Spring 2023

Keystroke Dynamics and User Identification

Atharva Sharma
San Jose State University

Keystroke Dynamics and User Identification

A Project

Presented to

The Faculty of the Department of Computer Science

San José State University

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

by

Atharva Sharma

May 2023

The Designated Project Committee Approves the Project Titled

Keystroke Dynamics and User Identification

by

Atharva Sharma

APPROVED FOR THE DEPARTMENT OF COMPUTER SCIENCE

SAN JOSÉ STATE UNIVERSITY

May 2023

Dr. Mark Stamp	Department of Computer Science
Dr. Katerina Potika	Department of Computer Science
Dr. Genya Ishigaki	Department of Computer Science

ABSTRACT

Keystroke Dynamics and User Identification

by Atharva Sharma

We consider the potential of keystroke dynamics for user identification and authentication. We work with a fixed-text dataset, and focus on clustering users based on the difficulty of distinguishing their typing characteristics. After obtaining a confusion matrix, we cluster users into different levels of classification difficulty based on their typing patterns. Our goal is to create meaningful clusters that enable us to apply appropriate authentication methods to specific user clusters, resulting in an optimized balance between security and efficiency. We use a novel feature engineering method that generates image-like features from keystrokes and employ multiclass Convolutional Neural Networks (CNNs) to verify our clustering results.

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my project advisor, Dr. Mark Stamp, for his guidance, support, and patience throughout this research. I would like to thank my committee members, Dr. Katerina Potika and Dr. Genya Ishigaki, for their invaluable inputs to improve the quality of my work. Lastly, I would also like to acknowledge the University at Buffalo for providing the datasets that were used in this research.

TABLE OF CONTENTS

CHAPTER

1 Introduction 1

2 Background 5

2.1 Related Work 6

2.2 Datasets 9

2.2.1 Buffalo Keystroke Dataset 9

2.3 Traditional Machine Learning Algorithms 10

2.3.1 Support Vector Machines 10

2.3.2 Decision Trees 11

2.3.3 Random Forests 11

2.4 Deep Learning Algorithms 12

2.4.1 Convolutional Neural Network 12

2.4.2 Cutouts 12

3 Feature Engineering 14

3.1 Keystroke Features 14

3.2 Keystroke Sequence 16

3.3 Image-Like Features from Keystrokes 16

3.4 Cutout Regularization 17

3.5 Dwell Time and Flight Time 18

4 Architecture 21

4.1 Multiclass Classification for User Identification 21

4.2 Hyperparameter Tuning 22
4.3 Implementation of Multiclass CNN Classifier 23
4.4 Implementation of traditional models on hard to identify users . . 24
5 Experiments and Results 25
5.1 Experiment Strategy 25
5.2 Experiment Metrics 26
5.2.1 Accuracy 26
5.2.2 Confusion Matrix 27
5.3 Results of Multiclass Classification 27
5.3.1 Extracting Confusion Matrix and Creating Clusters 29
5.4 Additional Experiments on Hard-to-Identify Users 33
5.4.1 SVM with minimal features 34
5.4.2 Traditional models with flattened keystroke images 34
5.4.3 Hyperparameter tuning for Random Forest 34
6 Conclusion 39
LIST OF REFERENCES 42
APPENDIX
Random forest on complete dataset 45
A.1 Experiment results 45

CHAPTER 1

Introduction

Authentication and intrusion detection are crucial aspects of online security, necessitating secure and efficient user identification and verification. Conventional authentication methods, such as passwords, have limitations, and biometric systems may require additional hardware or be unsuitable for specific user groups. This leads to an unfair and exclusionary system. Recent research highlights the need for accessible and inclusive authentication systems for all users, including elderly [1, 2] and disabled individuals [3]. Keystroke dynamics have emerged as a promising alternative for user authentication and identification. By analyzing keystroke patterns, a user can be identified based on their unique typing style, regardless of age or physical ability. Furthermore, keystroke dynamics can detect intruders who have gained unauthorized access to a system, making it a potent security tool.

Compared to traditional authentication methods like passwords, keystroke dynamics offer several benefits. Firstly, keystroke dynamics are challenging to break since each person has a unique typing pattern that is difficult to replicate or guess. In contrast, passwords can be compromised through data breaches or guessed through trial-and-error. Secondly, keystroke dynamics provide a more robust and reliable authentication form. Even if unauthorized users obtain correct login credentials, they may still be detected and denied access if their typing pattern does not match the authorized user. Additionally, keystroke dynamics offer continuous authentication, enabling ongoing user identity verification throughout the session, adding an extra layer of security. Overall, keystroke dynamics present a more secure and efficient authentication form, making them an attractive alternative to traditional methods.

Technologies like keystroke dynamics have led to increased interest in behavioral biometrics within the cybersecurity field. Banks and other organizations invest in

companies such as BioCatch, specializing in behavioral biometrics to prevent fraud and protect user accounts. Major investments from banks like HSBC, Barclays, Citi Ventures, and American Express [4] underscore the significant potential of keystroke dynamics and other behavioral biometrics in enhancing security and preventing cyber attacks.

For our research, we will use the Buffalo free-text keystroke dataset to study keystroke dynamics. Both free-text and fixed-text datasets have their advantages and drawbacks. Free-text datasets, collected while users type naturally without constraints, offer a more realistic representation of user behavior, providing better ecological validity and user acceptability [5]. On the other hand, fixed-text datasets, collected under controlled conditions where participants type specific phrases or sentences, enable more controlled experiments and easier result comparison by eliminating variations in text input [6]. Fixed-text datasets also enable the study of particular linguistic and typing features, useful for investigating the effects of variables such as typing posture on keystroke dynamics. However, considering the practicality and user experience aspects, we have chosen to work with free-text data in this study.

In this research, we explore various machine learning and deep learning techniques for free-text classification using the Buffalo free-text keystroke dataset. Inspired by successful outcomes in prior studies, we use a feature engineering approach that transforms features into a multi-channel image-like transition matrix. Within this matrix, rows and columns denote keyboard keys, while the depth signifies distinct feature categories.

We then conduct multi-class classification on a dataset of 148 users, employing a Convolutional Neural Network (CNN) model trained on the image-like features with cutout regularization. To assess the effect of keystroke sequence lengths on our model, we experiment with multiple sequence lengths. The CNN model's evaluation is based

on open free-text keystroke datasets, with the outcomes used to further cluster users. After generating the confusion matrix from the CNN, we aim to categorize users into three clusters, each representing a different level of user authentication difficulty. The main objective of our study is to determine these clusters and examine their relevance to the challenges encountered in the user authentication process.

By performing the above mentioned experiments, this paper makes the following contributions to the community.

- Applying machine learning techniques on the free-text keystroke dataset provided by Buffalo.
- Employing feature engineering on keystroke data to generate an image-like transition matrix for multiclass classifier. This method of using Image-like transition matrix has produced impressive results with binary classifier [7].
- Evaluating a CNN model with cutout regularization for multi-class user classification.
- Examining the impact of varying keystroke sequence lengths on multi-class CNN model performance.
- Analyzing three confusion matrix-derived clusters representing user authentication difficulty levels.
- Developing a strategy for hard to identify clusters of users, with tailored processing and classification methods such as Support Vector Machine (SVM), to improve user identification accuracy.

In this chapter, we looked at the fundamental concepts of keystroke dynamics-based authentication and outlined our approach to addressing the problem. The remaining paper is organized as follows: In Chapter 2, we delve into background topics such as the learning techniques utilized and the datasets leveraged in our study. This chapter also includes a review of prior research. Chapter 3 details the features

we employ and, specifically, discusses our feature engineering strategy for preparing input data for our classification models. Following this, in Chapter 4, we elaborate on the model architectures considered in this paper and discuss the hyperparameter tuning process. Chapter 5 encompasses our experiments and an analysis of the results. Lastly, Chapter 6 offers a conclusion and suggests avenues for future research.

CHAPTER 2

Background

Authentication is a fundamental aspect of security systems [8], and keystroke dynamics has emerged as a promising method for verifying user identity. Unlike traditional authentication methods, keystroke dynamics can detect intruders even after they have gained access to the system, making it a valuable tool for preventing security breaches. However, the effectiveness of keystroke dynamics-based systems depends on the ability to accurately classify users based on the difficulty of authentication.

In this context, clustering users based on their authentication difficulty can facilitate efficient allocation of resources and time. Users who are easy to authenticate can be handled with minimal authentication, while those who are difficult to authenticate may require additional analysis and preemptive security measures. SVM or k-Nearest Neighbors (k-NN) algorithms can be used for this purpose [9]. For instance, if a user is identified as hard to classify, the system can proactively deploy additional security measures, such as requiring a second factor of authentication.

One of the key benefits of this approach is that it can lead to improved resource usage and more efficient time management. By classifying users based on the level of authentication difficulty, the system can optimize resource allocation by only dedicating significant resources to users who are difficult to authenticate. This approach can help reduce the burden on the system and improve its overall efficiency.

Additionally, the use of free text datasets for keystroke dynamics-based systems is preferred over fixed text datasets [10]. Free text datasets are more representative of how users type on a regular basis and are not constrained by a pre-determined text input. This results in more accurate and reliable authentication outcomes. Moreover, free text datasets are more adaptable to a variety of use cases, including those that require longer input sequences for analysis.

Another advantage of keystroke dynamics-based systems is that they can benefit users of all ages and those with disabilities, as everyone types to use a system [11]. Therefore, this approach can provide a more inclusive and accessible authentication method that does not discriminate based on age or physical ability.

In summary, keystroke dynamics-based systems can offer a reliable and effective means of authentication, particularly when users are clustered based on the difficulty of authentication. This approach can lead to more efficient resource and time management, while also ensuring accessibility for all users. The use of free text datasets further enhances the accuracy and adaptability of keystroke dynamics-based systems.

2.1 Related Work

Keystroke dynamics is a behavioral biometric that has been extensively studied for user authentication and identification. In 1980, Gaines et al. [12] analyzed digraph latencies to examine the distinctiveness of typing patterns and found that specific digraphs could distinguish right-handed touch typists from one another with 92% accuracy in a limited sample. Following this, in 1990, Bleha et al. [13] proposed a real time pattern recognition based approach to classify users. The online verification system they developed had an error rate of 8.1% in rejecting valid users and 2.8% in accepting invalid users. Despite the limitations, such as allowing invalid users to watch and practice imitating valid users, their work laid the foundation for subsequent research in the field.

Machine learning techniques have been widely applied in the field of keystroke dynamics. Traditional classification algorithms, such as k-Nearest Neighbors (k-NN) and SVM, have demonstrated promising results in user authentication tasks. However, these methods often rely on handcrafted features, which can be less robust and less generalizable to diverse user groups and typing scenarios. The following papers laid

down foundations using novel approaches to generalize these methods to adopt to diverse user bases. Such as the SVM-based method by Giot et al. [14], requiring only five captures for initial enrollment, and the clustering-based method by Gingrich et al. [15], utilizing k-nearest neighbour approach. Resulting in a 66.7% gain in average authentication speed. These approaches offer more robust and generalizable methods with high accuracy and efficiency compared to traditional classification algorithms.

Clustering techniques have been employed in the context of keystroke dynamics to group similar users or typing patterns, and to identify potential outliers. The application of clustering algorithms, such as K-Means [16] and hierarchical clustering [17], has demonstrated promising results in user identification and authentication tasks. When used as a classifier, The K-Means clustering method achieved an accuracy of 96.20%. On the other hand, clustering techniques can also be used as data analysis tool. For example, hierarchical clustering was used [17] to evaluate the effect of hold times on the homogeneity of valid user timing vectors. The use of hierarchical clustering helped to establish the relative homogeneity of valid user timing vectors and improve the accuracy of the subsequent experiment.

Clustering can also be applied to keystroke dynamics for the purpose of detecting account sharing. Hwang et al. [18], propose that each user's keystroke patterns form a distinct cluster in Euclidean space and the number of sharers can be estimated by the number of clusters. The optimal number of clusters is estimated using a Bayesian model-selection framework. The results show a 2% false alarm rate, a 2% miss rate, and a 7% total user estimation error [18]. This work highlights the potential of clustering in detecting account sharing using keystroke dynamics.

