MOBILE DEVICE IDENTIFICATION VIA SENSOR FINGERPRINTING BASED ON USER BEHAVIOR ANALYSIS

(KULLANICI ALIŞKANLIKLARINA DAYALI SENSÖR TABANLI PARMAK İZİ SİSTEMİ İLE AKILLI CİHAZ TANIMA)

by

Kadriye DOĞAN, 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

Supervisor: Assoc. Prof. Dr. Özlem DURMAZ İNCEL

July 2019

This is to certify that the thesis entitled

MOBILE DEVICE IDENTIFICATION VIA SENSOR FINGERPRINTING BASED ON USER BEHAVIOR ANALYSIS

prepared by **KADRIYE DOĞAN** 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

Assist. Prof. Dr. Reis Burak Arslan
Department of Computer Engineering _____
Galatasaray University

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

Date: _____

ACKNOWLEDGMENTS

I would first like to thank to my thesis supervisor, Assoc. Prof. Dr. Özlem DURMAZ İNCEL for her mental and material support from the very beginning of the project. It is a great honour to work under her supervision.

Secondly, I would also like to thank my family who supported me with love and understanding. Without you, I could never have reached this current level of success.

I would also like to thank my friends, and colleagues each of whom has provided patient advice and guidance throughout the process. Thank you all for your unwavering support.

July 2019
Kadriye DOĞAN

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

Modern mobile devices are capable of sensing a large variety of changes, ranging from users' motions to environmental conditions. Context-aware applications utilize the sensing capability of these devices for various purposes, such as human activity recognition, health coaching or advertising, etc. Identifying devices and authenticating unique users is another application area where mobile device sensors can be utilized to ensure more intelligent, robust and reliable systems. Traditional systems use cookies, hardware or software fingerprinting to identify a user but due to privacy and security vulnerabilities, none of these methods propose a permanent solution, thus sensor fingerprinting not only identifies devices but also makes it possible to create non-erasable fingerprints.

In this thesis, we focus on distinguishing devices via mobile device sensors. To this end, a large dataset, larger than 140 GB, which consists of accelerometer, gyroscope, pressure, light and gravity sensor data from 25 distinct devices is utilized. We employ different classification methods on extracted features based on various time windows from mobile sensors. Namely, we use random forest, gradient boosting machine, generalized linear model and artificial neural network. In conclusion, we obtain the highest accuracy as 96% from various experiments in identifying 25 devices using random forest on the data from accelerometer and gyroscope sensors.

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

ÖZET

Bu yüksek lisans tez çalışması akıllı cihazlarda yer alan sensörleri kullanarak, kullanıcı alışkanlıklarına göre değişen sensör hareketlerindeki farklılıkları tespit edip cihazların tanımlanmasını sağlayan bir çalışma sunmaktadır.

Akıllı cihaz sensörleri, kullanıcı hareket tanılama, sağlık koçluğu, ani hareketlerin tespit edilmesi gibi birçok farklı alanda hali hazırda kullanılmaktadır. Bunların dışında, sensör verilerine dayanarak cihaz tanılama ile daha akıllı ve kullanışlı bir sistem tasarlanabilir. Klasik kullanıcı tanılama teknolojileri, web/mobil üzerinde kullanıcının tekilliğini ifade eden, donanım ya da yazılım tarafından üretilmiş kimlik bilgisini kullanmaktadır. Örneğin, tarayıcılarda yer alan çerez bilgisi, akıllı cihazlarda bulunan seri numarası mevcut sistemler tarafından kullanılmaktadır. Fakat, bu yöntemler gizlilik ve kalıcılık barındırmaması nedeniyle cihaz tanılama problemine çözüm getirmemektedir.

Akıllı cihaz sensörleri ile kullanıcı tanılama çalışması kapsamında 25 farklı kullanıcı için 140 GB dan fazla sensör verisi kullanıldı. Veri içerisinde yer alan farklı sensörler arasından ivmeölçer, jiroskop, basınç, yer çekimi ve ışık sensörleri tercih edildi. Sensör verilerinden çıkarılan öznitelikleri değerlendirmek için Rastgele Orman, Gradyan Artırma, Genelleştirilmiş Lineer Model ve Yapay Sinir Ağı algoritmaları kullanıldı. Sonuç olarak, ivmeölçer ve jiroskop sensörleri birlikte kullanıldığında, Rastgele Orman algoritması ile %96 gibi bir başarı oranı elde edildi.

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

LIST OF ABBREVIATIONS

RF	:	Random Forest
GLM	:	Generalized Linear Model
GBM	:	Gradient Boosting Machine
ANN	:	Artificial Neural Network
MEMS	:	Microelectromechanical Systems
CSIO	:	CrowdSignals.io
ML	:	Machine Learning
ACC	:	Accelerometer Features
GYRO	:	Gyroscope Features
ACCGYRO	:	Accelerometer and Gyroscope Features
ACCPRS	:	Accelerometer and Pressure Features
ACCGRVT	:	Accelerometer and Gravity Features
ACCLGTH	:	Accelerometer and Ligth Features

1 INTRODUCTION

Modern web or mobile applications aim to identify users uniquely to recommend more intelligent contents, products or display them only relevant ads. In addition to intelligent guidance, particular kind of applications need to remember the last state of a user to continue recent activity, such as showing shopping cart with items added before on an e-commerce system or starting a TV show from the last scene. Accordingly, traditional user identification systems use well known methodologies such as cookies, hardware and software fingerprinting. However, these solutions exhibit various disadvantages.

The first identifier, a cookie, can be removed from a browser by a user at any time and 3rd party applications can also access the information stored by a cookie. In addition to these problems, many security vulnerabilities are detected in browsers related to confidential data access such as credit card, email or passwords. Another identifier, hardware fingerprinting, represents device specific identification number and enables tracking users via device id such as IMEI (available only phones), WLAN MAC address, bluetooth MAC address, etc. However, applications which intend to use hardware fingerprint need to obtain user permission due to privacy regulations. Besides a user can change the device at anytime, so that the system loses one of the identified users.

The last identifier, software fingerprinting, consists of two main categories; browser and mobile advertisement fingerprinting. The browser fingerprinting methodology collects detailed information from a browser and the operating system, such as language, timezone, monitor settings, etc., and produces unique identification key for each user but in some cases this key can be similar to another individual due to the same configurations. For instance, two colleagues work in the same department of a bank and IT department configures all the notebooks exactly the same to reduce operation costs and then restricts specifications, such as language, timezone or plugin installations. Hence, the same browser fingerprint key will be generated for all users of the bank.

Lastly, mobile advertisement fingerprinting is both enabled for Android (AAID) and IOS (IDFA) operating systems that allow access to user id for advertising for various marketing purposes. However, a user can reset his/her device at anytime or remove permission from an application so this key is only reliable as much as hardware finger-

printing.

In consideration of traditional fingerprinting problems, researchers proposed a new methodology, sensor fingerprinting, which detects behavioral patterns of a user via collected sensor data from mobile devices. Sensor fingerprinting solves both user permission and persistent identifier problems and makes easier to develop reliable and robust systems. Modern browsers are capable of accessing device sensors such as accelerometer, gyroscope, etc. without any permission so it is possible to collect and analyze data on the device in the background. Furthermore, sensor fingerprinting only needs sensor data rather than additional device identifiers such as IMEI, AAID, etc. so software or hardware changes does not affect user identification. Eventually, users carry their smartphones or smartwatches almost all the day and have routines in their interaction patterns on the devices, such as reading news on an application after a morning walk. Hence, applications can track user behaviours via mobile sensors effortlessly and distinguish users from each other with different routines.

In this thesis, we explore the performance of sensor fingerprinting in order to identify mobile devices with a large dataset which is provided by CrowdSignals.io (*CrowdSignals.io*, n.d.) and collected from different individuals via various devices. The dataset differs from others which are used in previous works (ul Haq et al., 2017) due to the diversity from various aspects, such as participant characteristics, data size, sensors types, etc. CrowdSignals.io aims to build the largest and richest mobile sensor dataset so researchers, students or developers can access labeled sensor data easily and focus on their researches. Moreover, the dataset represents not only sensor data but also additional information per participant, such as age range, sex, occupation, etc., thereby reliable machine learning models can be created to develop real-time applications.

In particular, we focus on the use of motion sensors, namely accelerometer and gyroscope, in identifying 25 devices/users, considering different classifiers and feature sets. Our aim is to explore recognition with only motion sensors and simple features which are not computationally expensive.

We also explore different window sizes in the process, since our ultimate aim is to develop a real-time system working on the device. The highlights and contributions can be summarized as follows:

— Using only accelerometer sensor is more effective than gyroscope to identify devices correctly. However, combining sensors increases the performance of models

considerably, around 6%. On the other hand, accelerometer can be used standalone when resource usage is a concern with multiple sensors.

— Our experimental results show that Gradient Boosting Machine is the most efficient algorithm among others since each trained tree helps the next one to minimize the errors. Furthermore, Random Forest provides a significant performance slightly less than Gradient Boosting Machine. However, Generalized Linear Model is not a proper classifier when compared to others providing lowest accurate results.

Rest of the thesis is organized as follows: In Section 2, we present the related work and how our method differs from the related studies. Section 3 presents our methodology particularly the parameters and experiments considered in this work. In Section 4, we present the results of the experiments and discuss our findings. Finally, Section 5 concludes the thesis and includes the future studies.

2 RELATED WORK

Many studies have been presented in the area of sensor fingerprinting for different purposes, such as tracking users across applications, strong authentication or secure peer to peer data sharing. Existing studies show that accelerometer sensor has a great impact on identifying devices, hence it is commonly used in almost all experiments.

In a previous study (Dey et al., 2014), authors explain the details of why accelerometer sensor is chosen as a basis for sensor fingerprinting: accelerometer chips produce different signals for the same motion even if the same chip model is used due to hardware faultiness during the manufacturing process. Therefore, devices can be easily differentiated. During experiments, authors utilized 80 accelerometer chips, 25 Android phones and 2 tablets and extracted 36 features from both time and frequency domain. As a result, precision and recall metrics are obtained as 96% by a bagged decision tree classifier using only accelerometer sensor.

In another study (Das et al., 2016) it is shown that using accelerometer and gyroscope sensors together is an effective methodology to gain better performance via machine learning model and can be used in order to identify smartphones uniquely. In experiments, almost same feature set is utilized like study (Dey et al., 2014) from the time and frequency domain as shown at Fig. **2.1**. Despite having 100 features from both sensors in total, authors selected the most efficient 70 features consisting of 21 from accelerometer data and 49 from gyroscope data. Moreover, 44 of 70 features are obtained by frequency domain due to motion changes can be detected easily via spectral features. The results demonstrate that using both sensors increases average F-score from 85% - 90% range to 96%.

