

T.R.
GEBZE TECHNICAL UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

**DRIVER BEHAVIOR ANALYSIS BASED ON VEHICLE
DIAGNOSTIC DATA**

CANER ASLAN
**A THESIS SUBMITTED FOR THE DEGREE OF
MASTER OF SCIENCE**
DEPARTMENT OF COMPUTER ENGINEERING

GEBZE
2022

T.R.
GEBZE TECHNICAL UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

**DRIVER BEHAVIOR ANALYSIS BASED ON
VEHICLE DIAGNOSTIC DATA**

CANER ASLAN
**A THESIS SUBMITTED FOR THE DEGREE OF
MASTER OF SCIENCE
DEPARTMENT OF COMPUTER ENGINEERING**

THESIS SUPERVISOR
ASSIST. PROF. DR. YAKUP GENÇ

GEBZE

2022

T.C.
GEBZE TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

ARAÇ TANI VERİLERİNDEN SÜRÜCÜ
DAVRANIŞ ANALİZİ

CANER ASLAN
YÜKSEK LİSANS TEZİ
BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

DANIŞMANI
DR. ÖĞR. ÜYESİ YAKUP GENÇ

GEBZE
2022

GEBZE TEKNİK ÜNİVERSİTESİ	YÜKSEK LİSANS JÜRİ ONAY FORMU
----------------------------------	--------------------------------------

GTÜ Fen Bilimleri Enstitüsü Yönetim Kurulu'nun 10/02/2022 tarih ve 2022/10 sayılı kararıyla oluşturulan jüri tarafından 01/03/2022 tarihinde tez savunma sınavı yapılan Caner Aslan'ın tez çalışması Bilgisayar Mühendisliği Anabilim Dalında YÜKSEK LİSANS tezi olarak kabul edilmiştir.

JÜRİ

ÜYE

(TEZ DANIŞMANI) : Dr. Öğr. Üyesi Yakup GENÇ

ÜYE

: Prof. Dr. Fatih Erdoğan SEVİLGİN

ÜYE

: Prof. Dr. Behçet Uğur TÖREYİN

ONAY

Gebze Teknik Üniversitesi Fen Bilimleri Enstitüsü Yönetim Kurulu'nun
...../...../..... tarih ve/..... sayılı kararı.

SUMMARY

The goal of this research was to find solutions to two separate sub-problems in driving behavior analysis: driver identification and behavior detection. Firstly, driver identification task was performed using artificial neural networks and vehicle diagnosis data without feature engineering which is one of the most challenging steps of machine learning methods. Along with the data sets used in the literature, the data set including rural road, urban and highway routes with 5 different drivers was collected and used experimentally. With the FCN (Fully Convolutional Network) architecture established to solve the problem, the success rate of RNN (Recurrent Neural Network)-based architectures and traditional machine learning methods in the literature has been exceeded. The window size parameter has been reduced in training and validation. At the same time, model size and the number of floating-point operations were reduced, increasing the possibility of use in in-vehicle systems. While the results are obtained with the stratified fold validation, they are compared with the timeseries segmentation method in data sets containing a single drive on the route.

Secondly, by using specially designed hardware and developed software components for the driving behavior detection problem, natural driving data for up to 3 months was collected from the personal vehicles of 7 different drivers. Unlike studies in the literature, driving behavior data is automatically labeled with the proposed autoencoder architecture and clustering method. In addition to the Euclidean distance used for similarity measurement, the effect of Mahalanobis distance and outlier data points on the clustering results was visualized using t-SNE (t-distributed stochastic neighbor embedding). Machine learning methods were used to classify labeled behavior data with 99% accuracy, and the results were validated using a timeseries segmentation technique. Also, the effect of different window sizes was investigated with the proposed FCN architecture.

Keywords: deep learning, driver identification, driver behavior analysis, timeseries classification, timeseries clustering.

ÖZET

Bu araştırmanın amacı, sürüş davranışı analizinde iki ayrı alt probleme çözüm bulmaktır: sürücü kimlik tespiti ve sürücü davranış tespiti. İlk olarak, yapay sinir ağları ile araç tanı verileri kullanılarak makine öğrenme yöntemlerinin en zorlu adımlarından olan öznitelik çıkarımı olmaksızın sürücü kimlik tespiti yapılmıştır. Literatürde kullanılan veri kümeleri ile birlikte, deneysel olarak 5 farklı sürücü ile kırsal yol, şehiriçi ve otoyol güzergahlarını içerecek şekilde toplanan veri kümesi kullanılmıştır. Sürücü tespiti problemine çözüm için kurulmuş olan tamamen evrişimli sinir ağı mimarisi ile literatürde yer alan tekrarlayan sinir ağı tabanlı ve geleneksel makine öğrenme yöntemlerindeki başarı oranının üstüne çıkılmıştır. Veri eğitiminde ve doğrulamasında kayan pencere yöntemindeki pencere boyutu azaltılarak daha fazla veri üzerinde gözlemlene yeteneği kazandırılmıştır. Aynı zamanda modelin diskte kapladığı alan ve kayan nokta işlem sayısı düşürülerek araç içi sistemlerde kullanım olanağı arttırılmıştır. Sonuçlar katlamalı doğrulama ile elde edilirken, rota üzerinde tek sürüş içeren veri kümelerinde zaman serisi bölütleme yöntemi ile kıyaslanmıştır.

İkinci olarak, sürüş davranışı tespit problemi için özel olarak tasarlanmış donanım ve bu donanım üzerinde geliştirilmiş yazılım bileşenleri kullanılarak 7 farklı sürücünün kişisel araçlarından 3 aya kadar doğal sürüş verileri toplanmıştır. Literatürdeki çalışmalardan farklı olarak, önerilen otomatik kodlayıcı mimarisi ve kümeleme yöntemi ile sürüş davranışı verileri otomatik olarak etiketlenmektedir. Benzerlik ölçümü için kullanılan Öklid mesafesine ek olarak, Mahalanobis mesafesi ve aykırı veri noktalarının kümeleme sonuçları üzerindeki etkisi t-SNE (t-dağıtılmış stokastik komşu yerleştirme) kullanılarak görselleştirilmiştir. Makine öğrenmesi teknikleri kullanılarak, etiketlenmiş davranış verileri %99 doğrulukla sınıflandırılmıştır. Sonuçlar zaman serisi bölütleme tekniği ile doğrulanmıştır. Ayrıca, önerilen tamamen evrişimli sinir ağı mimarisi ile farklı pencere boyutlarının kümeleme ve sınıflandırmaya olan etkisi araştırılmıştır.

Anahtar Kelimeler: derin öğrenme, sürücü kimlik tespiti, zaman serisi sınıflandırma, zaman serisi kümeleme.

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to my supervisor Assist. Prof. Yakup Genç for his support, guidance, and encouragement through both my undergraduate and graduate education. I am the most grateful to have a remarkably supportive family and close friends. I am thankful to Hakan Alakoca for sharing my excitement and difficulties in my academic career, supporting, and encouraging me in the process of writing this thesis. I thank Tuğba Boydağ wholeheartedly for sharing my burden and offering help wherever she is and whenever she can. I would like to thank Utku Deniz Özen, who helped me develop the device by working countless extra hours with me. I am also grateful to Ortem Elektronik company for supporting this thesis.

TABLE of CONTENTS

	<u>Page</u>
SUMMARY	v
ÖZET	vi
ACKNOWLEDGEMENTS	vii
TABLE OF CONTENTS	viii
LIST OF ABBREVIATIONS and ACRONYMS	x
LIST OF FIGURES	xi
LIST OF TABLES	xiii
1. INTRODUCTION	1
2. BACKGROUND	4
2.1. Driver Identification	4
2.1.1. Traditional Machine Learning-Based Methods	4
2.1.2. Deep Learning-Based Methods	5
2.2. Driver Behaviour Detection	6
2.2.1. Supervised Methods	7
2.2.2. Unsupervised Methods	7
3. DRIVER IDENTIFICATION	10
3.1. Data Collection	11
3.2. Data Pre-processing	11
3.3. Timeseries Classification	13
3.3.1. FCN-Based Approach	14
3.3.2. RESNET-Based Approach	15
3.4. Experimental Results	16
4. DRIVER BEHAVIOR DETECTION	18
4.1. Data Collection	19
4.2. Data Pre-processing	20
4.3. Behavior Labelling	21
4.4. Behavior Classification	22
4.5. Experimental Results	22

5.CONCLUSION AND FUTURE WORK	36
REFERENCES	38
BIOGRAPHY	41
APPENDICES	42



LIST of ABBREVIATIONS and ACRONYMS

<u>Abbreviations</u>	<u>Explanations</u>
<u>and Acronyms</u>	
CAN	: Controller Area Network
CNN	: Convolutional Neural Network
DL	: Deep Learning
DT	: Decision Tree
ECU	: Electronic Control Unit
FCN	: Fully Convolutional Network (SORT)
GAP	: Global Average Pooling
GPS	: Global Positioning System
GPU	: Graphics Processing Unit
IMU	: Inertial Measurement Unit
KNN	: K-Nearest Neighbors
LGBM	: Light Gradient Boosting Machine
LSTM	: Long Short-Term Memory
ML	: Machine Learning
MLP	: Multi Layer Perceptron
OBD2	: On-Board Diagnostics
OEM	: Original Equipment Manufacturer
RELU	: Rectifier Linear Unit
RF	: Random Forest
RESNET	: Residual Neural Network
RNN	: Recurrent Neural Network
RPM	: Revolutions Per Minute
SDD	: Safe Driving Dataset
SMO	: Sequential Minimal Optimization
SVM	: Support Vector Machine
TS	: Time step
WS	: Window Size

LIST of FIGURES

<u>Figure No:</u>	<u>Page</u>
3.1: Driver Identification Framework.	10
3.2: Controlled Environment Data Collection Route.	11
3.3: Correlated Features.	12
3.4: Sliding Window Application.	13
3.5 FCN Architecture.	15
3.6: RESNET Architecture.	15
4.1: Driver Behavior Detection Framework.	18
4.2: Autoencoder Architecture.	21
4.3: Finding optimum cluster number.	23
4.4: Combining closest cluster.	23
4.5: Clusters with filtering algorithm using encoder output.	24
4.6: t-SNE results with filtering algorithm using encoder output.	25
4.7: The worst f-score values obtained in the 8 experiments.	26
4.8: Aggressive driving route (left) and calm driving route (right).	27
4.9: Up to three months of driving data results.	29
4.10: Fifteen minutes of driving data results.	31
4.11: Silhouette scores of experiments E1, E3, E2 and E4, respectively. Using encoded data with filtering algorithm has best score.	34
4.12: Davies-Bouldin scores of experiments E1, E3, E2 and E4, respectively. Using encoded data with filtering algorithm has best score.	34
4.13: Silhouette scores of experiments E5, E7, E6 and E8, respectively. Using encoded data with filtering algorithm has best score. We also highlight that, in clustering using Mahalanobis distance, data encoding produced more effective results than the filtering algorithm.	35
4.14: Davies-Bouldin scores of experiments E5, E7, E6 and E8, respectively. Using encoded data with filtering algorithm has best score.	35

A1.1: Cluster distributions in Clustering Experiment 1.	43
A1.2: Cluster distributions in Clustering Experiment 2.	44
A1.3: Cluster distributions in Clustering Experiment 3.	45
A1.4: Cluster distributions in Clustering Experiment 4.	46
A1.5: Cluster distributions in Clustering Experiment 5.	47
A1.6: Cluster distributions in Clustering Experiment 6.	48
A1.7: Cluster distributions in Clustering Experiment 7.	49
A1.8: Cluster distributions in Clustering Experiment 8.	50



LIST of TABLES

<u>Table No:</u>	<u>Page</u>
3.1: Safe Driving Dataset.	16
3.2: GEBZE Dataset.	16
3.3: UYANIK Dataset.	17
3.4: WS=30 FCN Performance Evaluation.	17
4.1: Clustering Experiment 1.	28
4.2: Clustering Experiment 2.	28
4.3: Clustering Experiment 3.	30
4.4: Clustering Experiment 4.	31
4.5: Clustering Experiment 5.	32
4.6: Clustering Experiment 6.	32
4.7: Clustering Experiment 7.	33
4.8: Clustering Experiment 8.	33
A1.1: Window size parameter comparison using FCN.	42

1. INTRODUCTION

The number of ECUs (Electronic Control Units) in the vehicles produced today is up to 80. These units, which are the control units of the vehicles, communicate with each other through the network technology invented by Bosch called CAN (Control Area Network). The OBD2 (On-Board Diagnostics) port is a communication interface that is used by vehicle manufacturers for after-sales fault diagnosis and maintenance, and where various ECUs in the vehicle broadcast real-time messages according to the determined standards. Sensor data broadcast through this port varies according to vehicle manufacturers, and sensor data such as gas pedal position, engine speed, and vehicle speed can usually be accessed.

Finding which driver is driving the vehicle and profiling driver behavior using vehicle diagnostic data is important for end-users, vehicle manufacturers, vehicle fleets and insurance companies. Solutions for these tasks include significant fuel savings, software customization, and hardware changes with its personalized driving mode. At the same time, it is of great importance in personal and industrial use, such as early detection of accidents, early warning systems in case of theft, and reducing maintenance costs. In order to meet this demand, the number of academic and industrial studies is increasing day by day. In the future, predicting the increase in the number of vehicles that can communicate effectively with each other, the widespread use of the internet of vehicles and artificial intelligence, and the evaluation of this sensor data will become widespread.

Data sources for driver behavior analysis studies include driving simulators, LiDAR and Radar, multimedia data, smartphone sensors, Global Positioning System (GPS) data, and Controller Area Network (CAN) bus data [1]. The most cost-effective solutions are studies that use only vehicle sensor data without the use of expensive external equipment. Although the solutions that use sensors on drivers' smartphones are also a cost-effective choice, keeping the phone fixed in the car limits the driver's ability to use it. At the same time, Inertial Measurement Unit (IMU) calibration is required for these solutions. Although there are rule-based and model-based studies in previous studies, most of the recent studies consist of traditional machine learning-based methods. These methods require feature engineering, one of the most difficult tasks regarding data analytics-based solutions.

Deep learning-based approaches eliminate this need and produce more robust solutions with the help of increasing data and the computational power of Graphics Processing Unit (GPU). In the literature, several approaches exist for driver identification and driver behavior detection problems. Because the RNN-based approaches are suitable for timeseries data, most of the studies used RNN-based algorithms. However, RNN-based algorithms based on training require high computational cost and time. The experiments were often performed with data collected over a short period of time in a controlled environment. Furthermore, numerous driving behavior detection studies have been conducted on data that has been manually labeled with driving behaviors.

Within the scope of this thesis, two different problems, namely driver identification and driver behavior detection, are discussed. For the first problem, driver identification is performed using the data set collected in a controlled environment with a commercial device. The results are observed using different validation techniques. For the second problem, a naturalistic dataset is collected with the end-to-end system architecture and an unsupervised clustering method is used with unlabeled data to determine the behaviors of the drivers. Obtained behaviors are visualized and classified using machine learning methods. In this thesis, our main contributions are summarized as follows:

- For driver behaviour analysis problems, researchers have difficulty finding a dataset. In this study, vehicle sensor data is collected from five different drivers on two different weekdays in a controlled environment.
- Different from traditional machine learning methods using feature extraction techniques and RNN-based studies in the literature, solutions are proposed using fully convolutional neural networks and RESNET (Residual Neural Network) architectures.
- We achieve close performance by proposing a Fully Convolutional Network (FCN) architecture to the state-of-the-art while reducing window size parameters used in the sliding window technique and model complexity.
- By establishing an automated data collection pipeline with custom-designed low-cost hardware, we collect naturalistic (uncontrolled environment) driving data involving seven individual drivers for up to three months.

- We propose a robust unsupervised learning technique to label timeseries vehicle sensor data automatically.
- Results are validated with the timeseries segmentation method using additional distance metrics.

This thesis is organized as follows. Section 2 provides the background material for the thesis. In Section 3, we discuss the first problem in detail and our solutions, along with experimental findings and discussions. Similarly, the second problem and our solutions to it, along with our experiments and discussions, are presented in Section 4. Finally, we conclude the thesis in Section 5, covering our future studies.



2. BACKGROUND

In this thesis, we cover the analysis of driver behavior and driver identification using only vehicular sensor data as background material. First, we present prior work in the field of driver identification, a subproblem of driver behavior analysis. Afterwards, driving behavior detection methods are reviewed. There have been a large number of studies analyzing driver behavior using rule-based or model-based methods. In our work, we focus on machine learning-based methods instead of rule-based or model-based methods.

2.1. Driver Identification

OBD2 protocol has been used in vehicles to diagnose the vehicle's statuses since 1998 [2]. This data is accessible using low-cost hardware. Because of easy accessibility to this sensor data, many researchers are interested in using only CAN-Bus sensor values as the source data for driver behavior analysis problems. Machine learning-based approaches are in high demand in this research area. There are numerous methods based on machine learning. Rule and model-based algorithms require further data analysis to get robust and generalized results. When processing the amount of data increases, these algorithms even more difficult to get robust results. Data-driven algorithms are more suitable when amount of data increases. Also, they have learning and adaption capabilities using only data, not fixed thresholds. Machine learning methods can also be grouped under two main methods. These methods are applying traditional machine learning techniques using feature engineering and applying deep learning techniques without feature engineering. Firstly, we present prior work belonging to traditional machine learning methods for driver identification. Afterwards, deep learning-based methods are reviewed.

