

# Federated Ensemble-Learning for Transport Mode Detection in Vehicular Edge Network

by

MD. Mustakin Alam  
19301105

Tanjim Ahmed  
22241192

Meraz Hossain  
19301152

Mehedi Hasan Emo  
19301245

Md. Kausar Islam Bidhan  
19301156

A thesis submitted to the Department of Computer Science and Engineering  
in partial fulfillment of the requirements for the degree of  
B.Sc. in Computer Science and Engineering

Department of Computer Science and Engineering  
School of Data and Sciences  
Brac University  
January 2023


© 2023. Brac University  
All rights reserved.

# Declaration

It is hereby declared that


1. The thesis submitted is my/our own original work while completing degree at Brac University.
2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. We have acknowledged all main sources of help.

**Student's Full Name & Signature:**



---

MD. Mustakin Alam  
19301105



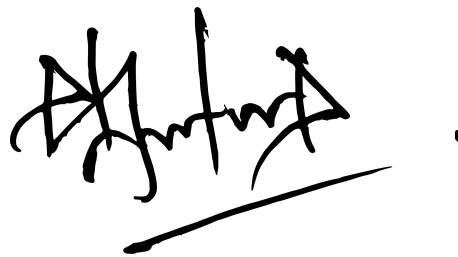
---

Tanjim Ahmed  
22241192



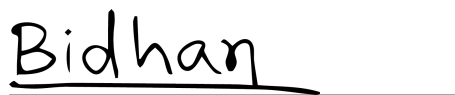
---

Meraz Hossain  
19301152



---

Mehedi Hasan Emo  
19301245



---

Md. Kausar Islam Bidhan  
19301156

# Approval

The thesis/project titled “Federated Ensemble-Learning for Transport Mode Detection in Vehicular Edge Network” submitted by

1. MD. Mustakin Alam (19301105)
2. Tanjim Ahmed (22241192)
3. Meraz Hossain (19301152)
4. Mehedi Hasan Emo (19301245)
5. Md. Kausar Islam Bidhan (19301156)

Of Fall, 2022 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science and Engineering on January 17, 2023.

## Examining Committee:

Supervisor:  
(Member)



---

Dr. Md. Golam Rabiul Alam  
Professor  
Department of Computer Science and Engineering  
School of Data and Sciences  
Brac University

Co-Supervisor:  
(Member)



---

Mr. Md. Tanzim Reza  
Lecturer  
Department of Computer Science and Engineering  
School of Data and Sciences  
Brac University

Program Coordinator:  
(Member)

---

Dr. Md. Golam Rabiul Alam  
Professor  
Department of Computer Science and Engineering  
School of Data and Sciences  
Brac University

Head of Department:  
(Chairperson)

---

Dr. Sadia Hamid Kazi  
Chairperson and Associate Professor  
Department of Computer Science and Engineering  
School of Data and Sciences  
Brac University

## **Ethics Statement**

This research paper is done by us and it is plagiarism free.

# Abstract

Transport Mode detection has become a crucial part of Intelligent Transportation Systems (ITS) and Traffic Management Systems due to the recent advancements in Artificial Intelligent (AI) and the Internet of Things (IoT). Accurately predicting a person's mode of transportation was challenging for many years until the computational power of smartphones and smartwatches expanded dramatically over time. This is a result of the numerous sensors built within smart devices, which enable the worldwide cloud server to acquire sensory data and anticipate a person's method of transport using multiple machine learning models. Currently, all smart devices and vehicular edge devices are interconnected by Vehicular Edge Networks (VEN). However, as the data are shared globally, the security of an individual's data is questioned, and hence a significant portion of the population is still unwilling to share their sensory data with the global cloud server. Also, the processing time for the massive amount of sensory data should be considered. In this paper, we present a distributed method, Federated Ensemble-Learning in VEN, in which a vast amount of data is used to train the model while the training data is kept decentralized. Federated Ensemble-Learning (FedEL), a hybrid approach, is proposed to enhance the performance of federated strategies. In addition, a majority voting ensembling method has been developed as part of the federated strategy to determine the mode of transportation of local customers. Two machine learning algorithms, XGBoost and Random Forest, and one deep learning technique Multi-Layer Perceptron (MLP) are trained with data from each local client. A prediction is then maintained based on a majority vote among the three models. The class with the most votes is taken into account, while the others are discarded. The FedEL technique has been shown to be highly effective on the TMD dataset, with an accuracy of 94-95% for the 5-second window dataset and 98-99% for the half-second window dataset, based on extensive testing.

**Keywords:** Transport Mode Detection; Artificial Intelligence; Internet of Things; Intelligent Transportation System; Vehicular Edge Network; Deep learning; Federated Learning; Federated Ensemble-Learning; Decentralized; Majority Voting; XGBoost; Random Forest; Multi-Layer Perceptron

## **Dedication**

This research is dedicated to all the people who lose their valuable time and miss important stuff due to the heavy traffic congestion.

## **Acknowledgement**

First and foremost, glory be to the Great Allah, with whose help we were able to finish writing our thesis without too many obstacles. Second, we appreciate the guidance and feedback provided by our co-supervisor Mr. Tanzim Reza sir and respected Supervisor Dr. Md. Golam Rabiul Alam sir. We also had the support of our parents, friends, and teachers whenever we needed it.



# Table of Contents

Declaration	i
Approval	ii
Ethics Statement	iv
Abstract	v
Dedication	vi
Acknowledgment	vii
Table of Contents	viii
List of Figures	x
List of Tables	xi
Nomenclature	xii
<b>1 Introduction</b>	<b>1</b>
1.1 Overview . . . . .	1
1.2 Problem Statement . . . . .	3
1.3 Research Contribution . . . . .	5
<b>2 Literature Review</b>	<b>6</b>
2.1 Intelligent Transportation System . . . . .	6
2.2 Vehicular Edge Network . . . . .	7
2.3 Road Side Unit . . . . .	8
2.4 Mobile Edge Computing . . . . .	9
2.5 Federated Learning . . . . .	11
2.6 Ensemble Learning . . . . .	13
<b>3 Dataset, Data Analysis, and Data Pre-processing</b>	<b>15</b>
3.1 Description of the Data . . . . .	15
3.2 Data Analysis . . . . .	16
3.3 Data Pre-processing . . . . .	18

<b>4</b>	<b>Methodology, Architecture, and Model Specification</b>	<b>22</b>
4.1	System Architecture- A Vehicular Edge Network . . . . .	22
4.2	Overview of the Proposed System . . . . .	25
4.3	Experimental Setup . . . . .	26
4.4	Model Specification . . . . .	26
4.4.1	Random Forest . . . . .	26
4.4.2	XGBoost . . . . .	27
4.4.3	Multi-Layer Perceptron . . . . .	29
4.4.4	Federated Learning . . . . .	31
4.4.5	FederatedMLP . . . . .	32
4.4.6	Federated Ensemble-Learning . . . . .	33
<b>5</b>	<b>Result Analysis</b>	<b>36</b>
5.1	Performance Evaluation Metrics . . . . .	36
5.2	Experimental Result Analysis . . . . .	38
<b>6</b>	<b>Conclusion</b>	<b>45</b>
6.1	Challenges . . . . .	46
6.2	Limitations . . . . .	46
6.3	Future Work . . . . .	46
	<b>Bibliography</b>	<b>47</b>

# List of Figures

2.1	Visual Representation of a Vehicular Edge Network . . . . .	8
2.2	Road Side Unit . . . . .	9
2.3	Mobile Edge Computing Servers . . . . .	10
2.4	Distributed Concept of Federated Learning . . . . .	12
2.5	Ensemble Learning . . . . .	14
3.1	User-Transportation mode relation for half-second window dataset . .	16
3.2	User-Transportation mode relation for 5-second window data-set . . .	17
3.3	Correlation Heat-map Before Updating for Both Datasets . . . . .	19
3.4	Updated Correlation Heat-map for 5-second window dataset . . . . .	20
3.5	Updated Correlation Heat-map for half-second window dataset . . . . .	21
4.1	Local Server Data Collection and Computing . . . . .	23
4.2	Global Server Architecture of FedEL enabled VEN . . . . .	24
4.3	Overview of the Proposed System . . . . .	25
4.4	Random Forest Architecture . . . . .	27
4.5	XGBoost Tree for 5-second window . . . . .	28
4.6	XGBoost Tree for half-second window . . . . .	28
4.7	Multi-Layer Perceptron Neural Network Architecture . . . . .	30
4.8	Proposed Federated Ensemble-Learning Architecture . . . . .	34
5.1	MLP Accuracy and Loss Curve for 5-second window dataset (20 Features) . . . . .	39
5.2	MLP Accuracy and Loss Curve for half-second window dataset (24 Features) . . . . .	40
5.3	Comparison Curves for FedMLP and FedEL on Both Datasets . . . . .	41
5.4	Confusion Matrix for Federated MLP . . . . .	43
5.5	Confusion Matrix for Federated Ensemble-Learning . . . . .	43
5.6	Accuracy on Different Models (Rounded Values) . . . . .	44

# List of Tables

3.1	Value counts for different modes in the half-second window data set . . .	17
3.2	Value counts for different modes in the 5-second window data set . . .	17
3.3	List of Sensors . . . . .	18
3.4	Categorical Encoded Values for Transport Modes . . . . .	18
4.1	MLP Model Summary for 5-second window dataset . . . . .	30
4.2	MLP Model Summary for half-second window dataset . . . . .	31
5.1	Evaluation Metrics Result for Traditional Machine Learning Models for 5-second window dataset (20 Features) . . . . .	38
5.2	Evaluation Metrics Result for Traditional Machine Learning Models for half-second window dataset (24 Features) . . . . .	38
5.3	Evaluation Metrics Result for Multi-Layer Perceptron . . . . .	39
5.4	Evaluation Metrics Result of Federated Approaches for 5-second win- dow dataset (20 Features) . . . . .	40
5.5	Evaluation Metrics Result of Federated Approaches for half-second window dataset (24 Features) . . . . .	41
5.6	Accuracy Scores on Different Transportation Modes of Communica- tion Round Number 200 for Federated MLP Approach . . . . .	42
5.7	Accuracy Scores on Different Transportation Modes of Communica- tion Round Number 200 for Proposed Federated Ensemble-Learning Technique . . . . .	42

# Nomenclature

The next list describes several symbols & abbreviation that will be later used within the body of the document

*AI* Artificial Intelligence

*FedAvg* Federated Averaging

*FedEL* Federated Ensemble-Learning

*FedMLP* Federated MLP

*FL* Federated Learning

*IoT* Internet of Things

*IoV* Internet of Vehicles

*ITS* Intelligent Transportation System

*ML* Machine Learning

*MLP* Multi-Layer Perceptron

*RF* Random Forest

*RSU* Road Side Unit

*TMD* Transport Mode Detection

*VEN* Vehicular Edge Network

*XGB* XGBoost

# Chapter 1

## Introduction

### 1.1 Overview

In recent years, road safety has become one of the biggest concerns of human civilization. According to a report by the Dhaka Tribune, the year 2021 saw more traffic accidents than in 2020, with 6284 people killed and 7468 wounded over the same time period in Bangladesh [1]. Many promising students have lost their lives on the roads of Bangladesh in the past few years due to road accidents. Besides, immense traffic congestion has also affected people around the world. Daily commutes have become nightmares due to traffic congestion. There is traffic congestion in all of the world's mega-cities at various points during the day. In contrast, pandemonium and inefficiency dominate the traffic in Dhaka. According to a survey by the World Bank, Over the past ten years, the average traffic speed in Dhaka has dropped from 21 km/h to 7 km/h, and it may plummet below walking pace by 2035. Another study by the BRAC Institute of Government and Development estimates that Dhaka's heavy traffic costs the economy USD 11.4 billion annually and takes up almost 5 million working hours each day. To calculate the amount of money lost due to traffic, the cost of lost time is added to the cost of operating vehicles for the additional hours [2]. The effective management of this problem of growing traffic on roads is posed by recent technological advancements and rising urbanization [3][4]. Several stakeholders, including governmental organizations and vehicle manufacturers, are making substantial efforts to address this issue by implementing numerous Intelligent Transportation Systems (ITS). Over the years, a variety of Intelligent Transportation Systems have been used, which has decreased the number of incidents involving traffic [5][6][7]. The ITS systems act as a network between Road Side Units (RSU) and the Internet of Vehicles (IoV). The network is known as the Vehicular Edge Network (VEN). Inside the VEN, modern ITS systems work by connecting internal and exterior sensors, such as density sensors, traffic cameras, and speed sensors, in real time. To process the data from the sensors for real-time traffic control, several ITS have installed cloud servers [8].

Inside the ITS, transport mode detection has multiple applications. It can help in managing traffic and finding optimal paths. Moreover, traditional transportation studies involve thorough surveys of commuters and modes of transportation. However, these are only fixed for a certain time and do not account for new environments. Thus, previous studies could have been more efficient [9][10]. Therefore, smartphone-based transport mode detection can be a scalable solution to the studies of transportation that can also be employed in dynamic settings. In addition, the ability to identify a person's mode of transportation can help reduce a vehicle's carbon emissions. Individuals can choose their mode of transportation by examining the carbon emissions of a certain vehicle on their smartphones or other smart devices [11]. Also, the detection of a person's mode of transportation can aid in the monitoring of their health using smartphone applications. These applications use transport mode identification to classify an individual's movement into a number of categories and track the calories burned based on their mode of transportation.

This study proposes a FedEL-based transport mode detection system to handle traffic concerns and road safety. The system is an ITS system in VEN. All the IoV, smartphones of passengers, and other smart road components, such as traffic signals and smart bus stops, will have an integrated mobile edge device in this system. These devices, including smartphones or smartwatches of pedestrians, will function as end nodes and will be linked via the vehicular edge network. This integration will help to improve the transport mode detection of a person and, as a result, will lead to less traffic congestion since all the edge devices will have knowledge about the other devices in the system.

Moreover, if there is any signal or traffic congestion when accelerating the car, the speed of the vehicle will gradually reduce with the received information from the vehicles upfront from the Vehicular Network in the edge device, utilizing the fuel of the vehicle efficiently. Furthermore, the majority of residents in a city rely on public transportation. People waiting for buses at bus stops will be able to see the current position and anticipated arrival time of buses on their route.

Additionally, emergency vehicles such as ambulances, fire trucks, and civil defense vehicles would be prioritized on the roads. Other vehicles will be notified if an emergency vehicle is coming and will be able to make room for the emergency vehicle using this information. Furthermore, there is concern about privacy because so many devices are linked. User device privacy could be protected by the federated learning approach.

## 1.2 Problem Statement

Revolutionary advances in computing and communication technology, as well as the expanding use of AI and the IoT, have all contributed to significant improvements in current ITS. This has resulted in vehicles becoming part of the Internet of Vehicles.

Additionally, as a large number of vehicles and other edge devices are linked through a vehicular edge network, it is difficult for centralized cloud servers to handle and evaluate transportation data [12]. Vehicle-to-vehicle communication, mobile edge computing, and cloud server-to-vehicle sharing are all part of the IoV and modern ITS. Terminals for data transmission initiate the whole trio of data exchange relating to moving vehicles. When compared to the limited network capacity on the Internet of Vehicles, data interchange becomes a problem due to the growing number of motor vehicles and the rising requirement for individualized consumer products today. Data processing at the network's edge is a challenge for the system. An additional essential purpose is an analysis of sensory input at resource-constrained edge devices using hybrid machine learning approaches in order to make educated decisions with enhanced prediction accuracy. Due to the vast number of connected devices, there are still concerns surrounding data privacy when utilizing conventional machine learning methods. Traditional cloud-based machine learning techniques need centralized data storage either at a data center or on a cloud server. However, this creates significant problems, such as unacceptable latency and inefficient communication [13].

As opposed to that, vehicle-related data sharing is often costly in terms of delay due to the difficulty of predicting vehicular mobility. In contrast, effective network routing often requires a full view of the IoV system as a whole. For the IoV system as a whole, the design of an intelligent network routing architecture with collaborative data exchange becomes crucial [14].

As a decentralized machine learning approach, federated learning is utilized to protect the privacy and reduce latency. It may use the processing capacity of several learning agents to accelerate learning and provide data owners with improved privacy protection [15].

According to [16], communications between local clients and the central server provide federated learning, which may also be used to improve the effectiveness of wireless systems. During the training process, updated model parameter sets are still being exchanged between the central server and local clients, using up a lot of communication resources, particularly as soon as the fragmented local clients are a large number of wirelessly connected devices, as in Internet-of-Things (IoT) applications. Compressing the data to be transmitted and maximizing the use of limited communication resources to address the issue. For federated learning, several data compression and communication resource management approach with wide applica-



bility may be used. In addition, both prunings of neural networks and the pruning of parameters in neural networks reduce the training communication cost.

Moreover, whereas federated learning conducts training on edge devices which leads to lower accuracy of the system most of the time, test data operations are carried out on a centralized cloud server. In this study, the central cloud server is the Vehicular Edge Cloud, and everything is connected via the Vehicular Edge Network. Therefore, the question that this research tries to answer is:

How can Federated Learning reduce latency and protect privacy in wireless mobile edge computing with the integration of the Federated Ensemble-Learning approach at the edge devices to classify the edge data with higher accuracy in Vehicular Edge networks as a whole Intelligent Transportation System?

This study will investigate a FedEL model to answer the above question.