Clustering methods such as Expectation Conditional Maximization (ECM) have also been combined with other approaches like Extreme Learning Machine (ELM) to improve accuracy and stability. ELM is a fast Single hidden layer feedforward network

with good generalization performance, and Sriram et al. [19] used the clustering-based semi-supervised approach of ECM-ELM to achieve stable accuracy of 87% with the CMU Keystroke Dataset, while ELM only achieved an unstable average accuracy of 90%.

Deep learning techniques for analyzing keystroke dynamics have shown promise in recent studies, with Convolutional Neural Networks (CNN) being employed to achieve notable results in user identification and authentication tasks. A novel approach by Liu et al. [20] involves converting keystroke data into personalized images, which allows for the mining of spatial information and results in an accuracy of 96.8% and a false acceptance rate (FAR) of 0.04%. In contrast, Piugie et al. [21] concentrate on using deep learning for passphrase user authentication, surpassing the performance of state-of-the-art methods in terms of Equal Error Rate (EER) score.

As researchers explore various deep learning model architectures for keystroke dynamics authentication, studies such as [22] and [23] investigate the application of recurrent neural networks. The CNN with Gated Recurrent Unit model is proposed by Lu et al. [22], while Mhenni et al. [23] examine the use of Long Short-Term Memory (LSTM) and Bidirectional Long Short-Term Memory (BLSTM) architectures. Both papers illustrate the potential of deep learning models in this domain. Mhenni et al. [23] show that BLSTM outperforms LSTM, achieving an accuracy of 86% and 71% for the GREYC-2009 and WEBGREYC databases, respectively, compared to LSTM's accuracy of 68% and 53% for the same databases. The issue of long-term dependence in RNNs is effectively addressed by the use of GRU in Lu et al. [22], resulting in a successful authentication model.

Recently, novel feature engineering approaches have emerged that transform keystroke data into image-like structures [7, 24]. These image-like representations can leverage the powerful capabilities of CNNs, which are known for their success in

image classification tasks. In this context, the work of Jianwei et al. [7] is particularly relevant, as it introduces a unique Keystroke Dynamic Image (KDI) that leads to improved results compared to standard features. We are utilising the same KDI image approach suggested in [7] for our multiclass classification as well.

In summary, the related work in the field of keystroke dynamics spans a wide range of techniques and methodologies, including traditional machine learning, deep learning, feature engineering, threshold-based techniques, and clustering ensembles. Building upon this rich body of research, the current study aims to advance the state of the art by proposing a novel two-level approach for keystroke-based user identification, leveraging deep learning models, as well as the integration of multiple classifiers for users who are more difficult to authenticate. By tailoring the identification process to the difficulty of distinguishing users, the proposed approach aims to strike a balance between security and efficiency in user authentication tasks.

2.2 Datasets

For our experiments, we are going to use keystrokes collected by SUNY Buffalo, this is a free-text dataset. Now we will discuss this dataset in more detail.

2.2.1 Buffalo Keystroke Dataset

The Buffalo Keystroke Dataset is a collection of free-text keystroke dynamics data obtained from 148 research participants at the University at Buffalo. The participants were asked to complete two typing tasks in a laboratory setting over the course of three separate sessions. The first task involved transcribing Steve Jobs' Commencement Speech, split into three parts, while the second task included free-text questions. To ensure the data's generalizability, there was a 28-day interval between each session.

Out of the 148 participants, 75 completed the typing test with the same keyboard across all three sessions, while the remaining 73 participants used three different

keyboards in each session. The dataset contains the timestamp of key presses (key-down) and key releases (key-up), organized in a tabular format with three columns. The first column indicates the key, the second column denotes whether the event is a key-press or key-release, and the third column records the timestamp of the event. The dataset includes information about the gender of each participant, and on average, each participant has a total of over $17,000$ keystrokes across three sessions.

While the Buffalo Keystroke Dataset may be relatively basic, its inclusion of timestamp information can still prove to be useful in developing keystroke dynamics-based authentication systems. Moreover, the dataset's free-text nature makes it more adaptable to a variety of use cases, such as those requiring longer input sequences for analysis. Overall, this dataset provides a valuable resource for researchers interested in developing more accurate and efficient authentication systems.

2.3 Traditional Machine Learning Algorithms

Despite the rapid growth of neural networks, traditional machine learning algorithms remain a reliable and effective approach in the field of keystroke dynamics. These algorithms are often based on statistical and mathematical techniques, and have been used for many years in various fields. We will employ the following traditional algorithms on our keystroke dataset.

2.3.1 Support Vector Machines

Support Vector Machine (SVM) [25] is a powerful supervised machine learning technique used to solve real-world problems. Its theoretical foundation is rooted in computational and mathematical principles, making it a highly effective method. SVM is designed to identify a hyperplane in an N-dimensional space that can accurately classify data points into different classes. The algorithm aims to maximize the margin distance between the hyperplane and all the data points. This approach boosts

confidence in the correct classification of future data points, as they are expected to fall within the region determined by the hyperplane. SVM is widely recognized for its practical effectiveness, as it can efficiently handle large and complex datasets. It has been used in various fields, including image classification, text classification, and bioinformatics.

2.3.2 Decision Trees

The Decision Tree (DT) [26] algorithm, a supervised learning method, is appreciated for its simplicity and transparency. Based on statistical principles, it forms a tree-like model of decisions, with each node, branch, and leaf representing a feature, rule, and outcome, respectively. The DT aims to predict a target variable by partitioning data based on attribute values, enabling efficient classification or regression. Despite handling large, complex datasets, DT provides a clear view of decision-making. The applications of DT range in fields like medical diagnosis, credit risk analysis, and natural language processing.

2.3.3 Random Forests

Random Forests (RF) [27] is a well-known ensemble learning technique widely used in supervised machine learning. It is highly regarded for its adaptability and robustness. RF extends the Decision Tree algorithm by generating numerous decision trees during training and making predictions based on the mode or mean prediction of the individual trees. The strength of RF lies in its theoretical foundation, which leverages statistical and computational principles to effectively address overfitting and enhance model generalization. By employing bootstrap sampling and random feature selection, RF excels in handling complex and high-dimensional datasets. Its versatile applications span various domains, including image classification, bioinformatics, and customer segmentation.

2.4 Deep Learning Algorithms

Given that our feature engineering method converts keystroke sequences into an image-like transition matrix, it makes sense to use a Convolutional Neural Network (CNN) for our experiments. CNNs have been shown to perform better when working with image-like data due to their ability to capture spatial dependencies between features. Additionally, we plan to incorporate cutout regularization to further improve the generalization performance of the model.

2.4.1 Convolutional Neural Network

Convolutional Neural Network (CNN) [28] is a specialized type of neural network that utilizes convolution kernels to capture local information from image-like data. Unlike traditional neural networks, CNNs share weights at different locations, resulting in a more efficient model with fewer parameters. Their multi-layer convolutional architecture enables them to extract semantic information at different resolutions in computer vision tasks, making them ideal for image processing. CNNs can analyze images and extract important features, such as edges, shapes, and textures, in a more effective manner. Additionally, the use of convolution kernels in CNNs enables the network to learn spatial features, such as orientation and scale, which is especially useful in image recognition tasks. CNNs have proven to be highly effective in a variety of applications, including object recognition, face recognition, and image classification. They can also be applied to non-image data, such as audio and text, by converting them into a format that can be processed by CNNs.

2.4.2 Cutouts

Dropout [29] is a commonly used technique to prevent overfitting in traditional computer vision tasks, but it may be less effective in convolutional layers due to their shared information and lower parameter count. To overcome this limitation,

cutout [30] is proposed as a Dropout technique that applies to the input space. Cutout generates new images by randomly creating square holes in the original image and filling them with a default value. This data augmentation technique encourages the model to learn more about the image context, resulting in a more robust model that can handle images with occlusions. Cutout also improves the model's ability to generalize and perform well with limited training data, effectively increasing the size of the training set. Overall, Cutout is a versatile and effective technique for image analysis that can enhance the performance of CNNs, especially in the presence of occlusions.

CHAPTER 3

Feature Engineering

We use Buffalo Keystroke Dataset for our analysis, which is a free-text dataset with limited information. Feature engineering will be critical to our analysis, and we will be exploring image-like features that are derived from time-based features existing in the dataset. These features capture the timing information of individual keystrokes and their relationships to other keystrokes, allowing us to build a detailed sequence of keystrokes for each participant. By carefully selecting and engineering these features, we hope to gain valuable insights into how keystroke dynamics can be used as a biometric for user identification and authentication.

3.1 Keystroke Features

Keystroke dynamics datasets generally provide two kinds of features: time-based information and pressure-based information. Both types of features can provide valuable insights into typing behavior, but pressure-based features have become redundant with the widespread adoption of mobile devices that use virtual keyboards. Therefore, our research will focus solely on time-based information. While our dataset does not include pressure-based features, we believe that time-based features alone can still provide useful information for studying typing behavior. Our feature engineering approach will leverage the available time-based features to extract meaningful insights and better understand how keystroke dynamics can be used for user identification and authentication. The data in the Buffalo Keystroke Dataset can be understood by examining the five time-based features that we will be using in our analysis as depicted in Figure 1.

- Duration: The time that the user holds a key in the down position
- Down-down time (DD-time): The time between the press of the first key and the press of the second key

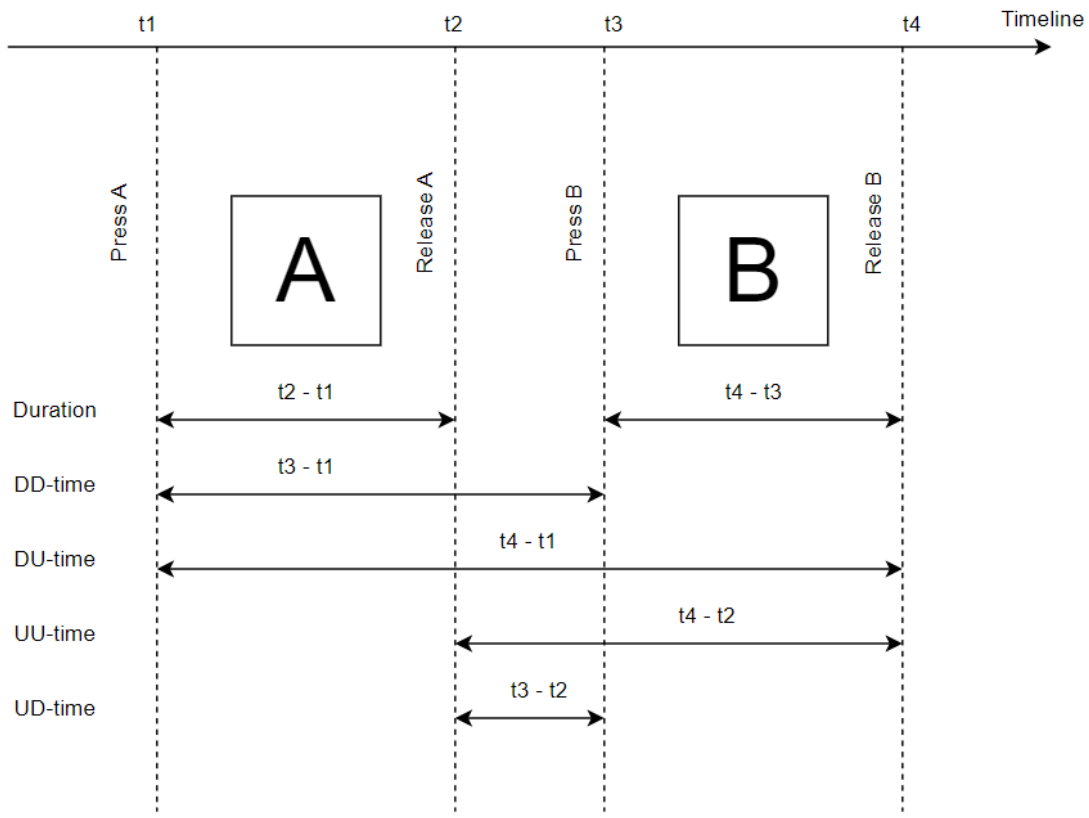

Figure 1: Time Based Features [7]

- Up-down time (UD-time): The time between the release of the first key and the press of the second key
- Up-up time (UU-time): The time between the release of the first key and the release of the second key
- Down-up time (DU-time): The time between the press of the first key and the release of the second key

It's worth noting that for any two consecutive keystroke events, such as A and B, six features can be extracted: duration-A, duration-B, DD-time, UD-time, UU-time, and DU-time. By carefully analyzing these features, we hope to gain insights into the unique patterns of typing behavior exhibited by individual users and how these

patterns can be used for user identification and authentication.