2.1.1. Traditional Machine Learning-Based Methods

The automatic driver identification problem has drawn the attention of researchers due to the growing privacy and safety concerns of smart automobiles. Some studies focused on biometrics solutions using face recognition [3]. Recent study [4] offers a

lower-cost hardware architecture using facial recognition algorithms. Some researchers used additional sensors such as microphones and finger printing devices [5]. However, biometric approaches for driver identification task suffer from high costs and physical limitations.

Reducing costs and having various sensors of smartphones, many researchers interested in smartphone-based approaches for driver identification task. Smartphone-based solutions strictly require the fixed position of the smartphone while driving. However, it is a well-known drawback that drivers can't move their smartphones during the data collection process. A recent study [6] used only GPS data for the driver identification task. The 5-driver set results are 90.06%, 89.01%, and 89.16% for Random Forest (RF), K-Nearest Neighbors (KNN), and Support Vector Machine (SVM), respectively, using the dataset of study in [7].

Kwak et al. feature engineering technique is used for driver identification problem [7]. They have used traditional machine learning methods, using feature engineering. There are 51 features in the collected dataset. They selected 15 features from the dataset and then gathered statistical and temporal timeseries features. They have divided timeseries data into sequences using 60 second windows and 1 second time steps. The Decision Tree (DT) and KNN algorithms were used to achieve 99% accuracy.

Driver gender and identity were determined by using the UYANIK dataset collected by Sabanci University as part of the Safe Driving Project [8]. For the features selected in the study, 8 and 216 statistical features (such as vehicle speed and engine speed) were extracted, then SMO (Sequential Minimal Optimization), RF, and SVM algorithms were used. While 97% success was achieved in gender prediction, 10% accuracy was achieved in driver identification [9].

2.1.2. Deep Learning-Based Methods

The feature engineering process is mostly time-consuming and challenging task of data preparation. It also required good domain knowledge to get meaningful features from the data. Traditional machine learning methods depend on feature engineering. At this point, deep learning methods do not require complex pre-processing and feature engineering. It can learn meaningful features from the data. Nowadays, with the growth of data exponentially, deep learning-based methods are becoming more

popular than traditional machine learning methods. The majority of studies in the field of driver behavior analysis used RNN in conjunction with Convolutional Neural Network (CNN)-based network architectures. These networks already offered the state-of-the-art results in activity recognition and speech recognition tasks where extracting temporal information was required [10]. This type of architecture is able to capture time dependencies with convolutional operations. There are two types of networks in the literature: DeepConv RNN-Attention [11] and LSTM-FCN [13], where high accuracy is achieved using the Safe Driving Dataset. In a recent study [12], the window size parameter was reduced from 60 seconds to 45 seconds with an architecture inspired by these two networks.

2.2. Driving Behavior Detection

Driver behavior detection is crucial, especially in the context of driver assistance systems and driving safety. Because of the limitations of current driving assistance technologies, researchers and developers are working on improving human driving experience. In the context of a driver behavior detection problem, there are multiple factors that affect driving behaviors [14]. It is possible to collect these factors in three main categories as follows:

- **Human-related factors:** Drivers may differ in their driving due to changes in both physical and psychological conditions due to human nature. Driver-specific and frequently changing situations such as fatigue, distraction, and stress level while driving can be detected and their effects on driving behaviors can be measured with wearable devices.
- **Vehicle-related factors:** Such as transmission type, horsepower, engine type, and sport mode. These vehicle specific factors could be affected by driving behavior such as aggressive acceleration and deceleration, which can be detected by collecting in-vehicle sensor data.
- **Environmental factors:** There are lots of factors that can be considered environmental factors. Mainly, it can be explained as being due to weather conditions, road conditions, and traffic conditions. Driving behaviors can be detected with data from various data sources.

There have been numerous approaches to detecting driving behavior with various data sources. Studies in the literature are reviewed in terms of supervised and unsupervised machine learning techniques.

2.2.1. Supervised Methods

The types of driving events detected during driving may vary depending on the types of sensors used in the dataset. Bejani et al. use smartphone accelerometer and magnetometer data to identify dangerous and normal maneuvers using ensemble methods [15]. They also used the Multi-Layer Perceptron (MLP) method to classify traffic levels into two categories: congested and normal. They considered three types of maneuvers: U-Turn, Turn, and Lane-Change. However, it is required for the smartphone IMU calibration process. In [16], authors evaluate the aggressiveness of driving patterns using drivers' cars following behaviors with Bayesian nonparametric approaches. They have used an external mobile-eye camera to calculate distances.

Recognizing driving style can assist drivers to manage their fuel consumption. Magaña et al. developed a driving assistant that advises drivers to decrease fuel consumption using several in-vehicle sensor values, road, and weather conditions according to driving style [17]. In [18], the authors applied RF to predict the fuel consumption of fleet vehicles using limited sensor data such as vehicle speed, acceleration, and GPS information.

Also, studies [19, 20] monitored driver attention using in-vehicle cameras. In [19], the drivers were required to do routine secondary activities such as radio, phone, and navigation system operation. Authors used their head movements to detect distractions. Nanxiang et al. applied machine learning methods to find drivers are distracted or not using eye and head movements [20]. It is also proposed to use semi-supervised methods for labeling training data.

2.2.2. Unsupervised Methods

The aforementioned driving behavior analysis studies suffer from getting labeled dataset or labeling training dataset. Correctly labeled behavior data and increased performance of driving behavior detection models are directly related to each other. Labeling behavior data can be done manually or automatically. Manually labeling

driving behavior data requires domain expertise. It is also prone to human error. In order to overcome this problem, labeling driving behaviors should be done automatically.

Clustering algorithms are one of the most useful techniques in unsupervised learning methods. In [21], the authors applied the K-Means clustering algorithm for driver aggressiveness identification. In-vehicle sensor data was collected on two different road types. They used a clustering technique to determine whether the roads were urban or highway. After labeling the road types, a driver aggressiveness rule was defined for each road type using the arithmetic mean of the acceleration sensor. If a driver exceeds defined limits, it is considered aggressive behavior. Peppes et al. applied a clustering technique in order to classify a driver's behavior as eco-friendly or not [22]. Vehicle speed and Revolutions Per Minute (RPM) sensor values were used in the clustering algorithm. After the labeling process, traditional machine learning and deep learning algorithms were used for classification. High classification accuracy was obtained according to the results. However, the authors did not mention the data normalization technique before applying the clustering algorithm. We highlight that it might have been caused to overfitting problem.

In one of the studies in the literature, the authors proposed an unsupervised method for the detection of driver behavior patterns [23]. They mainly focused on taxi driver behavior analysis in Sydney streets. In this work, only raw GPS data, which consists of speed, latitude, longitude, altitude, and bearing information, was used for analysis. Approximately 4000 vehicles participated in this study. They have applied the K-Means clustering algorithm for detecting driver behavior. Results show that five different clusters were obtained in the study. The proposed method is generic, and it can be used with custom GPS hardware or a smartphone. However, because of high level noises and a lack of driver and vehicle details, it is difficult to find a generalized solution using only raw GPS data.

Choi et al. considered a driver behavior profiling problem as anomaly detection [24]. The authors trained RNN as a regression model with normal data only. According to the results, when a sequence of aggressive driving behaviors is used, the proposed method produces a high regression error. In contrast, when a sequence of normal driver behavior is used, it produces a low regression error. However, the dataset used in their experiments consists of one normal behavior and six aggressive behaviors that were

labeled manually. It was collected using only smartphone IMU sensors with two drivers and lasted approximately 13 minutes.

Study in [25] examine K-Means clustering technique for driver behavior analysis purpose. Vehicle sensor data was collected in an uncontrolled environment involving 54 drivers. Selecting optimal cluster number techniques was investigated in this study.



3. DRIVER IDENTIFICATION

In this thesis, the driver identification problem is considered as a multivariate timeseries classification task. Driver identification problem is relied on different independent parameters such as weather conditions, traffic conditions, and vehicle physical constraints. Our focus is predicting the driver from only collected real time vehicle multivariate sensor data. It is required to obtain different datasets in order to apply timeseries classification. In addition, to perform highly robust driver identification, we need to fix predetermined constraints such as driving route and vehicle to constitute a controlled environment. In the literature, it is a well-known issue that the Original Equipment Manufacturer (OEM)'s vehicle data is isolated for security and privacy reasons. Therefore, in this thesis, a novel driving dataset using additional hardware and software components was collected.

In this scope, the target is to develop an end-to-end deep neural network framework for driver identification task. We propose a driver identification method that has better identification accuracy with a lower memory requirement and less computation time according to recent the state-of-the-art studies [11, 12, 13]. To validate the proposed methods, k-stratified fold validation and timeseries segmentation were applied. The proposed driver identification framework consists of three main modules, which are Data Collection, Data Pre-processing, and Timeseries Classification, as illustrated in Figure 3.1.

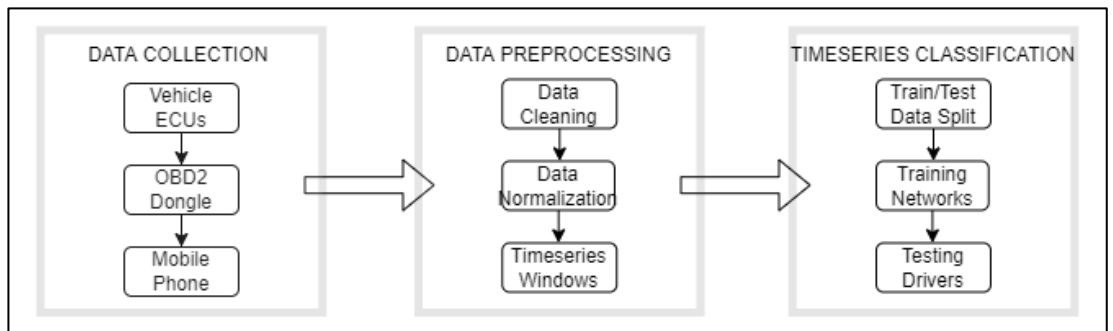


Figure 3.1: Driver Identification Framework.

3.1. Data Collection

During the dataset collection, 34 sensor data such as steering wheel angel, engine rpm, vehicle speed, etc. were gathered from the vehicle using an OBD2 socket every 1 second interval. An ELM327 Bluetooth device is used for data collection setup. In total, five drivers drove the same vehicle twice on different days of the week to generalize the problem. During the data collection process, drivers drove the vehicles on mainly 3 different road types. These road types consist of highways with higher speed limits and less traffic, city ways with dense traffic conditions that include crosswalks and traffic lamps, and rural ways that contain various ramps and bumps with low-speed limits. The route is illustrated in Figure 3.2. In the dataset collection process, the Renault Diesel 1.5 automatic transmission model was used as an experimental vehicle.

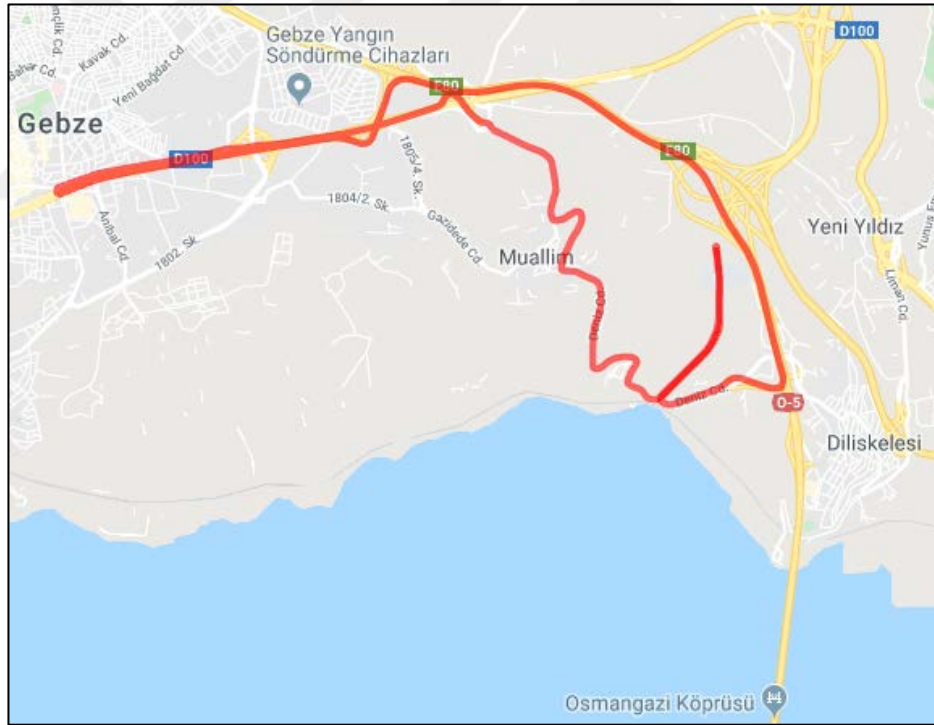


Figure 3.2: Controlled Environment Data Collection Route.

3.2. Data Pre-processing

Most of the studies reviewed in the literature have focused on data analysis, which includes feature selection and statistical feature generation techniques. The

computational and time costs of these processes can be quite high. Also, some selected features may not be available in all vehicles. This limits the applicability of the studies that have been done. In this study, only basic pre-processing has been done, which is normalization of the sensor data. All sensor data in the dataset is normalized as indicated by Equation 3.1.

$$X_i = \frac{X_i - \min(X_i)}{\max(X_i) - \min(X_i)} \quad (3.1)$$

Tree-based machine learning algorithms are robust to highly correlated features. In the decision process, the tree branches according to perfectly correlated features. During training, artificial neural networks can also handle correlated features. They can approximate any function theoretically. However, using highly correlated features increases model complexity, and these features don't improve models. For this purpose, feature correlations were calculated using the Pearson method. Figure 3.3 shows how vehicle speed and speed on wheels features were discovered to be perfectly correlated. These features were cleaned up to reduce the complexity of the models.

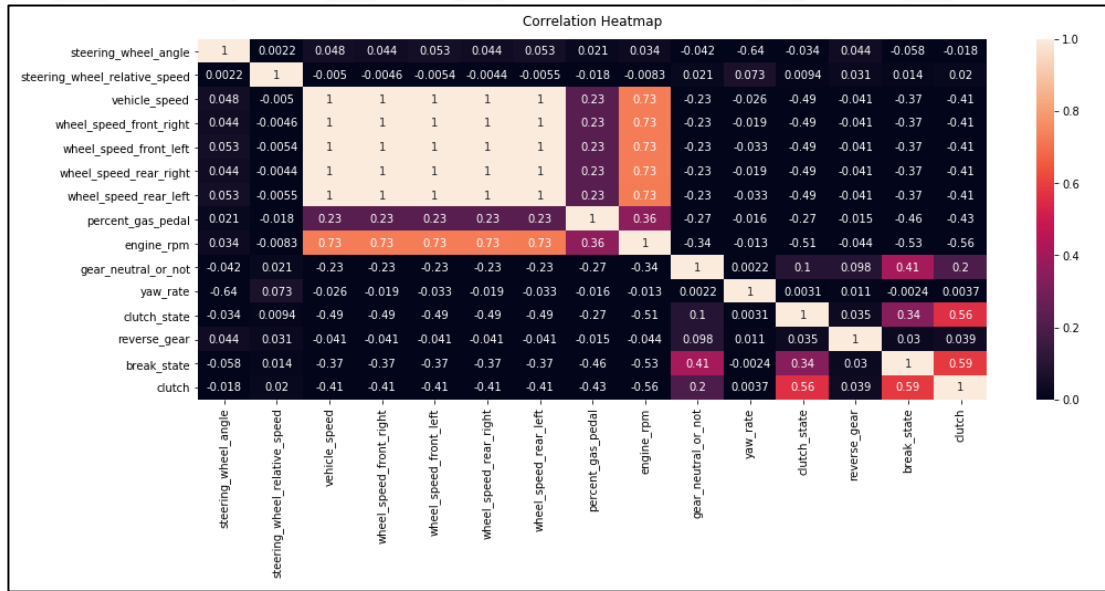


Figure 3.3: Correlated Features.

After data normalization and cleaning steps, timeseries data is segmented into windows using the sliding window technique. Vehicle sensor data can be thought of

as a time-ordered sequence of values. The sliding window technique is a well-known approach to ensure obtaining temporal information from timeseries data. As seen in Figure 3.4, timeseries sensor data is separated into continuous data segments using a fixed window size and time step. In the literature, overlapping windows capture temporal information well. However, we highlight that it also gives rise to overfitting problem in the training process.