## 1.3 Research Contribution

In this research, we develop and analyze a distributed learning technique that includes the idea of the Ensemble-Learning. In particular, we develop a novel Federated approach-based Ensemble Learning technique to improve the performance of the Federated Learning technique, in order to detect the transportation mode. The ensemble of a neural network and two other conventional ML methods is implemented which are MLP, Random Forest, and XGBoost. More specifically, the main contributions of this study are summarized as follows:

- We introduce a novel Federated approach based Ensemble Learning technique which is named Federated Ensemble-Learning (FedEL) technique in order to minimize the errors of the Federated approaches in Vehicular Edge Networks as a whole Intelligent Transportation System.
- We design an ensemble learning method based on majority voting collaborating MLP, Random Forest, and XGBoost inside the federated approach where the RF and XGB trees are trained locally and the exact trained local trees are passed to the corresponding global models for prediction. But the MLP neural network gets updated with weight computation based on the FedAvg technique.
- We develop a custom Multi-Layer Perceptron-based Federated Learning approach inside the Federated approach in order to compare and evaluate the proposed FedEL technique.
- We perform extensive experiments on the well-known TMD dataset implementing multiple centralized and decentralized approaches including the proposed FedEL technique which has the best performance.

# Chapter 2

## Literature Review

Many characteristics have been presented to us as a result of technological growth. Different types of technology may now be found in car systems. Vehicles now contain highly specialized onboard equipment as well as improved storage and calculation capabilities thanks to integrated mobile devices. The Federated Learning approach can be easily implemented on vehicles using VEN due to the reasons. Numerous pieces of research have been studied related to our proposed system and model. The study is elaborated briefly.

### 2.1 Intelligent Transportation System

Combining edge computing with digital twinning-enabled Internet of Vehicles has the potential to implement computing-intensive applications and enhance intelligent transportation capabilities. By updating digital twins of vehicles and offloading services to edge computing devices, inadequate computational resources in vehicles can be compensated for. Due to the computational intensity of Digital Twinning-enabled IoV, Edge Computing Devices would become overloaded with excessive service requests, thereby degrading service quality. The problem is addressed by analyzing a multi-user offloading system in which the Quality of Service is reflected by the response time of services. Then, a service offloading method with deep reinforcement learning is proposed for Digital Twinning-enabled IoV in edge computing; it is denoted by the acronym Service Offloading. Service Offloading uses a deep Q-network, which combines the value function approximation of deep learning and reinforcement learning, to obtain optimal offloading decisions. Eventually, comparative method experiments demonstrate that Service Offloading is effective and adaptable in a variety of environments [17].

Authors in [18] look into how edge computing technology can be used to jointly optimize the information and physical fusion-based intelligent transportation systems. The monitoring points in this article are set up at various traffic crossings, and data is gathered using both long-term memory networks and short-term memory networks at each traffic intersection. The experimental results demonstrate that edge computing technology can assist in the processing of traffic circumstances in the information- and physics-integrated intelligent transportation system, considerably increasing each system's overall productivity.

[19] built a framework for Maritime Transportation System security and privacy compliance while minimizing latency and power consumption while evaluating real-time data at the networks' edges. By evaluating the longevity, belief, and trustworthiness of each transaction, the inclusion of blockchain technology and smart contracts into the frameworks helps to validate each block's transactions at edge nodes and reduce various security risks. Additionally, using the framework, various classification models to forecast dangerous vessels using real-time maritime datasets, were offered.

## 2.2 Vehicular Edge Network

Along with personal vehicles such as cars, jeeps, and minibuses, public transportation such as buses plays an important role in the everyday lives of city residents. Even though bus transportation is one of the most crucial aspects of daily life, there is no simple and convenient system in place. A study [20] presented a solution to address this issue by combining a decision tree, a hidden Markov model, and naive bayes with a central server. All vehicles and road parts must be connected to a single network in order to create a unified transportation and traffic system.

The Vehicular Edge Network is another solution to the problem stated above where there is a central Vehicular Edge Cloud server (VEC). In cloud-based vehicular networks, a study [21] presented a hierarchical VEC offloading system. The study examined the task offloading process and came up with a Stackelberg game to model it and devised a distributed method for determining the ideal VEC server strategies, which maximize revenue while maintaining computation task latency limitations. Also, analytical and numerical statistics are used to confirm the revenue increase in the suggested strategy. Also, researchers in [22] suggested a vehicular edge cloud offloading (VECC) architecture for offloading the local computations of intelligent vehicles onto the edge cloud. They used a stochastic fair allocation (SFA) technique to assign randomly the minimum needed resource blocks to allowed vehicular users and to address the advantages of VECC.

Mashael Khayyat et al. proposed a distributed deep learning approach to acquire near-optimal computational offloading decisions using a group of concurrent deep

neural networks for a multilevel vehicular edge-cloud computing network in which an optimization problem is presented to reduce the vehicle’s time and energy consumption [23]. The study [24] investigated the use of machine learning technologies in an intelligent vehicle network. The study has examined the specific problems in vehicular communication, networking, and security, as well as machine learning-based solutions.

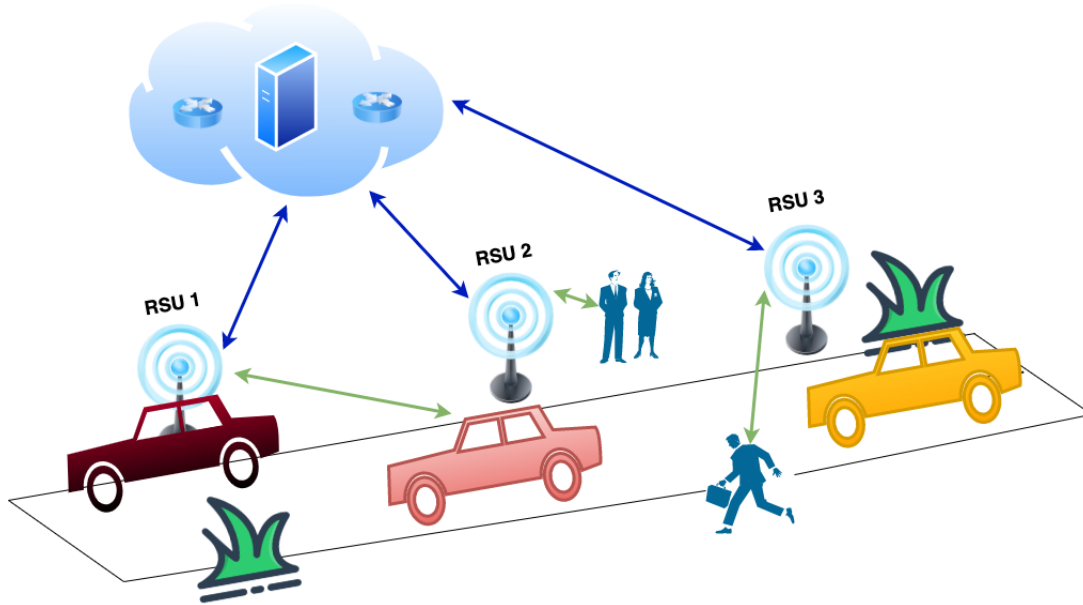


Figure 2.1: Visual Representation of a Vehicular Edge Network

Figure 2.1 shows the visual representation of a vehicular edge network across a road along with Road Side Units and Mobile Edge Servers. In the network, the vehicles and pedestrians are connected.

## 2.3 Road Side Unit

An RSU is a device that is placed alongside a road and is used to gather traffic data from a specific area. This data is then sent to traffic control devices and a central traffic management center. In addition, the RSU can provide information to intelligent vehicles about future traffic conditions. An RSU can gather traffic data using a variety of methods. Triangulation is one technique that uses mobile devices as covert traffic monitors. The mobile phone network receives signals from the phones that an RSU can pick up. Triangulation techniques are used to gather, evaluate, and transform this data into information on traffic flow. As long as there is a cell phone that is turned on inside the vehicle, this technique works for all kinds of vehicles. Vehicle re-identification is a different technique that makes use of distinctive identification from in-vehicle gadgets like Bluetooth MAC addresses or RFID toll tags. Multiple RSUs can identify a specific vehicle as it moves along a route and log a time stamp. To ascertain speed, trip times, and traffic flow for a road

segment, this data is exchanged and evaluated. This technique needs the vehicle’s technology to communicate a special ID. Wireless connection between components is a common feature of contemporary automobiles and can be employed for this reason. Additionally, data on traffic movement can be gathered using the V2I connectivity offered by intelligent cars. GPS or satellite navigation systems, inductive loop detection, traffic video cameras, and audio detection are further techniques for acquiring data on traffic flow. RSUs combine information from various sources to develop a thorough understanding of traffic flow on a particular road stretch. In order to provide a more accurate picture of traffic conditions than any one sensor method could, they do this by utilizing data fusion techniques, which intelligently blend data from many sources [25].

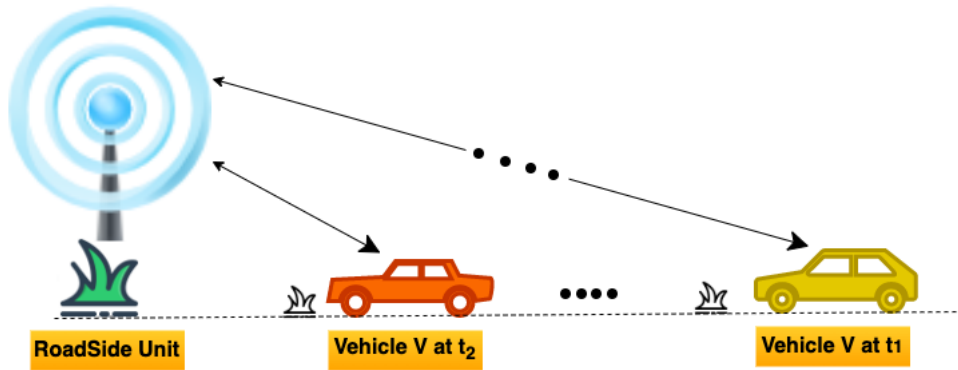


Figure 2.2: Road Side Unit

Figure 2.2 shows the visual representation of a Road Side Unit where it is gathering data from the vehicles. This helps construct a vehicular edge network.

## 2.4 Mobile Edge Computing

Authors in [26] mentioned the issue with the traditional cloud-based machine learning (ML) approach, that it requires the data to be centralized in a cloud server or data center and as a result, serious issues related to unacceptable latency and communication inefficiency occur. Mobile edge computing (MEC) was proposed to bring intelligence closer to the edge. Nonetheless, it still requires personal data to be shared with external parties. All these things raise security and privacy concerns for the users. To address this issue, Federated Learning in mobile edge computing (MEC) was proposed. Federated Learning (FL) can serve as an enabling technology in mobile edge networks since it enables the collaborative training of an ML model and also enables DL for mobile edge network optimization. However, in a large-scale and complex mobile edge network, heterogeneous devices with varying constraints are involved. In their paper, they described the fundamentals of DNN model training, federated learning (FL), and system design for federated Learning (FL). Later they described the challenges related to that which include communication cost,

resource allocation, data privacy, and security. In addition to that, they discussed the implementation of FL for privacy-preserving mobile edge network optimization.

Feng et al. [27] highlighted an issue in the mobile edge computing (MEC) system. Due to the unreliable wireless transmission circumstance and resource constraints in the MEC systems, both the performance and training efficiency of federated learning cannot be guaranteed. To solve it they proposed an optimization design of federated learning (FL) in the mobile edge computing (MEC) system. They introduced the paradigm of federated learning (FL) in mobile edge computing (MEC) devices. Then, they proposed the idea of deploying federated learning (FL) in mobile edge computing (MEC) devices. They provided an iterative algorithm to find an optimal solution. Model sparsification and parameter quantization was used to remove redundant data and get rid of long-digit input. First, a closed-form upper bound of model accuracy loss was derived, which is an efficient metric to evaluate the quality of federated learning. Then, an optimization problem is formulated to improve the model accuracy and training efficiency of federated learning with a limited budget of computation and communication resources. With a balanced dataset, they found a 9.80% performance increase and 7.80% for an unbalanced dataset using their optimization algorithm.

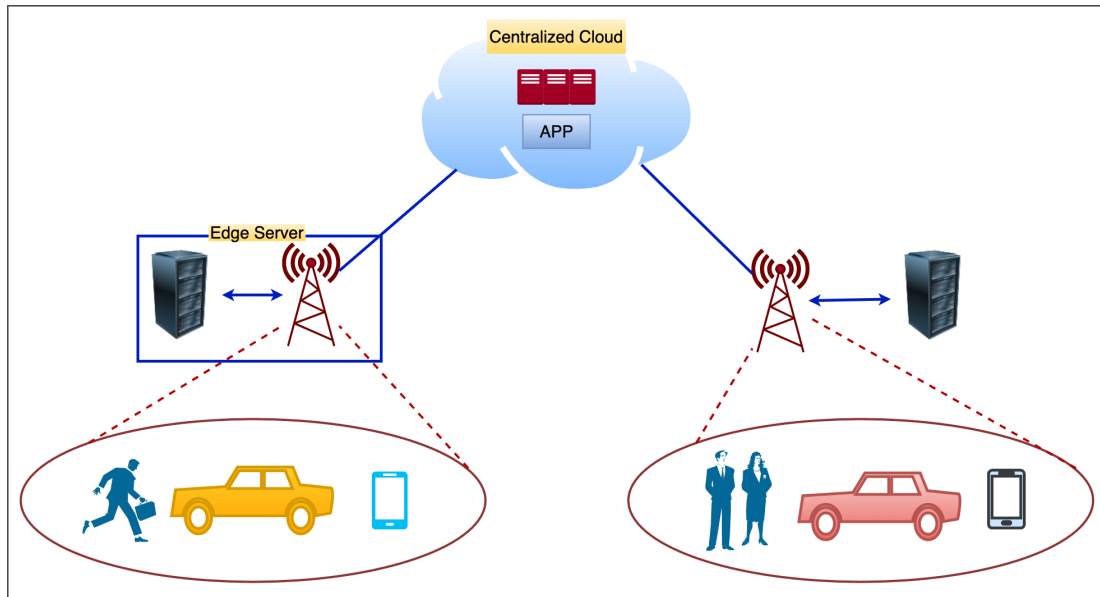


Figure 2.3: Mobile Edge Computing Servers

Figure 2.3 shows the visual representation of the wireless connections of Mobile Edge Computing Servers to edge devices and a centralized global cloud server.

## 2.5 Federated Learning

Federated Learning is a distributed ML approach. It could be used efficiently in wireless communication. Federated learning is facilitated through connections between regional users and the main server, which may also be utilized to improve wireless system performance. Throughout the training process, the central server and local clients continue to exchange updated model parameter sets, consuming significant communication resources, especially when the distributed local clients are a large number of wirelessly connected devices, as is the case with Internet-of-Things (IoT) applications. Compressing the data to be sent and making optimal use of limited connection resources were developed to remedy the problem. Numerous data compression and communication resource management strategies may be employed for federated learning [28]. Jakub et al. [29] demonstrated two methods to minimize uplink communication costs: structured update and sketching update. In structure update, they directly learned an update from a constrained space parameterized with fewer variables. In contrast, in sketched updating, a complete model update is learned and then compressed using a mix of quantization, random rotations, and subsampling before being sent to the server.

Choosing FL over traditional ML gives it an advantage. In contrast to traditional ML strategies based on centralized training on cloud servers, the authors in [30] describe an FL-based framework for distributed training of ML models as an efficient learning strategy for vehicular networks and edge intelligence. Machine learning algorithms have been developed to learn from sensor measurements. The current trend in vehicular network machine learning is to use centralized algorithms, in which a sophisticated learning algorithm is taught on local sensor data from edge devices. The purpose of federated learning (FL) is to bring machine learning down to the edge. The training technique is identical to that of machine learning, with the exception that FL does not require the transfer of the entire dataset. This makes it easier to deal with the ever-growing datasets at the edge devices of the vehicle.

The authors of [31] investigated the topic of clustered FL in vehicular networks. They attempted to bridge the gap between clustering in vehicular networks and clustering in FL by inventing a mobility-aware learning method for clustered FL. In the suggested design, vehicle-to-vehicle communication is considered a benefit for overcoming the communication bottleneck of FL in vehicular networks. Additionally, FL safeguards privacy via decentralized learning.

The thesis in [32] introduced GTR, a low-rank theory-based traffic data recovery system driven by edge computing. First, the authors conducted exploratory experiments using a large-scale Intelligent Transport System traffic dataset. The results showed the major problem of missing traffic data as well as its spatiotemporal connections. They implemented a Local Search-based Suboptimal Edge Node Deployment Algorithm and an FPC-based Accurate Traffic Data Recovery Algorithm inspired by the observation based on low-rank theory.



Yuanshao et al. [33] introduced a system for identifying a person’s mode of transport while preserving their privacy, using Federated Learning (FL). The system is based on an attention-augmented model architecture and utilizes FL to identify the travel mode without accessing raw GPS data. It is shown to be more accurate than previous centralized models, and is also able to handle non-Independent and Identically Distributed (non-IID) GPS data that is commonly found in the real world. In order to maintain accuracy with non-IID data, the system employs a secure data-sharing strategy that adjusts the distribution of local data for each user. The effectiveness of the proposed model is demonstrated through experiments on a real-world dataset.

The field of privacy-preserving Transportation Mode Identification (TMI) is an important area of research in the development of intelligent transportation systems. Crowdsourcing has been proposed as a cost-effective way to train TMI classifiers using federated learning (FL), which enables data sharing without compromising user privacy. However, traditional TMI approaches often require access to labeled data, which may not be available in real-world applications. To address this issue, [34] proposed a semi-supervised FL approach called Mean Teacher Semi-Supervised Federated Learning (MTSSFL). MTSSFL trains a deep neural network using a semi-supervised FL framework, allowing for accurate and private Transport Mode Identification classification using crowdsourced data without the need for large amounts of labeled data. MTSSFL incorporates consistency updating to improve the training of local models that only have access to unlabeled data by incorporating the global model in their gradient updates.

