, we chose to use two thirds of the data for training and the remaining one third for testing. It was also very important to ensure that the data was stratified so that the same number of sessions were taken and used as training / testing data for each user instead of randomly chosen. This resulted in 10 samples per user for training and 5 samples for testing.

The experiment that we designed called for the simple task of choosing different classification algorithms and testing them on the dataset to determine the accuracy of identifying a user. With the very first try, it was evident that this idea was going to need to be revised.

The failures that we experienced were related to the size of the dataset. With over 160,000 users that we were attempting to identify, each one represented a class that the classification algorithm had to take into account. Upon attempting to run experiments, we were immediately running into memory errors, scripts taking too long to run and timed out programs. We determined other methods would be needed in order to obtain better results.

We began attempting different ways to handle the large number of classes and memory issues. Some of these methods failed and some were a success. One approach that we attempted was to split up the dataset into much smaller subsets in order to run different classifiers and compare results on a usable set of classes. This worked well and we were able to get accuracy results for K-nearest Neighbor, random forest and support vector machine classifiers. Parameter tuning was also performed to understand the change in accuracy as the parameters were altered.

Another method that proved necessary throughout the research was to use the Naval Postgraduate School's High Performance Computing (HPC) infrastructure to enable experiments that otherwise would not be possible on a standard desktop or laptop. By sending scripts to the available nodes, we were generally able to avoid receiving memory errors and significantly speed up processing. Experiments were performed on a Linux server using an Intel Xeon E5-2683 processor with 503 GB DRAM in a Python environment using scikit-learn, pandas, and numpy.

Not all of the methods we attempted were successful. We tried utilizing principal component analysis (PCA) to significantly reduce the dimensionality. After reducing the dimensionality to various levels such as 20, 50 and 100 instead of the 218 features that we calculated, it was determined that this method affected the accuracy much too drastically to be feasible. Some of the classifiers, such as the random forest classifier, that we initially used to calculate accuracies on the small datasets of 100 or 1000 users instead of 160,000 showed promise and were extremely accurate on such small numbers of classes. However, they did not scale well, and these methods were not able to be used due to the time it took to fit data and make predictions using larger numbers of users.

3. Classification

For the baseline tests, we decided that one classifier was to be used to maintain consistency in the results. This classifier should be simple enough to be able to run on the entire dataset and should be a well-known algorithm that could produce a decent baseline accuracy. K-nearest neighbor (KNN) using Manhattan distance was chosen because this distance metric had been used successfully by Killourhy and Maxion's [24] work in comparing keystroke dynamic classifiers. KNN with Manhattan distance consistently performed well and was fast enough to be able to classify the full dataset in a reasonable amount of time. KNN is a simple classification algorithm that relies on the assumption that similar objects will be close in proximity to each other in a vector space. The algorithm calculates distances between points and classifies a data point based on closeness to its neighbors. In the case of this study, KNN examined each user on a multi-dimensional plane using all the features calculated from the keystrokes. KNN can use any distance metric to calculate the distance between points and we chose to use the Manhattan distance. Manhattan distance is the "city block" distance between two points, which is the sum of the absolute differences in each dimension. Another parameter that needed to be chosen was the number of neighbors to take into account. We chose k = 1000 and weighted by inverse distance because of the large number of classes in the dataset. After fine-tuning the k-nearest neighbor classifier, we were ready to gather results.

4. Accuracy Metrics

One of the most important results was determining the accuracy of the classifier on different sizes of the dataset. Due to the large number of classes, simply taking the accuracy with which the classifier correctly chooses the user would not give us enough information about the performance of the classifier. We used a similar strategy as the researchers in [7] to display classification results on a large dataset of blog writers. We calculated the rank of the correct prediction, which gives a much better understanding of how well the identification narrows down the results. For example, instead of only calculating what rate the classifier chose the correct user, we determined the rate at which the correct user fell into the top N predictions. For the variable N, we used ranks of 10, 100 and 1000.

Finding another method for calculating the accuracy on the entire dataset was necessary due to the amount of time that the classifier was taking. While attempting to identify users based on the full dataset of over 160,000 classes, the classifier ran for multiple days and failed to return results. To get around this, we subsampled the number of test examples. This way we could still get an accuracy that was very close to what it would have been training the classifier on the whole dataset, but would complete within a reasonable amount of time.

5. Profiling

The last step in the baseline experiment design was to calculate whether a user could be profiled based on the way he types. For example, profiling would involve taking a user that is not in the dataset and attempting to discern demographic traits based only on keystroke data. The categories that we used were gender, country, native language, age, typing skill and type of keyboard. For each category, the data needed to be processed and normalized into a certain number of classification groups. Gender was already in two distinct classes, male or female, but any users who did not report their gender needed to be removed. In order to convert country into a workable profiling problem, we needed to create a binary classification problem by converting the country of the user into one of two categories: US or non-US. Age required separating the users into four groups that were roughly the same size. Skill and type of keyboard demanded some pre-processing as well to eliminate unusable data. Next, we determined the chance accuracy of the category by finding the percentage of the largest group. This was the number that we needed to improve upon to see if profiling a user was possible.

B. RESULTS

This section describes the results that were achieved by using the baseline method described in the previous sections.

1. Accuracy

Much of the data collected was to identify at what accuracy the baseline machine learning technique could identify a user based on his or her keystroke patterns. Figure 10 shows the percentage accuracy for successfully identifying the correct user (top 1 accuracy), as well as the top 10, top 100 and top 1000 accuracies using different amounts of users from the dataset. As you can see from the figure, as the number of users increases to include the entire dataset, the accuracies for each of the rank estimations mostly converge to a single value. These values are presented in Table 5. Also included in Table 5 are the random chance accuracies that could be accomplished by randomly selecting a user instead of using machine learning techniques to take keystroke features into account. While still not providing extremely confident accuracy numbers, the accuracy values determined from the techniques presented in this thesis represent a significant increase in performance from selecting a random user.

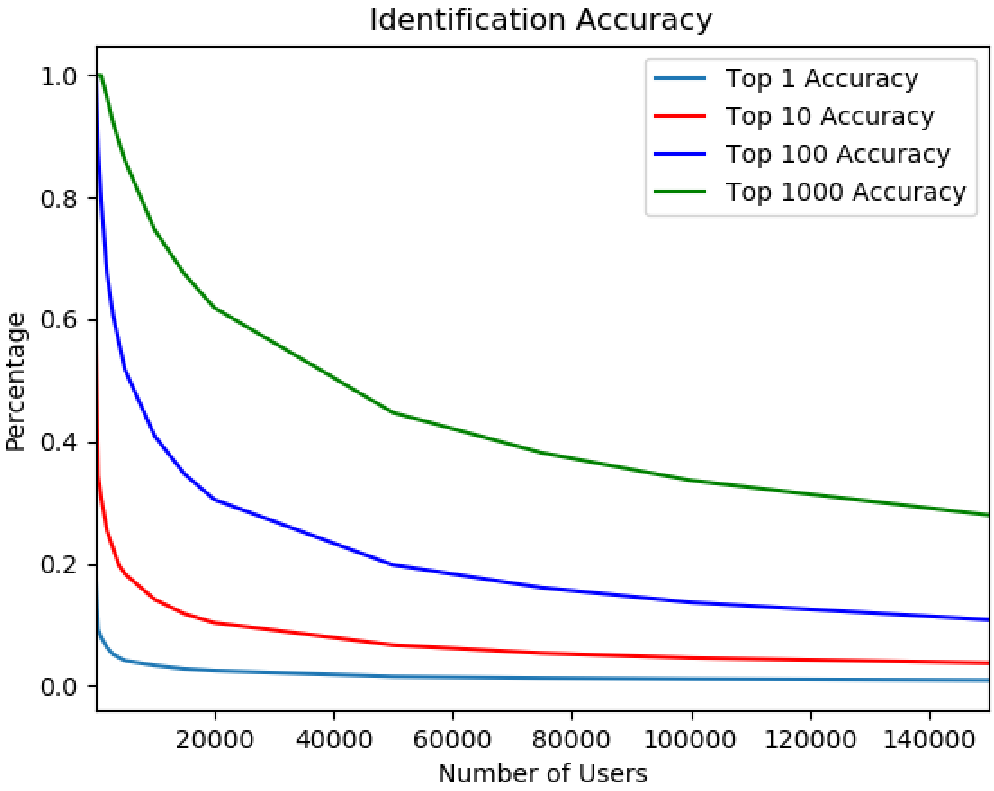


Figure 10. Top 1, 10, 100 and 1000 Identification Accuracies for Varying Numbers of Users

Table 5. Achieved Accuracy versus Random Chance Accuracy Using Entire Dataset

Rank	Achieved Accuracy	Random Chance Accuracy	Accuracy Improvement
1000	28%	0.7%	40
100	10.8%	0.07%	154.3
10	3.7%	0.007%	528.6
1	0.9%	0.0007%	1285.7

2. Time

Time is an extremely important factor in the experiments presented in this thesis. The running time of the machine learning techniques caused numerous methods and workarounds to be devised in order to make results feasible. Figure 11 displays the time in hours that the program took to run using an increasing number of classes / users. These times were calculated using resources on the Bowditch HPC at the Naval Postgraduate School. In addition, it is important to note that these times were observed while running one of the simplest classification algorithms, KNN. At 100,000 users, the time it takes to evaluate the accuracy approaches 24 hours and is increasing exponentially. As previously described, the time it takes to run the program on the entire dataset is not presented in this chart because an alternate method was used to calculate the accuracy for the entire dataset due to the amount of time it was taking to run for that many users.