Sensor fingerprinting not only solves tracking user problems but also provides smarter solutions for secure authentication systems as discussed in (Lee and Lee, 2017). Many smartphone applications need to re-authenticate the device user after logged-in to provide secure access to sensitive data or applications. Therefore, authors aimed to create a system called SmarterYou which uses smartwatch and smartphone sensors together to prevent illegal device access. We realized that several sensors such as magnetometer, light and orientation sensors eliminated in experiments by the reason of being affected

#	Domain	Feature	Description
1	Time	Mean	The arithmetic mean of the signal strength at different timestamps
2		Standard Deviation	Standard deviation of the signal strength
3		Average Deviation	Average deviation from mean
4		Skewness	Measure of asymmetry about mean
5		Kurtosis	Measure of the flatness or spikiness of a distribution
6		RMS	Square root of the arithmetic mean of the squares of the signal strength at various timestamps
7		Max	Maximum signal strength
8		Min	Minimum signal strength
9		ZCR	The rate at which the signal changes sign from positive to negative or back
10		Non-Negative count	Number of non-negative values
11	Frequency	Spectral Centroid	Represents the center of mass of a spectral power distribution
12		Spectral Spread	Defines the dispersion of the spectrum around its centroid
13		Spectral Skewness	Represents the coefficient of skewness of a spectrum
14		Spectral Kurtosis	Measure of the flatness or spikiness of a distribution relative to a normal distribution
15		Spectral Entropy	Captures the peaks of a spectrum and their locations
16		Spectral Flatness	Measures how energy is spread across the spectrum
17		Spectral Brightness	Amount of spectral energy corresponding to frequencies higher than a given cut-off threshold
18		Spectral Rolloff	Defines the frequency below which 85% of the distribution magnitude is concentrated
19		Spectral Roughness	Average of all the dissonance between all possible pairs of peaks in a spectrum
20		Spectral Irregularity	Measures the degree of variation of the successive peaks of a spectrum
21		Spectral RMS	Square root of the arithmetic mean of the squares of the signal strength at various frequencies
22		Low-Energy-Rate	The percentage of frames with RMS power less than the average RMS power for the whole signal
23		Spectral flux	Measure of how quickly the power spectrum of a signal changes
24		Spectral Attack Time	Average rise time to spectral peaks
25		Spectral Attack Slope	Average slope to spectral peaks

Figure **2.1**: Feature set from both time and frequency domain used in study (Das et al., 2016)

by environment conditions. Authors indicate that these sensors cause noise on data and not related to users' behavior so only accelerometer and gyroscope sensors are preferred in this work. Unlike previous works, device context is also added to feature set as well as sensor features while using smartphones. Therefore, 4 different device contexts are evaluated in experiments; (1) device is being used by participant without moving, (2) with moving, (3) device standing on a surface and using by owner and (4) user utilizes device in vehicle such as bus, train, etc. Hence, Random Forest algorithm is chosen between various ML classifiers in order to detect the user's context. The results show that sensor fingerprinting provides accuracy up to 98.1% by using basic features and user context besides consuming low battery.

In another study (Mayrhofer and Gellersen, 2007), only accelerometer sensor is preferred to generate secret keys in order to propose a new method for pairing devices and providing device-to-device authentication.

Traditional systems use PIN code to pair devices over wireless communication in order to prevent man-in-the-middle attacks. This study aims to pair devices by simultaneous movement of user and meantime simplify the communication process.

In (Bojinov et al., 2014), researchers aim to implement a new device identification system which is based on sensor fingerprinting via accelerometer and speakerphone-microphone sensors on a mobile device. The authors report that over 10000 devices are utilized in experiments and uniquely identifying devices is achieved by sensors instead of using software generated IDs. During experiments, many sensors are tested rather

than accelerometer and microphone but are eliminated due to various reasons. Briefly, magnetometer and light sensors are depend on the environment and GPS is not observable due to battery consumption. As a result, authors report that only 15.1% of devices were identified correctly since the running code which collects accelerometer sensor data on mobile browser is unverified and not reliable.

Another sensor based device identification system is proposed in (Ehatisham-ul Haq et al., 2017) which uses accelerometer, gyroscope and magnetometer sensors. However, this work differs from others by considering device position on body such as waist, wrist, etc. and user activity together. The user activity consist of walking, standing, sitting, running, walking upstairs and walking downstairs. In experiments, Bayes Net, K-Nearest Neighbor and Support Vector Machine are utilized and obtained highest accuracy as 95.58 % with Bayes Net classifier by using the data collected when carrying phone on waist.

Considering previous works, we aim to analyze the performance of complex models when compared to simple ones in this work. Therefore, we utilized Gradient Boosting Machine, Random Forest, Artificial Neural Network as complex models and compared the performance results with a Generalized Linear Model.

3 METHODOLOGY

In this section, we describe the steps of our methodology which are frequently used in machine learning studies. These steps consist of data collection, preprocessing, feature engineering, model training and evaluation.

3.1 Dataset Collection

One of the challenges faced in machine learning studies is the difficulty of collecting labelled datasets. A great effort is spent to find participants and develop data collection applications. Moreover, use of different datasets in different studies makes it difficult to generalize and compare the results. In this respect, publicly available and labelled datasets with high quality are important for reproducibility in the research community. CrowdSignals.io(*CrowdSignals.io*, n.d.) provides a mobile sensor dataset (*CrowdSignals User's Reference Document*, n.d.)(Welbourne and Munguia Tapia, 2014) from various devices and makes it easier to access labelled data.

In this work, we used the CrowdSignals.io dataset which includes data collected from several sensors, such as accelerometer, gyroscope, magnetometer, etc., from 30 different Android users. The data was collected for more than 15 days and contains 5 GB of sensor records per user on average. In spite of having more than 25 users, we eliminated others due to lack of data from motion sensors. Furthermore, although the dataset includes other sensors, such as GPS, proximity, etc. we used 5 different sensors; (1) accelerometer, (2) gyroscope, (3) pressure, (4) light and (5) gravity sensors and we will expand on the structure of these sensors in Section 3.1.1.

Table**3.1**: CrowdSignals.io dataset fields of accelerometer, gyroscope and gravity sensors

Field Name	Description
x	A list of x axis values
y	A list of y axis values
z	A list of z axis values
user id	Unique identifier of user
timestamps	A list of timestamp values

Table **3.1** and Table **3.2** show the utilized fields of all utilized sensors during experi-

ments.

Table**3.2**: CrowdSignals.io dataset fields of light and pressure sensors

Field Name	Description
value	A value of ambient light/pressure
user id	Unique identifier of user
timestamps	A list of timestamp values

While accelerometer, gyroscope and gravity sensors have values of x, y and z axes, light and pressure sensors only include the value of ambient light or pressure. However, user id and list of timestamps are common parameters; user id is unique and will be used as a label for classifiers and timestamps contain a list of timestamps when sensor value is captured. In Section 3.2, we will detail the use of timestamps as a window size to extract features between various time intervals.

In Table **3.3**, details about the participants are provided. All users are Android users, 8 of them are female, age span is between 18 to 60. In addition to the participant characteristics, we also added employment status (fifth column) and the data size (sixth column) per user. The statistics of data size can be summarized as follows;

— The data size is 142.7 GB in total

— Average data size is 5.7 GB for 25 users

— The participant with minimum size of data (301 MB) is user 12

— The participant with maximum size of data (18 GB) is user 23

Despite having 142.7 GB data in total, we reduced the data size due to limited hardware resources such as memory, CPU and disk capacity. In Section 3.2, we also share the details of methodology applied on raw data to sample.

3.1.1 Sensor Types

In this section, we describe sensors used in the experiments to identify devices. Despite CrowdSignals.io provides many of sensors, we only utilize 5 different sensors as follows;

— Accelerometer

— Gyroscope

— Gravity

— Pressure

— Light

Accelerometer and gyroscope sensors are both members of MEMS (Microelectromechanical systems)(*MEMS SENSORS*, n.d.), which integrate mechanical and electrical components to provide sensing mechanisms on mobile devices. MEMS sensors measure linear acceleration or angular motion for one or more axis and provide an input to applications in order to detect device movements.

Fig **3.1** shows that gyroscope sensor measures angular orientation for several axis whereas accelerometer only senses changes on direction. For instance, accelerometer sensor is being used to change screen orientation when phone switched from vertical to horizontal or vice versa.

Table**3.3**: Details of Users and the Data

User	Smartphone	Age Range	Gender	Data Size
1	Samsung Galaxy S5	21-29	Male	1 GB
2	-	-	-	666 MB
6	Samsung Galaxy Note	21-29	Male	2 GB
8	LG K7	30-39	Male	1 GB
9	LG	30-39	Female	1 GB
10	OnePlus 3	21-29	Male	6 GB
11	Samsung Core Prime	21-29	Male	1 GB
12	LG Realm	30-39	Female	301 MB
16	-	-	-	7 GB
19	HTC 10	30-39	Female	10 GB
20	Asus Zenfone 2	30-39	Male	6 GB
21	Samsung Galaxy S7	30-39	Male	14 GB
23	Sony Xperia Z3	60-69	Male	18 GB
24	Samsung S6 Plus Edge	21-29	Female	10 GB
26	Samsung Galaxy S6	30-39	Female	16 GB
27	Samsung SM-A800F	21-29	Female	4 GB
28	Moto G 3rd gen	18-20	Male	845 MB
31	LG G4	40-49	Female	9 GB
32	-	-	-	884 MB
34	Xiaomi Mi 4W	30-39	Male	8 GB
36	-	-	-	5 GB
37	Samsung Galaxy S5	40-49	Male	9 GB
38	Samsung SM-A800F	30-39	Female	3 GB
39	Samsung On5	60-69	Male	1 GB
41	Note 5	30-39	Male	8 GB

However, rotation sensitive applications/services need to use gyroscope to handle small angular changes on same direction such as racing games.

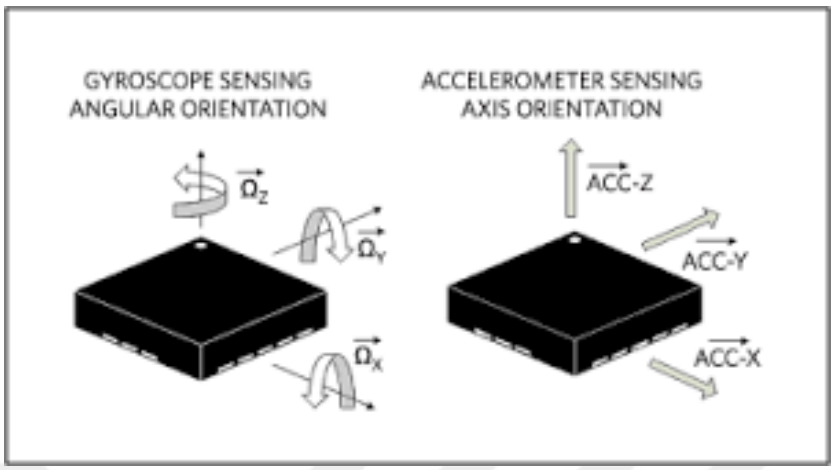

Figure **3.1**: Sensing orientations of accelerometer and gyroscope sensors (*MEMS SENSORS*, n.d.)