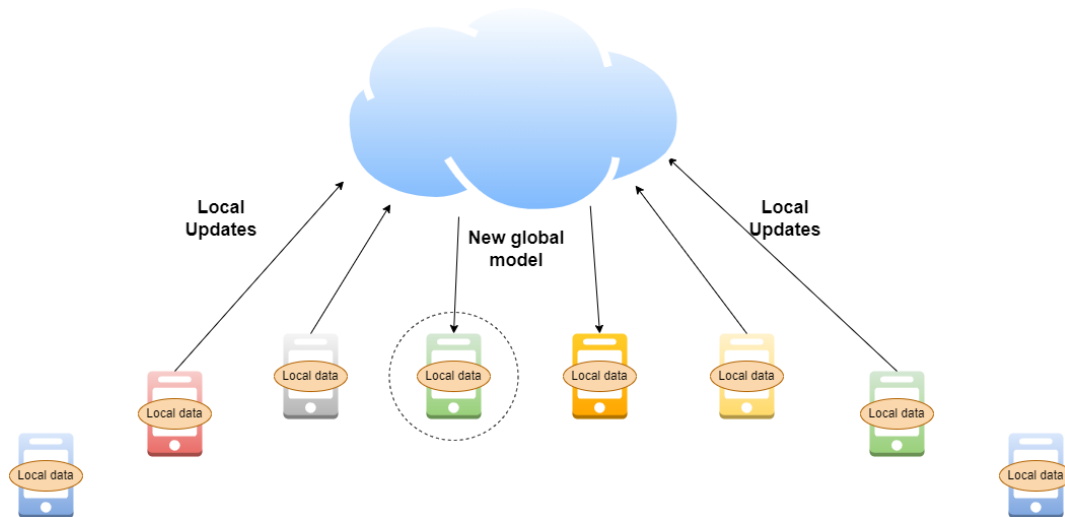


Figure 2.4: Distributed Concept of Federated Learning

Figure 2.4 shows the local and global concept of Federated Learning. Here multiple local devices are connected to a global server where the training of ML occur locally and the final prediction takes place in the global model of the global server.

## 2.6 Ensemble Learning

Researchers in [35] proposed a semi-supervised deep ensemble learning approach for travel mode identification that requires only a small amount of annotated data. This approach takes GPS trajectories of varying lengths as input and utilizes a custom feature engineering process to extract latent information. A new deep neural network architecture was also introduced to map this latent information to the desired travel mode. An ensemble was created by utilizing the annotated data to generate proxy labels for the unannotated data, allowing both types of data to contribute to the learning process. Through extensive case studies, it was found that the proposed approach performed significantly better than the approaches with partially-labeled training data.

Hong et al. [36] proposed a method for identifying modes of transportation (also referred to as locomotion recognition) using data obtained from Android mobile devices. The method, which was submitted as the solution for Team Jellyfish in the Sussex-Huawei Locomotion-Transportation recognition challenge, aimed to develop a body position-independent classifier utilizing data from commonly available sensors on a phone, including the accelerometer, gyroscope, magnetometer, and barometer. The proposed solution was an ensemble of XGBoost and neural network classifiers. By training the models using this method and combining XGBoost and neural network models, a strong performance on the validation data set (considering hand position) was achieved.

Zhibin et al. [37] proposed a hybrid transportation mode inference method based on ensemble learning using only GPS data. To distinguish between modes, a statistical approach to extract global and local features from segmented trajectories was employed, which was then used in the classification stage. To improve performance, tree-based ensemble models (Random Forest, Gradient Boosting Decision Tree, and XGBoost) instead of traditional methods (K-Nearest Neighbor, Decision Tree, and Support Vector Machines) were implemented. The efficacy of the proposed approach was demonstrated through experiments on the GEOLIFE dataset, where the XGBoost model achieved a classification accuracy of 90.77%. To reduce model complexity, a tree-based ensemble method for accurate feature selection was used.

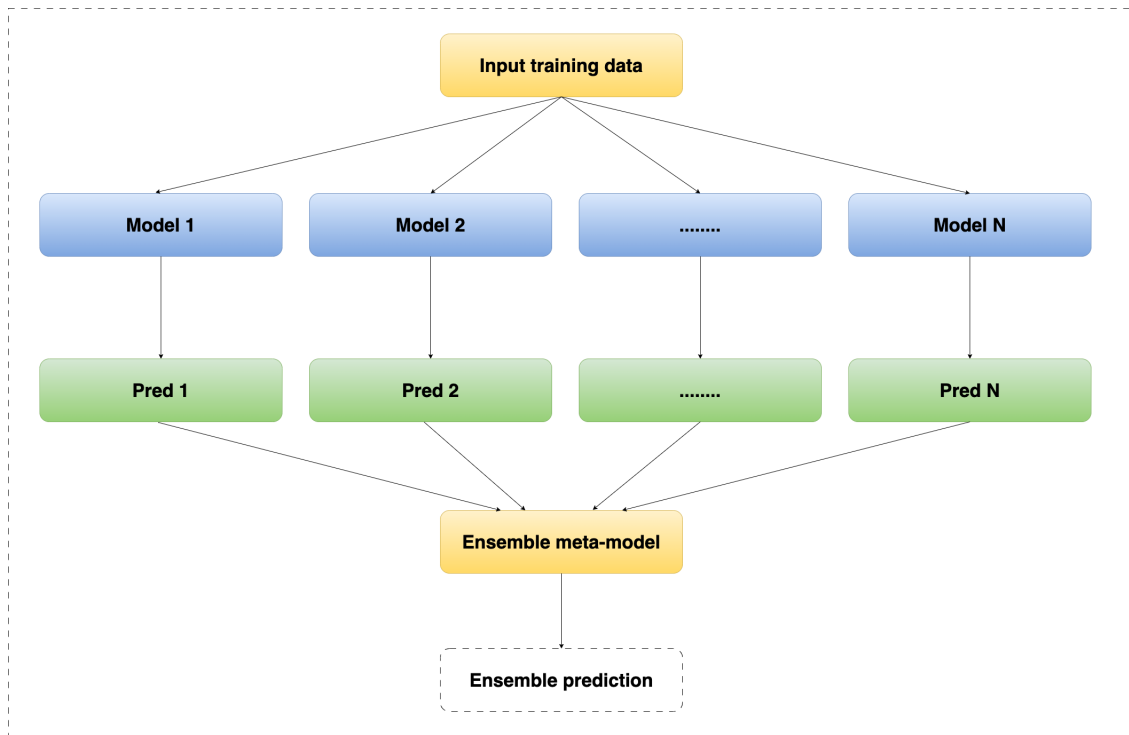


Figure 2.5: Ensemble Learning

Figure 2.5 shows how the ensemble learning method works by incorporating multiple models and makes predictions.

We can see from the above discussion that Ensemble approaches increase the accuracy of models. Integrating it with Federated Learning can improve the performance of the system by improving latency and training the model locally. The proposed FedEL works similarly.

# Chapter 3

## Dataset, Data Analysis, and Data Pre-processing

### 3.1 Description of the Data

The most major and important component of our thesis is the dataset. The dataset we acquired is called Transport Mode Detection (TMD), and it analyzes a user's mode of transportation from observations of the individual or the surrounding region. The objective of TMD is to determine the potential mode of transportation for a user, which might include driving, walking, etc. Applications that are essential for monitoring traffic conditions, providing travel assistance, etc., are enhanced by this information. Thirteen respondents, ten of whom were men and three of whom were women, had smartphones that were used to collect the data from sensors. The data set consists of 226 labeled files that are updated continuously and reflect the same amount of actions, or more than 31 hours' worth of activity. 26 percent of the data were classified as being on foot, 25 percent as driving a car, 24 percent as sitting motionless, 20 percent as being on a train, and the remaining as being on a bus [38].

Data cleaning procedures are carried out, such as removing exclusion measures from the sensors and making the speed and sound sensors' values positive, etc. Numerous sensors, including proximity and ambient (sound, light, and pressure) sensors, produce a single data value that can be used directly in data sets. All of the others, on the other hand, yield several values relating to the coordinate system in use, making their values significantly correlated with orientation. We can utilize magnitude, an orientation-independent measure, for nearly all applications.

After the pre-processing for cleaning the data is done, it is divided into two-time

windows, one for 5 seconds and the other for half a second. After the division, four features are extracted from the sensors which are minimum, maximum, mean, and standard deviation.

Thirteen users, designated U1 to U13 in the datasets broken into half-second and five-second windows, had their smartphones used to gather the data. Five modes of data collection were used. Car, Still, Walking, Bus and Train are among them. The half-second window dataset’s initial dimension is  $62585 \times 70$ . The half-second window dataset also has a dimension of  $5893 \times 70$ . The datasets’ features are values that have been recorded by various kinds of sensors. such as the gyroscope, accelerometer, rotation vector, and orientation, among others. (More details can be collected from [39])

In the ‘target’ feature, many transportation modes are allocated. It is used to train the FedEL model and predict the transportation mode.

## 3.2 Data Analysis

One of the most crucial steps when dealing with any type of data is exploratory data analysis. Since the thesis tends to focus on data, it is essential to thoroughly evaluate the data and prepare the datasets for model fitting.

First, the interaction between the modes and users is examined. Figure 3.1 and Figure 3.2, respectively, display the connection of users with the mode of transportation for a half-second window and a five-second window. They illustrate which users are connected to specific modes. Due to the fact that the data was initially collected and then separated into windows, the same behavior can be seen in both datasets.

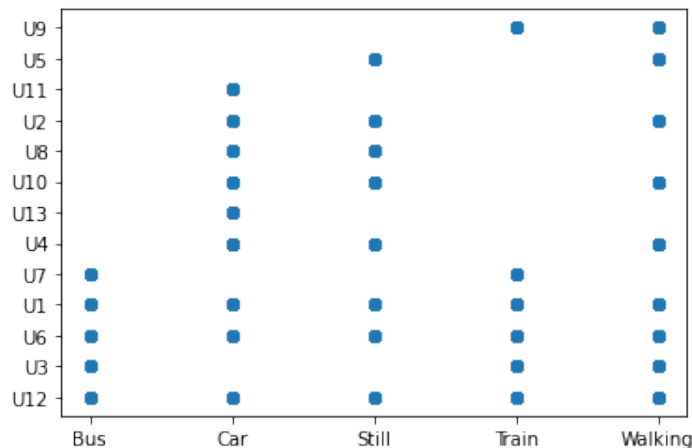


Figure 3.1: User-Transportation mode relation for half-second window dataset

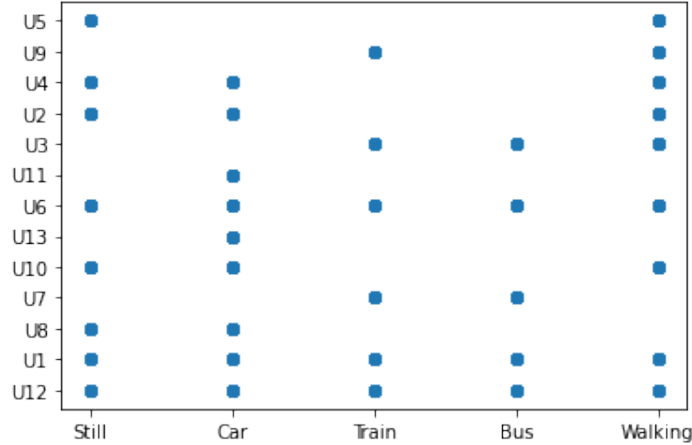


Figure 3.2: User-Transportation mode relation for 5-second window data-set

Second, the value counts for each of the modes across both datasets are then examined. How many data points are there for each of the five modes of transportation for the datasets for the half-second and 5-second windows are shown in Table 3.1 and Table 3.2, respectively. We can see that there are more data points in the half-second window than in the five-second frame. Because the dimensions of the five-second window dataset and the half-second window are drastically different.

Mode	Value Count
Car	12518
Bus	12517
Train	12517
Walking	12517
Still	12516

Table 3.1: Value counts for different modes in the half-second window data set

Mode	Value Count
Car	1180
Still	1179
Train	1179
Bus	1178
Walking	1177

Table 3.2: Value counts for different modes in the 5-second window data set

Furthermore, 70 features in each dataset are recorded. The features are captured data from different kinds of smartphone sensors and from the sensory data, 4 features are extracted as mentioned before which are minimum, maximum, mean, and standard deviation. Data were extracted from 15 sensors. Table 3.3 contains the list of the sensors of the users' android smartphones. Both calibrated and uncalibrated data were captured from the sensors. In the datasets, numerous cells were null. The imbalance state of the datasets is addressed in the data pre-processing part.

Sensors
Activity recognition
Accelerometer
Game Rotation Vector
Gravity
Gyroscope
Light
Linear Acceleration
Magnetic Field
Orientation
Pressure
Proximity
Rotation Vector
Step-counter
Sound
Speed

Table 3.3: List of Sensors

### 3.3 Data Pre-processing

In both of the datasets, there are many cells with missing values, so handling the missing values comes first. Features that have more than 60 percent of their values missing are removed. Furthermore, 0 is assigned to the features that have partial null values. Additionally, the feature, 'id' is removed because it does not carry any significance.

Transport Mode	Encoded Value
Bus	0
Car	1
Still	2
Train	3
Walking	4

Table 3.4: Categorical Encoded Values for Transport Modes

Then, it is seen in the dataset that the maximum values are float64 type except for 4 features. Two of them are int64 type and the rest are object type. Two of the object type features are 'target' and 'user'. The 'user' column is dropped and categorical encoding is applied to the 'target' features. This means numeric values are assigned instead of the object type data which is shown in Table 3.4.

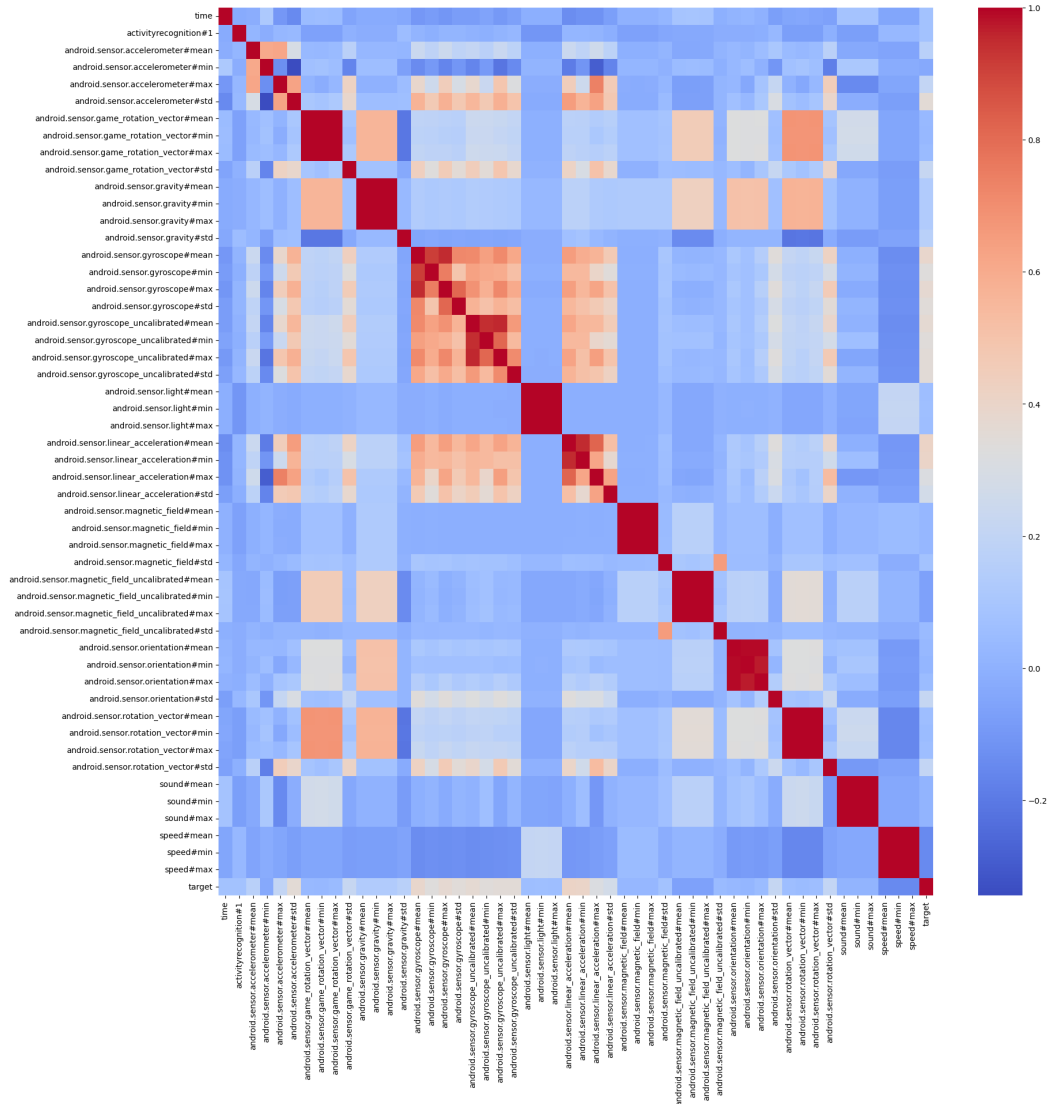


Figure 3.3: Correlation Heat-map Before Updating for Both Datasets

Additionally, the correlation coefficient equation 3.1 demonstrates the correlation between the features which is the Pearson product-moment correlation coefficient [40]. The features, in this case, are  $a$  and  $b$ . Numerous features are shown to be highly connected, which indicates they heavily rely on one another. These features are unable to add new information to the model. As a result, features having a connection of above 70 percent are dropped. Figure 3.3 shows the correlation heat-map for both of the datasets with 70 features before updating. After updating by removing the highly correlated features, only 20 remain for the 5-second window dataset. The 5-second window dataset shrinks to  $5893 \times 20$  in size. Figure 3.4 portrays the correlation heat-map with 20 features and ‘target’ after updating the dataset. Also, after updating by removing the highly correlated features, only 24 remain for the half-second window dataset. The half-second window dataset shrinks to  $62585 \times 24$  in size. Figure 3.5 portrays the correlation heat-map with 24 features and ‘target’ after updating the dataset. We currently have two cleaned datasets with crucial features. Finally, the 5-second window dataset has 20 features and the half-second window dataset has 24 features.