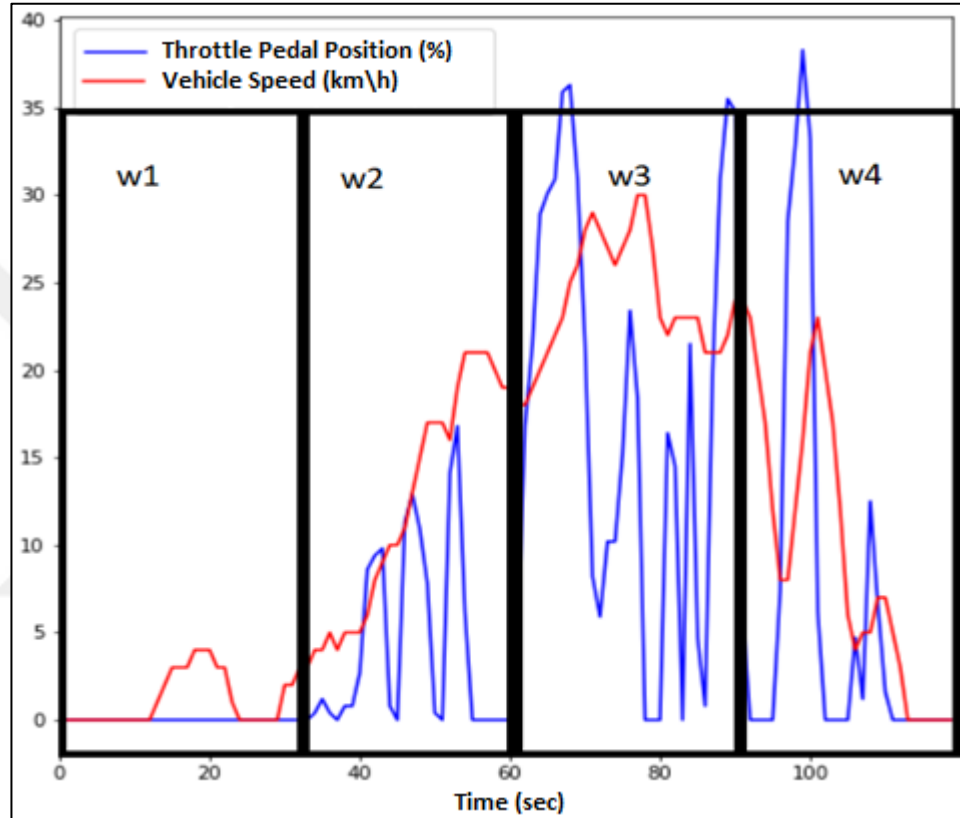


Figure 3.4: Sliding Window Application.

3.3. Timeseries Classification

In this thesis, we considered two different the state-of-the-art studies as a baseline for driver identification problem. The first of these, the DeepConv RNN-Attention model [11], proposes an RNN-based approach for driver identification problem. However, this approach has high computational costs for both training and evaluation process. For this reason, lightweight LSTM-FCN method [12], which is an approach that can perform driver identification with high accuracy with a less complex

model based on the Attention-Conv-LSTM model, became the secondary baseline model in our work.

For the driver identification problem, our goal is to design a solution that is as accurate as the state-of-the-art models in the literature while also proposing a less complex approach based on existing studies. In this context, firstly, the LSTM layer in the proposed architecture was removed. After the LSTM layer was deleted, the remaining model became the traditional CNN. The goal then shifted to using CNN to transfer temporal and spatial information from the input layer to the final layer. To ensure that the dimensions of the time series as input remain fixed throughout the network, the local pooling layers after each convolution layer have been removed. Then, by adding the global average pooling layer instead of the fully connected layer at the end of the CNN, both model complexity was reduced, and overfitting was prevented. The network obtained for this purpose became FCN. Another method we proposed is RESNET to observe the transferability of temporal and spatial information in a deeper CNN.

3.3.1. FCN-Based Approach

The FCN architecture is formed by 3 fundamental blocks, and each one has three convolutional layers. The convolution operator is used with L2 kernel regularizer penalty factor in each layer, followed by batch normalization. The result obtained is passed through the RELU (Rectifier Linear Unit) activation function. Then 0.2 dropout regularization was applied. After 3 fundamental convolutional blocks, GAP (Global Average Pooling) layer is used to prevent over-fitting problem. As a final step, after the GAP layer, SoftMax classifier is applied to identify which driver class it belongs to. The number of filters used in convolution operations is 64, 128, 128 and the filter lengths are 8, 5, and 3 respectively. The proposed FCN architecture diagram is illustrated in Figure 3.5.

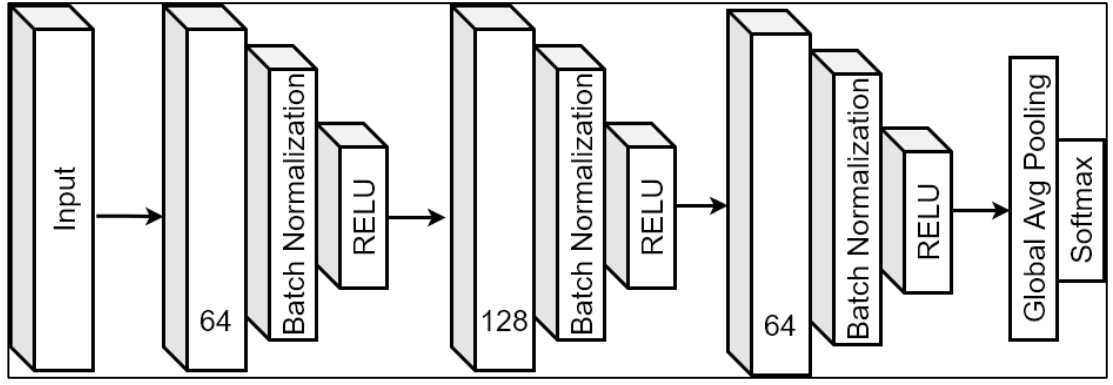


Figure 3.5: FCN Architecture.

3.3.2. RESNET-Based Approach

Like FCN architecture, the RESNET architecture consists of 11 layers. The first 9 layers are convolutional layers. The characteristic feature of RESNET architectures is that consecutive convolutional layers are connected to each other with residual links. Thanks to these connections, a solution to the lost slope problem in deep networks has been created. The RESNET architecture consists of 3 consecutive residual blocks. As in the FCN architecture, each residual block consists of 3 convolutional layers. The output of each block is fed as the input of the other residual block. The GAP layer is used at the end of these blocks. Then SoftMax classifier is applied to identify which driver class it belongs to. The number of filters used in convolution operations is 64, 128, 128 and the filter lengths are 8, 5, and 3 respectively. The proposed RESNET architecture diagram is illustrated in Figure 3.6.

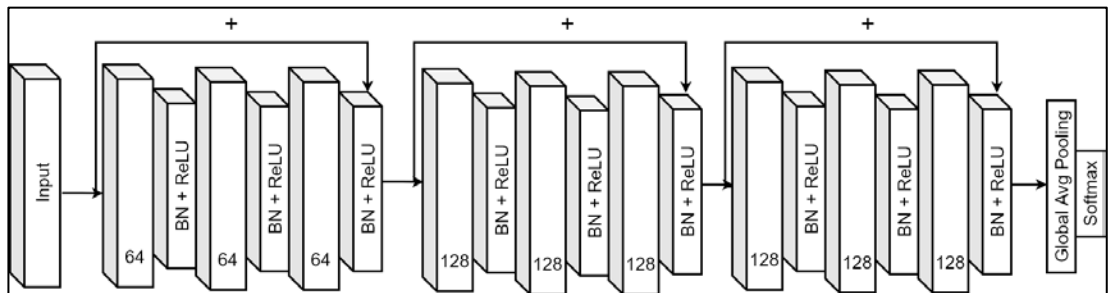


Figure 3.6: RESNET Architecture.

3.4. Experimental Results

In this scope, two different models were proposed to classify drivers according to the vehicle sensor data. The performance of the proposed networks was observed in 3 different datasets. While performing stratified k-fold validation for the Safe Driving Dataset (SDD) and GEBZE datasets that include driving on multiple routes, the timeseries segmentation method was applied to the UYANIK dataset consisting of a single route. UYANIK dataset. For timeseries segmentation, %80 of data divided for training set, rest of data divided to test set. In order to observe the different driver numbers in the UYANIK dataset, 2 sub-datasets were created with 35 and 10 drivers.

Table 3.1: Safe Driving Dataset.

METHOD	TS = 60		TS = 45		TS = 10	
	ACC	F-SCORE	ACC	F-SCORE	ACC	F-SCORE
RESNET	0.8876	0.8796	0.9377	0.9375	0.9979	0.9978
FCN	0.9165	0.9155	0.9367	0.9362	0.9961	0.9961

While creating training and test datasets, the effect of the sliding window method on performance with different time steps and window sizes was observed. Initially, the window size was chosen to be 60 seconds with time steps of 60, 45, and 10 seconds. Performance measurements were observed in two different networks with these parameters. In the next experiment, performance metrics were observed in all datasets of the FCN architecture by selecting WS (Window Size) for 30 seconds and TS (Time Step) for 30, 10 and 6 seconds. Accuracy and f-score are used as evaluation metrics for proposed methods. First experiment results for each dataset are given on Table 3.1, Table 3.2, and Table 3.3. On Table 3.4, the second experiment results are given.

Table 3.2: GEBZE Dataset.

METHOD	TS = 60		TS = 45		TS = 10	
	ACC	F-SCORE	ACC	F-SCORE	ACC	F-SCORE
RESNET	0.8715	0.8604	0.9722	0.9729	0.9994	0.9994
FCN	0.9256	0.9155	0.9651	0.9630	0.9981	0.9980

Table 3.3: UYANIK Dataset.

D	METHOD	TS = 60		TS = 45		TS = 10	
		ACC	F-SCORE	ACC	F-SCORE	ACC	F-SCORE
88	FCN	0.611	0.612	0.737	0.733	0.995	0.995
	RESNET	0.561	0.557	0.696	0.693	0.9	0.909
	FCN (*)	0.501	0.487	0.526	0.513	0.537	0.522
	RESNET (*)	0.541	0.544	0.549	0.526	0.602	0.592
35	FCN	0.759	0.758	0.924	0.923	0.999	0.999
	RESNET	0.743	0.742	0.826	0.823	0.983	0.983
	FCN (*)	0.705	0.706	0.699	0.695	0.673	0.672
	RESNET (*)	0.661	0.661	0.636	0.62	0.662	0.665
10	FCN	0.869	0.866	0.925	0.923	0.999	0.999
	RESNET	0.85	0.843	0.929	0.929	0.997	0.996
	FCN (*)	0.833	0.836	0.865	0.868	0.871	0.872
	RESNET (*)	0.782	0.782	0.855	0.858	0.854	0.853

According to the experimental results [11], [12] (95.1, 98.72%) higher success (99.01%) was obtained among the algorithms that produced the best results on the SDD used in the literature with FCN architecture. The window size (60, 40) used in the sliding window method has been reduced to 30 seconds, the number of floating-point operations has been reduced from 0.232 million to 0.154 million, and the memory used by the proposed network has been reduced from 1.69 MB to 0.93 MB.

Table 3.4: WS=30 FCN Performance Evaluation.

DATASET	TS = 30		TS = 10		TS = 6	
	ACC	F-SCORE	ACC	F-SCORE	ACC	F-SCORE
SDD	0.912	0.911	0.976	0.975	0.99	0.989
GEBZE	0.941	0.937	0.994	0.994	0.998	0.997
U - 10	0.821	0.821	0.995	0.995	0.999	0.999
U - 35	0.699	0.695	0.987	0.987	0.997	0.997
U - 88	0.558	0.549	0.944	0.943	0.991	0.991
U - 10 (*)	0.875	0.877	0.85	0.85	0.838	0.832
U - 35 (*)	0.72	0.717	0.658	0.655	0.696	0.691
U - 88 (*)	0.528	0.521	0.565	0.561	0.581	0.569

4. DRIVER BEHAVIOR DETECTION

In this thesis, we propose an unsupervised clustering technique for labeling driving behavior data. The target is to develop an end-to-end deep neural network framework for driver behavior detection task. As in Figure 4.1 the proposed driver behavior framework consists of four main modules, which are Data Collection, Data Pre-processing, Behavior Labeling and Behavior Classification. The data collection process should be automated. Most studies gathered each trip's data manually. In the data collection module, the automated data collection process is explained. In the data pre-processing module, the main goal is to prepare vehicle sensor data for clustering and classification tasks. It includes data cleaning, normalization, feature vector generation, and timeseries window splitting techniques. In the behavior labeling module, the main purpose is to find unique behaviors for each driver using timeseries clustering techniques. Clustering results were observed using different distance metrics and visualization techniques. In behavior classification module, various machine learning methods are applied to identify the behaviors of each driver.

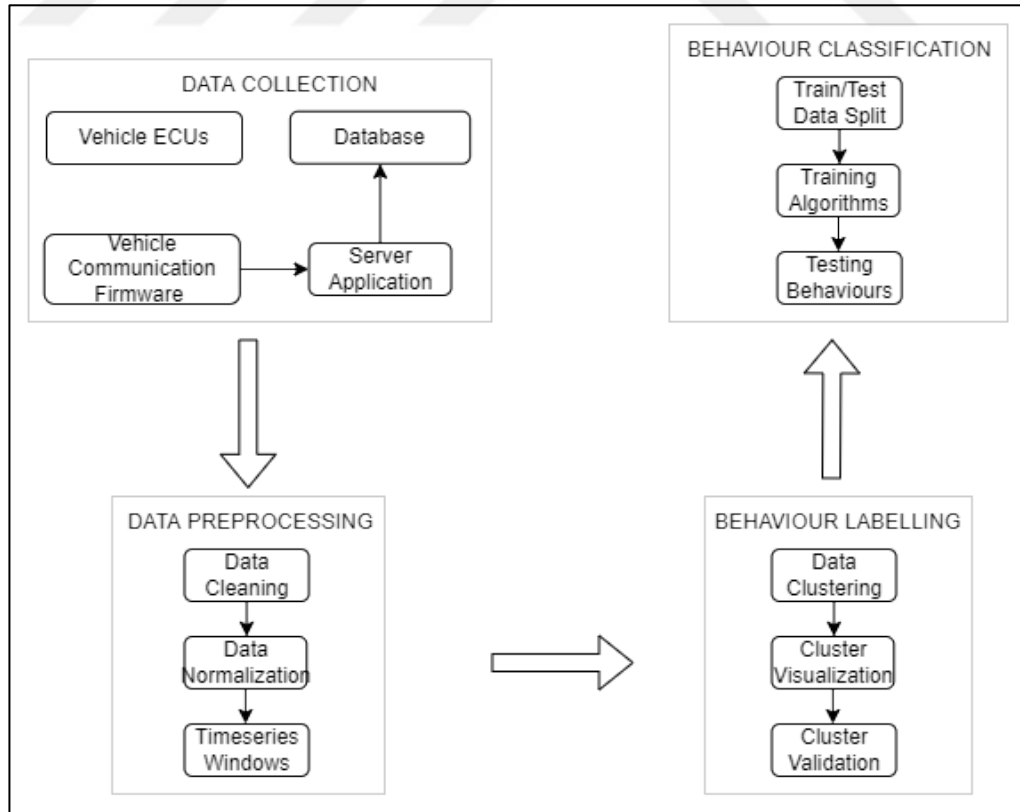


Figure 4.1: Driver Behavior Detection Framework.

4.1. Data Collection

Collecting driving behavior datasets is a challenging task for researchers. An external hardware is needed to access sensor data from vehicles. As explained in the previous section, it is not possible to access this data directly. In order to collect naturalistic long-term data, the data collection setup used in the previous section is not sufficient. For this reason, the vehicle-device communication and device-server communication modules on the device, whose hardware design was made by Ortem Elektronik, were developed within the scope of this thesis. Instead of using external ELM327 chips for OBD2 protocol implementation, a custom CAN-bus communication layer was implemented to reduce the cost of designed hardware. For the sake of simplicity and easy integration, the message queuing telemetry transport protocol was used on the server-client communication layer. For data format, Google's protocol buffers were used. These technologies provide us with programming language independence while processing and storing sensor data accessed from the vehicle.

The dataset contains seven drivers in seven different cars. Diesel and gasoline vehicles are involved. The OBD2 protocol supports 192 different sensor values. Which sensor data is published is at the discretion of the vehicle manufacturers. A total of eight common sensor values were selected during data collection. Below are descriptions of the attributes:

- `acc_x_axis_instant_g`: Calculated instant x axis g value on hardware when data read from vehicle.
- `acc_y_axis_instant_g`: Calculated instant y axis g value on hardware when data read from vehicle.
- `acc_z_axis_instant_g`: Calculated instant z axis g value on hardware when data read from vehicle.
- `pid_04_calculated_engine_load`: Indicates a percentage of peak available torque. Reaches 100% at wide open throttle at any altitude or RPM for both naturally aspirated and boosted engines. The value interval is 0–100 [2].

- `pid_11_intake_manifold_abs_pressure`: Pressure in the intake manifold derived from a manifold absolute pressure sensor. It is used for fuel consumption calculation. The value interval is 0–255.
- `pid_12_engine_speed_rpm`: The current engine speed in revolutions per minute. The value interval is 0–16383,75.
- `pid_13_vehicle_speed_kmh`: The current vehicle speed in kilometers per hour. The value interval is 0–255.
- `pid_73_accelerator_pedal_position_d`: The absolute pedal position (not the relative or learned) pedal position. Usually above 0% at idle and less than 100% at full throttle. The value interval is 0–255.

4.2. Data Pre-processing

The collected dataset contains different sensor variables which are measured in different units. In this study, clustering methods with different distance metrics are applied to determine whether a given set of sensor values belongs to a certain cluster. If a sensor value is significantly higher, it can dominate distance measures, suppressing other variables with small values. To prevent such a situation, we applied a data normalization technique. Following the normalization and cleaning of the data, the timeseries data is segmented into windows using the sliding window technique, as described in the previous section.

In order to perform clustering, feature vectors were obtained from the timeseries divided into windows with two different approaches. As a first approach, statistical information from the sensor data of fixed-size windows was extracted. A 1x40 feature vector was obtained, mainly consisting of standard deviation, mean, median, skewness, kurtosis, and Fast Fourier transformations of these values.