In this section, we will focus on the details of sensors we used in the experiments varying from accelerometer to pressure.

Accelerometer

An accelerometer senses axis orientation for x, y and z axis and detects displacement of the device. Therefore, we utilize motion changes of devices to understand users' activity patterns and distinguish from each other.

An accelerometer sensor measures the distance between fixed capacitor pairs when seismic mass moves through the direction. Under acceleration, the distance between fixed electrodes changes in the direction of device displacement so mobile devices manage to provide sensor recordings to the software layer. Therefore, applications/services are able to use accelerometer sensor records in numerous fields such as activity recognition and tracking, fall detection, screen orientation, etc.

Mobile operating systems are able to provide different sampling rates of accelerometer sensor from 4 Hz to 400 Hz.(Dey et al., 2014) For instance Android operating system (*Android Accelerometer Sensor Overview*, n.d.) ensure 4 different sampling delay mode as follows:

— SENSOR_DELAY_NORMAL (200.000 µs)

— SENSOR_DELAY_UI (60.000 µs)

— SENSOR_DELAY_GAME (20.000 µs)

— SENSOR_DELAY_FASTEST (0 µs)

Considering SENSOR_DELAY_NORMAL delay mode, 200.000 µs equals to 0.2 s and it represents that number of 5 sensor records are being captured in every 1 second so sampling rate can be calculated as 5 Hz. In SENSOR_DELAY_FASTEST mode, the sampling rate can be up to 400 Hz depending on device specifications. Generally, 100 Hz is an optimal sampling rate in order to detect motion changes but sensitivity might be crucial for applications which try to capture instant movements. However, in our dataset, sampling rate is around 100 Hz per second for each sensor and we applied windowing techniques to summarize data characteristics for several time intervals in Section 3.2.

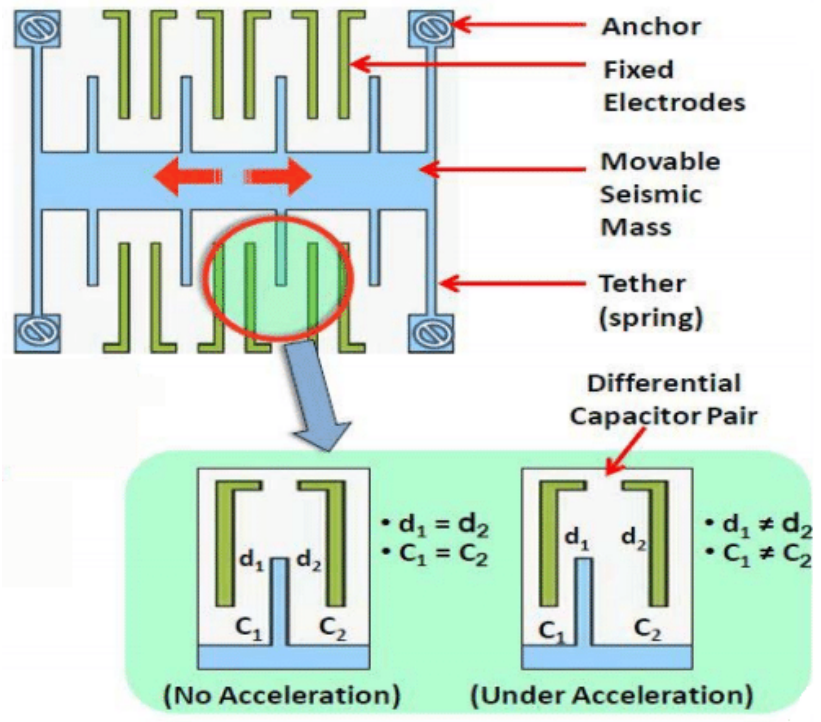

Figure **3.2**: A structure of MEMS accelerometer sensor (*MEMS Accelerometer Sensor*, n.d.)

Gyroscope

A gyroscope sensor is able to sense angular orientation while accelerometer only detects rotation changes of device. As an accelerometer, gyroscope sensor also measures angular velocity for the three coordinate axes (x, y and z) so we can observe that dimensional changes when device rotates to particular axis.(Liu, 2013)

Fig **3.3** shows that how gyroscope sensor detects angular orientation step by step. Normally, a drive arm vibrates in a particular direction until any rotation changes. When the gyroscope sensor is rotated, the Coriolis force acts on the drive arms, producing vertical vibration.

The stationary part bends due to vertical drive arm vibration and producing a sensing motion in the sensing arms. The motion of a pair of sensing arms produces a potential difference from which angular velocity is sensed. The angular velocity is converted to, and output as, an electrical signal.

The gyroscope sensor is frequently used in mobile sensing applications such as human activity recognition, activity tracking, mobile games, etc. We also utilized the gyroscope sensor separately and together with accelerometer sensor in experiments to analyze the effects on sensor fingerprinting.

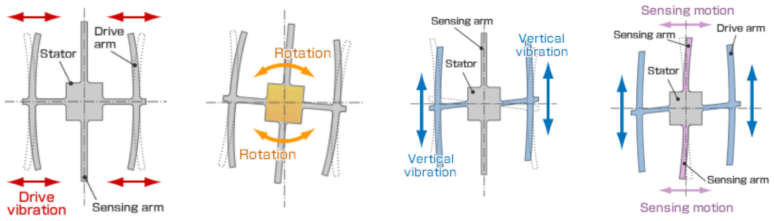

Figure **3.3**: The working mechanism of MEMS gyroscope sensor (*Gyroscope Sensor Working Mechanism*, n.d.)

Additionally, we also utilized other sensor types, which exist in CrowdSignals.io dataset and we see that most of them detect ambient conditions and are affected by environmental conditions. Moreover, conditions of living spaces such as humidity, pressure, etc. are mostly the same for users who share similar environments so we utilized these sensors with accelerometer to demonstrate that using only motion sensors is efficient to detect devices.

In addition to the environmental dependency of sensors, battery consumption and real-time performance were considered as an important issues, so we excluded location sensors from the early experiments. Moreover, observing users' location and constantly sending data to a remote system requires more Internet bandwidth which in turn impacts battery consumption. However, we utilized pressure, light and gravity sensors with accelerometer sensors in order to obtain similar conclusions with previous works.

Gravity

The gravity sensor measures the acceleration effect of Earth's gravity on the device sensor and helps to remove linear acceleration from the data which is measured by motion sensors such as accelerometer, gyroscope and magnetometer. The gravity sensor provides measurements the x, y and z axes like the other motion sensors such as accelerometer, gyroscope.

In experiments, we combined gravity sensor with accelerometer rather than using as separately in order to verify that using only accelerometer is sufficient to identify devices uniquely or should be combined with other sensors. However, not only gravity sensor but also pressure and light sensors utilized are with accelerometer sensor only for the same reason in experiments.

Pressure

The pressure (barometer) sensor can measure atmospheric pressure by how high the device is above sea level. The pressure sensor is affected by altitude changes which means that such a sensor can provide highly accurate information about vertical elevation. Therefore, pressure sensor can be used in various purposes such as indoor navigation, weather conditions, etc.

In experiments, accelerometer sensor is selected as a base sensor due to hardware imperfection during the manufacturing process so devices can be distinguished easily with this faultiness. Moreover, related works also focus on motion sensors rather than others but we need to observe the experiment results of combining accelerometer sensors with other such as pressure, light and gravity to testify using only motion sensors is more efficient than others.

Therefore, pressure sensor is used with accelerometer in various experiments and the results are explained in details in Section 4.4

The CrowdSignal.io dataset provides a list containing the pressure readings in mercury millibars with timestamps in nanoseconds at which each reading was observed.

Light

The light sensor measures the ambient light level and adjusts brightness of the device screen accordingly. Therefore, analyzing light sensor data from users' smartphones while they are conducting daily behaviors may contribute to fingerprinting system when used with accelerometer sensor. For instance, in study (Ali et al., 2019), a new mobile authentication system is proposed to detect user anomalies by only using light sensor.

The CrowdSignal.io dataset provides a list of ambient light readings with timestamps in nanoseconds at which each reading was observed.

3.2 Data Preprocessing

The CrowdSignals.io dataset consists of multiple files, which are categorized by sensor type and each row of these files includes a list of sensor recordings collected at the same time interval, 1 second. Firstly, we had to merge these files in the preprocessing step using the timestamp information. Secondly, since this is a streaming data, we work on windows of data to extract features, such as average, maximum, minimum, etc., from specific window sizes. In Table **3.4**, we summarized the number of instances for each window size and sensor/sensor combination after data processing.

Table**3.4**: The number of instances for each window size and sensor(s) after data processing

	Sensor					
Window Size	ACC	GYRO	ACCGYRO	ACCPRS	ACCGRVT	ACCLGTH
1s	197726	94507	104800	71347	130982	144522
2s	107349	48737	54311	37133	67500	77356
5s	46995	20930	23442	16096	28848	35548
10s	26257	11458	12807	8816	15605	20374
15s	23405	10343	11429	8208	13746	18544

However, reading all files for each window size caused a long preprocessing time, espe-

cially working with limited resources, such as memory size, number of CPU cores or disk capacity. Therefore, we read the data from raw files, transformed and exported as a H2O dataframe (*Data In H2O.ai*, n.d.), thus we reduced the file sizes 10 times from the original dataset and increased the data processing speed.

3.3 Feature Extraction

Feature extraction is a critical step of machine learning process to detect informative variables, those correlated with the result, so a model can produce better results while keeping over-fitting risk as minimum. The original dataset contains streaming data of motion sensors and only have raw values from three axes of the sensors with the timestamp and label information. Accelerometer and gyroscope readings are provided in a vector which includes readings from x, y and z axes. We also calculated the magnitude value given in Equation 3.1 for both sensors which makes it possible to integrate sensor information independent of the phone orientation.

$$|\overrightarrow{a}(t)| = \sqrt{a_x^2 + a_y^2 + a_z^2} \tag{3.1}$$

Moreover, we extracted various features, such as minimum, maximum, etc. from the readings of three axes and magnitude values. We extracted the features which are given in Table **3.5** for 5 (1,2,5,10 and 15 seconds) different window sizes.