$$c = \frac{\sum (a_i - a') (b_i - b')}{\sqrt{\sum (a_i - a')^2 \sum (b_i - b')^2}} \quad (3.1)$$

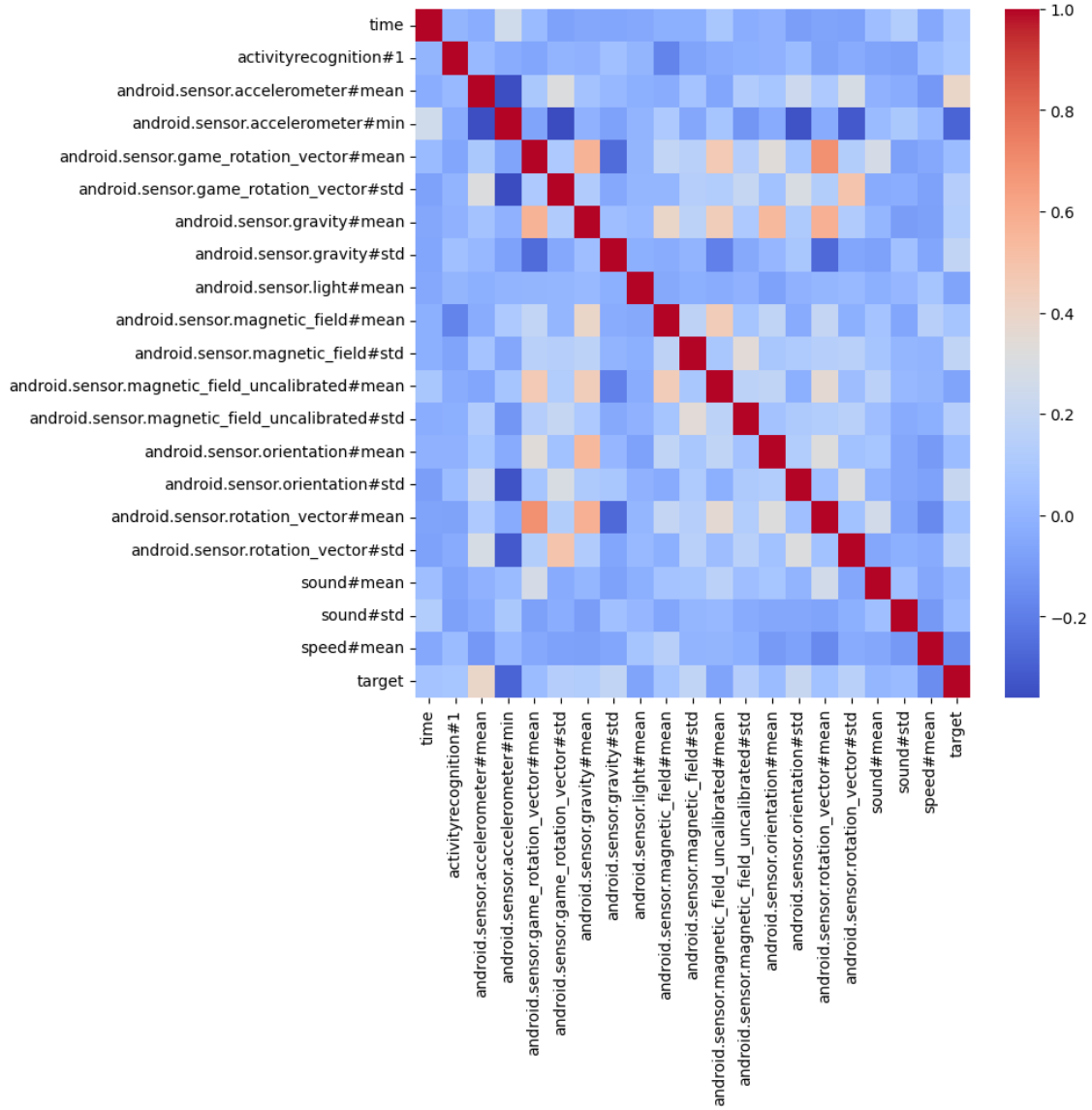


Figure 3.4: Updated Correlation Heat-map for 5-second window dataset

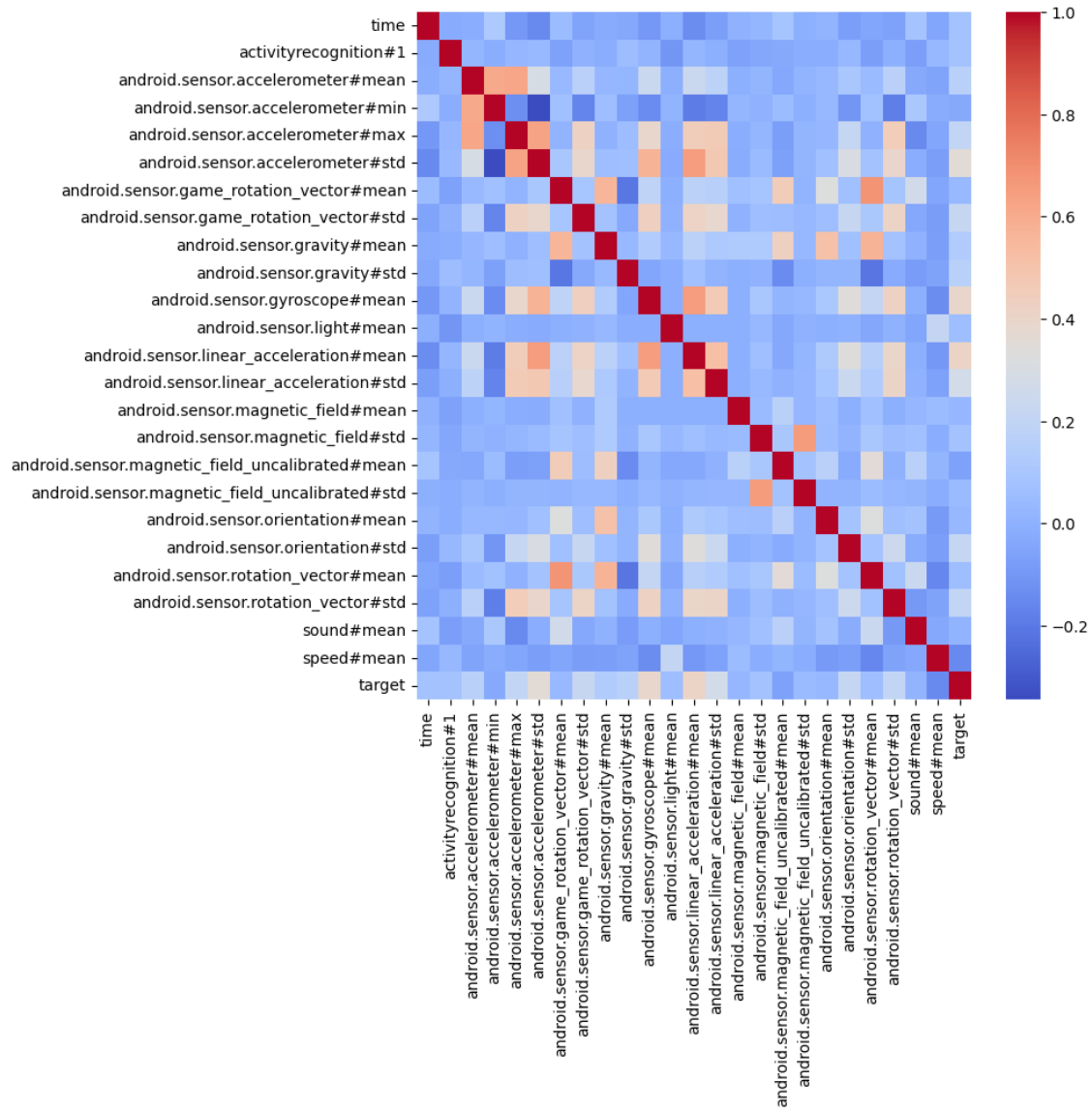


Figure 3.5: Updated Correlation Heat-map for half-second window dataset

In order to anticipate outputs, the features and the label are finally split into  $X$  and  $y$ . Additionally, the Min Max Scaler is utilized to transform features' values between 0 and 1. It addresses the issue of outliers and the models' propensity to select higher values. The Min-Max Scaler equation is shown in 3.2. The Min-Max scaling or normalization is a technique of taking values between a minimum and a maximum value [41].

$$MinMaxScaler = \frac{X - X_{min}}{X_{max} - X_{min}} \quad (3.2)$$

Finally, the label and target data are divided into train and test groups in an 8:2 ratio. 80 percent of data are kept for training and 20 percent of data are kept for testing. Now the data is completely ready to fit in the models.

# Chapter 4

## Methodology, Architecture, and Model Specification

### 4.1 System Architecture- A Vehicular Edge Network

The goal of the Vehicular Edge Network (VEN) architecture is to collect sensory data from smartphones or mobile edge devices through Road Side Units (RSU) via wireless transmission and utilize federated learning (FL) and mobile edge computing (MEC) in the MEC servers, to ensure data security in Internet of Vehicles (IoV) systems and enable intelligent data sharing, to effectively allow the exchange of collaborative data, particularly with regard to the data and VEN topology, as shown in Figure 4.1 and Figure 4.2. The figures show an Intelligent Transportation System based on VEN, powered by Federated Ensemble-Learning. The Figure 4.1 shows the data gathering through RSUs and mobile edge computing and local training in the local MEC servers which are incorporated with the RSUs. Figure 4.2 shows the centralized global cloud server where multiple locals are connected. Specifically, user data is often sensitive and confidential since it may disclose the latitude and private details of a driver. The 3 key levels of the system architecture of the VEN we took into consideration are the Data capturing layer (DCL), the Mobile edge server layer (MESL), and the Vehicular edge cloud layer (VECL). In the system, sensory data are captured from edge-centric devices of vehicles and from the smartphones of passengers and pedestrians using RSU. This data is processed and trained in a decentralized way with the blessing of federated learning in mobile edge servers through edge computing. In this study, the RSU and MEC servers are thought to be incorporated together. At each local point, the RSU is used for capturing data; where a MEC server is present along with RSU where the data processing and the local training are done. Later, the data is sent to the vehicular edge cloud server which is the centralized global cloud server; there the processing occurs in the global

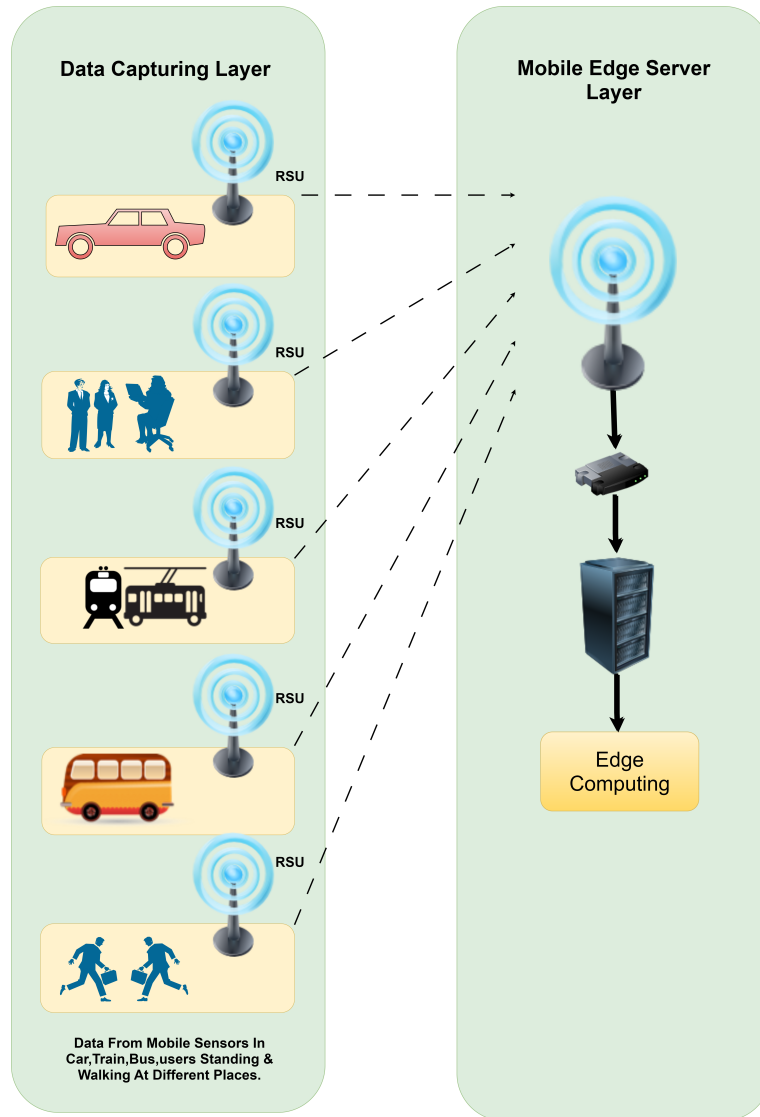


Figure 4.1: Local Server Data Collection and Computing

server, and real-time transport mode prediction is done in this layer. The transport mode of the vehicles that are present on that point is detected with the proposed FedEL method. As the data is not shared with the global server and is trained in a local server instead, it does not get exposed to a centralized area on a large scale. As a result, the privacy of the data is secured by the blessing of a decentralized federated learning approach. After the detection, this information is updated with a web application to get it accessed by a passenger waiting at various locations to avail the public transportation or for checking traffic congestion at each place. Also, this information can assist people in avoiding accidents by avoiding busy roads. The drivers of the vehicles can also access the information through the web app connected to the vehicular edge network. The drivers can view the information on the edge device installed on the vehicles or smartphones and choose the correct path to reach the destination faster by avoiding congested roads. Additionally, drivers can make paths for emergency vehicles based on the information if any emergency situation arrives like an accident or fire situation. Furthermore, transport mode detection can also help in studies on transport, reducing carbon emissions, and fitness tracking.

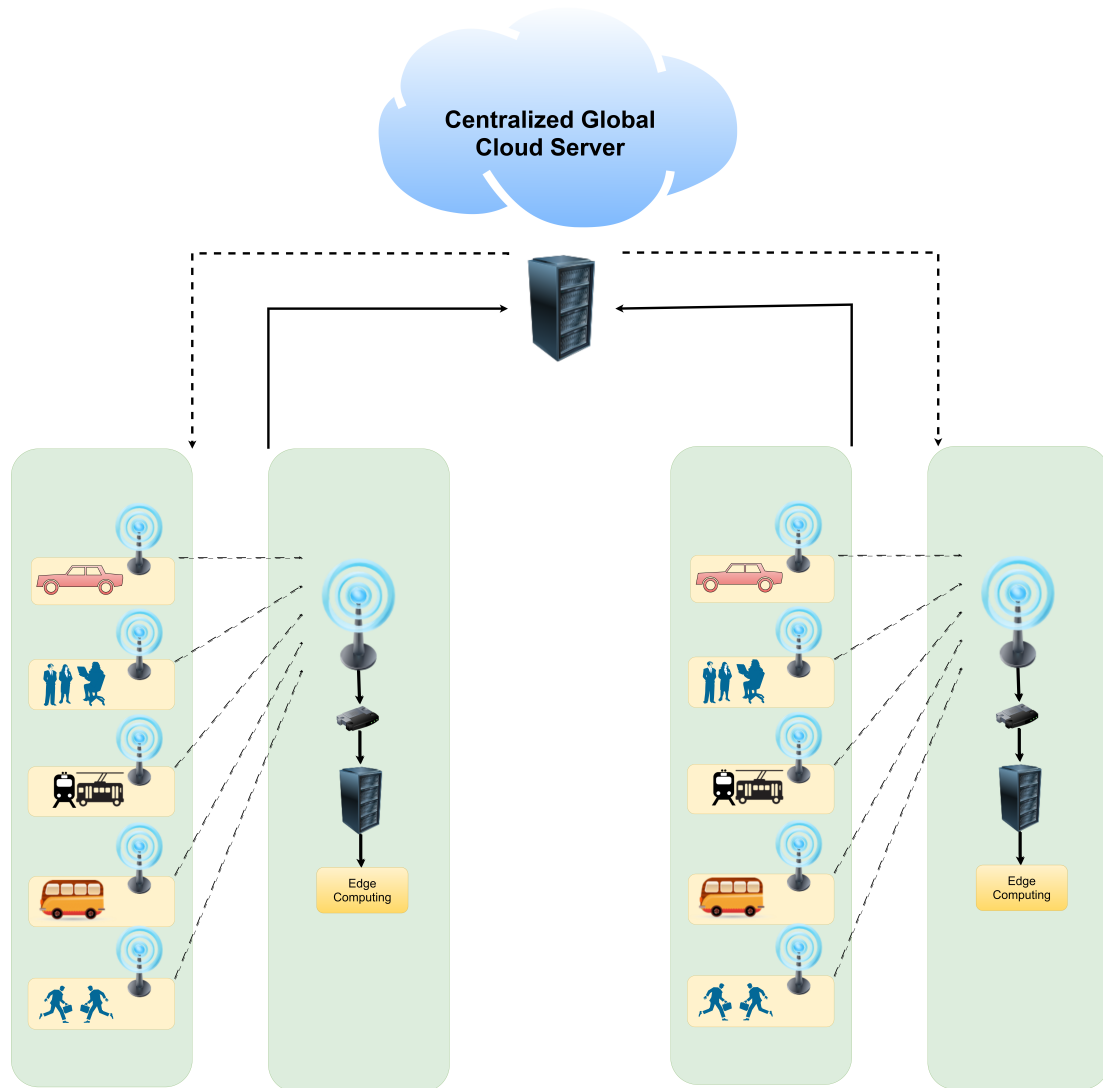


Figure 4.2: Global Server Architecture of FedEL enabled VEN

The combination of all of these makes it a whole ITS inside VEN which is FedEL enabled.

