CONTINUOUS AUTHENTICATION ON SMART PHONES THROUGH BEHAVIORAL BIO-METRICS

(AKILLI CİHAZLARDA DAVRANIŞSAL BİYOMETRİYLE SÜREKLİ KİMLİK DOĞRULAMA)

by

Hasan Can VOLAKA, B.S.

Thesis

Submitted in Partial Fulfillment

of the Requirements

for the Degree of

MASTER OF SCIENCE

in

COMPUTER ENGINEERING

in the

GRADUATE SCHOOL OF SCIENCE AND ENGINEERING

of

GALATASARAY UNIVERSITY

Sep 2019

This is to certify that the thesis entitled

CONTINUOUS AUTHENTICATION ON SMART PHONES THROUGH BEHAVIORAL BIO-METRICS

prepared by **HASAN CAN VOLAKA** in partial fulfillment of the requirements for the degree of **Master of Science in Computer Engineering Department** at the **Galatasaray University** is approved by the

Examining Committee:

Assoc. Prof. Dr. Özlem DURMAZ İNCEL
Supervisor, **Department of Computer Engineering**
Galatasaray University

[illegible]

Assist. Prof. Dr. Günce Keziban Orman
Department of Computer Engineering
Galatasaray University

[illegible]

Assist. Prof. Dr. Berk Gökberk
Department of Computer Engineering
MEF University

[illegible]

Date: 24/10/2019

ACKNOWLEDGMENTS

This work is supported by Tubitak under grant number 5170078 and by the Galatasaray University Research Fund under grant number 17.401.004.

I also want to thank Assoc. Prof. Özlem Durmaz İncel for her unlimited support and guidance with this thesis, Assoc. Prof. S. Emre Alptekin and Assoc. Prof. Gülfem Işıklar Alptekin for their support on the creation of our deep-learning dataset, my family, and my friends for their moral support and Ilgın Tezel for her endless patience and never stop believing me through this journey.

TABLE OF CONTENTS

LIST OF SYMBOLS

EER	**:**	Equal Error Rate
FAR	**:**	False Acceptance Rate
FRR	**:**	False Rejection Rate
MAE	**:**	Mean Absolute Error
MSE	**:**	Mean Square Error
ML	**:**	Machine Learning
SVM	**:**	Support Vector Machine
DL	**:**	Deep Learning
NN	**:**	Neural Network
ACC	**:**	Accelerometer Features
GYRO	**:**	Gyroscope Features
MAG	**:**	Magnetometer Features
STD	**:**	Standard Deviation
RBF	**:**	Radial Basis Function

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

Smartphones have become essential objects for our daily lives. Besides their original purpose of use, people use these devices as their personal assistants. Additionally, smartphones provide large internal storage which enables users to store their private information, such as personal photos, contact details, call histories, etc. On the other hand, because of their small sizes, these devices could easily get lost or stolen. Therefore, providing the security and privacy of smartphone users against unauthorized access is a significant and crucial area of research. One of the solutions is the use of behavioral biometrics, which tracks and identify users' interaction patterns with the device. In this study, we investigate the impact of using both touchscreen-based and sensor-based features in an authentication model using deep learning, multi-class and one-class machine learning models. Mainly, we train a three-layer deep network on the combined feature-sets and applied classification for revealing the behavioral characters of users for building an authentication model. Then we improved our feature set and used this data with our machine learning models. We use HMOG dataset that includes data from 100 users over 24 sessions. We train different networks with different combinations of input data, namely only touch-screen data, only sensor data, and their combination. Our results show that we can achieve 88% accuracy in average with deep learning network, and more than %99 f1 score and accuracy with svm models, and 15% EER values considering binary classification when different types of data are used together.

Keywords : sensor fingerprinting; mobile device identification; motion sensors; mobile device sensors

RÉSUMÉ

Les smartphones sont devenus des objets essentiels dans notre vie quotidienne. Outre leur objectif initial, les utilisateurs utilisent ces appareils comme assistants personnels. De plus, les smartphones offrent un grand stockage interne qui permet aux utilisateurs de stocker leurs informations privées, telles que leurs photos personnelles, leurs coordonnées, l'historique des appels, etc. Par contre, en raison de leur petite taille, ces appareils peuvent facilement être perdus ou volés. Par conséquent, la sécurité et la confidentialité des utilisateurs de smartphones contre les accès non autorisés constituent un domaine de recherche important et crucial. L'une des solutions consiste à utiliser la biométrie comportementale, qui permet de suivre et d'identifier les schémas d'interaction des utilisateurs avec le périphérique. Dans cet article, nous étudions l'impact de l'utilisation de fonctionnalités à la fois d'écran tactile et de capteurs dans un modèle d'authentification utilisant des méthodes d'apprentissage approfondi et des méthodes d'apprentissage automatique. Nous formons principalement un réseau profond à trois couches sur les ensembles de fonctionnalités combinés et sur la classification appliquée afin de révéler les caractères comportementaux des utilisateurs afin de créer un modèle d'authentification. Nous utilisons un jeu de données HMOG qui comprend les données de 100 utilisateurs sur 24 sessions. Nous formons différents réseaux avec différentes combinaisons de données d'entrée, à savoir uniquement les données d'écran tactile, uniquement les données de capteur et leur combinaison. Nos résultats montrent que nous pouvons atteindre des valeurs de précision de 88% pour apprentissage approfondi, et de prcision et de score f1 plus de %99 pour des modèles apprentissage automatique, et d'EER de 15% en tenant compte de la classification binaire lorsque différents types de données sont utilisés ensemble.

Mots Clés : capteur d'empreintes digitales; identification de l'appareil mobile; capteurs de mouvement; capteurs d'appareils mobiles

ÖZET

Akıllı telefonlar günlük hayatımız için vazgeçilmez eşyalar haline geldiler. Orijinal kullanım amaçlarının yanı sıra, insanlar bu cihazları kişisel asistanları olarak da kullanmaktadırlar. Ek olarak, akıllı telefonlar, kişisel fotoğraflar, iletişim bilgileri, çağrı geçmişleri vb. gibi özel bilgileri saklayabilen geniş dahili depolama da sağlar. Öte yandan, küçük boyutlarından dolayı bu cihazlar kolayca kaybolabilir veya çalınabilir. Bu nedenle, akıllı telefon kullanıcılarının yetkisiz erişime karşı güvenliklerini ve gizliliklerini sağlamak önemli bir araştırma alanı oluşturmaktadır. Çözümlerden biri, kullanıcıların cihazla etkileşim kalıplarını takip eden ve tanımlayan davranışsal biyometri kullanımıdır. Bu yazıda hem dokunmatik ekran hem de sensör tabanlı özelliklerinin, derin öğrenme ve makine öğrenmesi yöntemleri kullanarak, davranışsal biyometri modeline etkilerini araştırdık. Temel olarak, bir kimlik doğrulama modeli oluşturmak için kullanıcıların davranış karakterlerini ortaya çıkarmak için birleşik özellik setleri ve uygulamalı sınıflandırma üzerine üç katmanlı bir derin ağ kurduk. Veriseti olarak, 24 oturumda 100 kullanıcının verisini içeren HMOG verisetini kullanıyoruz. Farklı dokunmatik ekran verileri, sadece sensör verileri ve bunların kombinasyonları gibi farklı girdi verileri kombinasyonlarına sahip farklı ağları eğitiyoruz. Sonuçlarımız, farklı veri türleri bir arada kullanıldığında ikili sınıflandırma dikkate alındığında derin öğrenme modeli için ortalama %88 doğruluk ve %15 EER değerlerine, makine öğrenmesi modelleri için ise %99'u aşan f1 skoru ve doğruluk sonuçlarına ulaşabileceğimizi göstermektedir.

Anahtar Kelimeler : sensör verisine dayalı parmak izi oluşturma; mobil cihaz tanıma; hareket sensörleri; akıllı cihaz sensörleri

1 INTRODUCTION

Smartphones are essential tools in our daily lives. A survey report by Pew Research Center(Center, accessed April 2019) shows that 81% of U.S. adults have smartphones. The same research also shows that smartphone users use these devices not just for calling and texting but also for looking for a job, finding a date, reading a book or doing online shopping. Online banking, mailing, playing games can also be added to this list. People report that smartphones are the third essential tool after their wallets and keys when leaving home.

Providing security and privacy for these tools is essential since they carry personal and other sensitive information, such as passwords or photos. Moreover, they may get stolen, lost or can be accessed by non-users due to their small sizes. The same survey reports that 28% of U.S. smartphone owners say they don't use a screen lock or other features to secure their phone. Although a majority of smartphone users say, they have updated their phone's apps or operating system, around four-in-ten say they only update when it is convenient for them. However, some smartphone users forgot updating their phones altogether: 14% say they never update their phone's operating system, while 10% say they do not update the apps on their phone.

Considering these statistics, securing mobile devices is a leading security challenge because it depends on human attitude or preferences. There are different approaches for securing authentication on these devices. Widely-used methods include PIN and pattern-based authentication. However, these methods are weak against shoulder-surfing, smudge, and other attacks. Additionally, they provide one-time authentication. Regarding this behavior, methods which focus on passive security are gaining importance to answer questions about how to solve those security challenges.

Continuous authentication according to the interaction patterns of the user is an emerging alternative solution (Patel et al., 2016; Alzubaidi and Kalita, 2016). This approach is also called behavioral biometrics where interaction patterns of a user are tracked instead of using physical bio-metrical information, such as a fingerprint. Continuous authentication has several advantages, such as working in the background and not disrupting the user experience and working over a session rather than a one-shot authentication.

There are two survey papers (Patel et al., 2016; Alzubaidi and Kalita, 2016) that investigate the use of biometrics for continuous authentication on smartphones. In (Patel et al., 2016), it was emphasized that sensors such as camera, microphone, etc. can be used to collect physical data, while components such as accelerometers, gyroscopes, touch screens can be used to collect behavioral biometric data such as walking, screen touch gestures, and hand gestures. In the other review paper (Alzubaidi and Kalita, 2016), the studies in the literature were examined in terms of the type and size of data collected, classifiers used in identification, and results obtained.

In this study, we investigate continuous authentication on mobile phones by the identifying users using both sensor and touch screen data. Notably, we use the HMOG dataset(Sitová et al., 2016) which includes both data from motion sensors -accelerometer, gyroscope and magnetometer- available on smartphones and also from the touch-screen. It was collected from 100 users over 24 sessions. Although this dataset has been analyzed using traditional machine learning algorithms (Sitová et al., 2016), to the best of our knowledge, there is only one study (Amini et al., 2018) that applies LSTM and RNN based deep learning algorithms on the same dataset. They reported 81.32% accuracy using LSTM. We aim to increase accuracy which can be found in related works, by using a combination of screen and sensor data. Additionally, we use a different type of network constructed over feature sets. Instead of using the raw data, we extracted basic and simple features, such as mean, standard deviation and median from the readings of the sensors as well as the touch screen readings. We used those features both in our deep learning model and in our svm models. The highlights and contributions can be summarized as follows:

- We apply deep neural networks on a large dataset and model the problem as a binary classification problem, rather than a one-class problem, which was studied in (Sitová et al., 2016).
- We explore the effect of different types of modalities, such as touch-screen and motion sensors in identifying users.
- Our results show that we can identify users with 88% accuracy on a challenging dataset using a primitive deep learning model. Although results were similar, the combination of touch (scroll) and gyroscope features revealed a slightly better performance.

— With combining the screen and sensor data, we identified users with an f1 score more than 90% and 99% using one class svm and binary svm classification respectively.

Since deep learning algorithms work as supervised machine learning algorithms, we need to provide labeled and multi-class data. But if we are talking about biometric data, each user's data has a significant privacy importance. That's why, in literature, we can easily see one class algorithms.

Regarding this knowledge, we tested our dataset with a second methodology, OneClass SVM Algorithm. We changed the parameters and tried to find the best parameters for all users in HMOG data set. We also changed the creation of dataset which we will explain in Chapter 4. With this algorithm, we identified users with an f1 score more than 90%.

Additional to OneClass SVM, we wanted to have an intermediary model between One-Class SVM and Deep Learning Network, so we decided to use binary SVM classification which is a support vector and a multi-class classification. With biary SVM algorithm, we identified users with an f1 score more than 99%.

Rest of the study is organized as follows: In Chapter 2, we present the related work and how our method differs from the related studies. In Chapter 3, we describe the technologies we used as deep network and sensors, and tell what these sensors are used for. Chapter 4 presents our methodology particularly the parameters and experiments considered in this study. In Chapter 5, we present the results of model evaluation and discuss our findings. Finally, Chapter 6 concludes the study and includes future studies.

2 RELATED WORK

Biometrics are mainly grouped into two categories: behavioral and physical biometrics. Physical biometrics are based on physical attributes of a person, such as a retina or an iris scan and fingerprint, etc. As mentioned, behavioral biometrics are based on a person's behavior and analysis of a person's handwriting, timing keystroke and usage style, etc.

There are four broad categories of studies focusing on continuous authentication: i) keystroke-based authentication, ii) touchscreen-based authentication, iii) sensor-based authentication, iv) multi-modal authentication. Keystroke based authentication mainly focuses on the analysis of typing motions of users. However, in the utilized dataset keystroke-data was not recorded.

HMOG (Hand Movement, Orientation and Grasp) dataset (Sitová et al., 2016) includes recordings both from touch-screen and sensors. Accelerometer, gyroscope and magnetometer readings and tap-based features, such as x-y coordinates, finger covered area, pressure, etc., are collected from 100 smartphone users with 24 sessions. Besides the touchscreen related data, authors propose a new set of features, which are derived from micro-movements, obtained from accelerometer, gyroscope and magnetometer sensors data generated while users interact with the touchscreen. Feature selection, feature transformation with principal component analysis (PCA) and outlier removal are performed on these feature sets. They achieved EER of 15.1% using HMOG features combined with tap features.

In (Buriro et al., 2016), authors propose a new multi-modal biometric authentication model which is based on the features which are collected while the user slide-unlocks the smartphone to answer a call. The features were populated by slide/swipe, arm movements of the user answering a call (accelerometer, gyroscope, orientation sensors) and voice recognition. The complete system consists of four parts: slide movement recognition, pickup movement recognition, voice recognition, and fusion. Twenty-six participants (sixteen male and ten female) were recruited in various ages. Each participant performs at least 20 swipe, 20 pick-ups, and 10 voice sample. They applied to the feature set one-class Bayes-Net, one-class random forest, and one-class sequential minimal optimization (SMO) classifiers. They achieved best results with the naive Bayes network classifier with a FAR of 11.01% and an FRR of 4.12%.

In recent studies, deep learning methods for continuous authentication are also utilized. In (Centeno et al., 2018), a Siamese convolutional neural network is used to learn the signatures of the motion patterns from users. The network is used for deep feature extraction, while 1-class SVM is used for classification. They report verification accuracy up to 97.8% on a dataset (Yang et al., 2014) consisting of 100 users. Similarly in (Centeno et al., 2017), the feature extraction process is based on a deep learning autoencoder, using only accelerometer data. 2.2% EER was reported for a re-authentication time of 20 seconds.

In another study (Chang et al., 2018), Kernel Deep Regression Network is used for both feature extraction and classification. They report 0.121% EER for inter-week authentication on Touchalytics dataset (Frank et al., 2013). However, this dataset includes only touchscreen data. In another recent study (Amini et al., 2018), HMOG dataset is utilized, and a deep learning framework for user fingerprinting is proposed. However, they achieved 81.32% accuracy using LSTM which is lower than our findings.

In this thesis, we tried to make a different approach by combining sensor data and the touch data to recognize non genuine usage of the phone. Related works are used an approach to use sensor data or touch screen data individually and used machine learning algorithms in general. But in this paper we combined the dataset as we described in Chapter 4 and we also tried to use a primitive deep learning model in contrary of only using machine learning models.

3 TECHNOLOGIES

In this chapter, we describe what is a deep network and other classification methods, also we explain the details of the dataset.