Detecting the optimal window size for recognition is also important before deploying ML models for preventing over-consumption of mobile device resources. For instance, collecting accelerometer data once in a second may increase both internet usage and battery usage of a mobile device due to transferring data between a mobile application and remote server constantly and sampling the sensor at a high rate. Therefore, an application should keep the balance between resource consumption of a mobile device and model performance while identifying devices.

In this thesis, we used different types of features such as zero crossing rate, skewness, kurtosis, etc., since our plan is to provide a real-time detection framework that works accurate on both time and frequency domains.

Table**3.5**: Extracted feature list for all sensors

Feature	Description	Formula
Mean	The mean of the values	$\bar{x} = \frac{1}{N}\sum_{i=1}^{N}(x_i)$
Min	The minimum of the values	$Min = \min_{x_1,...x_N}(x_i)$
Max	The maximum of the values	$Max = \max_{x_1,...x_N}(x_i)$
Standard Deviation	Standard deviation of the values	$\sigma = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \bar{x})^2}$
Skewness	Unbiased skew over values	$\gamma = \frac{1}{N}\sum_{i=1}^{N}(\frac{x_i - \bar{x}}{\sigma})^3$
Kurtosis	Unbiased kurtosis over values	$\beta = \frac{1}{N}\sum_{i=1}^{N}(\frac{x_i - \bar{x}}{\sigma})^4 - 3$
RMS	root mean square of values	$RMS = \sqrt{\frac{1}{N}\sum_{i=1}^{N}((x_i)^2}$
ZCR	The rate of axis changes sign from positive to negative or back	$Z = \frac{1}{2N}\sum_{i=1}^{N}(sgn(x_(i)) - sgn(x_(i-1))\}$ where $sgn(x(i)) = \begin{pmatrix} 1, \mathrm{x}_i(n) >= 0 \\ -1, \mathrm{x}_i(n) < 0 \end{pmatrix}$

3.4 Classification And Performance Metrics

In this thesis, we utilize three supervised classifiers (Random Forest(*H2O.ai Distributed Random Forest*, n.d.), Generalized Linear Model(*H2O.ai Generalized Linear Model (GLM)*, n.d.) (GLM), Gradient Boosting Machine(*H2O.ai Gradient Boosting Machine (GBM)*, n.d.) (GBM)) and Artificial Neural Network (ANN)(*H2O Artificial Neural Network*, n.d.) to identify devices based on accelerometer and gyroscope sensors. One of the reason for selecting these classifiers was that both random forest and GBM are classifiers that utilize boosting for improving detection. Random forest was used in related studies, however GBM, GLM and ANN were not evaluated in previous studies. Hence, we wanted to investigate the performance with different classifiers.

Initially, we generated various datasets for 5 different window sizes (1, 2, 5, 10 and 15 seconds) and all of them consist of labeled records from 25 users. After that, each dataset was split into 10 folds that must contain records from all users (classes), so we applied stratified sampling which assigns roundly the same percentage of records from each class similar to the entire dataset.

The entire process was the same for all classifiers but we also tested particular classifier parameters to build efficient models on training. In random forest, the number of trees, which specifies the decision tree size in the forest, can be changed. We repeated the experiments for 6 different (5, 10, 15, 20, 25, 30) tree sizes per each time window. Our aim is to detect optimal number of trees which keeps balance between model complexity and performance. In GLM, the maximum number of iterations, which specifies the n^{th} iteration that terminates model training, can be parameterized. In ANN, we tested the epochs, which specifies the n^{th} iteration and the size of hidden layers. In the experiments, we utilized 6 different (5, 10, 15, 20, 25, 30) iterations per each time window so we were able to analyze the results with increasing number of iterations.

In the experiments, H2O.ai(*H2O.ai Machine Learning Platform*, n.d.) machine learning framework is used in all phases from dataset preparation to model performance evaluation. We evaluated with the metrics those provided by H2O.ai to measure the performance of multinominal/multiclass models. We utilized accuracy, precision and recall metrics which are given in Equations 3.2, 3.3 and 3.4.

$$Accuracy = \frac{Number\ of\ correct\ predictions}{Number\ of\ total\ predictions} \tag{3.2}$$

In addition to accuracy metric, average precision and recall are measured to evaluate overall performance of the model via a confusion matrix. Therefore, we had to calculate precision and recall per class since H2O.ai does not provide these metrics when dataset consists of more than two labels. In equation 3.3 and 3.4, TP_i represents the number of true positives and FP_i is defined as the number of false positives, likewise FN_i refer to the number of false negatives for class i.

$$precision_i = \frac{(TP_i)}{(TP_i + (FP_i)} \tag{3.3}$$

$$recall_i = \frac{(TP_i)}{(TP_i + (FN_i)} \tag{3.4}$$

As the last step, we calculated average precision and recall in order to evaluate the model performance as follows. Let m is the total number of classes:

$$Avg\ Precision = \frac{\sum_{i=1}^{m} precision_i}{m} \tag{3.5}$$

$$Avg\ Recall = \frac{\sum_{i=1}^{m} recall_i}{m} \tag{3.6}$$

4 PERFORMANCE EVALUATION

In this section, we present the results of the experiments. As discussed in Section 3, we have different parameters that are considered in the experiments and full list of parameters is given in Table **4.1**. Considering different sensors (5 sensors and combination), five different window sizes or number of iterations, 90 models were built for all classifiers.

In the following subsections, we present the results per sensor and their combination, considering the impact of different parameters. We start with the results obtained with the accelerometer sensor, and next we provide the results of the gyroscope sensor, followed by accelerometer sensor combination with gyroscope, pressure, light, gravity sensors. Results of each experiment are presented in terms of accuracy, precision and recall. Before discussing on the results, please note that comma (,) used as a decimal separator rather than dot(.) due to localization issues in accuracy comparison figures such as Fig. **4.1**, Fig. **4.2**.

Table**4.1**: Experiment setup parameters

Parameter	**Values**	**Extras**
Classifiers	GBM, GLM, RF	RF, GBM: 5, 10, 15, 20, 25, 30 trees GLM: 5, 10, 15, 20, 25, 30 iterations
Features	Mean, min, max, standard deviation, skewness, kurtosis, root mean square zero crossing rate	Calculated from X, Y, Z axes, magnitude for each axes and light, pressure values
Window Sizes	1, 2, 5, 10 and 15 seconds	
Sensors	ACC, GYRO, ACCGYRO, ACCPRS, ACCGRVT, ACCLGTH	

4.1 Accelerometer

In this section, we aim to analyze the effects of window size and classifier on the performance of the model which was created by the data from only accelerometer sensor.

In Fig. **4.1**, we compare the performance results of Random Forest, GBM and GLM

classifiers for 5 different window sizes which are presented in Table **4.1**.

The x-axis shows the tree size whereas y-axis displays the accuracy values, varying between 0 and 1. When we compare the accuracy results, GBM obtains the highest accuracy as 0.93 on 1 and 2 second windows with a small difference. The results show that Random Forest also works with accelerometer sensor efficiently as well as GBM and classifies 91% of the devices correctly. When the results of GLM are compared to others, we obtain the lowest accuracy as 0.41 on average considering every window size. The reason is that GLM is more convenient to handle simple linear problems which can be analytically solved rather than complex ones. Furthermore, we inferred that increasing the window size causes a slight decrease after 2 seconds for Random Forest and GBM classifiers. As a result, GBM is observed to be the best performing classifier to identify devices when compared to others.

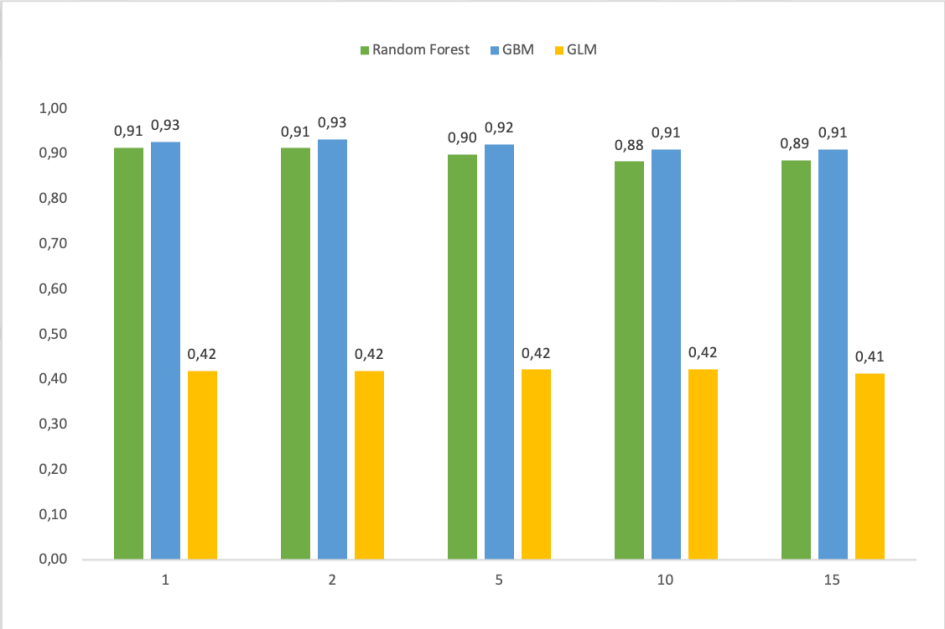

Figure **4.1**: ACC accuracy comparison of 3 classifiers for 5 different window

Table**4.2**: ACC accuracy comparison of RF, GBM and GLM classifiers based on 5 different window and 6 different tree/number of iteration size

		Tree Size / Number of Iterations					
Window Size	**Classifier**	**5**	**10**	**15**	**20**	**25**	**30**
1s	**RF**	0.903	0.909	0.911	0.912	0.912	0.913
	GBM	0.909	0.916	0.920	0.923	0.924	0.926
	GLM	0.417	0.418	0.418	0.418	0.418	0.418
2s	**RF**	0.900	0.908	0.910	0.912	0.912	0.913
	GBM	0.913	0.919	0.924	0.927	0.928	0.931
	GLM	0.417	0.418	0.419	0.419	0.419	0.419
5s	**RF**	0.887	0.893	0.896	0.898	0.898	0.899
	GBM	0.900	0.908	0.913	0.917	0.920	0.921
	GLM	0.420	0.422	0.422	0.422	0.422	0.422
10s	**RF**	0.869	0.876	0.879	0.881	0.882	0.883
	GBM	0.881	0.894	0.900	0.904	0.906	0.910
	GLM	0.419	0.422	0.422	0.422	0.422	0.422
15s	**RF**	0.871	0.878	0.879	0.883	0.885	0.886
	GBM	0.886	0.894	0.901	0.904	0.906	0.911
	GLM	0.413	0.414	0.414	0.414	0.414	0.414

4.2 Gyroscope

In this section, we evaluate the impact of both the classifier and the window size on model performance as in section 4.1 but here we use the gyroscope sensor.