As a second approach, autoencoders were applied to represent temporal timeseries features automatically from segmented time windows rather than feature engineering manually. A deep autoencoder network was created for this purpose. Autoencoders are mainly neural networks whose main goal is to compress their inputs to a latent space and then decompress them from the latent space to their output. Numerous types of autoencoder networks exist in the literature. Recent studies have mostly focused on data denoising and dimensionality reduction. The autoencoder

network consists of three main blocks, which are the encoder, latent vector, and decoder. The proposed autoencoder architecture diagram is illustrated in Figure 4.2.

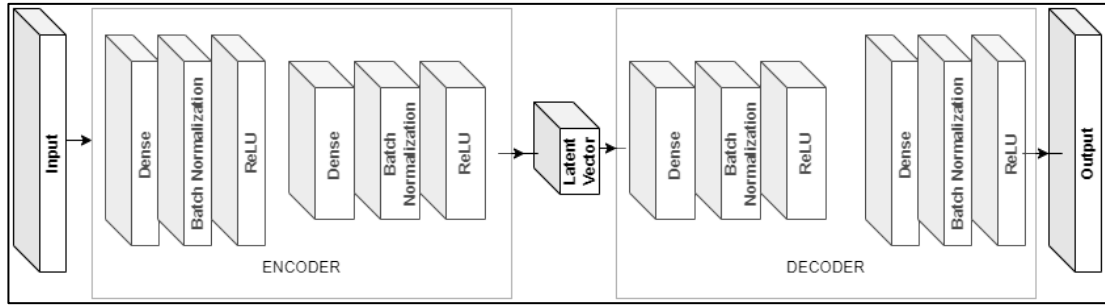


Figure 4.2: Autoencoder Architecture.

4.3. Behavior Labeling

Clustering is the task of identifying subgroups that have very similar data samples in the data. Several distance metrics can be used for similarity measurement, such as the Euclidean distance or Mahalanobis distance. Choosing distance metrics is application specific. Clustering is not one specific algorithm. There are various algorithms for this task. In our work, we have used the K-means clustering algorithm. It assigns data samples to a subgroup in which the sum of the squared distance between the data samples and the selected subgroup's centroid is at the minimum. The centroid is the arithmetic mean of all the sample points that belong to that subgroup. We have used the Euclidean distance metric for clustering behavior labels.

Like most clustering algorithms, the K-Means algorithm takes the number of clusters as an input parameter. There are various methods for identifying optimum cluster numbers. Silhouette score [26] is one of the most used methods for this purpose. The silhouette score ranges from -1 to 1. By representing clusters as being close to their centroids and separated from other clusters, the closer it goes to 1, the higher clustering performance it has.

Additionally, the Mahalanobis distance metric was used for the K-Means clustering algorithm. The Mahalanobis distance is the distance between two variables in multivariate data. It measures distance relative to the centroid, which can be considered as an overall mean for multivariate data. In other words, the centroid is a place where all of the variables' means intersect [27].

Visualizing clusters is an important task to observe clustering results. Since datasets used in this study were multivariate timeseries data, high-dimensional data visualization technique was required. t-SNE [28] is used for visualizing high-dimensional data. Basically, it computes a similarity measure between pairs of instances in high and low dimensional space. A cost function is used for optimizing these two similarity measures.

4.4. Behavior Classification

After the labeling process, the following machine learning methods were used to classify behaviors: RF, Gradient Boosting Machine, KNN, and FCN. The Random Forest algorithm is a supervised machine learning algorithm that is used for behavior classification tasks. It combines multiple decision tree classifiers and takes their most voted ones to solve the classification problem. Therefore, a larger number of trees in the algorithm prevents overfitting problem. It also belongs to the ensemble technique, which can be defined as combining multiple models for making predictions instead of individual models.

Another ensemble technique-based algorithm is the Gradient Boosting Classifier. Basically, the idea is to combine many weak learning models to create a stronger classifier model. We also applied LGBM (Light Gradient Boosting Machine) [29], which is an optimized variant of the Gradient Boosting method.

The KNN algorithm can be defined as a similarity-based classifier, and it belongs to supervised machine learning algorithms. It considers the similarity between observation data points and sample data points using a similarity distance metric. The Euclidean distance metric is used for behavior classification task.

4.5. Experimental Results

The performance of the proposed methods was observed in the ORTEM Naturalistic Driving Dataset. Since the collected dataset was unlabeled, the first step had to be labeling. In order to make the clustering technique that will be used for labeling more robust, the data normalization specified in Equation 3.1 has been applied as a preprocessing. While finding the optimum number of clusters required for clustering, the elbow technique was used. To avoid local optima, the clustering

algorithm was performed 10 times with the initial centroids assigned at random. According to the results obtained in Figure 4.3, the parameter $k = 3$ was chosen

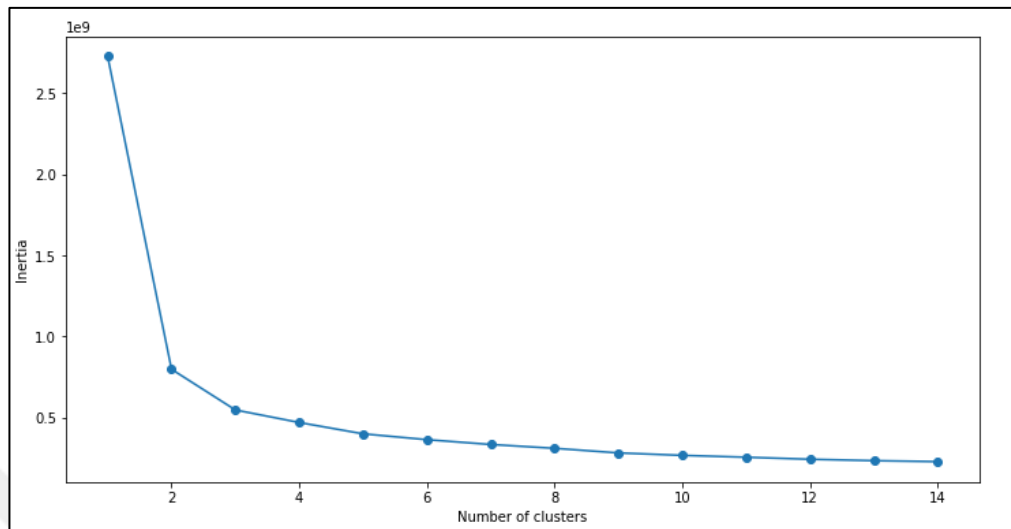


Figure 4.3: Finding optimum cluster number.

To see the effects of different k values on clustering and classification, $k = 5$ was chosen in the experiment. Then the cluster centroids found were sorted according to their distances. As a result of the classification of the new clusters, the confusion matrix in Figure 4.4 was obtained. As can be seen from the confusion matrix, it has been observed that clusters that are close to each other are misclassified. To verify this, nearby clusters were combined and labeled as a single class. In the subsequent classification, increased accuracy was observed in Figure 4.4.

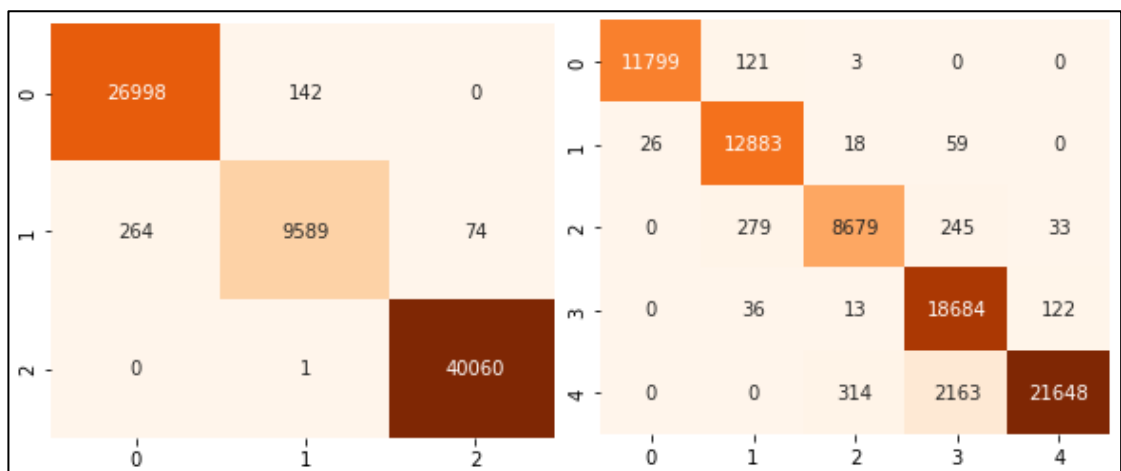


Figure 4.4: Combining closest clusters.

It has been determined that there are points that are very close to each other in the data set, and the filtering method is applied to see the effect of these points on clustering. A simple filtering algorithm is used for outlier points. According to this algorithm, the average of the distances of all the points belonging

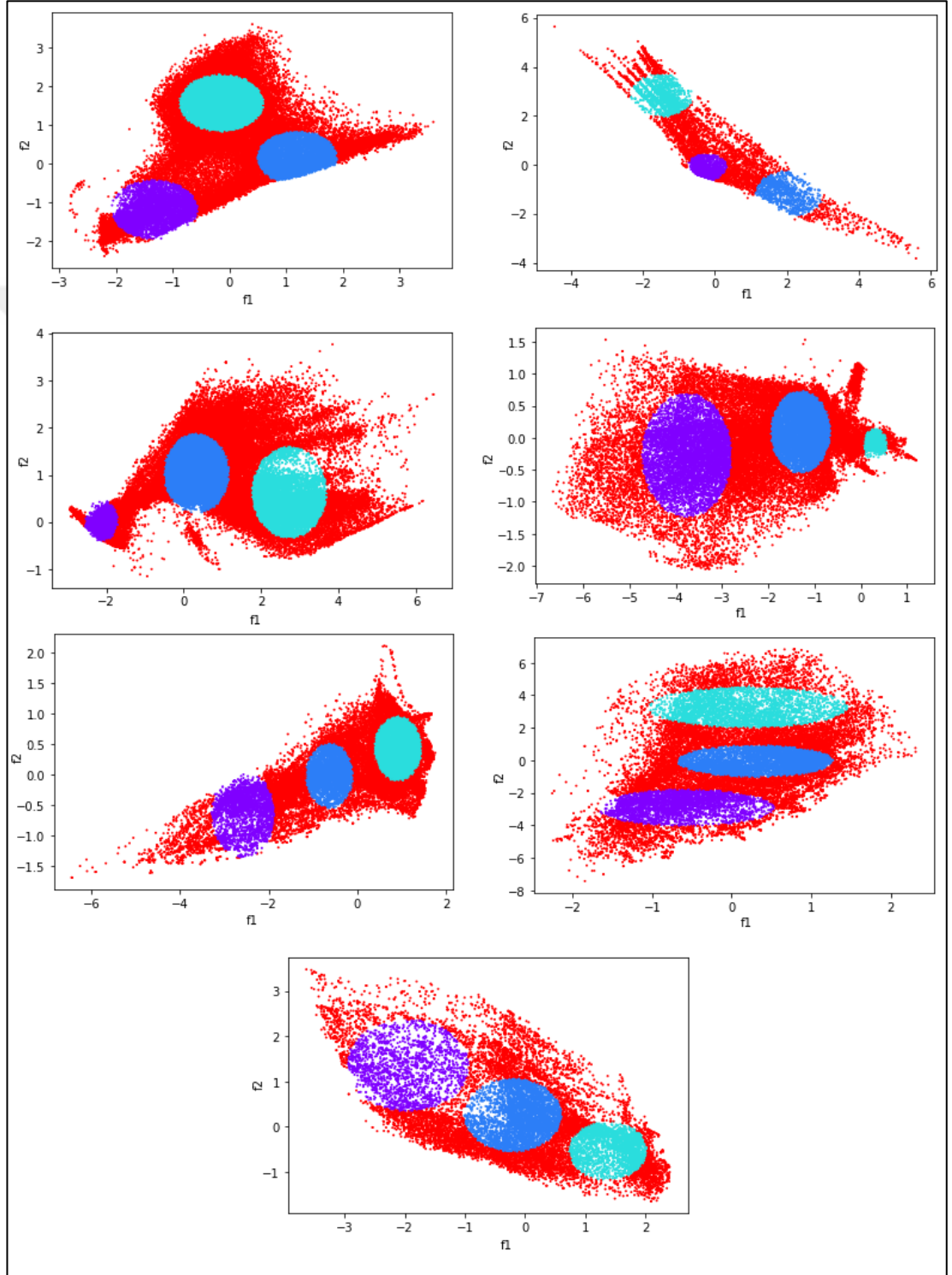


Figure 4.5: Clusters with filtering algorithm using encoder output.

the clusters was calculated as a result of clustering to the centroid. Then, the points that are far from the mean value calculated from the centroids are determined as outliers. The clusters obtained as a result of this process are visualized in Figure 4.5 and Figure 4.6.

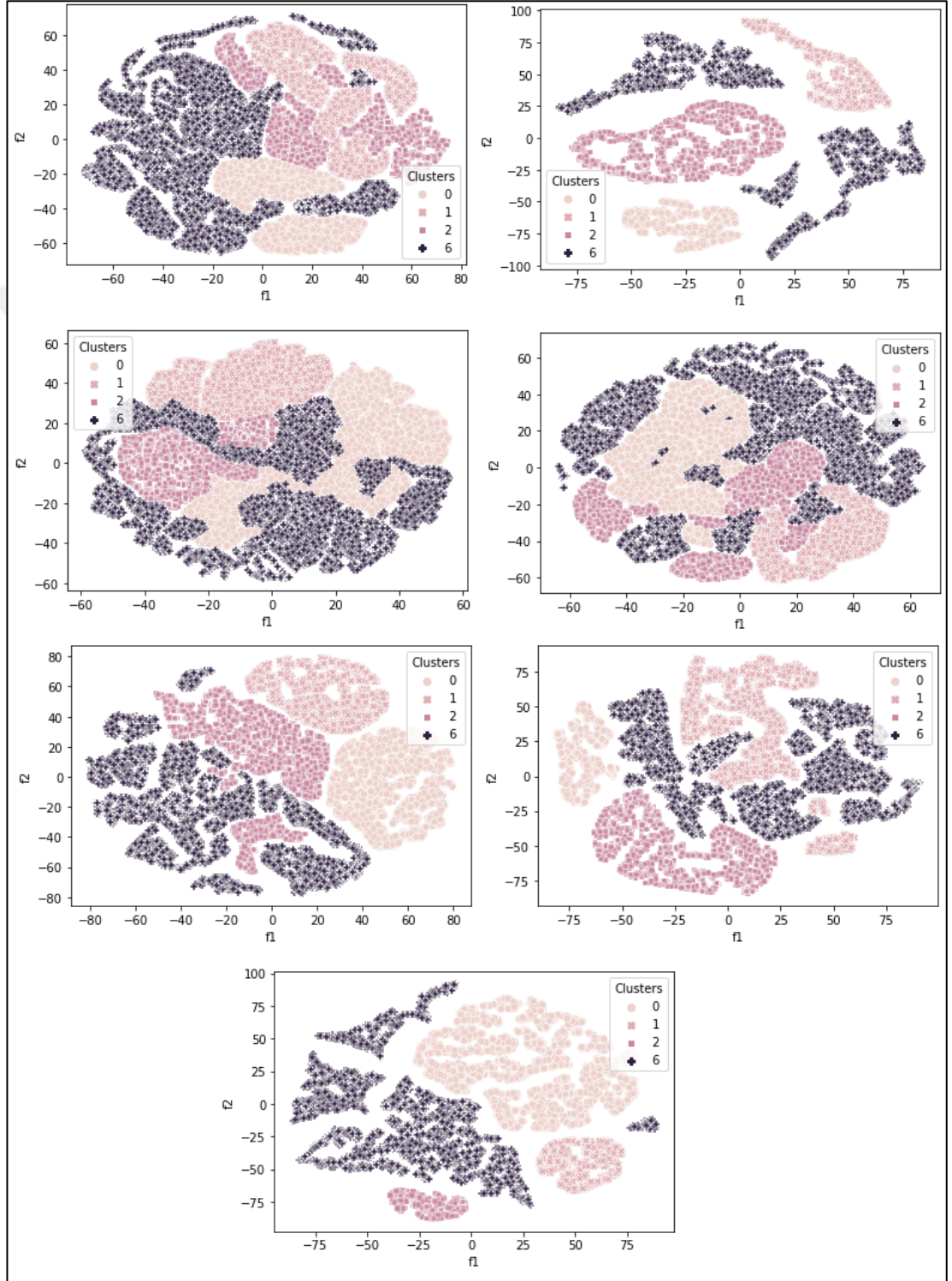


Figure 4.6: t-SNE results with filtering algorithm using encoder output.

In order to verify the results of the clustering process, 8 different experiments were carried out on the prepared data set. These experiments are as follows:

- E1: Raw data without filtering algorithm using Euclidean distance while training and testing, specified in Table 4.1.
- E2: Raw data with filtering algorithm using Euclidean distance while training and testing, specified in Table 4.2.
- E3: Encoder output without filtering algorithm using Euclidean distance while training and testing, specified in Table 4.3.
- E4: Encoder output with filtering algorithm using Euclidean distance while training and testing, specified in Table 4.4.
- E5: Raw data without filtering algorithm using Euclidean distance while training Mahalanobis distance while testing, specified in Table 4.5.
- E6: Raw data with filtering algorithm using Euclidean distance while training Mahalanobis distance while testing, specified in Table 4.6.
- E7: Encoder output without filtering algorithm using Euclidean distance while training Mahalanobis distance while testing, specified in Table 4.7.
- E8: Encoder output with filtering algorithm using Euclidean distance while training Mahalanobis distance while testing, specified in Table 4.8.

