2024

Application of Fusion based Deep Learning Models to Improve Millimeter Wave Beamforming

Abishek Subramanian
Michigan Technological University, absubram@mtu.edu

APPLICATION OF FUSION-BASED DEEP LEARNING MODELS TO IMPROVE MILLIMETER WAVE BEAMFORMING

By

Abishek Subramanian

A THESIS

Submitted in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

In Electrical and Computer Engineering

MICHIGAN TECHNOLOGICAL UNIVERSITY

2024

This thesis has been approved in partial fulfillment of the requirements for the Degree of MASTER OF SCIENCE in Electrical and Computer Engineering.

Department of Electrical and Computer Engineering

Thesis Advisor:	*Dr. Aurenice M. Oliveira*
Committee Member:	*Dr. K. C. Dukka*
Committee Member:	*Dr. Anthony Pinar*
Department Chair:	*Dr. Jim Choi*

Table of Contents

List of Figures

List of Tables

Author Contribution Statement

This report summarizes the research conducted to use side information from out-of-band sensor modalities to improve latency, and therefore improving beamforming pairs for vehicle-to-infrastructure (V2I) communication. The proposed method focuses on using non-RF multi sensor modalities incorporating Fusion based Deep Leaning (F-DL) techniques to embed environmental spatial information, which can be used for ray tracing. A centralized architecture encapsulating F-DL implementation was initially investigated and presented a series of shortcomings. In order to address these shortcomings, a distributed architecture was also investigated.

A generalized federated learning framework using biased client selection strategies was designed and implemented to incorporate distributed architecture. The distributed architecture proved to be the best approach for practical deployment and scalability when using multimodality driven F-DL models to improve beamforming process of millimeter Wave for V2I communication networks.

The results from this study will be published a research paper and conclude the research conducted as part of the fulfillment towards master's thesis.

Acknowledgements

I would to express my sincere gratitude to Dr. Aurenice Oliveira, my thesis advisor, for her mentorship, guidance, and continuous support throughout my Master's thesis research. I am also grateful to Dr. Aurenice Oliveira for the opportunity to identify the research project that I carried out for this thesis, and for allowing me to use the infrastructure resources at Communication Systems and Vehicular Network Technologies Laboratory.

I would like to thank Prof. Gowtham for his guidance and for providing me support to use the "Portage" High Performance Computing System at Michigan Technological University to train and test the models I used in this thesis.

I would like to also thank Prof. K.C Dukka and Prof. Anthony Pinar for kindly accept to serve as committee members for my defense and for providing valuable feedback to improve the thesis.

I would like to thank my fellow colleagues, faculty, and staff of the Department of Electrical and Computer Engineering at Michigan Technological University for their contributions to my graduate education.

I conclude by thanking my family and friends for their unconditional support throughout my education.

Definitions

Beam Forming: It is a signal processing technique used in wireless systems to improve sound to noise ratio by adjusting the phase and amplitude of the signals transmitted by an array of antennas, in order to create a more focused and directional signal focused at the receiver. [9]

Beam selection: The process of selecting the best beam or directional antenna to attain reliable communication without inference. [9]

Codebooks: It refers to the configuration set for the transmitter/receiver array of antenna.[9] [10]

Multimodal Data: Usage of multiple data points from sensors as inputs to derive and embed useful information. [5]

ReLU: It is an activation function in ML/NN where the unit ramp function is slightly modified with a small slope greater than 0 for negative x axis.

Unimodal: It is a deep learning network that extracts embedded features from the sensor information which can be further used in a fusion network. [5]

Fusion Network: I the context of this report, it is a way of combining multiple features extracted from different unimodal networks.[9]

SoftMax: It is a mathematical function that converts a vector of numbers into a vector of probabilities. It is given by $\sigma(\vec{x})_i = \frac{e^{\vec{x}_i}}{\sum_{j=1}^{K} e^{\vec{x}_j}}$., where σ is the SoftMax function, $\vec{x}_i$ is the input vector, $\vec{x}_j$ is the output vector and K is the number of classes in the multi-class classifier.

TanH: It is a widely used activation function in the field of neural networks and deep learning. It is given by $\tanh(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$. The TanH function outputs values in the range (−1,1), it is zero centered and has a sigmoidal shape. It is often used in hidden layers and preferred over ReLU and sigmoid function as it has a negative range.

End-End Latency time period: It is the time period taken from start to end of a process. In our context it is the time taken from collecting input data until establishing connection over a selected beam pair. [7], [8]

List of Abbreviations

V2X – Vehicle to Everything Communication

MEC – Mobile Edge Computing (Center)

ADAS – Advance Driver Assistance Systems

V2I – Vehicle to Infrastructure Communication

V2V – Vehicle to Vehicle Communication

Mm Wave – Millimeter Wave

LOS – Line of Sight

NLOS – Non-Line of Sight

RF – Radio Frequency

NR – New Radio

mm Wave - Millimeter Wave

ML – Machine Learning

DL – Deep Larning

F-DL – Fusion based Deep Learning

LiDAR – Light Detection and Ranging

GPS – Global Positioning System

DNN – Deep Neural Network

CNN – Convolutional Neural Network

RADAR – Radio Detection and Ranging

RF – Radio Frequency

UCB – Upper Confidence Bounds

MAB – Multi-Arm Bandit problems

Abstract

This study addresses the challenge of selecting millimeter Wave (mmWave) beamforming pairs for vehicle-to-infrastructure (V2I) communication, to mitigate latency in highly dynamic vehicular environments. We investigate the use of out-of-band sensor data as side information to model mmWave ray tracing paths and predicting a subset of top-K optimal beamforming pairs for efficient and low-latency searches. Unimodal-Fusion Deep Learning (F-DL) networks was applied to enhance mmWave beamforming process. We started by first investigating the centralized architecture, and then explored a novel distributed architecture through federated learning to minimize resource and latency overheads. The distributed architecture incorporates two biased client selection strategies: MaxLoss and heuristic Multi-Armed Bandit (MAB). This innovative approach streamlines beam selection, improving scalability, robustness and dynamic adaptability.

1 Introduction

With global automotive research and technology marching towards Level-5 autonomous driving, emerging automotive systems are equipped with various sensors. These sensors are used to model the physical environment, draw perceptions, and make decisions for driving safety. These sensors input the information required for safety-critical applications and acquire the instantaneous status of the surroundings. Additionally, achieving a completely autonomous driving environment requires the sharing and processing of vital data among various vehicles on the road with a mobile edge computing (MEC) center. This will enable the creation of driving instructions that can assist other vehicles in the environment, especially those unable to anticipate future changes in traffic conditions. The information sharing will also help optimize routes, energy usage, driving cycles, traffic grids and implementation of ADAS functionalities such as cooperative cruise control and collision avoidance systems [1] – [2].

The Vehicle-to-Infrastructure (collectively, V2I) communication paradigm provides technological framework to share information among vehicles and MECs. The size of these information is very large (4-10GB); and thus, a V2I communication system needs to have multi-Gbps transmission rates to transfer larger volumes of data in shorter time periods. With the need to achieve high-speed data transfer in the V2I communication networks, leveraging the advantages of millimeter waves (mmWave), proves to be an ideal candidate to be developed for V2I communications [3] – [4].

Millimeter Waves (mmWaves) are electromagnetic waves in the 1–10-millimetre wavelength range and frequencies ranging from 30 to 300 GHz. In practical applications, frequencies above 24 GHz are considered mmWaves, and the 57-70 GHz are vastly unused, which could be ideal for V2I communications. One significant impairment associated with using mmWave is its severe attenuation caused by obstacles in its path of propagation. Fundamentally, these attenuations are due to its higher interference with objects, given its larger wavelength in the millimeter range. To overcome these losses, mmWave communication deploys many directional antennas configured with different phase angles at both ends of the communication nodes, i.e. vehicles and MEC [5]. These antenna elements are termed as antenna codebooks which have many antenna elements at receivers and transmitters to massively increase the number of directional beamforming pairs. This arrangement eventually increases the possibility of attaining maximum coupling power at one particular beamforming pair between any two antenna elements placed at either ends of the nodes[6]. The optimal beamforming pair can be determined if all the beam tracing channel knowledges are available. As the nodes among vehicles are highly mobile and the environmental scenarios are highly dynamic, acquiring comprehensive beam tracing channel knowledge is largely impractical. Thus, leaving behind the only option to procedure an iterative search among all the combinations of antenna codebook beamforming pairs to return the one configuration with maximum coupling power.

The standardized procedure in IEEE802.11ad [7] and 5G New Radio (5G-NR) [8] to perform an exhaustive search by dividing the overall environmental space into sectors for

directional beams consumes a lot of time. The time to search across 360 beams (60 antenna codebook elements at MEC and six antenna codebook elements in the vehicle) takes almost 225 ms, adding a large delay to find the best beamforming pair. Moreover, as the vehicle is continuously in motion, such a latency period will never complete the initialization process of beamforming to establish a connection. For example, if the vehicle is moving at 70 miles/hour, within 225 ms (latency period) the vehicle would have moved by 4.5 meters requiring to re-initiate the standard beamforming search.

Although adopting multiple antenna elements at different phase angles to support directional beamforming is the best way to achieve mmWave communication beams with increased coupling pairs, the overhead incurred to determine the best beamforming pair is undesirable. This forms the primary motivation for this work, improving the tracking and predictability of the best beamforming pairs to establish V2I communication using mmWave technology. In this work, we perform a comprehensive study of out-of-band side information from vehicle onboarded sensors to evaluate ray tracing paths and select a smaller subset of "K" beamforming pairs. For instance, for K (=10), we can reach an accuracy of 97% with a reduction of 97% in end-end latency and for K (=20), we can achieve an accuracy of 99% with a reduction of 96.2% in end-to-end latency.

A very intuitive way to address this problem is to model the ray tracing paths of the Line of Sight (LOS) and Non-Line of Sight (NLOS) mmWave beams within empty environmental spaces [9] – [10]. Modelling the LOS ray tracing paths can be straightforward using the principles of trigonometry and geometry. However, modelling the NLOS ray tracing paths cannot be done using first principles of mathematics. The path of NLOS can be quite complex due to its vast number of available multiple paths through empty space as depicted in Figure 1. And thus, a more regressive predictive modelling is required to distinguish the most probable NLOS ray tracing path. The 3-D environmental space can be comprehended by processing the contextual information provided by sensor devices. Sensors such as Global Positioning System (GPS) can be very useful in determining the vehicle's location with reference to the MEC, while Light Detection and Ranging (LiDAR) sensors can provide 3-D point cloud data reflected by objects in its 360-degree surround space and detect potential obstacles in the ray tracing path. Images from camera sensors can also enhance object detection in the environment. These inputs from the sensors can be pre-processed and fed into a Fusion-based Deep Learning (F-DL) framework, which can output the probability of finding the global optimal beamforming pair. Based on the probability distribution for each pair, the top-K beamforming pairs can be selected to run the standard beam sweeping procedure defined in IEEE 802.11 ad and 5G-NR standards on the reduced set of “K” beamforming pairs. Figure 1 visualizes this approach towards using out-of-band sensor modalities and a F-DL model.

Figure 1 shows a framework developed in a centralized architecture. In this architecture, the sub-6 GHz communication channels are used to synchronize the vehicles and MEC and act as controlling lines while establishing connection. The vehicle collects instantaneous sensor data from LiDAR, GPS and camera and sends them to the MEC over the uplink sub-6GHz communication channels (as denoted by dotted lines). The

MEC turns these sensor values as inputs to the deep learning (DL) model and predicts the top-K beamforming pairs. The set of top-K beamforming pairs is later sent back to the vehicle over the downlink sub-6GHz communication channels (as denoted by solid lines). Finally, the vehicle and MEC run the standard beam sweeping procedure defined within IEEE.802.11 ad standard on the top-K beamforming subset.

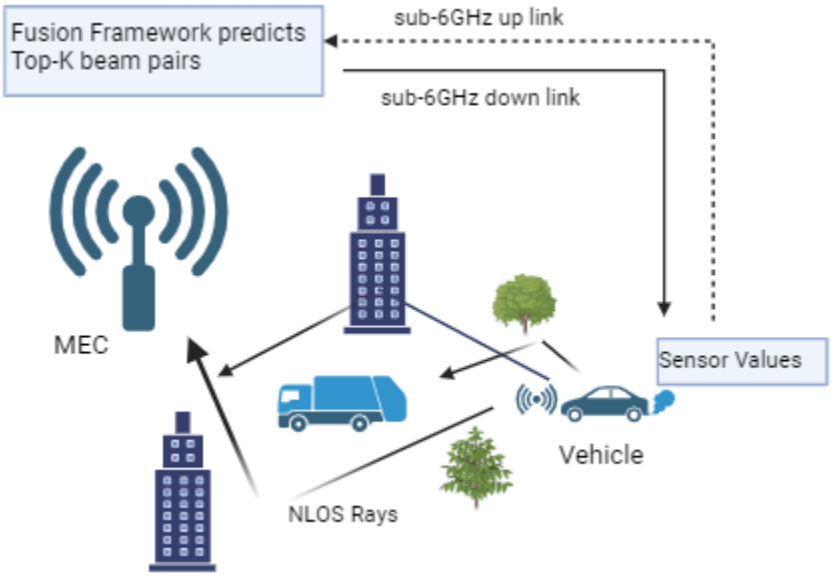

Figure 1: Visualization of usage of sensor modalities and F-DL architecture to predict top-K beamforming pairs.

The centralized architecture adds a critical bottleneck to the sub-6GHz channels by overly saturating them. Each time a vehicle needs to establish a connection with the MEC, it has to send a large volume of sensor data to the MEC to run the DL model and predict the top-K beamforming pairs. Furthermore, over time, the environment surrounding the MEC tends to change with a possibility of new structures coming up or increase in traffic density. These changes in the environment can be defined as “unseen scenarios” which can deteriorate the performance of the DL model as they have not been trained to accommodate the new setting. The model needs to be updated to improve the robustness to unseen scenarios. By using a centralized architecture, updating the model through in-field training to accommodate unseen scenarios post deployment is very difficult.

To address these two disadvantages of using a centralized architecture, we have defined a distributed architecture where a novel system-level framework is developed. The framework does not require sharing of sensor data with the MEC each time to predict the top-K beamforming pairs. Additionally, the framework addresses the implementation of in-field training of the DL model post deployment. The concept of federated learning is borrowed from the principles of ML and DL to use multiple users/clients to train and update a single global model. The weights of the global model are made available at the

vehicle itself to predict the top-K beamforming pairs. Each time, a connection with the MEC needs to be established, the MEC distributes the weights of its global model with the vehicle over the sub-6GHz communication channel. The distributed architecture designed in this work has the following contributions:

1. The data load on the secondary sub-6GHz communication channels are reduced by 50-70%, making the system more efficient and improving the need to upscale the system by increasing the total number of participating clients.
2. This approach opens the door for in-field training after deploying the global model at the MEC. This improves the system's adaptability to the changing environment around the MEC over time.

Summary

- The intuitive model of the system discussed above forms the primary approach to finding a solution to improve the communication overheads and reduce latency for adopting mmWave for V2I communication systems. In the following chapters, the details of the implementation and results are studied. The report is organized in the following manner:
- Chapter 2 summarizes a few preliminary concepts of using classical beam selection and formation, standardized procedure for beam sweeping discussed in the IEE802.11 ad, sensor modalities, deep learning networks and centralized and distributed models.
- Chapter 3 summarizes the literature review carried out towards improving mmWave beamforming overheads in a ramified manner. The chapter covers studies on the deep learning solution for improving beamforming. The literature review of the federated learning paradigm (distributed models) and different client biased selection strategies for federated learning are discussed. Finally, a survey is conducted on the available datasets used to train and validate DL models for predicting top-K beamforming pairs using multimodalities.
- Chapter 4 summarizes the details of implementing the F-DL networks for centralized and distributed architecture. The development of F-DL model is first introduced in a centralized architecture and then reused in a federated learning framework to realize distributed architecture. Two client-biased selection strategies for federated learning framework is discussed, and the overall system architecture is discussed.
- Chapter 5 presents and the simulation results. Along with this, the qualitative inferences (by analyzing the results) are discussed.

2 Background

The previous chapter introduced the advantages of using mmWave for the V2I communication systems. Although mmWave proves powerful given their high-speed communication links and vastly unused spectrum by any other application, they have shortcomings due to severe attenuation caused by obstacles in their path of beams due to larger wavelengths. An effective way to tackle this problem is to use multiple beamforming antenna elements at the receiver and transmitter nodes, massively increasing the number of directional beams and improving the probability of coupling between receivers and transmitters. The latencies incurred by using the standard beamforming processes for such a large number of massive directional beams are undesirable. An effective approach is to predict the top-K beamforming pairs using out-of-band side information and run the standard beamforming process on this smaller subset of top-K beamforming pairs.

To begin the derivation, we first discuss the background of classical approach to formulating multiple antenna codebooks and beamforming processes. These concepts are used to study the benefits of using top-K beamforming pairs as an added advantage when using the standard beam sweeping strategy defined in IEEE 802.11 ad and NR standard. The beam sweeping latency incurred during the standard procedure and its improvement using top-K beamforming pairs are further discussed. We then discuss the nature of side information provided by sensor modalities which can aid beamforming processes. This is followed by an introduction to the commonly used architecture of fusion based deep learning models. Finally, the definition of sub-6GHz channels is discussed along with its use case in a distributed and centralized architecture. These concepts will essentially serve as a background study towards familiarizing the discussions in the following chapters.

2.1 Classical Beam Selection and Formulation

Let us first introduce the concepts of configuration of analog antenna codebook elements for mmWave as also previously defined in [7], [9] and [10]. The analog antenna codebook contains the total number of analog elements at both the receiver and transmitter. Each element in the codebook has a different phase resulting in directional beams where each element focuses the rays to be sent or received along a particular three-dimensional coordinate in space. The codebook elements can be defined by equation (2.1).

$$C_t = \{ t_1, \ldots..., t_M \}, \quad C_r = \{ r_1, \ldots..., r_N \} \tag{2.1}$$

Where C_t is the codebook defined at transmitter with a total number of M antenna elements. C_r is the codebook defined at receiver with a total number of N antenna elements. From equation (2.1), the total combinations of antenna pairs that can be formed between the transmitter and receiver is given by equation (2.2).

$$\beta = \{ (t_m, r_n) \} \mid t_m \in C_t, r_n \in C_r \tag{2.2}$$

Where β defines the set of total number of beamforming pairs/combinations that can be formed. The size of β is M×N represented by |β| . For each of the element in the set β, the normalized coupling power for a combinations of codebook elements are defined by equation (2.3).

$$y (t_m , r_n) = | w_{tm}^{h} H w_{rn} |^{2} \quad (2.3)$$

where $H \in R^{M \times N}$ is the channel matrix and 'h' are the conjugate transpose operator. The weights w_{tm} and w_{rn} indicate the corresponding beam weight vectors associated with the codebook element t_m and r_n. Using equation (2.3), the fundamental idea is to find the combination of t_m and r_n that maximizes the normalized coupling power. The one configuration that maximizes the value of y in equation (2.3) is the optimum beamforming pair that we need and is given by equation (2.4).

$$(t^* , r^*) = \text{argmax} [y (t_m , r_n)] \quad (2.4)$$

Where $1 \leq m \leq M$ *and* $1 \leq n \leq N$.

2.2 Top "K" Subset Selection

The traditional exhaustive search methods typically take about 10 ms for IEEE80.2.11 ad [7] and 5 ms for 5G-NR [8] for a beam search among 30 beamforming pairs, respectively. However, many more codebook elements form larger sets of beamforming pairs. To draw a reference, from the Raymobtime S008 and S009 dataset (introduced in Chapter 3), there are 32 codebook elements at the MEC and eight codebook elements on the vehicle, which gives 256 pairs, and conducting an exhaustive search will add much latency which is undesirable in case of mobile nodes. This motivates us to use other relevant information to bring down the total number of pairs to a smaller subset of pairs with the highest probability of finding the best pair within this subset, thus reducing the search time. The use of multiple sensor modalities to gather out-of-band information coupled with DL models can predict a smaller subsets of optimal beam pairs.

Specifically, the key algorithmic model can be derived using equation (2.2) in identifying a subset $\beta_K \subseteq \beta$ of "K" beamforming pairs such that the subset of β_K contains the top-K beamforming pairs with the highest probability. This can be mathematically represented as:

$$\beta_K = \text{argmax } P ((t^*, r^*) \in A) \quad (2.5)$$

Where $A \subseteq \beta$ and $|A| = K$.

Once β_K is found, the standard search can be performed to find the best beamforming pair on this subset. Where 'P' represents the probability density calculated by the final layer of the DL model for each beamforming pair. Larger values of the probability densities correspond to the optimal beamforming pairs which are more likely to be the ones with maximum coupling power.

2.3 Beam Sweeping Latency for NR standard

This section introduces the standard beam sweeping strategy defined in the IEEE802.11ad and NR protocol standard. As discussed in section 2.1, for a codebook size of 'M' at the transmitter and 'N' at the receiver, the total number of beam pairs is given by $|\beta|$ (= M×N combinations). During the initialization access phase to establish a connection, the receiver and transmitter nodes exchange messages to identify the optimal beam pair. Each node transmits synchronization signals (SS) containing information related to each codebook element until all elements are exhausted. An SS burst constitutes multiple synchronized signals transmitted at different combinations within the set of $|\beta|$. The NR standard specifies a fixed duration of one SS burst (T_{ssb}) at 5 ms, transmitted periodically (T_p) at every 20 ms [9]. A maximum of 32 such synchronized signals can fit within one SS burst, allowing for exploring 32 pairs in each burst. Therefore, to explore all the beamforming pairs, the total exploration time can be calculated as follows:

$$T_{NR}(|\beta|) = T_p \times \lfloor (|\beta| - 1)/32 \rfloor + T_{ssb} \quad (2.6)$$

Where $T_{NR}(|\beta|)$ is the latency period to sweep through all the beam pairs according to the standard protocol. For selecting from $K \ll |\beta|$, which reduces the subset to only top-K beam pairs, the relationship in equation (2.6) is modified. Firstly, NR standard assumes that a pair of 32 beams can be swept in 5 ms (T_{ssb}). Thus, to sweep through one beam pair it takes approximately T_b => 5/32 = 156 ns. Finally, the required time to sweep though top-K beam pairs is given as:

$$T_{sweep}(K) = T_p \times \lfloor (K - 1)/32 \rfloor + T_b(1 + (K\text{-}1) \bmod 32) \quad (2.7)$$

2.4 Sensor Modalities to aid Beam Selection Process

Given the directional requirements inherent in mmWave communication, beam selection involves identifying the receiver's location from the transmitter based on the most substantial gain, whether in LOS or NLOS scenarios. Therefore, determining the positions of the transmitter, receiver, and potential obstacles is crucial for beamforming initialization. In LOS condition, the ray travels directly from the transmitter to the receiver without any reflections. In contrast, for NLOS scenarios, the ray undergoes multiple reflections and attenuation due to surrounding obstacles. While measuring LOS paths is relatively straightforward using trigonometric and 3-D geometric principles, estimating NLOS paths is more challenging. Quantitative estimation of available open space in the physical environment can model NLOS transmission paths effectively. Incorporating onboard sensor devices, such as Global Positioning System (GPS), Light Detection and Ranging (LiDAR) and camera images is capable of detecting free space in the environment, crucial for assessing potential obstacles and environmental conditions.

Figure 2 illustrates this scenario with a moving vehicle and a MEC system endeavoring to identify the optimal beam pair amidst multiple reflections and potential obstacles. The physical environment's status surrounding the system (path of ray) is captured by the multimodality sensors, including the GPS and LiDAR on the vehicle and cameras at the

MEC/vehicle. These sensors operate synchronously, providing 3-D information about the surroundings at regular intervals. The data from the sensors are fused to establish an empirical relationship, suggesting the top-K beam pairs and accelerating beam selection using a DL model. Subsequently, using the synchronous signals, the standard beam sweeping operation only on the subset of β_K is performed as detailed in section 2.3.

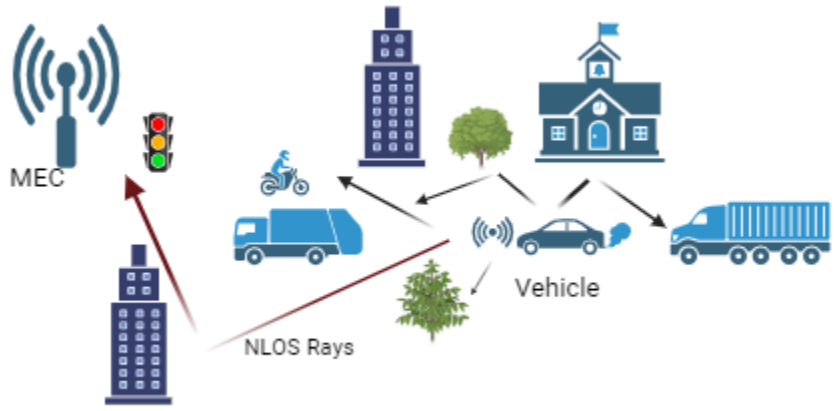

Figure 2: NLOS phenomenon caused by multiple reflections and potential obstacles

2.5 Multimodal Sensor Data Preprocessing

This section highlights the type of information available from different sensors as well as the necessary preprocessing steps that needs to be done to reduce the usage of sub-6GHz channels and improve training performances of the DL model. GPS generates the latitude and longitudinal position of the vehicle trying to establish a connection with the MEC. This information helps to draw the exact reference of the vehicle's position concerning the MEC. The LiDAR forms a 3-D point cloud representation of the environment which are generated by object reflections of emitted light beams calculated based on the time interval between reflection and transmission. The camera captures static RGB images of the environment, providing insights into dynamic situational scenarios and insights into the object's nature. The LiDAR and the camera can work simultaneously and enhance each other's embeddings by superposition of detected objects thus improving the prediction for the NLOS scenarios.

GPS inputs are pre-processed by a straightforward normalization factor to fall in the [0,1] range. The LiDAR inputs take a few more preprocessing steps due to its larger computational load associated with directly using raw point cloud information. The LiDAR point cloud data comprises spatial 3-D point cloud coordinates (x, y, z) corresponding to detected objects in the environment. One effective preprocessing technique for the LiDAR data points is to employ a 3-D histogram [12]. The 3-D histogram consists of three components along each axis with resolutions (bx, by, bz). Its coverage or range encompasses the space enclosing the vehicle and the MEC, which can be predefined with a calibrated distance threshold. This coverage is represented by (Xmin, Xmax), (Ymin, Ymax), and (Zmin, Zmax). Subsequently, the bins of histogram

are populated with values based on the information obtained from the point cloud along the three axes. The bin numbers are mapped based on the coordinates and resolution. Since the MEC's position is fixed, it corresponds to an indicator value of (-2) along all three axes in the histogram. Any detected object in space is mapped to a value of 1. This process results in a compressed form of sharing the data without losing the originality of the LiDAR data.

The camera images also need to be preprocessed using a more complex image processing techniques to detect multiple vehicle types, separate obstacles and free space in the region around the MEC and remove static background from a larger source of image pixel data. The details of camera image processing are not studied in this work, we used the results from [9] and will be one of the future scopes for study. As we will see in section 5.1 the benchmarked results using GPS-LiDAR fusion network without incorporating camera images performs on-par with three-senor fusion network. Moreover, the resource overheads for camera image processing are higher and developing a proof of concept incorporating just LiDAR and GPS can reduce processing overheads significantly.

2.6 Fusion Based Deep Learning on Multimodal Sensor Data

Fusion based deep learning (F-DL) is a multi-step deep learning process where the overall deep learning network is formed by stacking multiple deep learning models in a series-parallel configuration using multiple modalities as inputs. The DL models connected in parallel for each respective input modality can be termed as unimodal and a final concatenation of the outputs of these parallel unimodal in a series configuration can be referred to as a fusion model as shown in Figure 3. Each DL model is composed of multiple layers of parameterized filter kernels and activation function that first characterizes instantaneous preprocessed sensor inputs to a high-level feature set using the unimodal networks. The final fusion network associates the high-level feature set to a corresponding probability density for all the beamforming pairs.

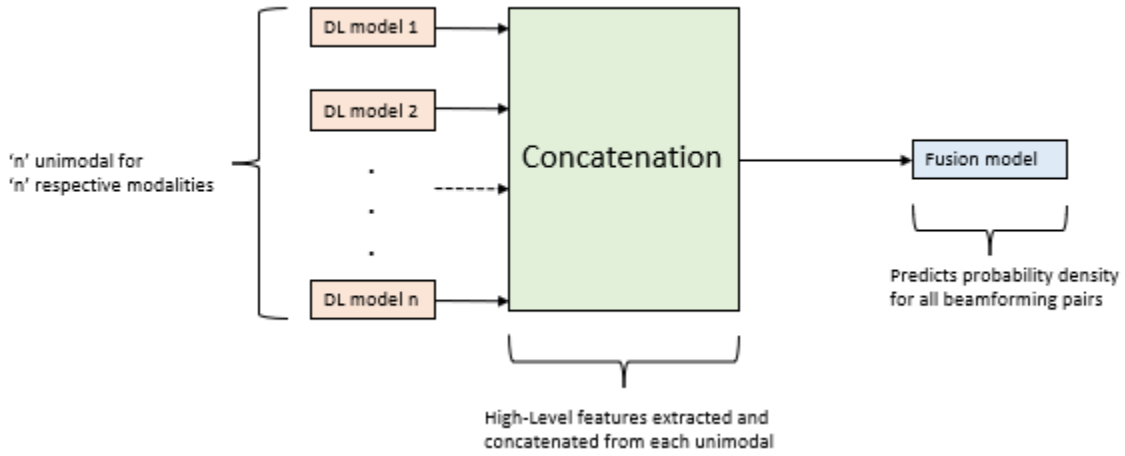

Figure 3: Architecture of Fusion based Deep Learning Model

The F-DL leverages the preprocessed sensor data to output probability density values for all the beamforming pairs (β), which will help determine a subset of top-K beamforming pairs (β_K) based on the set of highest values of the probability densities. Subsequently, this allows only a subset of the top-K beamforming pair to be searched in the standard beam sweeping operation, potentially including the most optimum beamforming pair (ground truth corresponding to maximum coupling power). Meanwhile, the preprocessing step previously discussed ensures that the neural network operates only on relevant and compressed data, enhancing its predictive capabilities and leading to better global convergence during training. In this way, F-DL provides a gateway for successfully predicting the NLOS cases, which cannot be directly deduced using mathematical principles. It provides a relationship between the NLOS ray tracing path and input sensor data, which is highly impossible to find using the first principles of mathematics as well as canvasing for more straightforward LOS scenarios during its training.

2.7 Centralized and Distributed Architecture

In the previous sections, we introduced sub-6GHz channels, which are channels defined in IEEE802.11p and do not operate in the mmWave range. However, they are vital in sharing control information between vehicles and MEC during the initialization phase of to establish mmWave connections. These channels are assumed to be fully functional and form a secondary protocol alongside mmWaves to establish V2I communications. Uplink communication is initiated at the vehicle, and downlink communication is initiated at the MEC.

In the centralized architecture, all the processes and computations to establish the mmWave communication link are centralized at the MEC. The data collected by the vehicles from the sensors are preprocessed and shared with the MEC over the sub-6GHz uplink channels. The MEC feeds these data as inputs during training and executes the learning of the F-DL model. Once the F-DL model is fully trained, the MEC predicts the top-K beamforming pairs and sends these predicted configurations back to the vehicle over the sub-6GHz downlink channels. From this point, the “vehicles” are referenced to the special vehicles participating in training the F-DL model for improving beamforming processes otherwise stated. These vehicles are different from other common vehicles on the road which will just use the trained F-DL model post deployment to predict the top-K beamforming pairs using the real-time values from the on-boarded sensor.

Unlike the centralized architecture, in the distributed architecture scheme (Figure 4), the vehicles do not share the data with the MEC, instead, the vehicles themselves train and run the F-DL model to predict the best beamforming pair. The information regarding the codebook in the MEC is assumed already available in the vehicle. During training, the parameters for the F-DL are trained at multiple vehicles and are shared with the MEC over the sub-6GHz uplink channels to update the global model at the MEC. The MEC orchestrates the arrival of these parameters from each vehicle and aggregates them and sends the updated parameters back to the vehicles using the sub-6Hz downlink channel for further training. One such process concludes one training round, and the training takes place for multiple rounds. The parameters updated at the MEC at each round are termed

global parameters, and the parameters at the vehicle are termed local parameters. The training ground is stopped when the global model converges. The distributed architecture uses a federated learning paradigm, which gets its etymology from applied machine learning terminologies.

Both these architectures have their advantages and disadvantages. The centralized learning method does not require higher computational processing at the vehicles since a dedicated processor is not necessary to carry out the training. The downside is that they significantly saturate the sub-6GHz channels with significant amount of data sharing. The distributed architecture, on the other hand, requires dedicated processors in the vehicles. However, they also significantly enhance the robustness of the F-DL network. Moreover, distributed learning provides a framework to carry out in-field training while the system is already deployed. This can cope with future environmental variations improving robustness with stochastic developments.