3.1 Classification Methodologies

3.1.1 Deep Network

Researchers were eager to create systems and machines which work and decide like humans. Therefore they came with the idea of mimicking the human brain, which is the decision center of a human, so they created perceptrons and neural networks. Even we can find research papers about multi-layer perceptron usage, which were written in the 40s(Horton, 1941); the critical developments happened in mid-90s. Then, neural networks have never stopped evolving since.

A neural network, in general, is a technology built to simulate the activity of the human brain – precisely, pattern recognition and the passage of input through various layers of simulated neural connections.

After researches in the neural network area, studies focused on more complex structures, as stacking multiple networks. A deep neural network is a neural network with a certain level of complexity, in other words a neural network with more than two layers. Deep neural networks use sophisticated mathematical modeling to process data in complex ways.

Deep network layers can be divided into three parts. First part is our input layer. In this layer, each of our instance (each row of our dataset) is fed into this layer. The size of this layer should be equal to our feature/column count. Second part is our hidden layers. These layers create the complexity of our deep network. The size and the count of these layers are changeable and finding the best numbers changes related to the type of our deep network. These layers are responsible for calculating features and creating information from input data. And our last part is the output layer. The size of this layer is related to expected outcome of the deep learning model (Figure **3.1**).

Finding the features and the information from this data is calculated through epochs. The epoch is the hyperparameter that specifies how many times our training dataset will be fed to our model to train it.

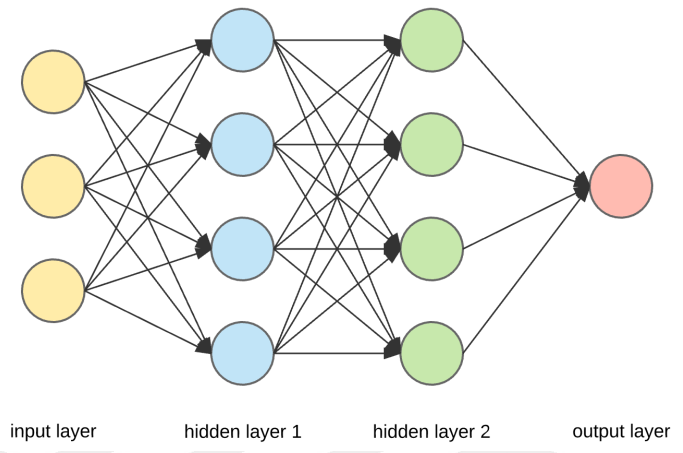


Figure **3.1**: Example of a Deep Network

3.1.2 Support Vector Machine (Binary Classification)

Support Vector Machine is described in MathWorks website as follows: "A support vector machine (SVM) is a supervised learning algorithm that can be used for binary classification or regression. A support vector machine constructs an optimal hyperplane as a decision surface such that the margin of separation between the two classes in the data is maximized. Support vectors refer to a small subset of the training observations that are used as support for the optimal location of the decision surface." (MathWorks, accessed June 2019)

3.1.3 OneClass Support Vector Machine

In Microsoft's machine learning documentation page(Microsoft, accessed June 2019), one class svm is defined as: "the support vector model is trained on data that has only one class, which is the "normal" class. It infers the properties of normal cases and from these properties can predict which examples are unlike the normal examples. This is useful for anomaly detection because the scarcity of training examples is what defines anomalies: that is, typically there are very few examples of the network intrusion, fraud, or other anomalous behavior."

The model has a learned frontier after the training which identifies the normal instance. Outside of the learned frontier is the area where anomalies are identified.(Figure **3.2**)

OneClass SVM is a model which can be trained with only using the genuine user's data. This allows us to protect the privacy of the user and helps us not to share the biometry data.

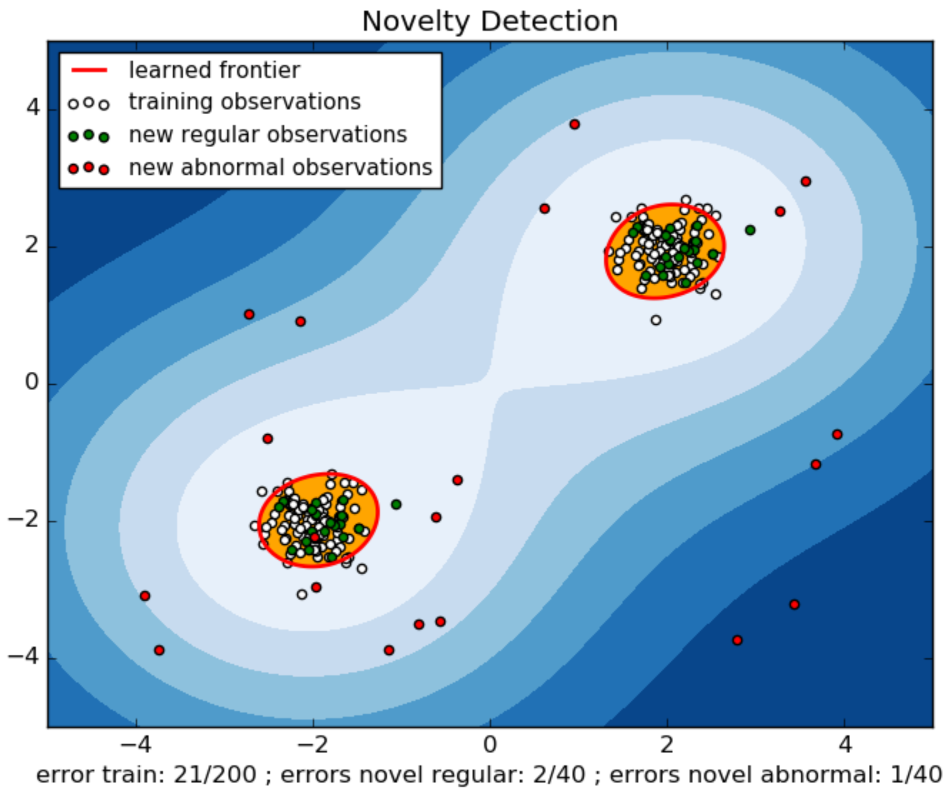


Figure **3.2**: Example of a One Class SVM Model

3.2 Data Collection Overview

Before analyzing HMOG dataset, understanding the details of HMOG dataset is crucial. In HMOG dataset, there are two types of data: sensor data and gesture data. Sensor data is the data collected from smartphone sensors like accelerometer, gyroscope, or magnetometer. Gesture data is the raw data received from user's screen interactions like touch, scroll, or double tap.

3.2.1 Sensor Data Collection

Smartphones have been evolving with increasing speed, and they are capable of understanding human activities or phone location or even how you are holding your phone. They contain many sensors, including an accelerometer for measuring acceleration, a gyroscope for measuring phone orientation, or a magnetometer for measuring the magnetic field surrounding the phone. These sensors can be analyzed and can be used to understand various information.

Motion sensors in smartphones produce nearly same amount of data in a second. This allows researchers to implement time series solutions to analyze this type of data.

In HMOG dataset, we have accelerometer, gyroscope, and magnetometer data; therefore, within the following subsections, accelerometer, gyroscope, and magnetometer are explained. The data from each sensor was collected at 16Hz.

3.2.1.1 Accelerometer

An accelerometer is a device that detects its own acceleration and is used in mobile phones to determine the phone's orientation. Once the orientation is determined, the phone's software can react accordingly, such as by changing its display from portrait to landscape. (Figure **3.3**)

There are many different ways to construct an accelerometer. Some accelerometers use the piezoelectric effect - they contain microscopic crystal structures that get stressed by accelerative forces, which causes a voltage to be generated. Another way to do it is by sensing changes in capacitance. If you have two microstructures next to each other, they have a certain capacitance between them. If an accelerative force moves one of the structures, then the capacitance will change. Add some circuitry to convert from capacitance to voltage, and you will get an accelerometer. There are even more methods, including use of the piezoresistive effect, hot air bubbles, and light.

In our research, we used accelerometer data to identify the vibrations and micromovements which are the responds to the impacts of phone usage.

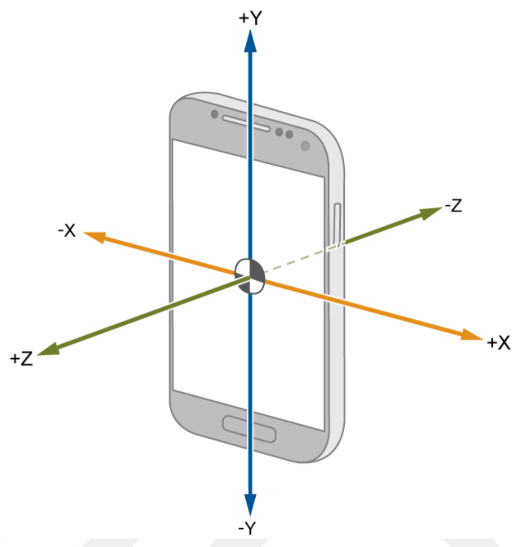


Figure **3.3**: Accelerometer

3.2.1.2 Gyroscope

A gyroscope is a device with a spinning disc or wheel mechanism that harnesses the principle of conservation of angular momentum: the tendency for the spin of a system to remain constant unless subjected to external torque. It helps to determine orientation.

The gyroscope data is used to identify the orientation of our phone and also how the user holds the phone.

3.2.1.3 Magnetometer

Magnetometers are devices that measure magnetic fields. A magnetometer is an instrument with a sensor that measures magnetic flux density B (in units of Tesla or As/m2). Magnetometers refer to sensors used for sensing magnetic fields OR to systems which measure magnetic field using one or more sensors.

3.2.2 Gesture Data Collection

Thanks to the operating systems' suitable libraries, developers can harness the data from touch interactions. This data can be raw, as X and Y coordinate of the screen or pressure or finger size, or this data can be more informative as if the interaction is a scroll or a double-tab.

These libraries do not provide the same amount of data contrary to sensor-based data. Therefore, we should apply preprocessing techniques to screen data if we want to analyze it with time series techniques. Or, we should extract the features by analyzing the data and train the machine learning / deep learning models with these extracted features. Because of the combination of those sensor-based data and screen data, the whole dataset becomes unavailable for time series methodology. There are ways to overcome this problem as using a hierarchical modeling. We could predict the classes from sensor based model and screen data based model separately and then we could make a weighted combination of the received results. But we decided to combine the datasets and go with a one multi-modal class.

From these libraries, we can log every screen interaction and then analyze this data with appropriate feature extraction methodologies.

4 METHODOLOGY

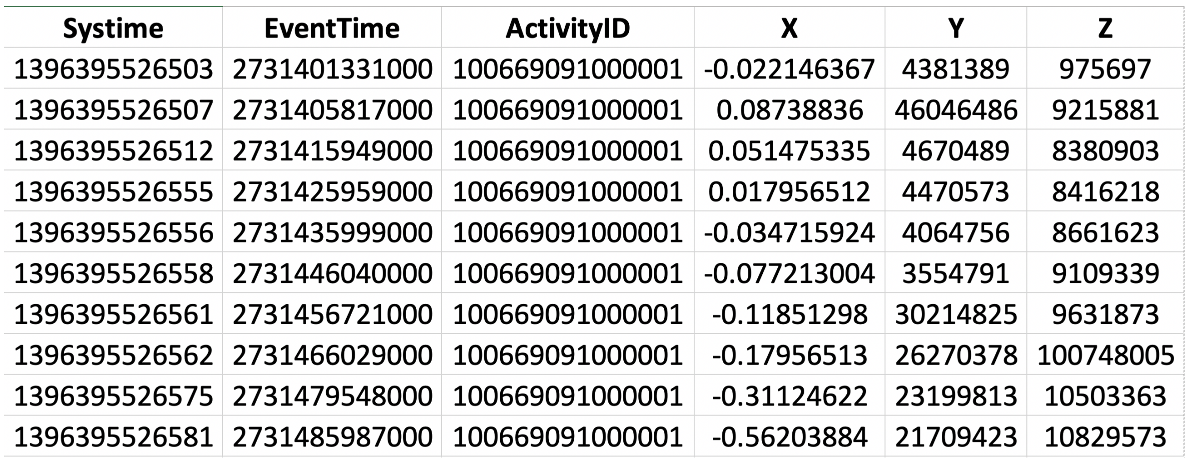
In this chapter, we describe how the HMOG dataset is formed, along with the details of how we processed the dataset, which features are used and how we created our deep network and our SVM models.

4.1 Dataset Overview

As mentioned, we utilized the HMOG dataset (Sitová et al., 2016) which is focused on continuous authentication studying hand movements, orientation and grasp. They collected a great amount of data from 100 participants during 8 free text typing sessions on Samsung Galaxy S4 phones. In this research, we used scroll event and sensor data from HMOG dataset to build a deep learning network.

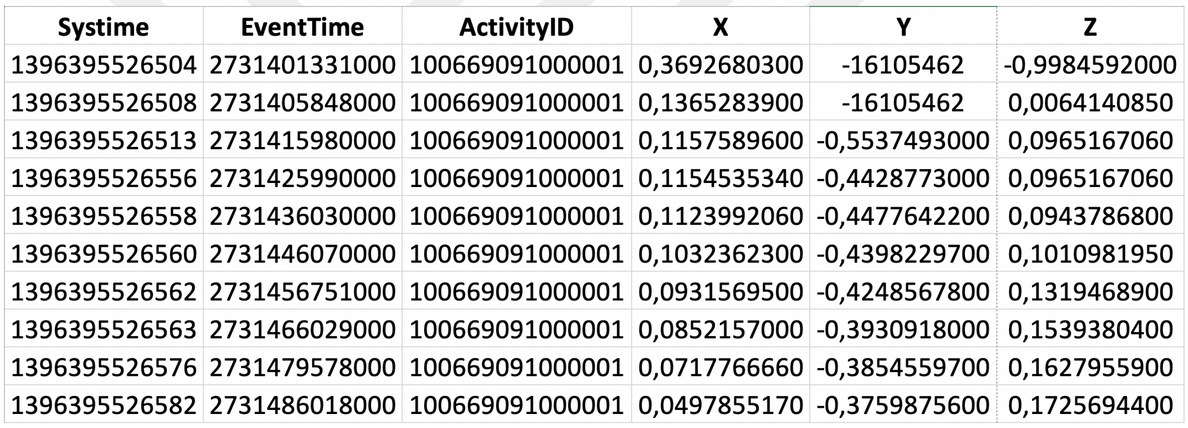
The data is sparsely diverse since the HMOG experiment was performed with 100 participants with different characteristics. When trying to track the original owner of the phone, it is crucial to have non-real user data as well if we are planning to build a multi-class model like deep network of SVM. Hence, we focus on a two-class classification problem. In HMOG paper (Sitová et al., 2016), they used one-class classification considering that it may not be possible to have non-user data. Regarding their research, they trained and tested the model only with correspondent users. Hence, they tried to resolve user authentication detecting the outliers. With this knowledge, we have studied binary classification and outlier detection.

We created a model for each user separately. To do that, we used all the user data and we randomly picked data from other participants. The reason we randomly pick data from other users and not to use all of their data is to create a model with a balanced user data. If we used all the data we had, we would have an imbalanced model. For example, we would have used 2000 instances from our legit user and 200.000 of instances from non-users. And the random pick helped our model to have a diverse user dataset.