3.2 Keystroke Sequence

A keystroke sequence refers to the entire set of keystrokes entered by a user, which can be thousands of characters long. To better analyze these sequences, they are often divided into smaller sub-sequences. Each sub-sequence can be viewed as an independent keystroke sequence from the same user.

In our research, we will experiment with different lengths of keystroke sub-sequences to optimize our analysis. A longer keystroke sequence can provide more information, but it can also be more resource-intensive to process. Shorter sub-sequences may not capture enough information and lead to decreased accuracy. Therefore, we will carefully select the length of keystroke sub-sequences to ensure that they are optimized for accuracy while being mindful of resource usage. Additionally, the length of the sub-sequences will impact the creation of image-like features that we will be using in our analysis. By selecting the optimal length of keystroke sub-sequences, we aim to improve the accuracy of our results.

3.3 Image-Like Features from Keystrokes

In the previous section, we described dividing the entire keystroke sequence into multiple subsequences. As discussed in Section 3.1, there are six types of timing features. For a subsequence of length N, we can determine $6(N-1)$ features from consecutive pairs of keystrokes. Repeated pairs are averaged and treated as a single pair. For instance, a subsequence of length 50 would yield at most $6 \cdot 49 = 294$ features. We consider each keystroke subsequence as an independent input sequence for the corresponding user. In this section, we propose a new feature engineering structure to better organize these features.

The features UD-time, DD-time, DU-time, and UU-time are determined by

consecutive keystroke events. We organize these four features into a transition matrix with four channels, resembling the structure of RGB images with a depth of three (R, G, and B channels). Each row and column in our four-channel $N \times N$ feature matrix corresponds to a key on the keyboard, with each channel representing one kind of feature. We can organize these into transition matrices as shown in Figure 2.

For example, In the first channel of the matrix, the value at row i and column j refers to the UD-time between any key presses of i followed by j within the current observation window. We add the final feature, duration, as a diagonal matrix to the transition matrix, creating a fifth channel. If a key or key-pair is pressed more than once, we use the average duration for that key or key-pair. In this channel, only diagonal locations have values because the duration feature is relevant for one key at a time. We can use this transition matrix as an image input for machine learning models. To avoid sparsity in the transition matrix, we only consider time-based features for the 42 most common keystrokes. These keys include:

1. The 26 English characters (A-Z)
2. The 10 Arabic numerals (0-9)
3. 6 meta keys (space, back, left-shift, right-shift, tab, and capital)

The shape of the transition matrix is $5 \times 42 \times 42$, with the five channels described above. The proposed feature engineering structure aims to improve the organization and readability of the keystroke features.

3.4 Cutout Regularization

In order to prevent overfitting in our CNN, we make use of cutout regularization. This method introduces artificial occlusions to the image-like features derived from keystroke sequences, encouraging the network to consider the whole image rather than

focusing excessively on specific parts. Cutouts are applied to our unique image-like data structure, as shown in Figure 2, where dark blocks represent the cutouts.

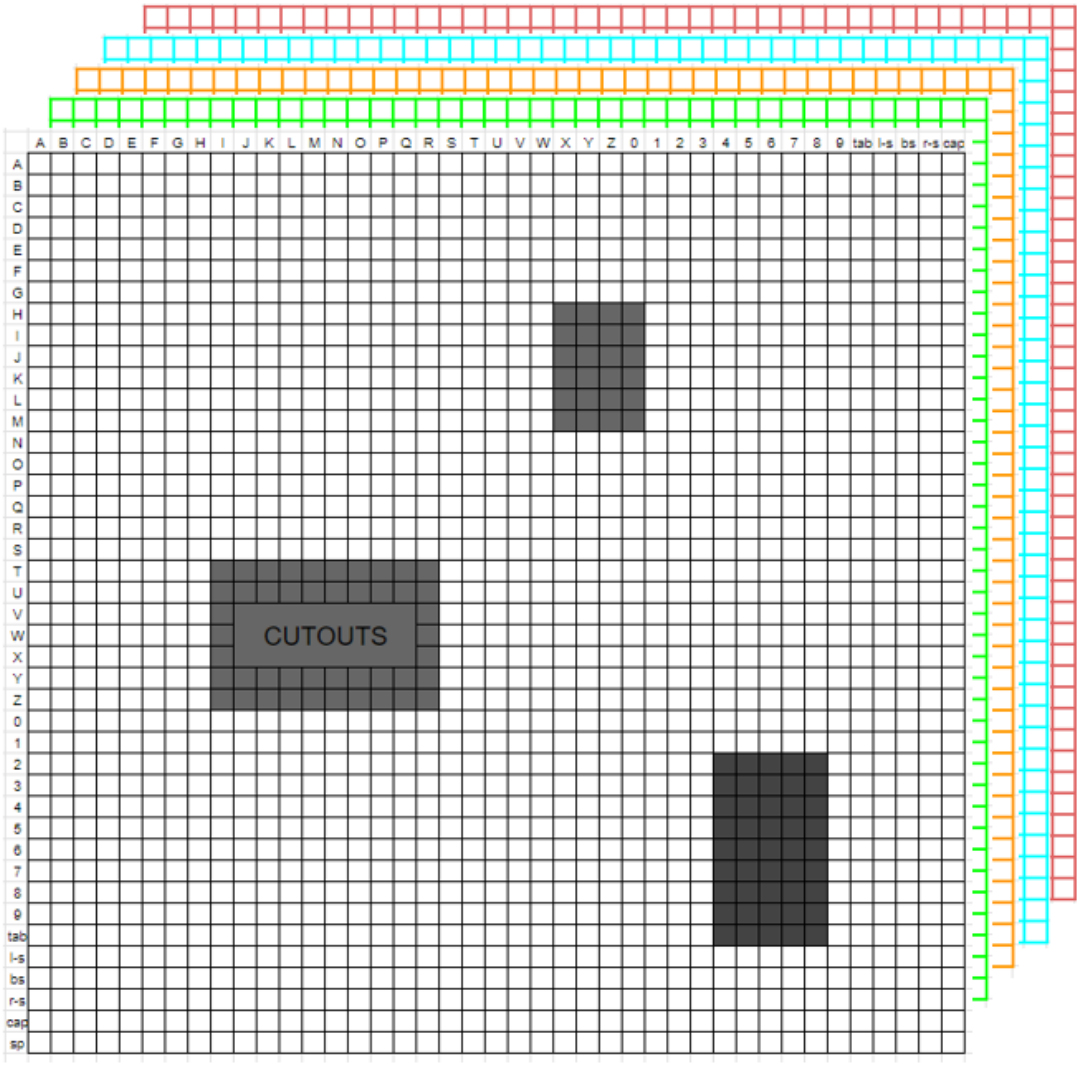

Figure 2: Image-like features from keystrokes.

3.5 Dwell Time and Flight Time

Feature engineering was performed on keystroke data in another way to create a more informative set of features for the multiclass SVM classifier. The initial dataset contained information about the key pressed, the event type (key down or key up),

and the timestamp of the event. For example, the dataset could contain a row with the key 'A', the event type 'KeyDown', and a timestamp '100'. From this raw data, two important features were extracted: dwell time and flight time. Dwell time refers to the duration between a key being pressed down and released, while flight time represents the time elapsed between two consecutive key presses. These features were derived by iterating through the events in the dataset and calculating the time differences accordingly. For instance, if the 'A' key was released at timestamp '120', the dwell time would be 20 (120 - 100). If the next key was pressed down at timestamp '130', the flight time would be 10 (130 - 120) as shown in Figure 3.

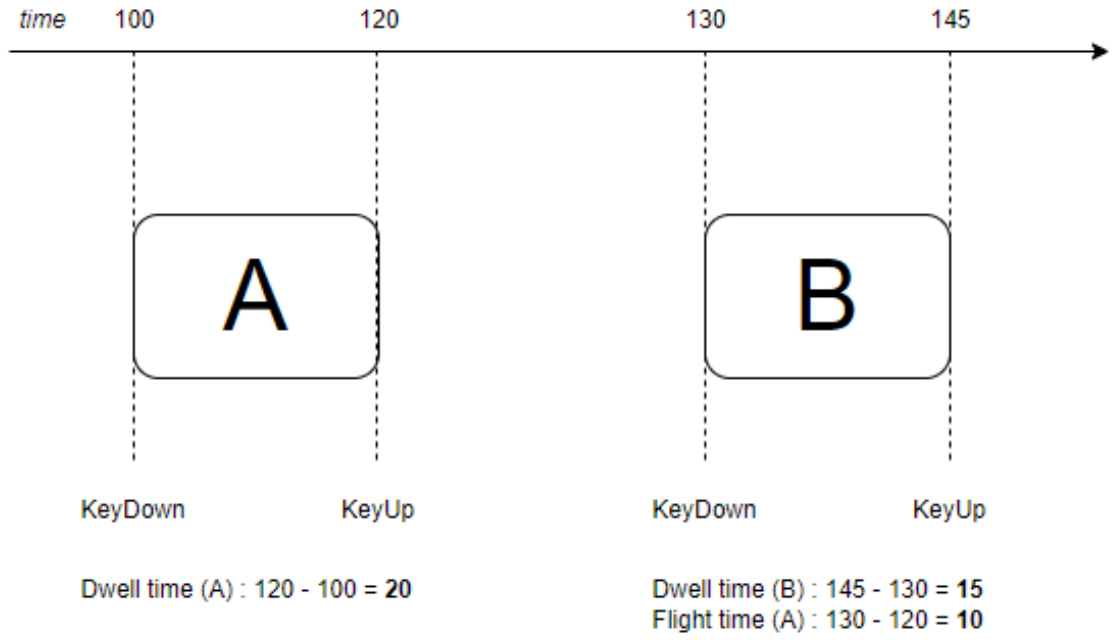

Figure 3: Flight time and dwell time from keystrokes data

After completing the preprocessing, the dataset would include one-hot encoded representations of the key presses, standardized dwell times, and standardized flight times. For example, if there are four unique keys in the dataset (A, B, C, and D), the one-hot encoding for the key 'A' would be [1, 0, 0, 0]. The dwell and flight times would

be standardized using the mean and standard deviation of the respective feature across the dataset. Thus, the final preprocessed dataset would consist of rows containing one-hot encoded key representations, followed by standardized dwell and flight times.

CHAPTER 4

Architecture

In this chapter, we first discuss why we chose multiclass CNN to generate confusion matrix which we can use to separate users in different clusters. These 3 clusters will define how hard it is to authenticate user using keystrokes. These clusters can also be understood as a confidence factor with which we can claim that user is actually what they are posing as. Further, this paper discusses various machine learning models used in experiments. Also how hyperparameters were tuned.

4.1 Multiclass Classification for User Identification

The Buffalo keystroke dataset is a large and publicly available dataset that contains keystroke dynamics collected from 148 users. With a total of $400,000$ keystrokes, it is one of the largest keystroke datasets available to the public. The primary goal of our study is to cluster users based on how easy or difficult it is to authenticate them using image-like features from the Buffalo keystroke dataset. By using a multi-classification model or multiple binary classification models, we can gain valuable insights into user behavior and authentication patterns.

The reason we chose a multi-classification approach is that it can provide more granular information about each user's authentication difficulty. This allows for finer-grained clustering of users based on their authentication difficulty and provides more nuanced insights into user behavior and authentication patterns. Furthermore, multi-classification models have the potential to achieve higher accuracy than binary classification models by better capturing the complex relationships between image-like features and the target variable, which in this case is authentication difficulty.

As the number of users in the dataset grows, training multiple binary classification models becomes increasingly difficult and resource-intensive. In contrast, a multi-classification approach can handle a large number of users in a single model, reducing

the computational burden and improving scalability.

One potential issue with clustering is how to handle the addition of new users to the dataset. Rather than retraining the entire model with all the data points, we can use the existing clustering model to determine which cluster the new user is closest to based on their image-like features. This can be accomplished by computing the distance between the new user's features and the centroids of each cluster.