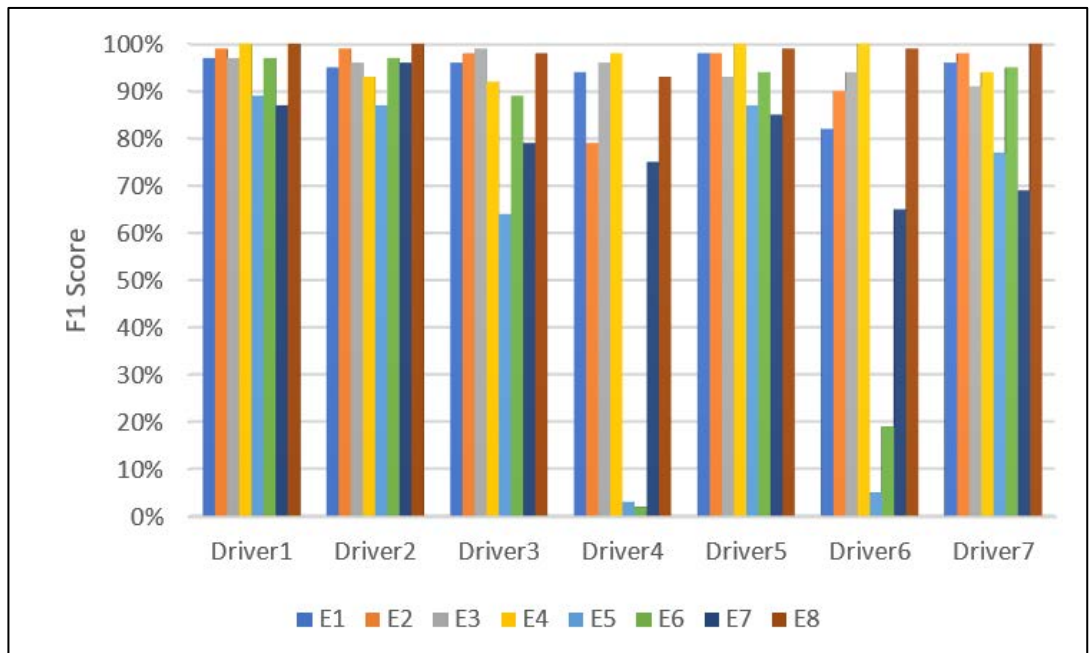


Figure 4.7: The worst f-score values obtained in the 8 experiments.

While proposed encoder output is used in the E3, E4, E7, and E8 experiments, preprocessed raw data is used in the E1, E2, E5, and E6. The worst F1-score values obtained in the classification of 3 clusters belonging to each driver are included in the Figure 4.7. When using Euclidean distance for both train and test data, clusters could be classified with high F1-score with preprocessed raw data. However, when the clustering results are visualized, it is seen that some points in the dataset were close to the centroids of two different clusters. In order to prevent misclustering of these points, which are close to two different cluster centroids, filtering algorithm was performed, and the results were obtained with E2 and E4 experiments. According to these results, it was observed that the average performance increased in both raw preprocessed data and proposed encoder output.

Since the collected dataset contains multivariate sensor data, only the Euclidean distance metric can lead to misleading results while clustering. It is well known issue that Mahalanobis distance is the distance between two points in multivariate space. It considers how correlated the variables are to one another. If the clustering results obtained are robust, close results should be obtained with the Mahalanobis distance metric. For this purpose, E5 and E7 experiments were conducted out. For example, although the clusters belonging to Driver1 were classified with 99% accuracy in first 4 experiments, it was observed that the F1-score value decreased by 10% in the E5 and E7 experiments. In the E6 and E8 experiments conducted for the same driver, it was observed that the performance of the clustering using Mahalanobis distance metric increased by removing the outlier points. However, the classification performance of some clusters belonging to Driver4 and Driver6 has decreased to 3%.

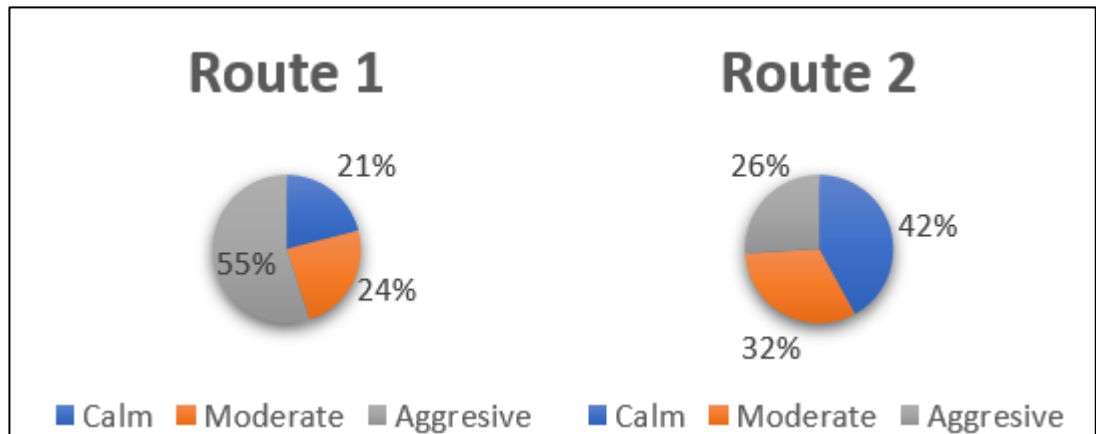


Figure 4.8: Aggressive driving route (left) and calm driving route (right).

As seen from these results, in the E4 and E8 experiments, the lowest f1-score value was 97%, while the average f1-score value was 99.57%.

Table 4.1: Clustering Experiment 1.

METHOD	CLS	D1	D2	D3	D4	D5	D6	D7
		F1S	F1S	F1S	F1S	F1S	F1S	F1S
RF	0	0.99	0.98	0.99	0.97	0.99	0.93	0.98
	1	0.99	0.98	0.97	0.96	0.99	0.88	0.99
	2	1.00	0.98	0.97	0.98	0.99	0.82	0.97
KNN	0	0.98	0.97	0.98	0.97	0.98	0.96	0.97
	1	0.97	0.95	0.96	0.94	0.98	0.92	0.98
	2	0.99	0.95	0.97	0.97	0.99	0.93	0.96
LGBM	0	0.99	0.96	0.99	0.98	0.99	0.95	0.98
	1	0.99	0.96	0.98	0.97	0.99	0.92	0.99
	2	1.00	0.96	0.98	0.99	0.99	0.92	0.98

Table 4.2: Clustering Experiment 2.

METHOD	CLS	D1	D2	D3	D4	D5	D6	D7
		F1S	F1S	F1S	F1S	F1S	F1S	F1S
RF	0	1.00	1.00	1.00	0.79	0.98	0.98	0.98
	1	1.00	0.99	0.98	0.88	0.98	0.97	0.99
	2	1.00	1.00	0.99	0.99	1.00	0.93	0.99
KNN	0	1.00	1.00	1.00	0.98	0.99	0.99	0.99
	1	1.00	1.00	0.99	0.98	0.99	0.97	1.00
	2	1.00	1.00	0.99	0.99	1.00	0.99	1.00
LGBM	0	1.00	0.99	1.00	0.93	0.99	0.96	0.99
	1	0.99	0.99	0.99	0.92	0.99	0.97	0.99
	2	1.00	1.00	0.99	0.98	1.00	0.90	0.98

In order to observe which behaviors, the obtained clusters represent, a driver was asked to drive aggressively and calmly, including 3 laps on a route of approximately 2 km. The distribution of the clusters obtained for the two drives is shown in Figure 4.8. Cluster distributions of 8 experiments are given in Figure A1.1, Figure A1.2, Figure A1.3, Figure A1.4, Figure A1.5, Figure A1.6, Figure A1.7, and Figure A1.8, respectively.

According to these results, it is assumed that the set "0" represents calm behavior, the set "1" represents moderate behavior, and the set "2" represents aggressive behavior. Behavior distributions of clustering experiments are illustrated in appendices section using these labels. To evaluate the results, the drivers participating in the experiment were asked to describe their own driving styles by conducting a questionnaire. According to this survey:

- Driver1 stated that while it was driving very aggressively or calmly when appropriate, depending on the current conditions.
- Driver2 stated that it could remain calm in all circumstances.
- Driver3 stated that it usually drives calmly but is very aggressive in traffic jam situations.
- Driver4 stated that it has a moderate driving style.
- Driver5 stated that it generally has a calm and moderate driving style.
- Driver6 stated that it has a calm driving style.
- Driver7 stated that it is prone to aggressive driving and in some cases, it drives extremely aggressively.

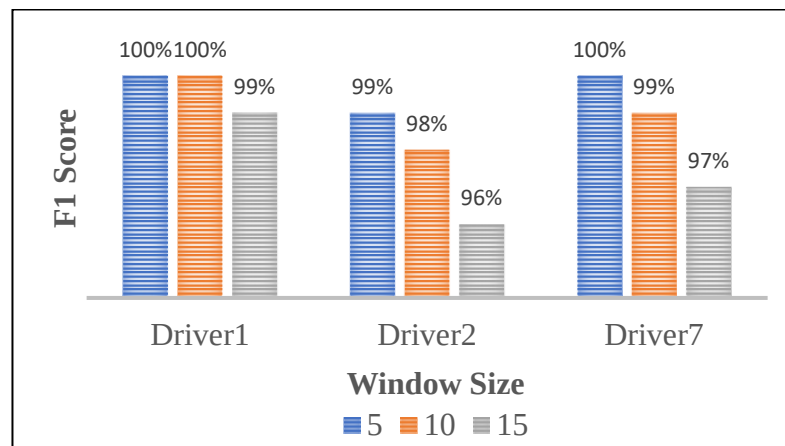


Figure 4.9: Up to three months of driving data results.

The drivers were surveyed to see what behaviors the 3 different patterns formed in the clustering algorithm represented. It was observed that the clustering distributions formed by the feedback received from the drivers and the methods used in the E4 and E8 matched. Since there was not any labelled data as a ground truth, internal evaluation measures were applied to validate the quality of the clustering results. The Silhouette score and Davies-Bouldin score metrics, which are widely used in the literature, were used as internal evaluation measures. The clustering method applied in each of the 8 experiments was evaluated with these techniques. The results are shown in Figure 4.11, Figure 4.12, Figure 4.13 and Figure 4.14. The closer the Silhouette score value is to 1, the better clustering is indicated, while a value close to 0 in the Davies-Bouldin score metric indicates better clustering. According to these results, it was seen that the performance increased in the raw data with the filtering algorithm. When the measurement distance is changed to Mahalanobis, it is seen that the data clustered with the proposed encoder network gives more robust results. Clustering methods used in the E4 and E8, which matched with the feedback of the drivers, both the Silhouette score and Davies-Bouldin score values were the best compared to the other experiments.

Table 4.3: Clustering Experiment 3.

METHOD	CLS	D1	D2	D3	D4	D5	D6	D7
		F1S	F1S	F1S	F1S	F1S	F1S	F1S
RF	0	0.98	0.99	0.99	0.99	0.99	0.98	0.95
	1	0.97	0.97	0.99	0.96	0.97	0.98	0.94
	2	0.98	0.96	0.99	0.97	0.93	0.94	0.91
KNN	0	0.98	0.98	0.99	0.99	0.99	0.98	0.95
	1	0.97	0.97	0.99	0.96	0.97	0.98	0.94
	2	0.98	0.96	0.99	0.96	0.93	0.94	0.91
LGBM	0	0.98	0.99	0.99	0.99	0.99	0.98	0.95
	1	0.97	0.97	0.99	0.96	0.97	0.98	0.94
	2	0.98	0.96	0.99	0.97	0.93	0.94	0.91

Another experiment was conducted to show the effect of the window size parameter in the sliding window method used in the data preparation phase on

clustering and classification. In this experiment, the window size parameter is divided into data sets of 5,10,15,20,30,45,60, and 120 seconds. Clustering results that are given at Table A1.1 were classified using the FCN architecture, which had better result in

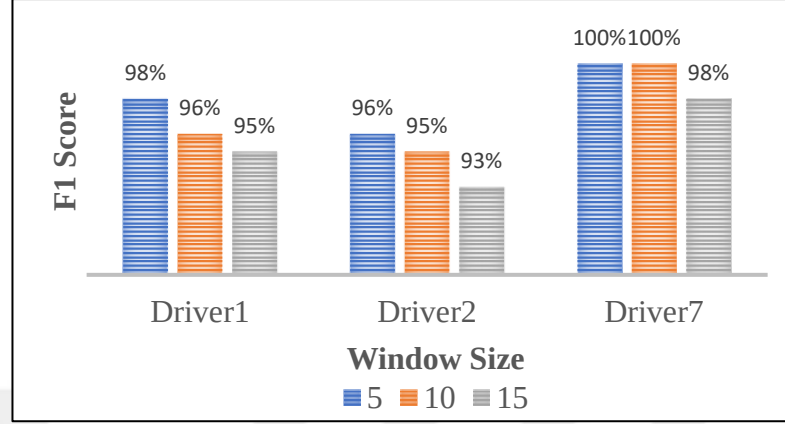


Figure 4.10: Fifteen minutes of driving data results.

different window sizes. Only three different drivers' driving data results are shown in Figure 4.9 and Figure 4.10. To overcome the imbalanced data problem, a study was also conducted on randomly selected 15-minute data for 3 different drivers.

Table 4.4: Clustering Experiment 4.

METHOD	CLS	D1	D2	D3	D4	D5	D6	D7
		F1S	F1S	F1S	F1S	F1S	F1S	F1S
RF	0	1.00	1.00	0.99	0.98	1.00	1.00	0.99
	1	1.00	0.97	0.95	0.99	1.00	1.00	0.97
	2	1.00	0.93	0.96	1.00	1.00	1.00	0.94
KNN	0	1.00	1.00	0.98	1.00	1.00	1.00	0.99
	1	1.00	1.00	0.96	1.00	1.00	1.00	0.99
	2	1.00	1.00	0.97	1.00	1.00	1.00	1.00
LGBM	0	1.00	0.97	0.96	1.00	1.00	1.00	0.97
	1	1.00	0.94	0.92	1.00	1.00	1.00	0.97
	2	1.00	0.94	0.95	1.00	1.00	1.00	1.00

Table 4.5: Clustering Experiment 5.

METHOD	CLS	D1	D2	D3	D4	D5	D6	D7
		F1S	F1S	F1S	F1S	F1S	F1S	F1S
RF	0	0.89	0.92	0.84	0.03	0.88	0.74	0.88
	1	0.92	0.87	0.64	0.55	0.88	0.11	0.91
	2	0.90	0.92	0.83	0.46	0.90	0.05	0.77
KNN	0	0.89	0.91	0.84	0.03	0.88	0.73	0.88
	1	0.92	0.87	0.64	0.55	0.87	0.13	0.92
	2	0.90	0.91	0.83	0.46	0.90	0.04	0.78
LGBM	0	0.89	0.92	0.84	0.03	0.89	0.73	0.87
	1	0.92	0.87	0.64	0.55	0.88	0.12	0.91
	2	0.90	0.93	0.83	0.47	0.90	0.05	0.78

Table 4.6: Cluster Experiment 6.

METHOD	CLS	D1	D2	D3	D4	D5	D6	D7
		F1S	F1S	F1S	F1S	F1S	F1S	F1S
RF	0	0.98	0.99	0.98	0.02	0.95	0.96	0.97
	1	0.98	0.98	0.89	0.69	0.94	0.23	0.98
	2	1.00	1.00	0.95	0.63	1.00	0.60	0.95
KNN	0	0.99	0.98	0.98	0.02	0.94	0.98	0.98
	1	0.99	0.97	0.91	0.72	0.94	0.36	0.99
	2	1.00	0.99	0.96	0.69	1.00	0.45	0.98
LGBM	0	0.98	0.99	0.98	0.02	0.97	0.97	0.98
	1	0.97	0.98	0.91	0.65	0.97	0.35	0.98
	2	0.99	1.00	0.97	0.55	1.00	0.19	0.95

Table 4.7: Clustering Experiment 7.

METHOD	CLS	D1	D2	D3	D4	D5	D6	D7
		F1S	F1S	F1S	F1S	F1S	F1S	F1S
RF	0	0.87	0.98	0.91	0.96	0.91	0.65	0.86
	1	0.89	0.96	0.79	0.76	0.85	0.96	0.69
	2	0.97	0.97	0.86	0.86	0.85	0.91	0.88
KNN	0	0.87	0.98	0.91	0.96	0.91	0.65	0.86
	1	0.89	0.96	0.79	0.75	0.85	0.96	0.69
	2	0.97	0.97	0.86	0.86	0.85	0.91	0.88
LGBM	0	0.87	0.98	0.91	0.96	0.91	0.65	0.86
	1	0.90	0.96	0.79	0.75	0.85	0.96	0.69
	2	0.87	0.97	0.86	0.86	0.85	0.91	0.88

Table 4.8: Clustering Experiment 8.