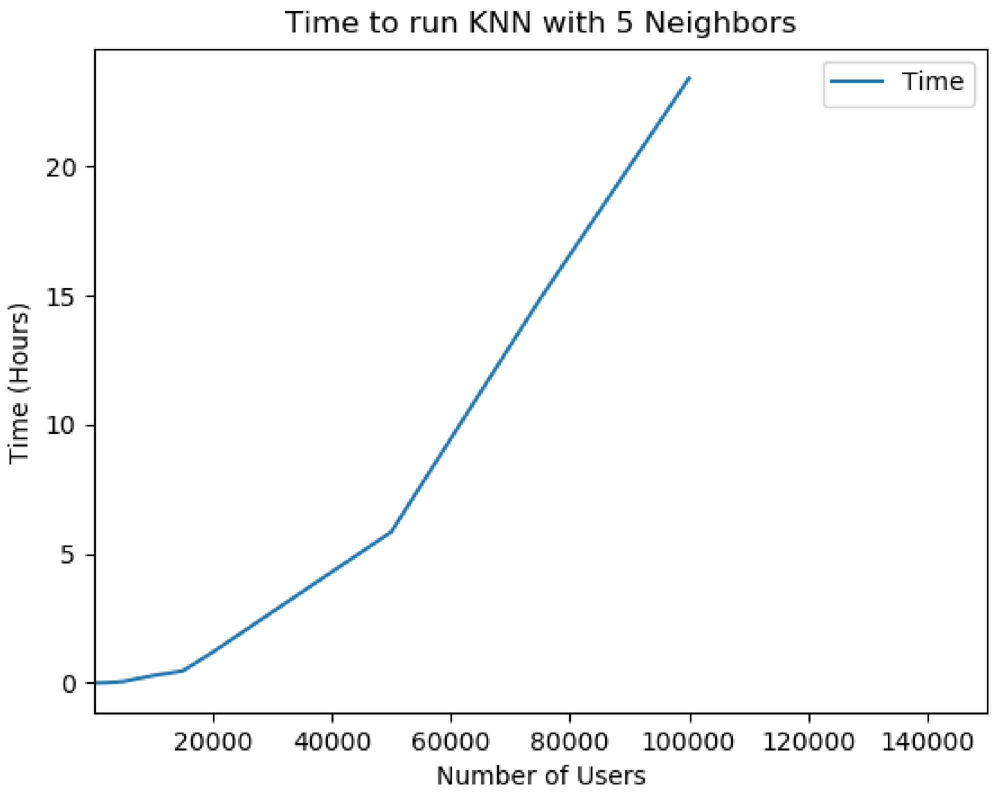


Figure 11. Time in Hours to Calculate Accuracy as Number of Users Increases

3. CDFs

The cumulative distribution function (CDF) of the rank that a user was identified using the classifier gives a useful way of presenting the accuracy data. As the rank increases, the CDF describes the percentage of users out of the total that were identified at that rank or better. As figures 12-16 show, there is a knee in the graph at around the 50% mark. An inference that can be made from these charts is that roughly half of the users are easy to identify and the other half are difficult to identify. Another interesting observation is that the number of users does not significantly affect the shape of the graph