Once we have determined the closest cluster, we can add the new user to that cluster and update the centroid of the cluster accordingly. This process is known as incremental learning or online learning. It enables the model to adapt to new data without having to retrain the entire dataset, reducing the computational burden.

4.2 Hyperparameter Tuning

To identify the best combination of hyperparameters, such as the initial learning rate, optimizer, number of epochs, and learning rate schedule, we employed a grid search approach. The results showed that the best performance was achieved with the parameters displayed in bold in Table 1. To ensure a fair comparison across different deep learning models, we used the same hyperparameters for all models.

After conducting extensive experiments, we determined that the optimal hyperparameters for our model were 20 epochs and a learning rate of 0.01, using the Adam optimizer for gradient descent. Additionally, we utilized the reduceLROnPlateau callback to dynamically reduce the learning rate if the model was unable to improve during training. To prevent overfitting, we implemented the earlyStopping callback to halt training if the model showed signs of overfitting. Our experimental results indicated that the model tended to overfit after the 20th epoch, which confirmed the results of our grid search.

Table 1: Hyperparameter tuning for multiclass CNN

Parameter	Values
Number of Epochs	10, **20**, 30, 40
Learning Rate	0.1, **0.01**, 0.001, 0.0001
Optimizer	**Adam**, SGD, SGD with Momentum
Learning Schedule Callbacks	StepLR, **reduceLROnPlateau**

4.3 Implementation of Multiclass CNN Classifier

The architecture shown in Figure 4 is a Sequential CNN-based neural network. The input shape of the model is (5, 42, 42), indicating that the input is a 3D array with a depth of 5 and a width and height of 42. The model includes five convolutional layers, each followed by a batch normalization layer and a max pooling layer. The first convolutional layer has 32 filters of size (5,5), while the subsequent convolutional layers have 64, 128, 256, and 256 filters of size (3,3), respectively. All convolutional layers use the Rectified Linear Unit (ReLU) activation function. The max pooling layers have a pool size of (2,2) and a stride of 2, ensuring that the output size of each layer remains the same.

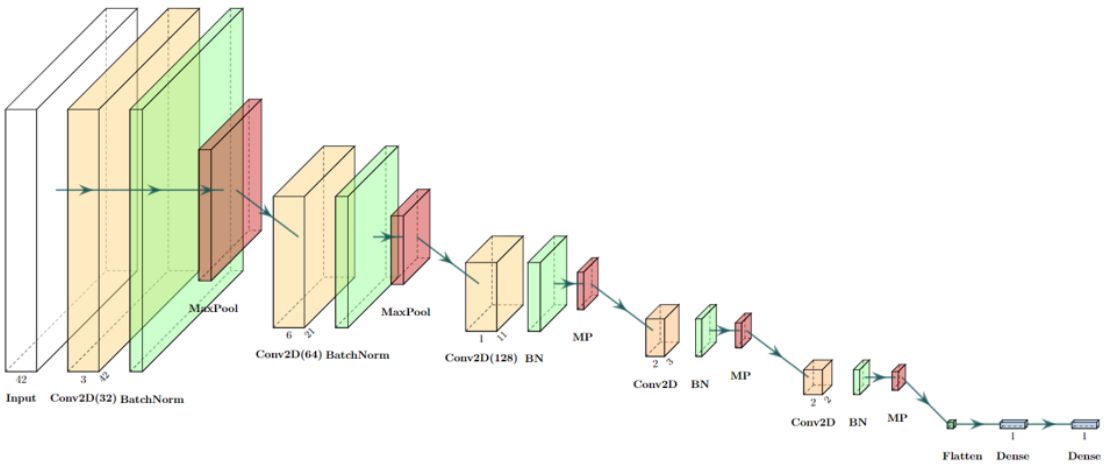

Figure 4: Architecture of sequential CNN used for multiclass classification of users

After the five convolutional layers, the model includes a flatten layer, followed

by two fully connected layers. The first fully connected layer has 128 units with the ReLU activation function. The final output layer has 148 units with the softmax activation function. The model is compiled using categorical crossentropy as the loss function, the adam optimizer, and accuracy as the evaluation metric.

4.4 Implementation of traditional models on hard to identify users

Traditional machine learning models, such as SVM, are implemented on the clusters of users which are hardest to identify, serving as an additional layer of classification. The implementation of the SVM classifier begins with data preprocessing, which includes extracting relevant features, such as dwell time and flight time, and applying one-hot encoding to key events. The continuous features are then standardized to have zero mean and unit variance.

Apart from SVM implementation on minimal features (dwell time and flight time) , we also experimented with traditional machine learning models, including SVM, Decision Trees, Random Forests, and k-nearest neighbors (knn) on flattened keystrokes images of the users who were most challenging to identify. This served as an additional layer of identification for these hard-to-identify users.

Following this extensive experimentation, the best performing model was selected for further optimization. This optimization involved hyperparameter tuning to enhance the predictive accuracy of the model. By integrating the multiclass CNN with other machine learning models and fine-tuning their parameters, we were able to substantially improve our ability to accurately identify the users.

CHAPTER 5

Experiments and Results

In this chapter, We discuss the strategy we followed during our experiments and the evaluation criterion we used to assess the performance of our models. Further, we compare the results between 3 CNN models trained over different keystroke lengths, which are 50, 75 and 100. We then choose the best-performing model and generate three clusters of users based on their keystroke dynamics, categorizing them as easy, moderate, and hard to identify. We then select the users from the hard to identify cluster and apply traditional machine learning models on them to improve the identification accuracy for these challenging cases.

5.1 Experiment Strategy

To cluster users based on ease of authentication, we train a CNN multiclass classifier over 148 users. Once we establish our best model, we will extract confusion matrix from this. The diagonals of confusion matrix indicate the number of true positives. We take these numbers, sort them and generate a histogram. The class of users displaying high numbers of correct classifications will be declared easy to identify. The class of users with second highest numbers of correct classifications will be declared moderate to identify. And intuitively, the third group of users will be called hard to identify and authenticate users.

We plan to apply a range of traditional machine learning models including SVM, Decision Trees, Random Forests, and k-nearest neighbors (knn) on the hard to identify cluster of users to enhance the identification accuracy for these challenging cases. These models will be trained on minimal features as well as on flattened KD images.

After a thorough experimentation and evaluation, the best-performing model will be selected. We will then fine-tune this model, enhancing its predictive accuracy and optimizing it for our specific use case.

The performance of the selected and optimized model will be evaluated using a confusion matrix and a sorted histogram. This comprehensive approach aims to build a robust multiclass classifier capable of accurately identifying users based on their keystroke dynamics, thereby significantly improving our user identification process.

5.2 Experiment Metrics

We will be using accuracy as our primary metric since the number of classes is large (148 users) and other metrics such as precision or recall may skew depending on the class size and other variables. However, we will also use the confusion matrix to visualize the distribution of correct and incorrect predictions across all classes and identify any patterns or biases in the model's performance. Additionally, as we are interested in clustering the predicted user identities, we will be using the diagonal elements of the confusion matrix to extract these clusters.

5.2.1 Accuracy

In a binary classification problem, accuracy is a measure of how well a model correctly classifies samples into one of two classes (positive or negative). It is the ratio of the number of correctly classified samples to the total number of samples in the dataset. The accuracy of a binary classifier is

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}, \tag{1}$$

where TP represents the number of true positive samples (samples correctly classified as positive), TN represents the number of true negative samples (samples correctly classified as negative), FP represents the number of false positive samples (samples incorrectly classified as positive), and FN represents the number of false negative samples (samples incorrectly classified as negative).

In a multiclass classification problem, accuracy is the proportion of correctly classified samples to the total number of samples in the dataset. We can calculate the

accuracy for multiclass classifier as

$$\text{Accuracy} = \frac{\sum_{i=1}^{n} TP_i}{N}, \tag{2}$$

where TP_i represents the number of samples of class i that are correctly classified and N is the total number of samples in the dataset.

5.2.2 Confusion Matrix

This research utilizes confusion matrices extensively. In a confusion matrix, each row and column represents a class in the dataset. The rows represent the actual classes of the samples, and the columns represent the predicted classes of the samples. The diagonal elements of the matrix represent the number of true positive (TP) and true negative (TN) samples, respectively, which are the samples that are correctly classified as positive and negative. The off-diagonal elements represent the number of false positive (FP) and false negative (FN) samples, respectively, which are the samples that are misclassified as positive and negative.

In a multiclass classification problem, the confusion matrix can have more than two classes. The diagonal elements of the matrix represent the number of correctly classified samples for each class, while the off-diagonal elements represent the number of misclassified samples for each pair of classes. By analyzing the confusion matrix, we can identify which classes are being misclassified and make improvements to the model to improve its performance.

5.3 Results of Multiclass Classification

After rigorous experimentation, we froze our hyperparameters after doing a grid search over the variables: number of epochs, learning rate, optimizer and learning schedule callbacks. Settling on a network which will run for 20 epochs, have learning rate of 0.01 and will user adam and reduceLROnPlateau as optimizer and callback

respectively. We also did some experiments with the architecture of the model itself but then settled on the model with 5 convolutional layers. Each convolutional layer followed by batch normalization and max pooling layers.

After freezing our hyperparameters and model architecture, we also did experiments with keystroke sequences used to generate KDIs. The comparitive analysis of training, testing and validation accuracies for keystrokes of length 50, 75 and 100 respectively is shown in Table 2. The model with keystroke length 50 shows signs of overfitting towards the end which can be seen from the Figure 5.

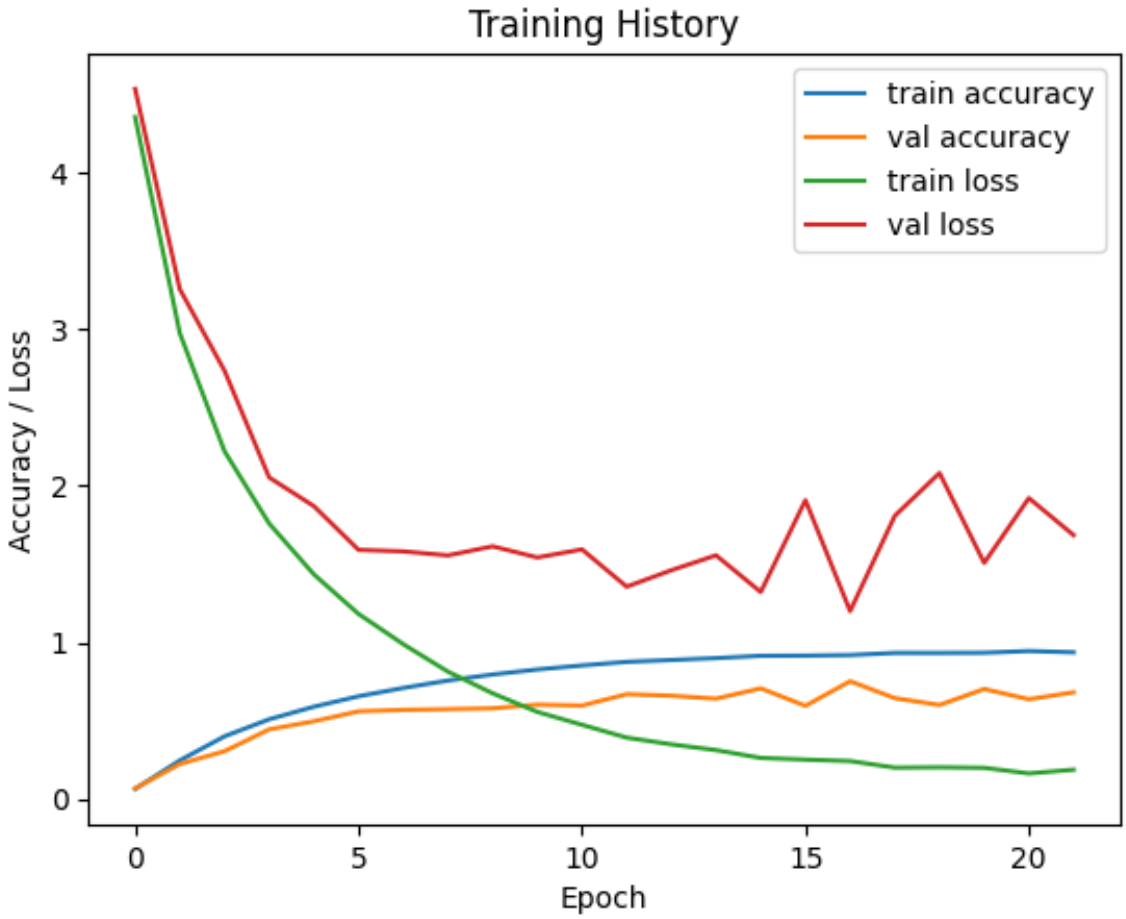

Figure 5: Epoch wise training of model with keystroke length 50

Whereas on the other hand models which were trained on keystrokes with length

75 and 100 are much more robust against overfitting as can be seen in Figure 6 and Figure 7. The validation loss is continuously dropping and both validation and training accuracies are steady climbing. Since the model with keystroke sequence 100 gave us the best testing accuracy of 79% (ref Table 2), we will extract the confusion matrix from this model to further separate user base in separate clusters.