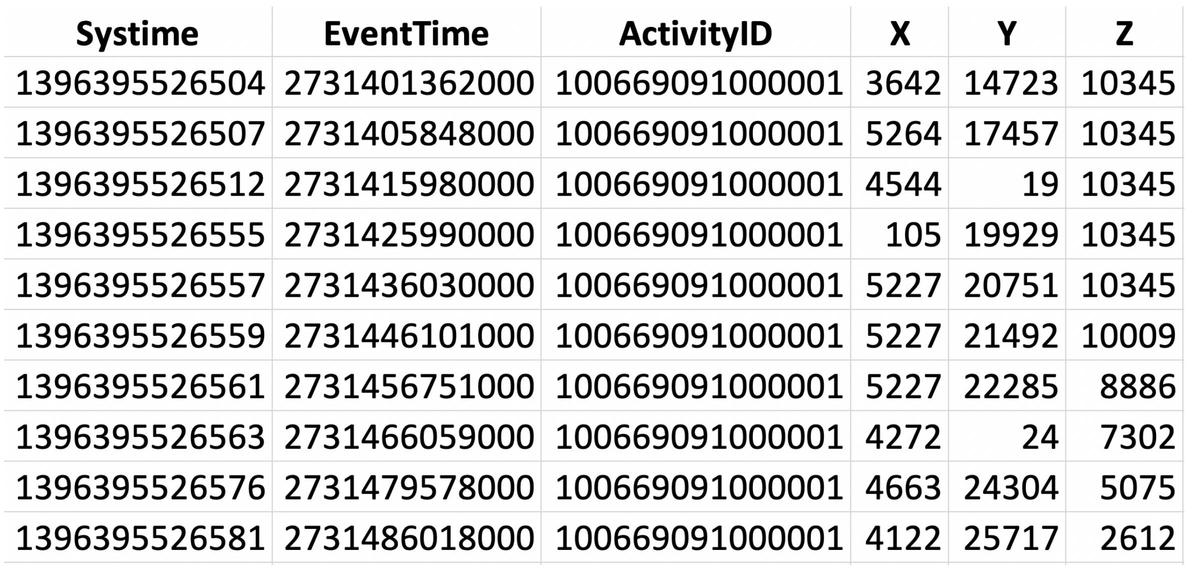
Systime	EventTime	ActivityID	X	Y	Z
1396395526503	2731401331000	100669091000001	-0.022146367	4381389	975697
1396395526507	2731405817000	100669091000001	0.08738836	46046486	9215881
1396395526512	2731415949000	100669091000001	0.051475335	4670489	8380903
1396395526555	2731425959000	100669091000001	0.017956512	4470573	8416218
1396395526556	2731435999000	100669091000001	-0.034715924	4064756	8661623
1396395526558	2731446040000	100669091000001	-0.077213004	3554791	9109339
1396395526561	2731456721000	100669091000001	-0.11851298	30214825	9631873
1396395526562	2731466029000	100669091000001	-0.17956513	26270378	100748005
1396395526575	2731479548000	100669091000001	-0.31124622	23199813	10503363
1396395526581	2731485987000	100669091000001	-0.56203884	21709423	10829573

Figure **4.1**: Accelerometer Data Example



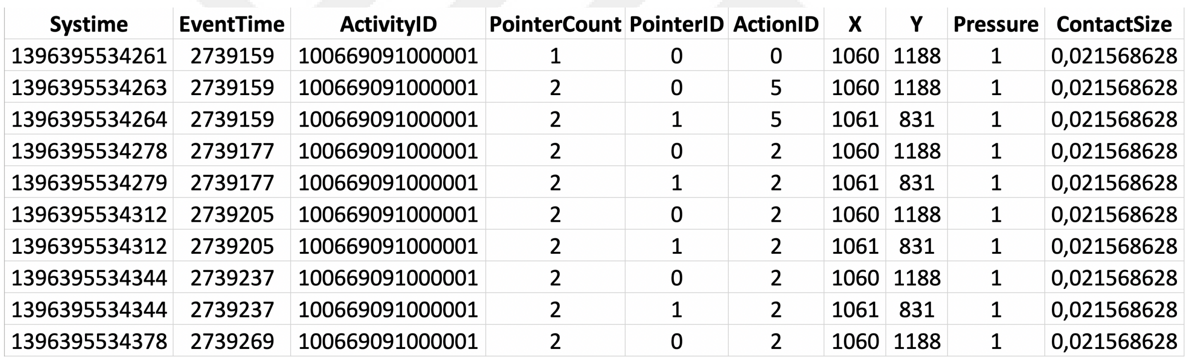
Systime	EventTime	ActivityID	X	Y	Z
1396395526504	2731401331000	100669091000001	0,3692680300	-16105462	-0,9984592000
1396395526508	2731405848000	100669091000001	0,1365283900	-16105462	0,0064140850
1396395526513	2731415980000	100669091000001	0,1157589600	-0,5537493000	0,0965167060
1396395526556	2731425990000	100669091000001	0,1154535340	-0,4428773000	0,0965167060
1396395526558	2731436030000	100669091000001	0,1123992060	-0,4477642200	0,0943786800
1396395526560	2731446070000	100669091000001	0,1032362300	-0,4398229700	0,1010981950
1396395526562	2731456751000	100669091000001	0,0931569500	-0,4248567800	0,1319468900
1396395526563	2731466029000	100669091000001	0,0852157000	-0,3930918000	0,1539380400
1396395526576	2731479578000	100669091000001	0,0717766660	-0,3854559700	0,1627955900
1396395526582	2731486018000	100669091000001	0,0497855170	-0,3759875600	0,1725694400

Figure **4.2**: Gyroscope Data Example



Systime	EventTime	ActivityID	X	Y	Z
1396395526504	2731401362000	100669091000001	3642	14723	10345
1396395526507	2731405848000	100669091000001	5264	17457	10345
1396395526512	2731415980000	100669091000001	4544	19	10345
1396395526555	2731425990000	100669091000001	105	19929	10345
1396395526557	2731436030000	100669091000001	5227	20751	10345
1396395526559	2731446101000	100669091000001	5227	21492	10009
1396395526561	2731456751000	100669091000001	5227	22285	8886
1396395526563	2731466059000	100669091000001	4272	24	7302
1396395526576	2731479578000	100669091000001	4663	24304	5075
1396395526581	2731486018000	100669091000001	4122	25717	2612

Figure **4.3**: Magnetometer Data Example



Systime	EventTime	ActivityID	PointerCount	PointerID	ActionID	X	Y	Pressure	ContactSize
1396395534261	2739159	100669091000001	1	0	0	1060	1188	1	0,021568628
1396395534263	2739159	100669091000001	2	0	5	1060	1188	1	0,021568628
1396395534264	2739159	100669091000001	2	1	5	1061	831	1	0,021568628
1396395534278	2739177	100669091000001	2	0	2	1060	1188	1	0,021568628
1396395534279	2739177	100669091000001	2	1	2	1061	831	1	0,021568628
1396395534312	2739205	100669091000001	2	0	2	1060	1188	1	0,021568628
1396395534312	2739205	100669091000001	2	1	2	1061	831	1	0,021568628
1396395534344	2739237	100669091000001	2	0	2	1060	1188	1	0,021568628
1396395534344	2739237	100669091000001	2	1	2	1061	831	1	0,021568628
1396395534378	2739269	100669091000001	2	0	2	1060	1188	1	0,021568628

Figure **4.4**: Touch Event Data Example

4.2 Data Preprocessing

HMOG dataset contains 24 sessions for each user. Each of these sessions includes sensor-based data and touch screen events, so they are time series data by nature. We need to preprocess data to use machine learning models. LSTM network can work with time series data, but in this study, we studied the dataset with features extracted as structured data.

For each user session, sensor data and scroll (touch) data is merged to understand phone micro-movements for each scroll event. This merging helped us to create a table with scroll data and their sensor reflections. To be able to merge them into one dataframe, we wrote a python code which looks for the scroll data. This code finds a scroll event, takes the start and the end time of found event and locates the sensor data which has happened between this gap by looking in their appropriate csv file. With this script, we extracted all the sensor data which is related to scroll events. This process was applied to all three sensor files: accelerometer, gyroscope and magnetometer.

4.2.1 Feature Extraction

When working in a machine learning process, feature extraction is a crucial step to identify information of data. With feature extraction, we convert a time series data to a structured dataset on which our model can find relations between those features. We apply feature extraction to be able to use this dataset because the data which is produced on each scroll action has different shape. Then these features become the identifiers of our classes. The sensor data; accelerometer, gyroscope, and magnetometer, are provided in time series vector so we should convert every time series to an action which becomes our input data.

For calculation of our features from raw sensor data, we used median, mean, and standard deviation. One may argue that, extraction of features is unnecessary when working with deep learning algorithms. Since we are interested in developing a real-time continuous authentication scheme, rather than transmitting raw data to a server where machine learning algorithm runs, we export features as the summary of the data to reduce the data size to be transferred and also again, this type of data is not suitable for a time series model and we also need features for SVM models.

Then scroll or touch-screen features are calculated using the features reported in Touchalytics study(Frank et al., 2013).

Sensor	Raw Data	Feature
Scroll	X, Y axis	First Value Last Value Max Deviation %20 Deviation %50 Deviation %80 Deviation
Scroll	Velocity	Pairwise 20 Pairwise 50 Pairwise 80 Median of the last three points Average
Scroll	Acceleration	Pairwise 20 Pairwise 50 Pairwise 80 Median of the first five points Average
Scroll	Trajectory	Length Distance Duration
Scroll	Finger Size, Pressure	Median
Sensor	X, Y, Z axis (for each sensor: accelerometer, gyroscope, magnetometer and their magnitude)	Mean Standard Deviation Median

Table**4.1**: Summarized Feature Table

After the feature extraction, our dataset resulted in 83 features and one label which implies if the user is genuine or not. We can summarize those features as follows in Table **5.4**, represent them as Figures **4.5 4.6 4.7 4.8**

4.2.2 Data Cleaning

Since we have structured numeric data, edge numbers as infinite or not numbers (NaN) can cause problems. Those cases were actually observed through the dataset. As tested with those cases, the deep network failed to learn, and model training step failed. So, data cleaning was a critical step. For any model, cleaning infinite numbers and N/A's are a really significant task that should not be overlooked.

After cleaning the data, the next step of data preparation is normalizing the data. Normalization of data can be crucial sometimes as we are handling a numerical dataset.

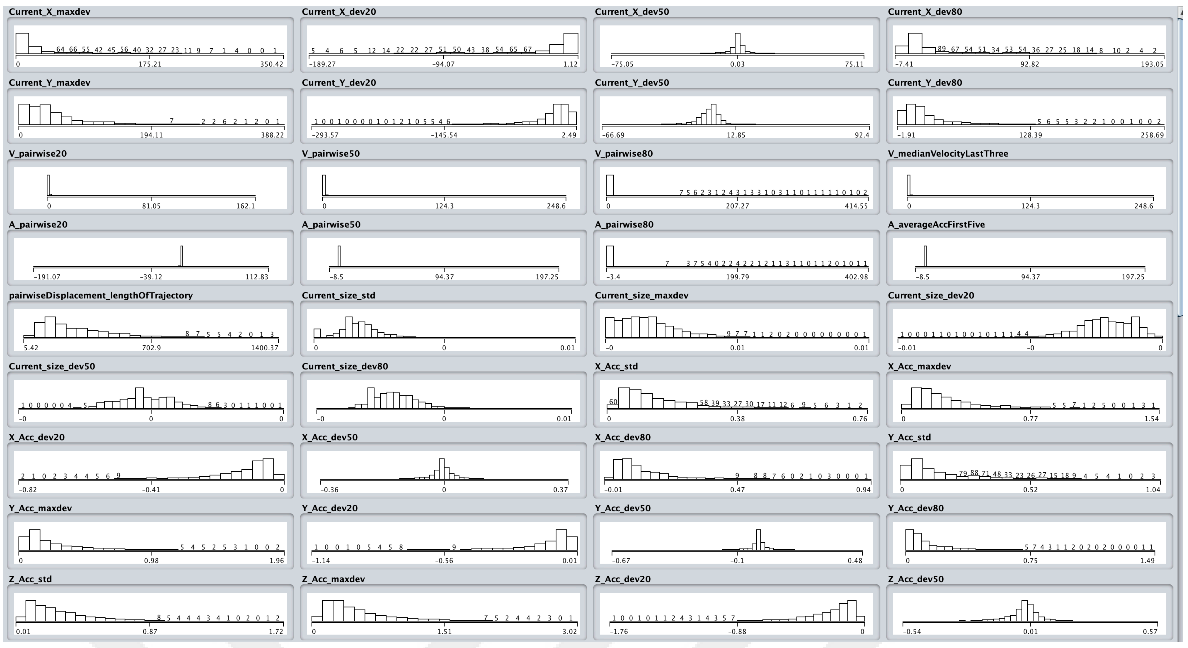


Figure **4.5**: SVM Features 1

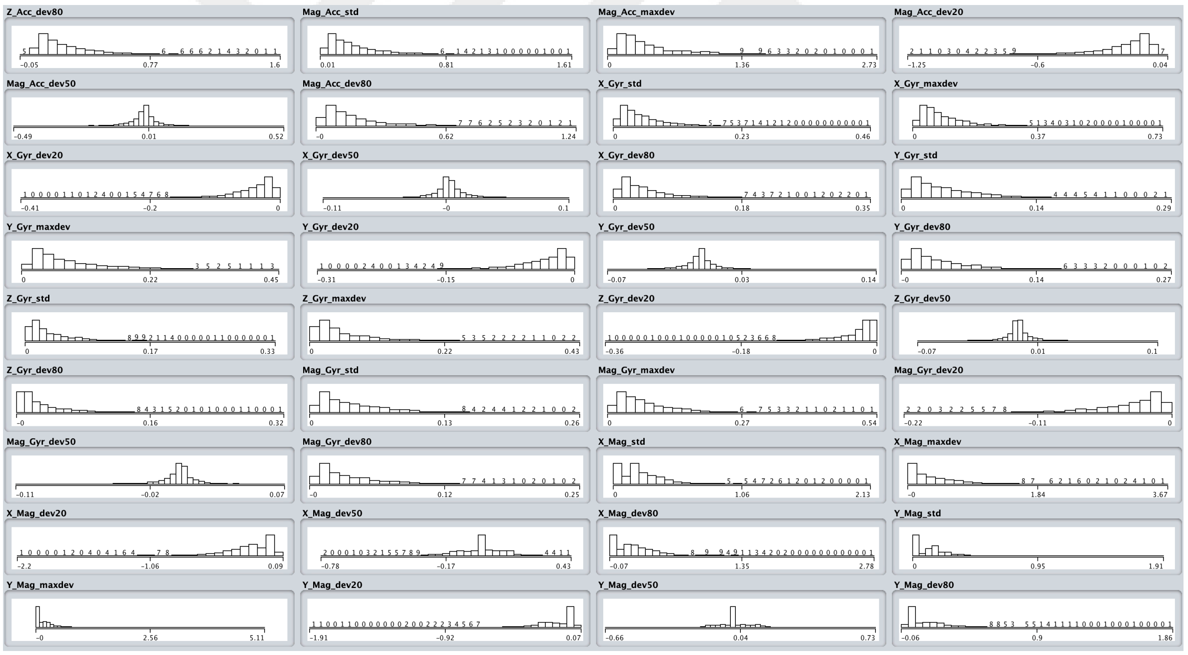


Figure **4.6**: SVM Features 2

If the difference between data in feature sets is more than 3 or 4 decimal points, this can affect the overall accuracy. As a result of our tests with normalized and un-normalized datasets, we observed that overall accuracy can differ up to 3%. In our research, we have used min-max normalization and the data is normalized between 0 and 1.