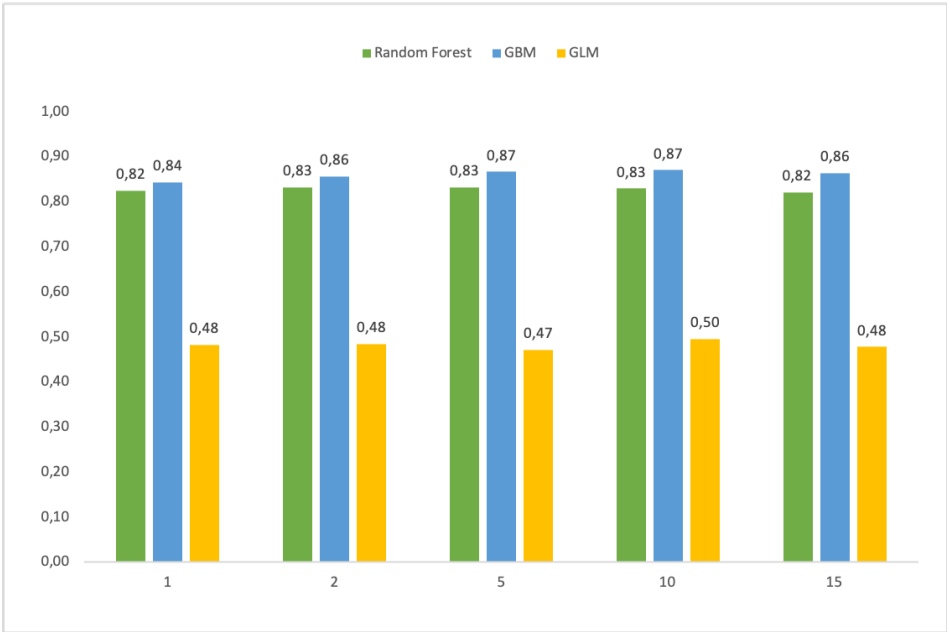

Figure **4.2**: GYRO accuracy comparison of 3 classifiers for 5 different window

Table**4.3**: GYRO accuracy comparison of RF, GBM and GLM classifiers based on 5 different window and 6 different tree/number of iteration size

		Tree Size / Number of Iterations					
Window Size	**Classifier**	**5**	**10**	**15**	**20**	**25**	**30**
1s	**RF**	0.809	0.821	0.822	0.824	0.824	0.824
	GBM	0.817	0.827	0.832	0.837	0.841	0.843
	GLM	0.481	0.481	0.483	0.483	0.483	0.483
2s	**RF**	0.815	0.827	0.829	0.830	0.831	0.831
	GBM	0.826	0.837	0.844	0.848	0.852	0.856
	GLM	0.479	0.484	0.484	0.484	0.484	0.484
5s	**RF**	0.816	0.826	0.828	0.830	0.832	0.832
	GBM	0.832	0.842	0.853	0.858	0.860	0.866
	GLM	0.461	0.470	0.471	0.472	0.472	0.472
10s	**RF**	0.808	0.820	0.826	0.829	0.828	0.829
	GBM	0.841	0.850	0.856	0.862	0.868	0.870
	GLM	0.482	0.492	0.494	0.494	0.494	0.495
15s	**RF**	0.792	0.814	0.820	0.821	0.819	0.819
	GBM	0.827	0.842	0.848	0.859	0.856	0.863
	GLM	0.464	0.476	0.477	0.478	0.478	0.477

In Fig. **4.2**, y-axis shows the accuracy as a performance metric and x-axis displays the window size. The highest accuracy is 91%, obtained by the GBM classifier on 1 second window and Random Forest achieved the second best accuracy 83% in almost every window size. However, GLM exhbits the lowest accuracy in every window size, maximum 31%, same as the results in Section 4.1. The performance of GBM models were affected positively by increasing window size starting with 2 seconds whereas Random Forest stays the same. However, 2 second windows led to significant increase on the performance of GLM classifier and enhanced accuracy from 20% to 30%. In conclusion, GBM provides the best accuracy, 91% same as the accelerometer on 1 second window and GLM is inadequate to classify devices for any window size.

4.3 Accelerometer And Gyroscope

In this section, our aim is to explore the impact of combining accelerometer and gyroscope features that are mentioned in Table **3.5** on the model performance.

Fig. **4.3** shows that all the classifiers exhibit better results while using both sensors together. Again, GBM is the most efficient classifier with the highest accuracy as 98% with 15 second window size.

However, increasing window size causes a decrease in the number of instances for some

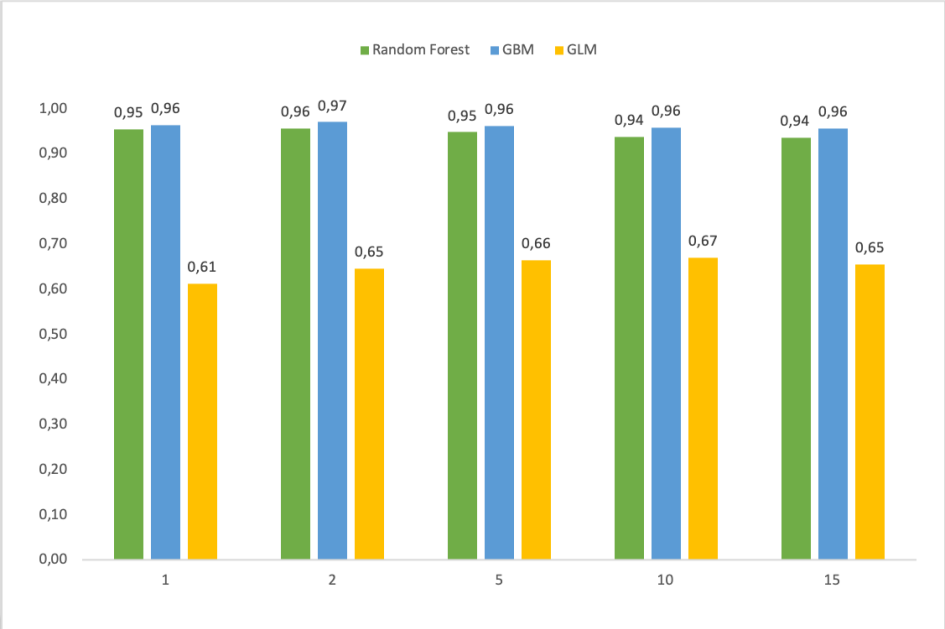

Figure **4.3**: ACCGYRO accuracy comparison of 3 classifiers for 5 different window

users since some users have less data. For this reason, 5 second window was utilized with the aim of comparing the performance of sensors reliably in Fig. **4.7**. When we consider different window sizes from 1 to 10 in Fig. **4.3**, GBM provides the highest accuracy, 97% on 1 second window. Besides, the performance of Random Forest is slightly smaller than GBM: 96% is the maximum accuracy of all the Random Forest models. However, Furthermore, the worst performance is obtained by GLM once again but better than the previous GLM models created by using accelerometer or gyroscope.

Table**4.4**: ACCGYRO accuracy comparison of RF, GBM and GLM classifiers based on 5 different window and 6 different tree/number of iteration size

		Tree Size / Number of Iterations					
Window Size	**Classifier**	**5**	**10**	**15**	**20**	**25**	**30**
1s	**RF**	0.949	0.952	0.953	0.953	0.954	0.954
	GBM	0.953	0.957	0.959	0.961	0.962	0.964
	GLM	0.607	0.612	0.613	0.613	0.613	0.613
2s	**RF**	0.949	0.953	0.955	0.956	0.957	0.956
	GBM	0.958	0.962	0.965	0.967	0.969	0.970
	GLM	0.640	0.645	0.647	0.646	0.646	0.646
5s	**RF**	0.937	0.943	0.945	0.946	0.947	0.948
	GBM	0.949	0.954	0.959	0.960	0.962	0.963
	GLM	0.657	0.663	0.664	0.664	0.664	0.664
10s	**RF**	0.922	0.930	0.933	0.935	0.937	0.938
	GBM	0.942	0.948	0.953	0.954	0.955	0.958
	GLM	0.661	0.669	0.669	0.669	0.669	0.669
15s	**RF**	0.923	0.929	0.932	0.934	0.935	0.936
	GBM	0.940	0.944	0.949	0.952	0.955	0.956
	GLM	0.644	0.654	0.654	0.655	0.655	0.655

4.4 Accelerometer And Pressure

In this section, our aim is to explore the impact of combining accelerometer and pressure features that are mentioned in Table **3.5** on the model performance.

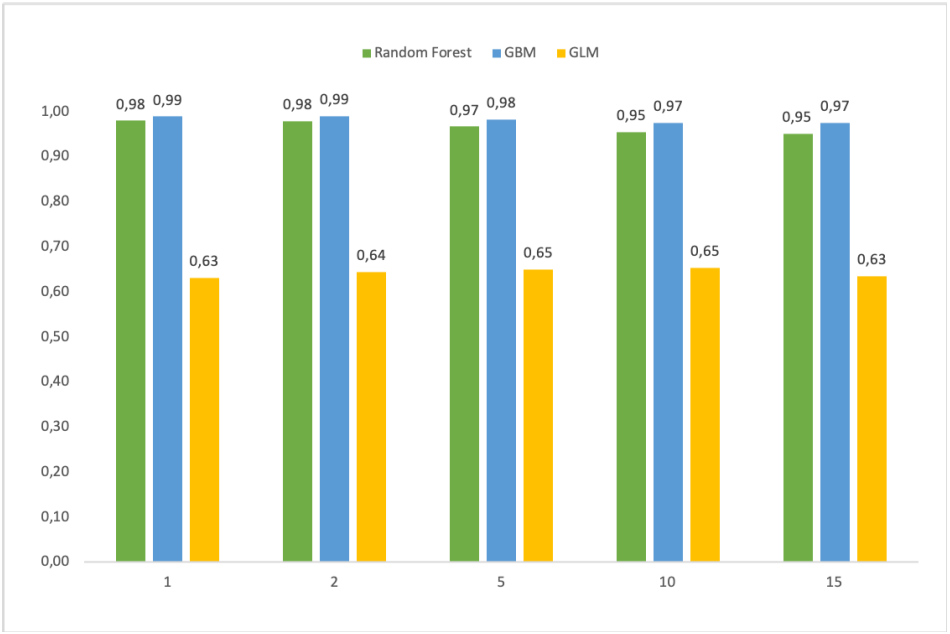

Figure **4.4**: ACCPRS accuracy comparison of 3 classifiers for 5 different window

Table**4.5**: ACCPRS accuracy comparison of RF, GBM and GLM classifiers based on 5 different window and 6 different tree/number of iteration size

		Tree Size / Number of Iterations					
Window Size	**Classifier**	**5**	**10**	**15**	**20**	**25**	**30**
1s	**RF**	0.974	0.977	0.978	0.978	0.979	0.980
	GBM	0.980	0.983	0.985	0.987	0.988	0.989
	GLM	0.622	0.630	0.631	0.631	0.631	0.631
2s	**RF**	0.971	0.976	0.977	0.978	0.979	0.979
	GBM	0.981	0.986	0.987	0.988	0.989	0.990
	GLM	0.626	0.642	0.643	0.644	0.644	0.644
5s	**RF**	0.959	0.963	0.966	0.966	0.966	0.966
	GBM	0.967	0.972	0.976	0.979	0.981	0.982
	GLM	0.634	0.648	0.649	0.650	0.650	0.650
10s	**RF**	0.941	0.947	0.951	0.953	0.954	0.954
	GBM	0.958	0.965	0.967	0.971	0.973	0.974
	GLM	0.634	0.651	0.652	0.653	0.653	0.653
15s	**RF**	0.941	0.944	0.948	0.950	0.951	0.951
	GBM	0.960	0.964	0.969	0.970	0.973	0.975
	GLM	0.627	0.633	0.634	0.634	0.634	0.634

4.5 Accelerometer And Light

In this section, our aim is to explore the impact of combining accelerometer and light features that are mentioned in Table **3.5** on the model performance.