**Data Capturing Layer (DCL):** The DCL consists of collecting data using RSU from the sensors of the edge devices and mobile phones inside buses, cars, trains, or pedestrians standing at different places of a particular point besides an RSU.

**Mobile Edge Server Layer (MESL):** There exists a MEC server at each point within the Road Side Units. The main objective of this layer is to store the collected data in the MEC server where it is processed and edge computations are done before it could be sent to the VECL which is a centralized global cloud server. The collected data is collaborated and passed through pre-processing and the FedEL implementation process which is the local training starts on the collected data in the MEC server.

**Vehicular Edge Cloud Layer (VECL):** This is the final layer of the VEN where the data is sent from the MESL from different RSU points and passes through final processing or computational processes (if needed). It is the centralized global cloud server. Here real-time transport mode prediction is done in this layer through global computations of the FedEL part. After the transportation modes are predicted with FedEL it is shown on a web application to users as mentioned before.

## 4.2 Overview of the Proposed System

The complete overview of the proposed system of our work is portrayed in Figure 4.3.

First, we load the dataset into the system. After, exploratory data analysis is done. Following data analysis, data pre-processing takes place. In the pre-processing part, the data is also scaled. Then, data is split into features and labels. Additionally, the data is divided into a ratio of 8:2 for the train and test sections where 80% of the data is used to train the model and the remaining 20% data is used to test the model.

After getting the pre-processed data, now it is prepared for fitting into the Federated approach. Later, 3 global models are initialized which are RF, XGB, and MLP. Here, the ensemble technique is implemented. Then data get fit in the local models of 3 of the models. Furthermore, for MLP neural network, weights get updated and local RF and XGB trees are passed into their corresponding global models. Then the prediction is done using majority voting. Finally, the results get evaluated with the required metrics.

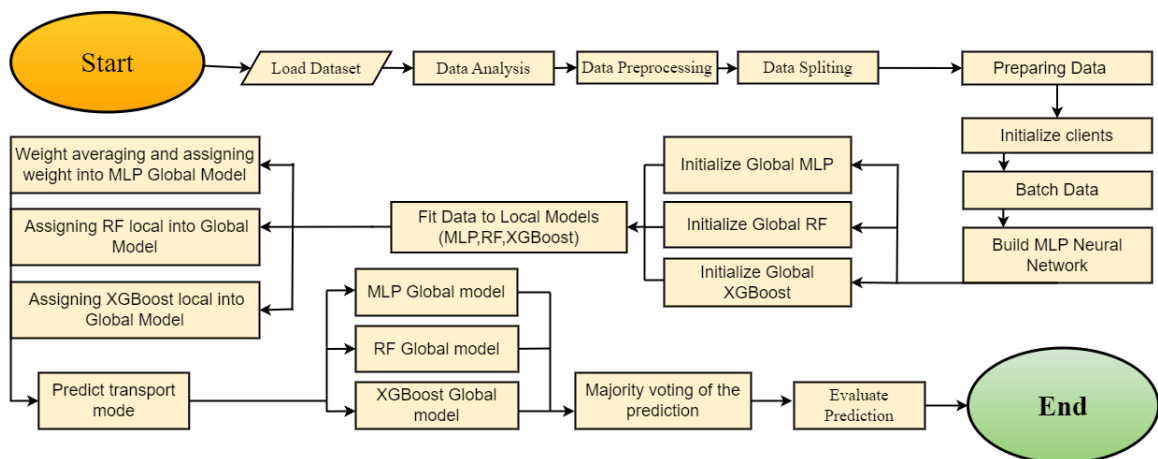


Figure 4.3: Overview of the Proposed System

## 4.3 Experimental Setup

The proposed Federated Ensemble-Learning method and other implementations are developed on the AMD Ryzen 5 3600 system, with 16GB RAM and Nvidia RTX2060 6GB graphics memory, and also with 256GB SSD storage. Every implementation is designed and developed in a python programming language with NumPy, Pandas, sklearn, and TensorFlow libraries, and the results are plotted using the Matplotlib and Seaborn libraries of python.

## 4.4 Model Specification

A series of implementations have been done in order to complete this thesis. Firstly, some usual machine learning techniques have been implemented to observe the reaction of the data. Decision Tree (DT), Random Forest (RF), XGBoost (XGB), Support Vector Machine (SVM), and K-Nearest Neighbour (KNN) have been implemented on the 5-second and half-second window datasets. Later, Multi-Layer Perceptron (MLP) was implemented. This neural network performs quite well on both of the datasets. After, we approached Federated Learning with MLP which is implemented to ensure the privacy of the sensory data. Moreover, to enhance the performance further, an ensemble technique with majority voting inside federated has been proposed which is called the Federated Ensemble-Learning. The ensemble technique is fit in the Federated MLP approach. While ensembling, Random Forest and XGBoost have been selected alongside MLP based on accuracy and cross-validation scores which will be discussed later in the result analysis part.

### 4.4.1 Random Forest

Random Forest is an ensemble learning method that is used for classification and regression tasks in machine learning. It is a collection of decision trees, where each tree is trained on a randomly selected subset of the data and makes predictions based on the majority vote of all the trees in the ensemble. Random Forest is known for its ability to handle large datasets and high-dimensional spaces, as well as its ability to handle missing values and outliers in the data. One of the main benefits of using Random Forest is that it reduces overfitting, which is a common problem in decision tree models. Overall, Random Forest is a powerful and widely-used machine learning algorithm that is known for its robustness and high accuracy. The data fit well in Random Forest and ensembling multiple decision trees, it provides a fruitful solution to the study. Figure 4.4 shows the architecture of Random Forest. For classification problems, Random Forest computes how nodes on a decision tree branch using 4.1 which is the Gini index. Also, the decision is taken based on the entropy which is

derived using 4.2. In the equations,  $G_{idx}$  is the Gini index,  $E_{ntr}$  is the entropy,  $x_a$  signifies the probability of the class 'a', and  $z$  represents the total number of classes.

$$G_{idx} = 1 - \sum_{a=1}^z (x_a)^2 \quad (4.1)$$

$$E_{ntr} = \sum_{a=1}^z -x_a * \log_2(x_a) \quad (4.2)$$

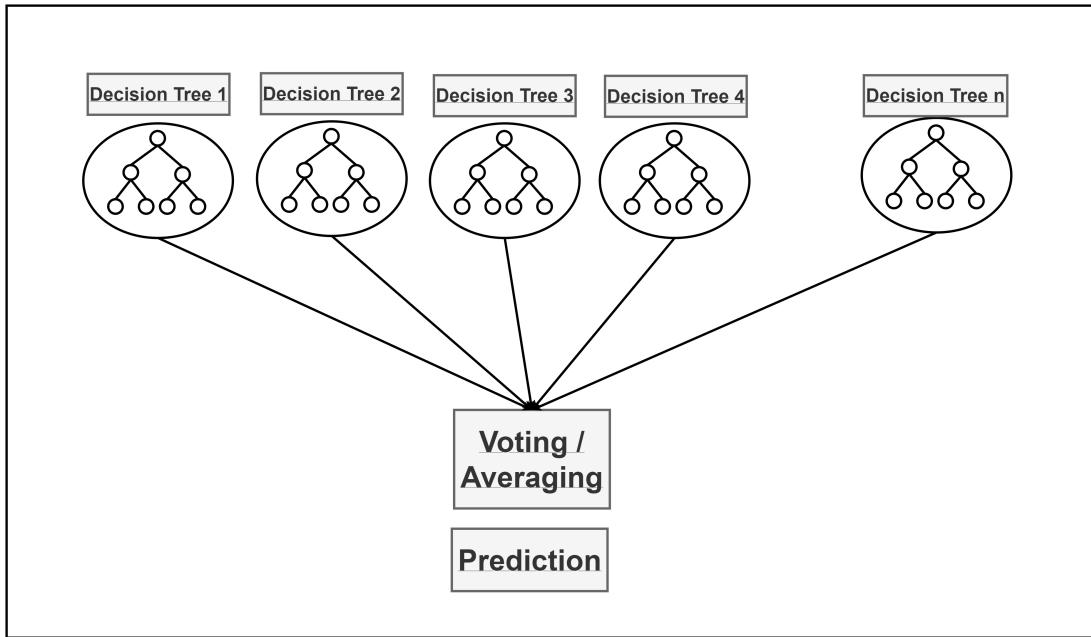


Figure 4.4: Random Forest Architecture

#### 4.4.2 XGBoost

XGBoost (eXtreme Gradient Boosting) is a powerful and popular machine-learning algorithm that is used for classification and regression tasks. It is an implementation of gradient boosting, which is a boosting algorithm that combines multiple weak learners to create a strong learner. XGBoost is known for its high speed and accuracy, as well as its ability to handle large and complex datasets. It also has several advanced features, such as regularization, parallel processing, and handling missing values, which make it a popular choice for many machine learning practitioners. Overall, XGBoost is a powerful and widely-used machine learning algorithm that is known for its high performance and flexibility. The boosting method works optimally on the data. Figure 4.5 shows the XGBoost architectural tree for the 5-second window dataset and Figure 4.6 shows the XGBoost architectural tree for the half-second window dataset. The XGB works using 4.3 and 4.4. The  $M_0$  is the initially defined model to predict the target  $t$ . The model is associated with a surplus  $t - M_0$ . The  $f_1$  is a new model which is fit to the previous surplus,  $t - M_0$ .



Finally,  $M_0$  and  $f_1$  get combined to pass the boosted version of  $M_0$  to  $M_1$ . In  $M_1$  the error decreases significantly. This is how the computations take place till  $n^{th}$  iteration using 4.4.

$$M_1(x) = M_0(x) + f_1(x) \tag{4.3}$$

$$M_n(x) = M_{n-1}(x) + f_n(x) \tag{4.4}$$

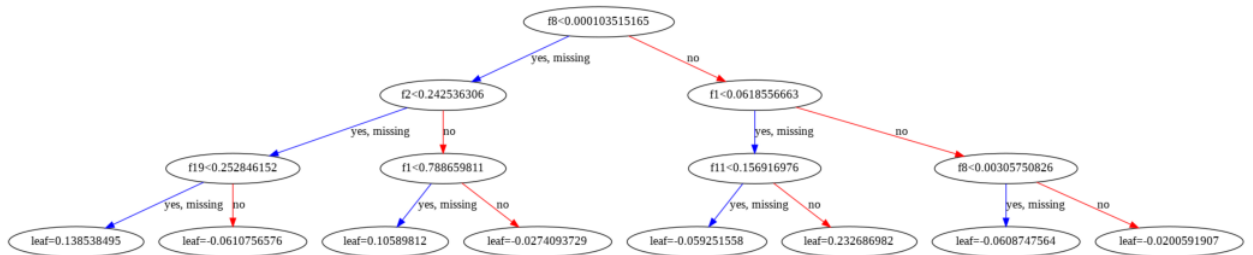


Figure 4.5: XGBoost Tree for 5-second window

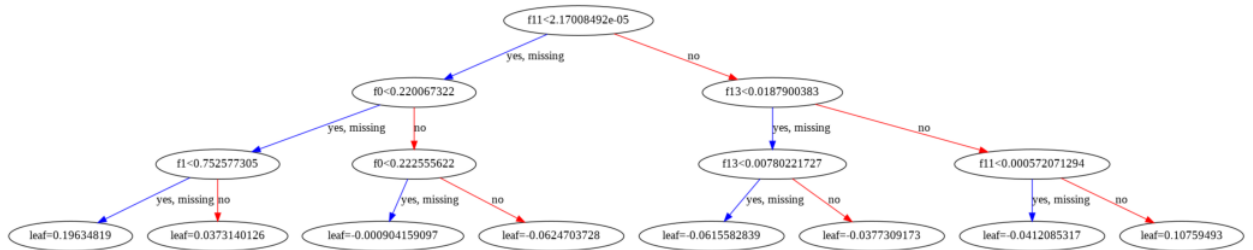


Figure 4.6: XGBoost Tree for half-second window

### 4.4.3 Multi-Layer Perceptron

Multi-Layer Perceptron (MLP) is a type of neural network that is used for classification and regression tasks in machine learning. It is composed of multiple layers of artificial neurons, with each layer connected to the next through weighted connections. MLP is known for its ability to learn complex relationships in the data and make predictions based on those relationships. One of the main benefits of using MLP is that it can be trained to approximate any continuous function, which makes it a flexible and powerful tool for many machine-learning tasks. However, MLP can be prone to overfitting, especially when the number of layers and neurons is too large, and it can be sensitive to the initialization of the weights. Overall, MLP is a widely-used and powerful machine learning algorithm that is known for its ability to learn complex relationships in the data. In our implementation, there is a total of 4 layers which include an input layer, 2 hidden layers, and an output layer. Figure 4.7 shows the architecture of the MLP neural network of our implementation. The model is implemented on both the 5-second and half-second window datasets. The 5-second window dataset has 20 features after pre-processing and for the half-second window, it is 24. Consequently, the MLP neural network has 20 inputs when implemented on the 5-second window dataset and 24 inputs for the half-second window dataset. It is shown from 1 to n in the Figure 4.7.

Moreover, each of the hidden layers has 256 neurons and the output layer has 5 neurons as there are 5 labels in our datasets which are Bus, Car, Still, Train, and Walking. We get a total of 72,453 parameters and 73,477 parameters for the 5-second window data and half-second window data respectively. All the parameters are trainable. Furthermore, the ReLU activation function is added at the end of each hidden layer and the softmax activation function is added at the output layer. Also, there are 2 dropouts of 40% is added after each of the hidden layers to eradicate overfitting. Sparse Categorical Crossentropy loss function and Adam optimizer are used to compile the model first on both of the datasets. Additionally, 60 epochs and 128 batch sizes are considered to perform the operations for both of the datasets. Table 4.1 and Table 4.2 show the model summary for 5-second window and half-second window datasets respectively. The weights in these two neural networks at each layer are calculated using 4.5 where  $w$  is the weight,  $lr$  represents the learning rate which is 0.01 in this case. Also,  $e_p$  and  $p_d$  define the expected values and predicted values respectively and  $i$  is the input vector. It is basically the bias getting added to the weight. The activation function ignites the neurons. Rectified Linear Unit (ReLU) is used in these neural networks as mentioned before.

$$w = w + lr * (e_p - p_d) * i \quad (4.5)$$

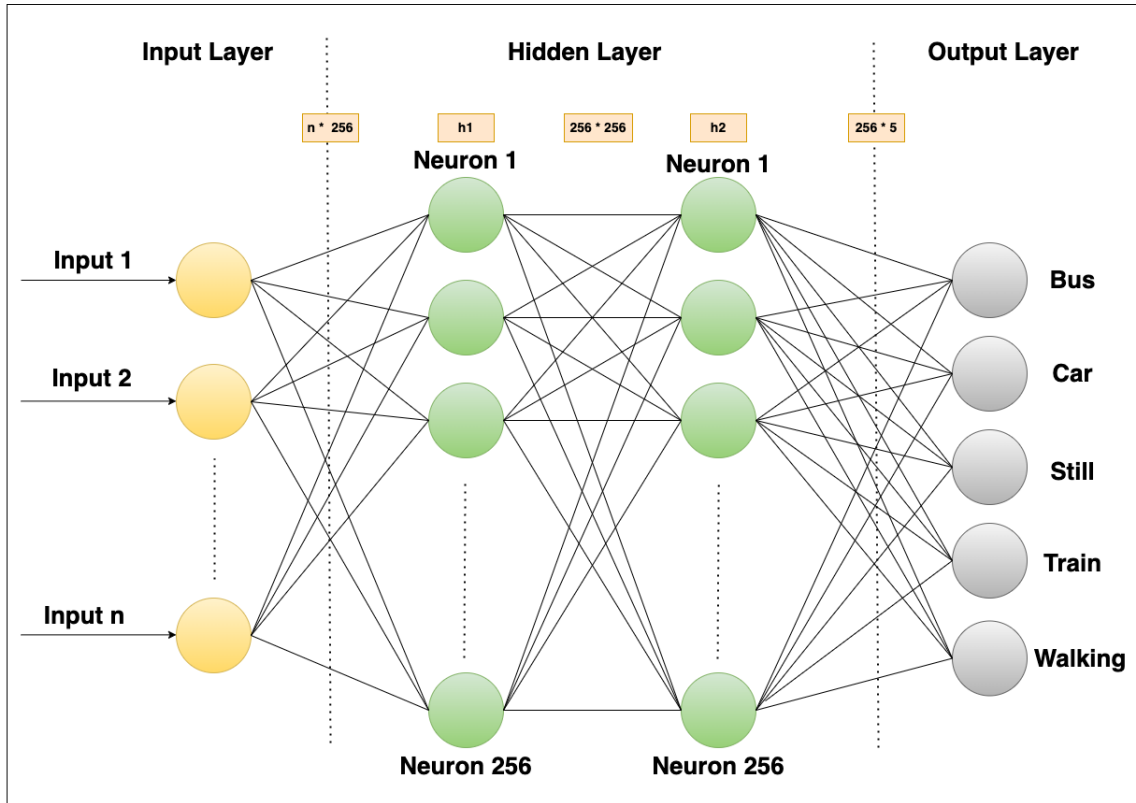


Figure 4.7: Multi-Layer Perceptron Neural Network Architecture

Layer	Output Shape	Parameters
inputLayer	(None, 20)	20
Dense: hiddenLayer1	(None, 256)	5376
Activation: ReLU	(None, 256)	0
Dropout: 40%	(None, 256)	0
Dense: hiddenLayer2	(None, 256)	65792
Activation: ReLU	(None, 256)	0
Dropout: 40%	(None, 256)	0
outputLayer	(None, 5)	1285
Activation: softmax	(None, 5)	0