Since we have a significant number of experimenters and we used the leave-one-out method to create our datasets for each user, the number of instances from non-genuine users can be more than the number of instances of real-user. For that reason, we sampled

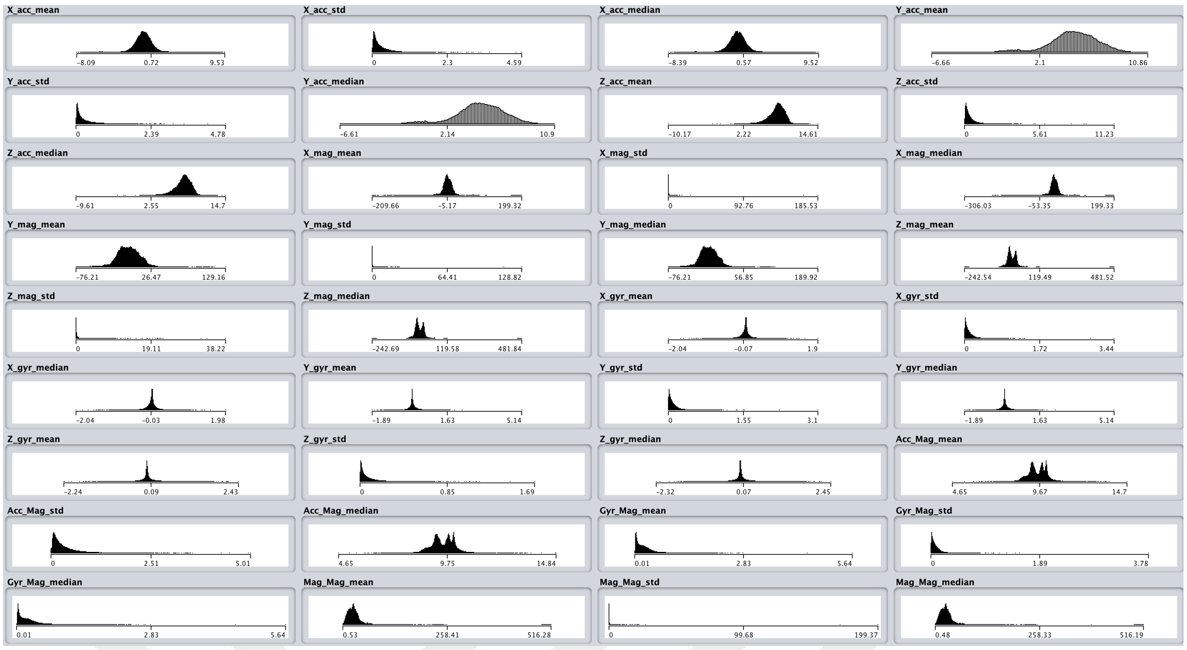


Figure **4.7**: Deep Learning Features 1

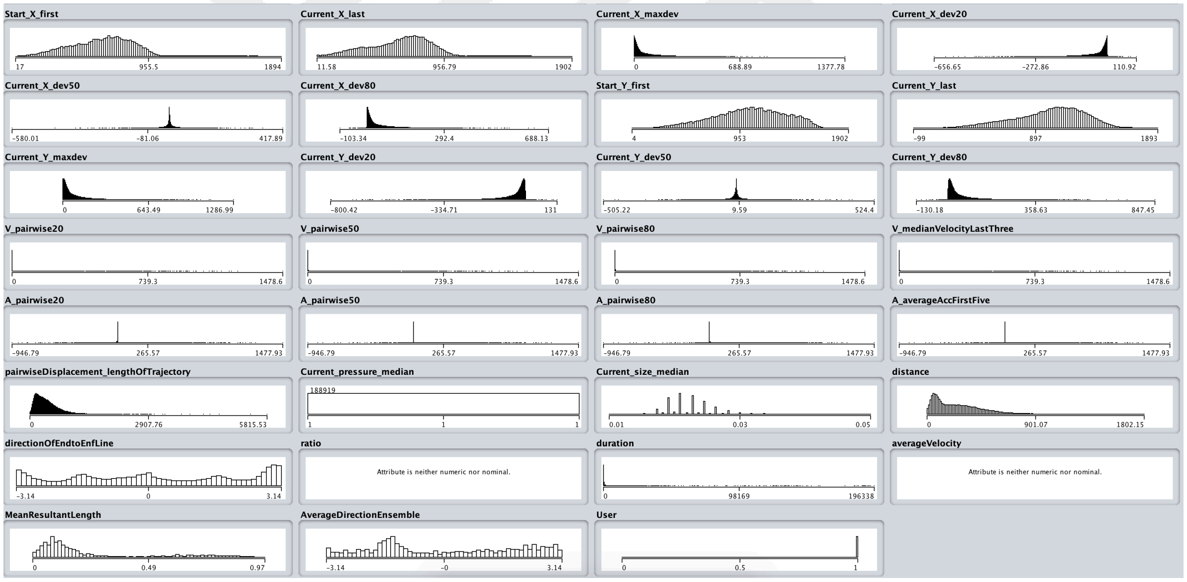


Figure **4.8**: Deep Learning Features 2

non-genuine users to reduce the number in the dataset and sampled them randomly to not to lose the generalization of non-users' patterns. After the sampling, each dataset has approximately ten thousands of row. The ratio of genuine users to non-genuine ones is 1 to 3.

After cleaning the data, PCA could be applied. As we see in HMOG paper (Sitová et al., 2016), PCA is used in biometrics. However, this model is intended to be used in a mobile application and the less computation we do, the faster we have our result. So we decided not to use PCA to reduce computation complexity.

4.3 Classifier Evaluation

To train our deep learning network, validate and test our models, we created three different sub-datasets for each dataset. Since we had approximately ten thousands of rows, we did not want to lower the number of rows in the training subsets. That is why as a first step, we sampled our dataset with 10% to create our validation subset which will validate our trained model in every epoch. Then we sampled our remaining subset with 10% again to create our test subset to evaluate the performance of the created model after training phase.

To train and test OneClass SVM, we trained with all of the data. We also tested our oneclass SVM model with the other users' data to test if our model can predict them as outliers.

Also we used %80 to %20 train-test split for our binary SVM classification.

4.4 Classification and Performance Metrics

In this study, we used a deep learning network to identify user actions. Deep learning and neural networks are generally used with image, audio and text datasets but recently they are also used in structured data. HMOG study used one classifier SVM to find outliners so we also wanted to see how our features would work with one-class svm; additionally, we used a binary classification to make our predictions with a deep learning network and with an SVM model.

As the initial investigation, we aimed to explore how a primary deep learning network would give a model, that is why our we have created our model with three dense layers. After some shallow tests, we decided to train our every data with two different networks which have 64 and 128 nodes in each layer. Then we batched our data as tensors with shape (8192, feature_count) and ran the network with 200 epochs.

We collected accuracy, mean absolute error and mean square error for train and validation test results and f1, accuracy, recall, and precision measurements and also true positive, false positive, true negative and false negative counts for test predictions. Then we calculated false acceptance rate and false rejection rate, and equal error rate by calculating their mean. These metrics are commonly used in the literature.

Then we investigated how a binary-classification with SVM will work. We applied a parameter tuning to find the best parameters for train and test data for each user. We also calculated F1, accuracy, recall and precision for train and test results.

Finally we created our one-class SVM model to build an outlier detector model. We again implemented a parameter tuning for the model to find the best parameters for train and test data. Then we calculated positive and negative predictions.

$$ACCURACY = \frac{TP + FP}{TP + FP + TN + FN} \tag{4.1}$$

$$PRECISION = \frac{TP}{TP + FP} \tag{4.2}$$

$$RECALL = \frac{TP}{TP + FN} \tag{4.3}$$

$$F1 = 2 * \frac{precision * recall}{precision + recall} \tag{4.4}$$

$$FAR = \frac{FP}{FP + TN} \tag{4.5}$$

$$FRR = \frac{FN}{FN + TP} \tag{4.6}$$

$$MAE = \frac{1}{n}\sum_{i=1}^{n} |x_i - x| \tag{4.7}$$

$$MSE = \frac{1}{n}\sum_{i=1}^{n} (x_i - x)^2 \tag{4.8}$$

5 PERFORMANCE EVALUATION

In this chapter, we explain how our prediction models were created and how we evaluated them. We explain our deep learning model and its results with respect to different neuron counts. Then we explain our binary svm model which is acting as an intermediary model between the deep network and the one class svm model. Finally we explain our oneclass svm model and its results.

After the evaluation of those three models, we will compare those models in the next Chapter 5.

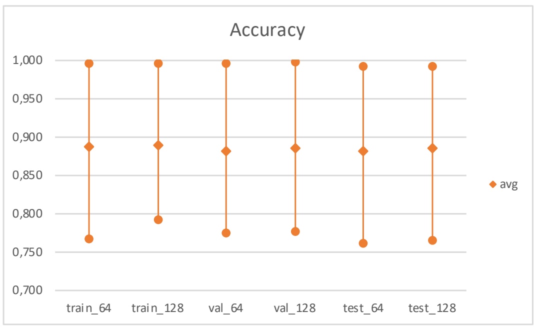
5.1 Deep Network

In this section, we present and explain the results obtained from our testing process. As mentioned in Chapter 4, we divided our dataset into four datasets which are created by only scroll (touch-screen) features, accelerometer plus scroll features, gyroscope plus scroll features, and all sensors plus scroll features. As mentioned, we build our deep network with three layers and we created the layers with 64 and 128 nodes.

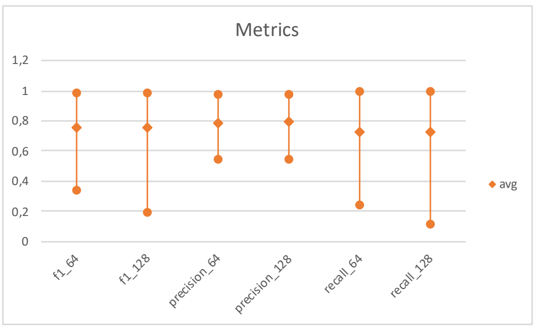
In the following, for each figure we display results for four different datasets, and we have four figures displaying test accuracy, test precision, test EER and test F1 metric. For each dataset, we also present the results for 64 nodes and 128 nodes. Although we present the average results, because of the diversity of user characteristics, we also discuss the minimum and the maximum values of each metric in the following subsections where we analyze the results for each feature-specific dataset and evaluate the model performance.

5.1.1 Scroll Features

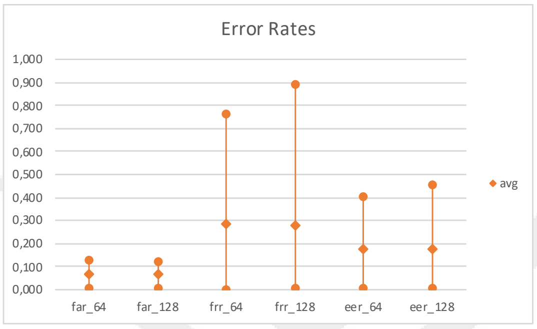
In this section, we only used the scroll features which are discussed in feature extraction Section 4.2.1 to train our model. For each user, these features are calculated. Then those features are fed to the deep network as inputs and trained for 200 epochs. As mentioned, for each user we have data coming from the user as well as from other users, sampled



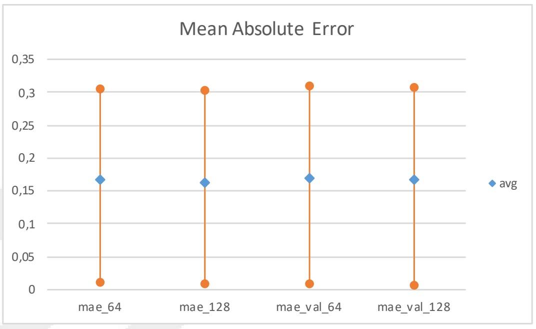
(a) Accuracy



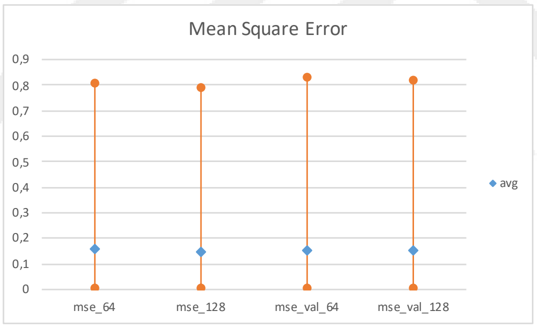
(b) Measures



(c) EER



(d) Mean Absolute Error



(e) Mean Square Error

Figure **5.1**: Only Scroll Data Features

randomly.

In Figure **5.1(c)**, we can see the results for EER metric on the test data. When we first analyze the average results, we can see that considering the two deep networks (regarding their node counts), the results are nearly the same, except the network with 128 nodes is slightly low. For performance requirements, it may be better to work with networks with smaller size. Also when we compare these results with HMOG dataset, the 128 nodes network has given similar results with their results where they used only HMOG features; on the contrary when we look at our smallest EER result which is 0.5%, only using the scroll features revealed better than similar works.

In Figure **5.1(a)**, we can see the accuracy results of test data. The test accuracy of our

Features	64	128
Accuracy		
min_train	0.767	0.793
train	0.886	0.889
max_train	0.996	0.996
min_val	0.776	0.777
val	0.882	0.884
max_val	0.996	0.997
min_test	0.761	0.764
test	0.881	0.884
max_test	0.992	0.993
Errors		
min_tr_mse	0.004	0.004
avg_tr_mse	0.081	0.079
max_tr_mse	0.153	0.145
min_tr_mae	0.011	0.009
avg_tr_mae	0.166	0.162
max_tr_mae	0.305	0.303
min_val_mse	0.003	0.002
avg_val_mse	0.083	0.082
max_val_mse	0.149	0.152
min_val_mae	0.009	0.007
avg_val_mae	0.169	0.165
max_val_mae	0.309	0.307
Metrics		
min_f1	0.344	0.190
avg_f1	0.746	0.750
max_f1	0.985	0.987
min_precision	0.547	0.549
avg_precision	0.784	0.791
max_precision	0.978	0.978
min_recall	0.239	0.110
avg_recall	0.720	0.724
max_recall	1.000	0.996
Rates		
min_far	0.008	0.008
avg_far	0.064	0.062
max_far	0.126	0.123
min_frr	0.000	0.004
avg_frr	0.280	0.276
max_frr	0.761	0.890
min_eer	0.005	0.006
avg_eer	0.172	0.169
max_eer	0.406	0.453

Table**5.1**: All Results of Only Scroll Feature Set

model which is created with only scroll features, is between 76% and 99%. Average of test accuracy is 88%.

Since we are studying continuous authentication, one of the most important metrics is the precision. Precision is the ratio of true positives to all true predictions labeled as true, including false positives. In an authentication solution, not authenticating the user is not crucial but authenticating the malicious user is a serious problem. That is the reason, in our results, the precision metric is as essential as the EER metric.

In Figure**5.1(b)**, we can see precision results of our test data. When we analyze the precision metric for scroll data training, we can see the values vary between 58% and 98% and with an average of 78-79%. With the best result, we authenticate only two persons out of 100 incorrectly. This result shows us, with enough data, we can keep precision as high as we would need in an authentication scenario.

5.1.2 Accelerometer & Scroll Features

First of all, we analyze the average results of EER again. With this dataset, we can see that network with 128 nodes has lower EER values comparing to the networks with 64 nodes. With this dataset, change in node count affects the result significantly.

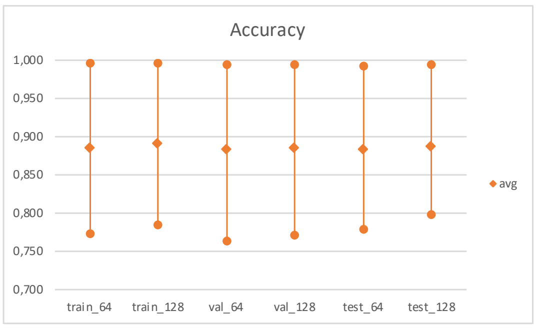
The precision for this model is between 57% and 99%, and the average precision value is 79%. If we compare this model with the model created with only scroll features, the results are very similar, but we see that the accelerometer data lowered precision by 2%.

The accuracy of this model is between 78% and 99% and the average of it is 89%. The node count change affects the accuracy slightly. Regarding these first results, for a network with three dense layers, accelerometer data makes a very slight change, 0.1% for accuracy. For other metrics, the change is approximately 0.01% with the network with 128 nodes.