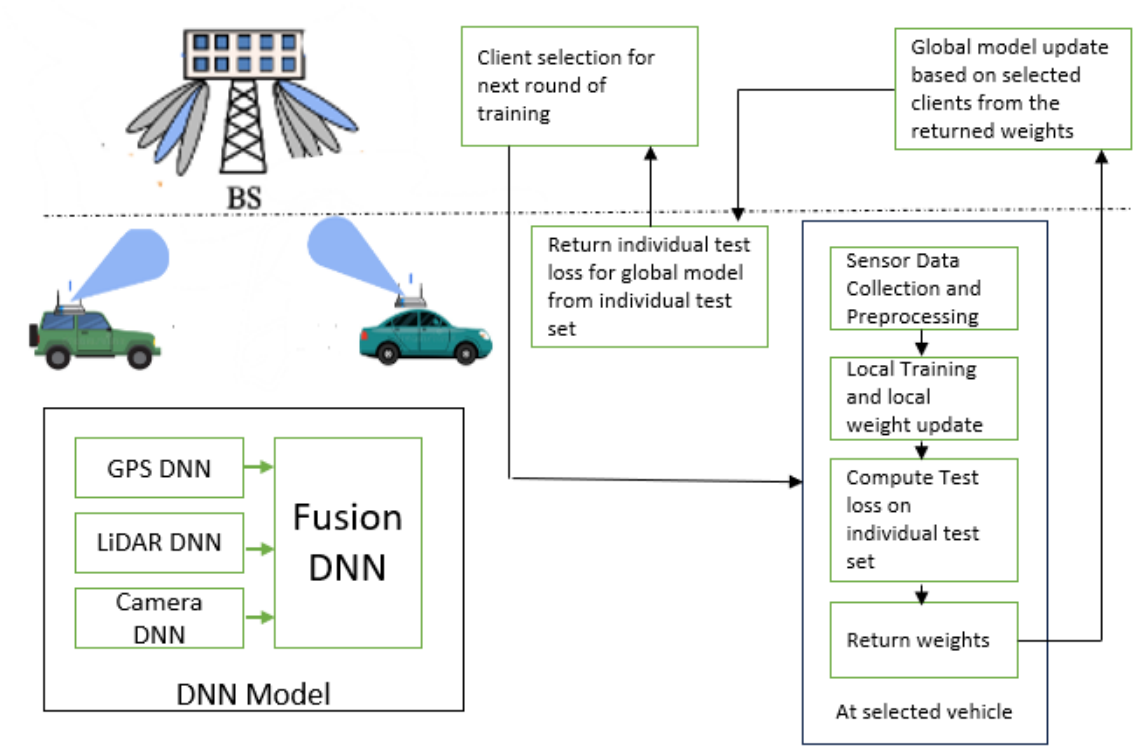

Figure 4: Distributed Architecture incorporating federated learning paradigm

In the following chapters, both centralized and distributed architecture are discussed separately. Chapter 3 introduces the background literature for both the architectures and the available data sets for both. Chapter 4 first discusses the framework of the F-DL model in a centralized architecture which is then reused in a federated learning paradigm to realize a distributed architecture. Chapter 5 discusses all the results for centralized and distributed architecture and qualitatively presents the advantages based on the results obtained.

2.8 Performance Metrics

We define the performance metrics that are used to evaluate the model in the section below

Top-K Accuracy: It is calculated by determining the percentage of times the model accurately predicts the correct outcome among the top-K probabilities. Specifically, considering the ground-truth beam pair (t*, r*) and the model's predictions of top-K beamforming pair is defined as:

$$\text{Accuracy} = \frac{1}{N} * \sum_{j=1}^{N} 1 \text{ iff } (t*, r*) \epsilon A' \quad (2.8)$$

Where the summation of '1s' are done for every test sample if the ground truth optimum beam pair is found in A' which are the top-K optimum beam forming pairs. N is the number of test samples. J is the iterator over all the test samples.

Validation Accuracy: It refers to the accuracy of a ML/DL model on a set of data that was not used during training. It is a metric used to evaluate how well the model is able to generalize to new, unseen data during training. It also provides an insight on if the model is improving on each epoch of training. An epoch is defined as one cycle of iteration for training a ML/DL model using all the training data.

Test Accuracy: Testing accuracy refers to the accuracy of a ML/DL model on a set of data that was not used during training or validation. It is a metric used to evaluate how well the model can perform on completely new and unseen data.

Throughput Ratio: It illustrates the ratio of the average throughput when sweeping only 'K' beam pairs predicted by the model compared to what could be achieved with exhaustive search.:

$$R_T = \frac{1}{N} \sum_{n=1}^{Nt} \frac{log2[1+y(\widehat{t*,r*})(n)]}{log2[1+y(t*,r*)(n)]} \quad (2.9)$$

Where (t*, r*) and $(\widehat{t*, r*})$ show the best beam pair in β and β_K respectively. N is the total number of test samples.

Global Test Loss: It depicts the convergence of parameters in the global model following a federated learning paradigm. It is an important measure which helps to determine the degree of training at the global model.

3 Literature Review

Beam selection strategies can be historically ramified, as shown in Figure 5. Traditional methods include hierarchical beam searching schemes incorporating approximations, interpolation, important sampling and other algorithmic schemes such as binary tree searches and uses beamforming channel information. For example, Yang et al [13], have focused on using hierarchical searching scheme to primitively study wider sectors of beamforming using lesser number of codebook elements and the narrowing down on the area of these sectors where beamforming coupling powers are maximum. Wang et al. [14], proposed to use the angle of departure of rays originating towards servers and within a geometric range and perform beam searches only on these beams. Although, these strategies have provided a pathway towards improving beamforming processes, they have not been robust to be adopted to a practical setup of having larger number of mobile nodes. These techniques that do not use any ML methods nor any other side information can be classified as In-Band Traditional methods.

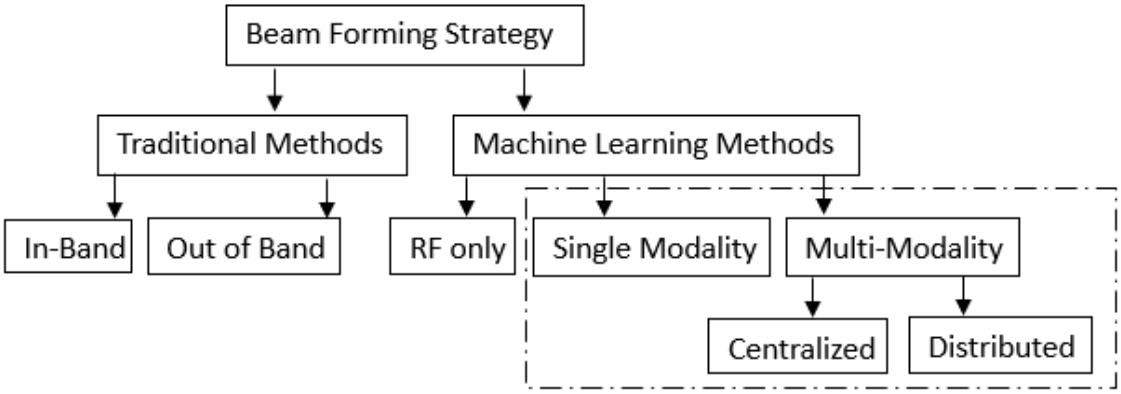

Figure 5: Beamforming Strategies

The use of side information outside the scope of classical beamforming approaches where only directional beams are analyzed gained popularity as they can provide more channel information to improve beamforming processes. T. Nitsche et.al [15],proposed “Steering with eyes closed” methodology where the legacy 2.4/5GHz band networks’ omni-directional beams whose channel knowledge is well known can be inferred to approximate mmWave LOS directions. Gonzalez-Prelcic et al. [16], proposed to use a compressive covariance estimation approach to map between RADAR ray tracing paths as side-information and mmWave ray tracing paths. These methods do not use ML methods but find relationships between side information provided and mmWave ray tracing. They can be classified as Out of Band Traditional methods.

Predicting the paths of mmWave ray tracing in environments where communicating nodes (receivers and transmitters) are moving involves understanding complex, non-linear relationships. Incorporating ML methods can address this challenge by using a framework based on artificial neural networks. This framework starts as a "black-box" setup, which essentially means its internal workings are not initially designed to be

understood or interpreted. It is first set up with a basic structure or "blueprint" of artificial neural networks, which are then trained to model these complex relationships effectively. He et al. in their research study [17] innovatively regarded all the possible beamforming pairs as a 2D noisy channel matrix. A denoising-based approximate message passing neural networks was trained and applied on the channel matrix to filter out the optimum beamforming pairs.

We focus on ML/DL-based techniques to use multiple non-RF modality sensors, while conventional beamforming techniques often rely on RF signals to adjust the direction of the antenna beams at the transmitter and receiver. However, the latter approach may not support simultaneous beamforming at both ends. Non-RF out of band beam selection, on the other hand, uses data from different sensors to make joint decisions about the best beamforming pair. This approach leverages information from sensors to input a DL to predict ray tracing paths and optimize the beamforming process.

3.1 Centralized Architecture

Focusing on incorporating centralized architecture-based deep learning techniques to maximize the sensor information to yield environmental embeddings and predict the top-K beamforming pairs, Klautau et al. [11] and Dias et al. [12], were the first to incorporate GPS and LiDAR data to leverage LOS path detection in mmWave communication. These two research studies have good architectural design and comprehensive results, forming a good base for comparative benchmarking the improvements to include NLOS path detection in our studies. Batool Salehi et al. [9], proposed the first of a kind fusion based deep learning model using GPS, LiDAR and camera modalities to predict top-K beamforming pairs. The design of the “unity-deep learning layers,” which is a combination of 2-D CNN, MaxPool and addition layers, is presented. This approach critically helps in the high-level feature extraction from raw LiDAR and camera data. The high-level extracted features from the respective sensor modalities using unimodal networks are concatenated to be fed into the fusion network providing an overall pipeline to develop a fusion based deep learning architecture based on multiple modalities. In their work they validate the model using the Raymobtime S008 and S009 dataset as well as additionally present details of their homegrown NEU dataset. They also present a detailed methodology of data pre-processing for LiDAR and camera datasets which was discussed in the previous chapter. The ground truth optimum beamforming pair with maximum coupling power is made one and the rest of the pairs to zero a strategy is termed as “One-Hot Strategy” is used to define the ground truth labels for multiclass classification. Finally, the work also illustrates the use of top-K beamforming pairs and restricting them in the standard NR search space. In our work, the F-DL model is used to benchmark the results to use out-of-band side information from sensor modalities (GPS, LiDAR and camera) to predict the top-K beamforming pairs using Raymobtime dataset. Moreover, we have also been able to produce on-par accuracies just by using GPS-LiDAR fusion network reducing resource overheads for camera image processing.

Mateo Zecchin et.al [10], outline the use of only GPS and LiDAR data leveraging Raymobtime dataset using a parameterization of DL model. However, a critical

distinguishing aspect of this study lies in utilizing knowledge distillation (KD) techniques, proposing a novel loss function. This unique loss function notably enhances beam selection for smaller values of "K", resulting in higher throughput. Moreover, the proposed algorithm incorporates a non-local attention scheme, notably enhancing beam classification accuracy, particularly for NLOS cases. Another noteworthy contribution of this research is the introduction of a curriculum training strategy, accelerating the convergence speed and rapidly improving training accuracy. The rationale behind this curriculum learning strategy is acknowledging that LOS paths are more straightforward to visualize. In contrast, NLOS path samples present significant challenges that cannot be easily modelled mathematically. The proposed curriculum learning strategy facilitates faster fine-tuning of weights towards convergence of the loss function by commencing training with LOS samples and gradually introducing NLOS samples as the initial weights are appropriately set. This study forms another good reference to benchmark our results.

3.2 Distributed Architecture

Distributed architecture can reduce the saturation of sub-6GHz communication channels and improve overall robustness. Thereby, focusing on the direction to incorporate federated learning techniques to predict the top-K beamforming pairs, the contribution of Batool et al. in their study [18] use a distributed architecture. A similar F-DL architecture akin to the one presented in [9] is used in this work, but in a distributed architecture setup leveraging the use of GPS, camera, and LiDAR modalities. A detailed analysis is provided on the reduced utilization of sub-6GHz secondary communication channels compared to the centralized architecture presented in [9]. In the presented multimodal federated learning framework, the local parameters trained at each client (vehicle) are globally averaged at the server (MEC) after each training round. The federated learning framework presented in [18] only supports an unbiased client selection strategy with variations of selecting different unimodal at each training round to update only the selected unimodal-network based on a fixed probability distribution. This strategy aims to reduce the use of sub-6GHz channels. However, the adoption of this global update strategy has the downside of slowing down convergence, thereby extending the duration of the training process. Our contributions show the use of biased client selection strategy to holistically update the entire unimodal and fusion network after each training round without consuming a lot of space in the sub-6GHz channels and also improving generalizability to achieve optimum convergence. Furthermore, this research work also presents the contribution of the first-known dataset for distributed architecture [19].

3.3 Federated Learning Client Selection Strategies

Unbiased Client Selection Strategy: The identification of training using distributed architecture for improving mmWave beamforming is acknowledged as an essential research direction first proposed by Batool et.al in [18]. A straightforward and practical algorithm used for this setting employs a simple pathway to aggregate model weights from all the clients at the end of each training round. More concretely, the “Federated Averaging” proposed by B. McMahan et.al [20] in their analytical derivation to average out all the parameters aggregated from every client to prove convergence of the global

model laid a foundation to use federated learning. As all the clients involved in training process at each training round, this strategy is termed as unbiased client selection strategy.

Biased Client Selection Strategy based on Max-Loss: B. McMahan et.al [20] formulated the gateway to mathematically comprehend federated learning after substantial results were presented out in predictive supervised learning problems using their "Federated Averaging" algorithm. The research by Yae Jee Cho et al. [21] further these studies by using the "Power of Choice" methodology, where the communication overheads between clients and the server are reduced by orchestrating the selection of only a small subset of clients based on their local test losses. The studies establish the "Power of Choice" methodology, where only a subset of selected clients participate in training at each training round. The clients are selected based on the maximum test loss returned by each client on the recently updated global model during the previous training round. They prove that selecting a subset of clients with maximum test losses at each round canvases the learning for all the other unselected clients. The incorporation of the "Power of Choice" methodology to realize biased client selection strategy in our research work to realize the novel distributed architecture reduces the overheads on the sub-6GHz channels. It also improves the rate of convergence compared to the fixed probability-based sampling criteria for selecting unimodal networks to be updated globally proposed by Batool et al. in [18]. Although, the "Power of Choice" methodology captures the ability of the subset of selected clients with maximum test losses to canvas for the unselected clients, there is a possibility of small biases being incorporated causing higher error floor in the convergence of the model.

Upper Confidence Bounds-based multi-arm bandit formulation for biased client selection: Leveraging maximum test loss to select a subset of clients at the end of each training round can bring a bottleneck when few other clients are entirely ignored raising a question on the fairness of client selection process. This causes specific biases to be unnecessarily incorporated into the global model diminishing its ability to generalize. At the verge of the convergence of the global model's gradient descent at the global minimum, it will oscillate the gradient descent about the global minimum unable to converge at the optimum minimum. Yae Jee et al. [22] address this issue by considering a fairness factor, where the losses from unselected clients after each training round are heuristically accumulated for each client. When these heuristically accumulated losses become more significant than the current maximum test loss in time, the particular client is forced to be selected for training. The heuristic accumulation of losses is based on Upper Confidence Bounds (UCB) algorithm, which is used in Multi-Arm Bandit (MAB) problems or Immediate Reinforcement Learning [23]. The discounted UCB index is calculated for each client at the end of each training round. This index combines two factors, the frequency of each client's participation in training and the accumulation of each clients' test losses from the past training rounds. The subset of clients with the largest values of discounted UCB index are selected to participate in the next round of training. The authors in [22] discuss this process of client selection strategy as a MAB problem wherein the decision to select a client is determined by the probability of

maximizing the reward upon selecting them with an end goal to achieve convergence of the global model with a low error floor. This methodology is termed as “Upper Confidence Bounds based Client Selection strategy” or UCB-CS. The incorporation of UCB-CS methodology to realize biased client selection strategy in our research work improves the robustness of the global model alongside reducing the overheads on the sub-6GHz channels. The orchestration process of biased client selection in our novel distributed framework using UCB-CS allows in-field training post deployment. This improves the dynamic adaptability of the model to changing environment and minimizes the cost of updates to update the global model.

3.4 Multimodality Datasets for Beamforming

Leveraging out-of-band information to enhance the selection of optimal beamforming pairs has been a significant focus among researchers. However, this approach comes with associated costs and logistical challenges in conducting experiments to gather sensor data. The Raymobtime simulation dataset [24] is a widely utilized and comprehensive resource. It offers high-fidelity virtual captures tailored for vehicle-to-everything (V2X) deployment and the development of innovative solutions. Notably, the S008 and S009 datasets within Raymobtime provide a combination of LiDAR, GPS, and camera data, along with ground truth beamforming gains. These datasets compromise 256 beamforming pairs with 32 codebook elements at the MEC and eight at the vehicle, serving as valuable assets for researchers exploring advanced beamforming techniques [9], [10], [11] and [12].

Table 1: Raymobtime Dataset

Dataset	**Number of Samples**	**LOS**	**NLOS**	**NLOS Percentage**	**Beam pairs**
S008	11194	6482	4712	42%	256
S009	9638	8165	8165	85%	256

The dataset from S008 is utilized for training and validation purposes, whereas samples from S009 are reserved for testing. The datasets exhibit a high degree of unbalance, with S009 containing 85% of NLOS samples in urban road scenarios, as summarized in Table 1. Therefore, achieving accurate results on the S009 dataset approximates real-world use cases. The simulation setup for the Raymobtime S008 and S009 datasets is detailed in [25]. To shed more light on the data collection process, a stationary base station (BS) was installed at a height of 4 meters, adjacent to the path of moving buses, cars, and trucks. Traffic flow was simulated using the Simulator for Urban Mobility (SUMO). Data from image and LiDAR sensors were gathered through the use of Blender, a 3-D computer graphics software toolkit, and Blender Sensor Simulation (BlenSor) software. In this setup, for every moment captured (referred to as an episode or sample), the system selects a single active receiver from all the vehicles in motion. The designated receiver vehicle then acquires data from the LiDAR point cloud and GPS positioning. Simultaneously, the camera installed at the BS captures an image. The quality of the communication channel, considering various beam pairings, is determined through the application of Remcom’s Wireless Insight ray-tracing software. Episodes are spaced 30 seconds apart, resulting in a collection rate of 2 samples per minute. Each episode is centered around identifying the best beamforming pair, which is the combination of signals between a moving vehicle

and a stationary base station (BS) that results in the highest signal strength. This optimal pair is determined through a standard process known as beam sweeping. Alongside identifying this beamforming pair, the system also records real-time data from LiDAR, cameras, and GPS devices. This data is timestamped to match the moment the optimal beamforming pair is identified, providing side information from these sensors to accompany the ground truth beamforming data.

These datasets feature situational scenarios encompassing multiple reflections of rays within a sophisticated environment of dense urban canyon, making them effective for testing ML models, particularly in practical NLOS cases as shown in Figure 6. Consequently, these datasets serve as benchmarks for conducting experiments, and improvements in accuracy and can translate well into real-world scenarios.

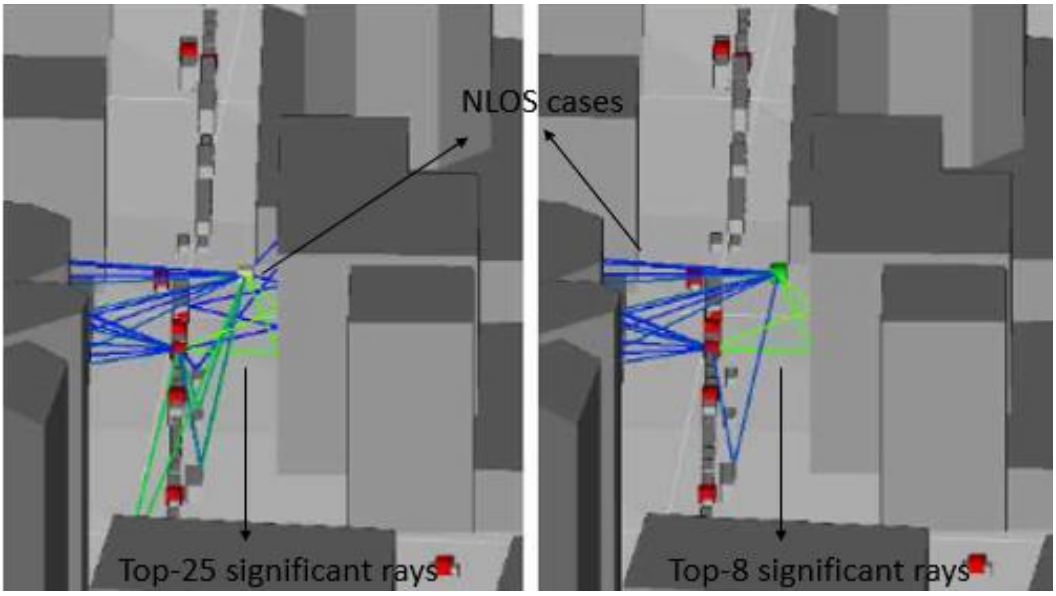

Figure 6: An environmental situation to record Top-K NLOS rays in [25] (Left images shows top-25 rays. Right image shows top-8 rays)

The Real-World NEU dataset [26] offers another valuable resource for validating ML-based multi-sensor out-of-band models for assessing beamforming pairs. The dataset depicts an outdoor urban road environment with two-way traffic, flanked by towering buildings ideally encapsulating NLOS situations. Unlike Raymobtime dataset which uses a virtual simulation setup, this dataset is collected in real-time with live sensors installed on vehicles.

The Infocom FLASH dataset also contains GPS, LiDAR, camera and ground truth values of RF beamforming pairs [19]. The FLASH dataset is well suited for training and updating DL model for distributed architectures. This dataset was assembled using real-time data collection with live sensors installed at ten clients or vehicles, each documenting all the scenarios as detailed in Table 2. From Table 2, it is inferred that the categories are designed to progressively incorporate NLOS scenarios in the setting. The

dataset represents beamforming pairs through sectors. Moreover, the dataset is collected in an urban environment which significantly helps to evaluate and benchmark the results to also work well post deployment in a physical scenario.

Table 2: FLASH Dataset Description sourced from [19]

Category	**Speed (mph)**	**Featuring**	**Scenarios**	**Number of samples**
1	10,15,20	NA		9729
2	15	Pedestrian	standing, walk right to left, walk left to right, walk back to front, walk front to back	7968
3	15,20	Static Car	On right, on left, Infront	8174
4	15,20	Moving Car	10mph same lane, 20mph same lane, 10mph opposite lane, 20mph opposite lane	6052

4 System Architecture

In Chapter 2 we introduced the background study to familiarize the derivation and usefulness of restricting beamforming process to a subset of top-K beamforming pairs. Using the classical approach, we modelled the selection of beamforming pairs based on maximum coupling powers between receivers and transmitters of the antenna codebook elements. We also discussed the influence of side-information form senor modalities to determine the ray tracing paths and predict the top-K beamforming pairs using F-DL models. Finally, the role of sub-6GHz communication channels as control line to establish mmWave communication was discussed in a centralized and distributed architecture setup.

In this chapter, we use these background studies along with the knowledge distilled from the literature survey to develop the overall system architecture. We first derive the relationship of the probability densities of the predictions returned by the DL model with the probability densities of the top-K beamforming pairs. We then present the design and development of the centralized architecture for predicting top-K beamforming pairs using the framework illustrated by Batool et al. in [9]. Finally, we design and develop the framework for the novel distributed architecture incorporating the orchestration of biased client selection strategies.

4.1 Deep Learning Model

The DL model can be broadly divided into four parts which are connected in a series-parallel configuration. One part each for each of the input sensor modalities (i.e. GPS, LiDAR and camera images) defined as unimodal networks which are connected in parallel. The final layer of each of these three unimodal networks is concatenated to be fed to the fourth part, the fusion network. The fusion network is connected in series with the three unimodal networks connected in parallel. The main advantage of defining such a deep learning architecture is that the low-level inputs from each sensor are classified to form high-level features at the final layers of each unimodal. The high-level features classified in the final layer of the unimodal help the fusion model to address the non-linear relationship between the multimodality sensor data and the beamforming probability densities to form the subset β_K.

The non-linear parameterized function defined by the three unimodal networks qualitatively works as a high-fidelity feature extractor. The feature extractor endeavors to encode the highly non-linear data gathered from the environment by the respective sensors into a high-level feature representation. Through every node of the neural network's layer, it extracts and maps valuable information from raw sensor data. The number of nodes in the penultimate layer of each of the network, heuristically corresponds to the number of high-level features representing the environmental space. This process can be likened to sequencing the most pertinent environmental features represented in a vector form from pre-processed inputs represented in a scalar matrix form. The kernels of the CNN layers define this sequencing and identification of the most persistent environmental features. The environmental features can represent a multitude of things such as vehicles, trees, bridges, traffic signals, sign boards, street lights,

buildings etc., along with their reflective indexes which the kernels of the CNN are trained to identify. The DL model can be thought of as a black-box whose blue-print is provided by us. The final layer has a SoftMax classifier, which gives the probability density for each configurable beamforming pair based on the high-level feature extracted. The multi class cross-entropy loss function initiates backpropagation and updates the weights after each training round based on the error calculated between predicted probability densities by the DL model and the actual ground truth values of the optimum beamforming pair "One-Hot" encoded. One-Hot encoding is used to pre-process the ground truth values of the coupling pairs of all the beamforming pairs. The optimum beamforming pair with maximum coupling power is coded with a value of one and the others with a value of zero. Such a scheme of coding the ground truth values simplifies the calculation of the error in a multiclass classification problem.

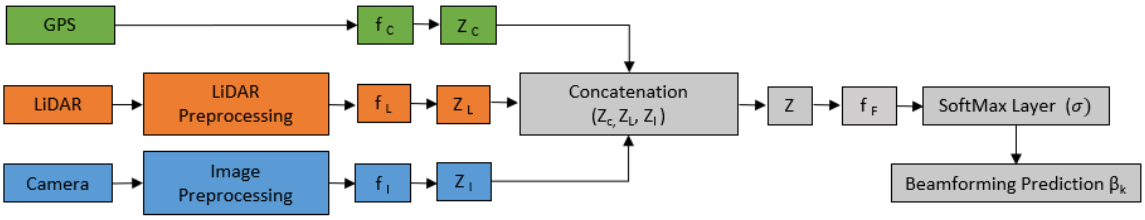

Figure 7: Fusion based Deep Learning Model Flowchart

The DL model for the three unimodal networks and the fusion network is illustrated from the F-DL model defined in FLASH by Batool et al. [18]. The inputs for the model come from the three sensors: GPS, LiDAR and camera. The LiDAR and camera images are first preprocessed so that only adequate information is fed into the unimodal network, eventually reducing the computational load and leading to faster convergence. The deep learning flowchart is presented in Figure 7 and the architecture of each of the unimodal networks is explained in Appendix A. Once the sensor inputs are pre-processed, they are ready to be fed into the respective unimodal.

Let us assume the that there are N samples available, then the size of each of the input modality for is given by :

$$X_C \in R_{N*2} \quad (4.1a)$$

Where, X_C is the input from GPS representing two coordinates.

$$X_L \in R_{N*dL1*dL2*dL3} \quad (4.1b)$$

Where X_L is the input matrix from LiDAR with a 3D shape and dL1,dL2 and dL3 being the dimensions of the pre-processed histogram along the three axes.

$$X_I \in R_{N*dI1*dI2*dI3} \quad (4.1c)$$

Where X_I is the input RGB matrix from images and dI1,dI2 and dI3 representing the R, G, and B values of the pixels after pre-processing. The **ground truth values** of the beam

pairs in the overall beamforming set ‘β’ for each sample in ‘N’ are encoded as one-hot encoding where the optimum beam is set to 1 and the others are set to 0. Thus, the ground truth output matrix can be defined as:

$$Y = \{0,1\}_{N*\beta} \quad (4.1d)$$

The unimodal networks can be mathematically represented as parametrized functions which try to classify high level features from the low-level respective sensor inputs. These functions can be represented as:

$$Z_c = f_C(X_C), \qquad f_C : R_2 \rightarrow R_{D1} \quad (4.2a)$$

$$Z_L = f_L(X_L), \qquad f_L : R_{dL1*dL2,dL3} \rightarrow R_{D2} \quad (4.2b)$$

$$Z_I = f_I(X_I), \qquad f_I : R_{dI1*dI2*dI3} \rightarrow R_{D3} \quad (4.1c)$$

Where Z_c, Z_L, Z_I show the extracted high-level feature vectors from the respective input data X_C, X_L, X_I. The parameters D1, D2, and D3 represent the size of high-level feature vector extracted by GPS, LiDAR and camera unimodal. Once the high-level features are available from the respective parallel unimodal networks, they are concatenated and fed into the fusion network which finds the relationship between the high-level features and the probability density for each of the beamforming pairs in the set ‘β’. The concatenated matrix is represented by:

$$Z = [\, Z_c, Z_L, Z_I \,] \in R_{D1*D2*D3} \quad (4.3)$$

The parametrized function representing the fusion network is represented as:

$$S = \sigma(f_F(Z)), \qquad f_F : R_{D1*D2*D3} \rightarrow R_{|\beta|} \quad (4.4)$$

Where f_F is the function trained by the fusion network on the concatenated set Z. The symbol “σ" denotes the SoftMax activation function and the set S indicates the predicted probability density value of each beamforming pair in the set ‘β’ by the fusion network. Equation (4.4) forms a probability distribution with $S_{optimum}$ being the optimum pair returned by the fusion network :

$$S_{optimum} \Rightarrow P((t^*, r^*)) = \text{argmax}(S_i),\ i \in \beta, \quad (4.5)$$

Where S_i is the set of all the probability densities predicted by the fusion network for each of the beamforming pairs. Correspondingly, equation (4.5) can be used to quantitatively evaluate equation (2.5) to find the subset β_K with top-K beamforming pairs given by:

$$\beta_K = \text{argmax}(S_i),\ i \in A \quad (4.6)$$

Where $A \subseteq \beta$ and $|A| = K$.

4.2 Centralized Architecture

In section 4.1, the pipeline for the DL model from using the pre-processed sensor information to predicting the top-K beamforming pairs was discussed. In this section, we use the DL model in a centralized architecture scheme. In a centralized architecture scheme, all the data samples for training and validating are made available at the server (MEC) over the upstream sub-6GHz channels. The LiDAR inputs are pre-processed at the vehicle before they can be shared over the sub-6GHz channels. The camera images are pre-processed at the vehicle or MEC based on where the camera is placed. The training and validation of the model always takes place at the server. Once the model is trained and deployed at the MEC, the vehicles send their real-time preprocessed sensor data to the MEC to establish a connection. The server runs the DL model on the real-time data and retunes the top-K beamforming pairs. The top-K beamforming pairs are shared with the vehicle to begin the standard beam sweeping procedure on the smaller top-K beamforming subset. Figure 8 illustrates the flow chart for training and post deployment testing processes for the centralized architecture based on which the pseudo code is implemented in Algorithm 1. The function of each of the states is defined chronologically as follows:

State (a): The vehicle collects multimodal sensor data and pre-process them. This step forms the input data matrix $X_{c,}$ $X_{L,}$ X_I and Y.

State (b): The pre-processed data is shred with the MEC using the Uplink sub-6 GHz channel.

State (c): The time stamped training data is available at the MEC. During this state, the initialization and training of the DL model takes place.

State (d): The trained DL model is deployed at the MEC.

State (e): The model can now predict the top-K beamforming pairs. At this step, the overall system is ready. The vehicle can now share their real-time test data to the MEC to establish connection.

State (f): The deployed DL model at the MEC predicts the top-K beamforming pairs on the test data.

State (g): The MEC share the top-K beamforming pairs with the vehicle using the Downlink sub-6 GHz channel.

State (h): The vehicle and server carry out the standard beam sweeping procedure on the top-K beamforming pairs.

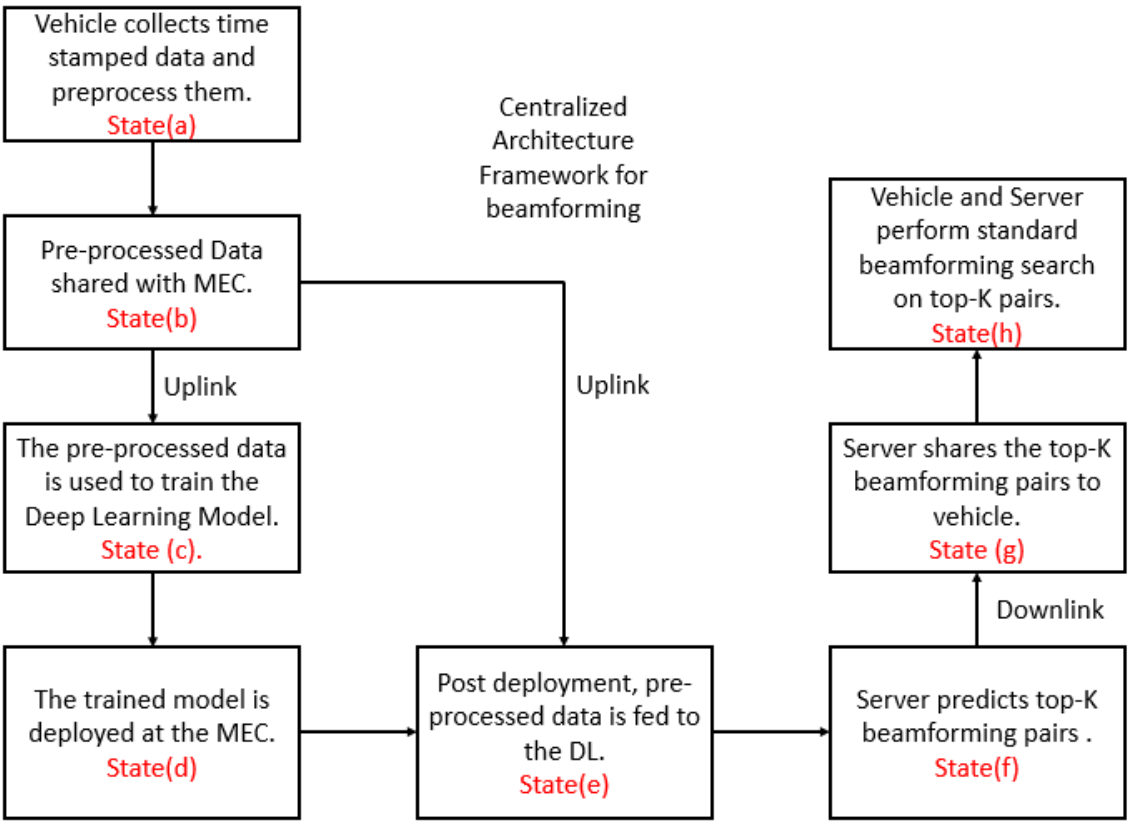

Figure 8: Centralized Architecture Framework for beamforming process

Algorithm 1 Pseudo Code for Centralized Architecture for beamforming training using multimodality

1: **Input at Vehicle**: X_c , X_L, X_I, Y for n ∈ M {Pre-processed sensor inputs at vehicle and MEC}, size of K

2: **Output:** $\beta_{K.}$

3: **Initialize:** f_C, f_L, f_I, f_F {Overall Deep Learning Model at MEC with random initial weights}

4: **Vehicle Do:**

5: **Return**: X_c , X_L, X_I to MEC {Using Uplink}

6: **MEC Do:**

7: **Train**: f_C, f_L, f_I, f_F {Until 100 epochs with batch size of 32 using Adam optimizer}

8: Compute $\beta_{K.}$ {Which are Top-K beamforming pairs corresponding to 'K' largest values using the trained model}

9: **Return**: $\beta_{K.}$ { Using Downlink}

Using Algorithm 1, a set of experiments are designed and its results are discussed in section 5.1.