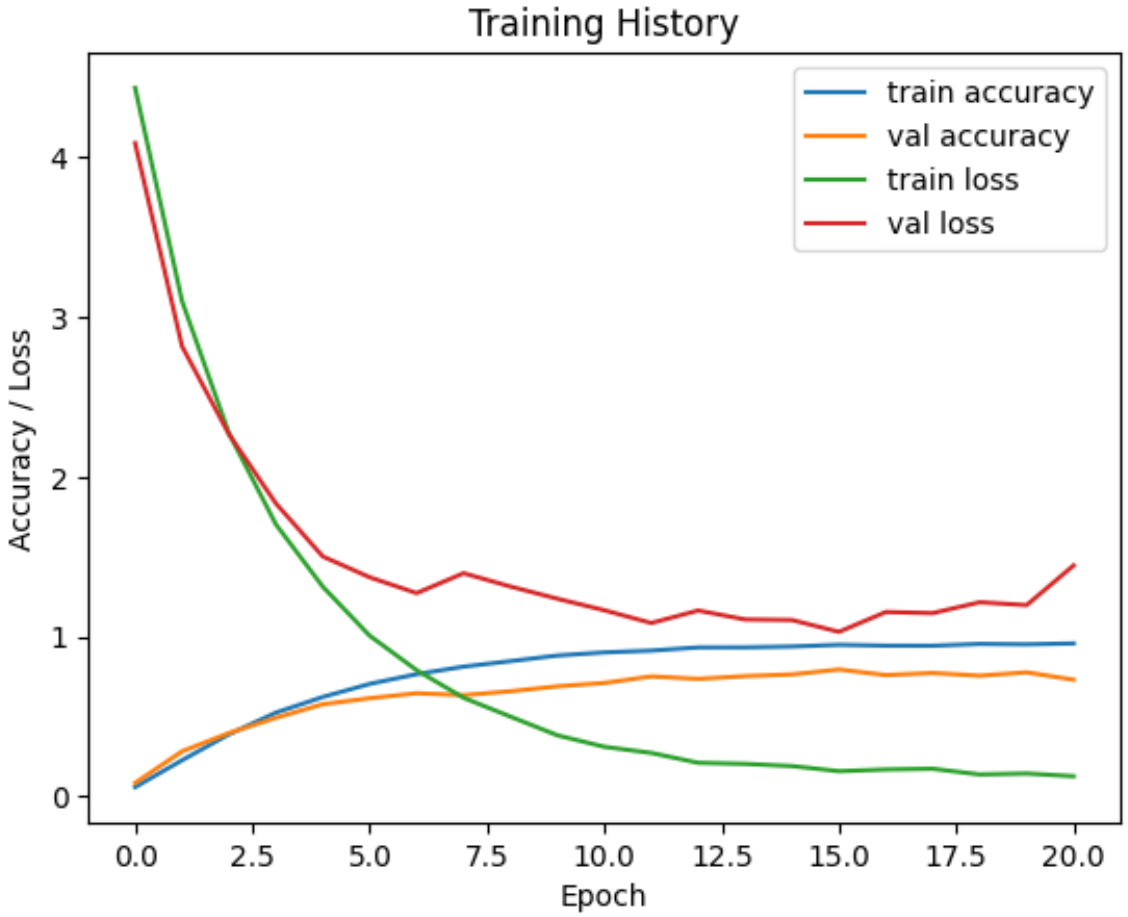

Figure 6: Epoch wise training of model with keystroke length 75

5.3.1 Extracting Confusion Matrix and Creating Clusters

As established in Section 5.3, the model which was trained on keystroke length 100, is providing the best results. We extracted the confusion matrix from this model which can be seen in Figure 8. The diagonal elements of this confusion matrix depict

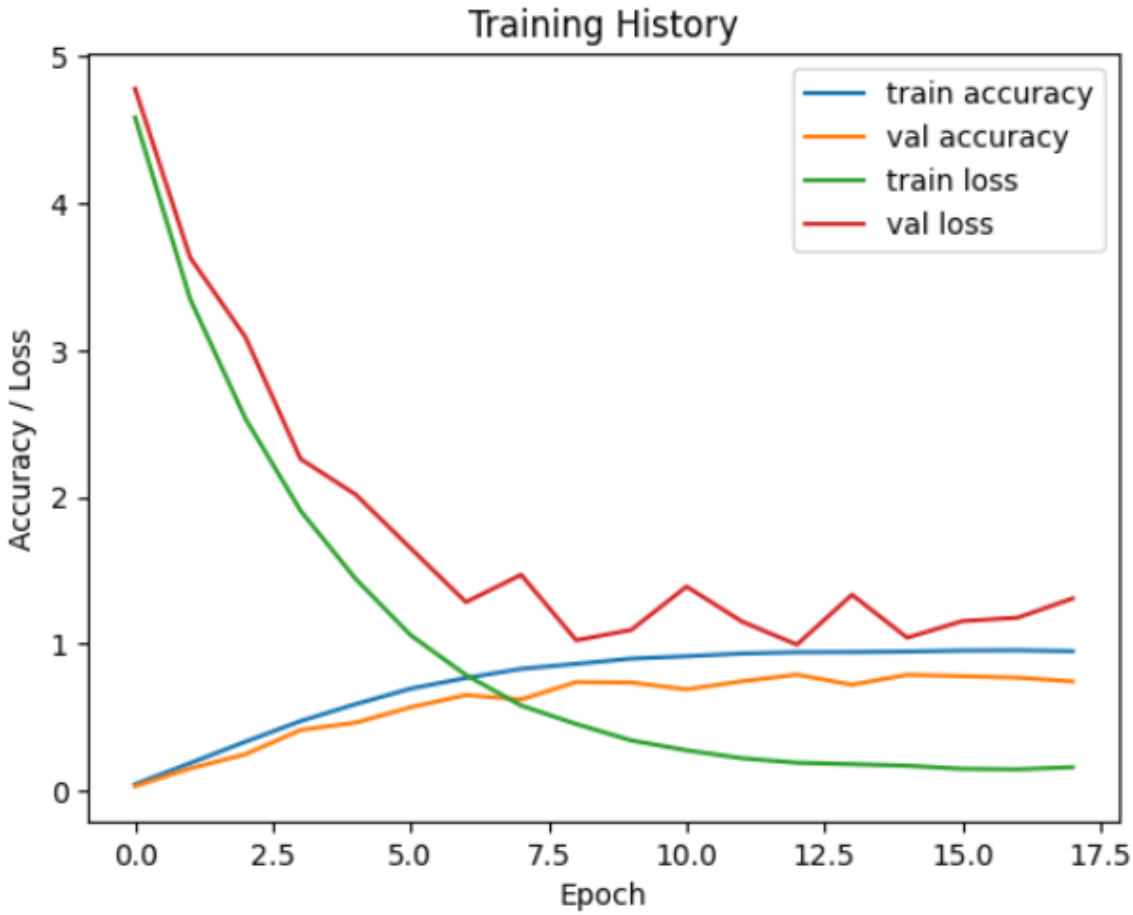

Figure 7: Epoch wise training of model with keystroke length 100

Table 2: Accuracies for different lengths of keystroke sequences

Length of Keystroke	Training Accuracy	Validation Accuracy	Testing Accuracy
50	90.74%	58.00%	67.00%
75	95.00%	73.00%	74.00%
100	97.00%	78.00%	79.00%

the correctly classified number of users in that particular class.

The histogram, as depicted in Figure 9, can be used to cluster users into three different sets based on the model's accuracy in identifying keystroke sequences for each user. The y-axis of this histogram represents the percentage of correctly identified keystroke sequences per user, while the x-axis contains the ID of the 148 users.

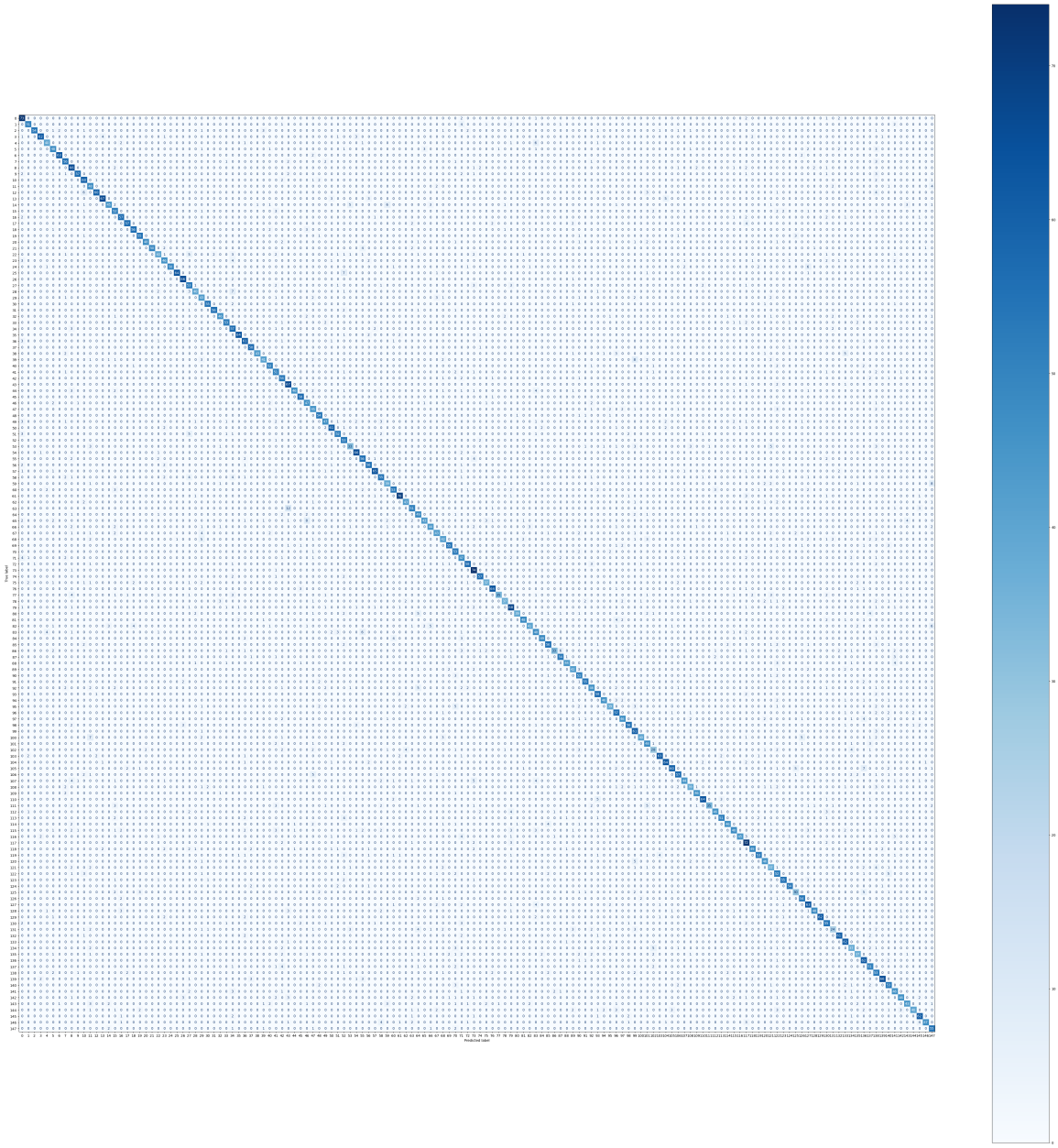

Figure 8: Confusion matrix for 148 users in dataset

We use the slope of the histogram to identify these clusters. The first change in slope occurs around $\frac{1}{5}$ of the users, and another change in slope is towards the last fourth of the users. Based on the histogram and the slope, we can establish three different accuracy thresholds for authenticating users. For easy-to-authenticate users,