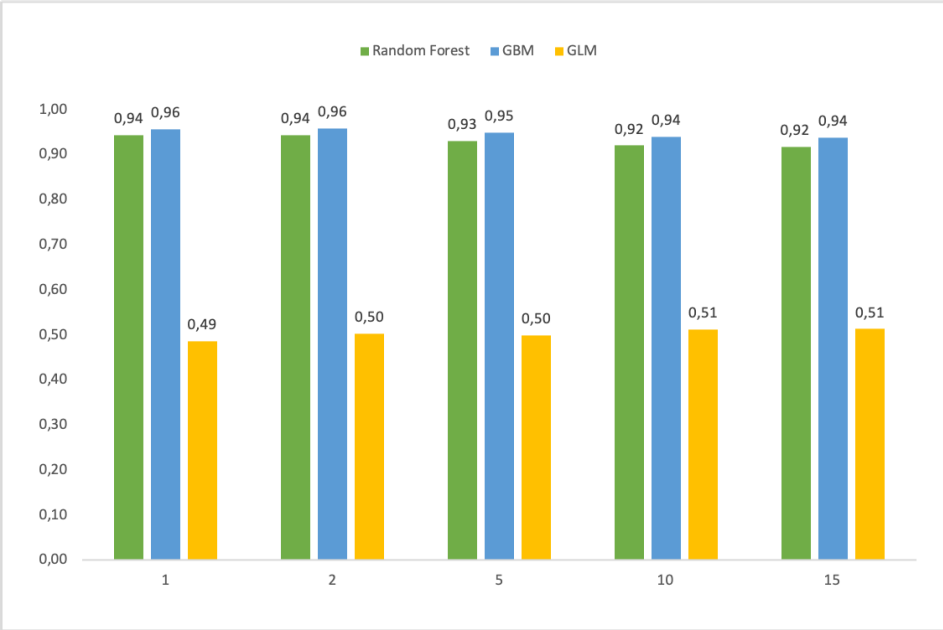

Figure **4.5**: ACCLGTH accuracy comparison of 3 classifiers for 5 different window

Table**4.6**: ACCLGTH accuracy comparison of RF, GBM and GLM classifiers based on 5 different window and 6 different tree/number of iteration size

		Tree Size / Number of Iterations					
Window Size	**Classifier**	**5**	**10**	**15**	**20**	**25**	**30**
1s	**RF**	0.934	0.940	0.943	0.943	0.944	0.944
	GBM	0.942	0.947	0.950	0.953	0.955	0.956
	GLM	0.482	0.486	0.486	0.486	0.486	0.486
2s	**RF**	0.933	0.939	0.942	0.943	0.943	0.943
	GBM	0.943	0.949	0.952	0.955	0.956	0.958
	GLM	0.499	0.502	0.502	0.502	0.502	0.502
5s	**RF**	0.918	0.925	0.929	0.930	0.931	0.931
	GBM	0.929	0.937	0.942	0.944	0.947	0.948
	GLM	0.492	0.498	0.498	0.498	0.498	0.498
10s	**RF**	0.906	0.913	0.916	0.918	0.919	0.920
	GBM	0.921	0.926	0.932	0.935	0.937	0.940
	GLM	0.500	0.511	0.512	0.512	0.512	0.512
15s	**RF**	0.902	0.911	0.916	0.916	0.916	0.917
	GBM	0.912	0.922	0.928	0.933	0.934	0.938
	GLM	0.495	0.510	0.512	0.512	0.513	0.513

4.6 Accelerometer And Gravity

In this section, our aim is to explore the impact of combining accelerometer and gravity features that are mentioned in Table **3.5** on the model performance.

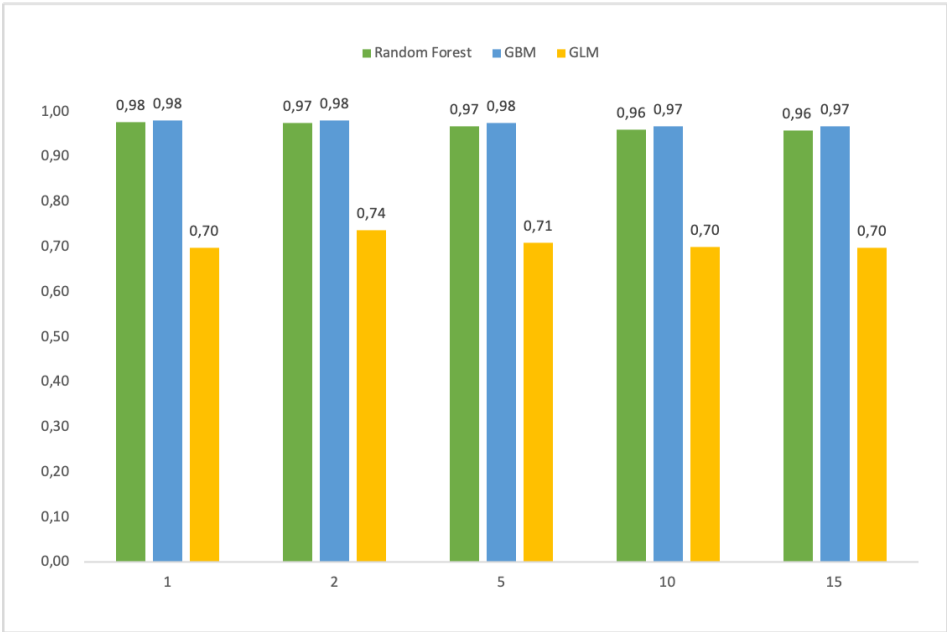

Figure **4.6**: ACCGRVT accuracy comparison of 3 classifiers for 5 different window

Table**4.7**: ACCGRVT accuracy comparison of RF, GBM and GLM classifiers based on 5 different window and 6 different tree/number of iteration size

		Tree Size / Number of Iterations					
Window Size	**Classifier**	**5**	**10**	**15**	**20**	**25**	**30**
1s	**RF**	0.974	0.976	0.976	0.976	0.977	0.976
	GBM	0.974	0.977	0.979	0.980	0.981	0.981
	GLM	0.683	0.695	0.696	0.697	0.697	0.697
2s	**RF**	0.970	0.973	0.973	0.973	0.974	0.974
	GBM	0.974	0.976	0.978	0.980	0.981	0.981
	GLM	0.730	0.736	0.737	0.737	0.737	0.737
5s	**RF**	0.960	0.965	0.967	0.968	0.968	0.968
	GBM	0.964	0.969	0.973	0.974	0.975	0.975
	GLM	0.698	0.706	0.707	0.708	0.709	0.709
10s	**RF**	0.951	0.954	0.957	0.958	0.960	0.959
	GBM	0.954	0.959	0.965	0.965	0.967	0.968
	GLM	0.687	0.696	0.697	0.698	0.699	0.699
15s	**RF**	0.949	0.953	0.955	0.957	0.958	0.958
	GBM	0.953	0.957	0.962	0.965	0.967	0.967
	GLM	0.676	0.692	0.694	0.695	0.696	0.698

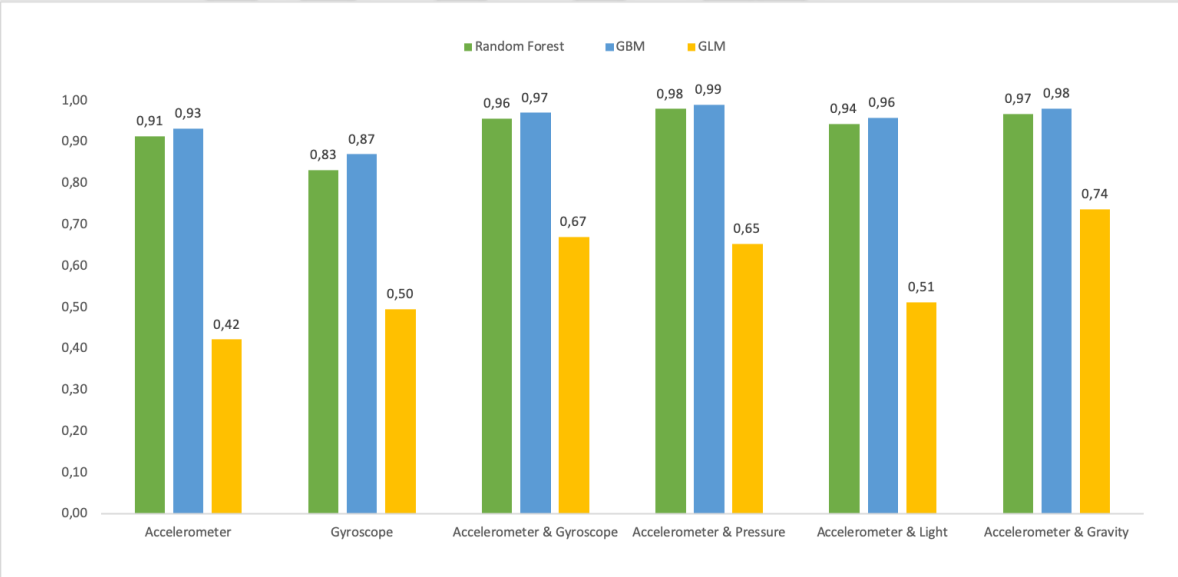

Figure **4.7**: Accuracy comparison for 3 different classifiers according to sensor types and combinations

Finally, we compare the performance of classifiers by using all sensors and their combination in Fig. **4.7**. The y-axis represents accuracy as in the previous graphs and x-axis shows which sensor(s) is used while training models by 3 different classifiers.