4.3 Distributed architecture using biased client selection strategy

Distributed architecture uses federated learning paradigm were a set of "M" different vehicles (clients) participate in training the global model at the MEC (server). The deep learning architecture at each of the clients (local model) and the server (global model) is same at the beginning of each training round. A training round comprises all the states from State (b) to State (h) as illustrated in the flow-chart in Figure 9. The DL model discussed in Section 4.1 is used in a distributed architecture setup. Two client biased selection strategies are studied and incorporated for federated learning forming the motivation to adapt the novel distributed architecture.

The important advantage of using a distributed architecture reduces the volume of data to be transferred on the sub-6GHz channels from the vehicles to the MEC. Moreover, the novel distributed architecture developed in this study is able to further reduce the use of sub-6GHz as only a subset of "m" clients ($m \subset M$) participate in training during each training round. This also reduces the overall training time for the convergence of the global model and supports in-field training, and helps to carry out updates on the global model to adopt to future variations in the immediate spatial environment around the MEC post deployment. The process of orchestrating the selection of 'm' clients at each training round is done by the server using the biased client selection procedures later introduced in section 4.5. The framework of the novel distributed architecture follows the flowchart illustrated in Figure 9 which have eight states from (a. to h.) during each training round. Post deployment of the DL model, the testing process follows the same procedure as in a centralized architecture (State (e) to State (h) in Figure 8). The downlink and uplink sub-6Ghz communication channel links are used to establish control and communication between the server and the clients. The function of each of the states is defined chronologically as follows:

State (a) : This is the initialization state. The global model is initialized at the server with using the defined DL model introduced in section 4.1. Random initial weight matrix is assigned for the global model at the server. In this step, the local models are also initialized without assigning random weights.

State (b) : For the first training round, the randomly initialized weights matrix at server from State (a) are shared with all the clients. For subsequent training rounds $k = \{2, 3, ..., \tau\}$, the updated weights matrix from the global model are shared with all the clients. Downlink-1 communication channel is used to share the weights from the server to all the clients.

State (c): All the clients use the weights matrix updated by the server in the global model during the previous training round (τ -1) except for the first training round. For the first training round, the randomly initialized weights matrix by the server is used. The clients use the local validation data set to test the weights matrix shared by the server and return the validation loss back to the server for orchestrating the biased client selection process. The Uplink-1 communication channel is used to return the validation loss back to the server.

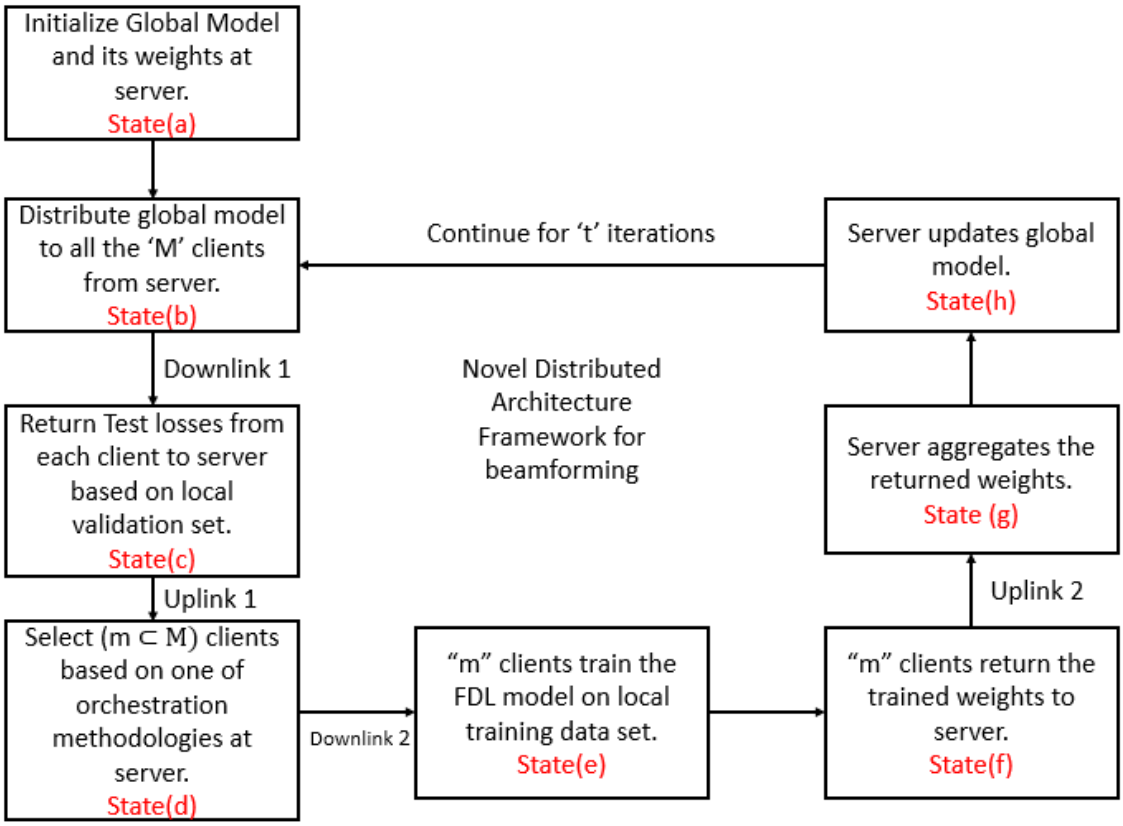

Figure 9: Distributed Architecture Framework for beamforming process

State (d): The server uses the returned validation losses from each client in State (c) to orchestrate the selection of m⊂M clients which will participate in the training for the current training round 'k'. The biased client strategies are used to orchestrate and select the subset of 'm' clients for training. Downlink-2 is used to trigger the 'm' selected clients to start training using the global model's weights shared in State (b).

State (e): The global model's weights matrix which is already available from State (b) are used on the local architecture which is available at the selected 'm' clients for training. The local training dataset available at each of the selected client is used for this purpose.

State (f): Once the training at the selected clients is concluded, the clients share the weights which were updated during the local training to the server. Uplink-2 communication channel is used to share the local trained weights.

State (g): Server aggregates all the corresponding elements of the weight's matrix returned by all the selected 'm' clients for the current round of training. A federated averaging methodology [20] is used for aggregating the corresponding elements from all the 'm' clients.

State (h): Based on the aggregation concluded in State (g), the global model is updated for the current training round. The server tests the updated global model on the global test

set and checks for its convergence. This concludes the current training round ‘k’. If the global test losses reach a global minimum, the training process is terminated and otherwise the process continues to begin the next training round. The server is ready to distribute the weights of the updated g-lobal model to begin the next round of training and goes back to state (b) again.

The functions of each state for the novel distributed architecture are summarized as a pseudo code in Algorithm 2. Figure 10 shows the timeline of each state between two different training rounds using a time axis. In Figure 10, the vertical axis represents the time axis and the green and orange lines represent the uplink and downlink channels respectively. The states executed by clients are represented in purple and the states executed by the server are represented in brown color codes.

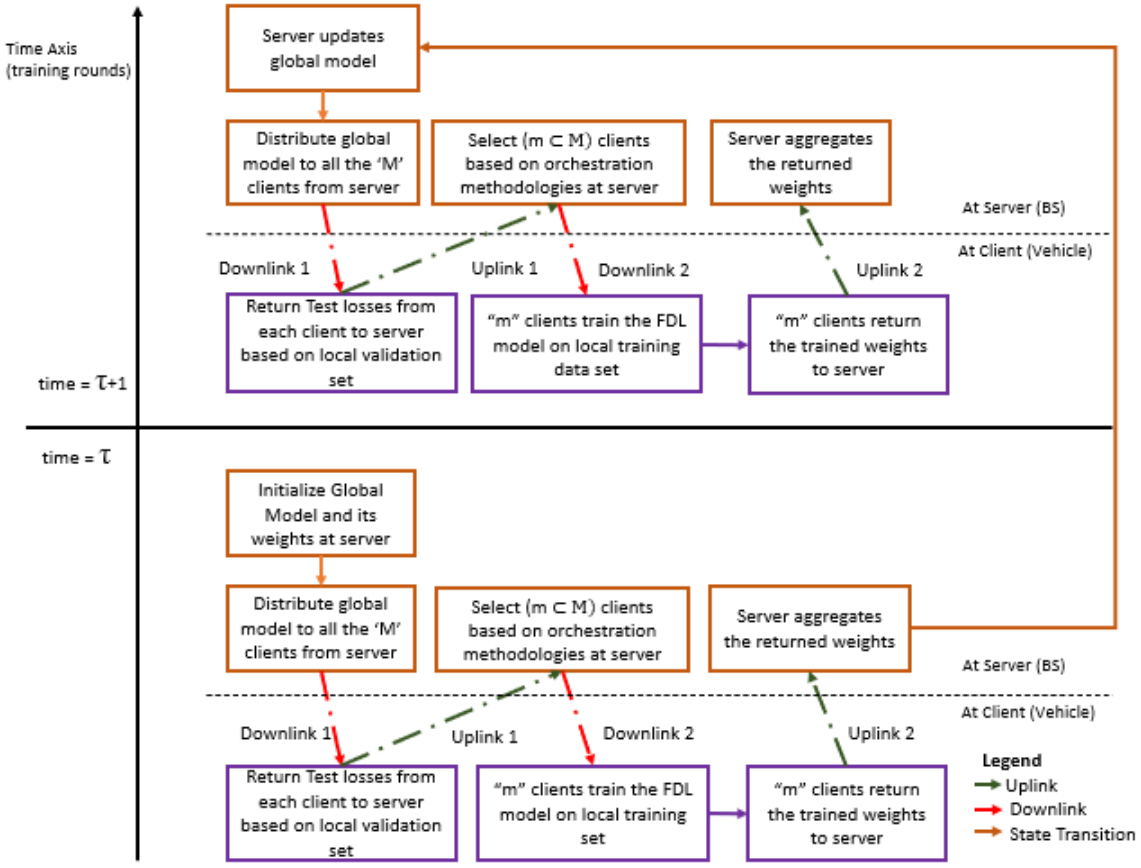

Figure 10: Time line of first training round and the following training rounds for distributed architecture.

Now that we have established the framework of the novel distributed architecture for simplifying beamforming process, the next section focuses on the orchestration steps of State (d). Using Algorithm 2, a set of experiments are designed and its results are discussed in section 5.2. The steps to run the python implementation of the novel distributed architecture is available in Appendix B.

Algorithm 2 Pseudo Code for Novel Distributed Architecture for beamforming training using multimodality and biased client selection strategy

1: **Input at Clients**: $X_c^{(n)}$, $X_L^{(n)}$, $X_I^{(n)}$, $Y^{(n)}$ for n ∈ M {Pre-processed sensor inputs at each client}, size of K

2: **Output:** β_K.

3: **Initialize:** $f_C^{(S)}$, $f_L^{(S)}$, $f_I^{(S)}$, $f_F^{(S)}$ {Overall Deep Learning Model at Server with random initial weights}

4: **Repeat-Until:** Training Round 'k' in {1,2, 3, ..., τ }

5: **Global Server Do:**

6: **Return**: $f_C^{(S)}$, $f_L^{(S)}$, $f_I^{(S)}$, $f_F^{(S)}$ to all the clients {Using Downlink 1}

7: **Clients n ∈ M Do:**

8: Compute Loss(n, k) for n ∈ M {using local validation dataset upon Global Weights $f_C^{(S)}$, $f_L^{(S)}$, $f_I^{(S)}$, $f_F^{(S)}$}

9: **Return:** Loss(n, k) for n ∈ M {Using Uplink 1}

10: **Global Server Do:**

11: Orchestrate using Algorithm 2 or Algorithm 3 to select 'm' clients

12: **Return:** m {Using Downlink 2}

13: **For Clients n ∈ m Do:**

14: **Use:** $f_C^{(n)}$ (=$f_C^{(S)}$), $f_L^{(n)}$ (= $f_L^{(S)}$), $f_I^{(n)}$ (=$f_I^{(S)}$), $f_F^{(n)}$ (=$f_F^{(S)}$)

15: **Train**: $f_C^{(n)}$, $f_L^{(n)}$, $f_I^{(n)}$, $f_F^{(n)}$ {Until 100 epochs with batch size of 32 using Adam optimizer}

16: **Return:** $f_C^{(n)}$, $f_L^{(n)}$, $f_I^{(n)}$, $f_F^{(n)}$ {Using Uplink 2}

17: **Global Server Do:**

18: $f_C^{(S)} = \sum f_C^{(n)}/\text{sizeof(m)}$, $f_L^{(S)} = \sum f_L^{(n)}/\text{sizeof(m)}$, $f_I^{(S)} = \sum f_I^{(n)}/\text{sizeof(m)}$, $f_F^{(S)} = \sum f_F^{(n)}/\text{sizeof (m)}$, n ∈ m {Update Global Weights by aggregating all the weights from selected client's 'm'}

19: **Global Server Do:** {For Validation}

20: Compute S. {Using Trained $f_C^{(S)}$, $f_L^{(S)}$, $f_I^{(S)}$, $f_F^{(S)}$}

21: Compute β_K.{Which are Top-K beamforming pairs corresponding to 'K' largest values in S}

22: **Return:** β_K

4.4 Distributed architecture using unbiased client selection strategy

Unbiased client selection strategy uses the contribution of all the clients' trained local weights at the end of each training round to update the global model at the server. An aggregating scheme defined in the "Federated Averaging" algorithm [20] is incorporated. The averaging scheme of the global model is given by:

$$f_C^{(S)} = \sum f_C^{(n)}/\text{sizeof(M)} \quad (4.7a)$$

$$f_L^{(S)} = \sum f_L^{(n)}/\text{sizeof(M)} \quad (4.7b)$$

$$f_I^{(S)} = \sum f_I^{(n)}/\text{sizeof(M)} \quad (4.7c)$$

$$f_F^{(S)} = \sum f_F^{(n)}/\text{sizeof(M)} \quad (4.7d)$$

For n ∈ M, where $f_C^{(S)}$, $f_L^{(S)}$, $f_I^{(S)}$ are the global unimodal networks for GPS, LiDAR and camera images respectively. $F_F^{(S)}$ is the global model fusion network. $f_C^{(n)}$, $f_L^{(n)}$, $f_I^{(n)}$ are the local unimodal fusion network at each client (n) for GPS, LiDAR and camera images respectively. $F_F^{(C)}$ is the local model at each client (n) for fusion network. To adapt the unbiased client selection strategy in the novel distributed architecture, the pseudo code in Algorithm 3 is tweaked wherein State (b), State (c) and State (d) are bypassed in Figure 9.

Algorithm 3 Pseudo Code for Novel Distributed Architecture for beamforming training using multimodality and unbiased client selection strategy
1: **Input at Clients**: $X_c^{(n)}$, $X_L^{(n)}$, $X_I^{(n)}$, $Y^{(n)}$ for n ∈ M {Pre-processed sensor inputs at each client}, size of K
2: **Output:** β_K.
3: **Initialize:** $f_C^{(S)}$, $f_L^{(S)}$, $f_I^{(S)}$, $f_F^{(S)}$ {Overall Deep Learning Model at Server with random initial weights}
4: **Repeat-Until:** Training Round 'k' in {1,2, 3, ..., τ }
5: **Global Server Do:**
6: **Return**: $f_C^{(S)}$, $f_L^{(S)}$, $f_I^{(S)}$, $f_F^{(S)}$ to all the clients {Using Downlink 1}
7: **For Clients n ∈ M Do:**
8: **Use:** $f_C^{(n)}$ (=$f_C^{(S)}$), $f_L^{(n)}$ (= $f_L^{(S)}$), $f_I^{(n)}$ (=$f_I^{(S)}$), $f_F^{(n)}$ (=$f_F^{(S)}$)
9: **Train**: $f_C^{(n)}$, $f_L^{(n)}$, $f_I^{(n)}$, $f_F^{(n)}$ {Until 100 epochs with batch size of 32 using Adam optimizer}
10: **Return:** $f_C^{(n)}$, $f_L^{(n)}$, $f_I^{(n)}$, $f_F^{(n)}$ {Using Uplink 1}
11: **Global Server Do:**
12: $f_C^{(S)} = \sum f_C^{(n)}/\text{sizeof(M)}$, $f_L^{(S)} = \sum f_L^{(n)}/\text{sizeof(M)}$, $f_I^{(S)} = \sum f_I^{(n)}/\text{sizeof(M)}$, $f_F^{(S)} = \sum f_F^{(n)}/\text{sizeof}$ (Mn ∈ M {Update Global Weights by aggregating all the local weights}
13: **Global Server Do**: {For Validation}
14: Compute S. {Using Trained $f_C^{(S)}$, $f_L^{(S)}$, $f_I^{(S)}$, $f_F^{(S)}$}
15: Compute β_K.{Which are Top-K beamforming pairs corresponding to 'K' largest values in S}
16: **Return:** β_K

4.5 Biased Client Selection Strategy

Two biased client selection strategies are studied and incorporated in the orchestration step. By using biased selection strategy, we are able to select a smaller subset of clients requiring only to participate in training. This helps to reduce the usage of sub-6GHz channels and easily upscale the model by having larger number of clients. As the number of clients participating in training is reduced, the volume of information to be shared between the clients and server is reduced bringing down the usage of sub-6GHz channels.

4.5.1 Max Loss Biased Client Selection Strategy

The outcome of reducing the overheads on the secondary sub-6GHz communication channels is accomplished by using biased client selection strategies in a federated learning setup. Previous studies on distributed architecture for beamforming process by Batool et.al in FLASH [18] use an unbiased client selection scheme. In an unbiased

selection scheme, all the clients participate in each round of training. Thus, the global weight matrix is always a contribution of all the clients after each round of training. Although, this method guarantees convergence of the global model as mathematically proved by B. McMahan in the "Federated Averaging" algorithm [20] for federated learning based stochastic gradient descent problems, the resource overhead and time for training are higher. The server needs to periodically wait for all the client to finish training at each round and process all the weights to update the global weight matrix. This also creates a bottleneck when the number of clients is larger. These shortcomings can be well addressed by using biased client selection strategies. In biased client selection strategy, only a subset of all the clients is chosen to train the model at each training round. The smaller subset of clients is chosen such that, their contribution towards the update of the global model is cognizant with using unbiased selection strategy. Yae Jee Cho et.al [21], mathematically proved that biased selection of clients with higher local losses leads to three times faster convergence rate and also leading to reduced resource overheads. Qualitatively, this can be understood by looking at these clients with higher local losses as having larger heterogeneity in their dataset. In our case, these clients would have larger NLOS cases and giving more emphasis to their local weight matrices can canvas for other non-selected clients with a high probability. Thus, selecting clients with higher local losses for each round of training is the first method of orchestration. Algorithm 4 shows the pseudo code for MaxLoss based client selection orchestration.

The Power-of-Choice (πpow-d) methodology from [21] is adapted and leveraged for this method. The processing steps referring to Figure 9 as follows:

Step 1: The Server sends the updated global weight matrix to all the clients in State (b).

Step 2: The clients return their local validation losses using the updated global weight matrix back to the server in State (c).

Step 3: The server selects a subset of 'm' clients from the total set of clients 'M' having largest local losses in State (d). The ties if occurred are broken at random.

Step 4: The selected clients belonging to set 'm' participate in training for the current training round.

Algorithm 4 Pseudo Code for MaxLoss based client selection strategy
1: **Input**: size of m, Loss(n), for n ∈ M
2: **Output:** $m^{(k)}$
3: **Global Server Do:**
4: Compute $m^{(k)}$ {which are the clients corresponding to largest values in Loss(n)}
5:Return: $m^{(k)}$

Although, this orchestration method is elegant in its simplicity, it may not be the best method for biased client selection to suit our application. Selecting clients with higher losses will qualitatively lead to improving NLOS cases but does not necessarily canvas for simpler LOS cases from other non-selected clients. Particularly, the global model tries

to fit outliers diminishing its ability to generalize LOS and NLOS cases. This encourages to use the heuristics first presented by Yae Jee Cho et.al [22], by treating client selection strategy as arms of bandit and resolving conflicts using the UCB-CS strategy.

4.5.2 Heuristic Multi Arm Bandit based biased client selection strategy

As discussed previously, to expedite convergence of the global model, it's imperative to prioritize clients exhibiting larger local losses at each training round. Alongside, it's equally important to uphold diversity among all the clients which only guarantees to mitigate a low error floor. This dilemma can be modelled as an exploration-exploitation trade-off which occurs in immediate reinforcement learning problems. The problem of choosing only a subset of clients with maximum error loss at each training round alongside giving opportunities to other clients to achieve a low error floor can be advocated as a Multi-Arm Bandit (MAB) problem [27]. Given the non-stationary nature of individual clients' local loss values at the end of each training round, the discounted Upper Confidence Bounds (UCB) algorithm can be adapted to achieve this balance. Conceptualizing clients as arms in the MAB problem, the discounted cumulative local loss values $L_k(\Upsilon, n)$, and the discounted count of times each client has been sampled, $N_k(\Upsilon, n)$, are computed for each client up to training round 'k'.

Utilizing these, the discounted UCB index for each client ($n \in M$) after each training round is computed. A subset of 'm' clients with the highest discounted UCB indices are selected to participate in the next round of training. Considering $k = \{2, 3, ..., \tau\}$ as the time indices for communication rounds until τ training rounds, the discounted UCB indices $A_k(\Upsilon, n)$ are computed for each client. To adapt to our application, discounted UCB indices $A_k(\Upsilon, n)$, the discounted cumulative local loss values $L_k(\Upsilon, n)$, the discounted count of times each client has been sampled, $N_k(\Upsilon, n)$ and the exploration term $U_k(\Upsilon, k)$ are modified from the equations presented in [22].

$$A_k(\Upsilon, n) = p_n \left(\underbrace{L_k(\Upsilon, n) \,/\, N_k(\Upsilon, n)}_{\text{exploitation}} + \underbrace{U_k(\Upsilon, n)}_{\text{exploration}} \right) \qquad (4.8)$$

The equation (4.8) represents the discounted UCB indices which models the exploration-exploitation trade-off. After each training round 'k', in State(b) referring to Figure 9, the server sends the updated global weight matrix to all the clients. The clients return their local validation losses using the updated global weight matrix back to the server in State (c). The server selects a subset of 'm' clients from the all the clients 'M' based on the largest values of the computed discounted UCB indices $A_k(\Upsilon, n)$. In context of UCB algorithms for modelling MAB problems, this is similar to taking actions (choosing clients) based on the maxima of computed rewards (values of $A_k(\Upsilon, n)$). In equation (4.8), p_n represents the fraction of validation samples available at a client 'n' as shown below in equation (4.9). This term forces the selection of those clients having higher number of data samples.

$$p_n = d_n/D,\ n \in M \qquad (4.9)$$

In equation (4.9), "d_n" represents the number of validation samples at client 'n' and D represents the total number of validation samples.

$$L_k(\Upsilon, n) = \sum_{k' \in k} fs * Loss(n, k') \quad (4.10)$$

The equation (4.10) represents the discounted cumulative local loss values $L_k(\Upsilon, n)$. This term computes the total rewards from the past training rounds associated with choosing a particular client 'n' until the current training round 'k'. The term Loss(n, k') represents the validation loss value returned by a particular client 'n' in a past training round k'. The magnitude of "Loss" returned by a particular client corresponds to the reward associated with choosing them. This essentially drives the exploitation term towards clients exhibiting larger losses which is reasonable from our discussion of using MaxLoss based client selection strategy. The term '*fs'* represents the unitary function given by equation (4.11)

$$fs = \Upsilon^{(k-k')} 1_{\{n \in m^{(k'-1)}\}} \quad (4.11)$$

In equation (4.11), 'Υ' is a hyper parameter whose significance emphasizes the influence of stale values for UCB algorithms to model MAB problems. In the context of UCB, which accounts for the rewards accounted from past actions, a value of reward can become 'stale' if its influence decreases on the action taken for the current state. When $\Upsilon = 1$, all past local loss values are given equal weight, whereas $\Upsilon = 0$ considers only the most recent local loss. For $0 < \Upsilon < 1$, less importance is placed on stale values, resulting in a robust estimation of the client's local loss and a reduction in noise from the latest evaluation. $1_{\{n \in m^{(k'-1)}\}}$ in equation (4.11) is an indicator function. The value is equal to one if the condition in the curly brackets is true otherwise its value is zero. The condition in the curly brackets check if a client 'n' belonged to the set 'm' in the immediate past training round.

$$U_k(\Upsilon, n) = \sqrt{2\sigma * \sigma * \log T_k(\Upsilon) / N_k(\Upsilon, k)} \quad (4.12)$$

$$N_k(\Upsilon, k) = \sum_{k' \in k} fs \quad (4.13)$$

In equation (4.12), $U_k(\Upsilon, n)$ represents the exploration term for a particular client 'n' and at a training round 'k'. The term $N_k(\Upsilon, k)$ gives the discounted count of times each client has been selected in the past given by equation (4.13). This term is particularly helpful to understand how many times a particular client has been selected in the past rounds of training and forms the idea for exploration. It suppresses the magnitude of exploration for clients which have been chosen larger number of times in the past and increases the chances of selection of other clients which have not been selected in the past. The term "σ" is the variance of the validation losses returned by all the clients at the beginning of each training round. The exploration term $U_k(\Upsilon, k)$ is amplified by a factor of σ, particularly for clients not selected recently, which encourages exploration of clients with potentially smaller exploitation values. This prevents the algorithm from exclusively selecting clients with larger local losses, thus mitigating a lower error floor, while

ensuring fairness of selection for each client. Finally, the term $T_k(\Upsilon)$, is calculated by the equation given by equation (4.14).

$$T_k(\Upsilon) = \Sigma\, \Upsilon^{(k-k')} \quad (4.14)$$

The fairness of selection for each client is defined by the fairness factor given by Jain's Index [28] and defined in equation (4.15). The fairness factor at the end of each training round (l) is given by J(l) ranges from a value of $1/M \leq J(l) \leq 1$ where 1 is when all clients have the same performance ensuring fairness of selection for each client.

$$J(l) = \frac{1}{M}\left[\sum_{n=1}^{M}\left(\frac{Loss(n,l)}{\sum_{m=1}^{M} Loss(m,l)}\right)^2\right]^{-1} \quad (4.15)$$

The heuristics developed to adapt UCB based client selection strategy for our process has another important practical advantage. It enables in-field training of the global model to improve its adaptability to the dynamically changing physical environment around the vicinity of the MEC. Once the global model is trained and deployed, the model can be updated by canvasing additional clients to accommodate the changes in the physical environment without disturbing the status of the current model. The account of historical rewards and actions from the past training rounds drives the updates to generalize the new model to not to be skewed towards newer clients but accommodate their knowledge alongside the past knowledge gained by the deep neural network. Algorithm 5 discusses the implementation of UCB based client selection strategy. To store the history of all the losses and clients set 'm' until training round 'k', two look up tables are used and updated. Algorithm 5 shows the pseudo code for Heuristic MAB based client selection orchestration.

Algorithm 5 Pseudo Code for Heuristic MAB based client selection orchestration
1: **Input**: Υ, size of m, k, Loss(n,k), $m^{(k-1)}$, p_n for $n \in M$
2: **Output:** $m^{(k)}$
3: **Initialize:** empty Loss Lookup Table, Indicator Lookup Table of size ($\tau \times$ size of M)
4: **Global Server Do:**
5: Append the Loss(n, k) for $n \in M$ list to Loss Lookup Table
6: Append the $m^{(k-1)}$ list to Indicator Lookup Table with clients in $m^{(k-1)}$ equal to 1
7: Calculate σ {variance of Loss(n,k) list for $n \in M$}
8: Compute $L_k(\Upsilon, n)$ {using Loss Lookup Table, Indicator Lookup Table}
9: Compute $N_k(\Upsilon, n)$ {using Indicator Lookup Table}
10: Compute $T_k(\Upsilon)$
11: Compute $U_k(\Upsilon, n)$ {using σ, $N_k(\Upsilon, n)$ and $T_k(\Upsilon)$}
12: Compute $A_k(\Upsilon, n)$ {using p_n, $L_k(\Upsilon, n)$, $N_k(\Upsilon, n)$ and $U_k(\Upsilon, n)$}
13: Get $m^{(k)}$ {which are the clients corresponding to largest values in $A_k(\Upsilon, n)$}
14:**Return:** $m^{(k)}$

4.6 Post Deployment Analysis for Centralized and Distributed Architecture

Until now we have established the design and development of both centralized and distributed architecture using a unimodal-fusion based DL model, we also established the algorithms to predict the top-K beamforming subset (β_K) for both the cases. The DL model deployed at any MEC post training adopts to work within the environmental condition in the vicinity of that particular MEC of interest. Thus, the DL model deployed at each MEC is location specific and is different.

We first understand the initialization phase for centralized and distributed architecture once the DL model is fully trained and deployed at the MEC. To initiate the beamforming process to establish a connection, the vehicle of interest collects real-time sensor data and pre-process them on site. $T_{process}$ is the time required to pre-process that input data at the vehicle which is mainly influenced by the time to process images from camera. The time taken to process LiDAR data can be assumed to be negligible. Once the data is pre-processed, it is shared with the MEC over the sub-6GHz uplink channel. T_{uplink} is the time taken to share the pre-processed data to the MEC over the sub-6GHz uplink channel. The MEC uses these data to run the trained DL model and predict the top-K beamforming subset. $T_{predict}$ is the time taken for the DL model to predict the top-K beamforming pairs. The top-K beamforming pairs are shared back to the vehicle of interest over the sub-6GHz downlink channel. $T_{downlink}$ is the time taken to share the top-K beamforming pairs back to the vehicle over the sub-6GHz downlink channel. Finally, the vehicle of interest and MEC run the standard beam sweeping search on the predicted top-K beamforming subset. Previously, in Chapter 2, we calculated the time required to sweep through top-K beamforming pairs given by equation (2.7) and represented by $T_{sweep}(K)$. Figure 11 illustrates the initialization process and total end-to-end latency ($T_{Total}(K)$) is calculated by Equation (4.16). In Figure 11, the vertical axis represents the time axis and the green and orange lines represent the uplink and downlink channels.

$$T_{Total}(K) = T_{process} + T_{uplink} + T_{predict} + T_{downlink} + T_{sweep}(K) \qquad (4.16)$$

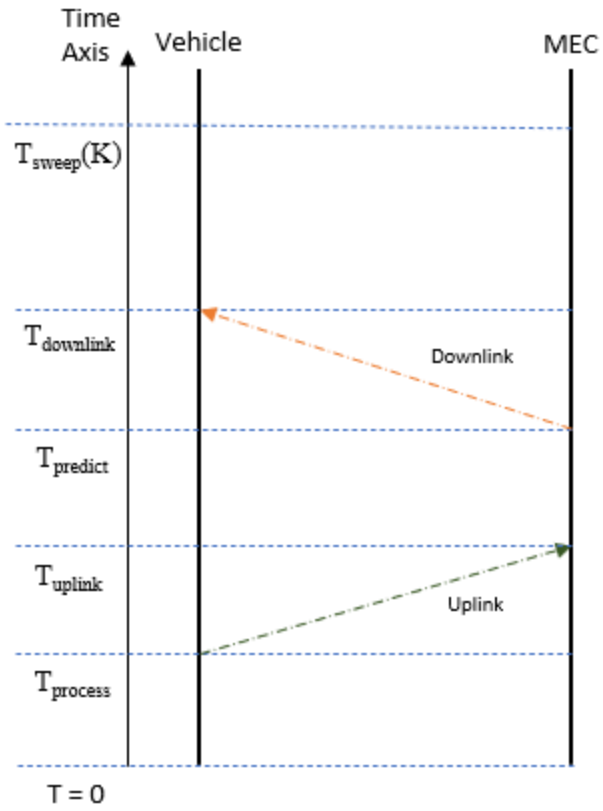

Figure 11: Initialization phase Post Deployment of DL Model at MEC

4.7 Pruning the GPS Unimodal Network

The unimodal-fusion network was studied in detail in section 4.1 and the architecture of each of the unimodal and fusion network is presented in Appendix A. The architecture of all the three unimodal and fusion network is based on the F-DL model proposed by Batool et.al in their study [18]. Analyzing the input-output relationship of each of the unimodal, the size of inputs in equation (4.1) and size of features in equation (4.2) is presented in Table 3.

Table 3: Input-Output Relationship for each Unimodal

Type of Unimodal	**Size of Input**	**Size of Output**
GPS	X_C (=2×1)	Z_C (=64)
LiDAR	X_L (=20×20×20)	Z_L (=512)
Camera Images	X_I (=90×160×3)	Z_I (=256)

Specifically looking into the GPS unimodal, the input of the GPS represents the coordinates of the latitude and longitude of the position of the vehicle. Extracting a feature set of 64 vectors from just two values of coordinates is overly redundant to the model leading to increased complexity and computational requirements. Moreover, from the analyzes on the performance of individual unimodal presented in section 5.1.1 we conclude that there is no significant improvement in the test accuracies. These results

suggest the need to reduce the size of the feature extracted vector from 64 to 2 and check for the performance. Eventually, the middle layers of the models can also be pruned as the output size is reduced. The size of the fully connected dense layers in the middle are reduced by a factor of 32 as a result the total number of parameters of the GPS unimodal is reduced from 696,600 to 3874. This brings a significant a reduction in the computational overheads and complexity of the model by 99.44%. The configurations of each of the layers of the pruned GPS unimodal is discussed in Appendix A and the results of the adaptation of pruned GPS unimodal is discussed in section 5.5.

5 Experiments and Results

This chapter outlines the experiments conducted to evaluate the effectiveness of using multi-modality sensors for predicting a subset of top-K beamforming pairs, utilizing a centralized and distributed architecture alongside the DL model as described in the previous chapter. Initially, the analysis focuses on both unimodal and fusion networks within a centralized framework. Following this, we discuss on how the beamforming process benefits from decreased latency times and increased throughput ratios for a centralized architecture. The design for simulation experiments employing a distributed architecture takes suggestions from the validated results achieved with the centralized approach. This includes confirmation and benchmarking of the Infocom FLASH dataset's application for distributed architecture. Subsequent simulation experiments explore the use of biased client selection strategies in a distributed architecture, a concept introduced in section 4.5, offering a detailed comparison on the benefits of this approach in terms of resource conservation and accelerated training times. Lastly, we present two case studies to investigate the practicality of in-field training after the deployment of the DL model. These case studies, a novel addition to this field, raise questions about the future readiness and adaptability of such a DL model post-deployment.

5.1 Experiments on Centralized Architecture

This section delves into the experimental evaluation of the centralized architecture outlined in section 4.2. We begin by examining the effectiveness of unimodal networks employing GPS, LiDAR, and camera image data individually. Each sensor type is analyzed in detail, providing insights into their respective advantages. Additionally, the study explores the performance of the fusion network combining all unimodal inputs.