METHOD	CLS	D1	D2	D3	D4	D5	D6	D7
		F1S	F1S	F1S	F1S	F1S	F1S	F1S
RF	0	1.00	1.00	0.99	0.98	0.99	0.96	1.00
	1	1.00	1.00	0.98	0.99	0.99	0.97	1.00
	2	1.00	1.00	1.00	1.00	1.00	0.98	1.00
KNN	0	1.00	1.00	1.00	0.98	1.00	0.93	1.00
	1	1.00	1.00	1.00	0.99	1.00	0.93	1.00
	2	1.00	1.00	1.00	1.00	1.00	0.97	1.00
LGBM	0	1.00	1.00	0.99	0.99	0.99	0.95	1.00
	1	1.00	1.00	0.98	0.98	0.99	0.97	1.00
	2	1.00	1.00	1.00	1.00	1.00	0.98	1.00

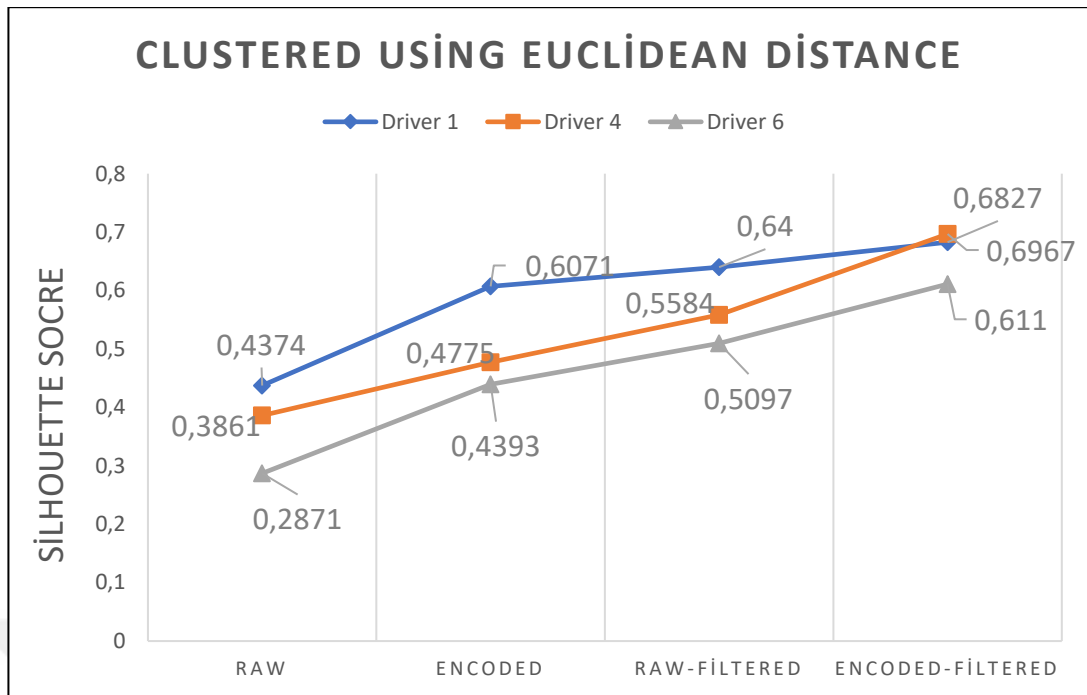


Figure 4.11: Silhouette scores of experiments E1, E3, E2 and E4, respectively. Using encoded data with filtering algorithm has best score.

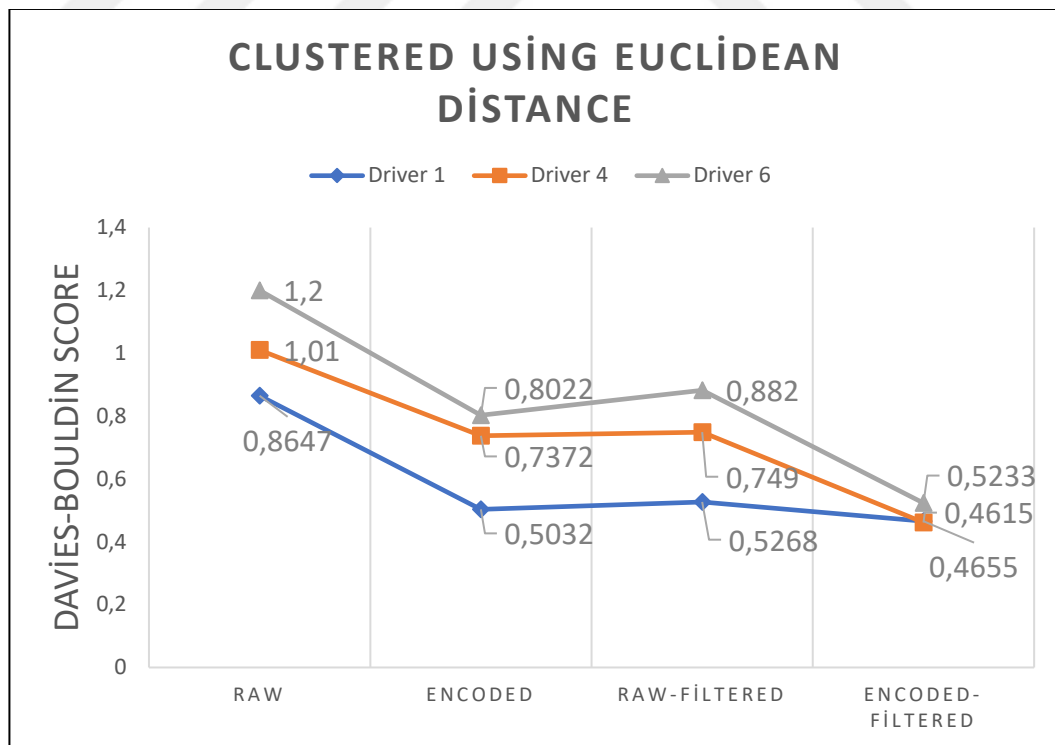


Figure 4.12: Davies-Bouldin scores of experiments E1, E3, E2 and E4, respectively. Using encoded data with filtering algorithm has best score.

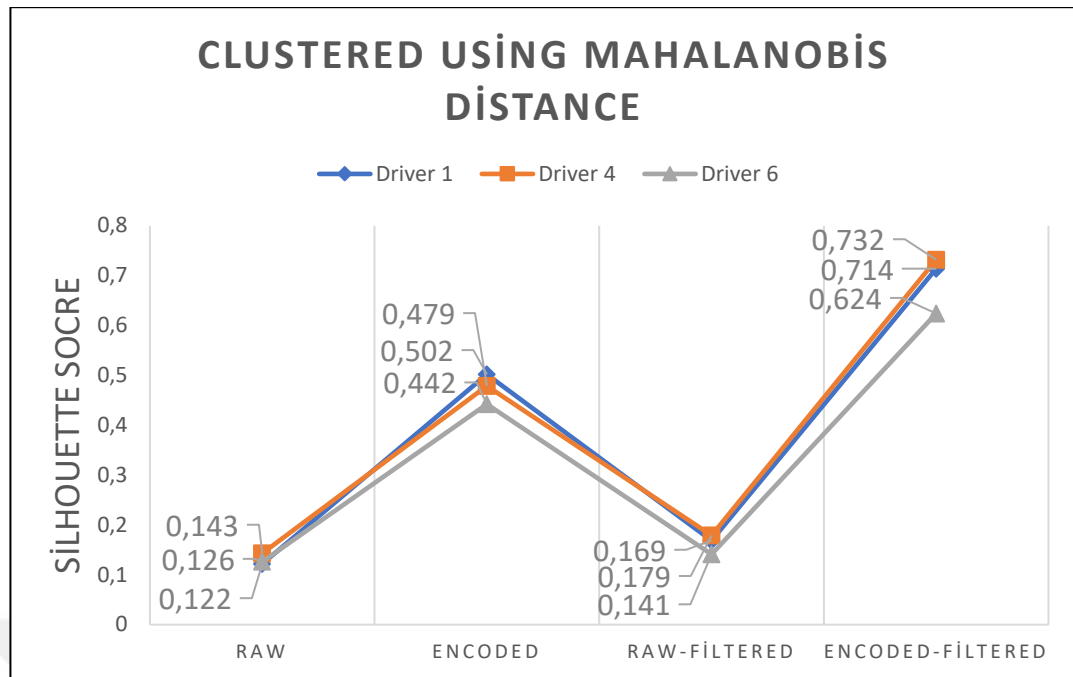


Figure 4.13: Silhouette scores of experiments E5, E7, E6 and E8, respectively. Using encoded data with filtering algorithm has best score. We also highlight that, in clustering using Mahalanobis distance, data encoding produced more effective results than the filtering algorithm.

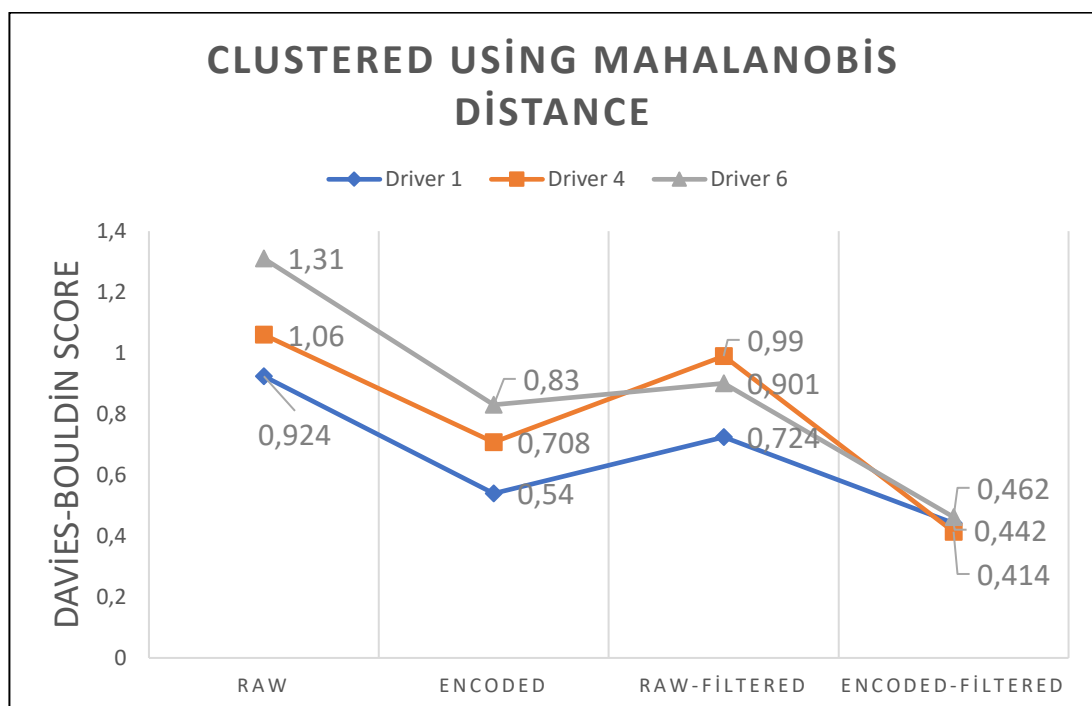


Figure 4.14: Davies-Bouldin scores of experiments E5, E7, E6 and E8, respectively. Using encoded data with filtering algorithm has best score.

5. CONCLUSION AND FUTURE WORK

In this thesis, driver identification has been performed with FCN architecture using vehicle sensor data. First, the driver identification problem has been handled with a new dataset collected from five different drivers consisting of different routes in real time. With the FCN architecture, which has been also applied in the datasets used in the literature, the performance results increase, while the evaluation and training costs have been reduced. In the UYANIK dataset consisting of a single route, the worst success rate has been obtained at 58.15% in the dataset consisting of 88 drivers, by using the timeseries segmentation method (using one part of the route in the training phase and the other part in the test phase) along with the stratified k-fold validation.

Secondly, by establishing an automated data collection pipeline with custom-designed low-cost hardware, we have collected naturalistic (uncontrolled environment) driving data involving seven individual drivers for up to three months. This collected data has been labeled automatically with unsupervised techniques instead of manual labeling. During the labeling process, the temporal information of the multivariate timeseries was represented with lower-dimensional vectors with the proposed autoencoder architecture, while a filtering algorithm is proposed to reduce the number of misclustered instances. The results obtained by conducting a survey to understand what behaviors the clusters formed as a result of clustering represent, were taken as ground truth. It has been determined that the clusters obtained in the experiments with the proposed encoding and filtering techniques match the feedback received from the drivers. Furthermore, while examining all the test results with inertial evaluation techniques, the best results were obtained with the proposed encoding and filtering techniques. With the proposed methods, a robust solution has been produced against different distance metrics and outlier points. While evaluating the classification results, the timeseries segmentation method instead of the cross-validation technique has been used on the data that the model had never seen. In addition, using the FCN architecture suggested for the driver identification problem, the effect of the window size parameter has been observed, and it is seen that the success of classification and clustering increases at low window sizes.

Because it is very difficult to change driving habits, a human-centered driving environment is needed to avoid dangerous situations. By understanding the relationship between traffic conditions, road infrastructure, and driver behavior, improvements can be made to smart transportation systems. The important thing is to be able to create driver profiles. Knowing how people can react will also be beneficial in the development of self-driving vehicles. It's challenging to draw a straight line between the performance of the proposed methodologies and the ambiguity and unpredictability of human behavior. For this reason, we aim to extend our work as examining the effects of environmental and human-related factors on behavioral patterns as future study.



REFERENCES

- [1] Azadani M. N., Boukerche A., (2021), “Driving Behavior Analysis Guidelines for Intelligent Transportation Systems”, IEEE Transactions on Intelligent Transportation Systems, 1-19.
- [2] Web 1, (2021), <http://www.totalcardiagnostics.com/support/Knowledgebase/Article/View/43/6/reading-performance-information-data-pid>, (Accessed on: 11/10/2021).
- [3] Liu Z., He G., (2005), “A vehicle anti-theft and alarm system based on computer vision”, IEEE International Conference on Vehicular Electronics and Safety, 326-330, Shaanxi, China, 14-16 October.
- [4] Elngar A. A., Kayed M., (2020), “Vehicle security systems using face recognition based on Internet of Things”, Open Computer Science, 10(1), 17-29.
- [5] Inalegwu O. C., Maliki D., Agajo J., Ajao L. A., Abu A. D., (2018), “Fingerprint Based Driver's Identification System”, 2nd International Conference on Information and Communication Technology and Its Applications (ICTA 2018), 238-242, Minna, Nigeria, 5-6 September.
- [6] Rahim M. A., Zhu L., Li X., Liu J., Zhang Z., Qin Z., Gai K., (2019), “Zero-to-stable driver identification: A non-intrusive and scalable driver identification scheme”, IEEE transactions on vehicular technology, 69(1), 163-171.
- [7] Kwak B. I., Woo J., Kim H. K., (2016), “Know your master: Driver profiling-based anti-theft method”, IEEE 14th Annual Conference on Privacy, Security and Trust (PST), 211-218, Auckland, New Zealand, 12-14 December.
- [8] Abut H., Erdoğan H., Erçil A., Çürüklü B., Koman H. C., Taş F., Karabat Ç. (2009), “Real-world data collection with UYANIK”, In-Vehicle Corpus and Signal Processing for Driver Behavior, Springer, 23-43.
- [9] Karataş B., (2018), “Araç sürüş verilerinden makine öğrenmesi tekniklerini kullanarak sürücü sınıflandırma”, Master's thesis, TOBB ETÜ Fen Bilimleri Enstitüsü.
- [10] Dhariyal B., Le Nguyen T., Gsponer S., Ifrim G., (2020), “An Examination of the State-of-the-Art for Multivariate Time Series Classification”, 2020 IEEE International Conference on Data Mining Workshops, 243-250, Sorrento, Italy, 17-20 November.
- [11] Zhang J., Wu Z., Li F., Xie C., Ren T., Chen J., Liu L., (2019), “A deep learning framework for driving behavior identification on in-vehicle CAN-BUS sensor data”, Sensors, 19(6), 1356.

- [12] Ullah S., Kim D. H., (2020), "Lightweight driver behavior identification model with sparse learning on In-Vehicle CAN-BUS sensor data", *Sensors*, 20(18), 5030.
- [13] El Mekki A., Bouhoute A., Berrada I., (2019), "Improving driver identification for the next-generation of in-vehicle software systems", *IEEE Transactions on Vehicular Technology*, 68(8), 7406-7415.
- [14] Terzi R., Sağıroğlu Ş., Çöçü Ö., Arkan R., Tosun M., Tulgar Y., (2021), "Filo yönetimi için büyük veri temelli yeni sürücü davranış modelleri", *Gazi Üniversitesi Mühendislik Mimarlık Fakültesi Dergisi*, 36(1), 543-558.
- [15] Bejani M. M., Ghatee M., (2018), "A context aware system for driving style evaluation by an ensemble learning on smartphone sensors data", *Transportation Research Part C: Emerging Technologies*, 89, 303-320.
- [16] Wang W., Xi J., Zhao D., (2018), "Driving style analysis using primitive driving patterns with Bayesian nonparametric approaches", *IEEE Transactions on Intelligent Transportation Systems*, 20(8), 2986-2998.
- [17] Magaña V. C., Muñoz O. M., (2015), "Artemisa: A personal driving assistant for fuel saving", *IEEE Transactions on Mobile Computing*, 15(10), 2437-2451.
- [18] Wickramanayake S., Bandara H. D., (2016), "Fuel consumption prediction of fleet vehicles using machine learning: A comparative study", 2016 IEEE Moratuwa Engineering Research Conference, 90-95, Moratuwa, Sri Lanka, 5-6 April.
- [19] Liu T., Yang Y., Huang G. B., Yeo Y. K., Lin Z., (2015), "Driver distraction detection using semi-supervised machine learning", *IEEE transactions on intelligent transportation systems*, 17(4), 1108-1120.
- [20] Li N., Jain J. J., Busso C., (2013), "Modeling of driver behavior in real world scenarios using multiple noninvasive sensors", *IEEE transactions on multimedia*, 15(5), 1213-1225.
- [21] Martinelli F., Mercaldo F., Nardone V., Orlando A., Santone A., (2018), "Cluster analysis for driver aggressiveness identification", *ICISSP*, 562-569, Funchal, Madeira, Portugal, 22-24 January.
- [22] Peppes N., Alexakis T., Adamopoulou E., Demestichas K., (2021), "Driving behaviour analysis using machine and deep learning methods for continuous streams of vehicular data", *Sensors*, 21(14), 4704.
- [23] Bender A., Agamennoni G., Ward J. R., Worrall S., Nebot E. M., (2015), "An unsupervised approach for inferring driver behavior from naturalistic driving data", *IEEE transactions on intelligent transportation systems*, 16(6), 3325-3336.