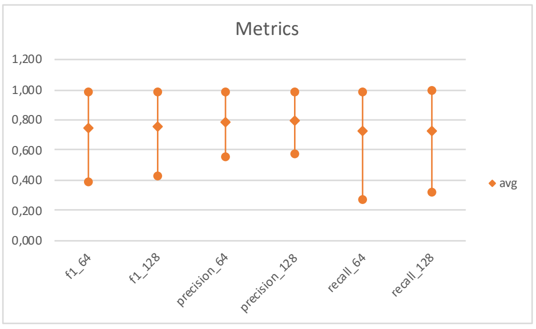
5.1.3 Gyroscope & Scroll Features

We start by analyzing the average results of EER. We can see that network with 128 nodes has the lowest EER values comparing to the other networks. With this dataset, the network with the 128 nodes achieved the best result.

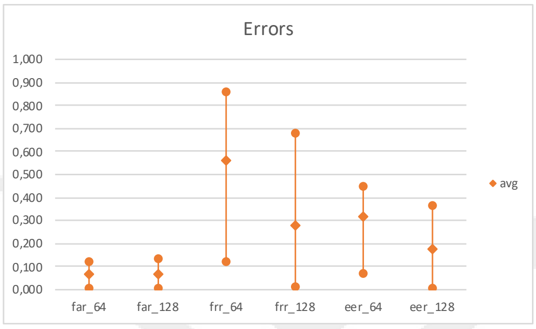
The precision for this model is between 59% and 99%, and the average precision value



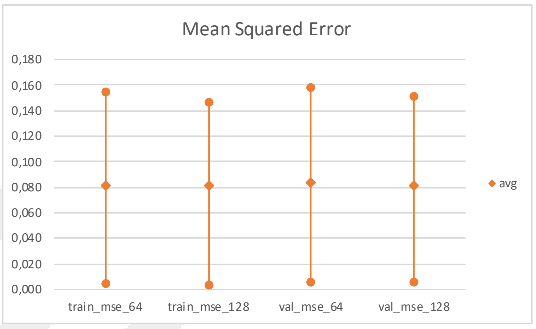
(a) Accuracy



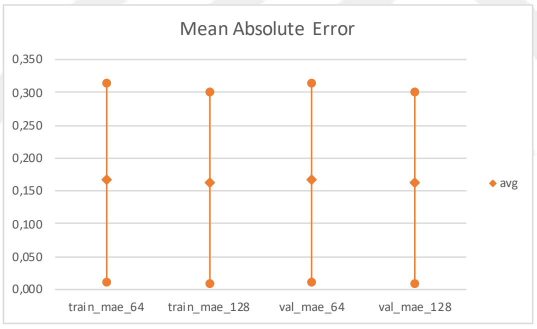
(b) Measures



(c) EER



(d) MSE



(e) MAE

Figure **5.2**: Accelerometer + Scroll Data Features

is 80-81%. If we compare this model with the other two models that we discussed, this model exhibited a result approximately 3 points higher comparing to the other models.

The accuracy of this model is between 78% and 99.7% and the average is 89-90%. The node count change affects the accuracy slightly. But again, we have the best average result compared to the other networks.

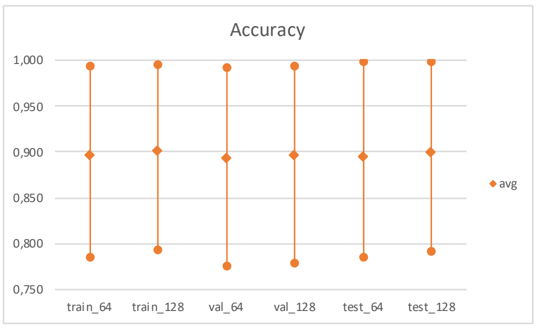
5.1.4 All Sensor Features & Scroll Features

When we look at the results of the model which we trained with all the features, we can see that the results did not increase. We can see that network with 128 nodes still

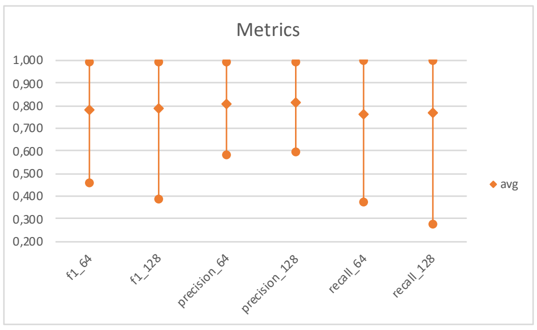
Features	64	128
min_train	0.773	0.785
train	0.885	0.889
max_train	0.995	0.996
min_val	0.764	0.771
val	0.882	0.885
max_val	0.994	0.994
min_test	0.778	0.798
test	0.882	0.886
max_test	0.993	0.994
min_train_mse	0.004	0.003
avg_train_mse	0.081	0.080
max_train_mse	0.154	0.147
min_train_mae	0.011	0.008
avg_train_mae	0.167	0.161
max_train_mae	0.314	0.301
min_val_mse	0.006	0.006
avg_val_mse	0.083	0.080
max_val_mse	0.158	0.151
min_val_mae	0.014	0.012
avg_val_mae	0.168	0.163
max_val_mae	0.318	0.307
min_f1	0.390	0.426
avg_f1	0.742	0.750
max_f1	0.986	0.988
min_precision	0.557	0.570
avg_precision	0.777	0.788
max_precision	0.986	0.986
min_recall	0.271	0.322
avg_recall	0.718	0.722
max_recall	0.990	0.991
min_far	0.004	0.004
avg_far	0.064	0.061
max_far	0.120	0.136
min_frr	0.118	0.009
avg_frr	0.558	0.278
max_frr	0.860	0.678
min_eer	0.071	0.007
avg_eer	0.311	0.169
max_eer	0.451	0.366

Table**5.2**: All Results of Accelerometer + Scroll Feature Set

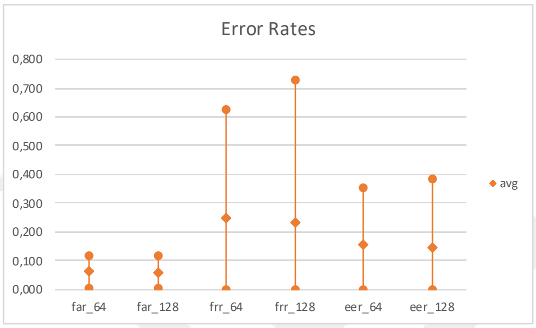
has the lower EER value compared to the model with 64 nodes. We have the minimum EER as 0.4% and the maximum EER as 36%, and the average is 17%. The precision for this model is between 53-58% and 99%, and the average precision value is 78%. The accuracy of this model is between 76% and 99.7% and the average of it is 88%.



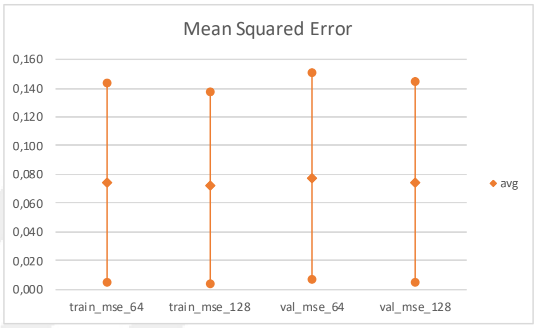
(a) Accuracy



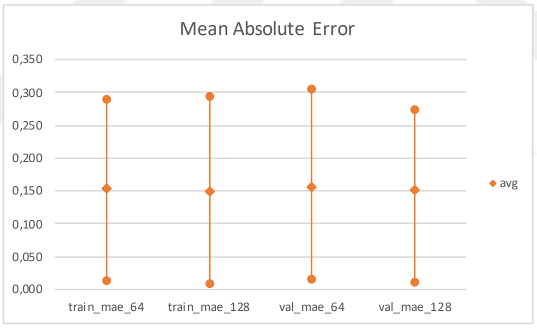
(b) Measures



(c) EER



(d) MSE



(e) MAE

Figure **5.3**: Gyroscope + Scroll Data Features

5.1.5 Comparison of All Datasets

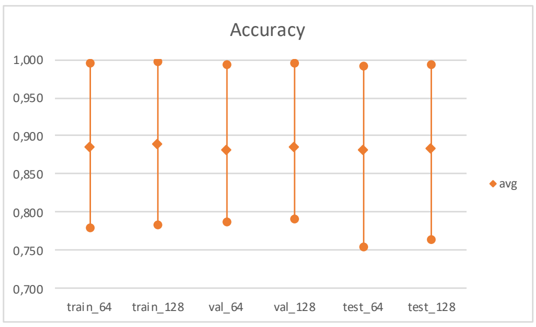
By analyzing these first results with a basic deep network with three layers, genuine we can say that using more features not mean better results. In other words we tried to feed all the features and data without feature selection algorithms. With these comparisons and the HMOG dataset, a deep network provided a better performance with gyroscope and scroll features.

As a result of the evaluations and demonstration of the results, we observe that the dataset which has the gyroscope features and the scroll features performed better than the other datasets.

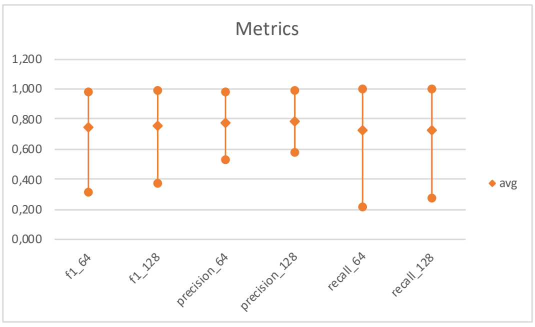
Feature	64	128
min_train	0.785	0.793
train	0.897	0.793
max_train	0.994	0.996
min_val	0.776	0.779
val	0.893	0.896
max_val	0.992	0.993
min_test	0.784	0.792
test	0.894	0.899
max_test	0.899	0.899
min_train_mse	0.005	0.004
avg_train_mse	0.074	0.071
max_train_mse	0.144	0.137
min_train_mae	0.013	0.009
avg_train_mae	0.153	0.147
max_train_mae	0.291	0.293
min_val_mse	0.007	0.005
avg_val_mse	0.076	0.074
max_val_mse	0.151	0.145
min_val_mae	0.015	0.011
avg_val_mae	0.155	0.149
max_val_mae	0.305	0.274
min_f1	0.457	0.389
avg_f1	0.777	0.786
max_f1	0.995	0.995
min_precision	0.584	0.595
avg_precision	0.802	0.812
max_precision	0.991	0.991
min_recall	0.376	0.274
avg_recall	0.757	0.769
max_recall	1.000	1.000
min_far	0.003	0.003
avg_far	0.060	0.058
max_far	0.116	0.116
min_frr	0.000	0.000
avg_frr	0.243	0.231
max_frr	0.624	0.726
min_eer	0.002	0.002
avg_eer	0.152	0.145
max_eer	0.355	0.384

Table**5.3**: All Results of Gyrosope + Scroll Feature Set

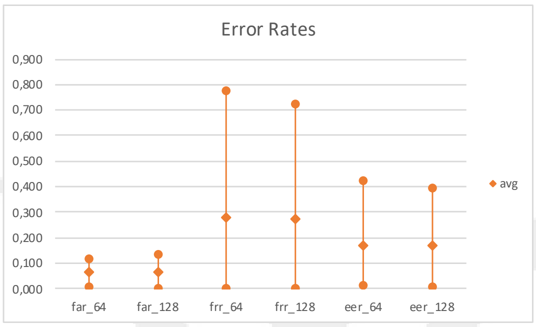
However, when we analyze the results one by one, we can see that in all feature-split datasets, there are users who performed with 99% accuracy and 98-99% precision, and nearly 0% EER. Since we aim to create a passive continuous authentication system using user behaviors, more data means better recognition. In HMOG study (Sitová et al., 2016), they removed 20 persons because of their low quality data; on the contrary,



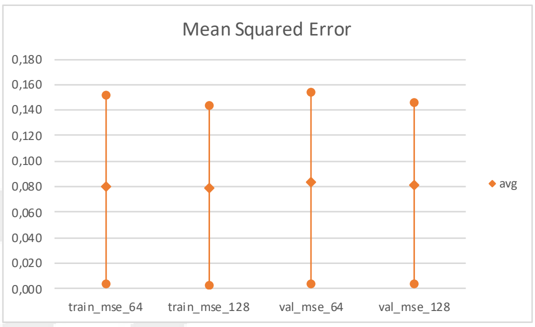
(a) Accuracy



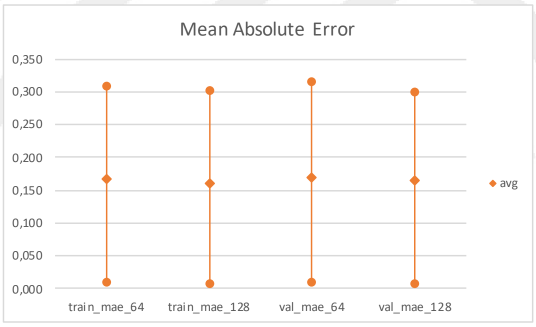
(b) Measures



(c) EER



(d) MSE



(e) MAE

Figure **5.4**: All Data Features

we wanted to use these persons' data to create a more realistic solution.

Features	64	128
min_train	0.779	0.784
train	0.886	0.889
max_train	0.995	0.996
min_val	0.788	0.791
val	0.882	0.885
max_val	0.994	0.996
min_test	0.755	0.763
test	0.881	0.884
max_test	0.991	0.993
min_train_mse	0.004	0.003
avg_train_mse	0.081	0.079
max_train_mse	0.152	0.144
min_train_mae	0.010	0.007
avg_train_mae	0.166	0.161
max_train_mae	0.309	0.301
min_val_mse	0.004	0.004
avg_val_mse	0.083	0.082
max_val_mse	0.154	0.146
min_val_mae	0.010	0.008
avg_val_mae	0.169	0.164
max_val_mae	0.316	0.300
min_f1	0.312	0.373
avg_f1	0.746	0.751
max_f1	0.982	0.986
min_precision	0.532	0.578
avg_precision	0.779	0.787
max_precision	0.982	0.986
min_recall	0.221	0.274
avg_recall	0.723	0.725
max_recall	0.995	0.995
min_far	0.006	0.004
avg_far	0.066	0.063
max_far	0.117	0.135
min_frr	0.005	0.005
avg_frr	0.277	0.275
max_frr	0.779	0.726
min_eer	0.012	0.009
avg_eer	0.172	0.169
max_eer	0.421	0.395

Table**5.4**: All Results of All Feature Set

5.2 Binary SVM

In this section, we will explain how we created the dataset for this methodology, and how we created our results.

First of all, we have trained the model with the data which we have trained our deep learning network. With this decision, we tested if the same dataset would work with a machine learning model. The answer is no. The first results that we received from SVM model was around %20, and this result did belong to train data.

After observing this, we were sure that a deep learning algorithm does not work like a machine learning algorithm. Even our previous dataset did not have perfect features, our deep network has given promising results. With this knowledge, we analyzed our dataset and tried to find the features which do not increase the overall success. After the analyzing process, we have changed our feature set as follows:

- Accelerometer, Gyroscope, Magnetometer and magnitude of these sensors:
  - Standard Deviation (STD)
  - Max. STD
  - %20 STD
  - %50 STD
  - %80 STD
- Screen Features:
  - Axis Values (X, Y) and Current_Size:
    - Standard Deviation (STD)
    - Max. STD
    - %20 STD
    - %50 STD
    - %80 STD
  - Vector and Acceleration:
    - Pairwise 20
    - Pairwise 50
    - Pairwise 80
- And other calculated features.

Here, we can see that in this dataset, sensor features are changed. In our deep learning dataset, we have used mean and median but in this dataset, we have only used standard deviation for sensor features, and this has improved the overall results.

We have also applied one versus all for all 100 users to train each of them individually. Each user has average of 1600 rows of data, that means when we apply one for all methodology, non-genuine-user data length will be much more bigger than our genuine-user data. That's why we sampled non-genuine-user data for each user to have equal number of instances for each label.