Table 4: Simulation settings for experiments conducted for Centralized Architecture

Experiment #	**Experiment Description**	**Data Used**	**Training Epochs**	**Training Settings**
1	GPS Unimodal	Raymobtime S008 and S009 – GPS and RF Labels	20	Learning Rate = 0.0001. Batch Size = 32. Adam Optimizer with beta1 = 0.9 and beta2 =0.999 Data Shuffle enabled
2	Camera Images Unimodal	Raymobtime S008 and S009 – Raw Images and RF Labels	45	
3	LiDAR Unimodal	Raymobtime S008 and S009 – LiDAR and RF Labels	50	
4	3-sensor fusion	Raymobtime S008 and S009 complete datasets	50	
5	GPS-LiDAR fusion	Raymobtime S008 and S009 – LiDAR, GPS and RF Labels	50	

Two distinct setups are assessed: one that uses GPS, LiDAR, and camera image data, and another that incorporates only GPS and LiDAR. The analysis further extends to compare

the impact of these configurations on latency reduction and throughput ratio enhancement. The experiments focus on using the Raymobtime S008 dataset for model training and the S009 for model testing, offering a comprehensive look at the centralized architecture's performance. The experimental results of using multimodality side-information to predict top-K beamforming pairs in a centralized framework further benchmark the validity of this technique previously studied in literature and forms the cues to design experiments for the distributed framework. The Table 4 defines the simulation settings for all the five experiments conducted to validate the centralized architecture.

5.1.1 Performances of Unimodal networks

In this section, we study and evaluate the effectiveness of individual sensor modalities, specifically focusing on their performance within their respective unimodal networks. The evaluation utilizes standalone models for GPS, LiDAR, and images, as detailed by equation (4.2), which defines the parameterized functions for each unimodal along with the inclusion of a SoftMax layer featuring 256 nodes. This layer is designed to represent 256 potential beamforming pairs derived from the Raymobtime dataset, predicting the probability distribution for each pair. The sensor data is preprocessed before entering the training, validation, and testing stages of the unimodal networks.

The analysis highlights that, for GPS data, the validation accuracy stagnates with no observable improvements during training. This stagnation suggests that GPS data, on its own, lacks the dynamic capability to influence environmental features significantly enough to predict beam tracing paths accurately, other than offering basic distance measurements from the MEC to the vehicle. As illustrated in Figure 12 and Figure 13, the GPS network's validation loss ceases to decline after the initial two epochs, with the loss curve plateauing and the training loss hovering around 3.6, indicating the limited predictive utility of GPS data in isolation.

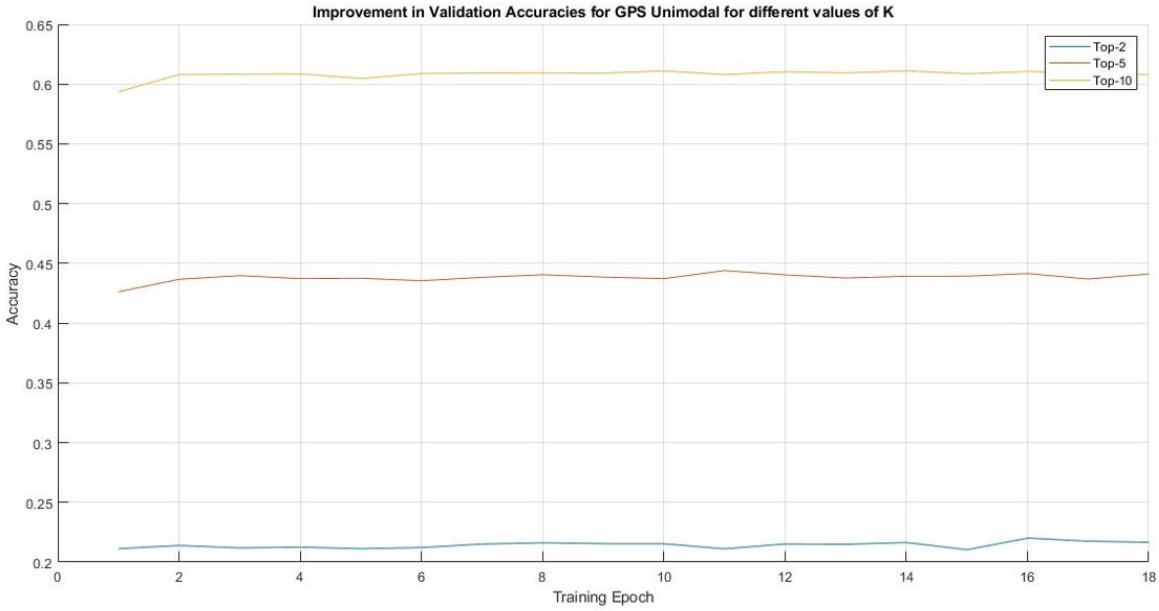

Figure 12: Validation Accuracy for GPS Unimodal for different values of "K"

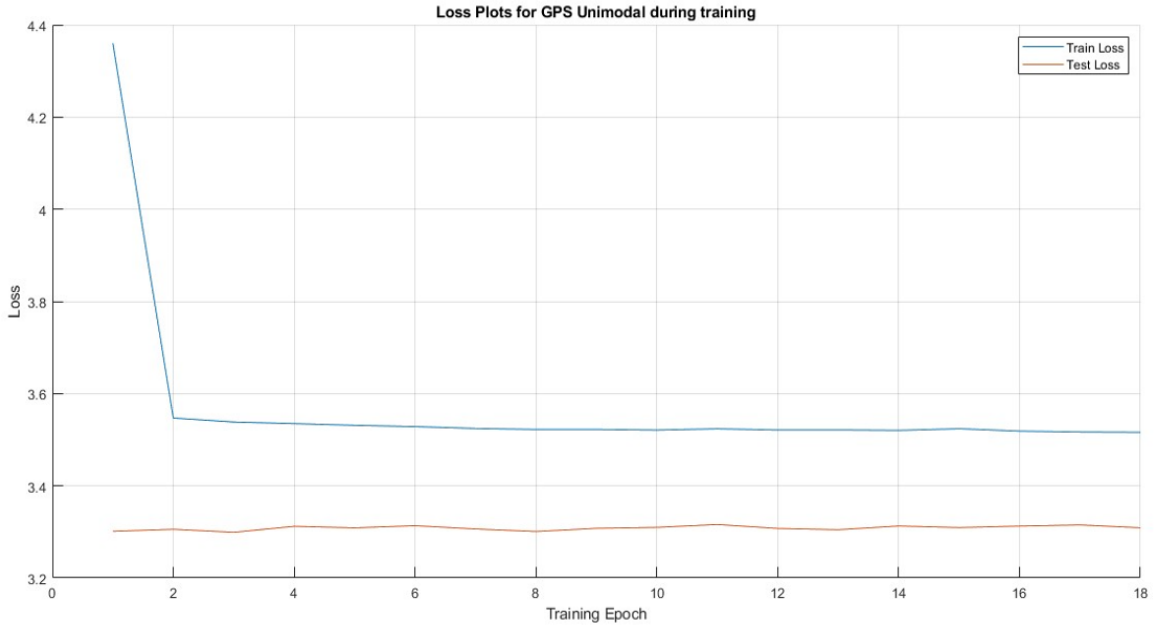

Figure 13: Plots of train and test loss for GPS unimodal

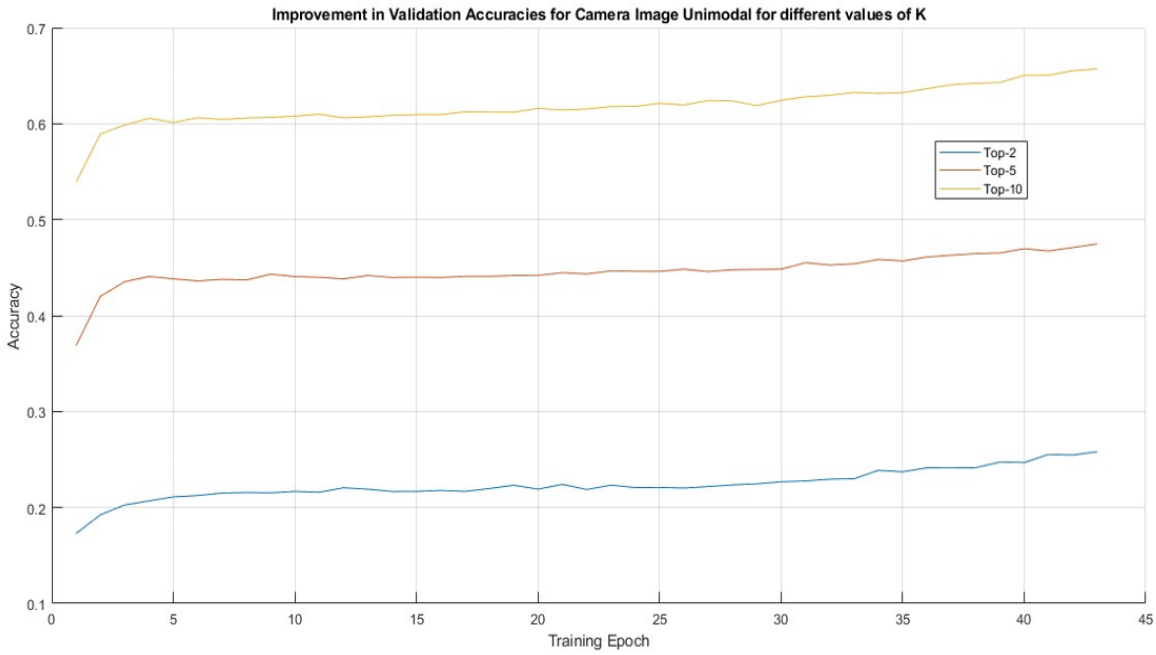

Figure 14: Validation Accuracy for Camera Images Unimodal for different values of "K"

Similarly, the performance of the camera images' unimodal network does not exhibit significant improvement over time in predicting beam pairs, mirroring the pattern observed with the GPS network in Figure 14. The camera's placement at the MEC and its inability to reference the vehicle's position relative to the MEC can be highlighted as primary shortcomings, rendering it ineffective for beam tracing as a standalone modality.

Despite the underwhelming results from the GPS and camera image unimodal networks, their analysis provides valuable insights. The LiDAR data, in contrast, demonstrates a significant improvement in validation accuracy when employed in a standalone network. It addresses the shortcomings of the previous modalities by including reference points of the vehicle's position relative to the MEC and identifying obstacles in the environment. Notably, by using LiDAR data accuracy rates of 0.834 for K=2 and nearly 0.95 for K=10 can be achieved, outperforming the GPS and image based unimodal networks when used independently as shown in Figure 15. LiDAR's comprehensive capture of instant point-cloud data surrounding the vehicle significantly enhances its ability to inform beam tracing paths through DL model. The LiDAR unimodal ability to learn and predict top-K beamforming pairs can be further concluded from Figure 16 as the test loss converges at a value of 1.5 after 27 epochs but the performance may not ne optimum. The results for LiDAR unimodal are good initially, but it starts to overfit after 27 epochs. The test loss starts to increase at this point while the training loss is still decreasing resulting in an overfitting of the model. It can also be concluded that the ability of the model to train reaches an upper saturation and cannot be improved any further. As the study progresses, the focus shifts to examining the potential benefits of combining multiple sensor modalities to enhance the beamforming process, with a particular emphasis on integrating

LiDAR as the primary sensor. This approach aims to leverage the strengths of each modality to overcome their individual limitations.

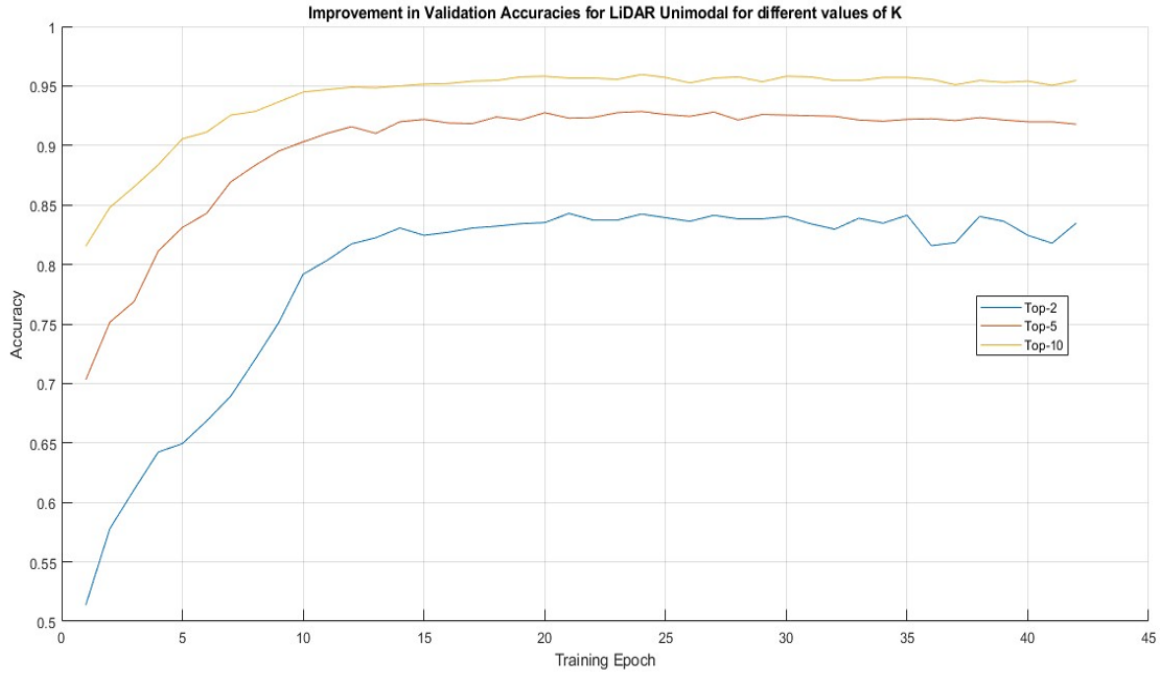

Figure 15: Improvements in Validation Accuracy for LiDAR unimodal for different values of K

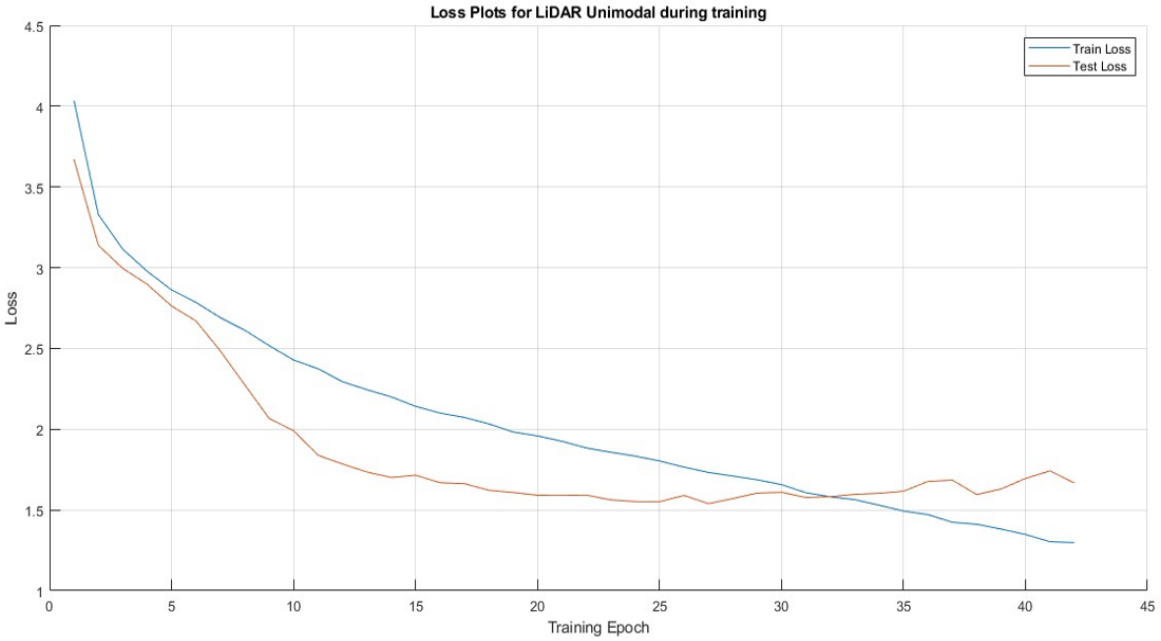

Figure 16: Plots of train and test loss for LiDAR unimodal

5.1.2 Performances of Fusion Framework using three unimodal

Building on the insights from unimodal network analyses, this section examines the outcomes of employing the fusion network discussed in section 4.1. This network integrates the outputs of individual unimodal networks through concatenation, as defined in equations (4.2) representing the high-level feature vectors. Equation (4.3) specifically outlines the creation of a concatenated matrix from the feature embeddings extracted from each sensor modality and represented as high-level feature vectors to serve as inputs for the fusion network. The fusion model's weights are trained following equation (4.4), which establishes the parameters for the fusion network function, f_F. A SoftMax activation function with 256 nodes in the final layer outputs the probability densities for all possible 256 beamforming pairs in the Raymobtime dataset.

The initial experiment incorporates all three sensor modalities—GPS, LiDAR, and camera images into the fusion network. The validation accuracy improvements depicted in Figure 17 illustrate that combining these sensors enhances performance. Specifically, the fusion network achieves validation accuracies of 0.82, 0.92, and 0.95 for top-K values of 2, 5, and 10, respectively. Figure 18 shows the convergence of training and test loss for the three-sensor fusion model. In contrast to the LiDAR unimodal network, both the test loss and training loss continuously decrease highlighting the model's ability to learn better and relate the non-linear relationship to predict the top-K beamforming pairs.

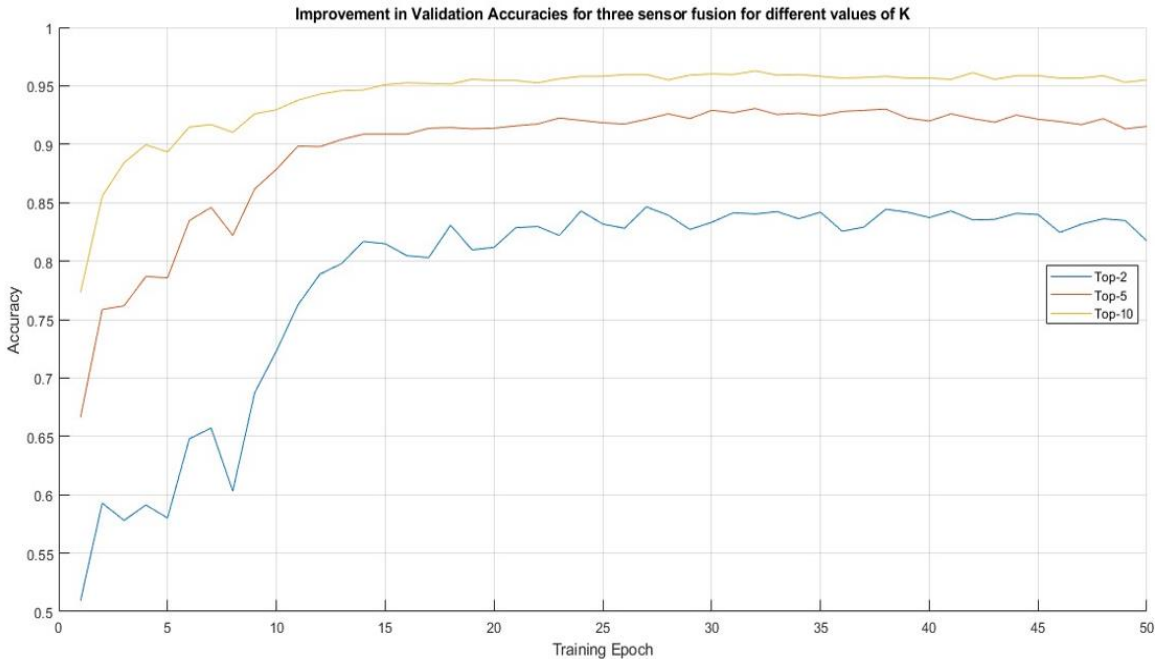

Figure 17: Improvements in Validation Accuracy for three-sensor unimodal-fusion network for different values of K

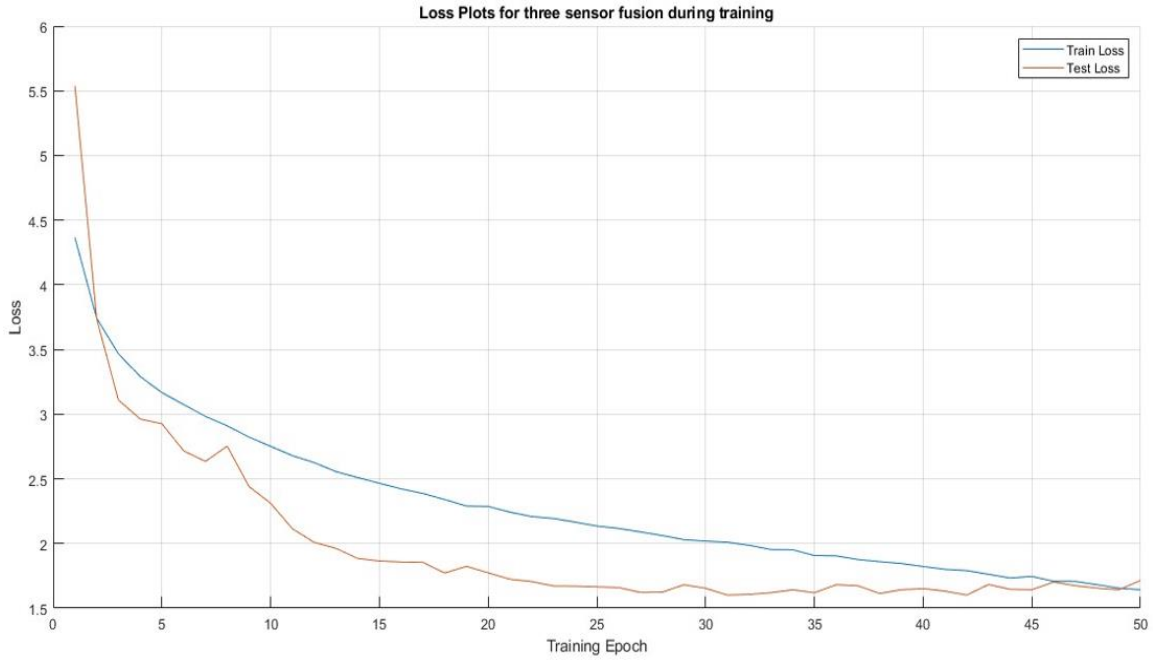

Figure 18: Plots of train and test loss for three sensor fusion

Table 5: Comparison of Validation and Test Accuracy for three-sensor unimodal-fusion network

	Top 1	**Top 2**	**Top 5**	**Top 10**	**Top 25**	**Top 50**
Validation Accuracy	0.65	0.83	0.91	0.95	0.98	0.99
Test Accuracy	0.51	0.70	0.83	0.90	0.97	0.98

To draw conclusive insights on the fusion network's enhancement using all three sensors, the section also presents an overview of the test performance across different K values (1, 2, 5, 10, 25, 50) as shown in Figure 19. Table 5 encapsulates the summary of both validation and test accuracies, providing a holistic view of the model's performance. Additionally, considering the importance of LOS and NLOS cases, Figure 20 offers a comparative analysis of test accuracies for LOS and NLOS scenarios. Notably, the test set (Raymobtime S009) comprises a larger proportion of NLOS samples (85%) compared to the training set, which includes fewer NLOS samples (42%). Despite the validation accuracy being marginally higher than the test accuracy, the substantial improvement in test accuracy, particularly given the increased presence of NLOS samples, underscores the model's efficacy in dense urban settings characterized by a high prevalence of NLOS scenarios. This disparity in NLOS sample distribution between training and test datasets underlines the DL model's capability to adapt and perform robustly within a centralized architecture framework.

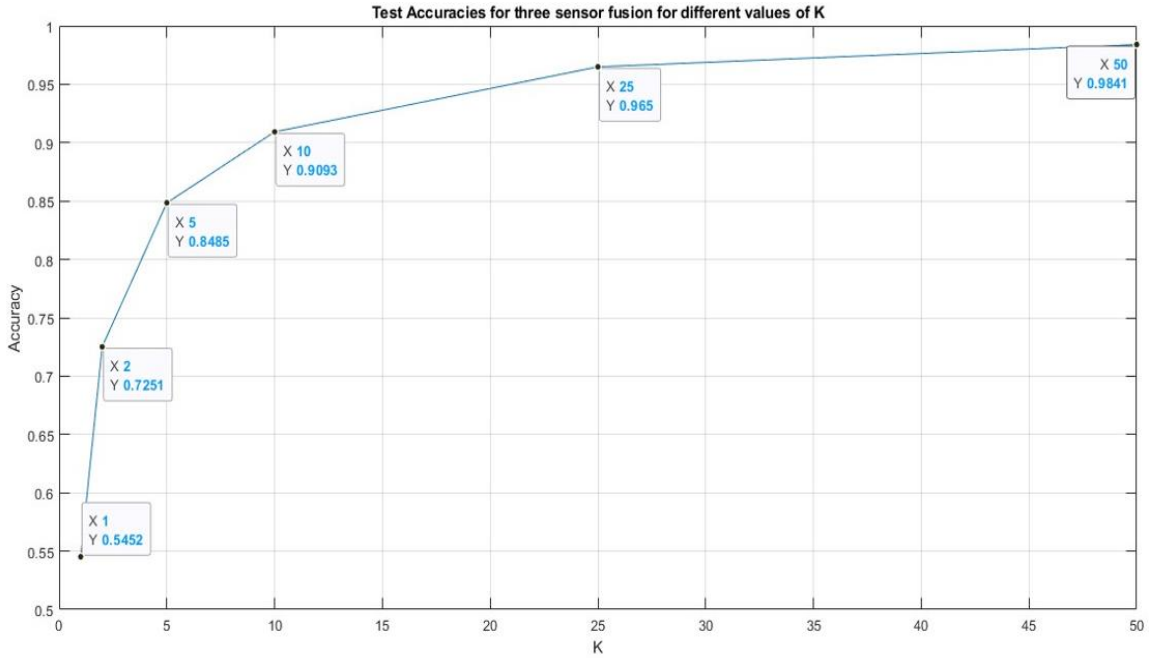

Figure 19: Test Accuracy for three-sensor unimodal-fusion network for different K

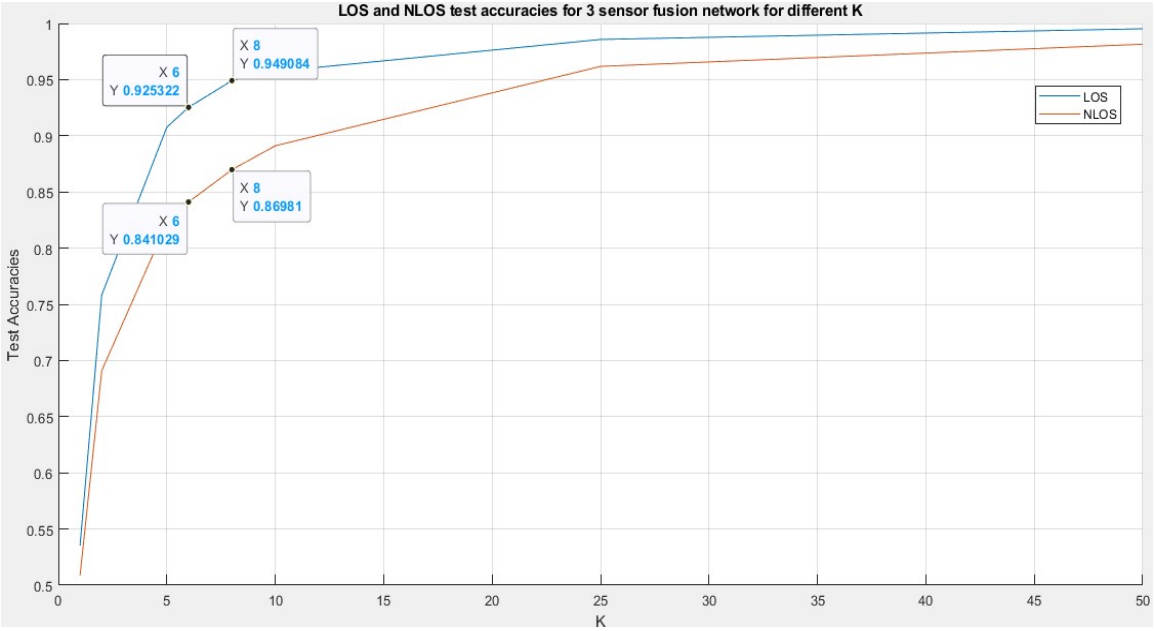

Figure 20: Test accuracy for LOS and NLOS cases for three-sensor unimodal-fusion network

Further illustrating the model's predictive accuracy, Figure 20 highlights the accuracies between LOS and NLOS samples in the test dataset, with LOS accuracies for K = 6 and K = 8 surpassing NLOS accuracies by 9.5% and 9.1%, respectively. These test accuracies, achieved using all three sensor modalities, align with the findings of Batool et

al. [9] in their research on leveraging out-of-band sensor information to enhance the beamforming process. Consequently, this study successfully benchmarks the utility of a deep neural network in facilitating ray tracing paths and accurately predicting the top-K beamforming pairs, leveraging the combined strengths of GPS, LiDAR, and camera image modalities.

5.1.3 Performances of Fusion Framework using LiDAR and GPS Unimodal

Following the successful demonstration of using all three sensor modalities to accurately predict top-K beamforming pairs, attention in section 5.1.1 shifted towards the exceptional performance of LiDAR as a standalone sensor due to its proficiency in predicting beamforming pairs. This insight led to a focused experiment that combines only LiDAR and GPS sensors, motivated by the goal of potentially minimizing resource usage and computational processing time. The complexity and high computational demand of processing camera images, primarily due to the object identification tasks in preprocessing, present an opportunity for efficiency gains if LiDAR and GPS alone can yield comparable results. To facilitate this experiment, adjustments were made to the initial layer of the fusion network to accept only two feature vectors, derived from LiDAR and GPS, as inputs. This modification is encapsulated in equation (5.1), which simplifies the input to two concatenated vectors:

$$Z = [\, Z_c, Z_L] \in R^{2*d} \qquad (5.1)$$

Here, Zc represents features from the GPS modality, and Z_L denotes features extracted from LiDAR.

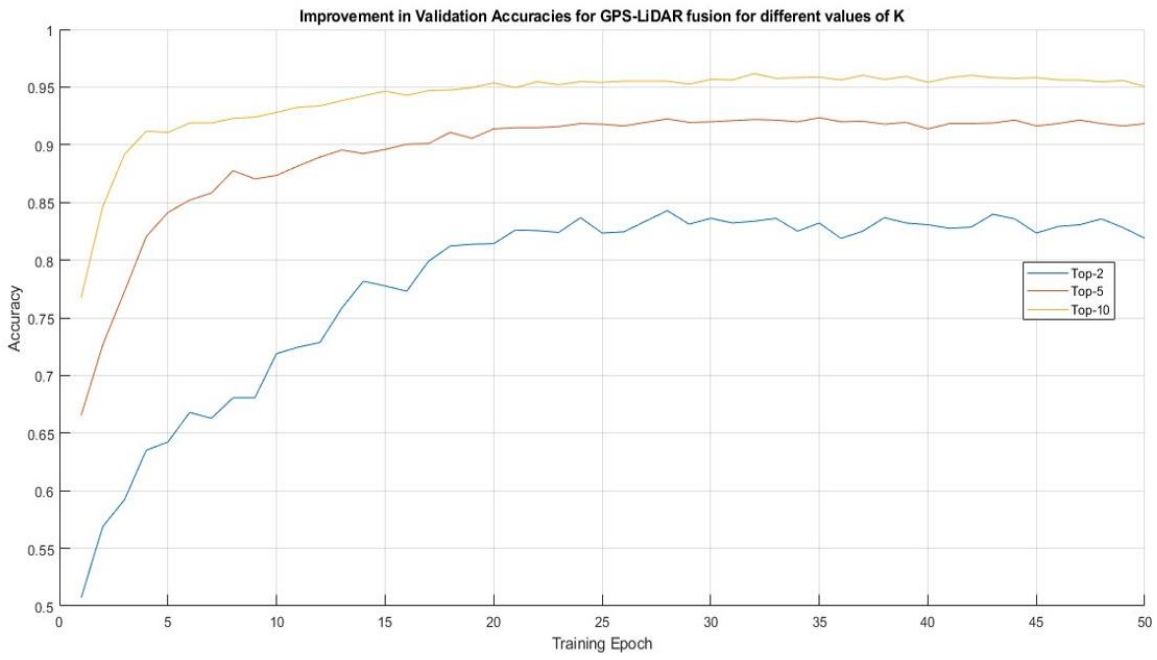

Figure 21: Improvements in Validation Accuracy for GPS-LiDAR unimodal-fusion network for different values of K

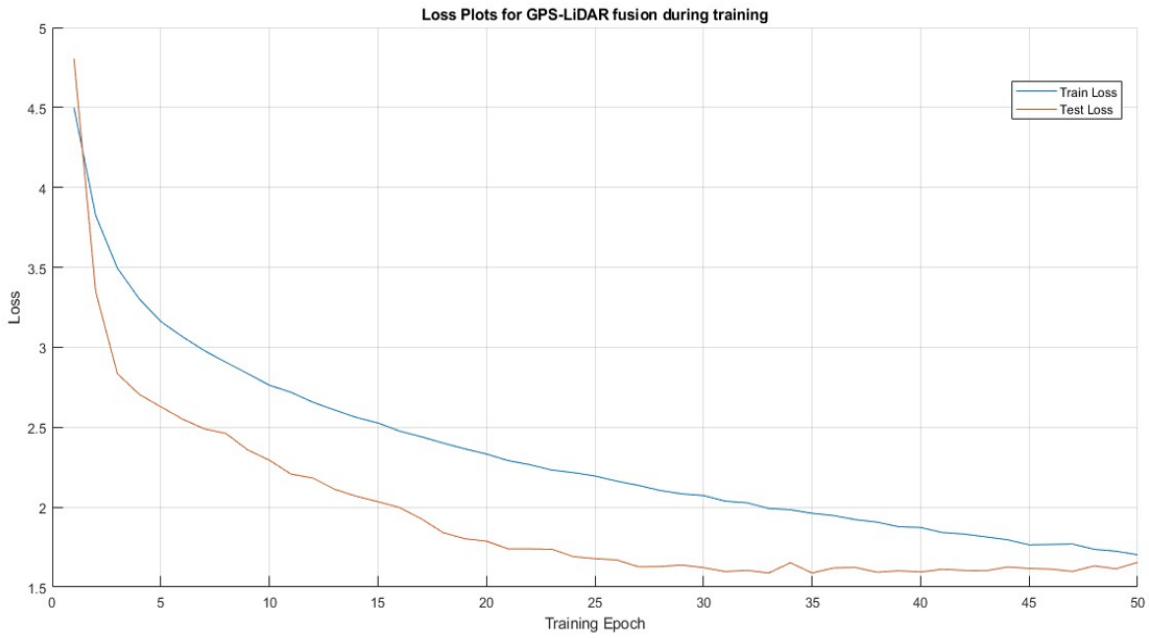

Figure 22: Plots of train and test loss for GPS-LiDAR fusion

Table 6: Comparison of Validation Test Accuracy for GPS and LiDAR unimodal-fusion network

	Top 1	**Top 2**	**Top 5**	**Top 10**	**Top 25**	**Top 50**
Validation Accuracy	0.62	0.82	0.92	0.96	0.98	0.99
Test Accuracy	0.55	0.73	0.85	0.91	0.97	0.98

The outcomes of this streamlined approach are showcased in Figure 21, which illustrates enhancements in validation accuracy utilizing the two-sensor fusion network. Specifically, the fusion network achieves validation accuracies of 0.82, 0.92, and 0.96 for K values of 2, 5, and 10, respectively. Figure 22 shows the convergence of both the test loss and training loss which are continuously reducing highlighting the model's ability to learn and predict. Figure 23 details the network's overall test accuracy across various Top K values (1, 2, 5, 10, 25, 50), with Table 6 consolidating these validation and test accuracy results. The results demonstrate that the performance of the fusion network, integrating only GPS and LiDAR, is on par with the results obtained when incorporating all three sensor modalities. Additionally, Figure 24 reveals the accuracy disparities between LOS and NLOS scenarios in the test dataset, with LOS accuracies for K = 6 and K = 8 outperforming NLOS by 9.5% and 9.1%, correspondingly.