- [24] Choi Y. A., Park K. H., Park E., Kim H. K., (2021), “Unsupervised Driver Behavior Profiling Leveraging Recurrent Neural Networks”, International Conference on Information Security Applications, 28-38, Jesu Island, South Korea, 11-13 August.
- [25] Fugiglando U., Massaro E., Santi P., Milardo S., Abida K., Stahlmann R., Ratti C., (2018), “Driving behavior analysis through CAN bus data in an uncontrolled environment”, IEEE Transactions on Intelligent Transportation Systems, 20(2), 737-748.
- [26] Rousseeuw P. J., (1987), “Silhouettes: a graphical aid to the interpretation and validation of cluster analysis”, Journal of computational and applied mathematics, 20(C), 53-65.
- [27] De Maesschalck R., Rimbaud J. D., Massart D. L., (2000), “The mahalanobis distance”, Chemometrics and intelligent laboratory systems, 50(1), 1-18.
- [28] Van der Maaten L., Hinton G., (2008), “Visualizing data using t-SNE”, Journal of machine learning research, 9(2605), 2579-2605.
- [29] Ke G., Meng Q., Finley T., Wang T., Chen W., Ma W., Liu T. Y., (2017), “Lightgbm: A highly efficient gradient boosting decision tree”, Advances in neural information processing systems, Long Beach, CA, USA, 4-7 December.

BIOGRAPHY

Caner Aslan earned a Bachelor of Science degree in Computer Engineering from Gebze Technical University in 2016. In 2018, he started his masters studies at Gebze Technical University in Computer Engineering. Since 2015, he has been with Ortem Elektronik in Istanbul, Turkey as a computer engineer. His research interests include machine learning and deep learning in smart and connected vehicles.



APPENDICES

Appendix A: Driver Behavior Detection Additional Figures and Tables

Table A1.1: Window size parameter comparison using FCN.

METHOD	CLS	D1	D2	D3	D4	D5	D6	D7
		F1S	F1S	F1S	F1S	F1S	F1S	F1S
5	0	0.99	0.99	0.99	0.99	0.99	0.99	0.99
	1	0.99	0.99	0.99	0.99	0.99	0.99	0.99
	2	0.99	0.99	0.99	0.99	0.99	0.99	0.99
10	0	0.99	0.98	0.99	0.99	0.99	0.99	0.99
	1	0.99	0.98	0.98	0.99	0.98	0.98	0.98
	2	0.99	0.99	0.99	0.99	0.99	0.98	0.99
15	0	0.99	0.98	0.99	0.97	0.99	0.98	0.97
	1	0.99	0.95	0.98	0.98	0.98	0.93	0.94
	2	0.99	0.95	0.99	0.99	0.98	0.97	0.98
20	0	0.99	0.99	0.99	0.97	0.98	0.98	0.97
	1	0.98	0.98	0.97	0.98	0.98	0.92	0.94
	2	0.99	0.97	0.99	0.99	0.98	0.98	0.97
30	0	0.98	0.97	0.99	0.95	0.98	0.98	0.98
	1	0.98	0.95	0.96	0.97	0.96	0.92	0.94
	2	0.98	0.93	0.98	0.98	0.96	0.97	0.97
45	0	0.98	0.97	0.96	0.97	0.99	0.97	0.96
	1	0.98	0.96	0.93	0.96	0.97	0.91	0.93
	2	0.98	0.92	0.95	0.98	0.95	0.92	0.97
60	0	0.98	0.98	0.98	0.96	0.98	0.92	0.96
	1	0.98	0.96	0.94	0.94	0.96	0.63	0.93
	2	0.98	0.85	0.96	0.97	0.95	0.93	0.97
120	0	0.98	1.00	0.97	0.97	0.97	0.92	0.96
	1	0.96	0.91	0.91	0.94	0.95	0.65	0.93
	2	0.96	0	0.96	0.97	0.93	0.83	0.92

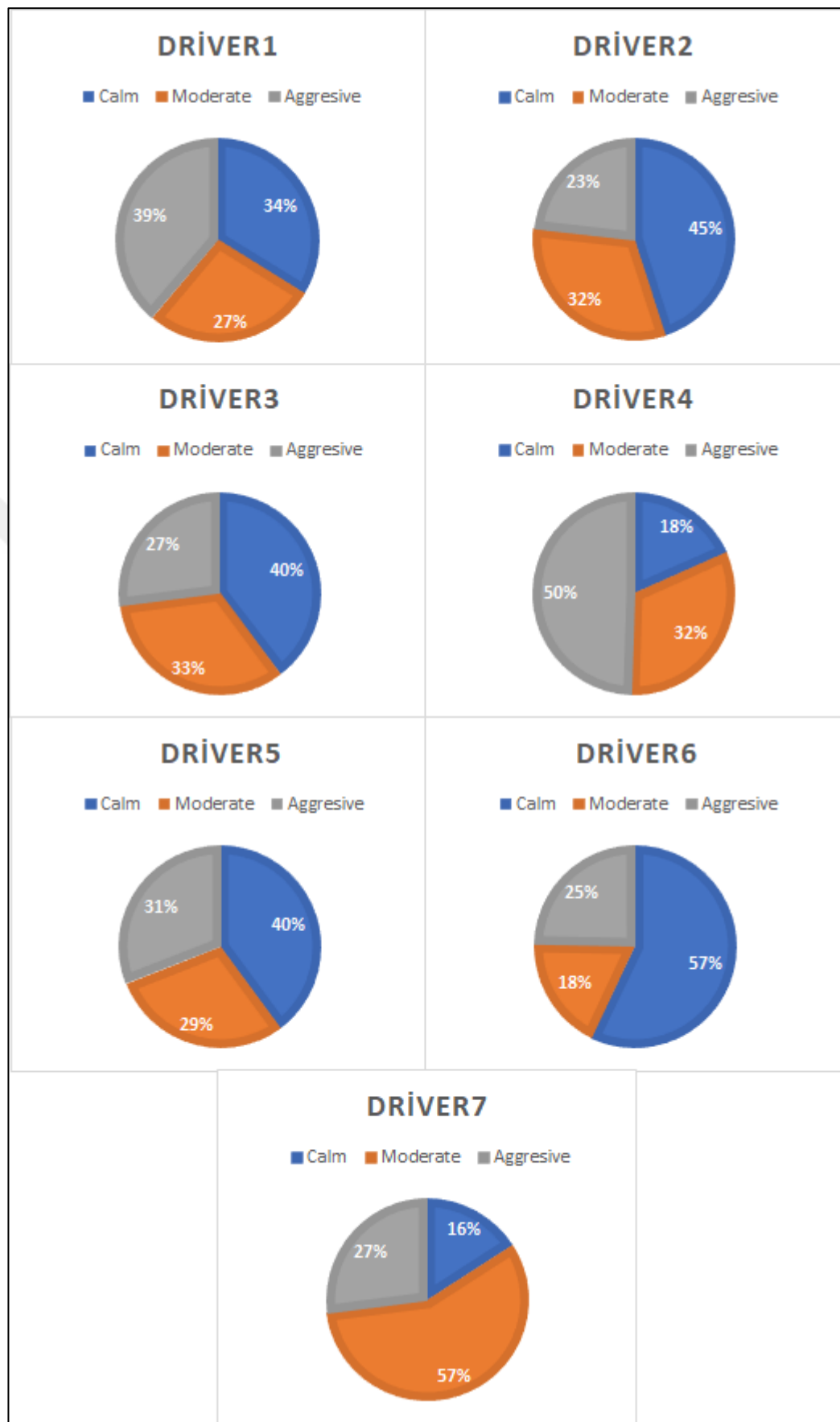


Figure A1.1: Cluster distributions in Clustering Experiment 1.

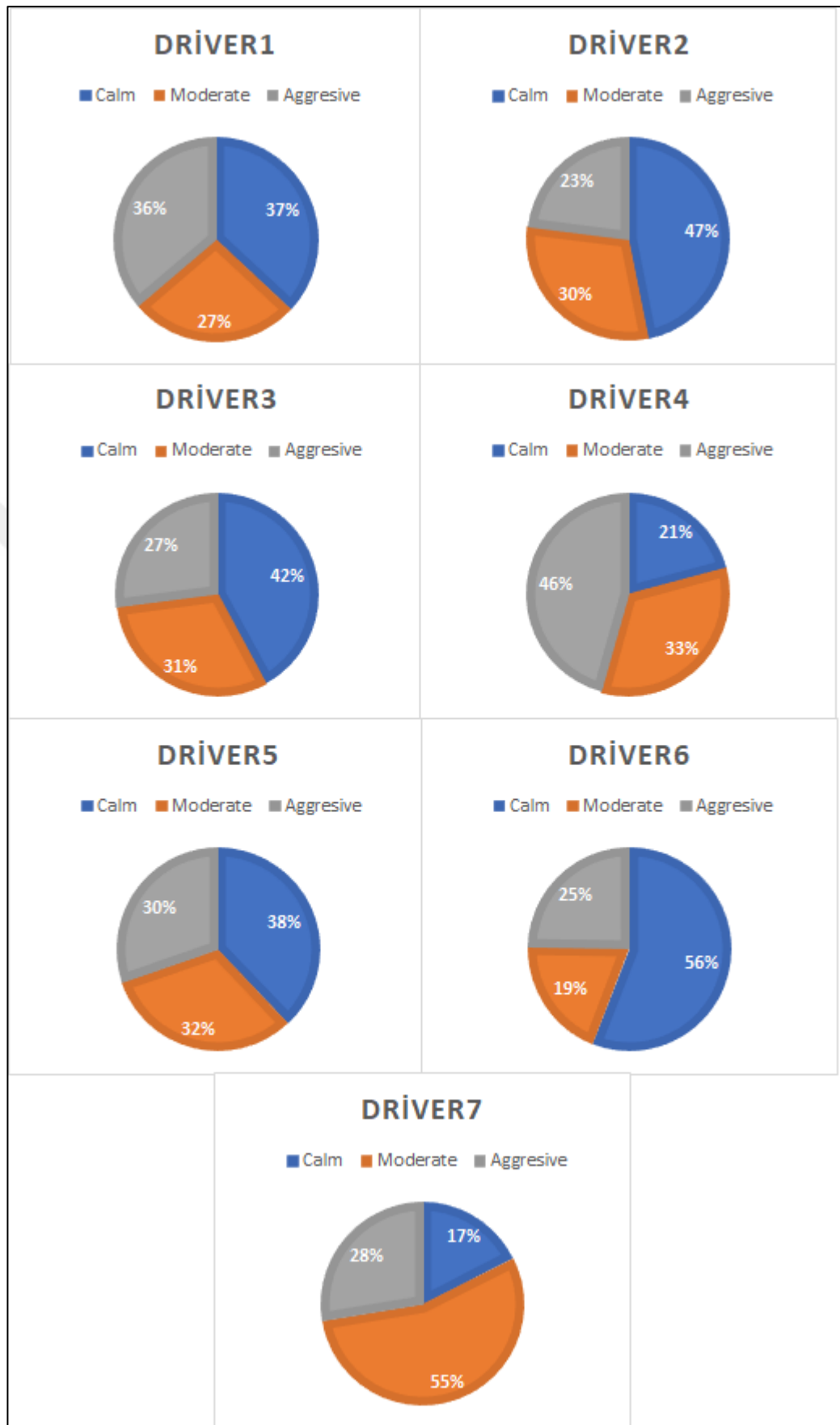


Figure A1.2: Cluster distributions in Clustering Experiment 2.

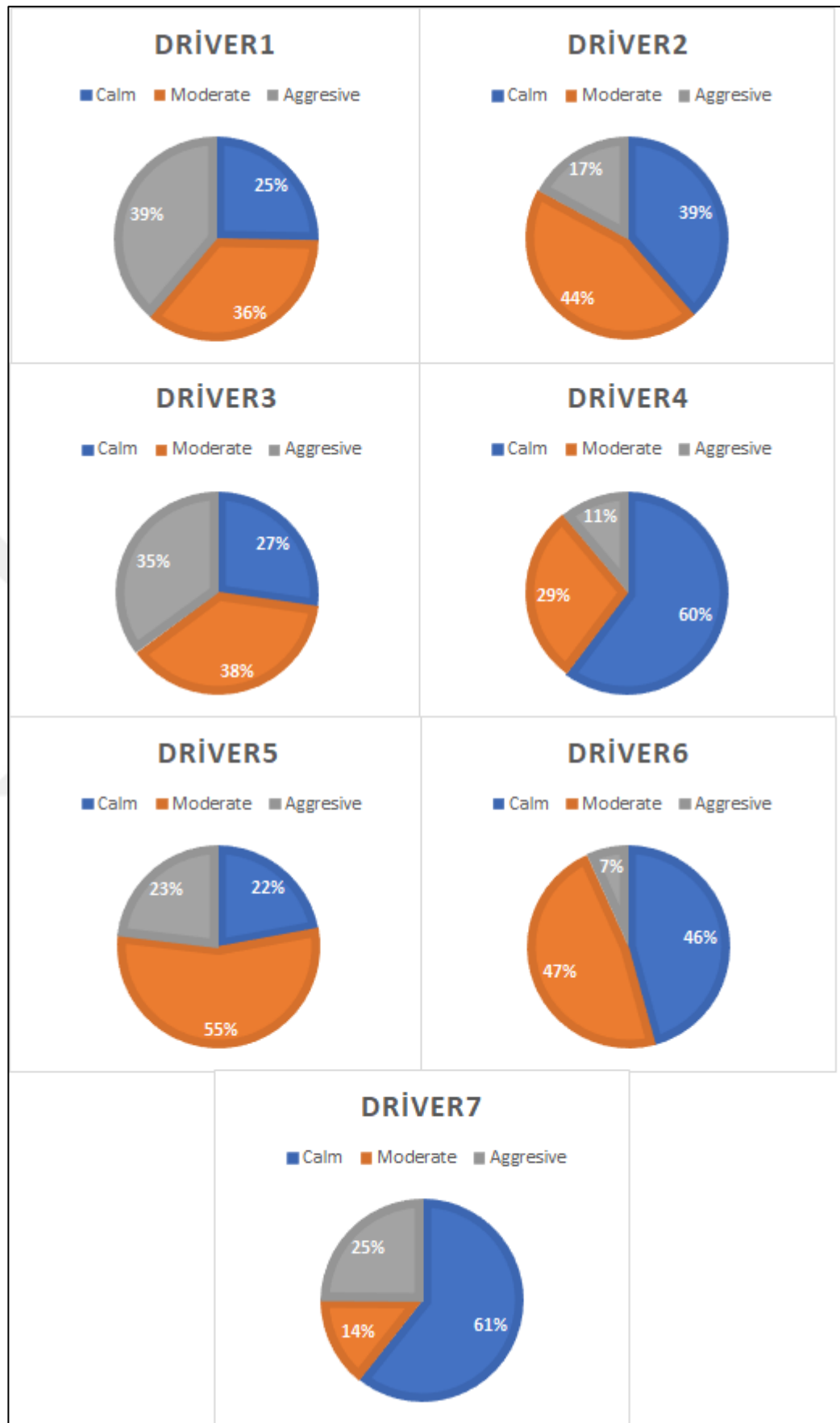


Figure A1.3: Cluster distributions in Clustering Experiment 3.