When comparing classifiers, best performance selected between 5 different window size. We clearly see that using both accelerometer and gyroscope sensors increases the accuracy at least by 6% for GBM and Random Forest.

Additionally, we also utilized several performance metrics in Table **4.8** such as precision, recall, etc. mentioned in Section 3.4. Table **4.8** shows the performance evaluation of Random Forest model for 6 different tree size on 2 second window. The results demonstrate that increasing tree size enhances accuracy, precision and recall metrics and improves all of them 1% in average.

Table**4.8**: Performance evaluation of Random Forest model for 6 different tree size on 2 second window

Tree Size	**Accuracy**	**Precision**	**Recall**
5	0.949	0.957	0.901
10	0.953	0.962	0.905
15	0.955	0.965	0.914
20	0.956	0.965	0.907
25	0.957	0.966	0.908
30	0.956	0.966	0.908

In addition to performance metrics, we also analyzed the importance of features in Fig. **4.8**, Fig. **4.9**, Fig. **4.10**, Fig. **4.11**, Fig. **4.12** and Fig. **4.12** for Random Forest models while using different sensor combinations on 2 second window. Considering feature importance rankings, we can infer that pressure and light sensor features are not sufficient as accelerometer features. However, gravity sensor features are included in Top 10 when combined with accelerometer sensor since accelerometer sensor also contains the acceleration effect of Earth's gravity. Therefore, we can infer that accelerometer sensor can be used separately to identify devices uniquely.

Lastly, we only utilized Top 20 features and repeated experiments on 2 second window for Random Forest in order to analyze the performance impact of significant features.

In Fig. **4.14**, y-axis represents accuracy as in the previous graphs and x-axis shows which sensor(s) is used while training Random Forest models by Top 20 features. Considering the results, eliminating other features decreases ACCGYRO and ACCPRS accuracy between 1% - 2% while not effecting other sensor combinations.

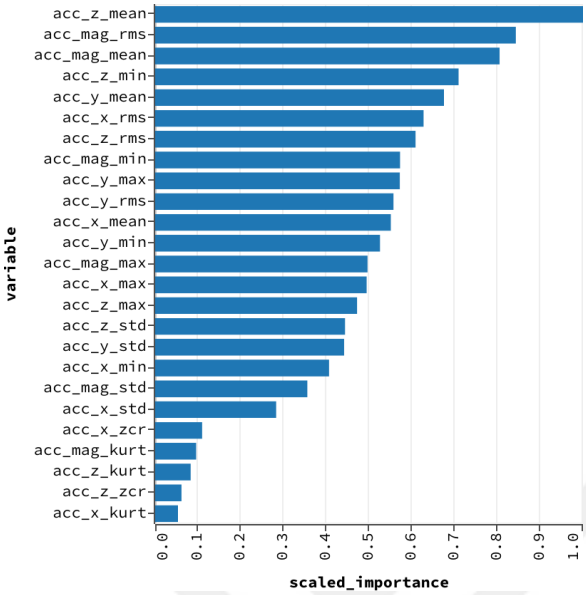

Figure **4.8**: ACC feature importance for Random Forest on 2 second window

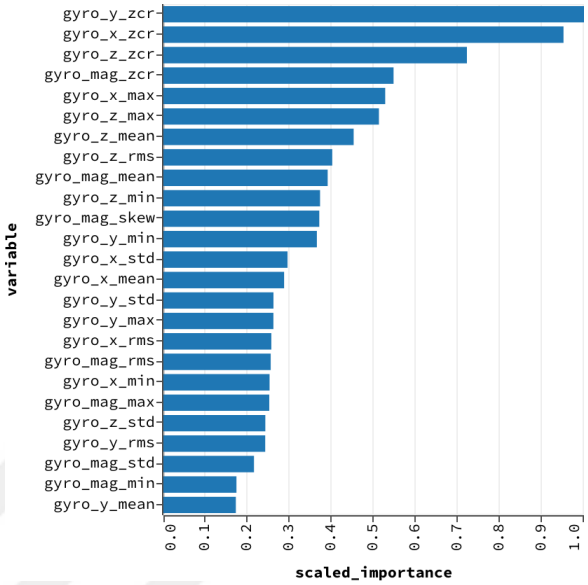

Figure **4.9**: GYRO feature importance for Random Forest on 2 second window

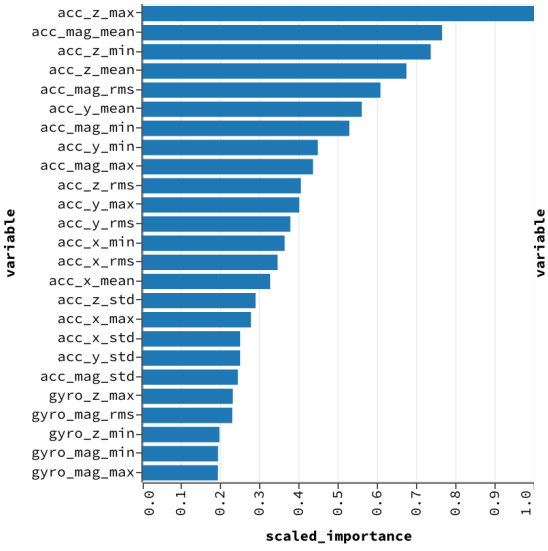

Figure **4.10**: ACCGYRO feature importance for Random Forest on 2 second window

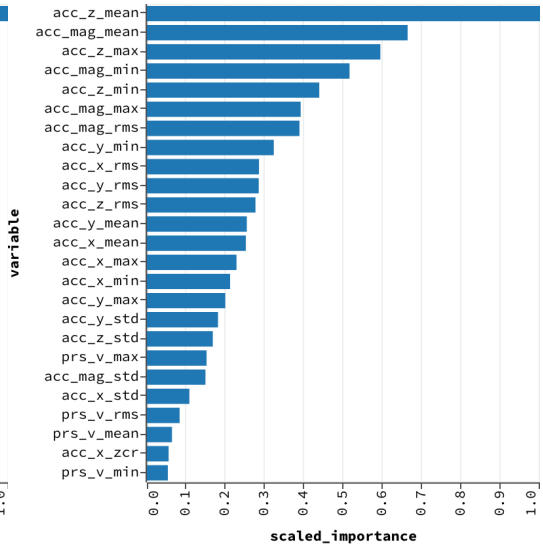

Figure **4.11**: ACCPRS feature importance for Random Forest on 2 second window

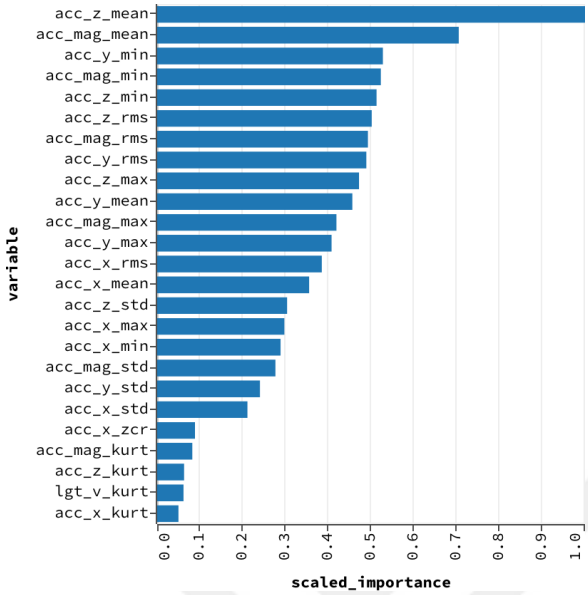

Figure **4.12**: ACCLGTH feature importance for Random Forest on 2 second window

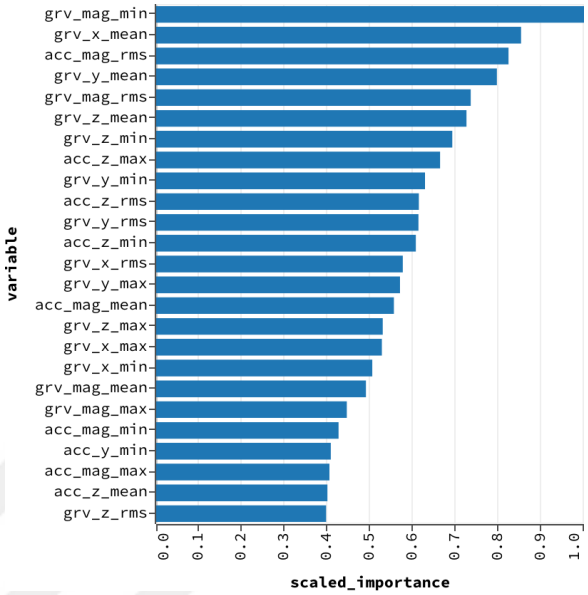

Figure **4.13**: ACCGRVT feature importance for Random Forest on 2 second window

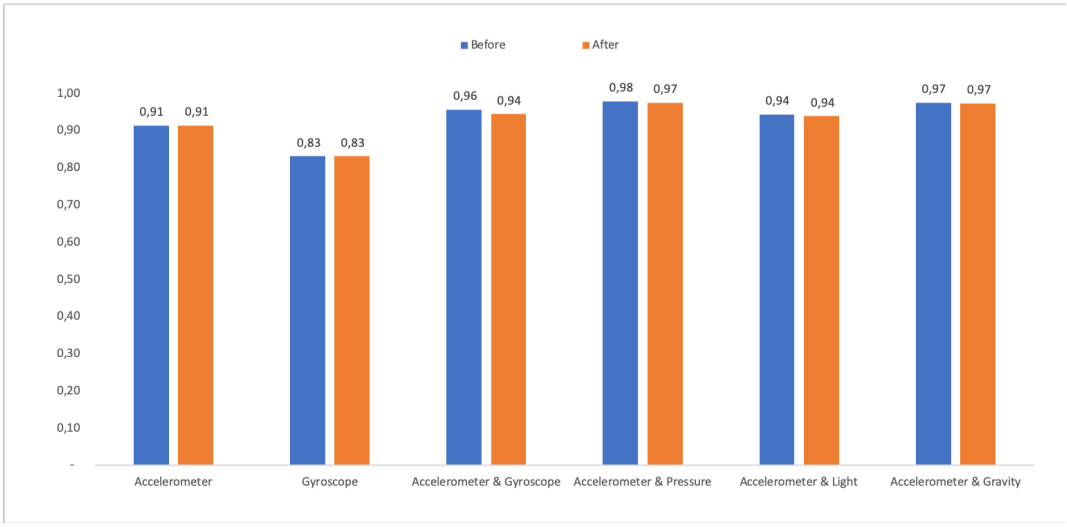

Figure **4.14**: Before/After accuracy comparison on 2 second window according to sensor types and combinations

4.7 Artificial Neural Network (ANN)

In Chapter 4 from Section 4.1 to Section 4.6, we studied on 3 different ML models; 1) Random Forest, 2) GBM and 3) GLM. However, we also analyzed the effects of ANN models on sensor fingerprinting problem. In early experiments, we see that accelerometer and gyroscope sensors play a key role while identifying devices uniquely so we only created ANN models for these two sensors.