When we are training our SVM model, we can tune parameters to obtain the best result. For each user, we experimented with two parameters: C, and Gamma. Also we only trained our data with 'RBF' kernel. Kernel parameter is deciding the type of the hyperplane to separate the data. Linear kernels provide faster training times, but non-linear kernels like RBF can provide flexibility to the model.

The C parameter is the penalty parameter for the error term. It is the optimization parameter for misclassifying the training data. Larger the C means that our model tends to overfit. The Gamma parameter is the parameter for the hyperplane too. The bigger the gamma is, plane fits more accurately to the training data.

We tested our models with different parameters, then we have chosen the best parameters for each user, by finding the best f1 scores for testing and training results. The results are very promising as we have seen %99.77 as minimum of f1 score. You can see the results for all sensors and screen features trained with binary SVM model in Figure **5.5**. This figure represents the best F1 scores for each user. Blue line is representing the our train results and the lowest F1 score received is 99.89%. But training results can also mean overfitting, so test results are also have importance. In the figure, the orange line is representing our test results. We can clearly see that the lowest result that we receive from a user is 99.77% for F1 score. With some users, we received 100% F1 score for both train and test results and you can see the detailed results in the Figure **APPENDIX .1**

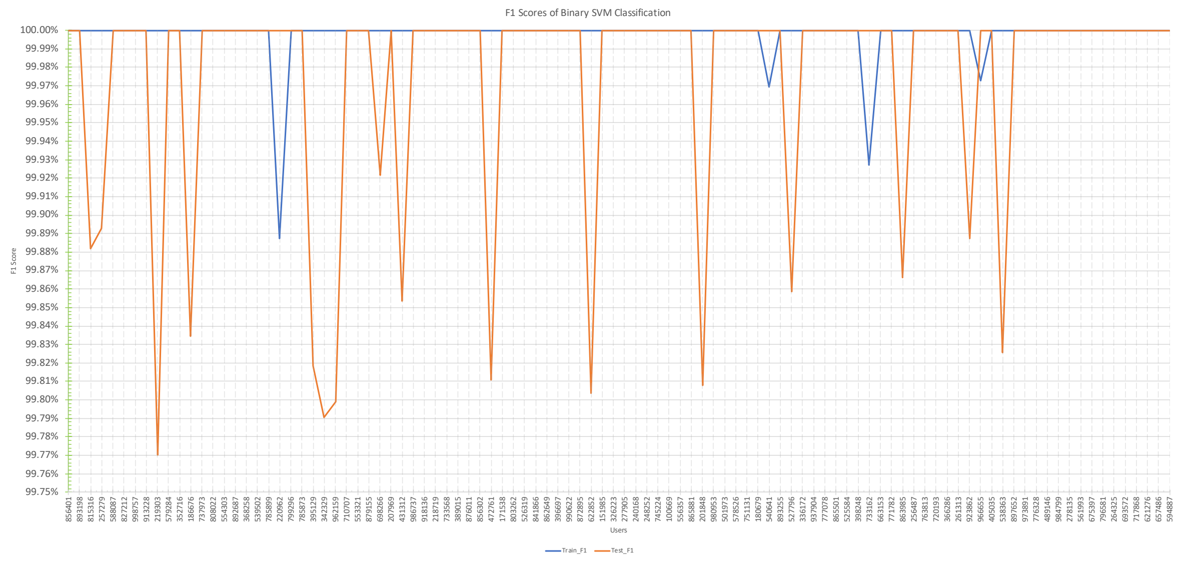


Figure **5.5**: Best F1 Scores for Binary SVM Classification

5.3 OneClass SVM

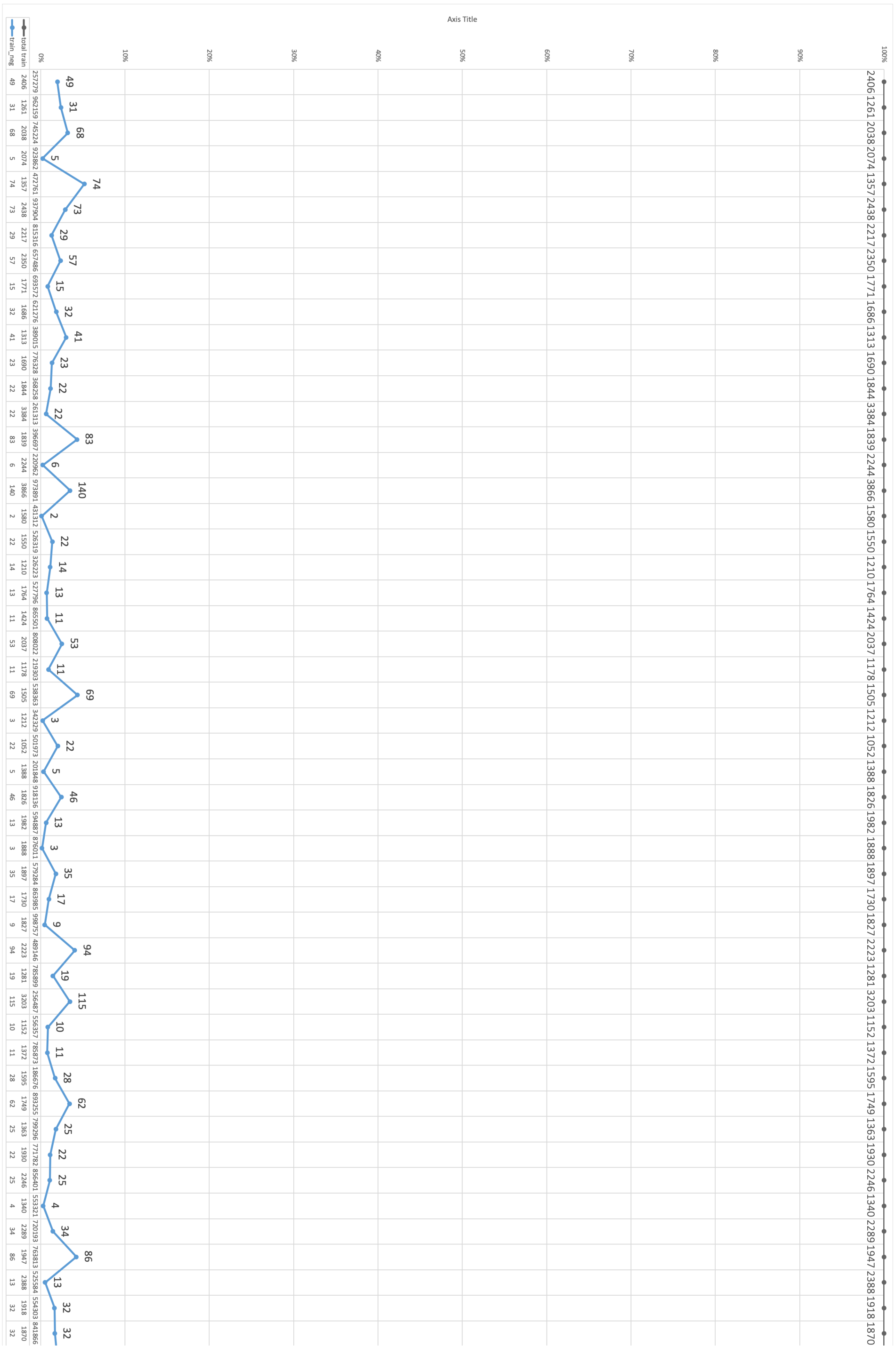
OneClass SVM is a popular classification method for biometric researches in academia.(Jia et al., 2014)(Hejazi et al., 2016)(Bergamini et al., 2009)(Sitová et al., 2016)(Buriro et al., 2016) Biometric data is a data type whom privacy is crucial. According to privacy regulations, biometric data should be kept secretly and in a protected area. According to this, sharing some other users biometric data with a user is against this rule. That is why creating prediction models with one labeled dataset can provide a more secure way.

OneClass SVM is an unsupervised learning methodology for outlier / novelty detection. With this methodology, we can use only the phone owner's data and can detect if an event is an outlier or is belong to the phone owner. With this approach, we minimize the risks that come with the biometric privacy regulations.

To train this dataset, we have used the features which was also used in binary SVM training. But in this evaluation, we have used all of the users data to train the model, then we have tested this model with a sample dataset of other users, to find the optimum parameters for oneclass SVM model.

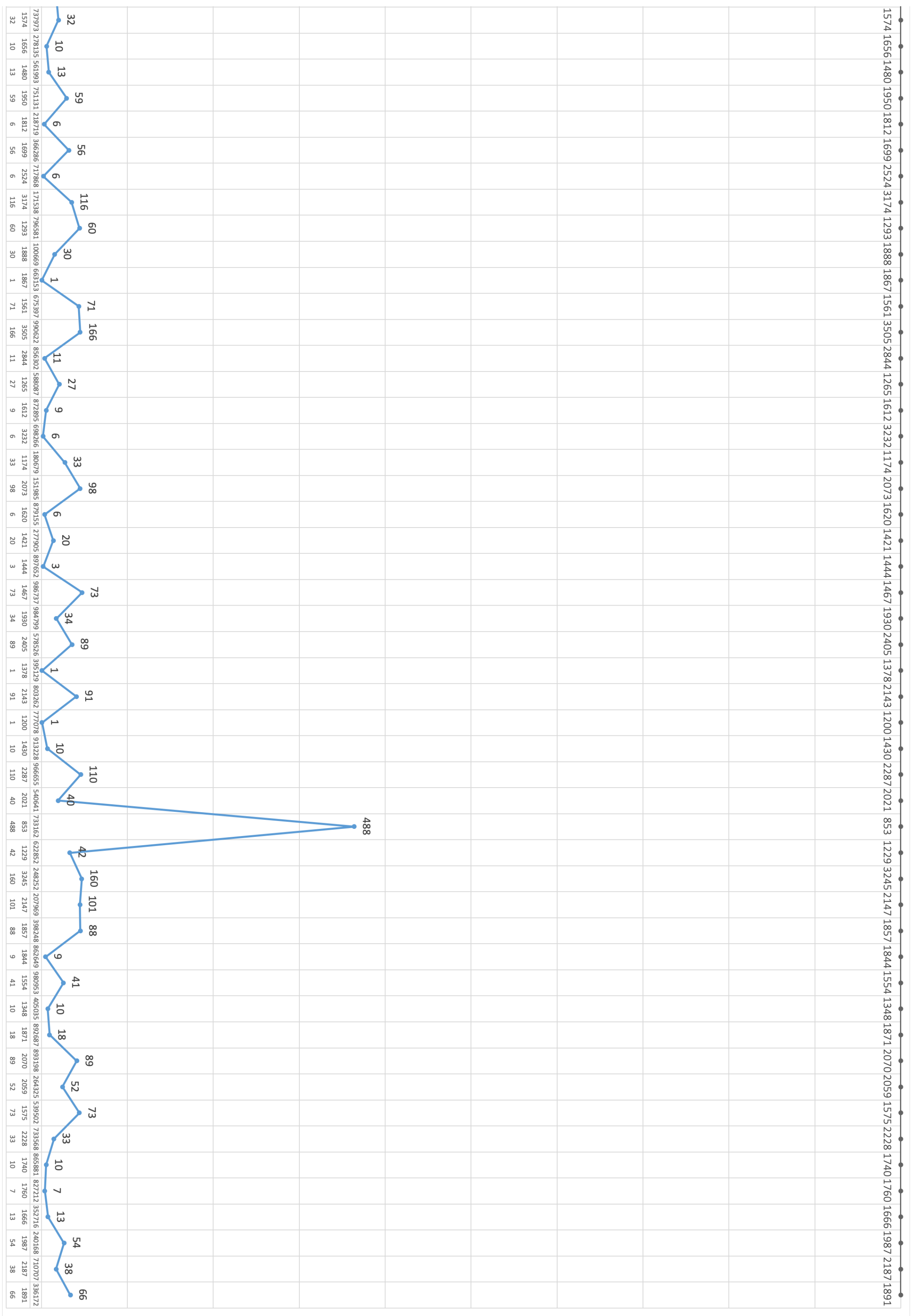
The test data we have used had approximately 20 thousands of instances sampled from all the other except the user whose data is used as training data. So we trained our model with the %80 of the users data, we tested itself with its own %20 of the data and then we tested the model with the test dataset. The results are very promising when we compare to (Sitová et al., 2016) and (Frank et al., 2013) papers because of the same dataset and features, we can achieve more than %90 of f1 measure for nearly each user. And we can also see that nearly in all of the test processes, the models with the best parameters predicted outliers perfectly.

In Figures **5.6**, **5.7**, **5.8** and **5.9**, we can see the results for training and test results for our One Class SVM model. Except the user with id 733162, all other users nearly recognised all of the outlier data. For the genuine data recognition, we received more than 94% accuracy. So the general usage of One Class SVM is novelty detection and fraud recognition. With that results, we can recognise the anomaly usage and normal usage with acceptable results. Also, the all results for One Class SVM can be found in the Figure **APPENDIX .2**



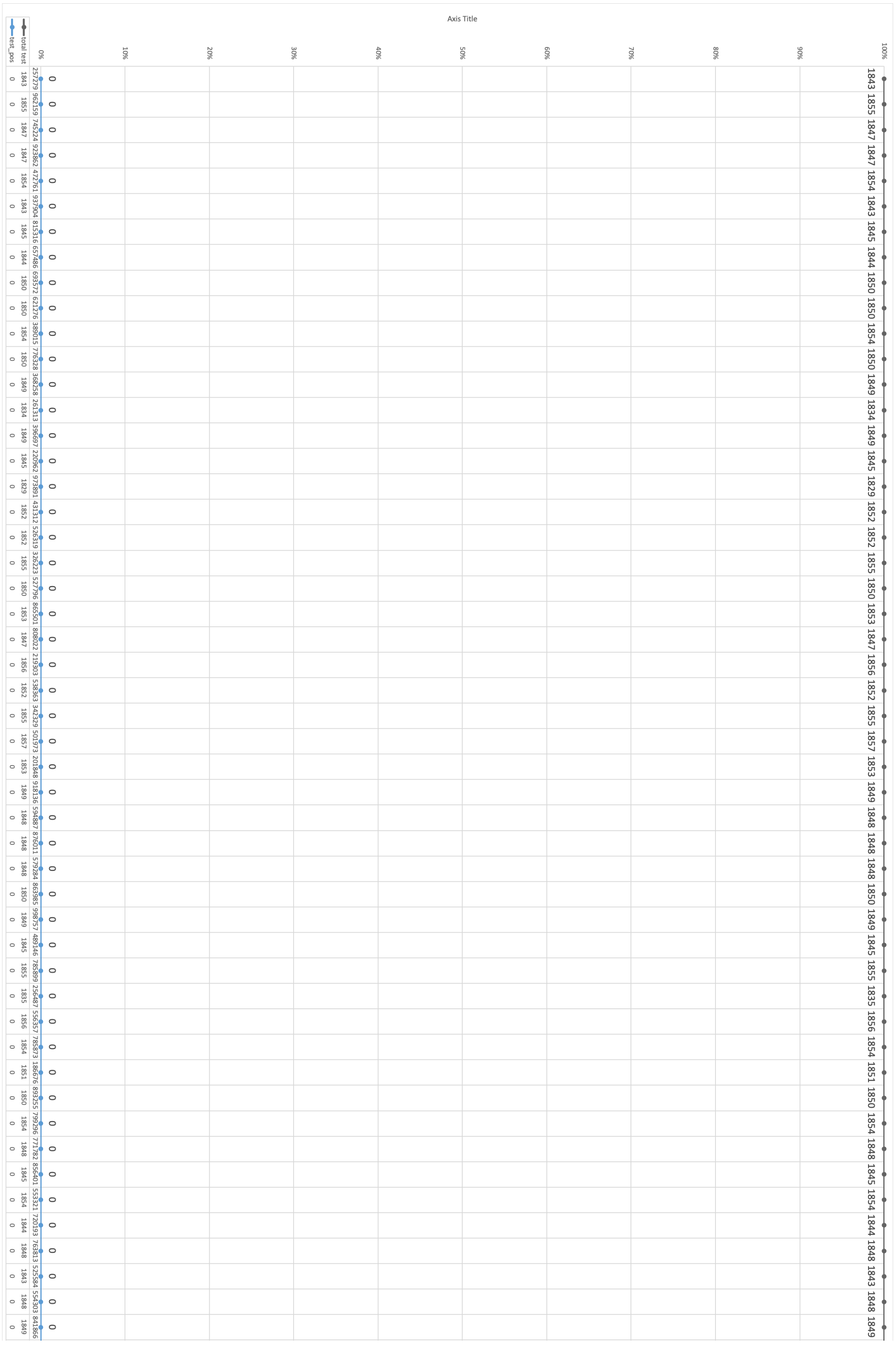
	total train	train_neg
257279	2406	49
962159	1261	31
745224	2038	68
923862	2074	5
472761	1357	74
937904	2438	73
815316	2217	29
657486	2350	57
693572	1771	15
621276	1686	32
389015	1313	41
776328	1690	23
368258	1844	22
261313	3384	22
396697	1839	83
220962	2244	6
973891	3866	140
431312	1580	2
526319	1550	22
326223	1210	14
527796	1764	13
865501	1424	11
808022	2037	53
219303	1178	11
538363	1505	69
342329	1212	3
501973	1052	22
201848	1388	5
918136	1826	46
594887	1982	13
876011	1888	3
579284	1897	35
863985	1730	17
998757	1827	9
489146	2223	94
785899	1281	19
256487	3203	115
556357	1152	10
785873	1372	11
186676	1595	28
893255	1749	62
799296	1363	25
771782	1930	22
856401	2246	25
553321	1340	4
720193	2289	34
763813	1947	86
525584	2388	13
554303	1918	32
841866	1870	32