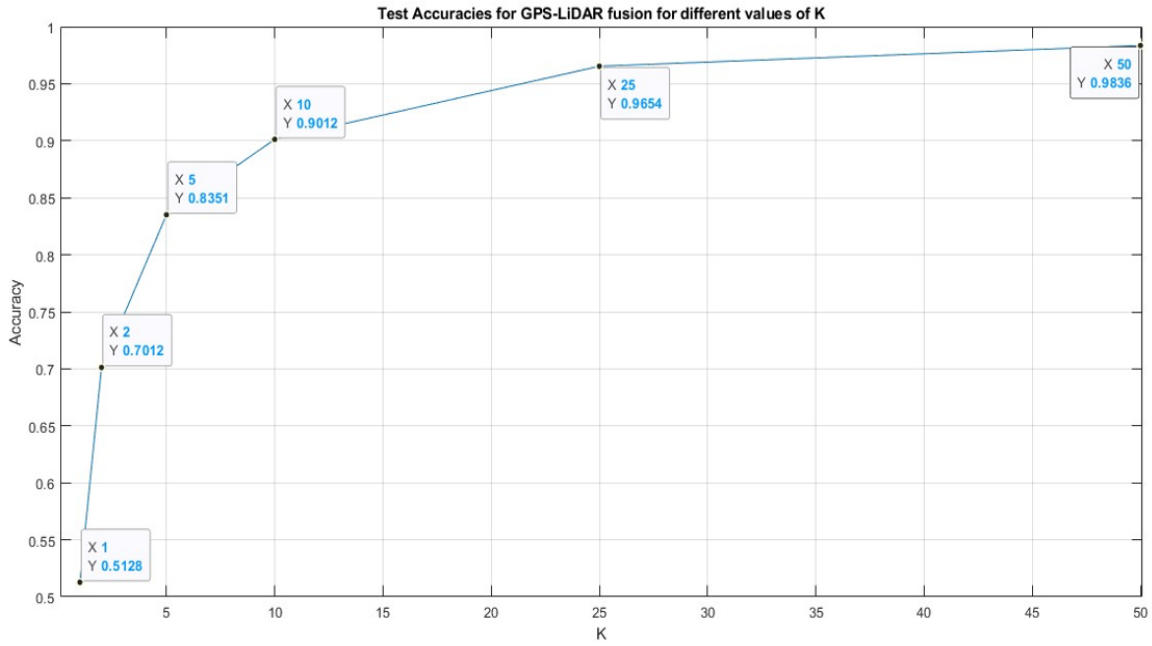

Figure 23: Test Accuracy for GPS-LiDAR fusion network for different K

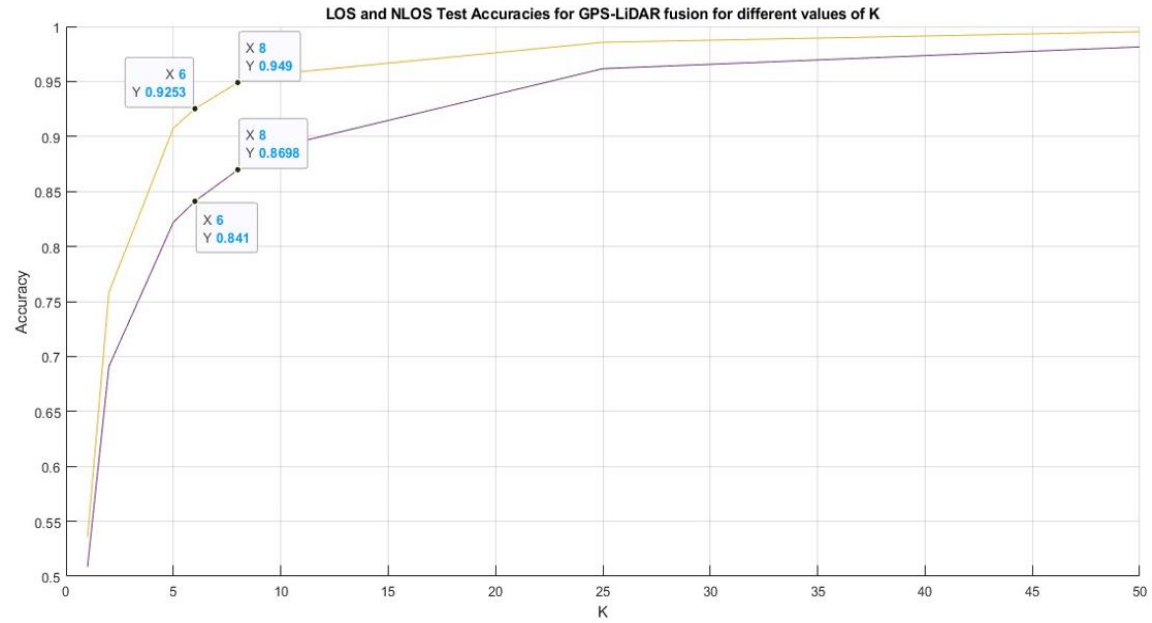

Figure 24: Test Accuracy for LOS and NLOS cases for GPS-LiDAR fusion network

The ensuing section will offer a comparative analysis between the dual-sensor (GPS and LiDAR) fusion network and the original tri-sensor fusion setup, further discussing the implications and benefits of the streamlined sensor integration strategy.

5.1.4 Comparison of LiDAR unimodal with 3 sensor unimodal fusion network and GPS-LiDAR unimodal fusion network

This section consolidates the outcomes of the experiments conducted with the LiDAR unimodal network, the three-sensor fusion network, and the GPS-LiDAR fusion network. We conclude this section with a qualitative discussion that paves the way for further exploration into the deployment of distributed architecture and its practical implications. The performance metrics from these experiments are detailed in Tables 7 and 8, presenting the validation and test accuracies respectively, while Table 9 showcases the throughput ratios.

Table 7: Validation accuracy for the three experiments studied

	Top 1	**Top 2**	**Top 5**	**Top 10**	**Top 25**	**Top 50**
LiDAR unimodal network	0.66	0.83	0.92	0.95	0.98	0.99
3-sensor fusion network	0.66	0.85	0.93	0.96	0.98	0.99
GPS-LiDAR fusion network	0.68	0.85	0.91	0.95	0.98	0.99

Table 8: Test accuracy for the three experiments studied

	Top 1	**Top 2**	**Top 5**	**Top 10**	**Top 25**	**Top 50**
LiDAR unimodal network	0.54	0.73	0.85	0.91	0.96	0.98
3-sensor fusion network	0.51	0.70	0.84	0.90	0.97	0.98
GPS-LiDAR fusion network	0.55	0.73	0.85	0.91	0.97	0.98

Table 9: Throughput ratios for the three experiments studied

	Top 1	**Top 2**	**Top 5**	**Top 10**	**Top 25**
Lidar unimodal network	0.72	0.86	0.93	0.97	0.99
3- sensor fusion network	0.70	0.83	0.92	0.96	0.99
GPS-LiDAR fusion network	0.74	0.87	0.94	0.97	0.99

The LiDAR unimodal network demonstrates steady improvement in accuracy with higher Top K values, reaching almost perfect scores for larger K. This indicates its reliable performance in scenarios where beamforming pairs are extensively explored. Three-sensor fusion network follows a similar pattern to the LiDAR unimodal network, it slightly edges out in accuracy for mid-range K values. This suggests that integrating additional sensor modalities can provide marginal benefits, particularly in the middle range of complexity. The GPS and LiDAR fusion network stands out for achieving the highest validation accuracy across a range of K values, indicating its superior performance and robustness in handling diverse scenarios.

The GPS and LiDAR fusion network not only excels in validation accuracy but also in test performance and throughput efficiency, especially notable at higher K values. This suggests a strong ability to manage the complexities of real-world scenarios with greater effectiveness and efficiency. The enhanced performance of the GPS and LiDAR fusion

network can be attributed to the complementary strengths of these two modalities. The fusion of GPS and LiDAR data adds a layer of depth to the model's understanding, leveraging LiDAR's precise environmental mapping and GPS's broad locational context attributing to a synergistic effect of combining GPS and LiDAR features. However, the introduction of camera images does not always contribute positively, possibly due to the complex nature of image data and the difficulty in extracting relevant features without introducing a higher convergence error floor.

Despite the strengths of GPS and LiDAR fusion, it's crucial to acknowledge the conditional limitations of LiDAR, such as its vulnerability to certain environmental factors, as discussed in [9]. In scenarios where LiDAR's efficacy is reduced, the inclusion of camera data along with GPS can still provide a fallback mechanism to maintain system performance, albeit at a reduced capacity. This nuanced understanding underscores the importance of a balanced approach in selecting sensor modalities for optimal performance across varying climatic conditions and requirements which is beyond the current scope of study.

A deeper analysis into the performance dynamics of the three-sensor fusion network compared to the LiDAR unimodal network is provided through Figure 25, focusing on validation accuracy across various K values Initially, LiDAR's unimodal network leads in accuracy due to fewer parameters needing training, which benefits early convergence. However, the three-sensor fusion network eventually surpasses LiDAR's performance as it reaches later stages of training, highlighting the advantage of integrating multiple sensor modalities for comprehensive learning.

Additionally, Figure 26 emphasizes the superior and consistent performance of the GPS-LiDAR fusion network in NLOS test accuracy across all models, particularly noting that LiDAR unimodal accuracy plateaus for K values above ~18 at around 0.96. This plateau suggests that beyond a certain point, the feature embeddings extracted from LiDAR data alone cannot further enhance NLOS test accuracy, underlining the benefits of fusion approaches in overcoming individual sensor limitations.

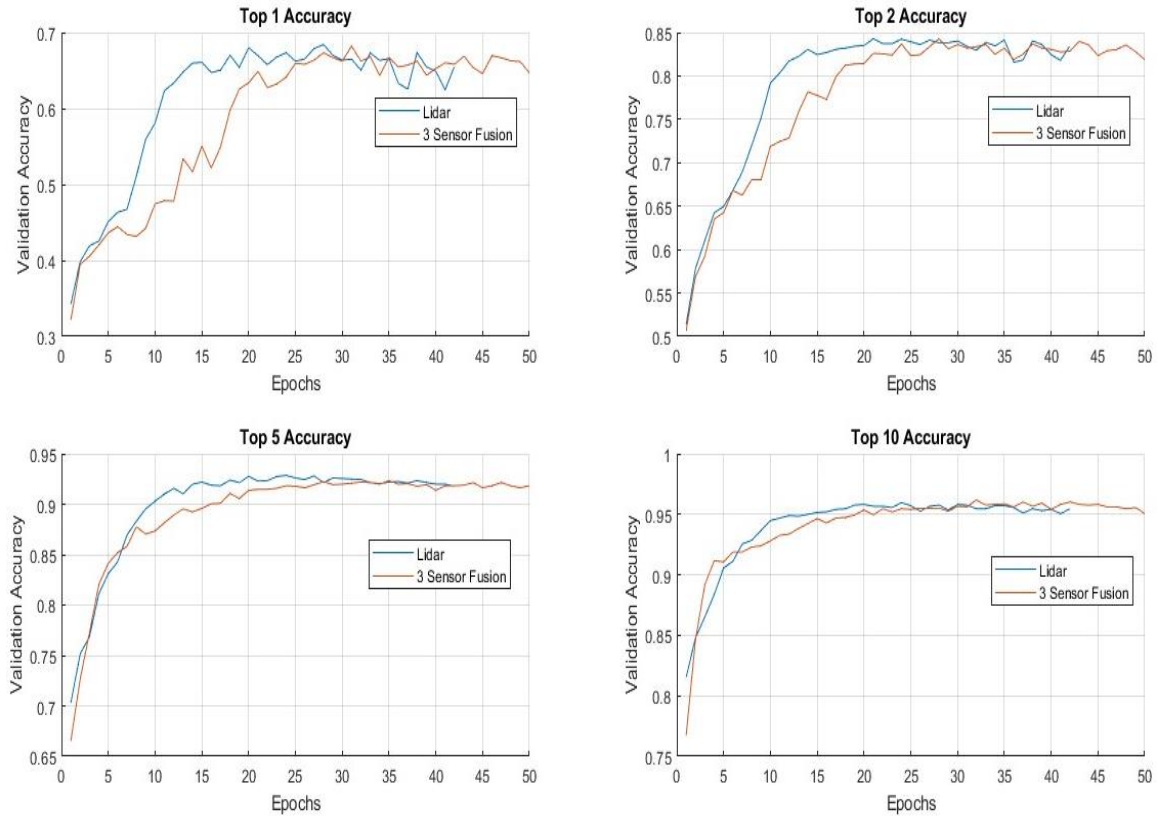

Figure 25: Plots showing the improvements in validation accuracy for unimodal LiDAR and 3-sensor fusion network

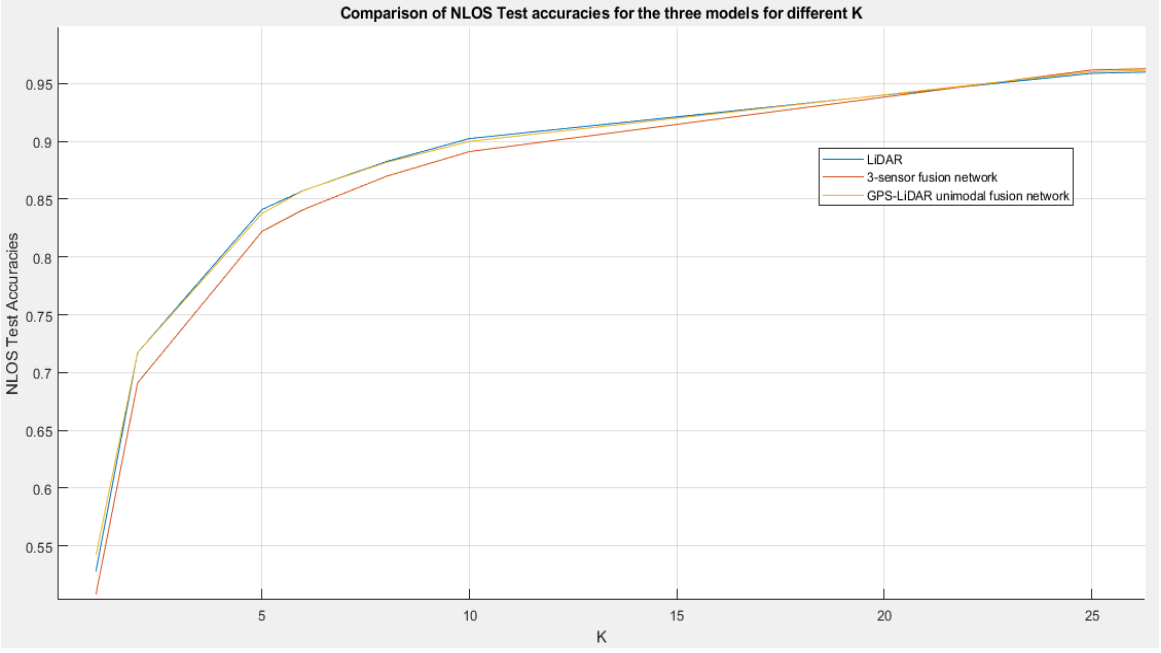

Figure 26: Comparison of NLOS Test accuracy for the three models for different K

5.1.5 Throughput Ratio versus End-to-end Latency for Centralized Architecture

Throughput ratio is a comparative metric which identifies the performance of using top-K beamforming pairs predicted by the DL model in comparison with the standard beam sweeping algorithm across all the possible beamforming pairs. Equation (2.6) calculates the total duration required to conduct a comprehensive beam search using the conventional beam search algorithm in 5G-NR, which amounts to approximately 145ms for analyzing 256 beam pairs, for the Raymobtime datasets. The application of equation (4.16) enables the graphical representation of the end-to-end latency period for the three-sensor unimodal-fusion network as shown in Figure 27. To better understand each component of equation (4.16), it is represented here where the total latency time in a centralized architecture is given by $T_{Total}(K) = T_{process} + T_{uplink} + T_{predict} + T_{downlink} + T_{sweep}(K)$. $T_{process}$ is the time required to pre-process that input data at the vehicle which takes 1.30 ms on average, which is mainly caused by the time to process images from the camera [9]. A sample of pre-processed LiDAR, GPS and camera images has a size of ~4Kbytes. The throughput of sub-6GHz channels which follow the standard 802.11p protocol is in the range of 3-27 Mbps. At this rate, T_{uplink} is the time taken to share the pre-processed data to the MEC over the sub-6GHz uplink channel, which takes 1.332 ms on average. $T_{predict}$ is the time taken for the DL model to predict the top-K beamforming pairs which takes around 0.37 ms. $T_{downlink}$ is the time taken to share the top-K beamforming pairs back to the vehicle over the sub-6GHz downlink channel, which takes 1.03 ms on average. $T_{sweep}(K)$ is the time taken to sweep through "K" beamforming pairs given by equation (2.7). In this scenario, the initial four components of equation (4.16) are estimated at around 3.662 ms.

As we can conclude from Figure 27, the improvements in end-to-end latency time achieved by the DL model when restricting to a top-K beamforming pairs is significant. For a K value of 80, the network achieves a throughput ratio of nearly 0.998 with a latency reduction of 68%, clocking in at 46.1ms compared to 145ms when using the standard beam sweeping search across all the 256 pairs. When K is set to 10, the latency drops dramatically to 5.224ms, a 96% decrease, while maintaining a good throughput ratio of 0.96. Additionally, to attain a throughput ratio of 0.99, setting K to 27 results in a latency of 7.88ms, indicating a 94.5% reduction.

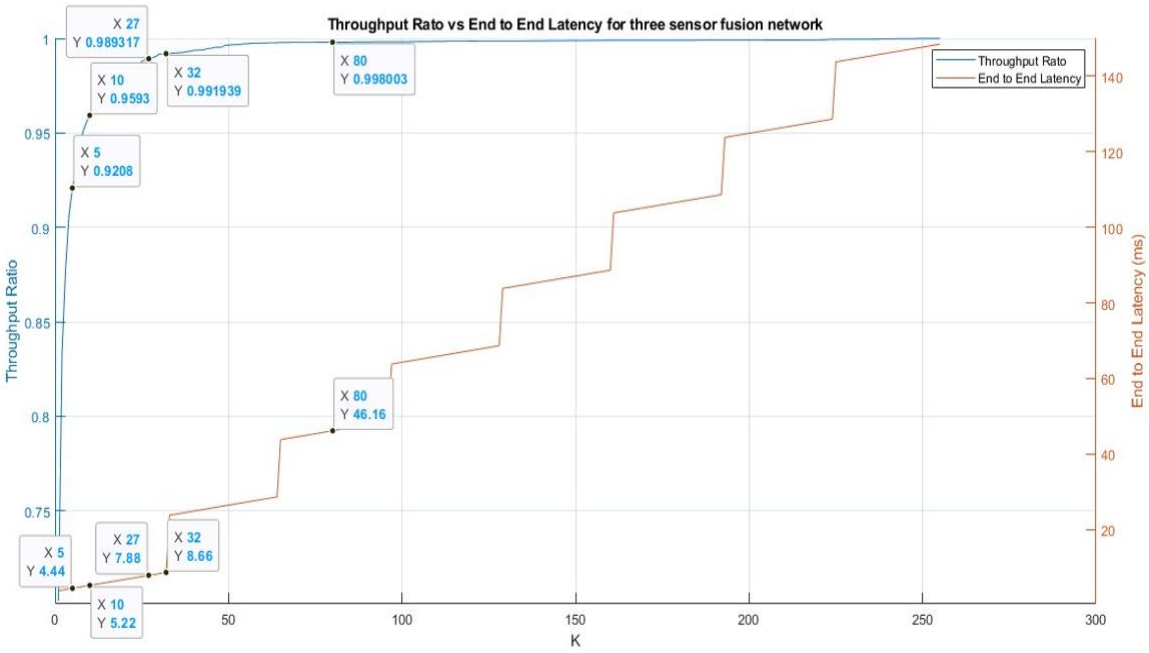

Figure 27: Plot showing throughput ratio and End-End latency time for proposed three-sensor unimodal-fusion network

The GPS-LiDAR unimodal-fusion network presents even more impressive latency reductions as shown in Figure 28. The processing time ($T_{process}$) for LiDAR pre-processing can be approximated to be negligible [9]. The first four components of Equation (4.16) are estimated around 2.362 ms for this configuration. With K set to 70, the system reaches a throughput ratio of 0.998 at a latency of 43.3 ms, marking a 70% reduction in latency times. At a K value of 10, the latency further decreases to 3.9 ms, a 97% reduction, while achieving a throughput ratio of 0.97. To accomplish a throughput ratio of 0.99, K is adjusted to 20, which correlates with a latency period of 5.48 ms, translating to a 96.2% reduction in latency times.

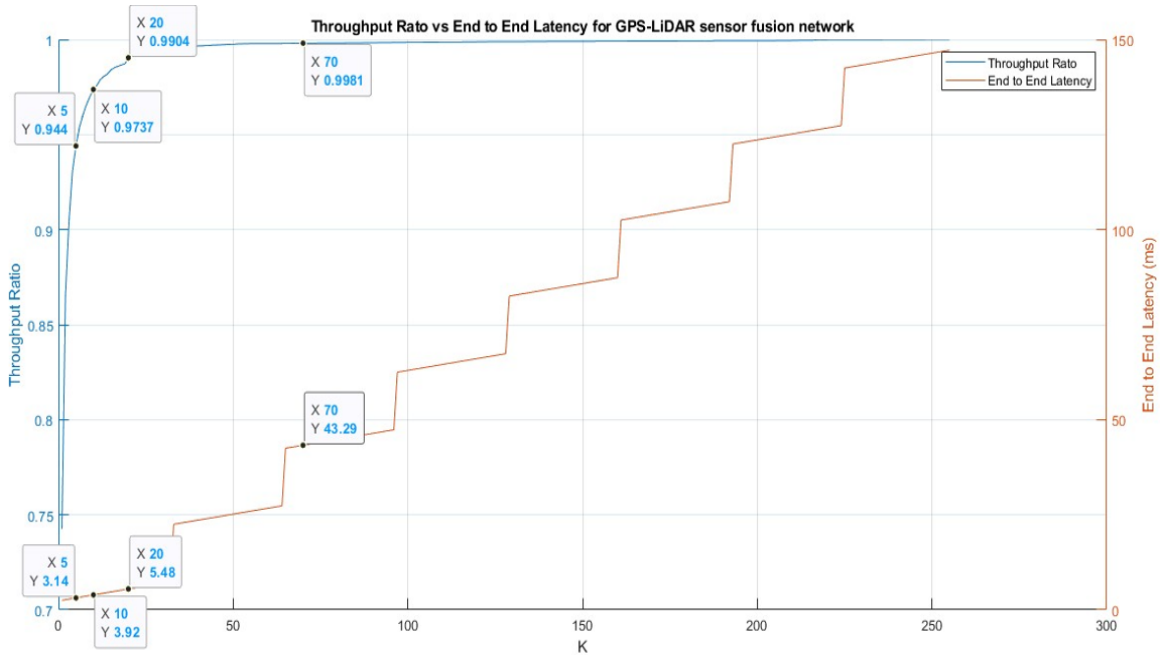

Figure 28: Plots showing throughput ratio and end-end latency for proposed GPS-LiDAR unimodal-fusion network

These findings from Figure 27 and Figure 28 illustrate the substantial efficiency gains with a significant reduction in latency periods achieved by using multi-modality sensors to relate mmWave ray tracing leveraging a DL model and predicting the top-K beamforming pairs. Considering practical condition which require the highest throughput, restricting the standard beam sweeping methods to a subset of top-K beam pairs can achieve a reduction in latency time unto 96% while maintaining a throughput ratio of 0.99.

5.2 Experiments on Distributed Architecture

This section outlines the experiments designed to evaluate the distributed architecture for improving beamforming process previously introduced in section 4.3 and 4.4. This innovative framework employs a network of multiple vehicles/clients to collaboratively train a global model. The advantage of this approach lies in its ability to enhance the global model's generalization capabilities, enabling it to effectively adapt to new environmental scenarios not encountered during training. The distributed architecture also reduces the usage of sub-6GHz channels compared to the centralized architecture. The structure of the individual unimodal networks and the fusion network remains consistent with the descriptions provided in section 4.1 and their network architecture is presented in Appendix A, with a notable modification in the final layer. To accommodate the Infocom FLASH dataset, which comprises only 64 beamforming pairs due to its coverage of a more limited sector area, the final SoftMax layer is adjusted to have 64 nodes.

The preceding sections detailed the performance evaluations of the centralized architecture using the respective unimodal and fusion networks. A key takeaway from these analyses is the superior performance of the GPS-LiDAR fusion networks. This finding is particularly influential in guiding the experimental approach for the distributed architecture, prompting a focus on similarly employing only the GPS and LiDAR fusion networks. The objective is to ascertain the use of distributed framework, assess its efficiencies and confirm its advantages. The forthcoming simulation experiments are thus tailored to explore the effectiveness of the GPS and LiDAR fusion approach within this distributed setting.

Table 10: Simulation settings for experiments conducted for Distributed Architecture

Experiment #	**Experiment Description**	**Global Training Rounds**	**Training Settings**
1	Unbiased Client Selection	200	Training Epochs = 100 Learning Rate = 0.0001. Batch Size = 32. Adam Optimizer with beta1 = 0.9 and beta2 =0.999 Data Shuffle enabled Infocom FLASH dataset with GPS, LiDAR and RF Labels are used
2	Max Loss Biased Client Selection with m = 3	300	
3	Max Loss Biased Client Selection with m = 4	200	
4	Max Loss Biased Client Selection with m = 5	200	
5	Heuristic MAB Biased Client Selection with Υ = 0.3	300	
6	Heuristic MAB Biased Client Selection with Υ = 0.7	350	
7	Heuristic MAB Biased Client Selection with Υ = 0.7	400	

We first begin our investigation by conducting simulation experiments on the distributed architecture, focusing on the unbiased client selection strategy introduced in section 4.4. These experiments serve to compare our findings with the pivotal study by Batool et al., in [18], which utilized a distributed architecture within a federated learning framework using unbiased client selection strategy. This experiment will also help us benchmark the Infocom FLASH dataset [19], which is the sole dataset to train and test a federated learning model to predict top-K beamforming pairs. Following this, we use the MaxLoss based client selection strategy to conduct experiments employing different client subset sizes (m). These experiments primarily help us to fix the client subset size to form a generalized perspective when the total number of participating clients are increase in future for practical deployment. In the subsequent section, we conduct experiments to validate the MAB based client selection strategy and conduct two case studies to validate the in-filed training capabilities by employing this strategy. The final section presents an analysis on the reduction in usage of sub-6GHz channels for different client sizes compared to the unbiased client selection strategy. The Table 10 presents the simulation

settings for all the experiments conducted for in this section using a distributed architecture.

5.2.1 Unbiased Client Selection Strategy

The first experiment within this study leverages the unbiased client selection mechanism, aligning with the studies conducted in section 4.4. This approach mirrors the methodologies applied in the research by Batool et al. in their FLASH architecture [18], serving as a critical reference point. The essence of this experiment is to further validate the Infocom FLASH dataset's applicability in distributed architectural analysis [19]. Figure 29 highlights the progression of test accuracies achieved by the global model employing the GPS-LiDAR fusion network within a distributed framework. By adopting an unbiased selection method, the model integrates contributions from all ten clients featured in the Infocom FLASH dataset throughout each iteration of global model training. Conducted over 200 training rounds, the global model's performance reached test accuracies for Top 1, 2, and 5 of 0.61, 0.81, and 0.96, respectively. These outcomes not only align with the findings of Batool et al. using the FLASH architecture but also underscore the Infocom FLASH dataset's reliability and relevance in dissecting distributed architecture's dynamics. Additionally, Figure 30 illustrates that the global test loss stabilized at approximately 1.69, marking the experiment's culmination point. The convergence of test loss at 1.69 marks a good training performance given 64 classes for prediction with a baseline cross-entropy loss of $Log_e(64)$ (= 4.158). This means that, without any learning or patterns recognition, the expected loss is around 4.158. Therefore, a converged loss of 1.69 is significantly lower than the baseline, indicating that our model is learning and performing much better.

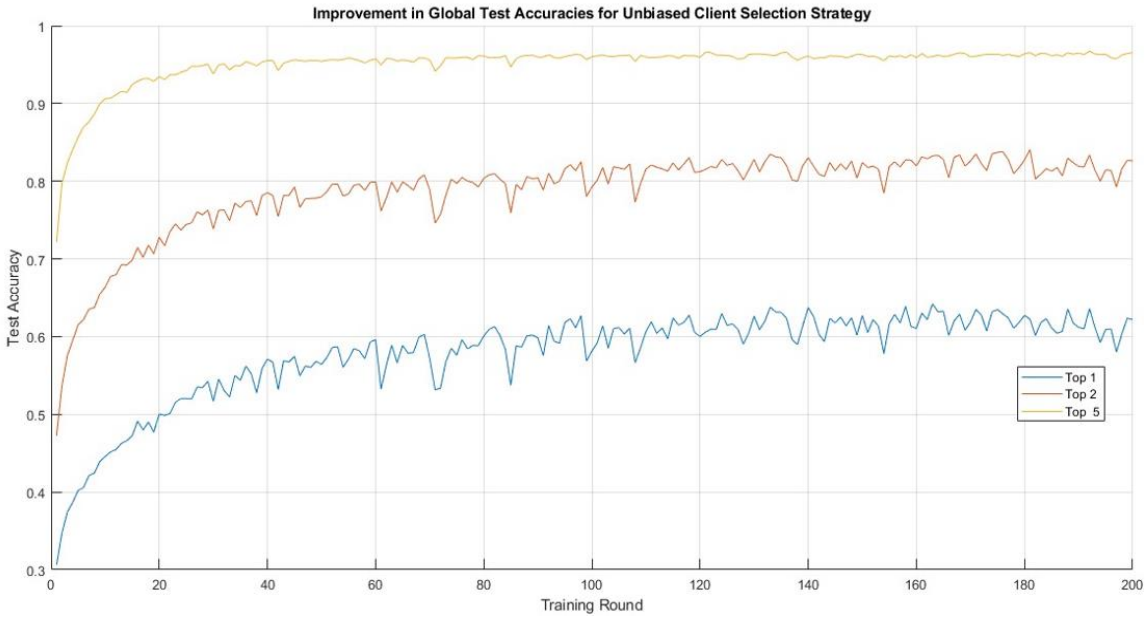

Figure 29: Improvement in global test accuracies for unbiased client selection strategy

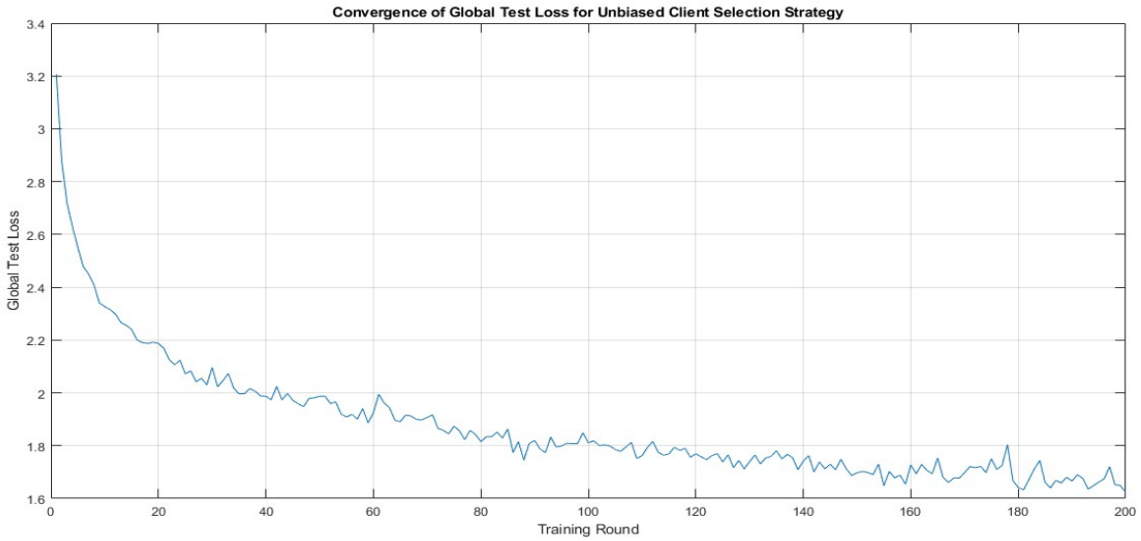

Figure 30: Convergence of global test loss for unbiased client selection strategy.