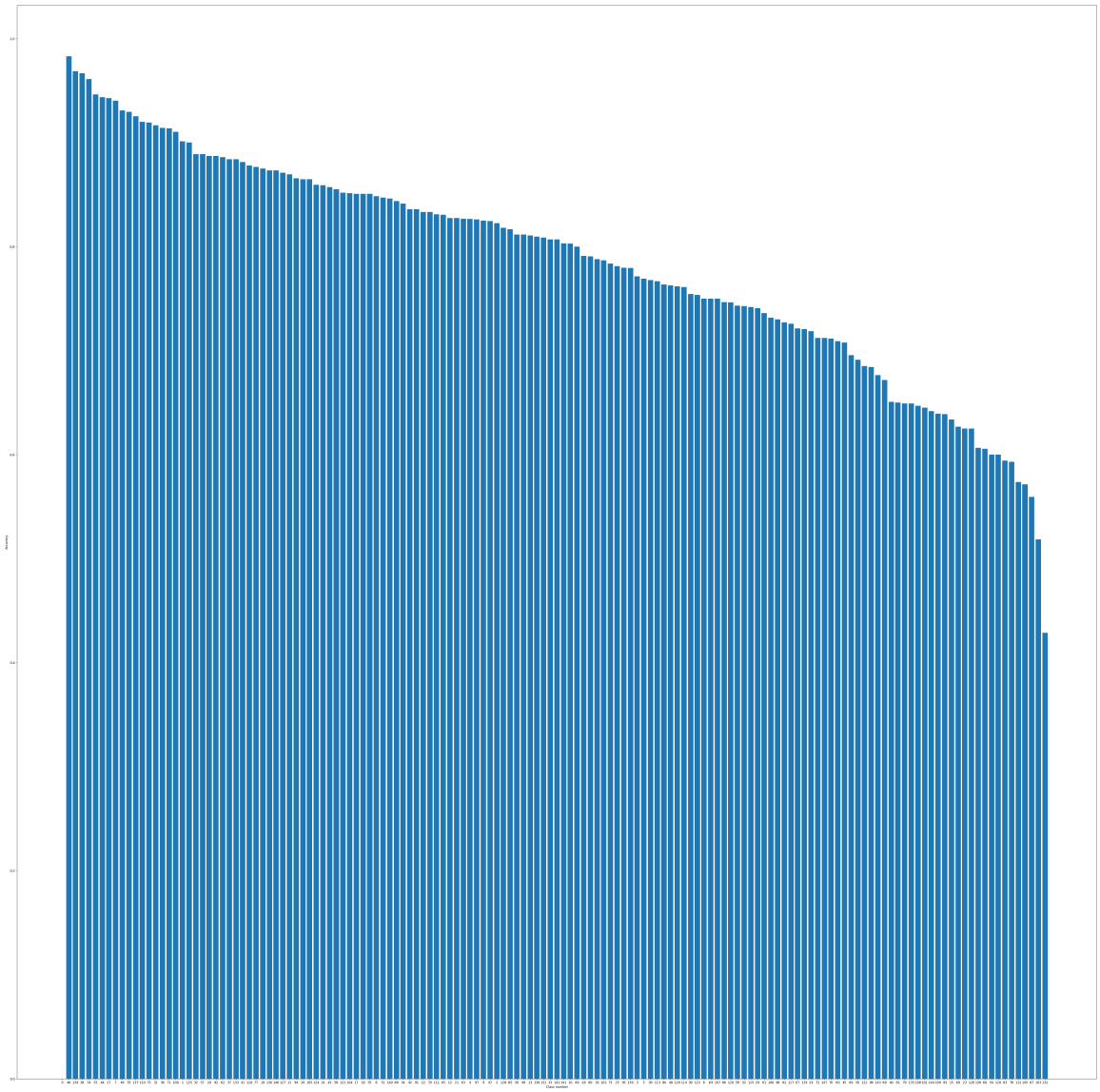

Figure 9: Histogram for 148 users in dataset

we can set the threshold to be over 90% accuracy. For moderate-to-authenticate users, we can set the threshold to be between 75% and 90% accuracy. Finally, for hard-to-authenticate users, we can set the threshold to be over 75% accuracy. The number of users in each cluster can be found in Figure 10.

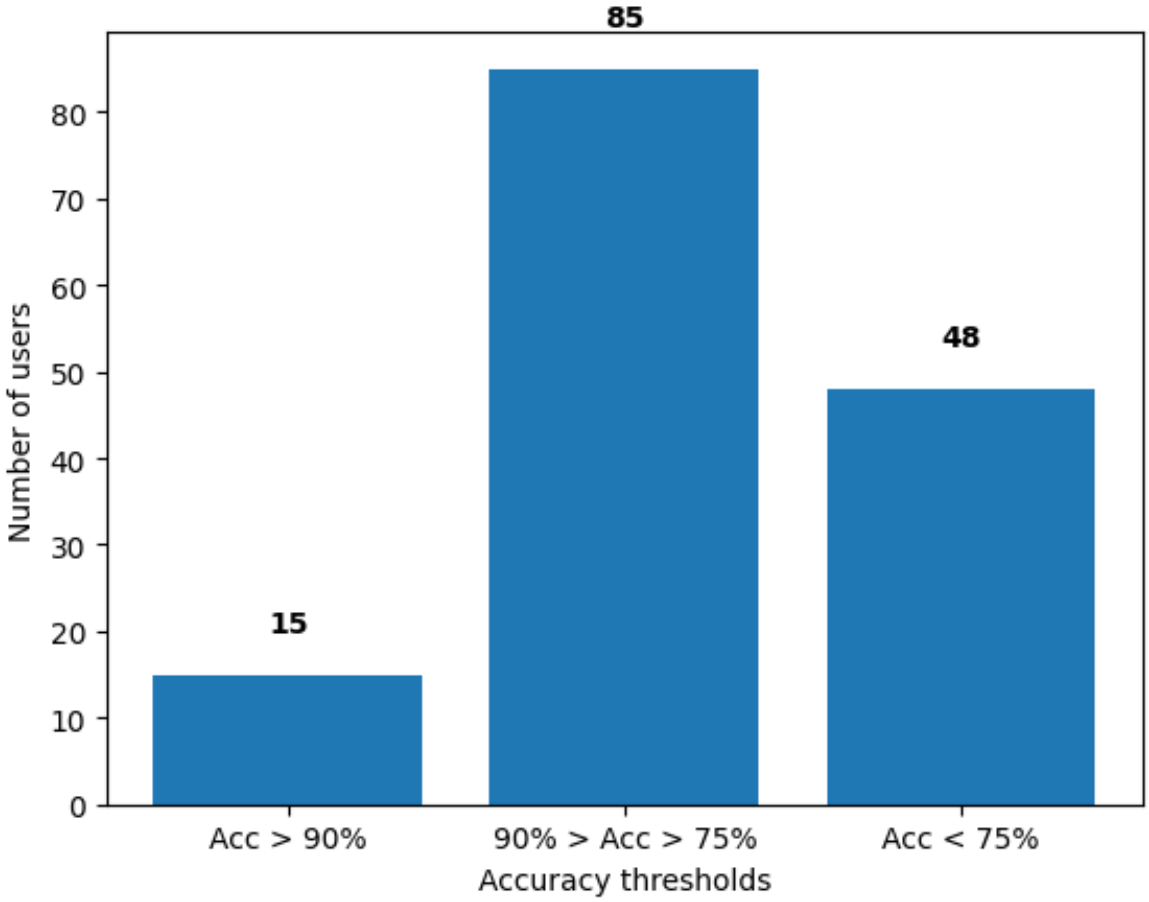

Figure 10: Count of users in each cluster.

5.4 Additional Experiments on Hard-to-Identify Users

Given the challenge in authenticating a significant portion of users using keystroke images, we decided to apply an array of traditional machine learning models to this subset of users. The logic behind this approach is to provide an additional layer of filtering for the hard-to-identify users using various algorithms, thereby maintaining uniform security across the system.

We focus on 48 users, categorized as hard-to-identify, and apply a suite of traditional machine learning models, including SVM, Decision Trees, Random Forests, and k-nearest neighbors (knn). These models serve as an additional security layer.

Before training, the data from these 48 users is pre-processed to extract minimal features like dwell time and flight time as well as flattened KD images.

5.4.1 SVM with minimal features

On the 48 users, which fall inside the hard-to-identify cluster, we apply OVO SVM multi-classification as additional security layer. Before training, the data of these 48 users has to be pre-processed to extract 2 features : dwell time and flight time. We did a stratified k-fold split with number of splits= 5 on the processed data. We achieved a total accuracy of 15%. This weak performance is no surprise as the dataset for this SVM are the hardest to identify users.

5.4.2 Traditional models with flattened keystroke images

In our approach to bolster user identification, we used flattened keystroke dynamics images as input for our machine learning models, with the user IDs serving as class labels. This methodology was applied particularly to a subset of users who were hard to identify, adding an extra layer of classification for these cases. We put four base models to test: Decision Trees, Random Forests, Support Vector Machines (SVM), and k-Nearest Neighbors (k-NN). The accuracy scores in Table 3 reveal interesting patterns. Random Forests stand out with an impressive accuracy score of 0.92, closely followed by Decision Trees with a score of 0.88. SVM, while not as high-performing, still holds its own with a moderate accuracy of 0.50. However, k-NN seems to struggle in this context, scoring only 0.06. These figures indicate a clear direction for our next steps. Given the superior performance of Random Forests, it is prudent to focus our efforts on further exploring and optimizing these model.

5.4.3 Hyperparameter tuning for Random Forest

Our approach was to further scrutinize the users that were hard to identify based on the results of a multiclass Convolutional Neural Network (CNN). In this context,

Table 3: Accuracy Scores of Machine Learning Models on Hard-to-Identify Users

Model Name	Testing Accuracy
SVM	0.50
k-NN	0.06
Decision Tree	0.88
Random Forest	0.92

we had previously defined hard-to-identify user as those for whom the CNN yielded an accuracy of less than 0.75. This approach helped us to home in on the cases that needed additional attention for reliable identification.

In order to optimize the performance of our Random Forest model, we embarked on a hyperparameter tuning exercise using a grid search methodology. The hyperparameters under consideration included the number of estimators, the number of features to consider when looking for the best split, the minimum number of samples required to split an internal node , and the minimum number of samples required to be at a leaf node. The results showed that the best performance was achieved with the parameters displayed in bold in Table 4.

Table 4: Hyperparameter tuning for Random Forest

Parameter	Values
n_estimators	100, 500, **1000**
max_features	**auto**, sqrt
min_samples_split	2, **5**
min_samples_leaf	1, **2**

Applying these parameters led to a significant improvement in model performance. The resulting model, trained on the 48 hard-to-identify users, yielded an impressive accuracy of 0.95. Furthermore, both the macro and weighted averages for precision, recall, and F1 score were also 0.95, indicating a strong balance between the model's sensitivity and specificity.

We observed that the accuracy improved significantly for the majority of these users. Specifically, 6 users moved to the the accuracy range of 0.8 to 0.9 from less than 0.75. More impressively, there were 42 users whose identification accuracy elevated to a level of 0.9 or above. This demonstrated the effectiveness of applying the Random Forest model as a supplementary layer of identification for the hard-to-identify cases.

The confusion matrix shown in Figure 11 extracted from the random forest model, provides us with a detailed breakdown of model's identification performance for each user. It allows us to see not only the overall accuracy but also where the model might be consistently misidentifying certain users, which could guide future improvements.

Alongside the confusion matrix, Figure 12 displays user-wise accuracy plots for this subset. Specifically, we created sorted histograms that visually represent the identification accuracy for each user. This visualization allows us to quickly identify any trends or patterns in the data, making it easier to understand how the model's performance across the board.

The promising results generated by random forest compelled us to try this model on all 148 users of the dataset. The results from those experiments are discussed in the appendix.