H2O's ANN provides a multi-layer feedforward artificial neural network that is trained with stochastic gradient descent using back-propagation. The network can contain a large number of hidden layers consisting of neurons with tanh, rectifier, and maxout activation functions.(*H2O Artificial Neural Network*, n.d.)

In experiments, we present the results per sensor and their combination, considering the impact of different parameters. We start with the results obtained with the accelerometer sensor, and next we provide the results of the gyroscope sensor, followed by accelerometer sensor combination with gyroscope. Results of each experiment are presented in terms of accuracy, precision and recall. Table **4.9** shows that experiment setup for 5 different window size while creating ANN models.

Table**4.9**: ANN Experiment setup parameters

Parameter	**Values**	**Extras**
Training Parameters	epochs=10000 hidden=[32, 32, 32] activation=rectifier	epochs: dataset iteration size hidden: Hidden layer sizes activation: activation function
Features	Mean, min, max, standard deviation, skewness, kurtosis, root mean square zero crossing rate	Calculated from X, Y, Z axes, magnitude for each axes
Window Sizes	1, 2, 5, 10 and 15 seconds	
Sensors	ACC, GYRO, ACCGYRO	

Considering the results in Fig. **4.15**, ANN method does not perform as Random Forest and GBM models when compared to the results in Section 4.1, 4.2 and 4.3. Based on the results, using both accelerometer and gyroscope sensors increased accuracy at least 8% in every window size. The highest accuracy is obtained from 10s and 15s window sizes by using both accelerometer and gyroscope sensors.

In Fig. **4.15** training logloss(*H2O Performance and Prediction - Logloss*, n.d.), we infer that logloss decreases from 8.80 to 4.20 at 515 number of iteration (epochs) and remains stable so training is terminated by H2O at that point.

Figure **4.15**: Accuracy comparison for 5 different window according to sensor types and combinations

In Fig. **4.17**, we analyzed a variable importance diagram for one of the best accurate models and infer that Top 10 features consist of both gyroscope and accelerometer features. Moreover, Top 10 features are not only extracted from a particular axes but also all of 3 axes (x, y and z).

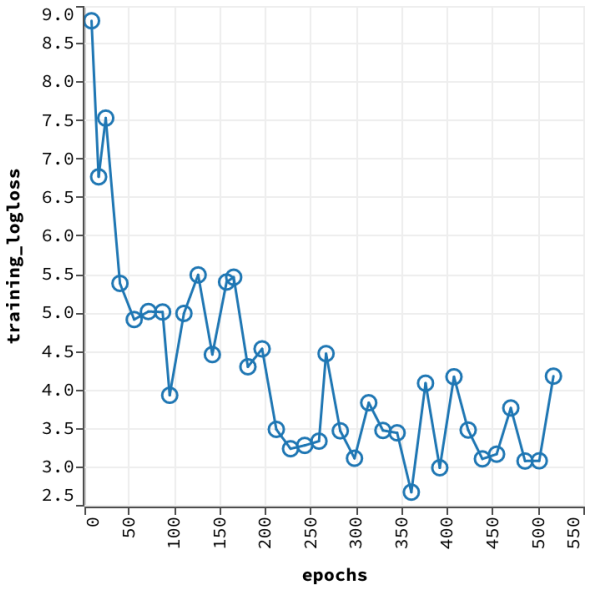

Figure **4.16**: ACCGYRO - ANN training logloss diagram for 10 second window size

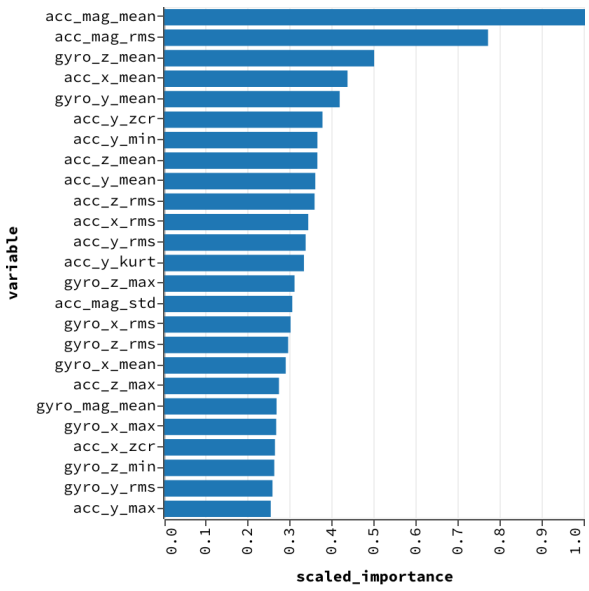

Figure **4.17**: ACCGYRO - ANN feature importance for 10 second window size

5 CONCLUSION AND FUTURE WORK

In this thesis, we investigated user/device identification using sensor fingerprinting particularly on mobile devices. We particularly investigated the performance of identification using 5 different sensors, accelerometer, gyroscope, pressure, gravity and light which are commonly available on smart phones using a set of time and frequency domain features. We utilized a large dataset, named as CrowdSignals.io, which is larger than 25 GB, and consists of sensor data from 25 distinct devices. We also investigated the performance with four different classification methods, namely random forest, gradient boosting machine, generalized linear model and artificial neural network using 64 features, also considering the effect of different time window sizes. Experiment results show that random forest and GBM classifier exhibit similar performances, while GLM performs much worse than the others. When only accelerometer is used, GBM achieves 91% accuracy on 1 and 2 second time windows, while random forest performs with 89% accuracy in these cases, and GLM with 31%. Similar results were achieved with different window sizes. When gyroscope is used, slightly lower results were achieved, ranging between 82 to 91% accuracy for random forest and GBM classifiers. However, GBM with 1 second window size exhibited a similar performance as the gyroscope case: 91% accuracy. When both sensors are considered, accuracy increases to 97% for GBM and 96% for random forest.

REFERENCES

Ali, M., Aiken, W. and Kim, H. (2019). The light will be with you. always – a novel continuous mobile authentication with the light sensor, pp. 560–561.

Android Accelerometer Sensor Overview (n.d.). `https://developer.android.com/guide/topics/sensors/sensors_overview`.

Bojinov, H., Michalevsky, Y., Nakibly, G. and Boneh, D. (2014). Mobile device identification via sensor fingerprinting, *CoRR* **abs/1408.1416**.

CrowdSignals.io (n.d.). `http://crowdsignals.io/`.

CrowdSignals User's Reference Document (n.d.). `http://crowdsignals.io/docs/AlgoSnap%20Sensor%20Data%20\Types%20Description.pdf`.

Das, A., Borisov, N. and Caesar, M. (2016). Tracking mobile web users through motion sensors: Attacks and defenses, *NDSS*.

Data In H2O.ai (n.d.). `http://docs.h2o.ai/h2o/latest-stable/h2o-py/docs/data.html`.

Dey, S., Roy, N., Xu, W., Roy Choudhury, R. and Nelakuditi, S. (2014). Accelprint: Imperfections of accelerometers make smartphones trackable.

Ehatisham-ul Haq, M., Azam, M. A., Naeem, U., ur Rhman, S. and Khalid, A. (2017). Identifying smartphone users based on their activity patterns via mobile sensing, *Procedia Computer Science* **113**: 202–209.

Gyroscope Sensor Working Mechanism (n.d.). `https://www5.epsondevice.com/en/information/technical_info/gyro/index.html`.

H2O.ai Distributed Random Forest (n.d.). `http://docs.h2o.ai/h2o/latest-stable/h2o-docs/data-science/drf.html`.

H2O.ai Generalized Linear Model (GLM) (n.d.). `http://docs.h2o.ai/h2o/latest-stable/h2o-docs/data-science/glm.html`.

H2O.ai Gradient Boosting Machine (GBM) (n.d.). http://docs.h2o.ai/h2o/latest-stable/h2o-docs/data-science/gbm.html.

H2O.ai Machine Learning Platform (n.d.). https://h2o-release.s3.amazonaws.com/h2o/rel-slater/9/docs-website/h2o-py/docs/index.html.

H2O Artificial Neural Network (n.d.). http://docs.h2o.ai/h2o/latest-stable/h2o-docs/data-science/deep-learning.html.

H2O Performance and Prediction - Logloss (n.d.). http://docs.h2o.ai/h2o/latest-stable/h2o-docs/performance-and-prediction.html?highlight=logloss#logloss.

Lee, W. and Lee, R. B. (2017). Sensor-based implicit authentication of smartphone users, *2017 47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, pp. 309–320.

Liu, M. (2013). A study of mobile sensing using smartphones, *International Journal of Distributed Sensor Networks* **9**(3): 272916.
URL: *https://doi.org/10.1155/2013/272916*

Mayrhofer, R. and Gellersen, H.-W. (2007). Shake well before use: Authentication based on accelerometer data, *Pervasive*.

MEMS Accelerometer Sensor (n.d.). http://www.instrumentationtoday.com/wp-content/uploads/2011/08/MEMS-Accelerometer.jpg.

MEMS SENSORS (n.d.). https://pdfserv.maximintegrated.com/en/an/AN5830.pdf.

ul Haq, M. E., Azam, M. A., Loo, J., Shuang, K., Islam, S. M. S., Naeem, U. and Amin, Y. (2017). Authentication of smartphone users based on activity recognition and mobile sensing, *Sensors*.

Welbourne, E. and Munguia Tapia, E. (2014). Crowdsignals: A call to crowdfund the community's largest mobile dataset, pp. 873–877.

BIOGRAPHICAL SKETCH

First Name: Kadriye
Surname: Doğan

Place of Birth and Date of Birth: Merkez/SİVAS and 1989
Bachelor Degree: Çanakkale Onsekiz Mart University, Computer Engineering (2011)
Current Position: Expert Software Engineer at iyzico, since March 2015
Work Experience: Netaş (2014-2015), Compugroup Medical (2012-2014), Business Information Technology Solutions (2011-2012)