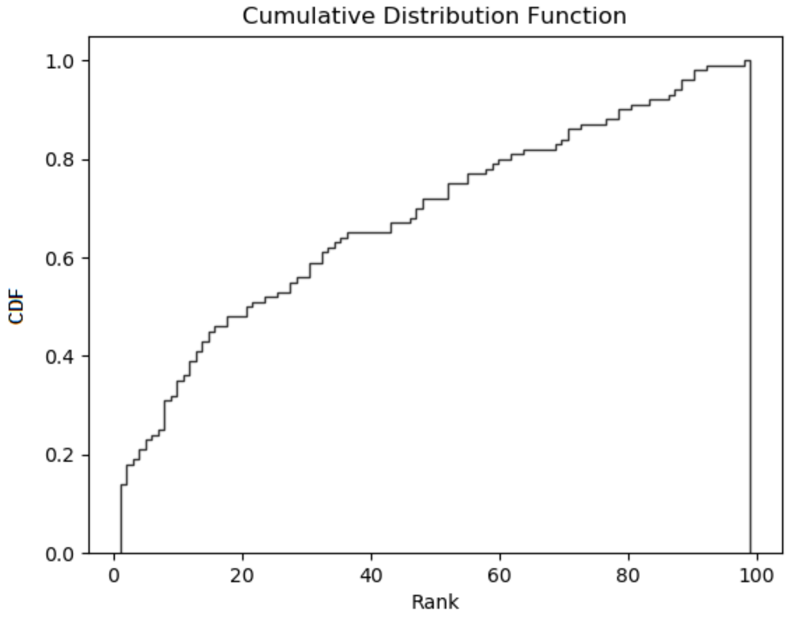


Figure 12. Cumulative Distribution Function 100 Users

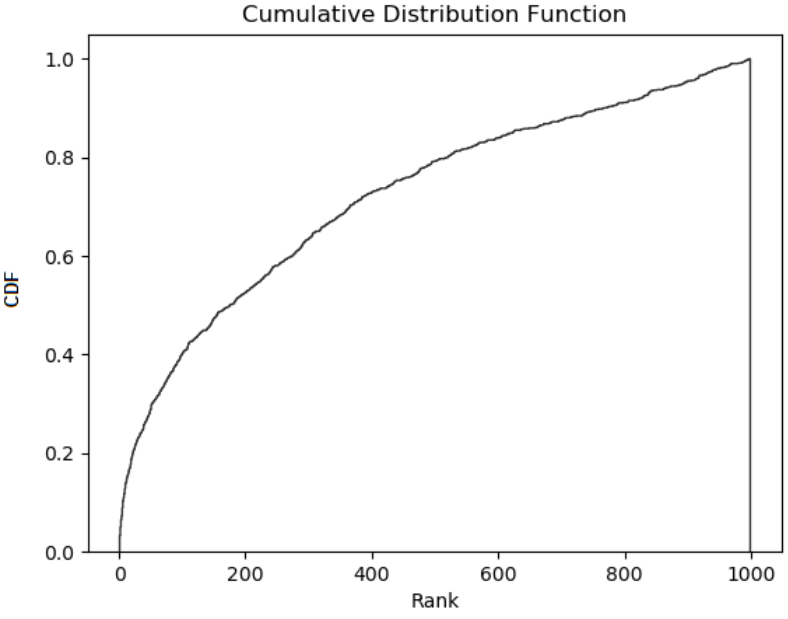


Figure 13. Cumulative Distribution Function 1000 Users

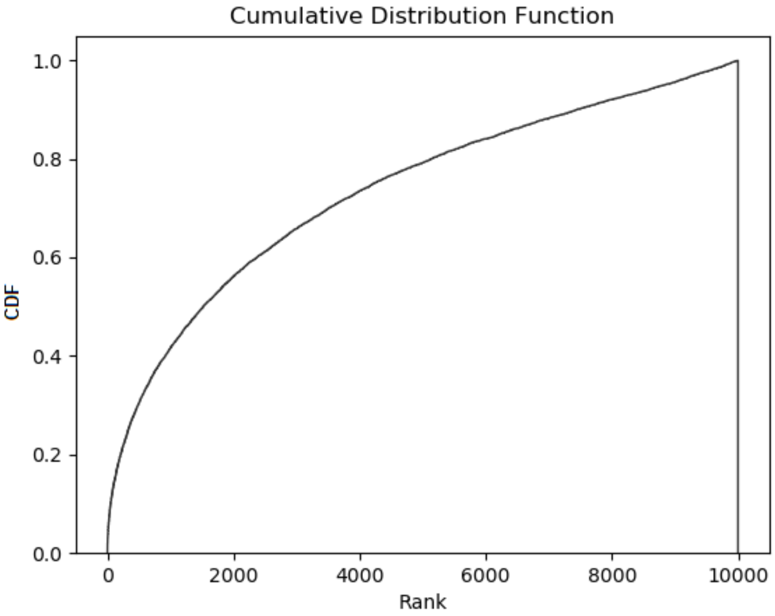


Figure 14. Cumulative Distribution Function 10,000 Users

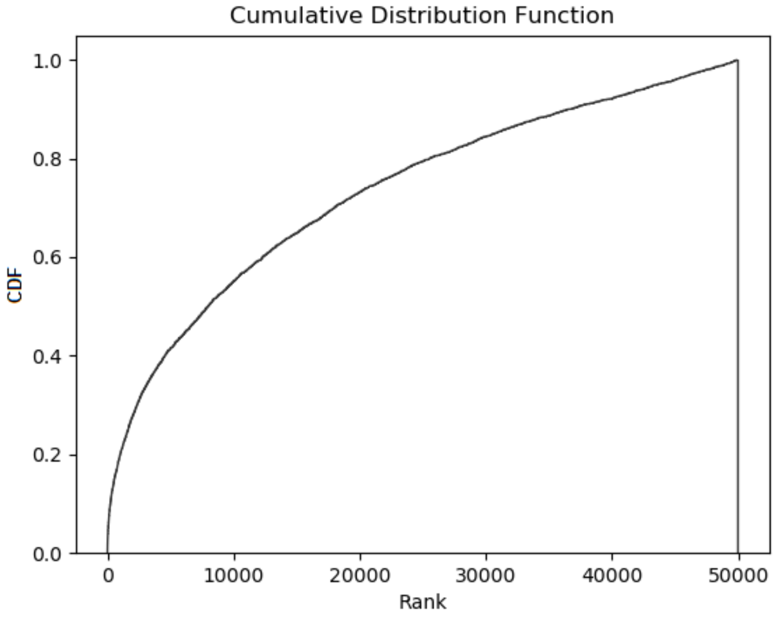


Figure 15. Cumulative Distribution Function 50,000 Users

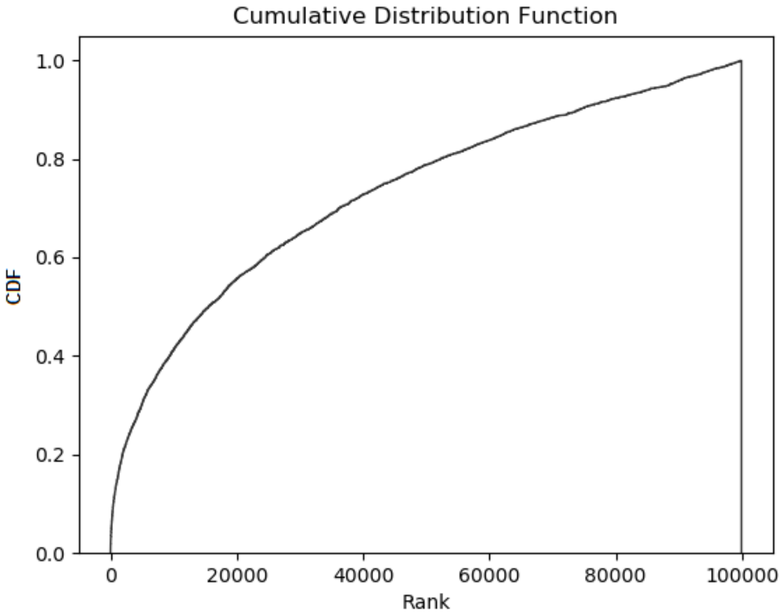


Figure 16. Cumulative Distribution Function 100,000 Users

4. Profiling Users

Using the metadata regarding the users that was provided by the dataset source, this thesis also focused on determining whether profiling a user based on certain demographic statistics was possible. Profiling based on demographics does not include the same problems of extremely large numbers of classes that we faced while identifying users. This is because in these cases, we only have a few classes. For example, we are not trying to identify the user, we are classifying into either male or female, or one of a small number of age groups. For these tests, we used the same KNN classifier we had been using previously and also introduced the classifier XGBoost which we knew would perform better since we were not facing size and time limitations. Figure 17 shows the results based on gender, country, native language, age, typing skill and keyboard type. The results show an increase in accuracy of profiling a user in almost all demographics. The most significant increases occurred when identifying based on gender and age. Language and skill were difficult to improve upon because the accuracies are already very high. Also of interest is that KNN did not perform much better than the baseline in this case. XGBoost was much more useful than KNN in this context.

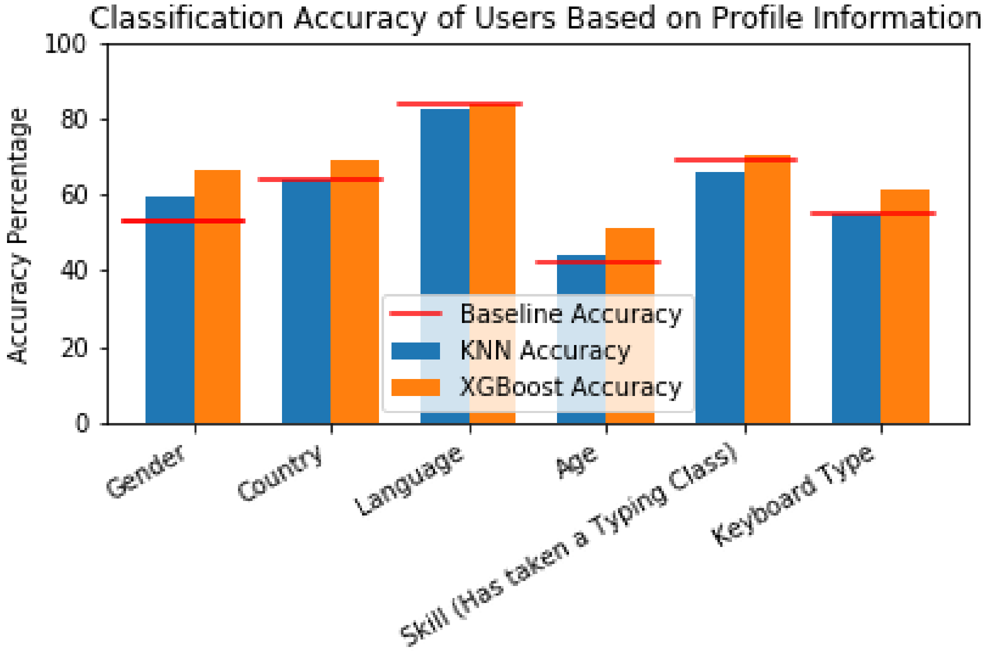


Figure 17. Classification Accuracy of Users Based on Profile Information

THIS PAGE INTENTIONALLY LEFT BLANK

V. NEURAL NETWORK APPROACH

In this phase of the research, we trained a deep neural network to learn as much as it can about the individual keystrokes, and then use this information to identify users based on his or her unique typing characteristics. We then evaluated the resulting accuracies achieved by the Neural Network and compared them with the results of the KNN classification algorithm.

The experimental design was modeled around the ImageNet research paper written by Krizhevsky et al [26]. In this paper, the authors created a novel new technique for identifying images based on a deep convolutional neural network. The method in this paper used a Neural Network with multiple convolutional layers that could identify the objects contained in an image based on the individual pixels in the image. It increased the identification accuracy of the ImageNet database significantly and was the state of the art at the time the paper was released.

In “A Guide to Convolutional Neural Networks for Computer Vision”, Khan et al [27] describe the characteristics of a CNN and its significance. CNNs are “essential for cases where we want to learn patterns from high dimensional input media, e.g., images or videos. CNN filters incorporate spatial context by having a similar (but smaller) spatial shape as the input media, and use parameter sharing to significantly reduce the number of learnable variables.” CNNs are highly effective at learning spatial relationships in a structure such as an image with a large number of pixels. CNNs achieve this by performing a series of convolutions over an image. A convolution refers to a mathematical operation in which a filter is convolved with an input image. As the filter slides across the image, weighted sums are calculated and then pass through a nonlinear “squashing function.” The values are then passed along to the next layer of the network.

While examining the keystroke data during this research, we determined it would be useful to take into account the spatial relationships of the keys on the keyboard while a person is typing. We wanted to create a neural network that could examine the durations and latencies of key presses while considering their locations on the keyboard that correspond to the finger and hand that type them. For example, keys “P” and “E” on the keyboard are located many keys away from each other and will most likely be typed using different fingers on different hands of the user. We wanted to make the most of these locational differences and use a neural network to respect these kinds of spatial relationships. In order to accomplish this, we decided to use a convolutional neural network. However, the format of the data lends little support to this neural network structure. We needed to convert the data into an array that mimics the layout of a multi-pixel image or the frames of a video. The bulk of the work for this approach resided in the preprocessing and data presentation to conform it into a format of this type. The next section describes this process.

A. METHODOLOGY

The following sections describe the process taken in order to prepare data for the neural network as well as the remaining steps required to obtain experimental results. Significant manipulation of the keystroke data was needed to create a data structure with the proper format that a convolutional neural network could ingest and take advantage of the spatial relationships between keys. After preprocessing was accomplished, many more design decisions needed to be made in order to tune the neural network and obtain the highest accuracy possible.

1. Keyboard Grid

The first step of this process was to create a keyboard grid that would represent the locations of the keys in a two-dimensional space. Creating this keyboard array was essential for the neural network to be able to take the locations of the keys into account during the learning process.

In order to represent a keyboard in this format, it was necessary to research the most common keyboard layouts and determine the best way to arrange the keys in a row and column presentation. The standard keyboard layout in the United States is the ANSI 101/104 keyboard. This keyboard layout is commonly found on laptops and desktops throughout the United States. While many different types and layouts of keyboards were used to collect keystroke information in the study, most of the participants were from the

US and likely use the ANSI 101/104 standard. Besides a few differences in the size and shape of the shift and enter keys, most keyboards do not differ drastically with the ANSI standard keyboard, so we decided to use it as our template.

We discovered that a keyboard can be represented by a 5 x 15 two-dimensional array and include all of the keys. As shown in Figure 18, in order to represent a keyboard in an array of this type, some keys take up multiple array locations. For example, the spacebar in this model takes up six columns on one row. Due to the diagonal nature of most keyboard keys, they also needed to be shifted in order to fit into an array with straight columns and rows. This was performed by sliding certain rows to the left until they match with the row above them.