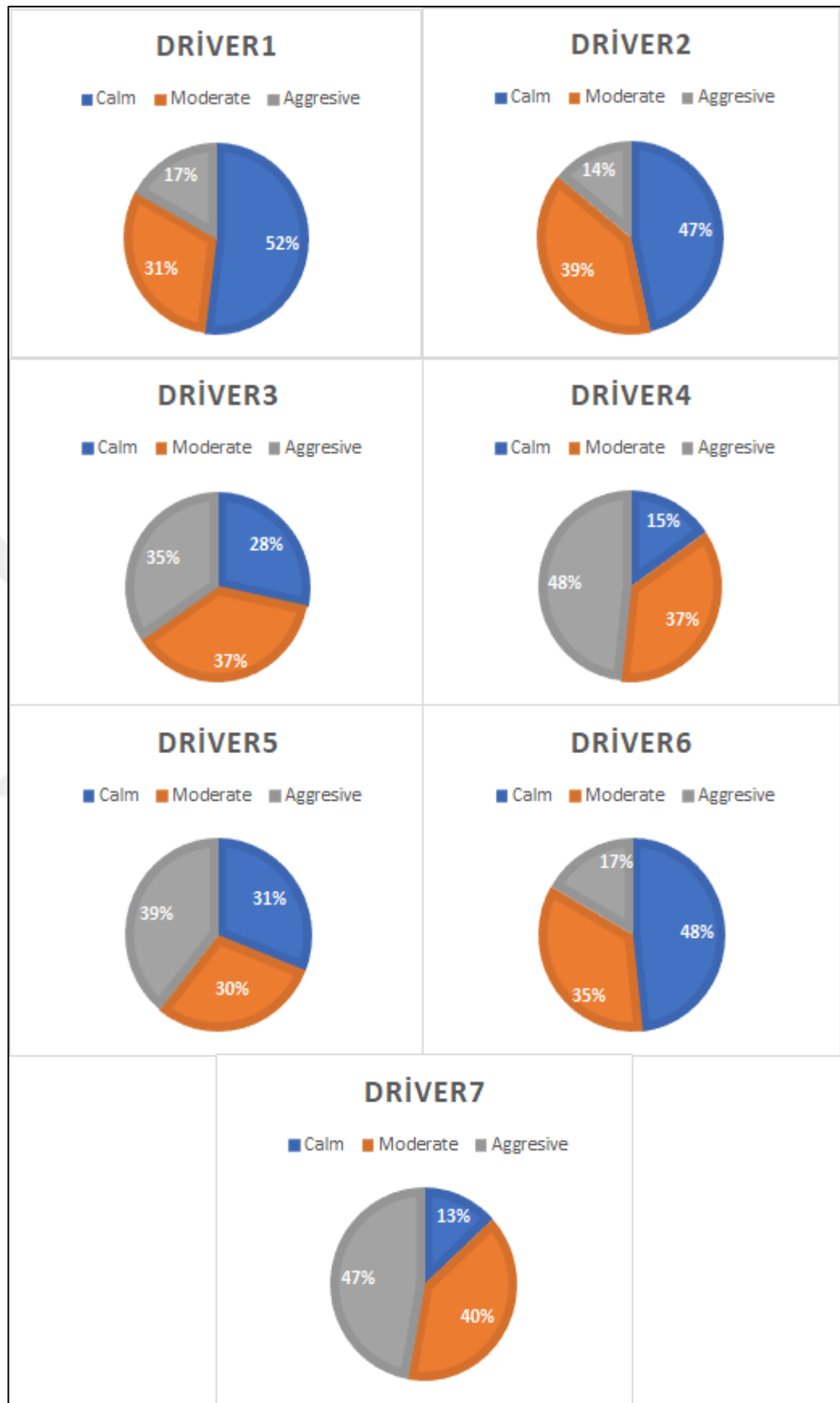


Figure A1.4: Cluster distributions in Clustering Experiment 4.

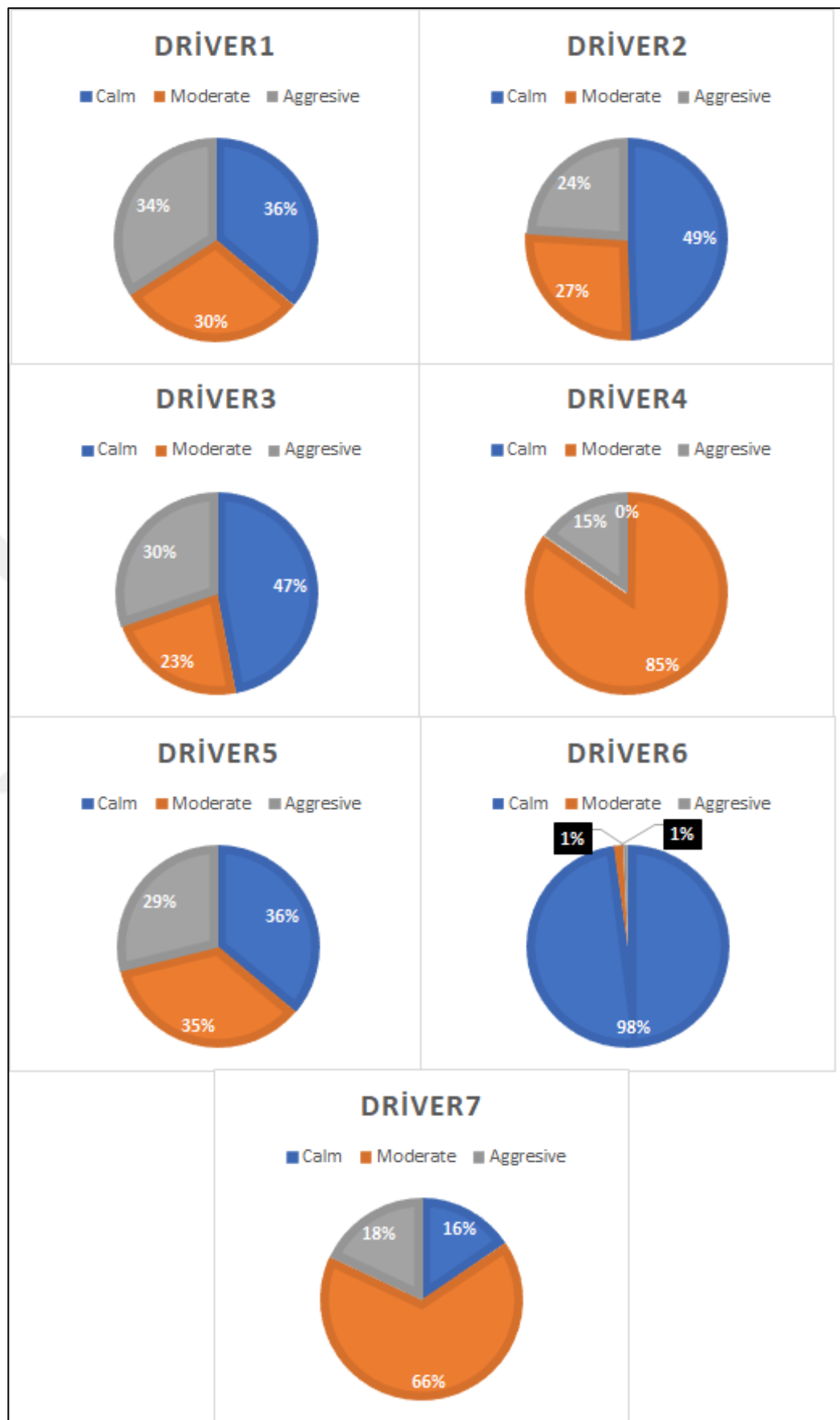


Figure A1.5: Cluster distributions in Clustering Experiment 5.

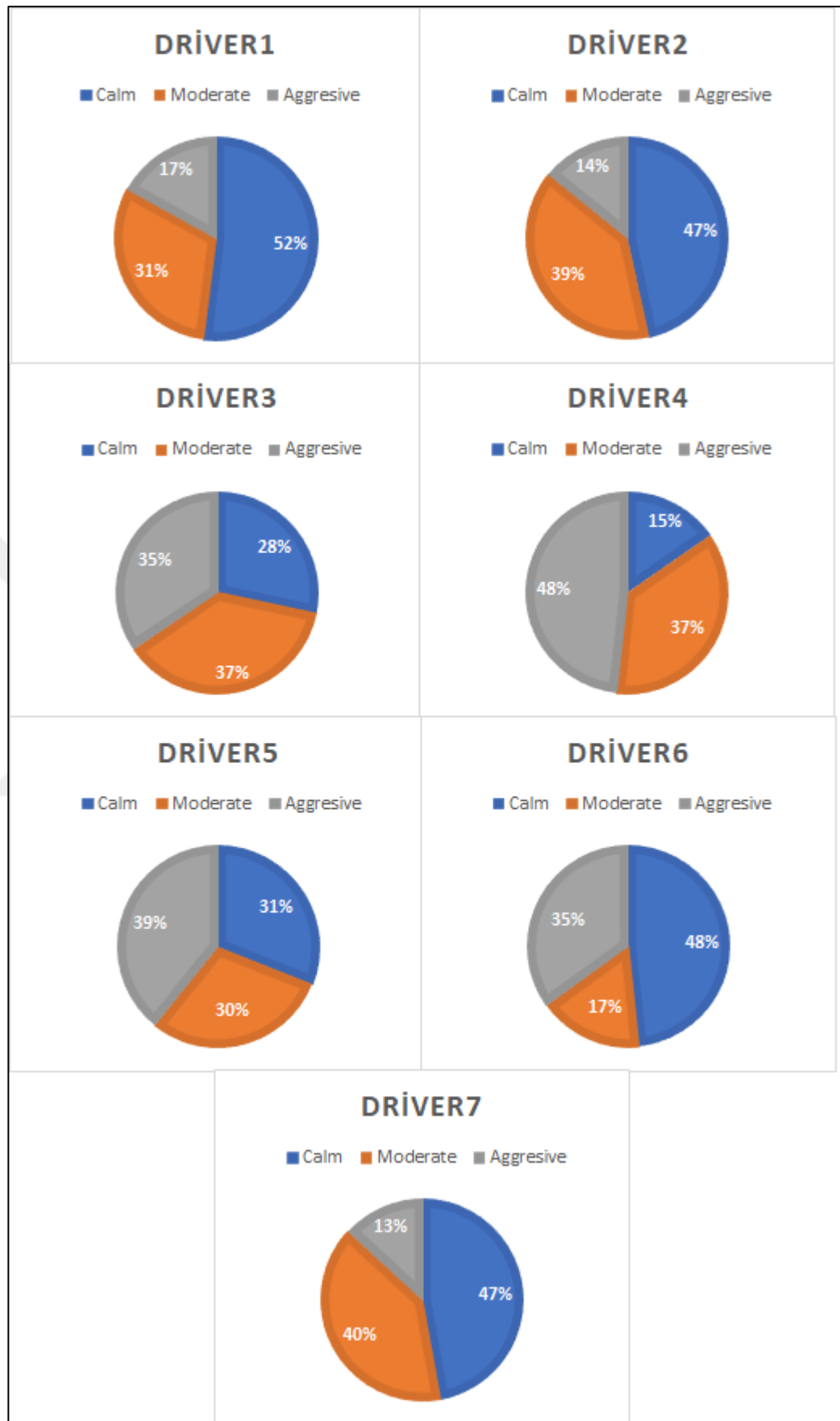


Figure A1.6: Cluster distributions in Clustering Experiment 6.

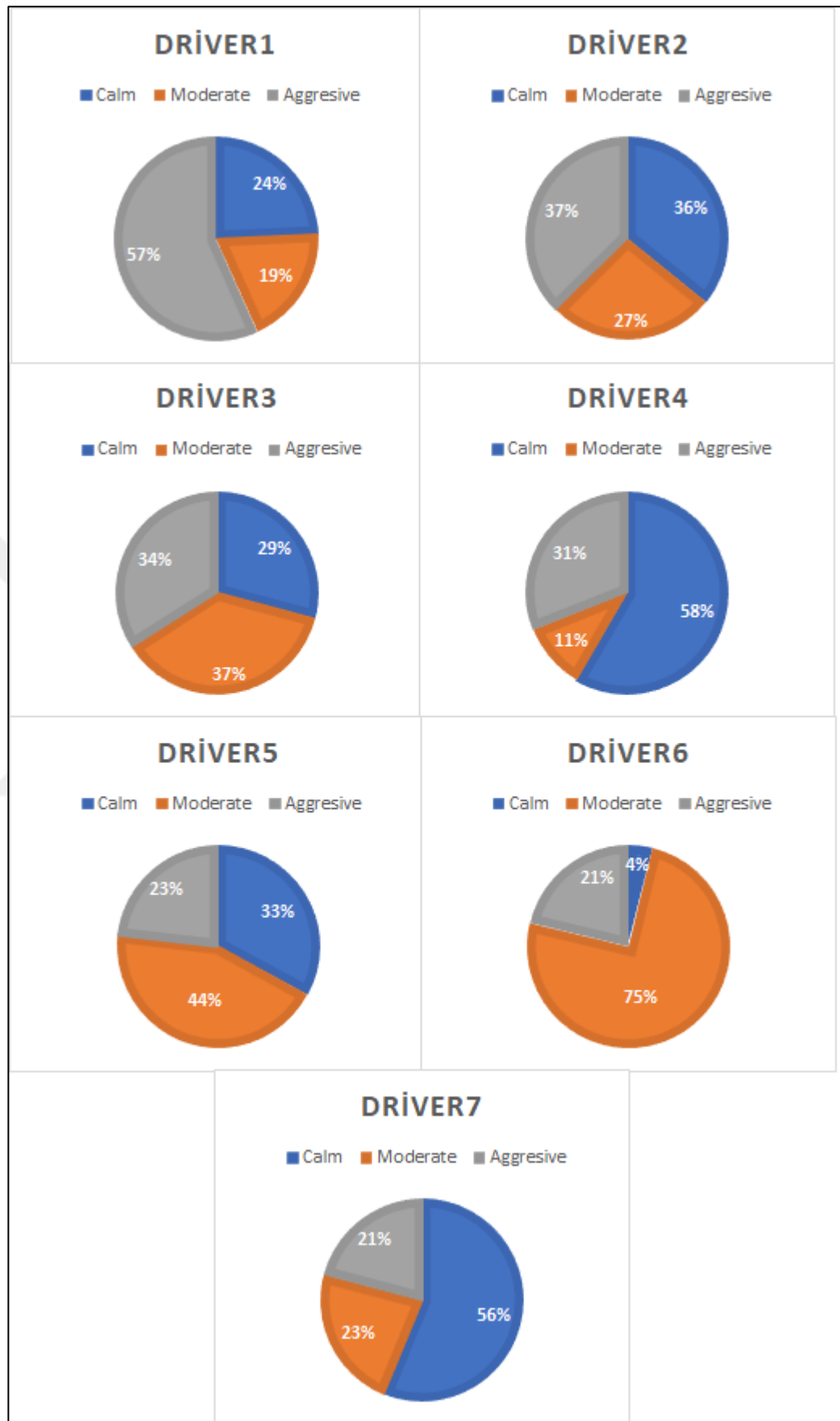


Figure A1.7: Cluster distributions in Clustering Experiment 7.

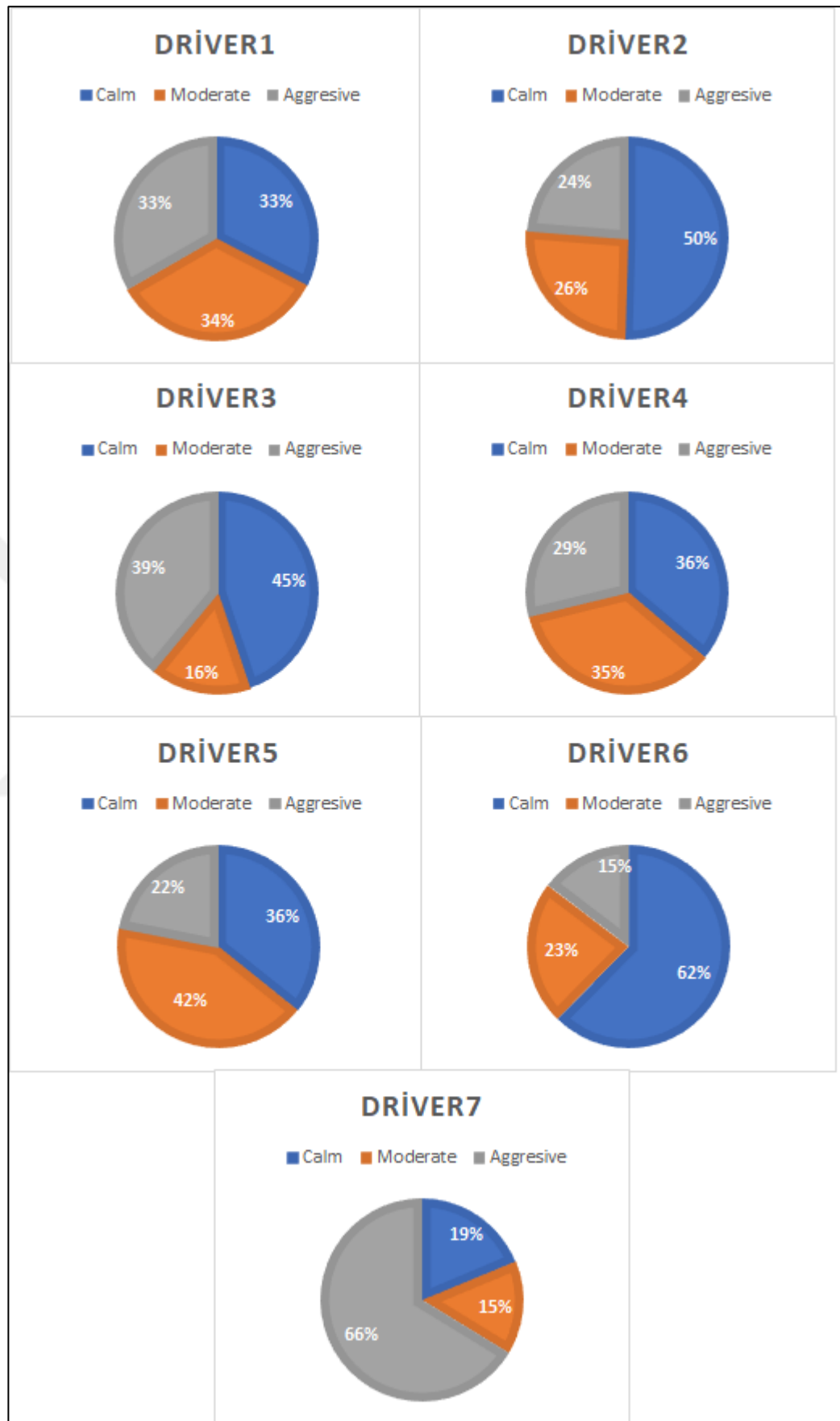


Figure A1.8: Cluster distributions in Clustering Experiment 8.