Table 4.1: MLP Model Summary for 5-second window dataset

Layer	Output Shape	Parameters
inputLayer	(None, 20)	24
Dense: hiddenLayer1	(None, 256)	6400
Activation: ReLU	(None, 256)	0
Dropout: 40%	(None, 256)	0
Dense: hiddenLayer2	(None, 256)	65792
Activation: ReLU	(None, 256)	0
Dropout: 40%	(None, 256)	0
outputLayer	(None, 5)	1285
Activation: softmax	(None, 5)	0

Table 4.2: MLP Model Summary for half-second window dataset

#### 4.4.4 Federated Learning

Federated Learning is a machine learning approach that enables the training of a model using data that is distributed across a large number of devices, such as smartphones, tablets, or laptops. The main advantage of federated learning is that it allows the training of a model without the need to centralize the data, which helps to preserve the privacy and security of the data. In federated learning, a global model is trained using data from multiple local devices, and the local devices receive updates to the model from the global model. This process is repeated until the global model has converged to a satisfactory level of accuracy. Overall, federated learning is a powerful and useful machine learning approach that enables the training of models on distributed data while preserving the privacy and security of the data. As a decentralized approach, federated learning often provide less accurate result than usual. Numerous variations of federated learning are present. The Federated Averaging (FedAvg) method has been used in this study. The weight averaging in FedAvg is computed using 4.6 where  $w_{p+1}$  is the central model weight parameter,  $L$  is the number of total participants,  $s_l$  represents the samples of participant  $l$ ,  $s$  denotes the samples of all participants, and  $w_{p+1}^l$  is the local model weight parameter of participant  $l$ .

$$w_{p+1} = \sum_{l=1}^L \frac{s_l}{s} * w_{p+1}^l \quad (4.6)$$

#### 4.4.5 FederatedMLP

FederatedMLP is a type of federated learning algorithm that is used to train a multi-layer perceptron (MLP) model on data that is distributed across multiple devices. FederatedMLP uses the same basic principles as federated learning, but it is specifically designed to train MLP models. In FederatedMLP, a global MLP model is trained using data from multiple local devices, and the local devices receive updates to the model from the global model. This process is repeated until the global model has converged to a satisfactory level of accuracy. FederatedMLP is a useful tool for training MLP models on distributed data, as it allows the training of a model without the need to centralize the data, which helps to preserve the privacy and security of the data. In the implementation, 5 local clients are created. For the 5-second window dataset, the size of the training dataset is 4714. These data are distributed to 5 clients for training. Each client gets 942 data points. Also, for the half-second window dataset, the size of the training dataset is 5068. These data are distributed to 5 clients for training. Each client gets 10013 data points. The MLP neural network in Figure 4.7 is built inside this approach. 200 communication round is initialized. Moreover, the Sparse Categorical Crossentropy loss function and SGD optimizer with a 1% learning rate is used to compile the model. Algorithm 1 shows step by step workflow of the Federated MLP approach.

In the algorithm, the  $S_f$  is the scaling factor which is computed by equation 4.7, 4.8, and 4.9 inside the `WeightScalingFactor()` function with client batched data,  $C_b$  and client names which are the keys of the dictionary  $C_b$ . In the equations,  $g_c$ ,  $l_c$ ,  $C_b^c$ , and  $b_s$  are the global counts of the total training data points across clients, local counts of data points held by a single local client, the cardinality of  $C_b$ , and client's batch size respectively. The `WeightScalingFactor()` function determines the fraction of the total training data that all clients have is made up of local training data for each client. The  $S_w$  gets the scaled weights from each of the local model's weights based on the value of their scaling factor held by  $S_f$ . Additionally, the  $W_A$  holds the sum of the scaled weights. It gets the sum of the scaled weights together of all the clients'. It has the sum of the listed scaled weights which is equivalent to the scaled average of the weights. In the algorithm, the calculation of  $S_f$  and  $S_w$  is the operation of FedAvg which is shown in 4.6.

$$g_c = (\sum C_b^c \text{ for all clients}) * b_s \quad (4.7)$$

$$l_c = C_b^c \text{ for single local client} * b_s \quad (4.8)$$

$$S_f = \frac{l_c}{g_c} \quad (4.9)$$

---

**Algorithm 1** Federated MLP

---

**INPUT:**  $C_b$  : Dictionary of client and their data as client batched data,  
 $C_r$  : Number of communication rounds,  $C_l$  : Number of Clients,  
 $I_s$  : Input Shape,  $C$  : Number of classes

**OUTPUT:**  $T_m$  : Transport mode

- 1: Initialize global model,  $G_m$  as MLP
- 2:  $G_m \leftarrow BuildMLP(I_s, C)$
- 3: **for** each communication round  $C_r = 1, 2, 3... \mathbf{do}$
- 4:   Update global weights,  $W_G$  from  $G_m$
- 5:   Initialize scaled local weights,  $W_s$  as an empty list
- 6:   **for** each key of  $C_b = C \mathbf{do}$
- 7:     Initialize local model,  $L_m$  as MLP
- 8:      $L_m \leftarrow BuildMLP(I_s, C)$
- 9:      $W_L \leftarrow SetWeight(W_G)$
- 10:     Fit  $C_b$  for each  $C$  in  $L_m$
- 11:      $S_f \leftarrow WeightScalingFactor(C_b, C)$
- 12:      $S_w \leftarrow (S_f * W_L)$  for each  $W_L$
- 13:     Update  $W_s$  with  $S_w$
- 14:     Clear Session
- 15:   **end for**
- 16:    $W_A \leftarrow (\sum scaledweights)$  for each  $W_s$
- 17:    $W_G \leftarrow SetWeights(W_A)$
- 18:   Update  $G_m$  with  $W_G$
- 19:   Do classification with the trained model,  $G_m$
- 20:   Predict the  $T_m : G_m \rightarrow T_m$
- 21:   **return**  $T_m$
- 22: **end for**

---

#### 4.4.6 Federated Ensemble-Learning

A hybrid approach named Federated Ensemble-Learning is proposed to boost the performance further in a federated approach. An ensembling technique with majority voting is established inside federated learning. In this technique, we have initialized 5 local clients. Each local client's data is trained with two machine-learning algorithms and one deep-learning algorithm which are Random Forest, XGBoost, and MLP. The data get trained locally with MLP first with the received weights from the global model. After the training locally, the local weights get updated and it also updates the global model weights. Later, Random Forest and XGBoost fit data locally for each client and pass the trees to the global model for Random Forest and XGBoost. As Random Forest and XGBoost are not neural networks, they do not have any assigned weights. That is why instead of updating weights, data is getting fit in the local Random Forest and XGBoost locally and the trees get assigned to the global model for both of the approaches. This is how hybrid ensembling with a neural network and two non-neural networks is done. Subsequently, the prediction of the classes is made with the 3 models. After that majority voting decides which prediction to keep based on the voting of the 3 models. The predicted

class with the highest votes is considered and the remaining one is discarded (if any). For example, [3, 3, 3, 1, 0, 3], [3, 1, 1, 3, 0, 4], and [3, 1, 1, 3, 0, 4] are predicted classes for 3 models. These are the predicted classes for MLP, XGBoost, and Random Forest respectively. The predicted class for index 0 is 3 for all three models. So 3 is considered as the final predicted class here. Moreover, for index 1, the first model says 3 but the other two say 1; consequently, 1 is considered here.

Similarly, all of the indexes are providing the final predicted value through voting. Also, for 3 different votes, a class will be chosen randomly. In that case, the success probability is only 33%. But this is an extremely rare case as all the 3 models have very high accuracy. Most of the time they vote in the same class. As a result, the error rate falls by a considerable margin and we get an accurate result most of the time even after using a decentralized (federated) method. Figure 4.8 shows the architecture of the Proposed Federated Ensemble-Learning technique.

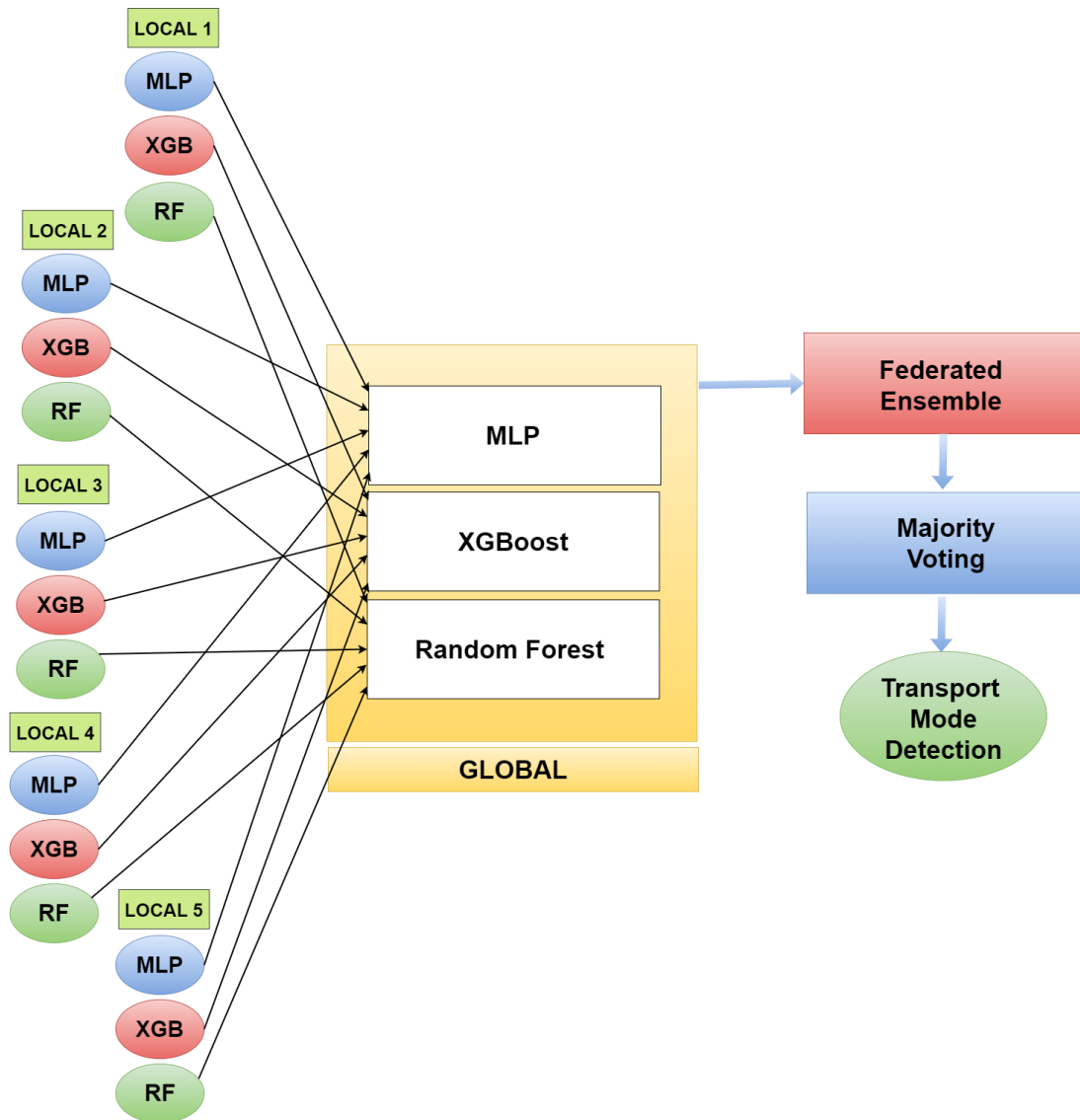


Figure 4.8: Proposed Federated Ensemble-Learning Architecture

Furthermore, the data processing and data distribution are exactly the same as the FedMLP. The MLP neural network in Figure 4.7 is built inside this approach. 200 communication round is initialized. Moreover, the Sparse Categorical Crossentropy loss function and SGD optimizer with a 1% learning rate is used to compile the model. Which is the exact same setting as Federated MLP. Algorithm 2 shows step by step workflow of the FedEL technique. In the algorithm, the scaling factor,  $S_f$  is computed by equation 4.7, 4.8, and 4.9 inside the `WeightScalingFactor()` function similar to the FedMLP Algorithm. The  $S_w$  and  $W_A$  are also computed as Algorithm 1.

---

**Algorithm 2** Proposed Federated Ensemble-Learning Technique

---

**INPUT:**  $C_b$  : Dictionary of client and their data as client batched data,

$C_r$  : Number of communication rounds,  $C_l$  : Number of Clients,

$I_s$  : Input Shape,  $C$  : Number of classes

**OUTPUT:**  $T_m$  : Transport mode

- 1: Initialize global model 1,  $G_{m_1}$  as MLP
  - 2:  $G_{m_1} \leftarrow BuildMLP(I_s, C)$
  - 3: Initialize global model 2,  $G_{m_2}$  as XGBoost
  - 4: Initialize global model 3,  $G_{m_3}$  as Random Forest
  - 5: Initialize local model 2,  $L_{m_2}$  as XGBoost
  - 6: Initialize local model 3,  $L_{m_3}$  as Random Forest
  - 7: **for** each communication round  $C_r = 1, 2, 3... \mathbf{do}$
  - 8:   Update global weights,  $W_G$  from  $G_{m_1}$
  - 9:   Initialize scaled local weights,  $W_s$  as an empty list
  - 10:   **for** each key of  $C_b = C \mathbf{do}$
  - 11:     Initialize local model 1,  $L_{m_1}$  as MLP
  - 12:      $L_{m_1} \leftarrow BuildMLP(I_s, C)$
  - 13:      $W_L \leftarrow SetWeight(W_G)$
  - 14:     Fit  $C_b$  for each  $C$  in  $L_{m_1}$
  - 15:     Fit  $C_b$  for each  $C$  in  $L_{m_2}$
  - 16:     Fit  $C_b$  for each  $C$  in  $L_{m_3}$
  - 17:      $S_f \leftarrow WeightScalingFactor(C_b, C)$
  - 18:      $S_w \leftarrow (S_f * W_L)$  for each  $W_L$
  - 19:     Update  $W_s$  with  $S_w$
  - 20:     Clear Session
  - 21:   **end for**
  - 22:    $W_A \leftarrow (\sum scaledweights)$  for each  $W_s$
  - 23:    $W_G \leftarrow SetWeights(W_A)$
  - 24:   Update  $G_{m_1}$  with  $W_G$
  - 25:    $G_{m_2} \leftarrow L_{m_2}$
  - 26:    $G_{m_3} \leftarrow L_{m_3}$
  - 27:    $P_1 \leftarrow ClassPrediction(G_{m_1})$
  - 28:    $P_2 \leftarrow ClassPrediction(G_{m_2})$
  - 29:    $P_3 \leftarrow ClassPrediction(G_{m_3})$
  - 30:    $T_m \leftarrow MajorityVoting(P_1, P_2, P_3)$
  - 31:   **return**  $T_m$
  - 32: **end for**
-



# Chapter 5

## Result Analysis

### 5.1 Performance Evaluation Metrics

The proposed Federated Ensemble-Learning technique for transport mode detection has been proven to be effective through extensive experimentation with the TMD dataset. Its superiority over other centralized and decentralized approaches has been demonstrated through the use of well-known performance metrics such as accuracy, precision, recall, F1-score, and confusion matrix. These results demonstrate that the Federated Ensemble-Learning technique is a reliable and effective method for transport mode detection using sensory data fetched from smartphones.

**Accuracy:** The accuracy metric displays what percentage of all samples correctly identified the various transport mode classes. The accuracy of the proposed method has been evaluated with the test dataset using 5.1.

$$Acc = \frac{S_c}{T_c} \quad (5.1)$$

where  $Acc$ ,  $S_c$ , and  $T_c$  represent the accuracy, the correctly detected transport mode category samples, and the total number of samples respectively.

**Precision:** The proportion of samples for the same transportation mode that can be accurately identified out of all the predictions is known as the precision metric. The suggested method's precision was evaluated on the test dataset using 5.2.

$$P_r = \frac{T_p}{T_p + F_p} \quad (5.2)$$

where  $P_r$ ,  $T_p$ , and  $F_p$  represent the precision, the true positive, and the false positive, respectively.