~		1	2	3	4	5	6	7	8	9	0	-	+	BACK
TAB		Q	W	E	R	T	Y	U	I	O	P	[	]	\
CAPS	CAPS	A	S	D	F	G	H	J	K	L	;	‘	ENTER	ENTER
SHIFT	SHIFT	Z	X	C	V	B	N	M	,	.	/	SHIFT	SHIFT	SHIFT
CTRL		OS	ALT	SPACE	SPACE	SPACE	SPACE	SPACE	ALT	ALT			CTRL	CTRL

Figure 18. Keyboard Layout

2. Preprocessing

We began the neural network method with the same keystroke data as the baseline method. It was in the format of a data frame that presented the user, session, press and release times and key pressed for every keystroke in the study. This format was not adequate for a neural network. Neural networks require multidimensional arrays, also called tensors, as inputs.

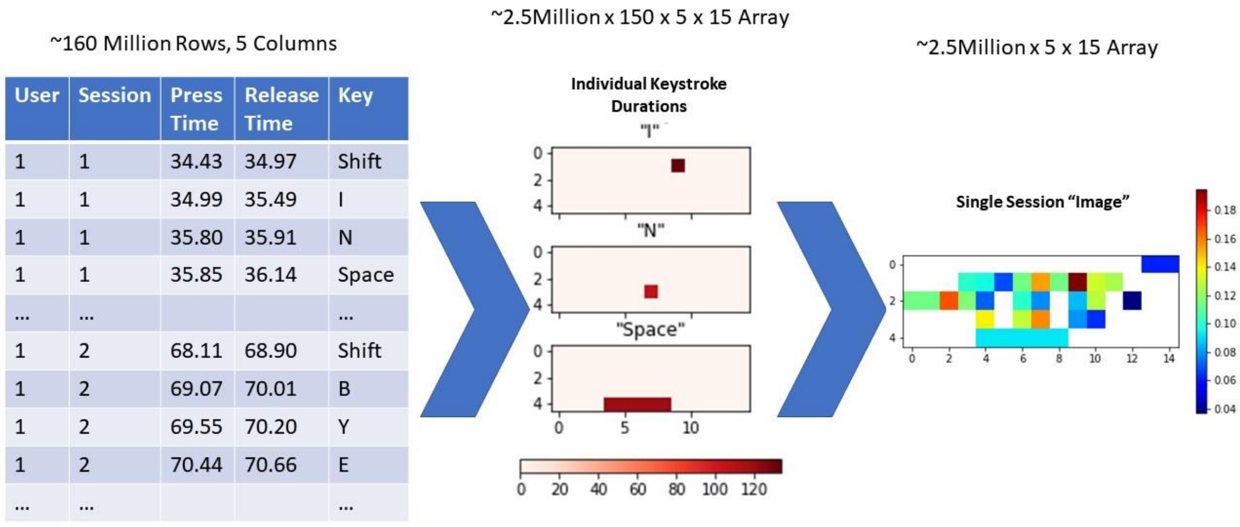
To begin, we created a five-dimensional array with shape: (~2.5 Million x 150 x 5 x 15 x 1). The number 2.5 million represents the number of sessions in the data (~165,000 users multiplied by 15 sessions per user). The second dimension, of length 150, represents the number of keystrokes per session. Recall from Chapter III that the number of keystrokes per session in the dataset ranges from 10 to over 700 keystrokes. However, only a small fraction (0.0001%) of the sessions have more than 150 keystrokes. If a session contained more than 150 keystrokes, it was cut off at 150. Similarly, when a session had less than 150 keystrokes, we padded the array with zeros so that every session contained the same length of keystrokes. The next two dimensions, 5 x 15, represented the keyboard array and the location of the single key on that array. If a key was larger than one column, it was represented with multiple array locations. Lastly, the final dimension in our array denotes the calculated duration of that key press.

3. Adding Features

To begin, the final dimension included only the duration. Once the array was created and functioning, it was necessary to include as much information as we could in the array. Besides the duration, there are four other latencies that could be calculated. These latencies are the PP, PR, RP and RR latencies. Each one was calculated using the press and release times of the keystrokes in each session. After inserting this data into the array, we were left with an array with dimensions: (2.5 million x 150 x 5 x 15 x 5).

4. Shaping the Data

The keystrokes were then contained in a multidimensional array but the preprocessing was still not complete. The neural network needed full session data as an input instead of individual keystrokes. The array contained 150 individual keystrokes for each session. In order to get the average for a session and better represent the typing patterns of the users, the next step was to calculate the average durations and latencies over the entire session. Figure 19 shows the complete process of transitioning from the keystroke dataset to the sequence of keystrokes contained within a 5 x 15 keyboard array, and finally, to the average of a single session which contains all the keystrokes represented as a single array.



~160 Million Rows, 5 Columns

User	Session	Press Time	Release Time	Key
1	1	34.43	34.97	Shift
1	1	34.99	35.49	I
1	1	35.80	35.91	N
1	1	35.85	36.14	Space
…	…			…
1	2	68.11	68.90	Shift
1	2	69.07	70.01	B
1	2	69.55	70.20	Y
1	2	70.44	70.66	E
…	…			…

Figure 19. Preprocessing Steps

For each session, a 5 x 15 array was created for the duration as well as the four latencies. Each session now had five distinct arrays with the per-session averages laid out in a keyboard array as depicted in Figure 20.

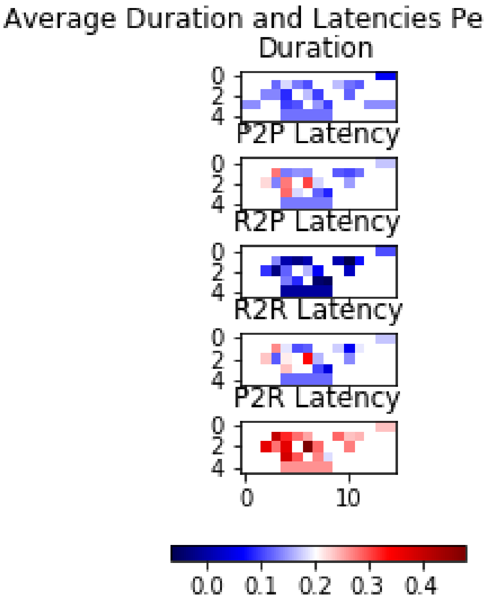


Figure 20. Duration and Latencies per Session

This method is very similar to the way in which a video is depicted using a multidimensional array. A video is in the format: (frames x pixels x pixels x color channels). In the video example, the frames would be equivalent to the session. The two pixel dimensions would be represented by our 5 x 15 keyboard array. Finally, the color channels, namely red, blue and green, each have an intensity value associated with them; much like our 5 feature channels that consist of mean duration, PP, PR, RP, and RR latency.

In order to demonstrate how a neural network would be able to distinguish between users, we have included Figure 21 that illustrates the contrast between two typing sessions that are from the *same* user, with that of two typing sessions from two *different* users. The figure depicts the keyboard array with normalized values of the duration feature. In this example, the similarities and differences in typing patterns are evident to the human eye because the two samples from the same user are very similar. This is not the case in every situation, as some users may type in a similar manner to other users and cause the system to produce false positives. This became more prevalent as the number of users increased.

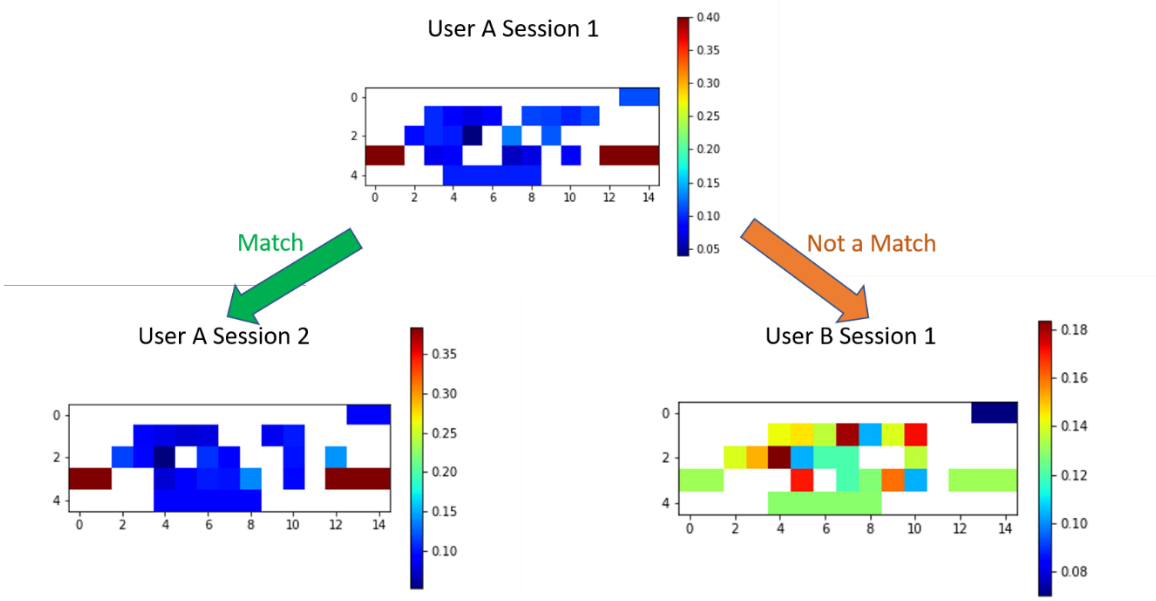


Figure 21. Differences in Typing Between Users

5. Feature Normalization

There were a few other tasks that needed to be completed in order to prepare the data for the neural network. Normalization is a key step that places the data into an appropriate scale for the network. Normalization involves reducing the data to a value between zero and one. In the case of an image, the values can be divided by 256 in order to reduce to a unit value. In our example, some of our durations could be very large and some of the RP latencies could even be negative. We decided to divide by 1000 to move from a millisecond scale to a second scale. After performing this calculation, most values were between the range of zero and one, and increased the accuracy of the results significantly.