5.2.2 MaxLoss based Client Selection Orchestration

MaxLoss based client selection strategy is a method focused on selecting clients that have shown the highest local test losses in the preceding training cycle. This strategy aims to prioritize the inclusion of clients that potentially contribute the most towards model improvement in subsequent training round. The experiments detailed in this segment evaluate the impact of varying client sizes on the model's performance, exclusively utilizing the LiDAR and GPS fusion networks.

The first set of experiments, depicted in Figures 31 and 32, explored the effect of choosing the client subset size (m) to be three. Over 300 training rounds, this arrangement led the global model to achieve test accuracies of 0.60, 0.79, and 0.96 for Top 1, 2, and 5, respectively, with the global test loss stabilizing at a value of 2.

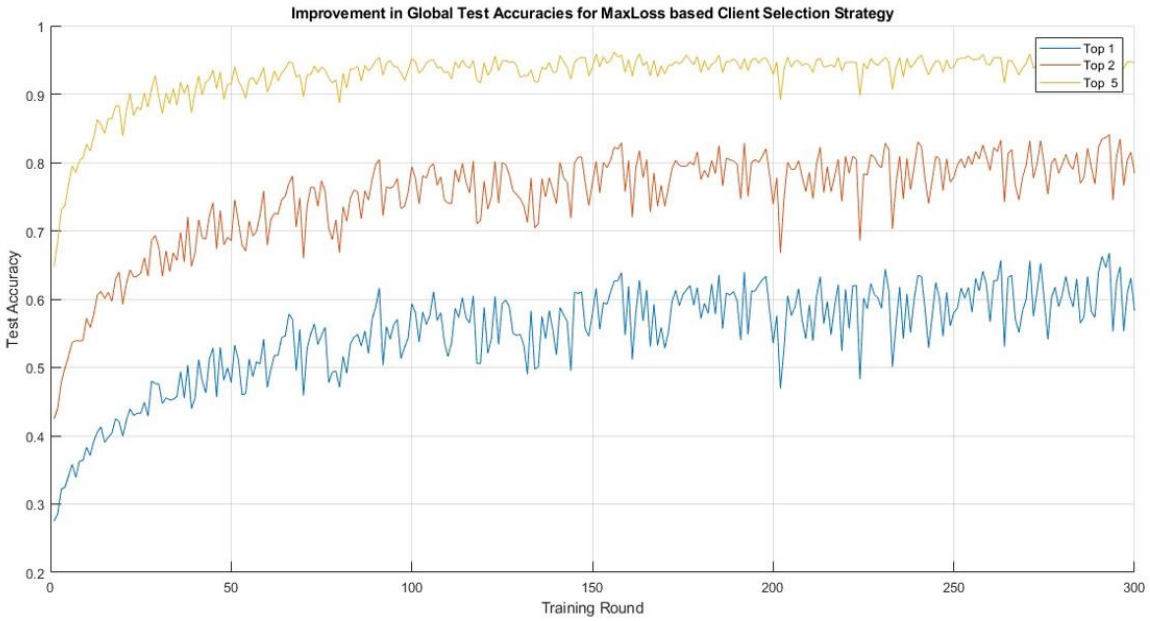

Figure 31: Improvement in global test accuracies for MaxLoss based client selection with a client size of three

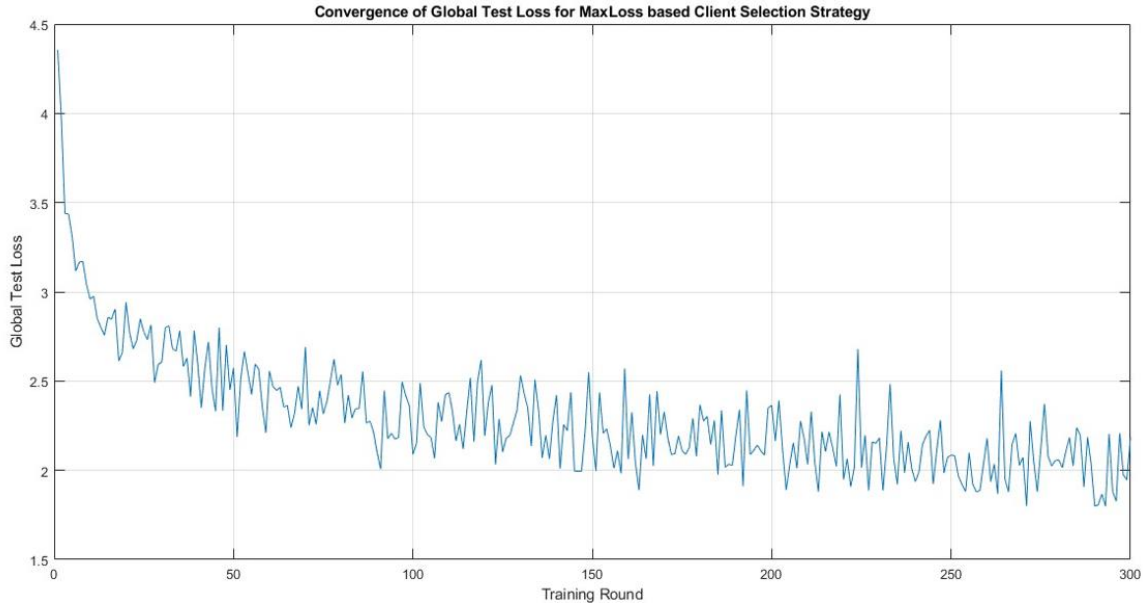

Figure 32: Convergence of global test loss with a MaxLoss based client selection strategy with a client size of three

In the following simulation experiment, the client subset size was scaled up to include four clients to participate in training. This trial was conducted for 200 training rounds and the results are illustrated in Figure 33 and Figure 34. There is slight improvement in the global model's test accuracies from the previous simulation experiment using three

clients. Accurately , the top 1,2 and 5 global test accuracies reach 0.61, 0.80, and 0.96 respectively. The global test loss converges at 1.85.

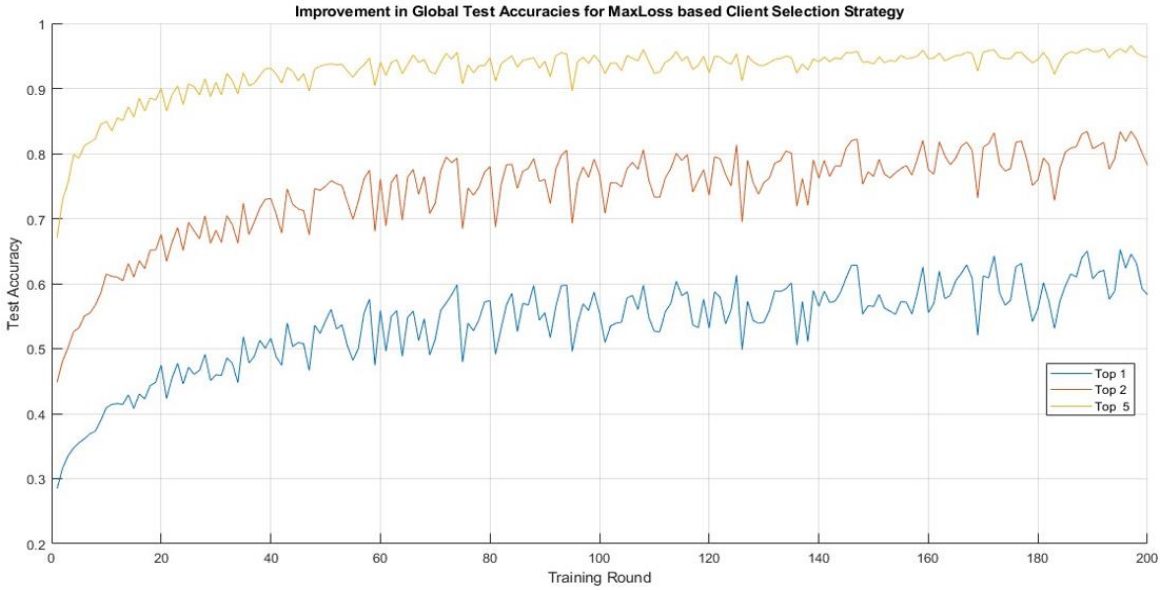

Figure 33: Improvement in global test accuracies for MaxLoss based client selection with a client size of four

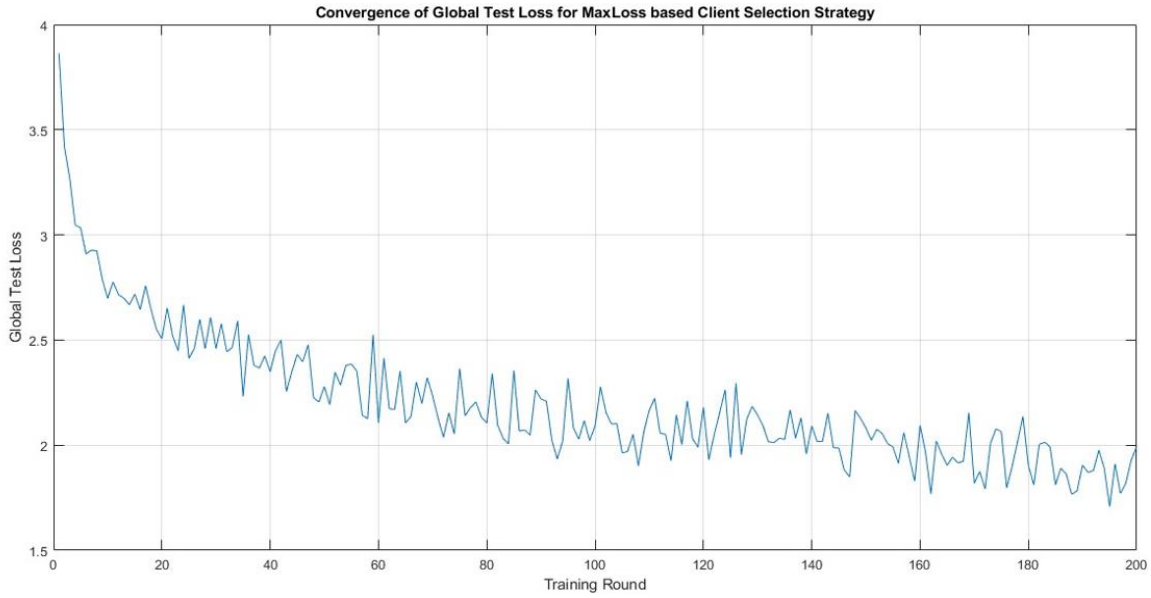

Figure 34: Convergence of global test loss with a MaxLoss based client selection strategy with a client size of four

Another valid inference that can be drawn is the reduction in the training rounds from the previous experiment from 300 to 200. As we increase the client subset size (m), the generalizability of the global model increases and is able to learn more at a lesser training

round. But the downside of this results in a greater number of clients trying to use the sub-6GHz channels increasing the load on them. Further expansion to a client subset size to five is illustrated through Figures 35 and 36, maintained the trend of incremental improvements. This configuration is also run for 200 training rounds with the global test accuracies reaching values of 0.62, 0.81, and 0.96 for Top 1, 2, and 5, respectively, and the global test loss converging at 1.8. At this point, we refrain from increasing the client subset size any further as we would like to maintain the utilization of sub-6GHz channels at the highest of 50% less than the maximum. In the next section, we present these implications and the detailed usage of the sub-6GHz channels for different client subset sizes.

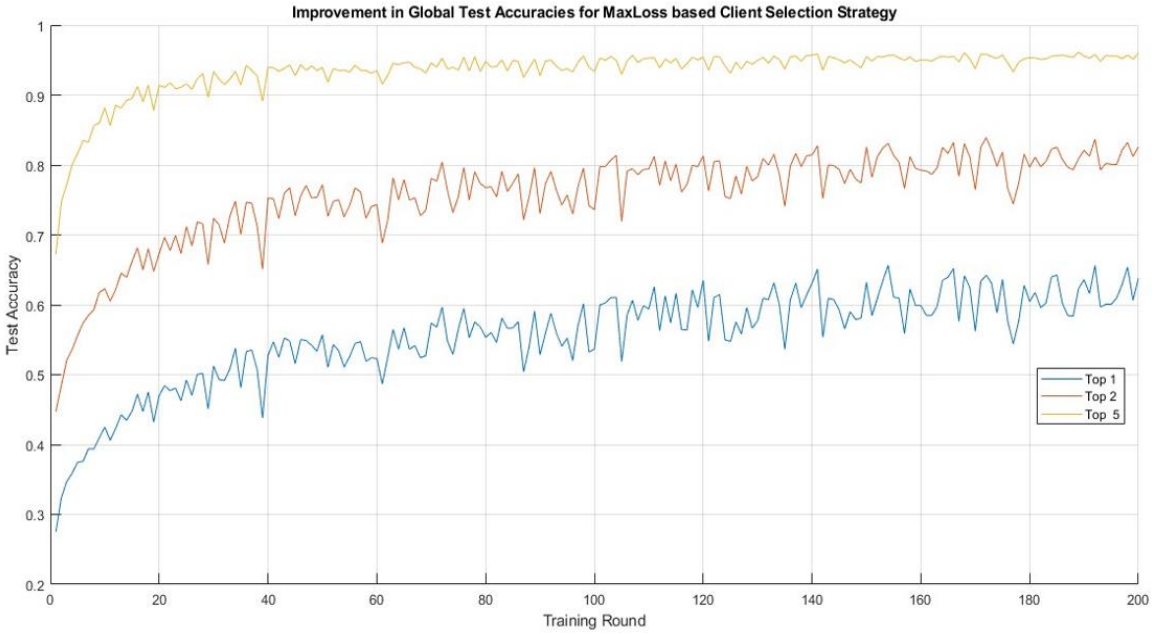

Figure 35: Improvement in global test accuracies for MaxLoss based client selection with a client size of five

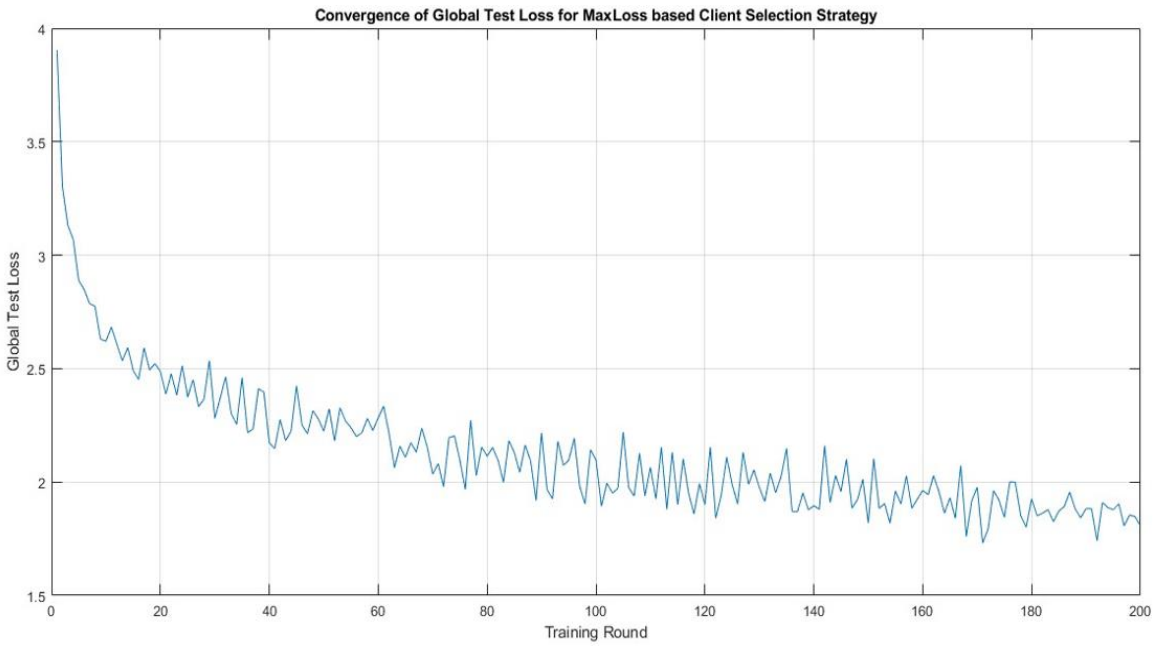

Figure 36: Convergence of global test loss with a MaxLoss based client selection strategy with a client size of five

Table 11: Summary of experiments conducted with MaxLoss based client selection strategy

Experiment	**Top 1 Accuracy**	**Top 2 Accuracy**	**Top 5 Accuracy**	**Global Test Loss**	**Average Training Time per training round (min)**
Client size = 3 (MaxLoss)	0.60	0.79	0.96	2	3.8
Client size = 4 (MaxLoss)	0.61	0.80	0.96	1.85	4.2
Client size = 5 (MaxLoss)	0.62	0.81	0.96	1.8	5.5
Client size = 10 (Unbiased Selection)	0.61	0.81	0.96	1.69	6.15

An aggregated view of these experiments is presented in Table 11, highlighting the relationship between client size and the efficiency of the learning process. A key observation is the positive impact of increasing client size on the convergence rates of global test loss. Specifically, Table 11 suggests that optimal client size might be approximately 40% of the total number of available clients. Adopting this ratio could potentially reduce network overhead on sub-6-GHz channels and expedite the convergence process by an impressive 15-32%. This series of experiments underscores the viability of the MaxLoss based client selection strategy, particularly in terms of enhancing federated learning systems' efficiency and effectiveness by judiciously selecting client subset sizes (m). The trend indicates a promising direction for future

deployments, aiming for a balanced client size that optimizes resource utilization and learning outcomes.

5.2.3 Analyzes of Communication Overheads using federated learning paradigm

Distributed architecture includes two communication downlinks and two uplinks, facilitating data exchange between the server (MEC) and the clients (vehicles). In this scenario, the weights for the GPS, LiDAR, and fusion models in the proposed GPS-LiDAR fusion network are reported to be 2.78MB, 3.73MB, and 6.23MB, respectively (refer section 5.3 for detailed calculation).

The Table 12 outlines the communication overheads associated with employing different numbers of clients and selection strategies. It highlights that for downlink 1, where the global model weights are disseminated to all 10 clients, the data transfer amounts to 127.4MB. Downlink 2, relevant only under a biased client strategy, incurs a negligible overhead of less than 0.001 MB, essentially serving as a notification to the selected clients for training participation. In uplink 1, a minuscule data packet (returning the local test loss) is less than 0.1 MB representing a "double" datatype value of 64 bytes, is sent from all clients to the server, a step omitted in the unbiased strategy. Uplink 2, which involves returning the trained weights to the server, matches the downlink 1 size of 127.42 MB in the unbiased strategy scenario.

Table 12: Communication overheads for choosing different number of clients and strategy

Client size	**Downlink 1 (MB)**	**Downlink 2 (MB)**	**Overall Downlink**	**Uplink 1**	**Uplink 2**	**Overall Uplink**	**Overall**
2 – biased strategy	127.4	0.001	127.401	0.1	25.48	25.64	153.041
3 – biased strategy	127.4	0.001	127.401	0.1	38.22	38.36	165.761
4 – biased strategy	127.4	0.001	127.401	0.1	50.96	51.06	178.461
10 – unbiased strategy	127.4	NA	127.4	NA	127.42	127.42	254.84

Analysis of Table 12 reveals that employing a biased client selection strategy can lead to a reduction in overall communication overheads by 30-40% compared to an unbiased approach. Notably, downlink 1's data transfer could potentially be halved to approximately 63.7 MB by adopting a probabilistic sampling strategy that updates specific networks as suggested in prior research [18].

Extending this discussion to a larger scale where the number of clients significantly exceeds 10, for instance, N=100, the communication overhead in the unbiased scenario would escalate to 2544 MB. Conversely, in a biased strategy, selecting a fraction of clients, m=c*N (with c being a fraction within the range [0,1], for example, c=0.4), would result in m=40 clients. This approach substantially reduces the uplink 2 communication overhead to 508 MB and, consequently, the overall communication overhead for 40 clients participating in each training round to 1783 MB reducing communication overheads by 30% compared to unbiased client selection strategy. Such a strategy can cut communication overhead by nearly 35% if we choose a fraction one third clients participating in training, showcasing the efficiency of selective client involvement in reducing bandwidth and resource consumption during the model training process.

5.2.4 Heuristic Multi Arm Bandit based Client Selection Orchestration

The Heuristic MAB based client selection strategy was developed in section 4.5.2. The experiments in this section are conducted using Algorithm 5 for different values of the hyperparameter "ϒ" in the range of $0 < \Upsilon < 1$. The experiments are conducted only on the GPS-LiDAR unimodal-fusion networks based on the conclusions drawn in the previous experiments with the client size 'm' is fixed at four. Figure 37 and Figure 38 presents the first experiment conducted with a hyperparameter value "ϒ" of 0.3. The experiment was run for 300 training rounds and the global model reaching a top-K (= 1, 2, 5) test accuracies of 0.65, 0.82 and 0.95 respectively and the global test loss converges at 1.8. Figure 39 represents the number of times each client is sampled to participate in training for a total of 300 rounds. Client '4' followed by clients '7' and '8' are sampled more often than others. Finally, the fairness factor calculated equals 0.924 which is relatively good considering the ideal case to be equal to 1 [28]. In the next two experiments we use a "ϒ" value of 0.5 and 0.7 and finally analyze these results comparatively.

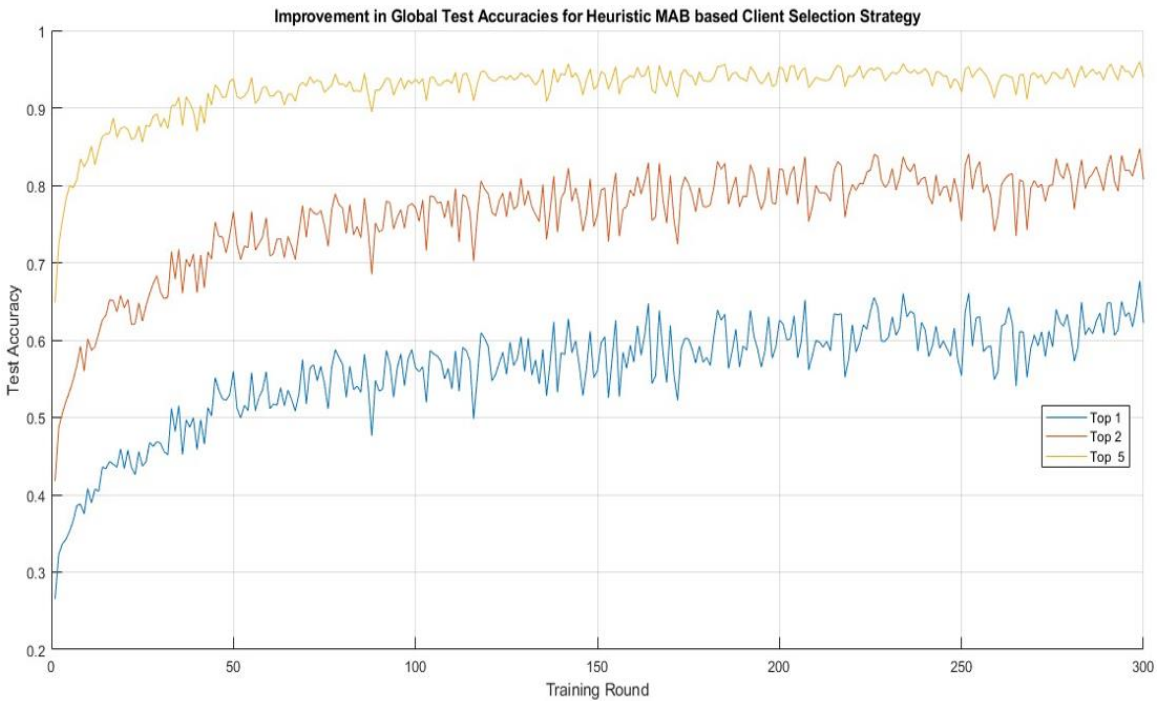

Figure 37: Improvement in global test accuracies for Heuristic MAB based client selection with a hyperparameter "ϒ" = 0.3

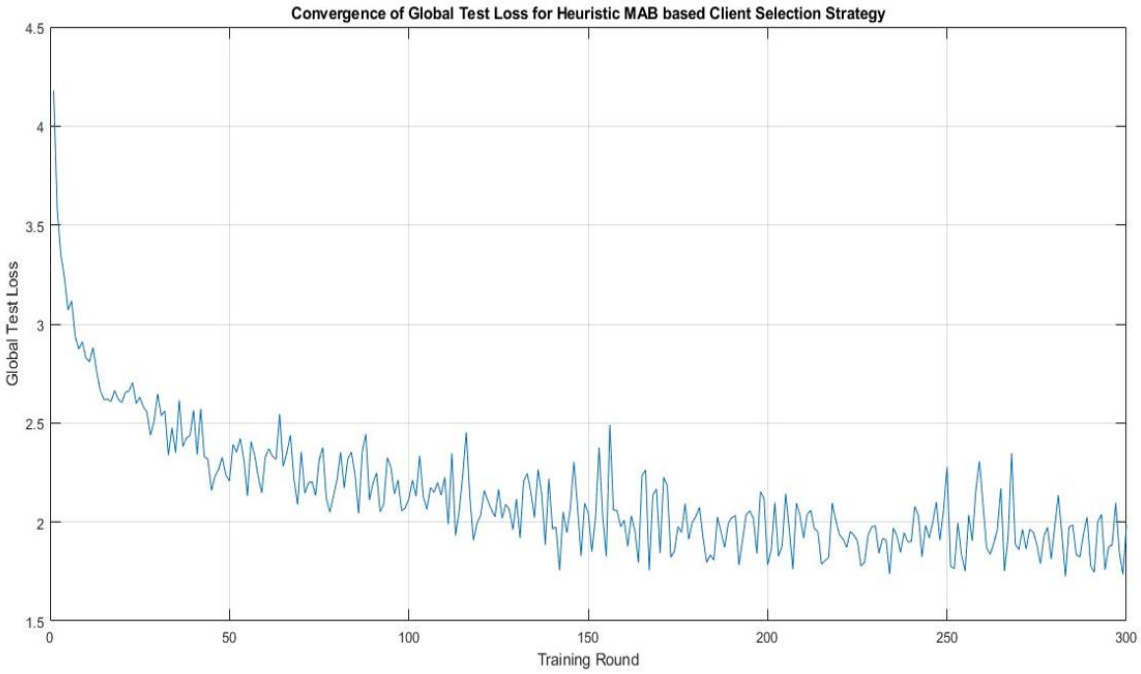

Figure 38: Convergence of global test loss with a Heuristic MAB based client selection strategy with a hyperparameter "ϒ" = 0.3

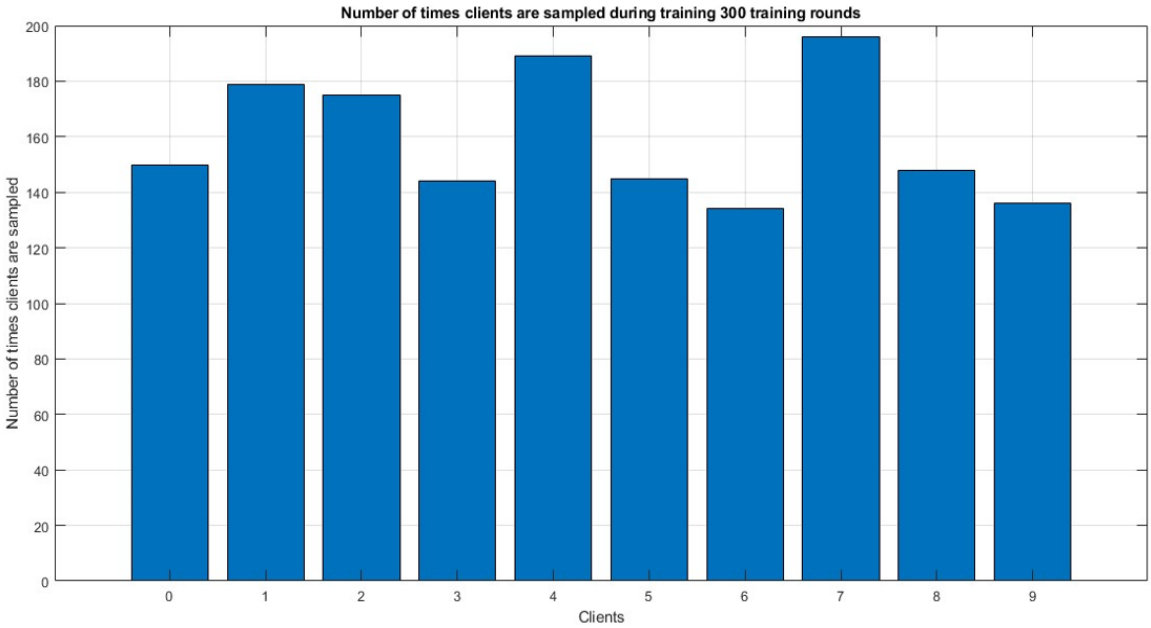

Figure 39: Number of times each client is sampled with a hyperparameter "ϒ" = 0.3

Figure 40 and Figure 41 presents the second experiment conducted with a hyperparameter value "ϒ" of 0.5. The experiment was run for 350 training rounds and the global model converges with a top-K (= 1, 2, 5) test accuracies of 0.59, 0.79 and 0.94 respectively and the global test loss converges at 1.9. Figure 42 represents the number of times each client is sampled to participate in training for a total of 350 training rounds. Client '1' is sampled more often than the others which is different from the previous experiment. Finally, the Fairness factor calculated equals 0.918 which is comparatively lesser than the previous case.

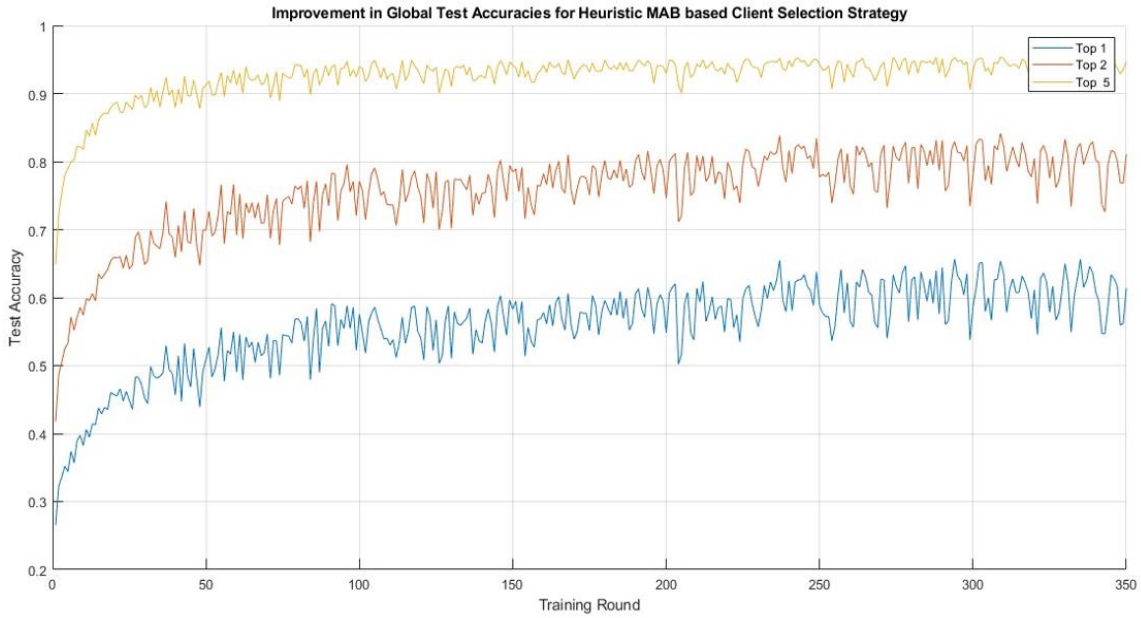

Figure 40: Improvement in global test accuracies for Heuristic MAB based client selection with a hyperparameter "ϒ" = 0.5

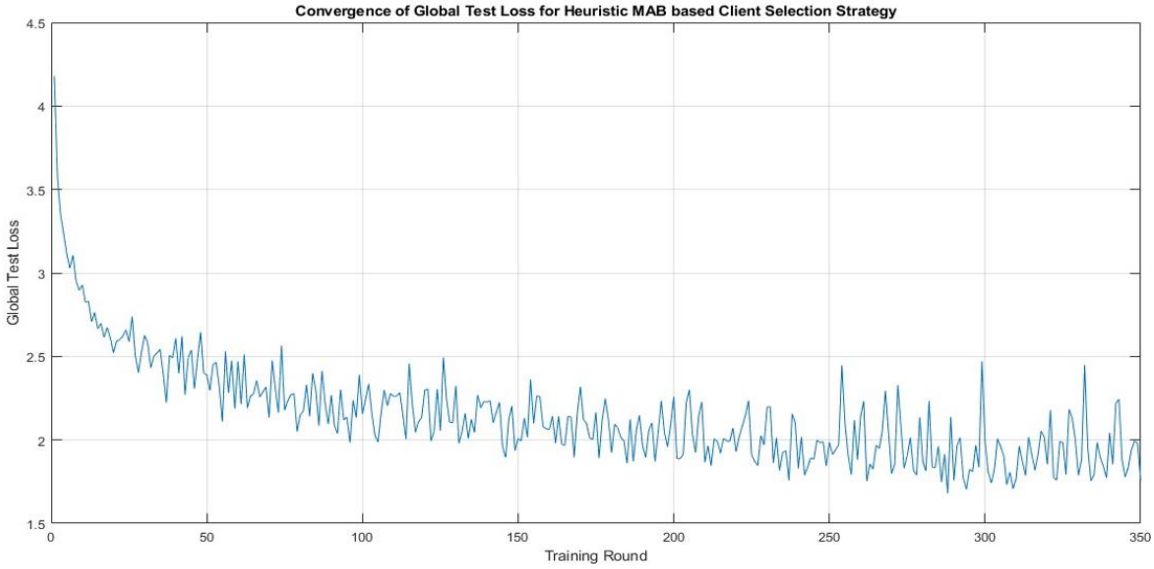

Figure 41: Convergence of global test loss with a Heuristic MAB based client selection strategy with a hyperparameter "ϒ" = 0.5

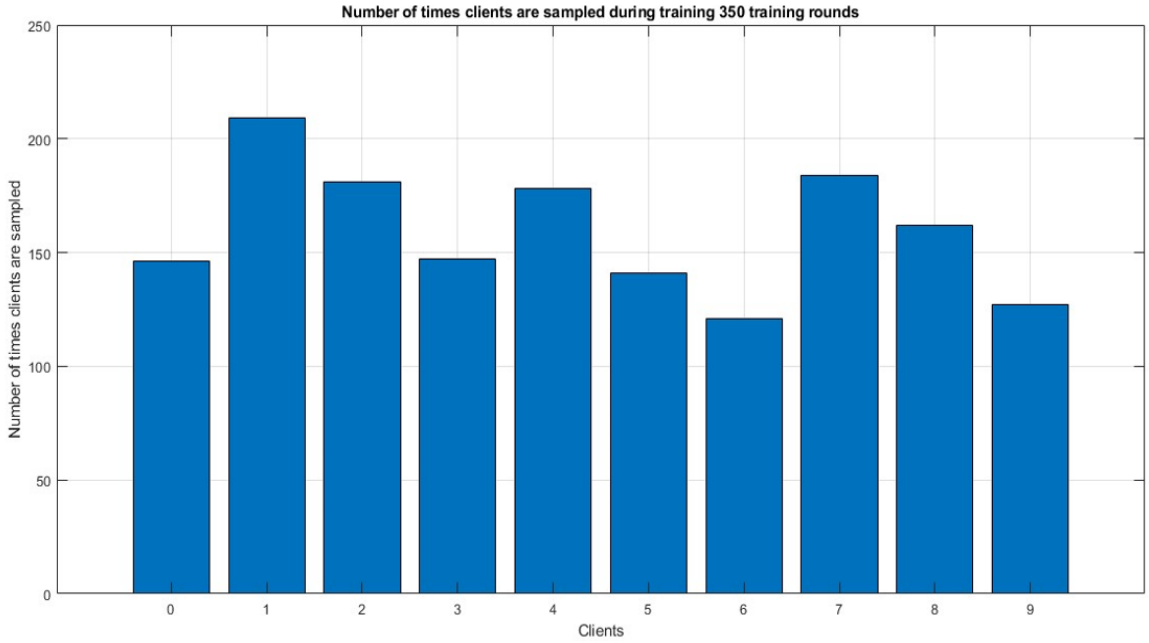

Figure 42: Number of times each client is sampled with a hyperparameter "ϒ" = 0.5

Figure 43 and Figure 44 presents the third experiment conducted with a hyperparameter value "ϒ" of 0.7. The experiment was run for 400 training rounds and the global model converges with a top-K (= 1, 2, 5) test accuracies of 0.62, 0.82 and 0.95 respectively and the global test loss converges at 1.8. Fairness factor calculated equals 0.926 for this case and client '1', '4' and '7' are sampled a greater number of times as shown in Figure 45.