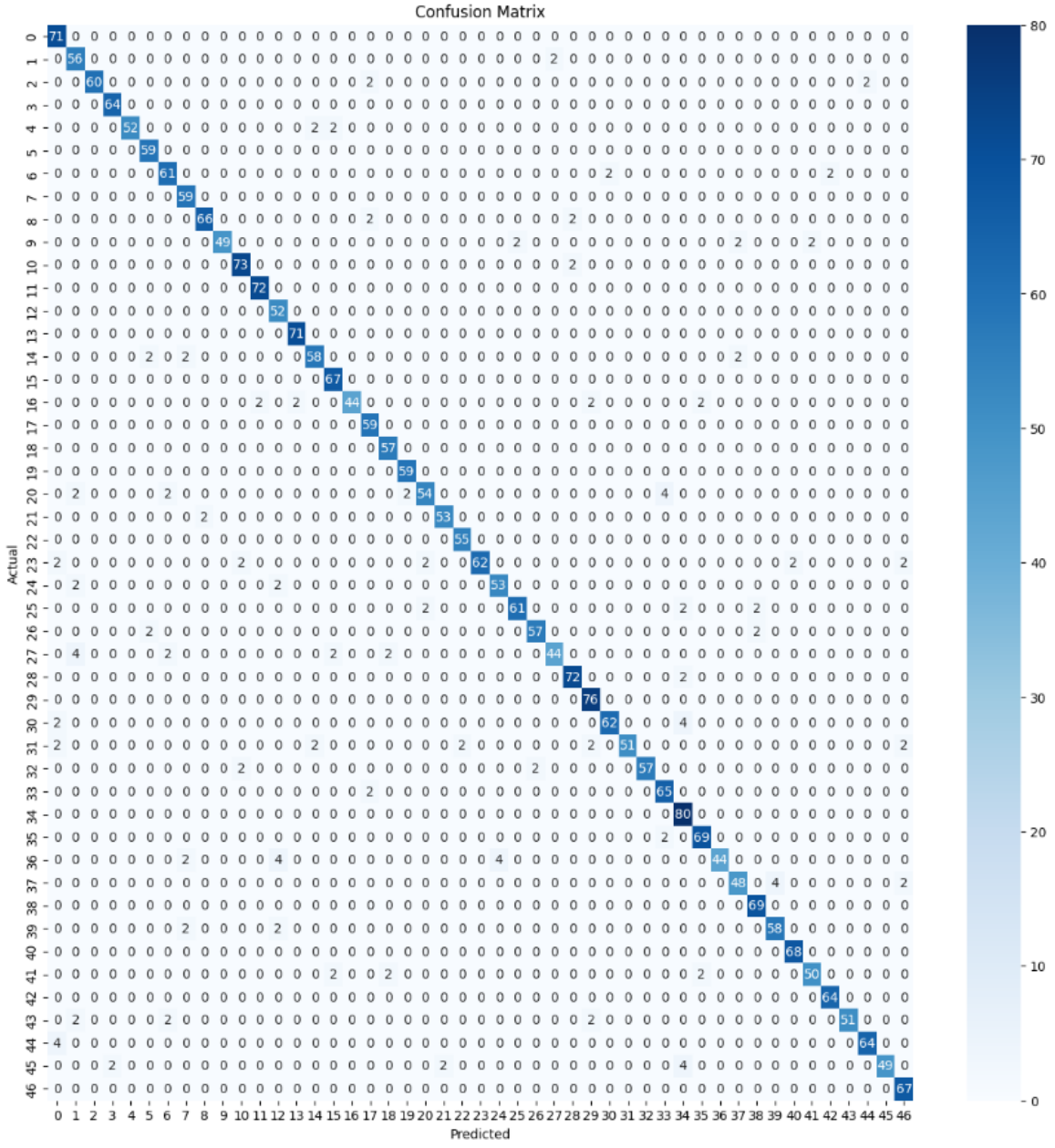

Figure 11: Confusion matrix of 48 users derived from random forest model.

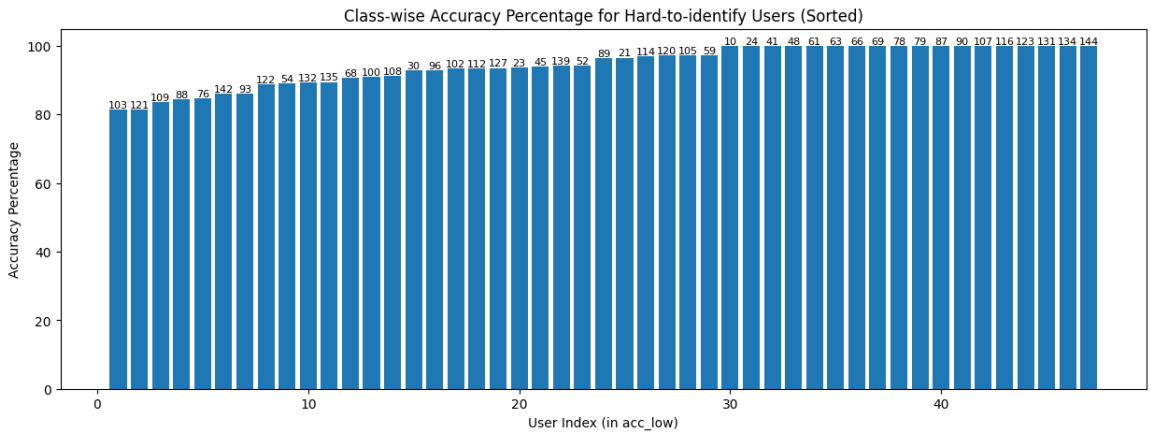

Figure 12: Sorted accuracy histogram of 48 users derived from random forest model.

CHAPTER 6

Conclusion

This project has presented a novel approach to user authentication by employing keystroke dynamics and a multiclass Convolutional Neural Network (CNN) classifier. While using keystroke image-like features with binary CNN classifiers is not new, the innovative aspect of this research lies in the application of a multiclass CNN for categorizing users into easy, moderate, and hard to authenticate clusters. This approach allows the system to optimize resource allocation and employ more intensive authentication methods for users who are harder to authenticate.

The multiclass CNN performed well, splitting the dataset into three clusters of size 15, 85, and 48. Due to the nature in which the experiment is set up, further classification methods only need to be applied to 48 users, saving resources and time that would be wasted if these methods were applied to the entire user base. This approach cuts down processing by reducing the dataset size by almost 33%.

In our quest to improve the identification accuracy of 33% hard-to-identify users, we pursued several experimental routes. Recognizing that the SVM classifier, using a minimal feature set, only attained an accuracy of 15%, we explored different methodologies. One such approach involved using flattened keystroke images as inputs to a variety of base models: Decision Tree, Random Forest, SVM, and k-NN. In these preliminary trials, the Random Forest model consistently emerged as the most accurate.

Subsequently, we performed hyperparameter tuning on this model using a grid search strategy. The resulting optimized Random Forest model demonstrated significant performance improvement. Upon applying our refined model to the hard-to-identify user cluster, we observed a substantial increase in identification accuracy. The majority of these users now achieved an accuracy of greater than or equal to

0.9, a finding that was further substantiated by our confusion matrix and accuracy histograms. This successful enhancement of user identification accuracy is a testament to the potential of this methodological approach.

The research has demonstrated that keystroke sequence length plays a significant role in the accuracy of the multiclass classifier. However, there are some limitations to the study. Such as, the research only considers time-based keystroke features, which provide limited information. Future research could explore incorporating additional features, such as digraph and trigraph latencies or biometric data, to improve the model's performance.

The Buffalo keystroke dataset was created by using mechanical keyboards. In one session the keyboard was in a natural position while in another session the keyboard was rotated by an angle. This setup is not ubiquitous. In future, it would also prove helpful to come up with a method to store keystrokes from a mobile device and then apply the same experimental setup on that data.The feasibility of implementing this approach for touch-based keystroke datasets would also be interesting, as the dynamics of touch-based interactions could differ from those of traditional keyboard inputs. Additionally, researchers could explore the integration of this authentication method with other security measures to create a more robust and adaptive multi-factor authentication system.

One potential area of future research is the development of an efficient strategy for adding new users to the existing authentication system. Currently, incorporating new users into the multiclass CNN requires retraining the entire model, which may not be practical in real-world scenarios. To overcome this challenge, researchers could explore methods for determining the cluster proximity of the new user's keystroke image. By calculating the distance between the new user's data and the existing clusters, it may be possible to assign the user to the closest cluster without the need

for retraining the entire model. This approach could facilitate the seamless integration of new users into the system while maintaining the efficiency and accuracy of the authentication process.

Another interesting direction for future research would be the development of a data collection and integration method for new users. Instead of immediately adding a new user to the system, a mechanism could be designed to accumulate the user's keystroke data over time. Once a sufficient amount of data has been collected, it could then be preprocessed, converted into an image-like representation, and appended to the appropriate cluster in the system. This method could provide a more effective and dynamic approach to handling new user data, ensuring that the authentication system remains accurate and up-to-date as new users are introduced.

In summary, the proposed approach offers a promising foundation for user identification, with the potential to improve system security and resource management. By addressing the limitations and expanding the scope of the research, this work can contribute to the development of more efficient and accurate authentication techniques, ultimately benefiting both users and systems alike.

LIST OF REFERENCES

[1] M. Kowtko, ''Biometric authentication for older adults,'' in *2014 IEEE Long Island Systems, Applications and Technology Conference (LISAT)*. IEEE, 2014, pp. 1--6.

[2] A. Sasse and K. Krol, *Usable Biometrics for an Ageing Population*. Institution of Engineering and Technology, 2013.

[3] B. C. Stanton, M. F. Theofanos, and C. L. Sheppard, ''A study of users with visual disabilities and a fingerprint process,'' National Institute of Standards and Technology, Tech. Rep. NISTIR 7435, 2008.

[4] BioCatch, ''Major global banks invest $20 million in biocatch and join american express ventures on new client innovation board,'' https://www.biocatch.com/press-release/major-global-banks-invest-20-million-in-biocatch-and-join-american-express-ventures-on-new-client-innovation-board, February 2021, accessed: April 13, 2023.

[5] J. R. Montalvão Filho and E. O. Freire, ''On the equalization of keystroke timing histograms,'' *Pattern Recognition Letters*, vol. 27, no. 13, pp. 1440--1446, 2006. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167865506000602

[6] K. S. Killourhy and R. A. Maxion, ''Comparing anomaly-detection algorithms for keystroke dynamics,'' in *2009 IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*. IEEE, 2009, pp. 125--134.

[7] J. Li, H.-C. Chang, and M. Stamp, ''Free-text keystroke dynamics for user authentication,'' 2021. [Online]. Available: https://arxiv.org/abs/2107.07009

[8] R. J. Anderson, *Security Engineering: A Guide to Building Dependable Distributed Systems*, 1st ed. USA: John Wiley & Sons, Inc., 2001.

[9] K. Revett, H. Jahankhani, S. T. de Magalh aes, and H. M. D. Santos, ''A survey of user authentication based on mouse dynamics,'' in *Global E-Security*, 2008, pp. 210--219.

[10] A. Ahmed and I. Traore, ''Biometric recognition based on free-text keystroke dynamics,'' *IEEE Transactions on Cybernetics*, vol. 44, 05 2013.

[11] H. Saevanee and P. Bhatarakosol, ''User authentication using combination of behavioral biometrics over the touchpad acting like touch screen of mobile device,''

in *2008 International Conference on Computer and Electrical Engineering*, 2008, pp. 82--86.

[12] R. Gaines, W. Lisowski, S. Press, and N. Shapiro, ''Authentication by keystroke timing: Some preliminary results,'' *Rand Report*, no. R-2526-NSF, 1980.

[13] S. Bleha, C. Slivinsky, and B. Hussien, ''Computer-access security systems using keystroke dynamics,'' *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 12, pp. 1217--1222, 1990.

[14] R. Giot, M. El-Abed, and C. Rosenberger, ''Keystroke dynamics with low constraints svm based passphrase enrollment,'' in *2009 IEEE 3rd International Conference on Biometrics: Theory, Applications, and Systems*, 2009, pp. 1--6.

[15] J. Hu, D. Gingrich, and A. Sentosa, ''A k-nearest neighbor approach for user authentication through biometric keystroke dynamics,'' in *2008 IEEE International Conference on Communications*, 2008, pp. 1556--1560.