A few steps we took to ensure the data was ready for testing did not prove as fruitful as our other attempts. Each user's session data included all the keystrokes averaged together with the values placed in the keyboard array. But what if the user did not press a certain key during that session? The location of that key in the array would contain a 0 for that session. We believed that the presence of unfilled data was impairing our results. Figure 22 describes the two methods we used to get rid of these values. First, we took the average of all the values in the array and replaced all the values where there were no keystrokes with the average. The second technique we used was to interpolate the data based on the nearest values. This method produced an array with no non-zero values. The values were filled in according to the closest value in the array. Surprisingly, neither of these methods were more effective than filling in the missing values with zeros. After testing with all three methods, the tests with zero values were consistently a few percentage points higher in accuracy than the others.

We also needed to split the dataset up into training, validation and testing sets. We completed this in a manner that was similar to the way we split up the dataset for the baseline method. We split the data in a stratified way so each set would have the same number of samples from each user. We used 33% of the data for testing and then used 10% of the training data for validation. A validation dataset is needed for a neural network to evaluate how the model is working as the neural network goes through the process of training.

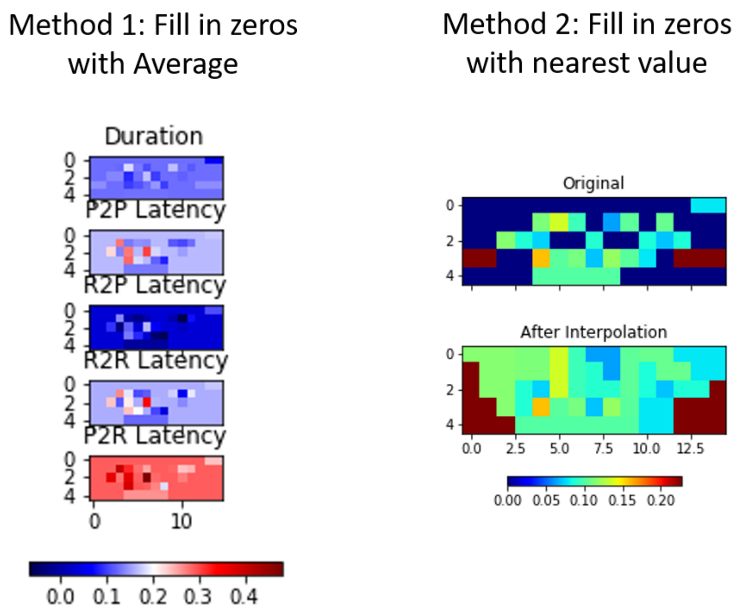


Figure 22. Two Methods of Filling Zero Values

B. NEURAL NETWORK

After preprocessing was completed, the data was in a format that could be accepted by the neural network. We created a multidimensional array with our data divided by the fifteen sessions completed by the users. This data was split into training, testing and validation subsets and the data had been normalized. The next step was to devise the network model and choose the parameters.

1. Types of Neural Networks

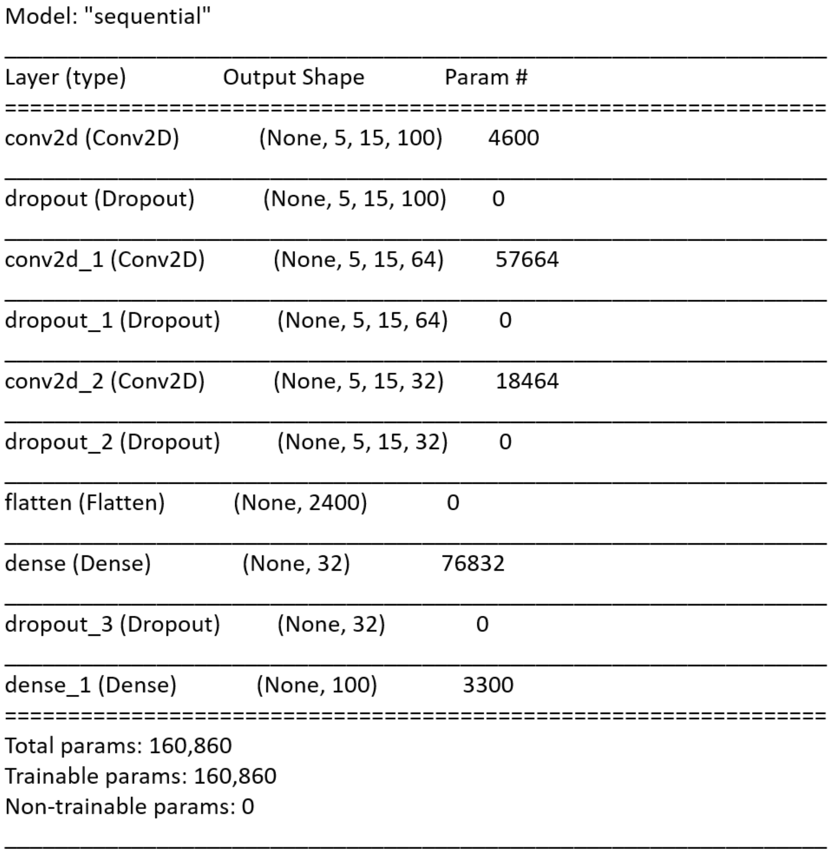
To begin, we created a feed forward neural network with only one fully connected dense layer. This was to ensure we were able to get the network working and obtain a classification accuracy above chance accuracy. The single layer network was created with 32 neurons that were each fully connected to all inputs and all outputs. The feed forward network accepted the normalized input data and attempted to classify each input based on the weights at each neuron. Then it output the predictions and updated the weights according to what it has learned. Next, it repeated the sequence over the entire input data for as many cycles as the user specifies. As expected, the neural network did not perform very well, but we used this network to ensure our data shapes and output were correct.

The next type of network we created was a convolutional neural network. It built upon the feed forward example by adding convolutional layers to the model. The convolutional layers convolve filters, also called kernels, with the input data to output a feature map.

A type of neural network that we did not attempt to build is a recurrent neural network (RNN). According to Khan et al.[27], "Since RNNs process information in a manner that is dependent on the previous computational states, they provide a mechanism to remember previous states." These networks require a different architecture than CNNs. We chose not to use an RNN for this task and instead focused on the CNN as the main emphasis of our research, because we hypothesized that CNNs would be best able to take into account the locations of the keys on a keyboard as a person types.

2. Network Architecture

Before testing the network, we needed to determine the network architecture, which includes the number of layers and shape of each layer. We found it was best to begin small and incrementally grow the neural network by adding layers. We envisioned a CNN would fare the best so we first added a single convolutional layer to the network. This layer had an input shape of 5 x 15 x 5 with an output shape of the same size. In a typical CNN, for example one with an input shape of 256 x 256, the convolutional layers will narrow the input array into a smaller dimension array. We elected to include padding in our convolutional layers in order to keep the input and output arrays the same size. This is because our keyboard array is much smaller than a normal input seen by a CNN and we did not want to shrink our data down to the point that it would not be useful. Next, we added two more convolutional layers, each with less filters than the one before. We then included a dense, fully connected layer and a final dense layer to complete the model. For the last layer, we chose a dense layer with a softmax activation and the number of neurons in the layer equal to the number of classes in our data. Softmax activation is necessary to finish the model because it outputs a probability distribution between 0 and 1. The closer the probability for a class is to 1, the more likely our model believes the data belongs in that class. We also decided we should include dropout layers after the convolutional and dense layers to enable adjusting the dropout rate throughout the network. Once we refined the model, we were ready to perform tests with various combinations of parameters and determine the parameters that give us the greatest performance. Figure 23 shows the final model used for 100 users. The layers and number of parameters created by the neural network are displayed in this figure.



```
Model: "sequential"
_________________________________________________________________
Layer (type)              Output Shape            Param #
=================================================================
conv2d (Conv2D)           (None, 5, 15, 100)      4600
_________________________________________________________________
dropout (Dropout)         (None, 5, 15, 100)      0
_________________________________________________________________
conv2d_1 (Conv2D)         (None, 5, 15, 64)       57664
_________________________________________________________________
dropout_1 (Dropout)       (None, 5, 15, 64)       0
_________________________________________________________________
conv2d_2 (Conv2D)         (None, 5, 15, 32)       18464
_________________________________________________________________
dropout_2 (Dropout)       (None, 5, 15, 32)       0
_________________________________________________________________
flatten (Flatten)         (None, 2400)            0
_________________________________________________________________
dense (Dense)             (None, 32)              76832
_________________________________________________________________
dropout_3 (Dropout)       (None, 32)              0
_________________________________________________________________
dense_1 (Dense)           (None, 100)             3300
=================================================================
Total params: 160,860
Trainable params: 160,860
Non-trainable params: 0
_________________________________________________________________
```

Figure 23. CNN Model for 100 Users

3. Hyperparameter Tuning

The number of parameters that can be changed in a neural network is overwhelming. There is an extremely large number of combinations that can be chosen to tune the network. The choices include bounded factors, such as the type of activation or the name of the optimizer, as well as limitless selections, as in the number of filters for each layer or the number of epochs (cycles) to continue training. The method we used was to start with a small number of users so we could obtain results quickly and tune parameters at a faster pace. We chose initial parameters and then altered them one at a time to see what that did to the accuracy of the model. One technique that proved especially useful was to create an accuracy and a loss graph that we could examine after each run of the network. This model helped us decide if the model was overfitting, underfitting, learning too fast or too slow, or needed to be trained for more or less epochs. Figure 24 shows an example of the network overfitting, and Figure 25 is an example of the network needing to be trained longer.