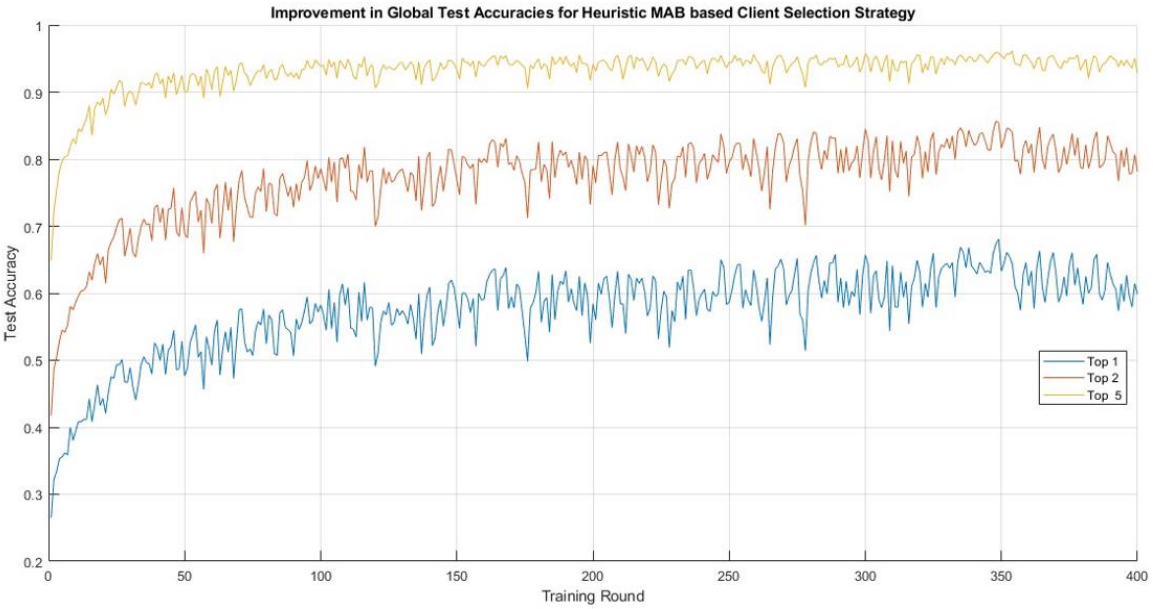

Figure 43: Improvement in global test accuracies for Heuristic MAB based client selection with a hyperparameter "ϒ" = 0.7

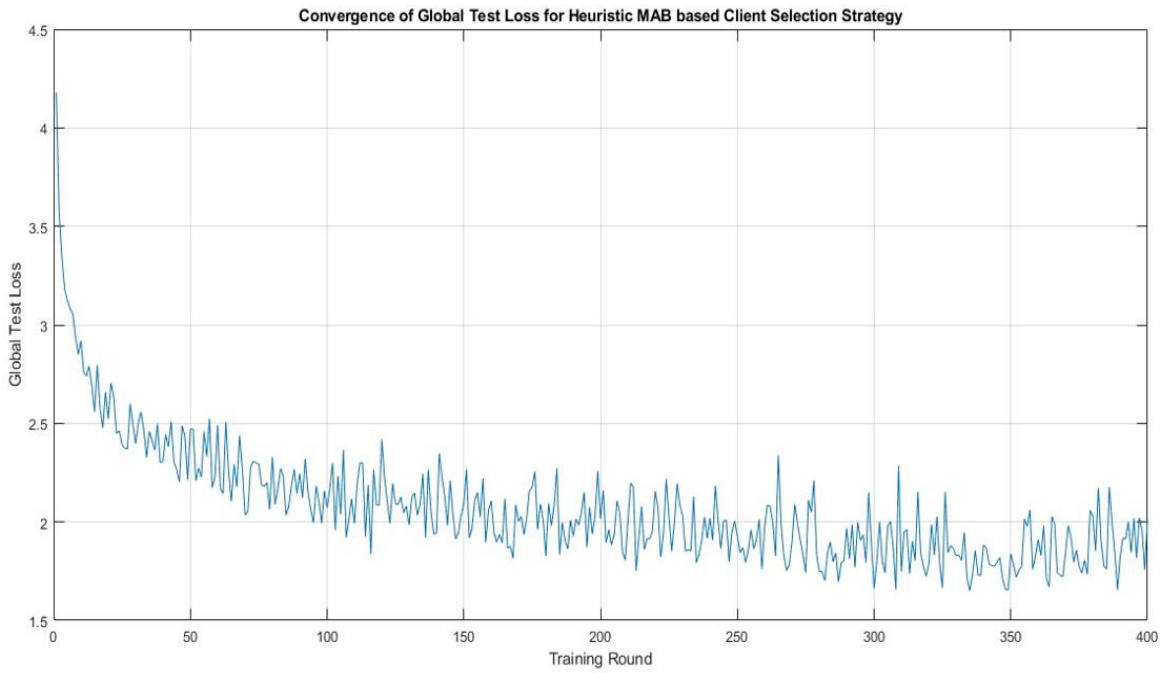

Figure 44: Convergence of global test loss with a Heuristic MAB based client selection strategy with a hyperparameter "ϒ" = 0.7

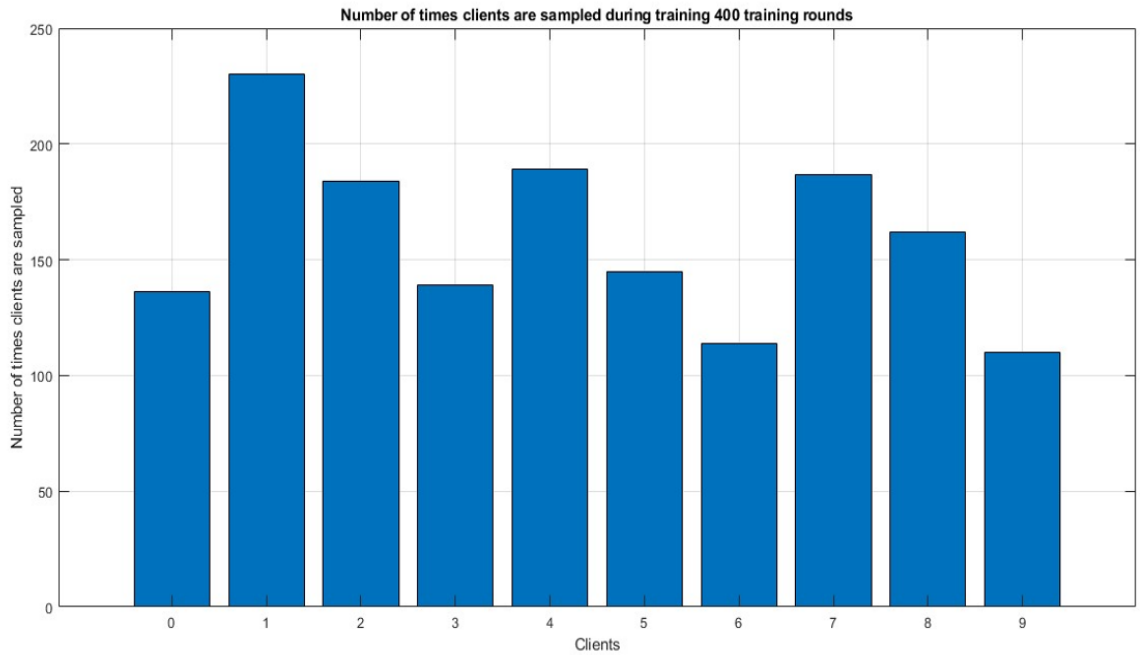

Figure 45: Number of times each client is sampled with a hyperparameter "ϒ" = 0.7

Based on the three experiments conducted above using the MAB client selection strategy using a client subset size (m) of four, there are few important conclusions that needs to be inferred based on three factors:

1. The hyperparameter "ϒ" and its influence
2. The rate of convergence
3. The most frequent sampled client.

These three factors are correlated and need to be evaluated before we can optimize the experimental setup of MAB based client selection strategy. In the context of UCB, the hyperparameter "ϒ", influences the current decision based on past rewards gained form clients which were sampled more often. The value of ϒ ranges from $0 < \text{ϒ} < 1$ and when its limit tends towards 1 all the past local loss values are given equal weight and when its limit tends to 0, the local loss from the current training round influences the client selection decision. Giving more importance to all the past local losses might not be ideal as the global model would have already sufficiently learned from the client selected in the past and we need to move on to learn from other clients. On the contrary, selecting different clients at each training round diminishes the ability of the global model to learn from any client. Thus, choosing the value of hyperparameter in the range of $0.3 < \text{ϒ} < 0.7$ is well suited.

The Infocom FLASH dataset has 10 clients in total and clients '1', client '6', client '9' and client '8' have the highest number of NLOS samples in the decreasing order and often return a higher value of local loss favoring its participation in training. This is also evident from Figure 42 and Figure 45 (ϒ = 0.5 and ϒ = 0.7 respectively showing the frequency of client participation) where client '1' is sampled the greatest number of times. On the contrary, client '7', client '0' and client '4' have the least number of NLOS samples in the increasing order and often return the least value of local loss. Their participation in training is often ignored resulting in outliers being incorporated. But, Figure 39 and Figure 45 (ϒ = 0.3 and ϒ = 0.7 respectively showing the frequency of client participation) show that these clients are sampled more often to participate in training. This is an important contribution using MAB client selection strategy improving the overall generalizability of the global model.

Both the values of ϒ (= 0.3 and 0.7) work well in this setting while for the latter case the number of training rounds substantially increases taking more training time. In this case a value of 0.3 for ϒ is better but in general the value of hyperparameter (ϒ) works well in the range of $0.3 < \text{ϒ} < 0.7$. In the next sections, we design two case studies to facilitate in-filed training post deployment using MAB based client selection strategy. The value of ϒ for these two experiments are chosen to be 0.3 which is best performing form the above analyses.

5.2.5 Case Study-1 for In-field Training post deployment

From the perspective of MEC, one crucial strategy for enhancing the adaptability of global models to dynamic changes in the environment around the vicinity of the MEC is the implementation of in-field training post-deployment. This method is particularly significant for dynamically adjusting the model in response to changes in the physical surroundings. To illustrate this approach, the Infocom FLASH dataset, which includes ten clients labeled from '0' to '9', serves as a valuable case study.

In this experiment, during the initial 200 training rounds before the model's deployment, clients '4' and '7' are deliberately excluded from the training process. This exclusion strategy is designed to mimic a scenario where certain data sources or clients are not available or not considered during the early stages of model training. However, after the model is deployed and has completed the first 200 training rounds, the strategy shifts. At this point, the contributions from clients '4' and '7' are incorporated into the training process, effectively simulating the introduction of new data sources or clients into the model's learning environment post-deployment. This inclusion is crucial for enhancing the model's robustness by enabling it to adapt to environmental changes and incorporate new patterns or information that was not initially available.

The process of integrating these new clients does not entail restarting the training process from scratch. Instead, it continues from the existing state of the global model, leveraging the knowledge and weights that have already been developed. This approach ensures efficiency and continuity in model training. The effectiveness of this strategy is evidenced by significant improvements in model accuracy, as depicted in the provided Figures. For instance, the top-1 accuracy of the model increases from 0.54 to 0.60 after the in-field training that includes clients '4' and '7' as shown in Figure 46. Such improvement highlights the value of dynamically incorporating new data sources into the training process post-deployment.

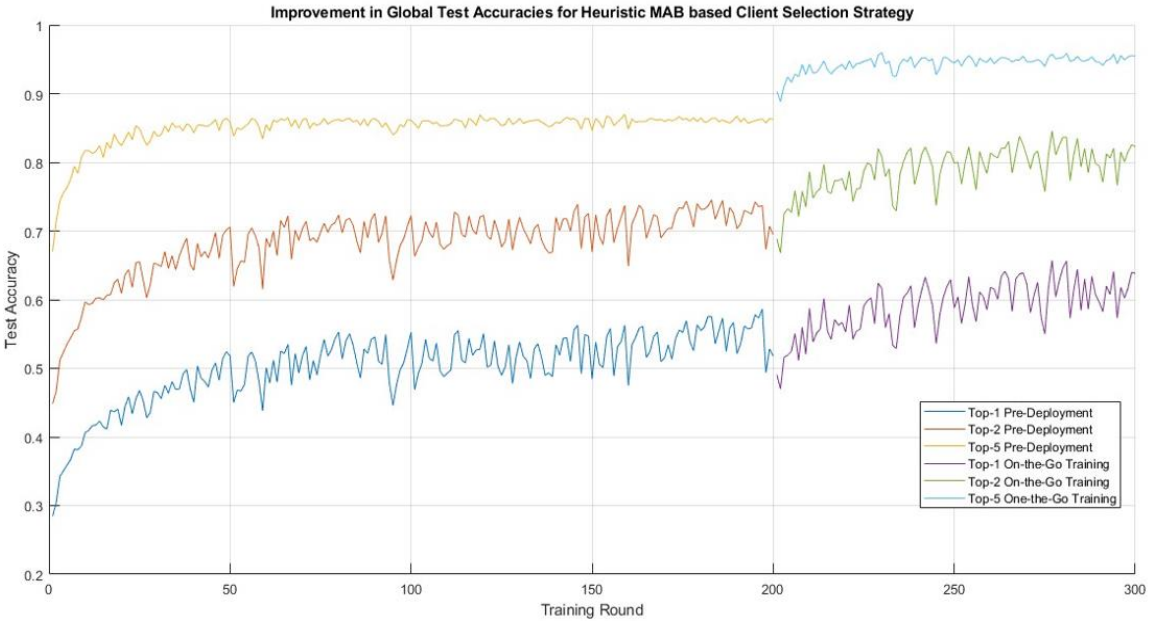

Figure 46: Improvement in global test accuracies for Case Study-1 to see the feasibility of In-field training post deployment.

Further analysis reveals a notable enhancement in the convergence of global test loss, indicating an overall improvement in model performance from a test loss of 2.8 to 1.78 post in-field training as shown in Figure 47. This achievement can be attributed to the employment of the MAB algorithm. The MAB algorithm plays a pivotal role in

efficiently integrating the contributions of new clients by leveraging historical data on client participation in the training process. It allows for a balanced exploration of new clients without compromising the model's generalizability or necessitating a complete retraining from the beginning. Remarkably, the incorporation of clients '4' and '7' was achieved with only 100 additional training rounds, half the number of the initial training rounds.

The strategic sampling of client post-deployment, as shown in Figure 48, demonstrates a judicious approach to exploring the contributions of newly added clients while minimizing potential biases. Clients '4' and '7' are sampled more frequently, contributing significantly to the model's enhanced robustness and error reduction. At the same time, the continued sampling of other clients ensures that the model remains generalizable and resilient to overfitting to the new clients' data. This balanced approach underscores the effectiveness of the MAB algorithm and the in-field training strategy in fostering a robust and adaptable global model in the ever-changing landscape of MEC.

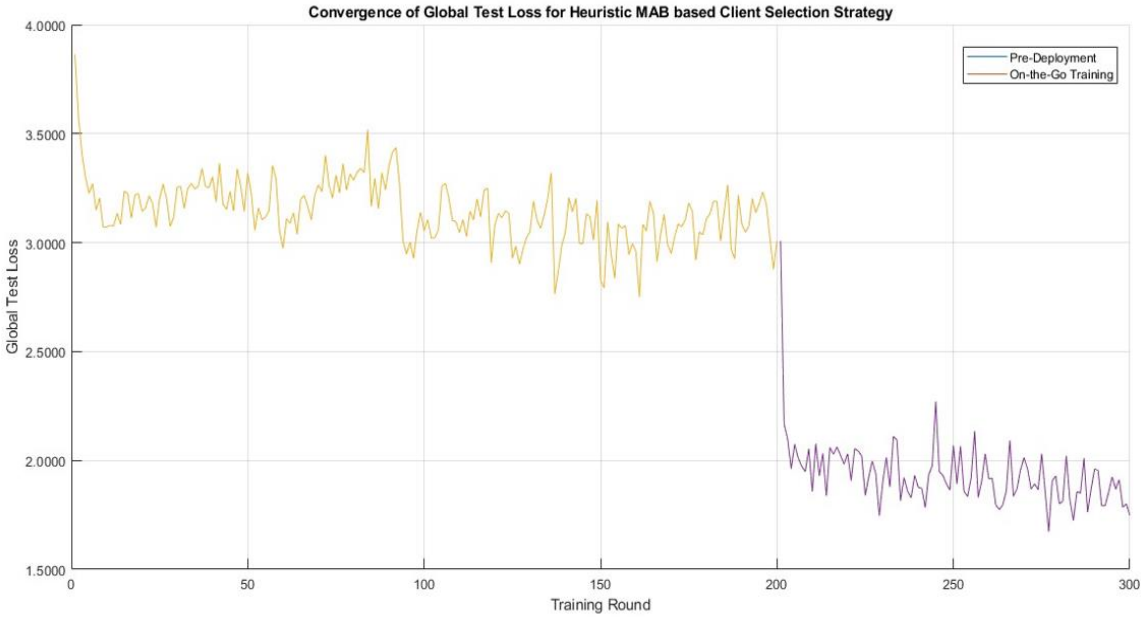

Figure 47: Convergence of global test loss for Case study-1 to see the feasibility of In-field training

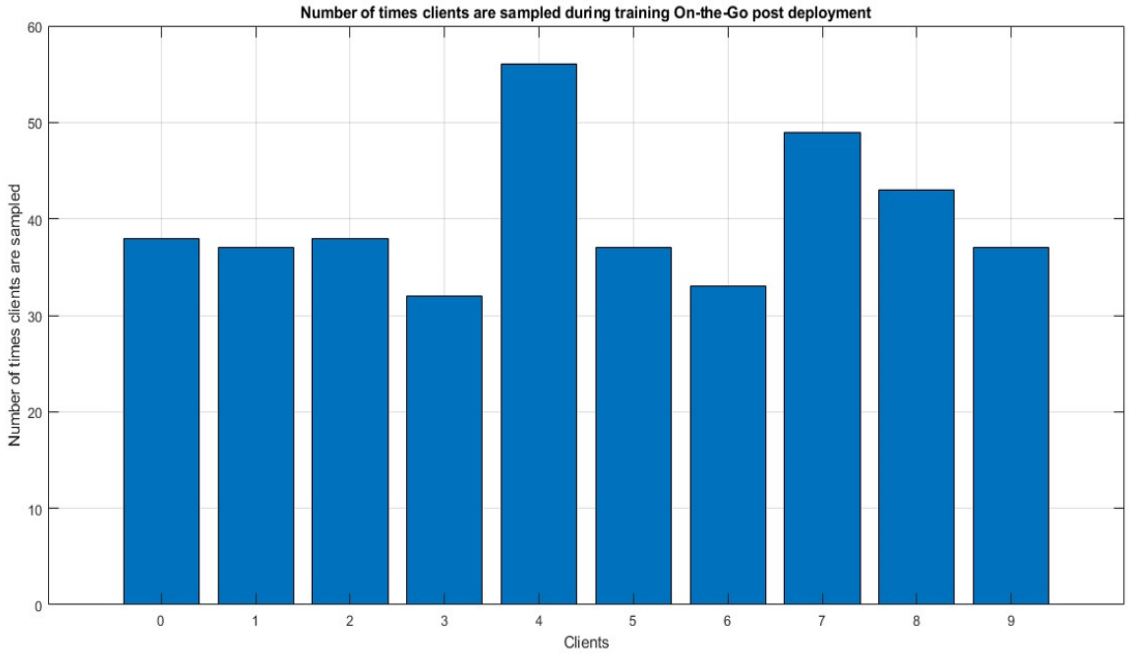

Figure 48: Number of time clients are sampled for Case study-1 during In-field training post deployment

5.2.6 Case Study-2 for In-field Training post deployment

We conduct another simulation experiment, to explore the dynamic integration of previously excluded clients into the training process of a global model, post its deployment with a larger set of clients deliberately omitted for initial round of training. Clients labeled '0', '2', '3', and '5' are deliberately kept out of the training loop for the first 150 rounds. This decision creates a scenario where the model learns to generalize from a subset of the available data, simulating situations where access to all data sources might not be available or selected for the initial training phases. However, once the global model is deployed and has undergone 150 training rounds, a strategic shift is implemented. At this juncture, the data and patterns from clients '0', '2', '3', and '5' are introduced to the already deployed model, thereby enhancing its learning with new insights and information. This transition does not entail starting the training process from scratch, instead, the model continues to learn and adapt based on its pre-deployment training, leveraging the accumulated knowledge and weight adjustments. This approach ensures efficiency in training and the seamless integration of new data. The impact of this methodological shift is evident in the performance metrics. As illustrated in the plots, there is a notable improvement in the model's accuracy across various top-K measures post the inclusion of the new clients. Specifically, the top-1 accuracy sees an uplift from 0.47 to 0.58 following the in-field training post-deployment as shown in Figure 49.

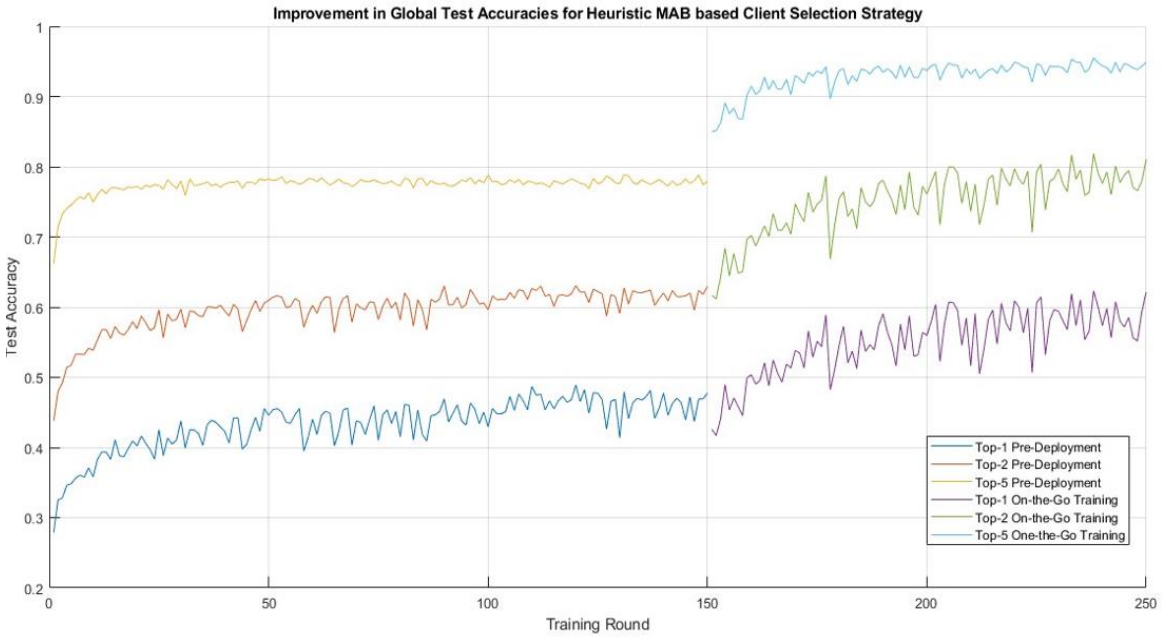

Figure 49: Improvement in global test accuracies for Case Study-2 to see the feasibility of In-field training post deployment.

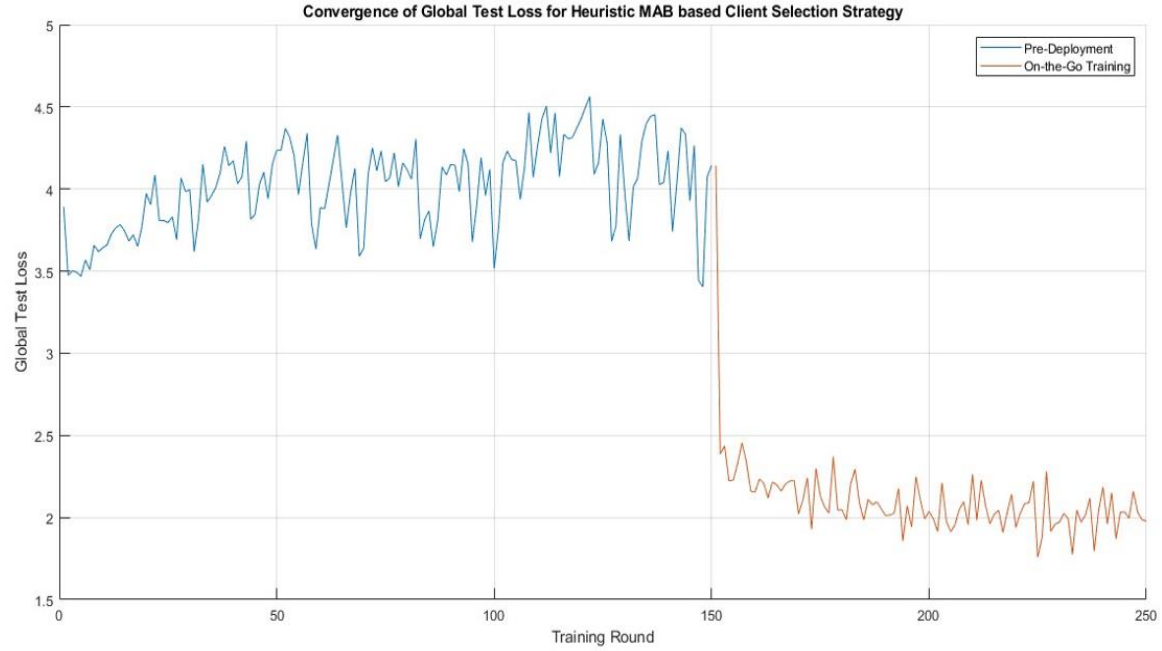

Figure 50: Convergence of global test loss for Case study-2 to see the feasibility of In-field training

Further examination reveals significant progress in the reduction of global test loss, highlighting an overall improvement in model performance. The test loss diminishes from 3.7 to 1.95 after the in-field training post-deployment as shown in Figure 50. Prior

to deployment, the global model is trained for only 150 training rounds as we can see the global test loss starts to increase denoting overfitting of the model already. This can be attributed to the fact that, for this experiment the test set already has a larger set of clients which are omitted from training initially. This improvement not only indicates enhanced prediction accuracy but also reflects the model's growing adaptability to diverse data patterns and environments. An interesting facet of this experiment is the strategic sampling of clients both before and after the model's deployment, as shown in Figure 51. Prior to deployment, clients '1', '4', '6', '7', '8', and '9' exclusively participate in the training rounds. Post-deployment, during the in-field training phase of 100 epochs, the newly incorporated clients '0', '2', '3', and '5' are sampled with higher frequency, aiming to integrate their unique data and environmental insights into the global model. The model continues to frequently sample other clients, to ensure that the integration of new clients does not skew the model's learning. This balanced sampling strategy is pivotal in enhancing the robustness and generalizability of the global model.

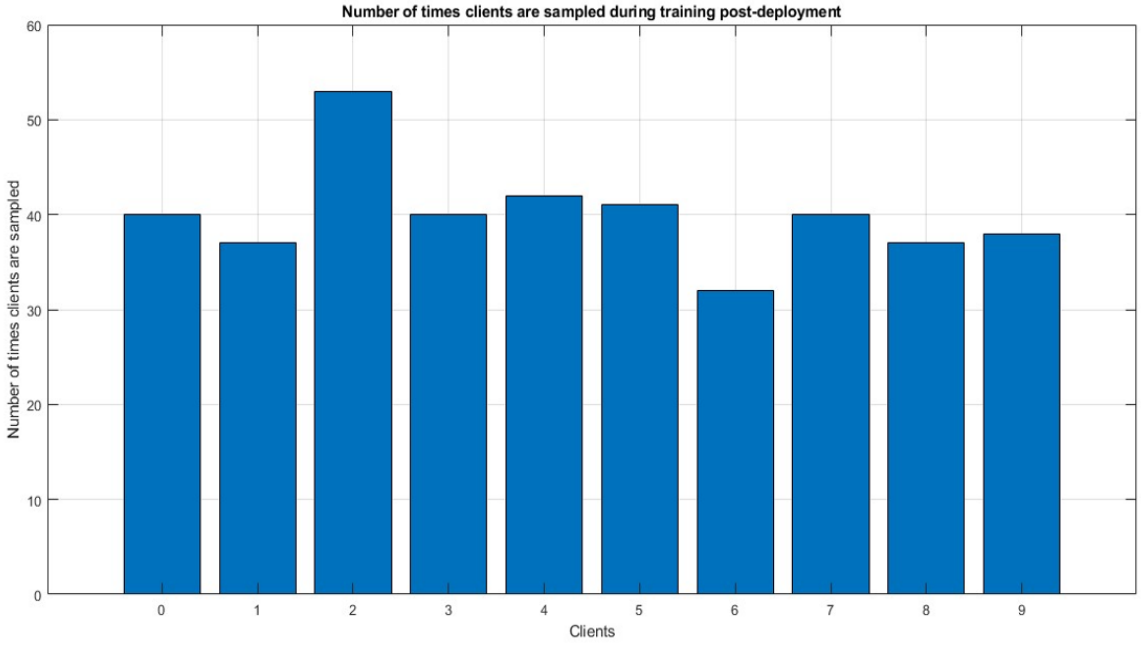

Figure 51: Number of time clients are sampled for Case study-2 during In-field training post deployment

5.2.7 Analyzes of experiments conducted to study the feasibility of In-field training post deployment of the global model

To encapsulate the findings from the two experiments aimed at evaluating the efficacy of in-field training post-deployment, it's clear that the adoption of a heuristic Multi-Armed Bandit (MAB) approach for client selection plays a pivotal role. This strategy effectively integrates new clients into the ongoing learning process without compromising the generalizability cultivated from previous training phases and the contributions from longstanding clients. Moreover, this approach facilitates the update of the global model

without necessitating a complete retraining. This efficiency translates into substantial savings in time, resources, and costs.

Table 13: Improvement to Global Model during In-field training post-deployment

Experiment (Client's post deployment)	**Top-1 Accuracy pre-deployment**	**Top-1 Accuracy post-deployment**	**Global Test Loss pre-deployment**	**Top-1 Accuracy post-deployment**
Case Study-1, Two clients added post-deployment	0.54	0.60	2.8	1.78
Case Sudy-2, Four clients added post-deployment	0.47	0.58	3.7	1.95

Compared to strategies that either do not discriminate between clients or solely prioritize clients based on MaxLoss, the MAB based selection method significantly accelerates the update process of the global model, reducing the time required by more than half. This efficiency is documented in Table 13, which summarizes the outcomes of both experiments conducted in Case Study-1 and Case Study-2. Notably, in both cases, the post-deployment training spanned 100 rounds, incorporating 2 and 4 new clients, respectively. This structured approach not only underlines the feasibility of in-field training post-deployment but also showcases the MAB strategy's capability to seamlessly blend new client data into the model's learning trajectory, thereby preserving its robustness.

5.3 Statistical Analyzes of Test Accuracies for different client selection strategy

In Section 5.2, we analyzed results from a distributed architecture using three different client selection strategies: unbiased client selection, MaxLoss-based biased client selection, and Heuristic MAB-based biased client selection. These strategies were proposed to enhance the convergence rate of the global model, reduce the load on sub-6GHz channels, and improve scalability, robustness, and generalizability. All simulations discussed in the previous section were conducted using a randomized sampling of the training dataset to eliminate training biases. Consequently, each training iteration of the DL model maintains parameter values within a standard deviation range centered around a mean value. The top-K accuracy predictions (for K=1, 2, and 5) from each simulation also exhibited variability, defined by specific mean and standard deviation values. This necessitates a statistical analysis comparing these measures to evaluate the DL model's performance effectively.