Figure **5.6**: Best Results for One Class SVM Classification - Train Data 1



737973	1574	32
278135	1656	10
561993	1480	13
751131	1950	59
218719	1812	6
366286	1699	56
717868	2524	6
171538	3174	116
796581	1293	60
100669	1888	30
663153	1867	1
675397	1561	71
990622	3505	166
856302	2844	11
588087	1265	27
872895	1612	9
698266	3232	6
180679	1174	33
151985	2073	98
879155	1620	6
277905	1421	20
897652	1444	3
986737	1467	73
984799	1930	34
578526	2405	89
395129	1378	1
803262	2143	91
777078	1200	1
913228	1430	10
966655	2287	110
540641	2021	40
733162	853	488
622852	1229	42
248252	3245	160
207969	2147	101
398248	1857	88
862649	1844	9
980953	1554	41
405035	1348	10
892687	1871	18
893198	2070	89
264325	2059	52
539502	1575	73
733568	2228	33
865881	1740	10
827212	1760	7
352716	1666	13
240168	1987	54
710707	2187	38
336172	1891	66

Figure **5.7**: Best Results for One Class SVM Classification - Train Data 2



	total test	test_pos
257279	1843	0
962159	1855	0
745224	1847	0
923862	1847	0
472761	1854	0
937904	1843	0
815316	1845	0
657486	1844	0
693572	1850	0
621276	1850	0
389015	1854	0
776328	1850	0
368258	1849	0
261313	1834	0
396697	1849	0
220962	1845	0
973891	1829	0
431312	1852	0
526319	1852	0
326223	1855	0
527796	1850	0
865501	1853	0
808022	1847	0
219303	1856	0
538363	1852	0
342329	1855	0
501973	1857	0
201848	1853	0
918136	1849	0
594887	1848	0
876011	1848	0
579284	1848	0
863985	1850	0
998757	1849	0
489146	1845	0
785899	1855	0
256487	1835	0
556357	1856	0
785873	1854	0
186676	1851	0
893255	1850	0
799296	1854	0
771782	1848	0
856401	1845	0
553321	1854	0
720193	1844	0
763813	1848	0
525584	1843	0
554303	1848	0
841866	1849	0

Figure **5.8**: Best Results for One Class SVM Classification - Test Data 1

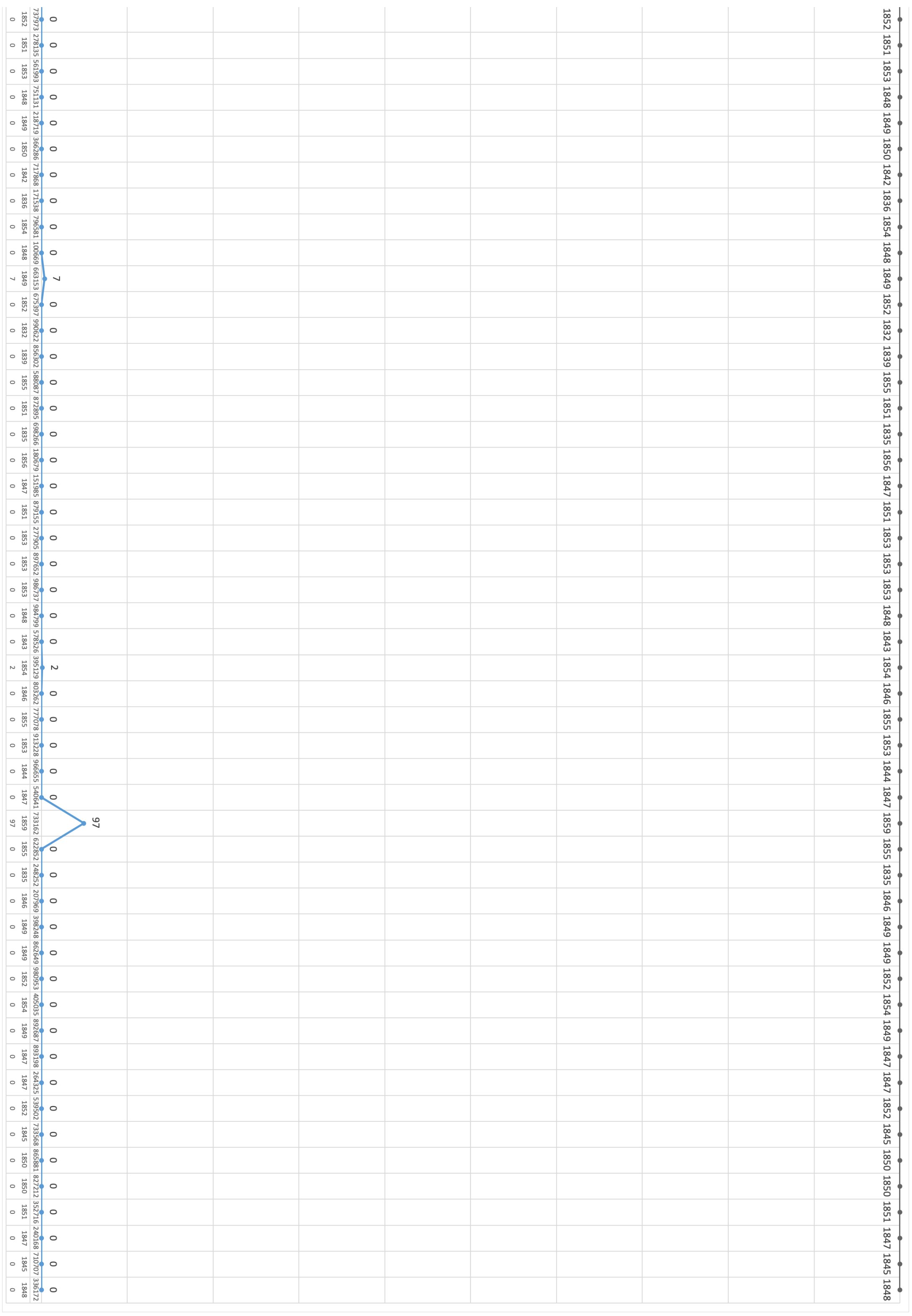


Figure **5.9**: Best Results for One Class SVM Classification - Test Data 2

6 CONCLUSION AND FUTURE WORK

In this study, we investigated a primitive deep network with three layers, a binary svm model and one-class svm model for continuous authentication on mobile phones with a challenging dataset which consists of 100 users over 24 sessions. We analyzed the results of a deep learning algorithm with two labels, a machine learning algorithm with two labels and a machine learning algorithm that predicts outliers.

With these three different evaluation processes, we can also compare a deep learning model with a machine learning model, and a multi-class machine learning algorithm with a one-class algorithm. These two different comparisons can help us to explain which algorithm is more usable in terms of security, performance and easiness.

A deep learning model usually needs a lot of data but we observed that even the features were not optimized, our primitive deep network provided a usable solution. Even though the accuracy and the f1 score are significantly high, the deep network cannot answer to the question of security. How can we build and maintain a multi-class deep neural network without compromising another users biometric data?

A binary class SVM algorithm provided very high performance with an f1 score more than %99.8 on average. These results can show that with optimized parameters, a machine learning algorithm can provide a good result compared to a deep learning algorithm. But we should not forget that, to achieve these results, we had to optimize the features too. But with this algorithm again, we have the same question that we asked also in deep network process. How can we resolve privacy?

With the results of the outlier detector, the one class svm algorithm, we can see that an unsupervised algorithm can also provide good results. But compared to others, this algorithm can answer to the question related to security. With this algorithm, we do not have some other users' biometrics to build and train our machine learning algorithm. With this model, we can only use the phone owner's sensor and screen data and then we can create a user specific model, and then we can predict outliers without needing a non-genuine-user data.

All of these three methods provided great results, so the use cases can affect the model decision. If the system uses a cloud or a server, then the privacy question can be answered and we can use a multi class classification model. If the system makes predictions locally, than one class svm can help the process.

As a future work, time series algorithms can be applied to dataset to avoid feature extraction and the real-world data can be used to verify results. Also a mobile banking application data will be used to verify this work and the determined model can be used in a real world example. Currently, we are collecting our own dataset with a logger integrated in a mobile banking app.

As another future work, other than scroll data can be used to create a user model. In this research we only used user scrolls and the micro-movements related to these usages.But we are not only doing scrolls when we are using a mobile phone. So how would the model change if we also include click events to the dataset? This question can also be answered in a future study.

REFERENCES

Alzubaidi, A. and Kalita, J. (2016). Authentication of smartphone users using behavioral biometrics, *IEEE Communications Surveys Tutorials* **18**(3): 1998–2026.

Amini, S., Noroozi, V., Bahaadini, S., Yu, P. S. and Kanich, C. (2018). Deepfp: A deep learning framework for user fingerprinting via mobile motion sensors, *2018 IEEE International Conference on Big Data (Big Data)*, pp. 84–91.

Bergamini, C., Oliveira, L., Koerich, A. and Sabourin, R. (2009). Combining different biometric traits with one-class classification, *Signal Processing* **89**(11): 2117 – 2127.
URL: *http://www.sciencedirect.com/science/article/pii/S0165168409001844*

Buriro, A., Crispo, B., Del Frari, F., Klardie, J. and Wrona, K. (2016). Itsme: Multi-modal and unobtrusive behavioural user authentication for smartphones, *in* F. Stajano, S. F. Mjølsnes, G. Jenkinson and P. Thorsheim (eds), *Technology and Practice of Passwords*, Springer International Publishing, Cham, pp. 45–61.

Centeno, M. P. n., Guan, Y. and van Moorsel, A. (2018). Mobile based continuous authentication using deep features, *Proceedings of the 2Nd International Workshop on Embedded and Mobile Deep Learning*, EMDL'18, ACM, New York, NY, USA, pp. 19–24.
URL: *http://doi.acm.org/10.1145/3212725.3212732*

Centeno, M. P., v. Moorsel, A. and Castruccio, S. (2017). Smartphone continuous authentication using deep learning autoencoders, *2017 15th Annual Conference on Privacy, Security and Trust (PST)*, pp. 147–1478.

Center, P. R. (accessed April 2019). Smartphone ownership is growing rapidly around the world, but not always equally.
URL: *https://www.pewglobal.org/2019/02/05/smartphone-ownership-is-growing-rapidly-around-the-world-but-not-always-equally/*

Chang, I., Low, C., Choi, S. and Teoh, A. B. (2018). Kernel deep regression network for touch-stroke dynamics authentication, *IEEE Signal Processing Letters* **25**(7): 1109–1113.

Frank, M., Biedert, R., Ma, E., Martinovic, I. and Song, D. (2013). Touchalytics: On the applicability of touchscreen input as a behavioral biometric for continuous authentication, *Information Forensics and Security, IEEE Transactions on* **8**(1): 136 –148.

Hejazi, M., Al-Haddad, S., Singh, Y. P., Hashim, S. J. and Aziz, A. F. A. (2016). Ecg biometric authentication based on non-fiducial approach using kernel methods, *Digital Signal Processing* **52**: 72 – 86.
URL: *http://www.sciencedirect.com/science/article/pii/S1051200416000373*

Horton, R. E. (1941). An approach toward a physical interpretation of infiltration-capacity 1, *Soil science society of America journal* **5**(C): 399–417.

Jia, X., Zang, Y., Zhang, N., Yang, X. and Tian, J. (2014). One-class svm with negative examples for fingerprint liveness detection, *in* Z. Sun, S. Shan, H. Sang, J. Zhou, Y. Wang and W. Yuan (eds), *Biometric Recognition*, Springer International Publishing, Cham, pp. 216–224.

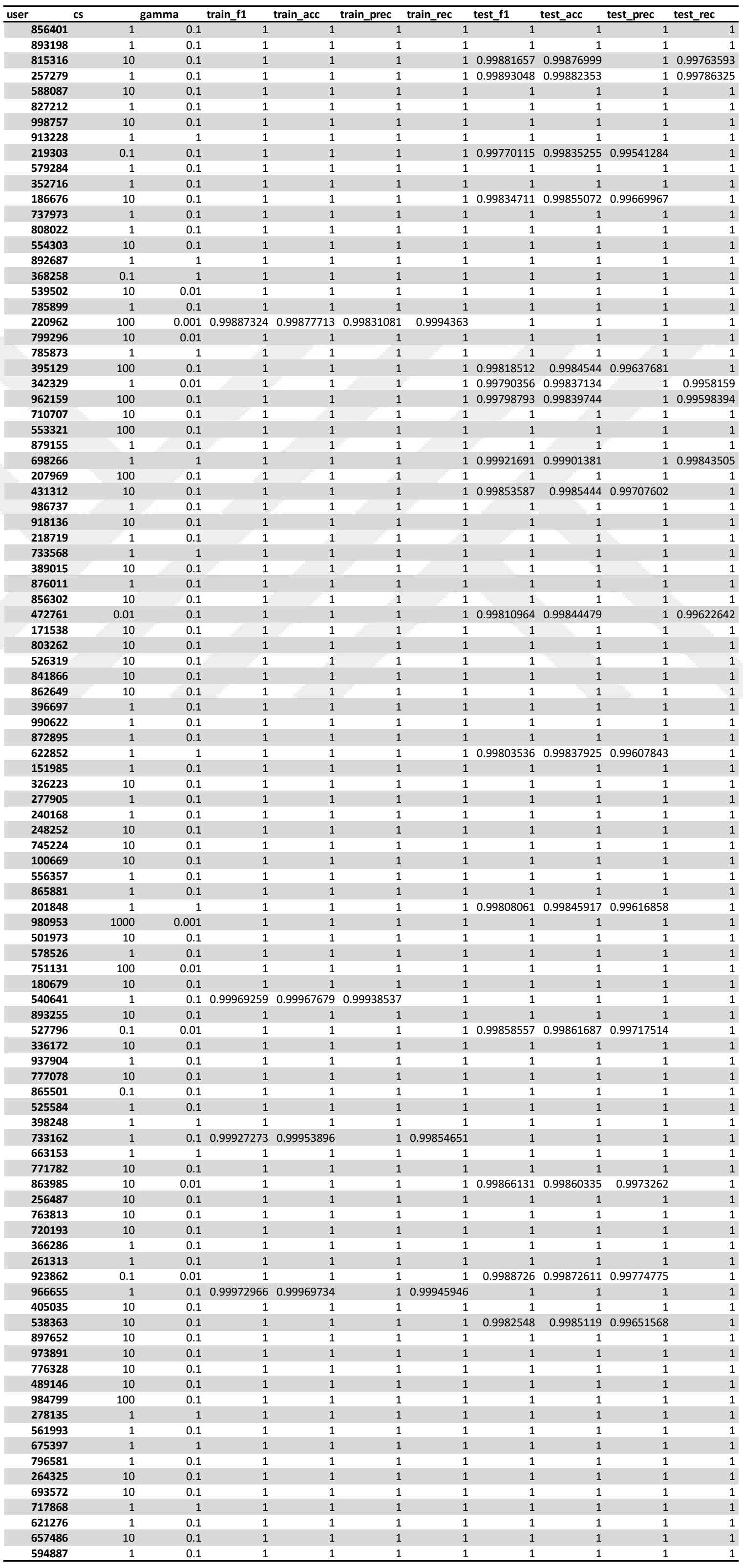
MathWorks (accessed June 2019). Support vector machine.
URL: *https://www.mathworks.com/discovery/support-vector-machine.html*

Microsoft (accessed June 2019). One-class support vector machine.
URL: *https://docs.microsoft.com/en-us/azure/machine-learning/studio-module-reference/one-class-support-vector-machine*

Patel, V. M., Chellappa, R., Chandra, D. and Barbello, B. (2016). Continuous user authentication on mobile devices: Recent progress and remaining challenges, *IEEE Signal Processing Magazine* **33**(4): 49–61.