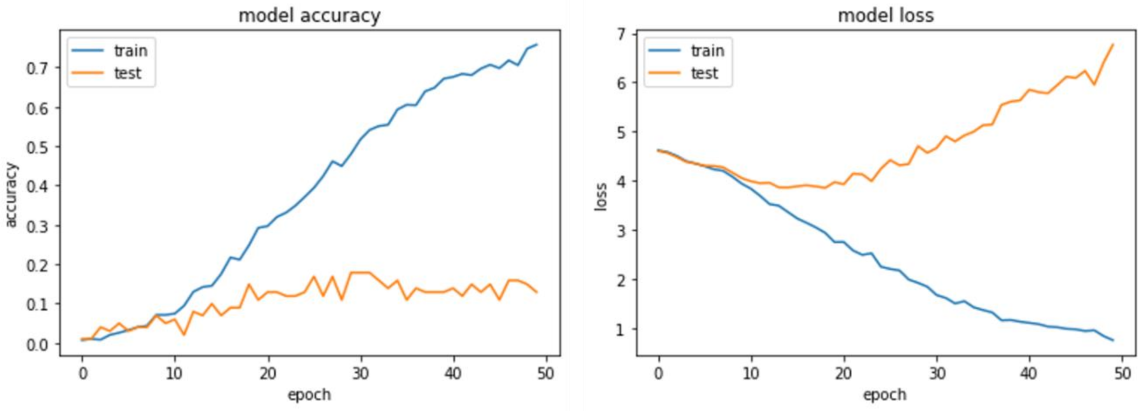


Figure 24. Accuracy and Loss Graphs Depicting Overfitting

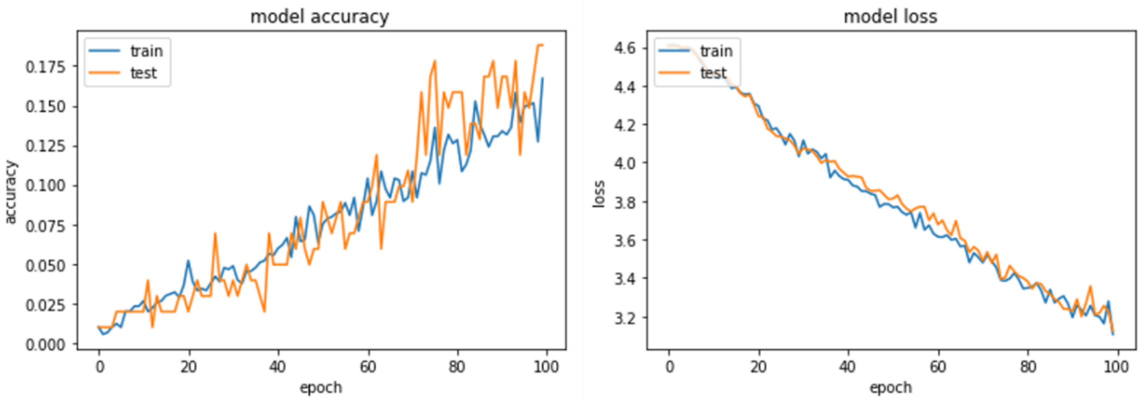


Figure 25. Accuracy and Loss Graphs Depicting a Situation to Keep Training for More Epochs

While we adjusted many parameters such as the activation type, kernel initializer, convolutional filter size, number of filters in each layer, batch size and dropout rate, a few of the parameters proved to be much more effective to tune. The number of neurons in the final dense layer was determined to be the parameter that carried the most weight in the model. This parameter severely affected the speed of the network as well as the accuracy produced. The other most consequential parameter was the number of epochs to train. The number of epochs had a substantial impact on how long the network took to train as well as the accuracy at completion. We also determined that these two factors needed to be tuned as the number of users increased. During testing of the different number of users, we would need to tune the number of filters and the number of epochs in order to obtain the best results. The final number of filters and epochs in order to get the top results is displayed in Table 6.

Table 6. Number of Filters and Epochs Used to Train the Neural Network

Users	Neurons in Final Dense Layer	Epochs
100	32	450
500	128	400
1,000	128	400
2,000	128	400
3,000	128	400
4,000	128	400
5,000	128	400
10,000	600	200
15,000	1024	100
20,000	1024	100
50,000	2048	90
100,000	4096	40

C. RESULTS

This section describes the results achieved by using the neural network technique of classification. The results will be compared with findings from the baseline section.

1. Accuracy

Using our method to train a neural network using keystroke data in this form proved very successful. The results represented a consistent increase from the previous baseline method. We were able to calculate top 1 accuracy results up to 100,000 users. Top 10, 100 and 1000 accuracy results could not be determined higher than 50,000 users. The reason we were not able to obtain results above these numbers is due to not having enough memory on our HPC systems.

The top 1 accuracy for 100,000 users more than doubled with our neural network method. The accuracy increased from 1.06% to 2.5%. Across the board, we observed doubling of our initial results. Figure 26 shows the improvement of the top 1, 10, 100, and 1000 accuracy results from the previous method. The darker lines correspond to the neural network method and are significantly higher than the baseline method.

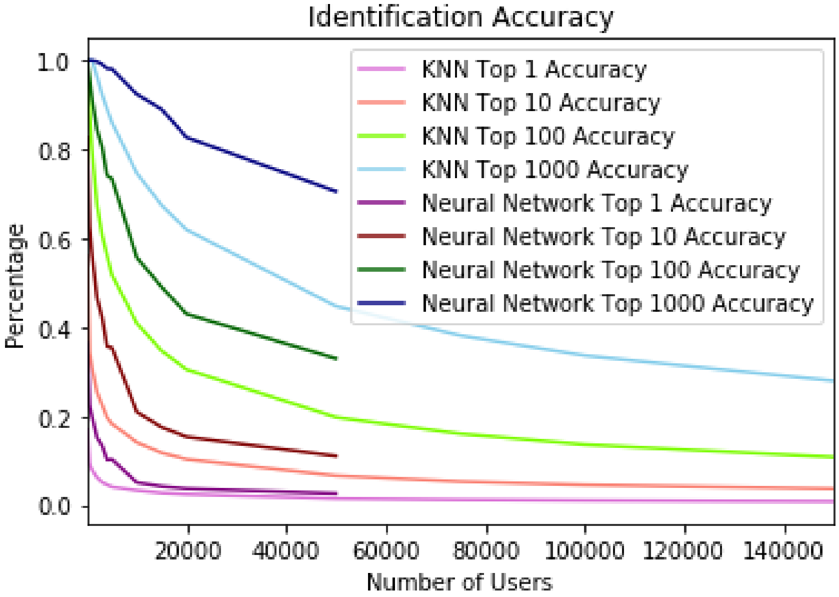


Figure 26. Identification Accuracy for KNN and NN Methods

Another way to visualize the increase in accuracy with the neural network method is to examine the top 1, 10, 100 and 1000 accuracies for one population size at a time. Figures 27 and 28 display these accuracies at 2000 and 50000 users. A few interesting observations can be made from these two graphs. The increase in accuracy at all levels is evident across both of the figures. Also, as the number of users increases, the accuracy increase becomes more pronounced at the right side of the graph, which indicates a larger N. The top 100 and 1000 accuracies become more separated from the baseline method than the top 1 and 10 accuracies. This indicates that while the neural network method increased performance across the board, its most significant increase was in the ability to narrow down the field of potential users rather than pinpoint the exact user.