**Recall:** The recall metric is defined as the proportion of truly recognized transportation mode samples out of the total number of samples of the same transportation mode. On the test dataset, the proposed method’s recall metric was calculated using 5.3.

$$R_e = \frac{T_p}{T_p + F_n} \quad (5.3)$$

where  $R_e$ ,  $T_p$ , and  $F_n$  represent the recall, the true positive, and the false negative, respectively.

**F1-score:** The F1-score metric is the harmonic mean of the precision metric and the recall metric. Based on the precision and recall values of a model, it presents a score for relative performance. The F1-score of the proposed approach was determined on the test dataset using 5.4.

$$F_1 = 2 * \frac{P_r * R_e}{P_r + R_e} \quad (5.4)$$

where  $F_1$ ,  $P_r$ , and  $R_e$  represent the F1-score, the precision, and the recall, respectively.

**Confusion Matrix:** An  $N \times N$  square matrix, where  $N$  is the number of classes in a multi-class classification task, is referred to as a confusion matrix. Five different modes of transportation were taken into consideration, giving  $N$  a value of 5. In a confusion matrix, the rows stand in for the expected classes, while the columns represent the true classes. As a result, a confusion matrix shows how the actual and expected modes of transportation might be compared.

## 5.2 Experimental Result Analysis

Model	Precision	Recall	F1-Score	Accuracy	CV Score
XGBoost	97%	97%	97%	97%	96%
Random Forest	98%	98%	98%	98%	98%
Decision Tree	92%	92%	92%	92%	92%
SVM	84%	84%	84%	84%	55%
KNN	94%	94%	94%	94%	92%

Table 5.1: Evaluation Metrics Result for Traditional Machine Learning Models for 5-second window dataset (20 Features)

Model	Precision	Recall	F1-Score	Accuracy	CV Score
XGBoost	96%	96%	96%	96%	77%
Random Forest	99%	99%	99%	99%	78%
Decision Tree	98%	98%	98%	98%	93%
SVM	86%	86%	86%	86%	34%
KNN	98%	98%	98%	98%	63%

Table 5.2: Evaluation Metrics Result for Traditional Machine Learning Models for half-second window dataset (24 Features)

First, the data is trained and tested with 5 traditional machine-learning models as mentioned earlier which are XGBoost, Random Forest, Decision Tree, Support Vector Machine (SVM), and K-Nearest Neighbour (KNN). Table 5.1 and Table 5.2 show the evaluation metrics results of the models for 5-second window and half-second window datasets respectively. It is observed that all of the models perform well on both of the datasets apart from SVM. Additionally, the cross-validation (CV) score for KNN is not satisfactory for the dataset with 24 features. Cross-validation is a powerful technique for evaluating machine learning models as it allows for an unbiased estimate of performance, and it makes use of all the data available for training and testing. It estimates the performance of a model on independent datasets. Moreover, even though Decision Tree has a better overall outcome compared to the rest of the models, Random Forest and XGBoost are selected for the ensembling technique as they perform better in ensembling methods. Also, Random Forest is just an ensemble of numerous Decision Trees. Consequently, the traits of Decision Trees are received inside Random Forests. Overall, Random Forest and XGBoost are comparatively the best fit for the proposed technique as they have a very good response as shown in the tables.

Dataset	Accuracy	Precision	Recall	F1-Score	Loss
5-second window (20 Features)	92%	93%	91%	92%	0.26
Half-second window (24 Features)	97%	98%	97%	97%	0.09

Table 5.3: Evaluation Metrics Result for Multi-Layer Perceptron

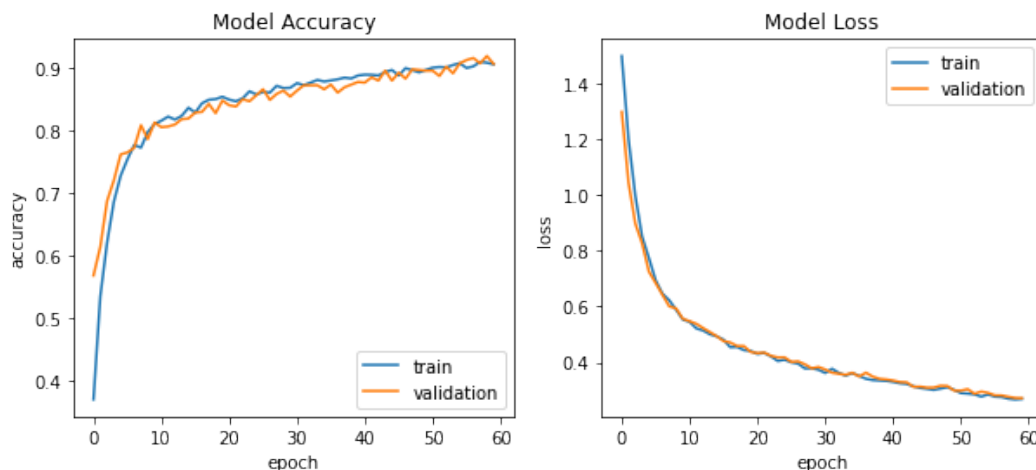


Figure 5.1: MLP Accuracy and Loss Curve for 5-second window dataset (20 Features)

Second, the Multi-Layer Perceptron (MLP) neural network is implemented to fit into a federated averaging approach. This deep-learning approach performs well for both of the datasets. Accuracy, precision, recall, f-1 score, and loss (sparse categorical crossentropy) of MLP for both of the datasets is shown in Table 5.3. It is observed that the results get better for bigger datasets. With more features and bigger size, it is performing better in the half-second window dataset as the half-second window dataset has more than 50K training data and the 5-second window dataset has less than 5k training data. The performance boost-up is noticeable. The accuracy jumps from 92% to 97% for the half-second window dataset and the loss decreases from 0.26 to 0.09. Figure 5.1 and Figure 5.2 show the MLP model accuracy and loss for training and validation data for the 5-second window and half-second window datasets respectively. Both of the figures show that the accuracy curves are upward and the loss curves are downward which is going on as the number of epochs increases. Also, the validation data is following the characteristics of the training data.

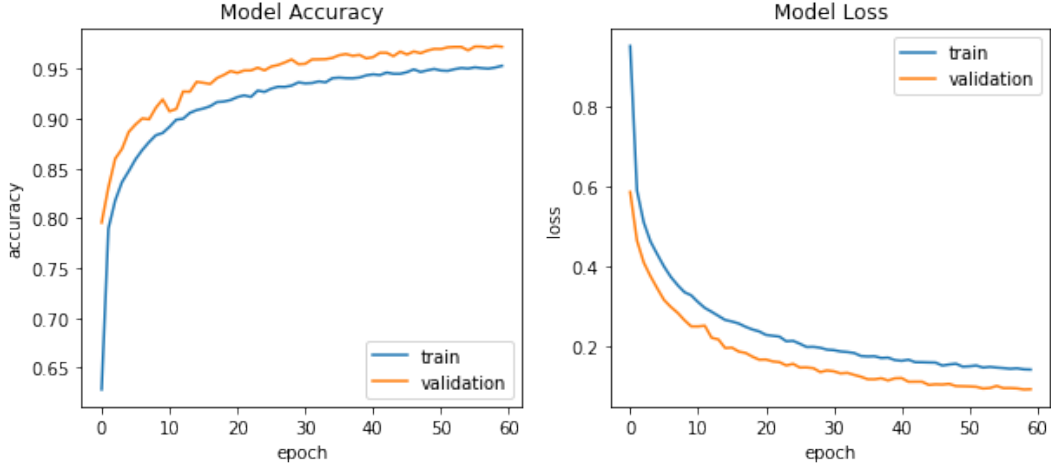


Figure 5.2: MLP Accuracy and Loss Curve for half-second window dataset (24 Features)

Model	Communication Round Number	Precision	Recall	F1-Score	Accuracy
Federated MLP	198	87%	86%	86%	86.18%
	199	87%	86%	86%	86.00%
	200	86%	86%	86%	85.75%
Federated Ensemble-Learning	198	95%	95%	95%	94.91%
	199	95%	95%	95%	94.83%
	200	95%	95%	95%	95.10%

Table 5.4: Evaluation Metrics Result of Federated Approaches for 5-second window dataset (20 Features)

Furthermore, a federated approach called Federated MLP is implemented first using the MLP neural network. As federated learning is a decentralized approach, inaccuracy increases due to distributed training. From Table 5.4 and Table 5.5, it is observed that for Federated MLP the accuracy dropped from 92% to 86% for the 5-second window dataset and it dropped from 97% to 90.5% for the half-second window dataset. The observation is from the last three communication rounds of 200 communication rounds. Though FedMLP started with 30% accuracy for the 5-second window dataset and 58% accuracy for the half-second window dataset, it increases as the communication round proceeds but gets saturated after a certain point of time. In this case, it is 86% and 90.5% for the FedMLP. The accuracy, precision, recall, and F-1 score do not seem optimal for the system. That is why our proposed technique, Federated Ensemble-Learning gets introduced. Table 5.4 and Table 5.5 are showing the dominance of the proposed method over the usual federated approach for both of the datasets. It dominates FedMLP in every case. It occurs due to the technique of ensemble. The FedEL is designed inside the same setup as the FedMLP approach. The only difference is integrating Random Forest and XGBoost with the ensemble method along with MLP which is discussed earlier.

Consequently, where MLP made mistakes with the FedMLP approach, Random

Model	Communication Round Number	Precision	Recall	F1-Score	Accuracy
Federated MLP	198	91%	90%	90%	90.42%
	199	90%	90%	90%	90.28%
	200	91%	91%	91%	90.54%
Federated Ensemble-Learning	198	99%	99%	99%	98.58%
	199	99%	99%	99%	98.64%
	200	99%	99%	99%	98.61%

Table 5.5: Evaluation Metrics Result of Federated Approaches for half-second window dataset (24 Features)

Forest and XGBoost rectify the mistakes with majority voting. This is how the proposed method achieves 95% accuracy after 200 communication rounds for the 5-second window dataset and almost 99% accuracy after 200 communication rounds for the half-second window dataset. The accuracy is similar since the first communication round as Random Forest and XGBoost tree is passed after the first iteration. Because of that, they return to the correct class through majority voting. Both of them do not have to update weights at each iteration and make the model better, unlike MLP. As a result, they are able to classify the labels properly from the beginning. Again, the accuracy is better on the half-second window dataset due to the bigger size and more features than the 5-second window one. From this observation, it can be perceived that collecting data frequently (in a half second) is more fruitful rather than collecting data with a pause (5-second gap) from smartphones or edge devices.

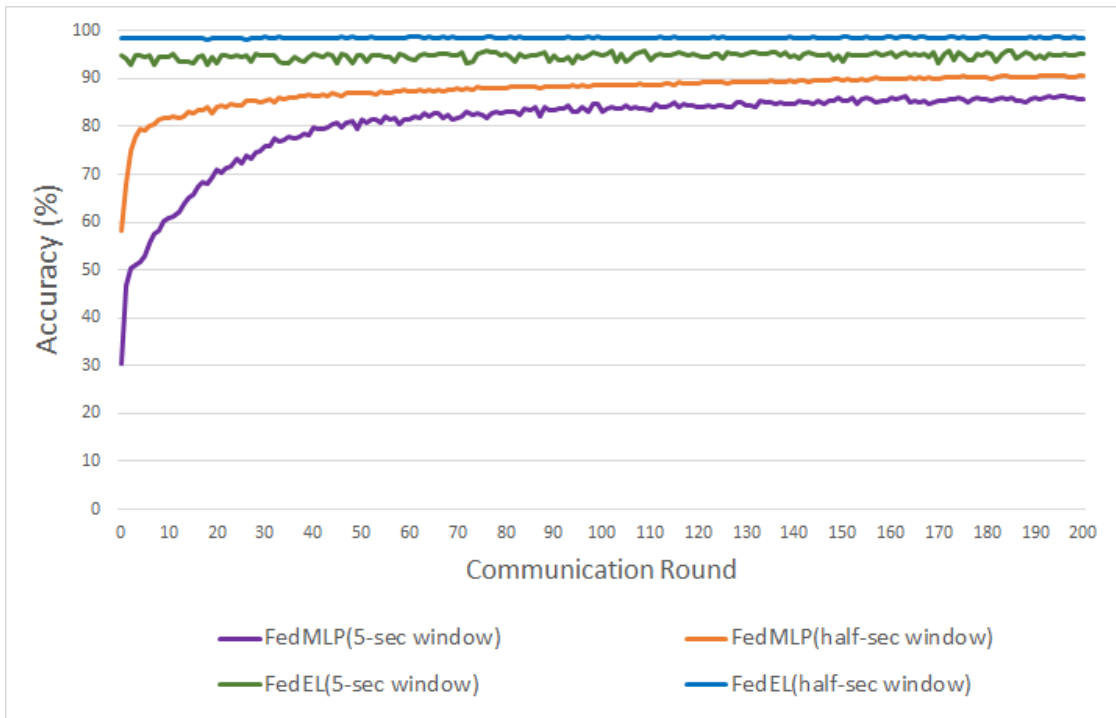


Figure 5.3: Comparison Curves for FedMLP and FedEL on Both Datasets

The Figure 5.3 shows the comparison curves for FedMLP and FedEL on both datasets. It is observed that for the FedMLP approaches, the accuracy starts at a lower point initially and then reaches a saturated point as the communication rounds proceed. The saturated points are slightly greater than 86% for the 5-second window dataset and just above 90% for the half-second window dataset which started from 30% and 58% respectively. But for the proposed FedEL technique, the accuracy starts at the peak of it and holds it till the last. It happens due to the ensemble of RF and XGB. Even though there are some ups and downs in the 5-second window dataset but for more data, in the half-second window dataset, the line stays stable holding an accuracy close to 99%.

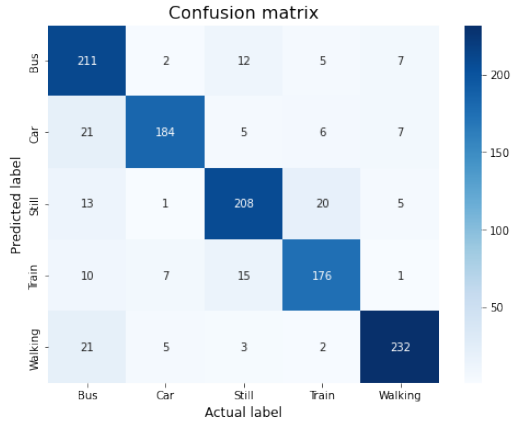
Transportation Mode	5-second Window Dataset		Half-second Window Dataset	
	Value Count	Accuracy	Value Count	Accuracy
Bus	237	82%	2490	89%
Car	223	87%	2546	92%
Still	247	85%	2536	91%
Train	209	84%	2416	87%
Walking	263	90%	2529	94%

Table 5.6: Accuracy Scores on Different Transportation Modes of Communication Round Number 200 for Federated MLP Approach

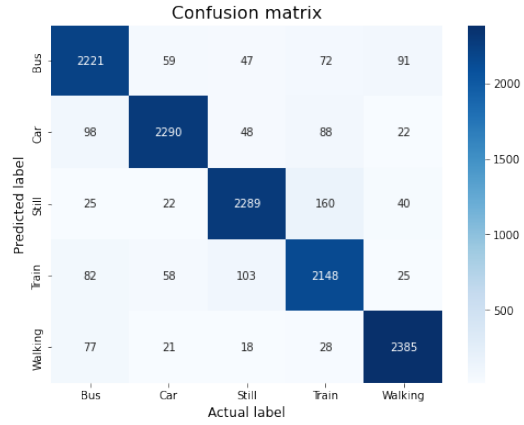
Transportation Mode	5-second Window Dataset		Half-second Window Dataset	
	Value Count	Accuracy	Value Count	Accuracy
Bus	242	93%	2535	98%
Car	235	94%	2543	99%
Still	236	97%	2438	99%
Train	238	95%	2504	99%
Walking	228	96%	2497	98%

Table 5.7: Accuracy Scores on Different Transportation Modes of Communication Round Number 200 for Proposed Federated Ensemble-Learning Technique

Moreover, Table 5.6 and Table 5.7 show the accuracy scores on different transportation modes of communication round number 200 for both of the federated approaches. The value counts for the 5-second window dataset sums to 1179 and for the half-second window dataset, it is 12517 which is the total size of the test data. The value count is the finding of the number of each transportation mode. It is seen that the proposed FedEL technique dominates the FedMLP approach in every mode. An observation is that the Bus transportation mode has the least accuracy for both of the models on both of the datasets. But it is negligible for the FedEL approach on the half-second window dataset.