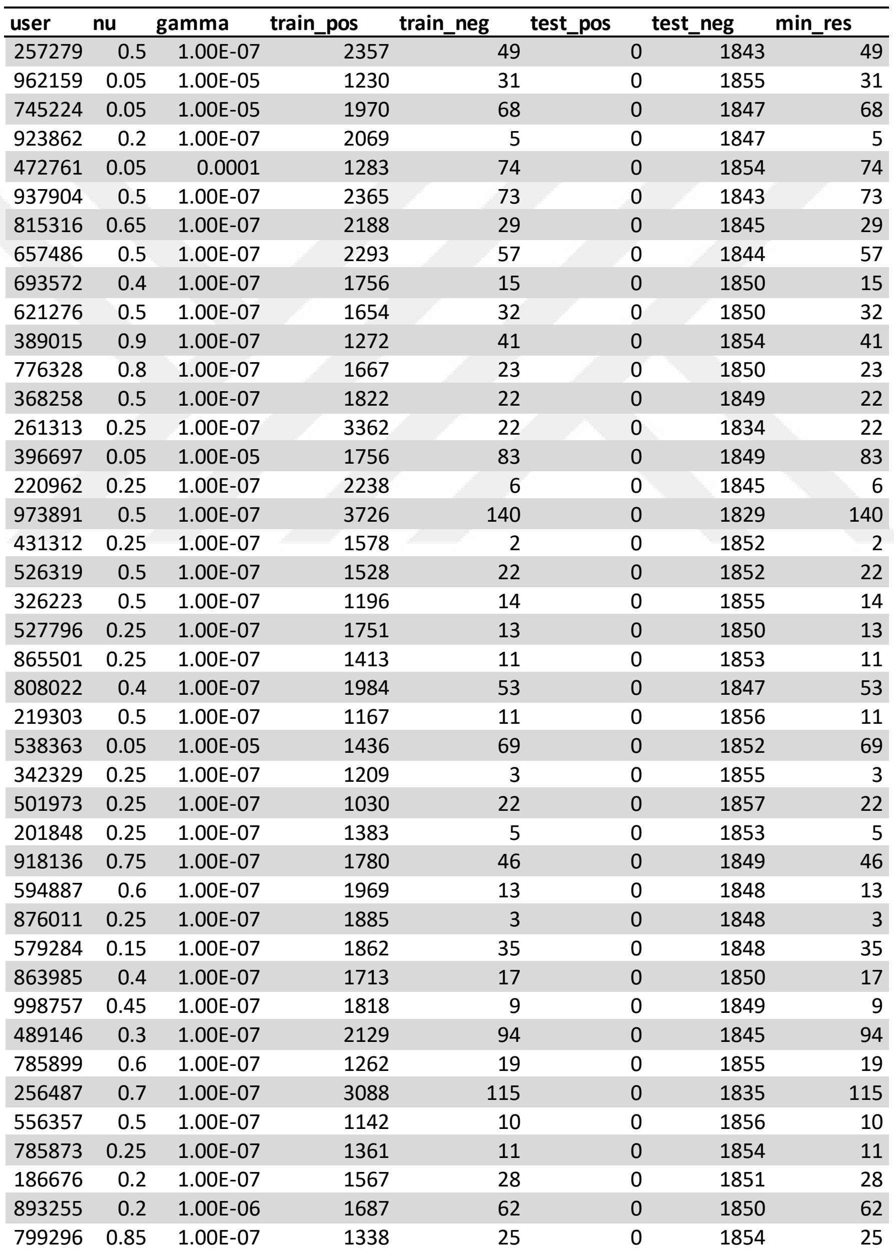
Sitová, Z., Šeděnka, J., Yang, Q., Peng, G., Zhou, G., Gasti, P. and Balagani, K. S. (2016). Hmog: New behavioral biometric features for continuous authentication of smartphone users, *IEEE Transactions on Information Forensics and Security* **11**(5): 877–892.

Yang, Q., Peng, G., Nguyen, D. T., Qi, X., Zhou, G., Sitová, Z., Gasti, P. and Balagani, K. S. (2014). A multimodal data set for evaluating continuous authentication performance in smartphones, *Proceedings of the 12th ACM Conference on Embedded Network Sensor Systems*, SenSys '14, ACM, New York, NY, USA, pp. 358–359.
URL: *http://doi.acm.org/10.1145/2668332.2668366*



user	cs	gamma	train_f1	train_acc	train_prec	train_rec	test_f1	test_acc	test_prec	test_rec
856401	1	0.1	1	1	1	1	1	1	1	1
893198	1	0.1	1	1	1	1	1	1	1	1
815316	10	0.1	1	1	1	1	0.99881657	0.99876999	1	0.99763593
257279	1	0.1	1	1	1	1	0.99893048	0.99882353	1	0.99786325
588087	10	0.1	1	1	1	1	1	1	1	1
827212	1	0.1	1	1	1	1	1	1	1	1
998757	10	0.1	1	1	1	1	1	1	1	1
913228	1	1	1	1	1	1	1	1	1	1
219303	0.1	0.1	1	1	1	1	0.99770115	0.99835255	0.99541284	1
579284	1	0.1	1	1	1	1	1	1	1	1
352716	1	0.1	1	1	1	1	1	1	1	1
186676	10	0.1	1	1	1	1	0.99834711	0.99855072	0.99669967	1
737973	1	0.1	1	1	1	1	1	1	1	1
808022	1	0.1	1	1	1	1	1	1	1	1
554303	10	0.1	1	1	1	1	1	1	1	1
892687	1	1	1	1	1	1	1	1	1	1
368258	0.1	1	1	1	1	1	1	1	1	1
539502	10	0.01	1	1	1	1	1	1	1	1
785899	1	0.1	1	1	1	1	1	1	1	1
220962	100	0.001	0.99887324	0.99877713	0.99831081	0.9994363	1	1	1	1
799296	10	0.01	1	1	1	1	1	1	1	1
785873	1	1	1	1	1	1	1	1	1	1
395129	100	0.1	1	1	1	1	0.99818512	0.9984544	0.99637681	1
342329	1	0.01	1	1	1	1	0.99790356	0.99837134	1	0.9958159
962159	100	0.1	1	1	1	1	0.99798793	0.99839744	1	0.99598394
710707	10	0.1	1	1	1	1	1	1	1	1
553321	100	0.1	1	1	1	1	1	1	1	1
879155	1	0.1	1	1	1	1	1	1	1	1
698266	1	1	1	1	1	1	0.99921691	0.99901381	1	0.99843505
207969	100	0.1	1	1	1	1	1	1	1	1
431312	10	0.1	1	1	1	1	0.99853587	0.9985444	0.99707602	1
986737	1	0.1	1	1	1	1	1	1	1	1
918136	10	0.1	1	1	1	1	1	1	1	1
218719	1	0.1	1	1	1	1	1	1	1	1
733568	1	1	1	1	1	1	1	1	1	1
389015	10	0.1	1	1	1	1	1	1	1	1
876011	1	0.1	1	1	1	1	1	1	1	1
856302	10	0.1	1	1	1	1	1	1	1	1
472761	0.01	0.1	1	1	1	1	0.99810964	0.99844479	1	0.99622642
171538	10	0.1	1	1	1	1	1	1	1	1
803262	10	0.1	1	1	1	1	1	1	1	1
526319	10	0.1	1	1	1	1	1	1	1	1
841866	10	0.1	1	1	1	1	1	1	1	1
862649	10	0.1	1	1	1	1	1	1	1	1
396697	1	0.1	1	1	1	1	1	1	1	1
990622	1	0.1	1	1	1	1	1	1	1	1
872895	1	0.1	1	1	1	1	1	1	1	1
622852	1	1	1	1	1	1	0.99803536	0.99837925	0.99607843	1
151985	1	0.1	1	1	1	1	1	1	1	1
326223	10	0.1	1	1	1	1	1	1	1	1
277905	1	0.1	1	1	1	1	1	1	1	1
240168	1	0.1	1	1	1	1	1	1	1	1
248252	10	0.1	1	1	1	1	1	1	1	1
745224	10	0.1	1	1	1	1	1	1	1	1
100669	10	0.1	1	1	1	1	1	1	1	1
556357	1	0.1	1	1	1	1	1	1	1	1
865881	1	0.1	1	1	1	1	1	1	1	1
201848	1	1	1	1	1	1	0.99808061	0.99845917	0.99616858	1
980953	1000	0.001	1	1	1	1	1	1	1	1
501973	10	0.1	1	1	1	1	1	1	1	1
578526	1	0.1	1	1	1	1	1	1	1	1
751131	100	0.01	1	1	1	1	1	1	1	1
180679	10	0.1	1	1	1	1	1	1	1	1
540641	1	0.1	0.99969259	0.99967679	0.99938537	1	1	1	1	1
893255	10	0.1	1	1	1	1	1	1	1	1
527796	0.1	0.01	1	1	1	1	0.99858557	0.99861687	0.99717514	1
336172	10	0.1	1	1	1	1	1	1	1	1
937904	1	0.1	1	1	1	1	1	1	1	1
777078	10	0.1	1	1	1	1	1	1	1	1
865501	0.1	0.1	1	1	1	1	1	1	1	1
525584	1	0.1	1	1	1	1	1	1	1	1
398248	1	1	1	1	1	1	1	1	1	1
733162	1	0.1	0.99927273	0.99953896	1	0.99854651	1	1	1	1
663153	1	1	1	1	1	1	1	1	1	1
771782	10	0.1	1	1	1	1	1	1	1	1
863985	10	0.01	1	1	1	1	0.99866131	0.99860335	0.9973262	1
256487	10	0.1	1	1	1	1	1	1	1	1
763813	10	0.1	1	1	1	1	1	1	1	1
720193	10	0.1	1	1	1	1	1	1	1	1
366286	1	0.1	1	1	1	1	1	1	1	1
261313	1	0.1	1	1	1	1	1	1	1	1
923862	0.1	0.01	1	1	1	1	0.9988726	0.99872611	0.99774775	1
966655	1	0.1	0.99972966	0.99969734	1	0.99945946	1	1	1	1
405035	10	0.1	1	1	1	1	1	1	1	1
538363	10	0.1	1	1	1	1	0.9982548	0.9985119	0.99651568	1
897652	10	0.1	1	1	1	1	1	1	1	1
973891	10	0.1	1	1	1	1	1	1	1	1
776328	10	0.1	1	1	1	1	1	1	1	1
489146	10	0.1	1	1	1	1	1	1	1	1
984799	100	0.1	1	1	1	1	1	1	1	1
278135	1	1	1	1	1	1	1	1	1	1
561993	1	0.1	1	1	1	1	1	1	1	1
675397	1	1	1	1	1	1	1	1	1	1
796581	1	0.1	1	1	1	1	1	1	1	1
264325	10	0.1	1	1	1	1	1	1	1	1
693572	10	0.1	1	1	1	1	1	1	1	1
717868	1	1	1	1	1	1	1	1	1	1
621276	1	0.1	1	1	1	1	1	1	1	1
657486	10	0.1	1	1	1	1	1	1	1	1
594887	1	0.1	1	1	1	1	1	1	1	1

Figure **APPENDIX .1**: Best Binary SVM Results



user	nu	gamma	train_pos	train_neg	test_pos	test_neg	min_res
257279	0.5	1.00E-07	2357	49	0	1843	49
962159	0.05	1.00E-05	1230	31	0	1855	31
745224	0.05	1.00E-05	1970	68	0	1847	68
923862	0.2	1.00E-07	2069	5	0	1847	5
472761	0.05	0.0001	1283	74	0	1854	74
937904	0.5	1.00E-07	2365	73	0	1843	73
815316	0.65	1.00E-07	2188	29	0	1845	29
657486	0.5	1.00E-07	2293	57	0	1844	57
693572	0.4	1.00E-07	1756	15	0	1850	15
621276	0.5	1.00E-07	1654	32	0	1850	32
389015	0.9	1.00E-07	1272	41	0	1854	41
776328	0.8	1.00E-07	1667	23	0	1850	23
368258	0.5	1.00E-07	1822	22	0	1849	22
261313	0.25	1.00E-07	3362	22	0	1834	22
396697	0.05	1.00E-05	1756	83	0	1849	83
220962	0.25	1.00E-07	2238	6	0	1845	6
973891	0.5	1.00E-07	3726	140	0	1829	140
431312	0.25	1.00E-07	1578	2	0	1852	2
526319	0.5	1.00E-07	1528	22	0	1852	22
326223	0.5	1.00E-07	1196	14	0	1855	14
527796	0.25	1.00E-07	1751	13	0	1850	13
865501	0.25	1.00E-07	1413	11	0	1853	11
808022	0.4	1.00E-07	1984	53	0	1847	53
219303	0.5	1.00E-07	1167	11	0	1856	11
538363	0.05	1.00E-05	1436	69	0	1852	69
342329	0.25	1.00E-07	1209	3	0	1855	3
501973	0.25	1.00E-07	1030	22	0	1857	22
201848	0.25	1.00E-07	1383	5	0	1853	5
918136	0.75	1.00E-07	1780	46	0	1849	46
594887	0.6	1.00E-07	1969	13	0	1848	13
876011	0.25	1.00E-07	1885	3	0	1848	3
579284	0.15	1.00E-07	1862	35	0	1848	35
863985	0.4	1.00E-07	1713	17	0	1850	17
998757	0.45	1.00E-07	1818	9	0	1849	9
489146	0.3	1.00E-07	2129	94	0	1845	94
785899	0.6	1.00E-07	1262	19	0	1855	19
256487	0.7	1.00E-07	3088	115	0	1835	115
556357	0.5	1.00E-07	1142	10	0	1856	10
785873	0.25	1.00E-07	1361	11	0	1854	11
186676	0.2	1.00E-07	1567	28	0	1851	28
893255	0.2	1.00E-06	1687	62	0	1850	62
799296	0.85	1.00E-07	1338	25	0	1854	25

Figure **APPENDIX .2**: Best One Class SVM Results

BIOGRAPHICAL SKETCH

First Name: Hasan Can

Surname: Volaka

Place of Birth and Date of Birth: Konak/İZMİR and 1992

Bachelor Degree: Galatasaray University, Computer Engineering (2016)

Current Position: Developer Advocate at IBM, since August 2018