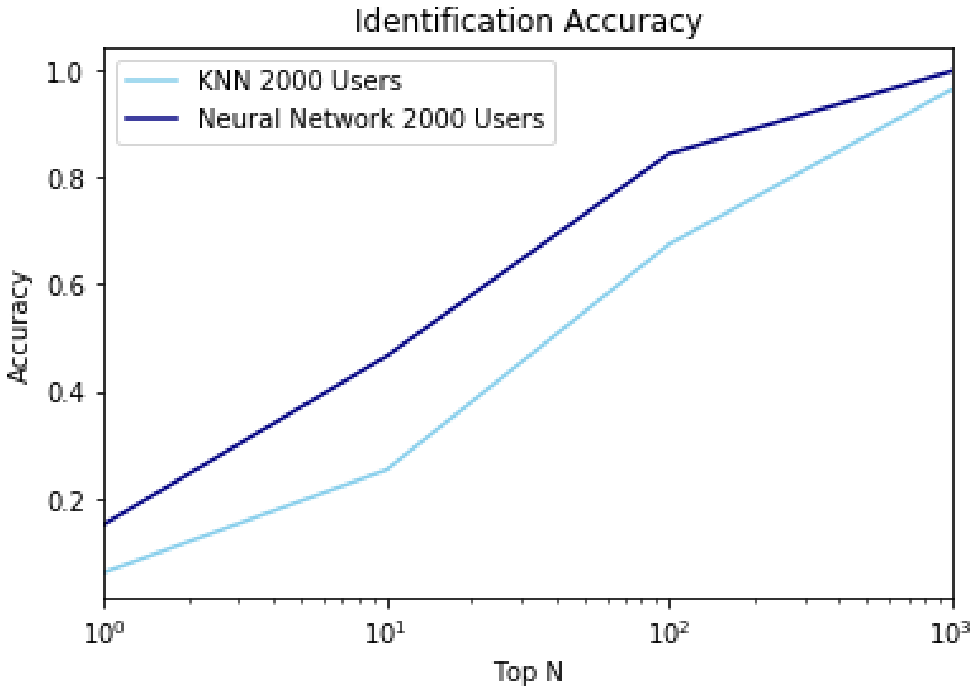


Figure 27. Top N Accuracies at 2000 Users

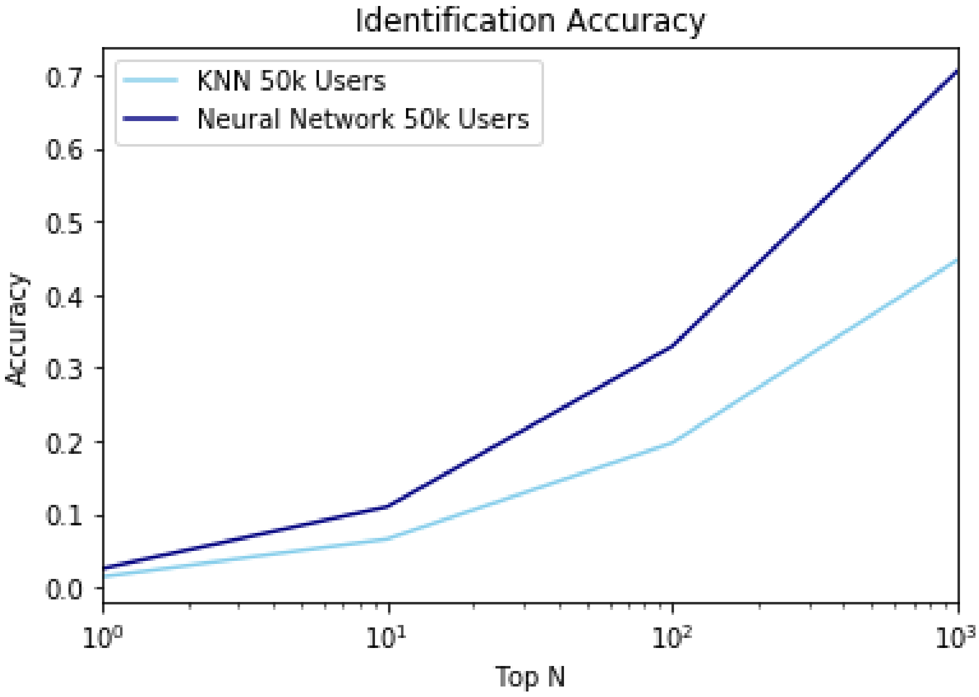


Figure 28. Top N Accuracies at 50000 Users

2. Profiling

Promising results were also uncovered while attempting to profile users with the neural network method. There were a few noteworthy differences in the neural network model from the baseline method of profiling. The number of classes stayed the same for each demographic category. For example, gender had two classes (male and female), language had two classes (English and non-English) and age had 4 classes (0-19, 19-29, 29-39 and 39+). The major difference was in the format of the data used to profile the users. In the baseline method, each user was represented by a row of data that contained the demographic statistic as well as the calculated features. For the neural network method, we used the same method of inputting data to the neural network as we did to determine the identification accuracy. However, the data needed to be split into training and testing datasets in a different manner. All 15 sessions of a user's typing needed to stay together in either the training or testing set. This meant the stratified method of splitting up the users would not work in this case. Instead, we split the data while keeping all sessions of a user together, called a "group" cross validation. The network was then trained using a smaller CNN model. The profiling results represented a substantial increase in the ability to determine the demographic information of a user. Figure 29 shows the accuracy percentages of the neural network method as compared to the results from the baseline method. Of the categories that we were able to obtain results for, the percentages jumped considerably to the upper 80s and low 90s. One noteworthy difference is that the baseline method included all users in this calculation while the neural network method was only able to obtain results using a smaller subset of the total number of users. These accuracies are approximations calculated with 10,000 users, which affects the results and the ability to accurately compare with previous methods.

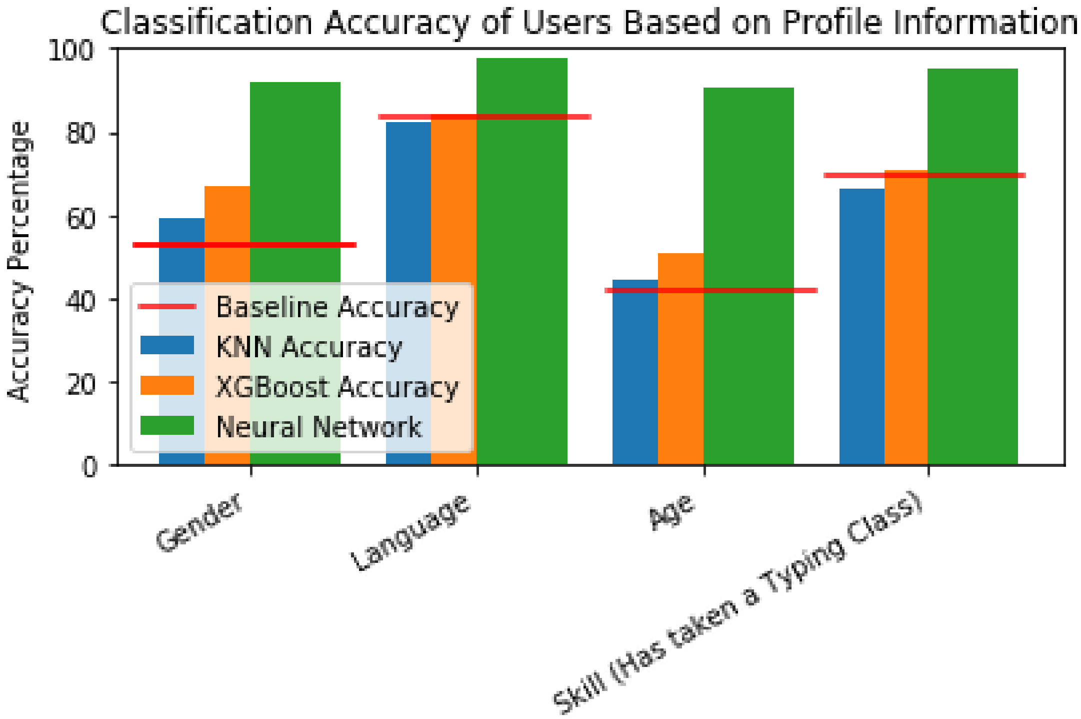


Figure 29. Profiling Accuracies Using the Neural Network

THIS PAGE INTENTIONALLY LEFT BLANK

VI. CONCLUSION

This thesis explored user identification via keystroke dynamics at an Internet scale. Two methods were evaluated: first, using conventional techniques to establish an identification accuracy baseline, and then using deep learning techniques in an attempt to maximize classification accuracy.

Traditional methods of identifying users based on the way he or she types work well with small groups but do not scale well past 100 to 1000 users. This is relevant for a small group or corporate network but does not extend to identification on the scale of a much larger network or the Internet. The higher the number of classes in a dataset, the harder it is to perform identification because more potential matches are available in the pool of users. Identification on this scale could be especially useful in the realm of cyber security. The ability to identify, profile or narrow down the possible pool of users is extremely valuable in defensive applications. Envision a scenario in which an offensive threat is detected and his or her keystrokes are collected. Comparing these keystrokes to a database of potential adversaries in order to determine who the user is or who the user is not would be an incredible accomplishment. Even being able to determine which country this person is from or the gender of the attacker would be crucial information.

A. DISCUSSION

The accuracy of even the simplest techniques was a significant increase over random chance. The KNN increased the accuracy of identifying the correct user by a factor of 1285 and the neural network improved upon these results even further. The first research question asks to what extent can identification be performed at this scale. Accuracies in this research did not reach levels considered acceptable for identification applications, but many lessons were learned about the process. Users can be confidently narrowed down based on the top N accuracies, and a new approach for spatially representing keystroke dynamics features was described. It is promising that even at this scale, results could significantly improve based on the methods that were used.

The first method used handcrafted features created by calculations made on the press and release times of each keystroke. On every level, this technique paled in comparison to the method that utilized deep learning through a neural network. The neural network learned what it could about the duration and latencies of each key press and then adjusted the weights of the network to make predictions. Top one accuracies more than doubled no matter the number of users present. Top 10, 100, and 1000 accuracies increased significantly as well, making the task of narrowing down the field of users much easier.