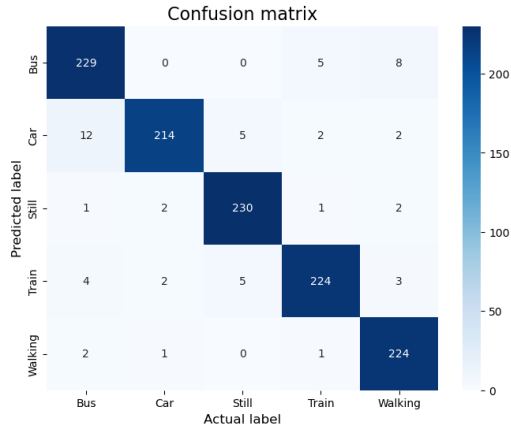


(a) 5-second Window Dataset

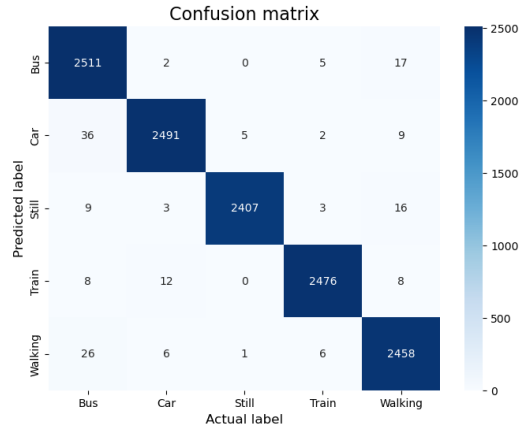


(b) half-second Window Dataset

Figure 5.4: Confusion Matrix for Federated MLP



(a) 5-second Window Dataset

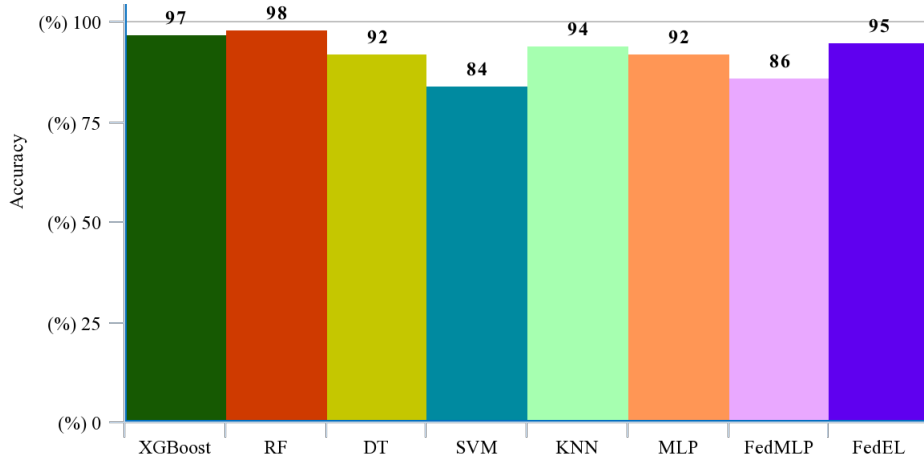


(b) half-second Window Dataset

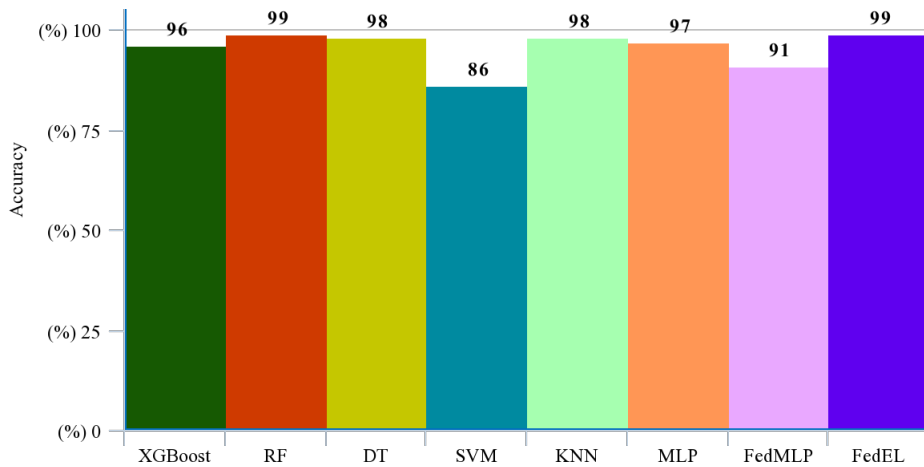
Figure 5.5: Confusion Matrix for Federated Ensemble-Learning

Additionally, Figure 5.4 and Figure 5.5 portray the confusion matrix of FedMLP and FedEL respectively for both of the datasets. It consists of true and false values which are derived from precision and recall. Numerous false values are observed for the FedMLP approach compared to the FedEL approach. FedEL on the half-second window dataset is the most accurate matrix compared to the others. Even though it has a big number of false values for Bus but the percentage is very low which is negligible.





(a) 5-second Window Dataset



(b) half-second Window Dataset

Figure 5.6: Accuracy on Different Models (Rounded Values)

Finally, the Figure 5.6 shows the comparison of accuracy on different models. From the figure, it is observed that our proposed FedEL model outperforms almost every implemented centralized and decentralized model if enough data is provided for training the model. The proposed model achieves a promising result with an accuracy of 98.6% which is almost 99%. This approach can be used as a solution to secure the privacy of the sensory data from smartphones and other edge devices because even after being a distributed approach, it has excellent accuracy and precision in classifying target labels like the transport mode detection of this study. This method can assist in detecting the transportation mode with smartphone or edge device data securely.

# Chapter 6

## Conclusion

The transportation and traffic systems are being advanced due to the ITS, which is a blessing brought about by the progression of modern technology. The system would not function properly without TMD as one of its components. In order to address road issues, the Vehicular Edge Network connects all of the vehicles and vehicular edge devices on the roads. In addition, the Federated Learning (FL) approach has been implemented because, in contrast to conventional machine learning, FL does not require the transfer of all data in order to complete the learning process. This results in a reduction in the complexity of the ever-growing database for edge devices that the vehicle utilizes. Our proposed Federated Ensemble-Learning model outperforms practically every other implemented centralized and decentralized model with promising results. This is due to the fact that it is a hybrid model that was established using the ensemble technique. If sufficient training data are available, then our model accomplishes great results. Our strategy can be utilized to protect the privacy of the sensory data collected by smartphones and other edge devices because, despite being a distributed method, it possesses outstanding accuracy and precision in classifying target labels, such as the detection of transport modes in this research. This is made possible by the fact that our strategy retains excellent accuracy and precision in categorizing target labels. In wireless mobile edge computing at the edge devices, the suggested technique offers a solution to the problem of reducing latency while also protecting users' privacy. This would allow for more precise classification of edge data within Vehicular Edge Networks, which are part of the whole Intelligent Transportation System.

## 6.1 Challenges

First, at least one neural network approach was required in order to implement the FedAvg method as a distributed ML. Finding the correct one was not easy. After numerous experiments with Extreme Learning Machine, Deep Belief Networks, Long Short-Term Memory, and MLP, we found the most accurate approach with the correct combinations of layers and other parameters which is MLP. Also, the process requires strong hardware resources to be trained and validated fast on a large dataset. This issue is addressed by completing the operations in a moderately high-performance system powered by AMD Ryzen 5 3600 CPU and Nvidia RTX2060 GPU. It can perform even faster in a better system.

## 6.2 Limitations

The federated approach of RF and XGB is not addressed in this research by handling weights. The proposed method passes locally trained trees of RF and XGB to their corresponding global models. The data gets full security here as it is trained locally and passes the trained model to the global model but there are no operations of weights and no use of FedAvg for these two models. As federated approaches are already well-built, this study made an effort to propose a new approach by ensemble learning instead of average weights locally. Additionally, due to the limitations of resources, the method is not evaluated in actual distributed systems using real-world data. As a result, the latency improvement in wireless edge devices compared to centralized approaches is not assessed.

## 6.3 Future Work

In future research on the proposed technique, an attempt will be made to fit the FedEL model on time series data integrating Long Short-Term Memory (LSTM) as the neural network. The success of this attempt might assist in controlling and maintaining the traffic flow effectively in a distributed way. An AI-based smart traffic management system can be introduced grounded by the method. Also, the performance of the overall system in the real world will be measured by implementing a VEN topology powered by Road Side Units incorporating Mobile Edge Computing Servers which will be the proposed FedEL enabled.

# Bibliography

- [1] “2021 saw more road accidents than 2020.” <https://www.dhakatribune.com/nation/2022/01/08/study-nearly-6300-killed-in-road-accidents-in-a-year>, Jan. 2022.
- [2] A. A. Haider, “Traffic jam: The ugly side of dhaka’s development.” <https://www.thedailystar.net/opinion/society/traffic-jam-the-ugly-side-dhakas-development-1575355>, May 2018.
- [3] S. Guo, C. Chen, J. Wang, Y. Liu, K. Xu, Z. Yu, D. Zhang, and D. M. Chiu, “Rod-revenue: Seeking strategies analysis and revenue prediction in ride-on-demand service using multi-source urban data,” *IEEE Transactions on Mobile Computing*, vol. 19, no. 9, pp. 2202–2220, 2019.
- [4] C. Meng, X. Yi, L. Su, J. Gao, and Y. Zheng, “City-wide traffic volume inference with loop detector data and taxi trajectories,” in *Proceedings of the 25th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pp. 1–10, 2017.
- [5] Z. MacHardy, A. Khan, K. Obana, and S. Iwashina, “V2x access technologies: Regulation, research, and remaining challenges,” *IEEE Communications Surveys & Tutorials*, vol. 20, no. 3, pp. 1858–1877, 2018.
- [6] M. Abbasi, H. Rezaei, V. G. Menon, L. Qi, and M. R. Khosravi, “Enhancing the performance of flow classification in sdn-based intelligent vehicular networks,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 7, pp. 4141–4150, 2020.
- [7] C. Xiang, Z. Zhang, Y. Qu, D. Lu, X. Fan, P. Yang, and F. Wu, “Edge computing-empowered large-scale traffic data recovery leveraging low-rank theory,” *IEEE Transactions on Network Science and Engineering*, vol. 7, no. 4, pp. 2205–2218, 2020.
- [8] K. Wang and Z. Shen, “Artificial societies and gpu-based cloud computing for intelligent transportation management,” *IEEE Intelligent Systems*, vol. 26, no. 4, pp. 22–28, 2011.
- [9] D. Shin, D. Aliaga, B. Tunçer, S. M. Arisona, S. Kim, D. Zünd, and G. Schmitt, “Urban sensing: Using smartphones for transportation mode classification,”

*Computers, Environment and Urban Systems*, vol. 53, pp. 76–86, 2015. Special Issue on Volunteered Geographic Information.

- [10] N. Ohmori, M. Nakazato, N. Harata, K. Sasaki, and K. Nishii, “Activity diary surveys using gps mobile phones and pda,” in *85th Annual Meeting of the Transportation Research Board, Washington, DC*, pp. 22–26, 2006.
- [11] J. Froehlich, T. Dillahunt, P. Klasnja, J. Mankoff, S. Consolvo, B. Harrison, and J. A. Landay, “Ubigreen: investigating a mobile tool for tracking and supporting green transportation habits,” in *Proceedings of the sigchi conference on human factors in computing systems*, pp. 1043–1052, 2009.
- [12] M. H. Alkinani, A. A. Almazroi, M. Adhikari, and V. G. Menon, “Artificial intelligence-empowered logistic traffic management system using empirical intelligent xgboost technique in vehicular edge networks,” *IEEE Transactions on Intelligent Transportation Systems*, 2022.
- [13] W. Y. B. Lim, N. C. Luong, D. T. Hoang, Y. Jiao, Y.-C. Liang, Q. Yang, D. Niyato, and C. Miao, “Federated learning in mobile edge networks: A comprehensive survey,” *IEEE Communications Surveys & Tutorials*, vol. 22, no. 3, pp. 2031–2063, 2020.
- [14] X. Li, L. Cheng, C. Sun, K.-Y. Lam, X. Wang, and F. Li, “Federated-learning-empowered collaborative data sharing for vehicular edge networks,” *IEEE Network*, vol. 35, no. 3, pp. 116–124, 2021.
- [15] Z. Du, C. Wu, T. Yoshinaga, K.-L. A. Yau, Y. Ji, and J. Li, “Federated learning for vehicular internet of things: Recent advances and open issues,” *IEEE Open Journal of the Computer Society*, vol. 1, pp. 45–61, 2020.
- [16] Z. Qin, G. Y. Li, and H. Ye, “Federated learning and wireless communications,” *IEEE Wireless Communications*, vol. 28, no. 5, pp. 134–140, 2021.
- [17] X. Xu, B. Shen, S. Ding, G. Srivastava, M. Bilal, M. R. Khosravi, V. G. Menon, M. A. Jan, and M. Wang, “Service offloading with deep q-network for digital twinning-empowered internet of vehicles in edge computing,” *IEEE Transactions on Industrial Informatics*, vol. 18, no. 2, pp. 1414–1423, 2022.
- [18] G. Yan and Q. Qin, “The application of edge computing technology in the collaborative optimization of intelligent transportation system based on information physical fusion,” *IEEE Access*, vol. 8, pp. 153264–153272, 2020.
- [19] A. Munusamy, M. Adhikari, M. A. Khan, V. G. Menon, S. N. Srirama, L. T. Alex, and M. R. Khosravi, “Edge-centric secure service provisioning in iot-enabled maritime transportation systems,” *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–10, 2021.
- [20] R. R. Patil and V. N. Honmane, “Smart transportation for smart cities,” in *International Conference on Soft Computing Systems*, pp. 53–61, Springer, 2018.
- [21] K. Zhang, Y. Mao, S. Leng, S. Maharjan, and Y. Zhang, “Optimal delay constrained offloading for vehicular edge computing networks,” in *2017 IEEE International Conference on Communications (ICC)*, pp. 1–6, IEEE, 2017.

- [22] X. Li, Y. Dang, and T. Chen, “Vehicular edge cloud computing: Depressurize the intelligent vehicles onboard computational power,” in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pp. 3421–3426, IEEE, 2018.
- [23] M. Khayyat, I. A. Elgendy, A. Muthanna, A. S. Alshahrani, S. Alharbi, and A. Koucheryavy, “Advanced deep learning-based computational offloading for multilevel vehicular edge-cloud computing networks,” *IEEE Access*, vol. 8, pp. 137052–137062, 2020.
- [24] F. Tang, B. Mao, N. Kato, and G. Gui, “Comprehensive survey on machine learning in vehicular network: technology, applications and challenges,” *IEEE Communications Surveys & Tutorials*, 2021.
- [25] M. Woodard, M. Wisely, and S. S. Sarvestani, “Chapter three - a survey of data cleansing techniques for cyber-physical critical infrastructure systems,” vol. 102 of *Advances in Computers*, pp. 63–110, Elsevier, 2016.
- [26] L. Lim, “Lim wyb, luong nc, hoang dt, jiao y., liang y.-c., yang q., niyato d., miao c,” *Federated learning in mobile edge networks: a comprehensive survey, IEEE Commun. Surv. Tutor.*, vol. 22, no. 3, pp. 2031–2063, 2020.
- [27] C. Feng, Z. Zhao, Y. Wang, T. Q. Quek, and M. Peng, “On the design of federated learning in the mobile edge computing systems,” *IEEE Transactions on Communications*, vol. 69, no. 9, pp. 5902–5916, 2021.
- [28] Z. Qin, G. Y. Li, and H. Ye, “Federated learning and wireless communications,” *IEEE Wireless Communications*, vol. 28, no. 5, pp. 134–140, 2021.
- [29] J. Konečný, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, “Federated learning: Strategies for improving communication efficiency,” *arXiv preprint arXiv:1610.05492*, 2016.
- [30] A. M. Elbir, B. Soner, and S. Coleri, “Federated learning in vehicular networks,” *arXiv preprint arXiv:2006.01412*, 2020.
- [31] A. Taïk, Z. Mlika, and S. Cherkaoui, “Clustered vehicular federated learning: Process and optimization,” *IEEE Transactions on Intelligent Transportation Systems*, 2022.
- [32] C. Xiang, Z. Zhang, Y. Qu, D. Lu, X. Fan, P. Yang, and F. Wu, “Edge computing-empowered large-scale traffic data recovery leveraging low-rank theory,” *IEEE Transactions on Network Science and Engineering*, vol. 7, no. 4, pp. 2205–2218, 2020.
- [33] Y. Zhu, S. Zhang, Y. Liu, D. Niyato, and J. J. Yu, “Robust federated learning approach for travel mode identification from non-iid gps trajectories,” in *2020 IEEE 26th International Conference on Parallel and Distributed Systems (ICPADS)*, pp. 585–592, 2020.

- [34] C. Zhang, Y. Zhu, C. Markos, S. Yu, and J. J. Q. Yu, “Toward crowdsourced transportation mode identification: A semisupervised federated learning approach,” *IEEE Internet of Things Journal*, vol. 9, no. 14, pp. 11868–11882, 2022.
- [35] J. J. Yu, “Semi-supervised deep ensemble learning for travel mode identification,” *Transportation Research Part C: Emerging Technologies*, vol. 112, pp. 120–135, 2020.
- [36] H. Lu, M. Pinaroc, M. Lv, S. Sun, H. Han, and R. C. Shah, “Locomotion recognition using xgboost and neural network ensemble,” in *Adjunct Proceedings of the 2019 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2019 ACM International Symposium on Wearable Computers*, UbiComp/ISWC ’19 Adjunct, (New York, NY, USA), p. 757–760, Association for Computing Machinery, 2019.
- [37] Z. Xiao, Y. Wang, K. Fu, and F. Wu, “Identifying different transportation modes from trajectory data using tree-based ensemble classifiers,” *ISPRS International Journal of Geo-Information*, vol. 6, no. 2, 2017.
- [38] C. Carpineti, V. Lomonaco, L. Bedogni, M. Di Felice, and L. Bononi, “Custom dual transportation mode detection by smartphone devices exploiting sensor diversity,” in *2018 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, pp. 367–372, IEEE, 2018.
- [39] “Welcome to tmd dataset; the first free dataset about transportation mode detection.” <https://tempesta.cs.unibo.it/projects/us-tm2017/>, 2017.
- [40] J. Lee Rodgers and W. A. Nicewander, “Thirteen ways to look at the correlation coefficient,” *The American Statistician*, vol. 42, no. 1, pp. 59–66, 1988.
- [41] A. Jain, K. Nandakumar, and A. Ross, “Score normalization in multimodal biometric systems,” *Pattern recognition*, vol. 38, no. 12, pp. 2270–2285, 2005.