[16] K. Revett, H. Jahankhani, S. T. de Magalh aes, and H. M. D. Santos, ''A survey of user authentication based on mouse dynamics,'' in *Global E-Security*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 210--219.

[17] J. Robinson, V. Liang, J. Chambers, and C. MacKenzie, ''Computer user verification using login string keystroke dynamics,'' *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 28, no. 2, pp. 236--241, 1998.

[18] S.-s. Hwang, H.-j. Lee, and S. Cho, ''Account-sharing detection through keystroke dynamics analysis,'' *International Journal of Electronic Commerce*, vol. 14, no. 2, pp. 109--126, 2009. [Online]. Available: https://doi.org/10.2753/JEC1086-4415140204

[19] S. Ravindran, C. Gautam, and A. Tiwari, ''Keystroke user recognition through extreme learning machine and evolving cluster method,'' in *2015 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC)*, 2015, pp. 1--5.

[20] M. Liu and J. Guan, ''User keystroke authentication based on convolutional neural network,'' in *Mobile Internet Security*, I. You, H.-C. Chen, V. Sharma, and I. Kotenko, Eds. Singapore: Springer Singapore, 2019, pp. 157--168.

[21] Y. B. W. Piugie, J. Di Manno, C. Rosenberger, and C. Charrier, ''Keystroke dynamics based user authentication using deep learning neural networks,'' in *2022 International Conference on Cyberworlds (CW)*, 2022, pp. 220--227.

[22] L. Xiaofeng, Z. Shengfei, and Y. Shengwei, ''Continuous authentication by free-text keystroke based on cnn plus rnn,'' *Procedia Computer Science*, vol. 147, pp. 314--318, 2019, 2018 International Conference on Identification, Information and Knowledge in the Internet of Things. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877050919302935

[23] A. Mhenni, C. Rosenberger, and N. E. B. Amara, ''Keystroke dynamics classification based on lstm and blstm models,'' in *2021 International Conference on Cyberworlds (CW)*, 2021, pp. 295--298.

[24] T. Anurag, ''An improved user identification based on keystroke-dynamics and transfer learning,'' *Webology*, vol. 19, pp. 5369--5387, 01 2022.

[25] M. Hearst, S. Dumais, E. Osuna, J. Platt, and B. Scholkopf, ''Support vector machines,'' *IEEE Intelligent Systems and their Applications*, vol. 13, no. 4, pp. 18--28, 1998.

[26] J. R. Quinlan, ''Induction of decision trees,'' *Machine Learning*, vol. 1, no. 1, pp. 81--106, 1986. [Online]. Available: https://doi.org/10.1007/bf00116251

[27] L. Breiman, ''Random forests,'' *Machine Learning*, vol. 45, no. 1, pp. 5--32, 2001. [Online]. Available: https://doi.org/10.1023/a:1010933404324

[28] S. Albawi, T. A. Mohammed, and S. Al-Zawi, ''Understanding of a convolutional neural network,'' in *2017 International Conference on Engineering and Technology (ICET)*, 2017, pp. 1--6.

[29] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, ''Dropout: A simple way to prevent neural networks from overfitting,'' *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929--1958, 2014.

[30] T. Devries and G. W. Taylor, ''Improved regularization of convolutional neural networks with cutout,'' *ArXiv*, vol. abs/1708.04552, 2017.

APPENDIX

Random forest on complete dataset

A.1 Experiment results

Following the impressive results obtained from the refined Random Forest model on the subset of hard-to-identify users, we felt compelled to extend our experiment to include all 148 users in our dataset.

When we employed the Random Forest model with flattened images as input across the entire user base, the accuracy markedly increased to 0.93. Figure A.13 shows the confusion matrix extracted from this model for all 148 users. This outcome signifies a substantial improvement over the original multiclass CNN on 5-channel KDI, which had only achieved an accuracy of 0.791. For comparison with histograms extracted from the multiclass CNN, Figure A.14 displays the sorted histogram of user-wise accuracy scores we got from our random forest

The new clusters are displayed in Figure A.15. We found that only a small fraction of users, specifically 8, now had an accuracy of 0.75 or lower. Meanwhile, 27 users fell into the accuracy range of 0.75 to 0.9. The vast majority of users, 109 in total, achieved an accuracy of 0.9 or higher.

These results underscore the efficacy of the Random Forest model with flattened image input in significantly enhancing user identification accuracy across a broad user base.

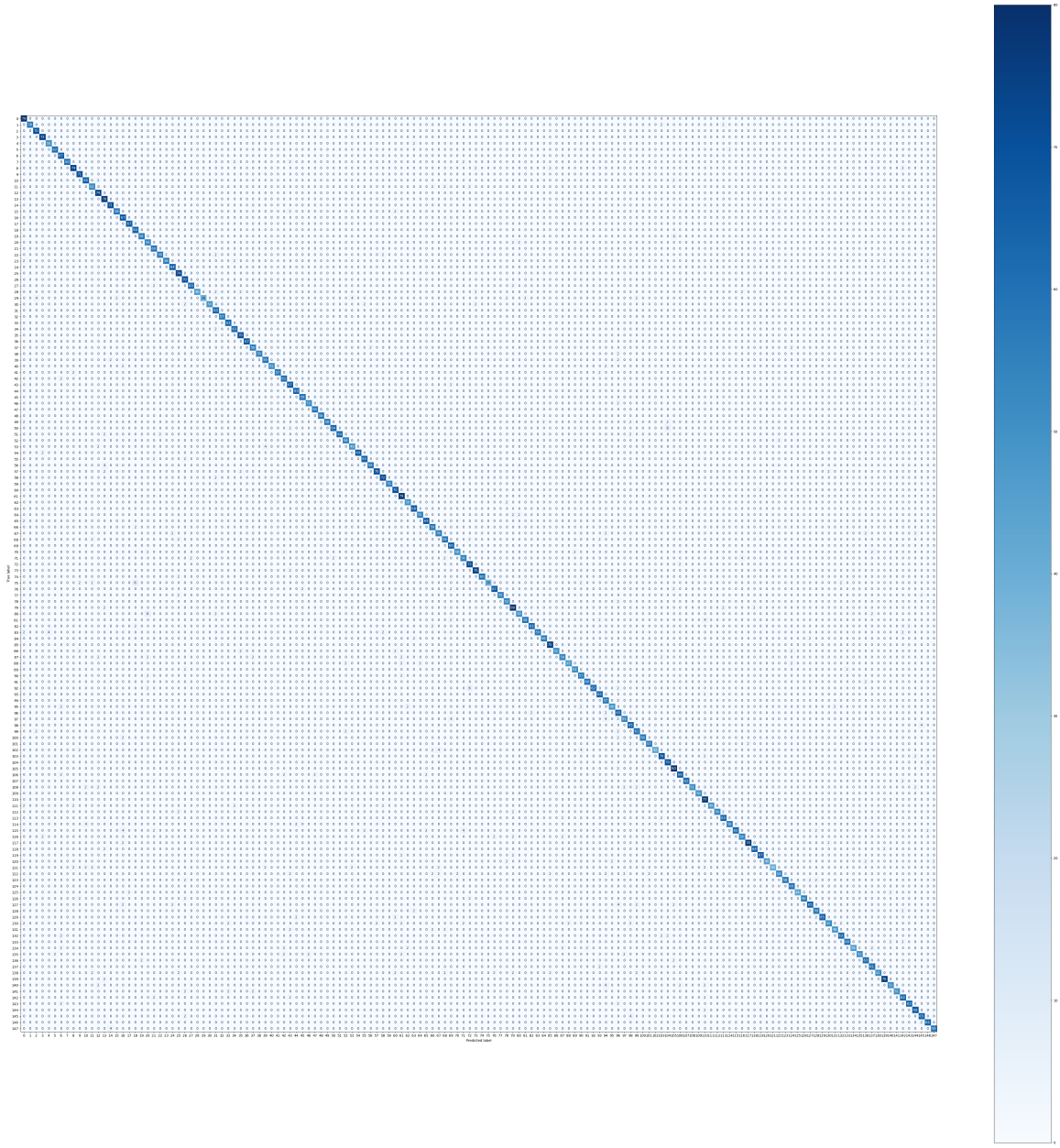

Figure A.13: Confusion matrix of 148 users derived from random forest model.

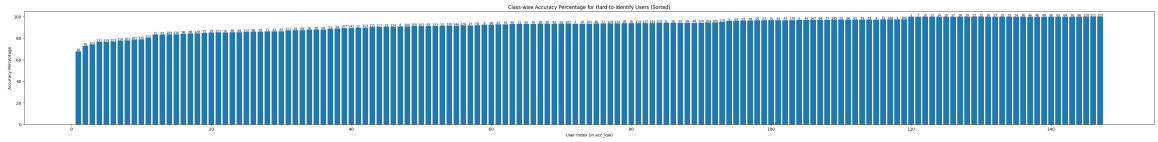

Figure A.14: Histogram of 148 users derived from random forest model.

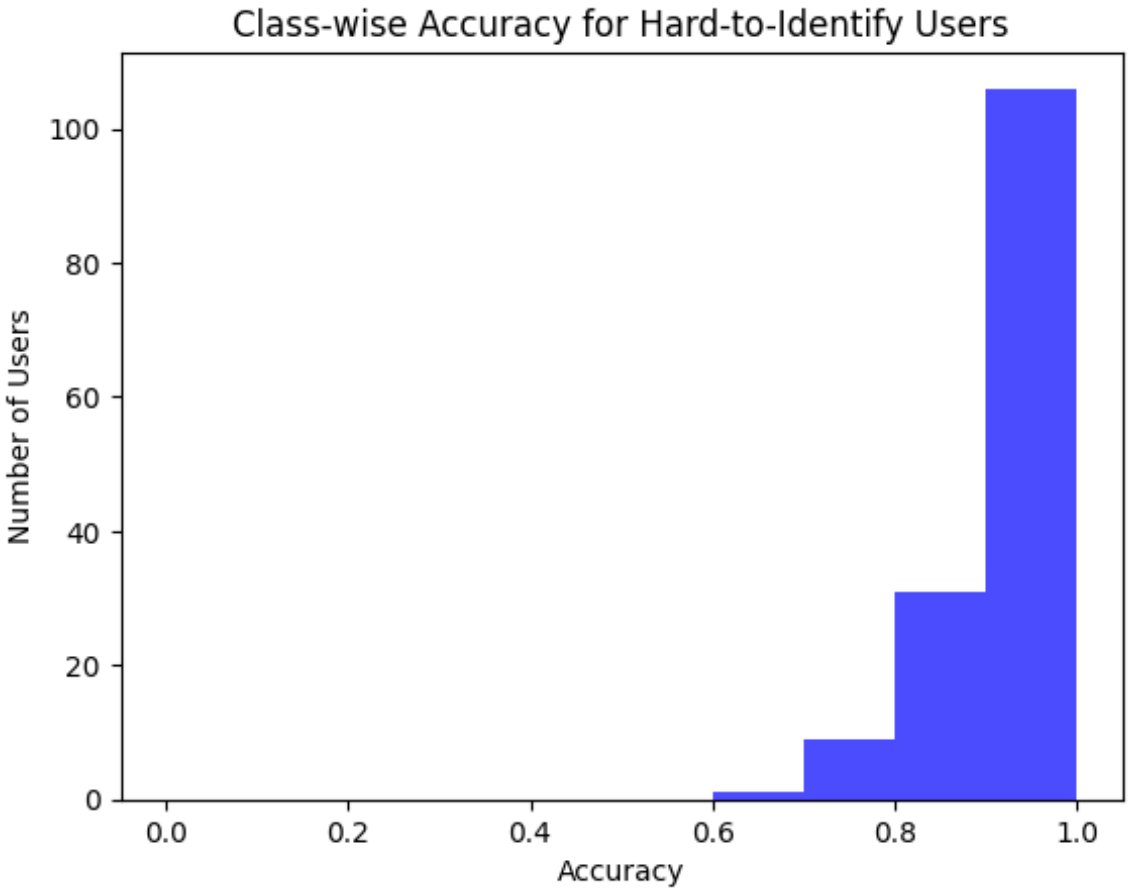

Figure A.15: Accuracy range of 148 users derived from random forest model.