A novel method of converting the keystroke data to a multidimensional array and using it as input to a neural network was also introduced. Neural networks are routinely used to identify pictures or videos by using the pixels in the form of multidimensional arrays. Neural networks have also been used to accurately identify users based on his or her keystroke dynamics. A technique that has not been previously used is to combine the two by converting the way a user types into a multidimensional array and training with a neural network. Much like a sequence of frames in a video, we developed a way to convert keystroke press and releases over a typing session into a series of 2D arrays. This technique allowed us to make allowances for the locations of the keys in relation to each other and better integrate the distances between keys into our results.

Another result of this research is that much was learned during the various attempts to identify users based on demographic characteristics. Profiling that used certain characteristics was much more successful than profiling that used other characteristics. Gender, country, age and keyboard type were factors that were able to be used to profile a user. Language and typing skill were not as effective at this task.

B. INTERNET PRIVACY IMPLICATIONS

This thesis raises significant Internet privacy concerns. While none of the methods attempted in this thesis can correctly identify a user with high confidence, the results demonstrate that identification, to a certain extent, could be possible. With more research, it is possible that the identity of users on an Internet scale could be significantly narrowed down and even pinpointed with similar methods as presented in this thesis. Identification based on typing data has many more positives than other available methods such as browser fingerprinting. If the neural network method presented in this thesis were refined, it could present a viable alternative to other methods of identification that are not as universal or robust over time, devices and users.

C. FUTURE WORK

This thesis was conducted in the timeframe of less than a year. With more time, additional efforts would likely improve upon the results presented within this research.

One approach would be to use a Recurrent Neural Network that takes as input the sequence of keystroke "frames" instead of the features averaged over the entire session. An RNN could possibly perform better than the CNN we used because the feature averages throw away potentially important information in the keystroke sequence. Additional hyperparameter tuning could also be made to the CNN model we created. Increasing the number of layers, neurons and/or epochs during training, coupled with faster equipment to facilitate testing new combinations could significantly alter the results for the better.

Another technique that could possibly improve results is to use a triplet network similar to the structure used by the FaceNet research [28]. The researchers created a successful identification method using a deep CNN that trains with triplets of matching and non-matching pairs. The triplets include 2 matching examples and a non-matching example, and the model attempts to separate the matching from the non-matching pairs. This approach has had some success in face recognition and could potentially enable scaling keystroke biometrics beyond 100k users.

Using a different dataset with greater diversity could also be helpful. Sessions of a greater length than one sentence would enable the neural network to pick up on patterns and characteristics of a user's typing much easier and potentially lead to higher identification accuracy. Greater diversity in the population would also help to eliminate some of the inherent bias that is present in the dataset, namely users who are interested in typing fast.

The promising results of using a neural network to attempt to profile a user should be further investigated as well. With more time and computing power, these results could be expanded upon. Accuracies of 90 percent and above for certain demographic characteristics on a dataset such as this are very hopeful.

The method of turning keystroke data into a two-dimensional array that resembles the frames of a video could potentially be adapted for other uses. Similar applications include tracking the mouse movement of users in order to perform identification. Another application is collecting the finger movements of a user who is using a touchscreen device. Speed, location, pressure and size could all be factors that are represented as channels and included in the spatial array.

LIST OF REFERENCES

[1] Defense Advanced Research Projects Agency, “Active Authentication.” Accessed Oct 21, 2019. [Online]. Available: https://www.darpa.mil/program/active-authentication.

[2] A. K. Jain, A. Ross, and S. Prabhakar, “An introduction to biometric recognition,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 14, no. 1, pp. 4–20, Jan. 2004.

[3] G. Doddington, W. Liggett, A. Martin, M. Przybocki, and D. Reynolds, “Sheep, goats, lambs and wolves: A statistical analysis of speaker performance in the NIST 1998 speaker recognition evaluation,” in *5th International Conference on Spoken Language Processing (ICSLP 98)* Sydney, Australia, 1998.

[4] N. Yager and T. Dunstone, “The biometric menagerie,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 2, pp. 220–230, Feb. 2010.

[5] S. P. Banerjee and D. Woodard, “Biometric authentication and identification using keystroke dynamics: A survey,” *J. Pattern Recognit. Res.*, vol. 7, no. 1, pp. 116–139, 2012.

[6] S. Li, H. Guo, and N. Hopper, “Measuring information leakage in website fingerprinting attacks and defenses,” in *Proceedings of the 2018 ACM SIGSAC Confernce on Computer and Communications Security*, 2018, pp. 1977-1992.

[7] A. Narayanan *et al.*, “On the feasibility of internet-scale author identification,” in *2012 IEEE Symposium on Security and Privacy*, 2012, pp. 300–314.

[8] C. C. Tappert, M. Villani, S.-H. Cha, “Keystroke biometric identification and authentication on long-text input,” in *Behavioral Biometrics for Human identification*, Hershey, PA, USA: Medical Information Science Reference, 2009, pp. 342-367.

[9] M. Villani, C. Tappert, Giang Ngo, J. Simone, H. St. Fort, and Sung-Hyuk Cha, “Keystroke biometric recognition studies on long-text input under ideal and application-oriented conditions,” in *2006 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW’06)*, 2006, pp. 39–39.

[10] V. Dhakal, A. Feit, P. O. Kristensson, and A. Oulasvirta, “Observations on typing from 136 million keystrokes,” *in Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, 2018, pp. 1–12.

[11] A. Alsultan and K. Warwick, “Keystroke dynamics authentication: A survey of free-text methods,” *Int. J. Comput. Sci. Issues IJCSI*, vol. 10, no. 4, pp. 1–10, 2013.

[12] P. D. Varcholik, J. J. Laviola, and C. E. Hughes, “Establishing a baseline for text entry for a multi-touch virtual keyboard,” *Int. J. Hum. - Comput. Stud.*, vol. 70, no. 10, pp. 657–672, 2012.

[13] A. Feit, D. Weir, and A. Oulasvirta, “How we type: Movement strategies and performance in everyday typing,” in *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, 2016, pp. 4262–4273.

[14] P. Khanna and M. Sasikumar, “Recognizing emotions from keyboard stroke pattern,” *Int. J. Comput. Appl. N. Y.*, vol. 11, no. 9, Dec 2010.

[15] T. Arroyo-Gallego *et al.*, “Detecting motor impairment in early Parkinson’s disease via natural typing interaction with keyboards: Validation of the neuroQWERTY approach in an uncontrolled at-home setting,” *J. Med. Internet Res.*, vol. 20, no. 3, p. 89, Mar. 2018.

[16] A. Narayanan and V. Shmatikov, “Robust de-anonymization of large sparse datasets,” in *2008 IEEE Symposium on Security and Privacy (sp 2008)*, 2008, pp. 111–125.

[17] P. Eckersley, “How unique is your web browser?” in *Lecture Notes in Computer Science, Privacy Enhancing Technologies: 10th International Symposium*, M. J. Atallah, N. J. Hopper, Eds. Berlin, Germany: Springer Berlin Heidelberg, 2010, pp. 1-18.

[18] A. Vastel, P. Laperdrix, W. Rudametkin, and R. Rouvoy, “FP-STALKER: Tracking browser fingerprint evolutions,” in *2018 IEEE Symposium on Security and Privacy*, 2018, pp. 728–741.

[19] T. Kohno, A. Broido, and K. C. Claffy, “Remote physical device fingerprinting,” *IEEE Trans. Dependable Secure Comput.*, vol. 2, no. 2, pp. 93–108, 2005.

[20] Conti Mauro *et al.*, “Selfrando: Securing the Tor browser against de-anonymization exploits,” *Proc. Priv. Enhancing Technol.*, vol. 2016, no. 4, pp. 454–469, 2016.

[21] R. A. Fisher, “The use of multiple measurements in taxonomic problems,” *Ann. Eugen.*, vol. 7, no. 2, pp. 179–188, 1936.

[22] J. Deng, A. C. Berg, K. Li, and L. Fei-Fei, “What does classifying more than 10,000 image categories tell us,” in *Computer Vision – ECCV 2010*, 2010, pp. 71–84.

[23] M. Gupta, S. Bengio, and J. Weston, “Training highly multiclass classifiers,” in *Journal of Machine Learning Research*, vol. 15, pp. 1461–1492, Apr, 2014.

[24] K. S. Killourhy and R. A. Maxion, “Comparing anomaly-detection algorithms for keystroke dynamics,” in *2009 IEEE/IFIP International Conference on Dependable Systems Networks*, 2009, pp. 125–134.

[25] J. V. Monaco, N. Bakelman, S.-H. Cha, and C. C. Tappert, “Recent advances in the development of a long-text-input keystroke biometric authentication system for arbitrary text input,” in *2013 European Intelligence and Security Informatics Conference*, 2013, pp. 60–66.

[26] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” *Commun. ACM*, vol. 60, no. 6, pp. 84–90, Jun. 2017.

[27] S. Khan, H. Rahmani, S. A. A. Shah, and M. Bennamoun, “A guide to convolutional neural networks for computer vision,” *Synth. Lect. Comput. Vis.*, vol. 8, no. 1, pp. 1–207, Feb. 2018.

[28] F. Schroff, D. Kalenichenko, and J. Philbin, “FaceNet: A unified embedding for face recognition and clustering,” in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 815–823.

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California