For each strategy, 10 individual simulation experiments were carried out and the test accuracies for top-K predictions were recorded. Based on the insights from the previous section, the optimal simulation settings for each strategy were identified. For the MaxLoss-based strategy, a client subset size of four was selected for achieving high accuracies while minimizing sub-6GHz channel overheads. Similarly, for the Heuristic

MAB-based strategy, a client subset size of four and a hyperparameter 'ϒ' value of 0.3 were chosen to improve model robustness and generalizability. The mean and standard deviation across the set of 10 experiments for each biased selection strategy were calculated and are presented in equations 5.2 and 5.3, respectively.

$$\mu = \frac{1}{N}\sum_{i=1}^{N} x \qquad (5.2)$$

$$\sigma = \sqrt{\frac{\sum(x-\mu)^2}{N}} \qquad (5.3)$$

Where, μ is the mean, σ is the standard deviation, N is the total number of simulations conducted i.e. 10 and 'x' is the value of accuracy at ith simulation.

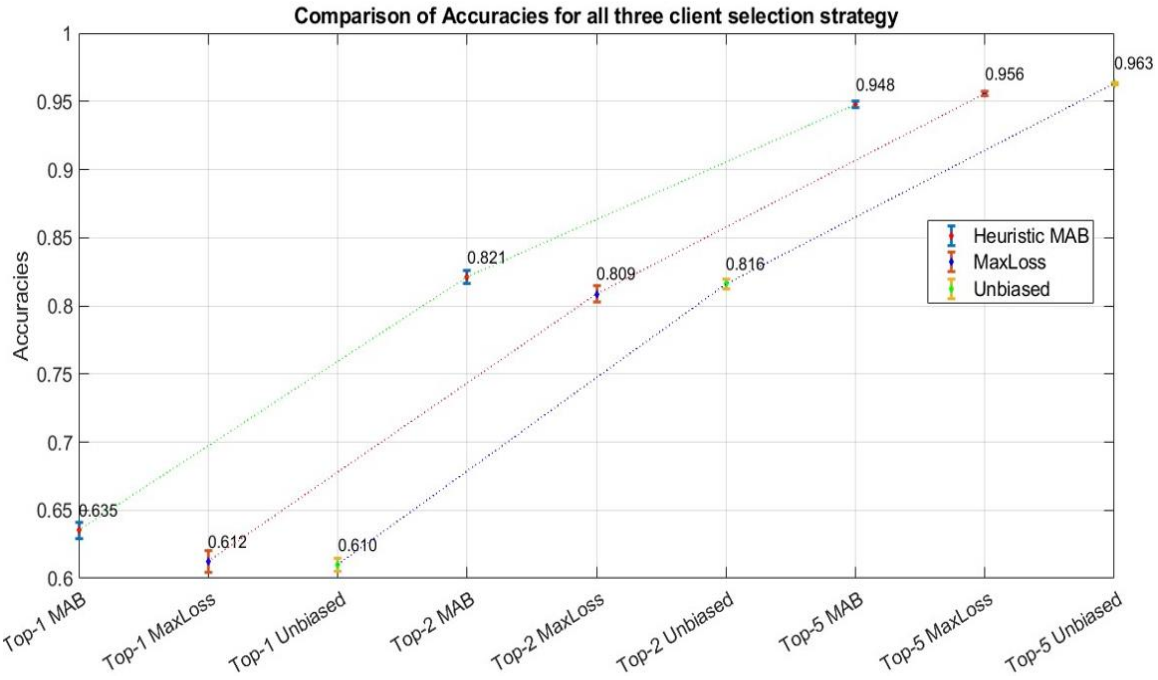

Figure 52: Statistical Comparison of all three-client selection strategy

Figure 52 presents the mean test accuracies for top-K values (K=1, 2, 5) alongside error bars representing one standard deviation about the calculated mean value for each of the three client selection strategies. For K=2 and K=3, the Heuristic MAB strategy outperforms the other two strategies in terms of mean values achieved. For K=5, the unbiased client selection strategy achieves a higher mean, followed by the MaxLoss selection strategy. This indicates that the Heuristic MAB client selection strategy achieves higher accuracies for smaller values of K but marginally under performs for bigger values of K. Figures 53 through 55 provide enlarged views to better illustrate the standard deviation associated with each client selection strategy across different K values, aiding in a more detailed analysis of the variability in model performance.

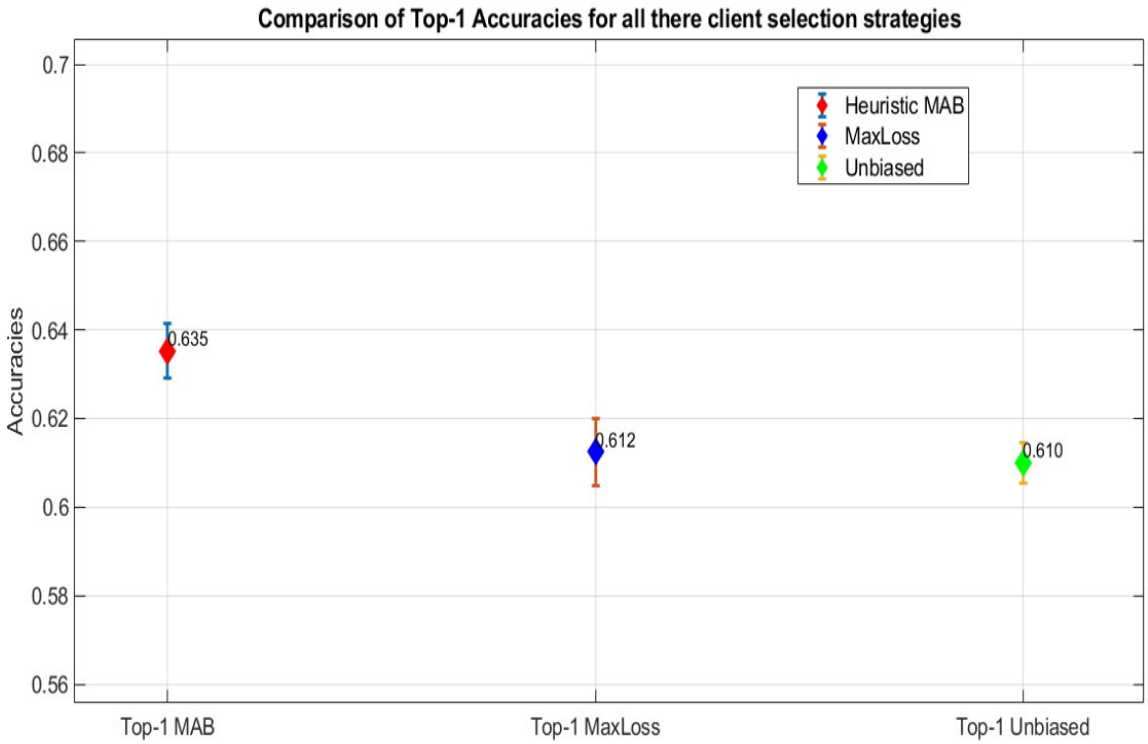

Figure 53: Statistical comparison of Top-1 accuracy for all three-client selection strategy

Figure 53 provides a detailed view of Figure 52, specifically highlighting the top-1 accuracies for the three client selection strategies. The accuracies achieved by the Heuristic MAB strategy are notably higher than those of the other two methods. This superior performance can be credited to the strategic sampling of clients at each training round, which is designed adopting the UCB algorithm to maximize rewards (i.e. accuracy in our case). The Unbiased strategy shows the smallest standard deviation, reflecting its consistent performance across training sessions due to the equal contribution from all clients.

Figure 54 provides a detailed view of Figure 52, specifically highlighting the top-2 accuracies for the three client selection strategies. While the Heuristic MAB method continues to have the highest mean accuracy among the three, its performance shows a notable variation, as indicated by an increased standard deviation. Statistically, this higher value of standard deviation results in a slight decrease in performance reliability for the Heuristic MAB method. Compared to this, the Unbiased strategy shows improved performance relative to the Heuristic MAB, particularly because the gap between their mean accuracies has narrowed. Additionally, similar to previous observations, the Unbiased strategy maintains the smallest standard deviation, underscoring its consistent performance across different training instances.

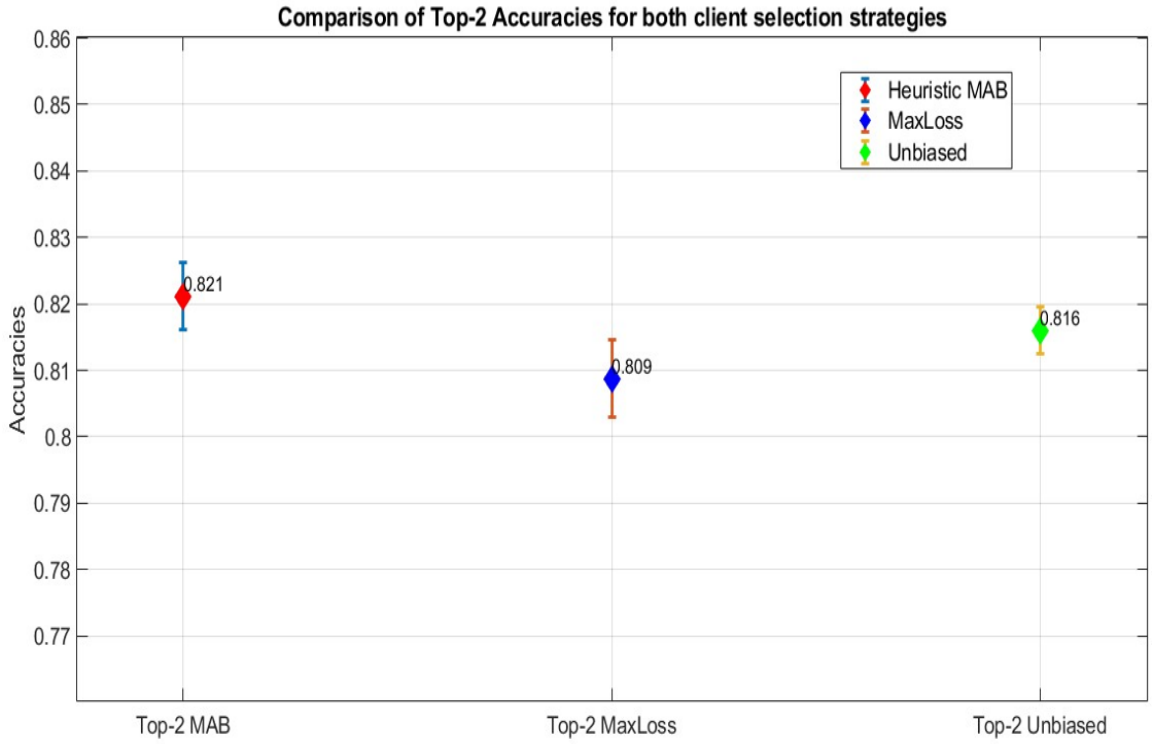

Figure 54: Statistical comparison of Top-2 accuracy for all three-client selection strategy

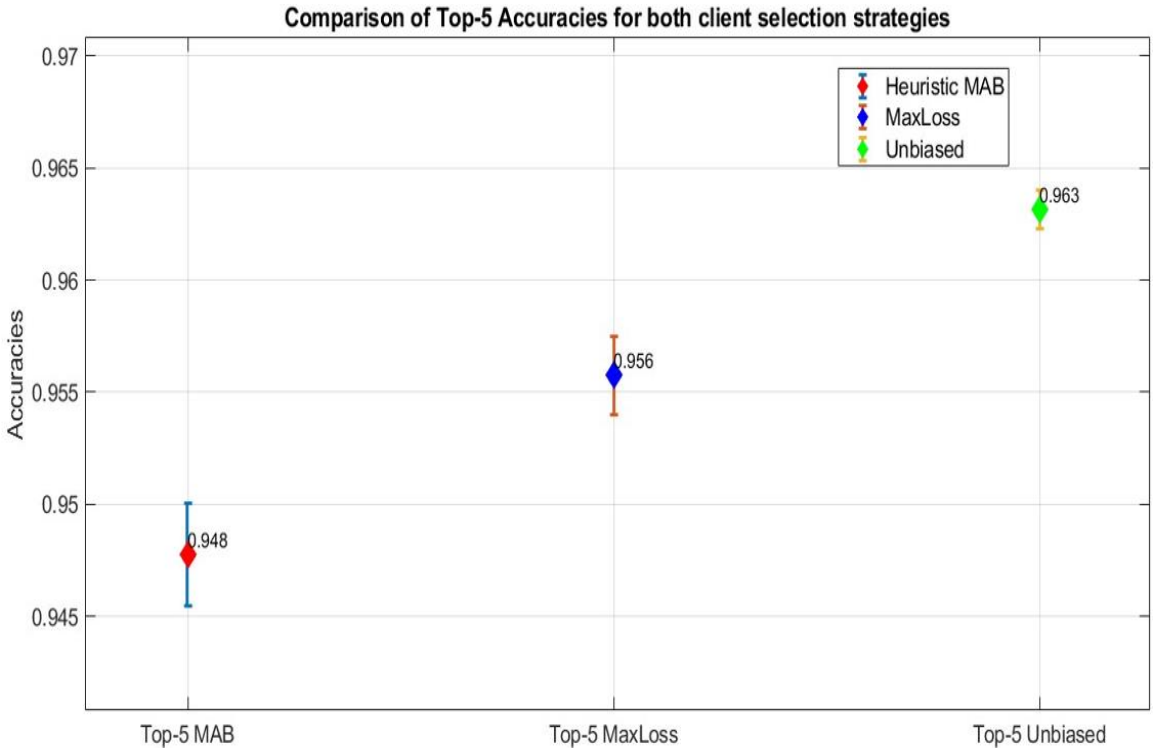

Figure 55: Statistical comparison of Top-5 accuracy for all three-client selection strategy.

Figure 55 provides a detailed examination of the top-5 accuracies from Figure 52, focusing on the three client selection strategies. In this scenario, the Unbiased strategy outshines the other two methods, thanks to its comprehensive inclusion of all clients in each training round and enhanced channel information when the subset of beamforming pairs is expanded. This method also continues to exhibit the lowest standard deviation, highlighting its consistent performance. Statistically, the Unbiased strategy is superior at this higher value of "K". However, it's also noteworthy that the MaxLoss strategy shows improved performance compared to its results at lower "K" values, likely benefiting from the larger subset sizes. On the other hand, the performance of the Heuristic MAB strategy dips slightly at higher "K" values. This decrease can be attributed to the underlying mathematical model of the UCB algorithm, which aims to maximize rewards with minimal costs; here, the cost being the increased size of the subset of beamforming pairs.

Although we have a good statistical inference for both Heuristic MAB and Unbiased selection strategies, the Heuristic MAB strategy remains the most effective at smaller "K" values, which is crucial for our purposes. Higher accuracies at these levels lead to reduced latencies and enhanced throughput ratios. While there is a minor performance drop at higher "K" values, this is offset by other advantages, such as faster convergence and reduced channel overhead, which the Heuristic MAB strategy offers. The Unbiased strategy, while statistically reliable due to its minimal standard deviation, does suffer from slower convergence rates, scalability issues, and increased overhead on sub-6GHz channels. The MaxLoss strategy, in comparison, consistently shows a larger standard deviation across all scenarios, reflecting its selective client participation approach that potentially excludes certain clients from training sessions.

5.4 Computational Resource Requirements

The DL model proposed in section 4.1 has a total of 696,600 parameters for GPS unimodal network, 104,450,848 parameters for camera-images unimodal network, 933,280 parameters for LiDAR unimodal network and 1,557,952 parameters for fusion parameters. The total number of parameters adds up to 107,638,680 parameters and considering each parameter to take up 32 bits the overall size of parameters results around 430.5 MB. The size of individual network parameters accounts to 2.78 MB for GPS, 417.8 MB for camera, 3.73MB for LiDAR and 6.23 MB for fusion network. The camera unimodal alone accounts for 97% of the total parameters and as discussed in section 5.1.4, removing camera unimodal can substantially reduce computational resources. Post deployment, the testing requirement will require at most 1GB RAM on a 2GHz 64-bit processor. The centralized architecture was trained and tested on a 2GHz 64-bit processor with a RAM requirement of 8-16GB. The distributed architecture was trained on the Portage High Performance Computing Platform on a 2.1GHz 64-bit processor with a RAM requirement of 80-102GB. The boilerplate information of the Portage HPC is available in Appendix C.

5.5 Analyzes of adopting Pruned GPS Unimodal

This section explores the pruned GPS unimodal network, which now produces only two output features, significantly reducing computational overhead by nearly 99%. As previously discussed in section 4.7, the original model was inefficient, generating 64 output features from just two input values, leading to unnecessary computational overheads. The reduction in complexity is further supported by findings from section 5.1.1, which showed no improvement in top-K accuracies to analyze the use of only GPS unimodal, thus underscoring the need for a simpler model.

The pruned GPS unimodal network was retrained and tested within the distributed architecture outlined in Section 4.3, utilizing the Heuristic MAB client selection strategy for GPS and LiDAR data fusion. The training involved a subset of four clients, with the hyperparameter 'ϒ' set to 0.3, adhering to the simulation parameters detailed in Table 10 from section 5.2.

Figure 56 shows the improvements in top-K accuracies (for K=1, 2, and 5) achieved through simulations using this pruned GPS unimodal model. The accuracies achieved in Figure 56 after 300 training rounds using the pruned GPS unimodal is on par with the accuracies achieved in section 5.2.4 using the same simulation settings before pruning the GPS unimodal, demonstrating the model's efficiency and effectiveness post-pruning.

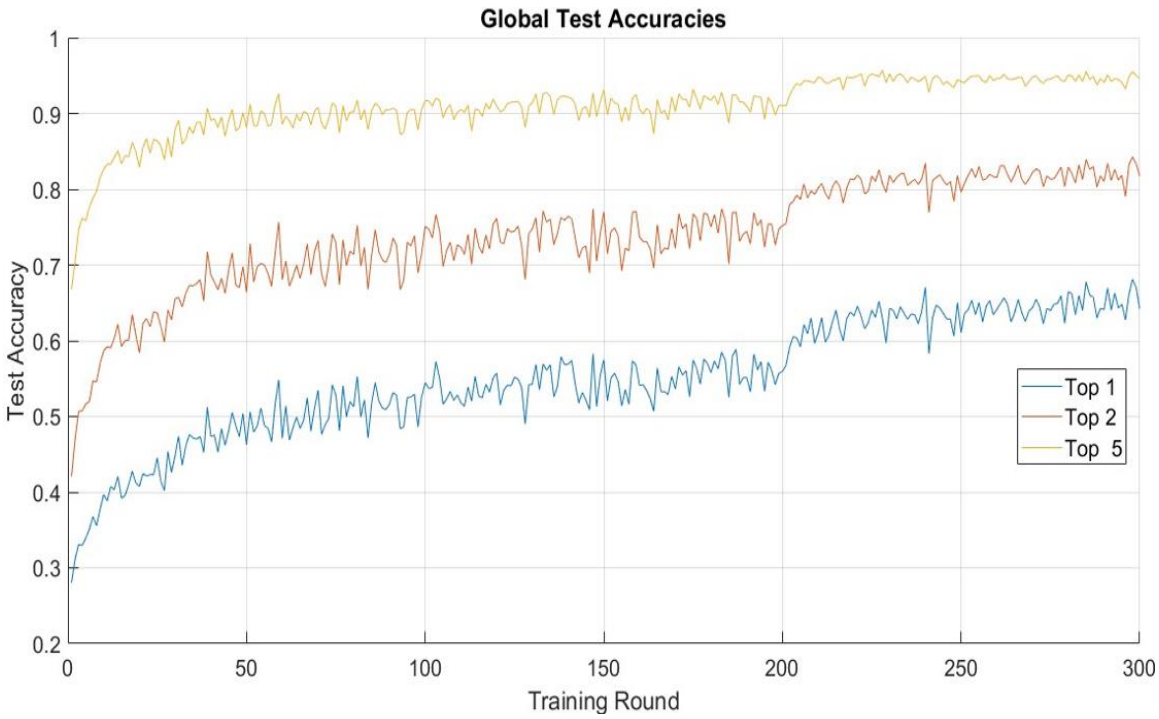

Figure 56: Improvements in test accuracies using the pruned GPS Unimodal

Table 14: Comparison of results after pruning GPS Unimodal

	Using Original GPS Unimodal	**Using Pruned GPS Unimodal**
Top-1 Accuracy	0.65	0.655
Top-2 Accuracy	0.82	0.815
Top-5 Accuracy	0.95	0.955
Global Test Loss	1.8	1.77

Table 14 presents a comparison between the performance metrics of the pruned GPS Unimodal and the original GPS Unimodal networks. Both experiments employed the Heuristic MAB biased client selection strategy and utilized GPS-LiDAR data fusion, with all other experimental settings remaining consistent across the two simulations. Analysis presented in Table 14 shows the performance of the pruned GPS Unimodal is marginally is on par with the original model. This is a significant achievement, as the computational demands of the GPS Unimodal have been precisely reduced by 99.44% without compromising performance, thereby maintaining equivalent accuracy levels.

5.6 Comparison of all the methods discussed with state of art methodologies

In this section, experiments results are compared with state of art methodologies to predict top "K" beamforming pairs using out of band multimodalities. Distributed architecture brings the overall communication overhead down to a maximum of 256MB when compared to the centralized architecture of 2.5GB of data. From Table 15, it can also be seen that there is significant improvement in the accuracies when using distributed architecture. This is due to the fact that the global model in the distributed architecture is able generalize well for unforeseen scenarios, which have been trained for by other clients. Although the Raymobtime (S008 and S009) is a larger dataset, the Infocom FLASH dataset encompasses a significant number of NLOS cases within an urban canyon setting. Moreover, the Infocom FLASH datasets are based on real-world, non-simulated observations collected in live environments, and serve as a good proof of concept for practical implementation. Utilizing multimodal side information from sensors to trace millimeter-wave (mmWave) rays, and predicting Top-K beamforming pairs, demonstrates a promising approach to minimize latency in establishing connectivity between vehicles and Mobile Edge Computing (MEC).

Table 15: Comparison of results with current state of art methodologies

Methods	**Datasets**	**Modalities**	**Inference**	**Top 1**	**Top 2**	**Top 5**
Dias et.al[12]	Raymobtime (S007)	LiDAR	Centralized	20.5 $\pm$ 1%	25.5 $\pm$ 1%	54.5 $\pm$ 1%
Klautau et.al[11]	Raymobtime (S008)	LiDAR	Centralized	30.5 $\pm$ 1%	43.5 $\pm$ 1%	57.5 $\pm$ 1%
Matteo Zecchin et.al [10]	Raymobtime (S009)	GPS, LiDAR	Centralized	50.6%	-	83.85%

Methods	**Datasets**	**Modalities**	**Inference**	**Top 1**	**Top 2**	**Top 5**
Reproduced Results of [9] using 3-sensor fusion	Raymobtime (S009)	GPS, LiDAR, Camera	Centralized	51 ± 0.5%	70 ± 0.5%	83.5 ± 0.5%
Proposal to use only GPS and LiDAR unimodal.	Raymobtime (S009)	GPS, LiDAR	Centralized	54.5 ± 0.5%	72.5 ± 0.5%	84.5 ± 0.5%
Max Loss based biased client selection strategy (client size = 4)	FLASH	GPS, LiDAR	Distributed	61 ± 1%	80 ± 1%	96%
Heuristic multi-arm-based client selection strategy (client size = 4)	FLASH	GPS, LiDAR	Distributed	62 ± 3%	80.5 ± 1.5%	94.5 ± 0.5%
Federated averaging based unbiased client selection strategy	FLASH	GPS, LiDAR	Distributed	61%	81%	96%

6 Conclusion

In this work we proposed the use of Multimodality-Fusion based Deep Learning (DL) networks on out-of-band side information from sensors to aid faster selection of beam forming pairs, which can be incorporated to V2I communication paradigm such as IEEE 802.11ad and 5G-NR. Firstly, the classical configurations of antenna codebook elements to form massive directional beams was studied by representing all the configurations of beamforming pairs using normalized coupling power to return the generalized channel information (β). Based on this, the channel information restricting to a subset of top-K beamforming pairs (β_K) was represented using an empirical probability density distribution "P". The DL model uses a parameterized function to predict the set of probability densities corresponding to every beamforming pair using which the top-K beamforming pairs can be found by assessing the elements with highest values in this set. The idea is to find the optimum beamforming pair in this set of top-K beamforming pairs with a very high probability and restricting the standard beam sweeping procedure on this set to significantly reduce latencies to establish a connection.

The DL model leverages the use of multimodality sensors such as GPS, LiDAR and camera as inputs to find the probability densities of each beamforming pair. The DL architecture uses a multi-step fusion-based network where individual sensor inputs are connected to their respective unimodal networks in a parallel configuration and finally the outputs of each of the unimodal are concatenated to feed to the fusion network in series. This particular architecture is efficient as it allows the individual unimodal networks to extract high-fidelity features representing various environmental embeddings and use these high-level features to predict the top-K beamforming pairs using the fusion network. The trained parameters of each unimodal fits a parameterized function to relate the non-linear sensor data to feature vectors. These feature vectors are concatenated to finally predict the probability densities. The raw information from sensor is simplified using pre-processing techniques to reduce complexity and overheads handling the deep learning framework.

The DL model developed in this work is used in two system-level architecture – centralized and distributed. Both these architectures have their own advantages and disadvantages. Using the centralized architecture, the DL model is completely trained and updated at the MEC. The data required for training and validating the DL model is acquired on-site by a vehicle, pre-processed and shared with the MEC over the sub-6GHz channels. The MEC trains the DL model and once deployed can predict the top-K beamforming pairs. The distributed architecture uses a federated learning paradigm where vehicles train a local model on-site and share only the model weights to the MEC to update the global model by aggregating individual local models. The centralized architecture consumes a lot of space in the sub-6GHz channels saturating them and moreover updating the DL model to stochastic variations in the environment around the vicinity of the MEC is difficult. On the other hand, distributed architecture brings down the usage of sub-6GHz channels as the model is trained locally and also improves robustness as multiple vehicles are training a single global model. In our study, we were able to bring down the usage of sub-6GHz to share 2.5GB of sensor data to just 256MB

of information on the sub-6GHz channels using a distributed architecture. The downside of distributed architecture is that it requires more computational power on the training vehicles. Moreover, the distributed architecture adds two additional steps to orchestrate the training vehicle to participate in training and aggregate the local weights to update the global model after each training round.

The performance of individual sensor modalities and fusion network was studied using a centralized architecture. The dataset from Raymobtime (S008 and S009), which contains episodes of timestamp-synched values of GPS, LiDAR and Camera as inputs and ground truth ray tracing pairs, was used to train, validate and to test the network. The primary performance metrics, namely the top-K accuracy, and throughput ratio, were used to evaluate and benchmark the multimodality-fusion network in a centralized architecture. The performance metrics indicated a significant improvement just by using GPS and LiDAR sensors which is on-par with the performance metrics using all three sensors. This is an important conclusion as removing camera images reduces computational resource requirements and preprocessing latencies significantly. The top-K (= 1, 2, 5) test accuracies attained by using GPS and LiDAR sensors attained 54.5%, 72.5% and 84.8% respectively. We were able to achieve a throughput ratio of 97% for a value of K =10 with a 97% reduction in latency compared to the standard beams sweeping procedure. The test accuracies are validated using the Raymobtime S009 dataset which has significantly higher number of NLOS scenarios at 85%, translating to real-world scenarios which would contain larger NLOS scenarios. This proves the ability of the DL model leveraging the use of multimodality sensors to predict top-K beamforming pairs.

The shortcomings of the centralized architecture encouraged the development of a distributed architecture using federated learning paradigm. The centralized architecture saturates the sub-6GHz channels with high volume of initial data transfer between vehicles to server. The architecture could be quite challenging to upscale and generalize to stochastic changes in the physical environment around the vicinity of the MEC which are termed as unseen environmental scenarios. The distributed architecture is able to address these issues, it reduces the use of sub-6GHz by not requiring the vehicle to transfer large volumes of sensor data to the MEC. Additionally, it improves the scalability, robustness and adaptability of the model to unseen environmental scenarios as it is capable of in-field training post deployment to update the global model.

The novel distributed architecture uses the federated learning paradigm to incorporate distributed learning using both unbiased and biased client selection strategy. Two client biased selection strategy – MaxLoss and Heuristic MAB are proposed to select a small subset of clients to participate in each training round. Our suggestion to select a small subset of clients capable of canvasing for other unselected clients helps in reducing the use of sub-6GHz communication channels further. The use of biased client selection strategy can reduce the use of sub-6GHz channels by 50-70% by selecting a fraction of one-half to one-third clients to participate in each round of training respectively. Moreover, there is 10% -15% additional increase in the top-1, top-2 and top-5 accuracies for predicting the best beamforming pairs/sectors using the distributed architecture. These

improvements are based on the latest available dataset to use federated learning paradigm for predicting the best mmWave beamforming pairs/sectors available in the literature.

Finally, we also propose to prune the GPS unimodal and reduce its complexity without compensating the performances and accuracies achieved by the overall system. The computational overheads of the GPS unimodal was reduced by 99% which is a significant contribution. Initially, the GPS unimodal was extracting 64 features from just two coordinate values without achieving any improvements in accuracies and analyzing the network architecture of the GPS unimodal proved to be redundant without adding any additional benefits. This encouraged to prune the GPS unimodal network layers and reduce the output feature vector size to two. The pruned GPS unimodal presents on-par results when trained and tested in a distributed architecture setup using Heuristic MAB client selection strategy from before.

For future work the use of camera images for the distributed architecture can be further studied, which proves to be robust to changing weather patterns as LiDAR sensors are unable to work at maximum throughput during rain and snowfall [29]. Moreover, in adverse condition of missing information due to failure of sensor, the incorporation of camera images can supplement LiDAR information and vice-versa as a fallback mechanism without compromising the system performance. In this work, the F-DL network proposed can be further pruned to reduce the computational and memory resources bringing down the cost of deployment. An additional future study in the scalability to adapt to different sizes of antenna codebooks at MEC is relevant.

7 Reference List

[1] M. Malinverno, et.al, Edge-based Collision Avoidance for Vehicles and Vulnerable Users,

[2] M. Bachmann, M. Morold, K. David, "Improving smartphone-based collision avoidance by using pedestrian context information," IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops), 2017.

[3] J. Choi, V. Va, N. González-Prelcic, R. Daniels, C. R. Bhat, and R. W. Heath, "Millimeter-wave vehicular communication to support massive automotive sensing," IEEE Commun. Mag., vol. 54, no. 12, pp. 160–167,Dec. 2016.

[4] I. Rasheed, F. Hu, Y. Hong, and B. Balasubramanian, "Intelligent vehicle network routing with adaptive 3D beam alignment for mmWave 5G-Based V2X communications," IEEE Trans. Intell. Transp. Syst., vol. 22, no. 5, pp. 2706–2718, May 2021.

[5] W. Roh et al., "Millimeter-wave beamforming as an enabling technology for 5G cellular communications: Theoretical feasibility and prototype results," IEEE Commun. Mag., vol. 52, no. 2, pp. 106–113, Feb. 2014.

[6] J. Wang et al., "Beam codebook-based beamforming protocol for multi-gbps millimeter-wave WPAN systems," IEEE J. Sel. Areas Commun., vol. 27, no. 8, pp. 1390–1399, Oct. 2009.

[7] Y. Yaman and P. Spasojevic, "Reducing the LOS ray beamforming setup time for IEEE 802.11 ad and IEEE 802.15. 3c," in Proc. IEEE Mil.Commun. Conf., 2016, pp. 448–453.

[8] M. Giordani, M. Polese, A. Roy, D. Castor, and M. Zorzi, "A tutorial on beam management for 3GPP NR at mmWave frequencies," IEEE Commun.Surv. Tut., vol. 21, no. 1, pp. 173–196, Jan.–Mar. 2019.

[9] Batool Salehi et.al, Deep Learning on Multimodal Sensor Data at the Wireless Edge for Vehicular Network, IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, VOL. 71, NO. 7, JULY 2022.

[10] Matteo Zecchin, Mahdi Boloursaz Mashhadi, Deniz Gündüz, Marios Kountouris, David Gesbert, LIDAR and Position-Aided mmWave Beam Selection With Non-Local CNNs and Curriculum Training, EEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, VOL. 71, NO. 3, MARCH 2022

[11] A. Klautau, N. González-Prelcic, and R. W. Heath, "LIDAR data for deep learning-based mmWave beam-selection," IEEE Wireless Commun. Lett., vol. 8, no. 3, pp. 909–912, Jun. 2019.

[12] M. Dias, A. Klautau, N. González-Prelcic, and R. W. Heath, "Position and LIDAR-aided mmWave beam selection using deep learning," in Proc. IEEE 20th Int. Workshop Signal Process. Adv. Wireless Commun., 2019, pp. 1–5.

[13] Z. Xiao, T. He, P. Xia, and X.-G. Xia, "Hierarchical codebook design for beamforming training in millimeter-wave communication," IEEE Trans.Wireless Commun., vol. 15, no. 5, pp. 3380–3392, Jan. 2016.

[14] S. Wang, J. Huang, and X. Zhang, "Demystifying millimeter-wave V2X: Towards robust and efficient directional connectivity under high mobility," in Proc. 26th Annu. Int. Conf. Mobile Comput. Netw., 2020, pp. 1–14.

[15] T. Nitsche, A. B. Flores, E. W. Knightly, and J. Widmer, “Steering with eyes closed: mm-Wave beam steering without in-band measurement,” in Proc. IEEE Conf. Comput. Commun., 2015, pp. 2416–2424.
[16] N. González-Prelcic, R. Méndez-Rial, and R. W. Heath, “Radar aided beam alignment in mmWave V2I communications supporting antenna diversity,”in Proc. Inf. Theory Appl. Workshop, 2016, pp. 1–7.
[17] H. He, C.-K. Wen, S. Jin, and G. Y. Li, “Deep learning-based channel estimation for beam space mmWave massive MIMO systems,” IEEE Wireless Commun. Lett., vol. 7, no. 5, pp. 852–855, Oct. 2018.
[18] Batool Salehi et.al, FLASH: Federated Learning for Automated Selection of High-band mmWave Sectors
[19] “FLASH Dataset,” https://genesys-lab.org/ multimodal-fusion-nextg-v2x-communications.
[20] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Acras, “Communication-Efficient Learning of Deep Networks from Decentralized Data,” in Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, ser. Proceedings of Machine Learning Research, vol. 54, 2017, pp. 1273–1282.
[21] Yae Jee Cho et al, Towards Understanding Biased Client Selection in Federated Learning, Proceedings of the 25th International Conference on Artificial Intelligence and Statistics (AISTATS) 2022, Valencia, Spain. PMLR: Volume 151.
[22] Yae Jee Cho, et.al, Bandit-based Communication-Efficient Client Selection Strategies for Federated Learning. 2020 54th Asilomar Conference on Signals, Systems, and Computers | 978-0-7381-3126-9/20/$31.00 ©2020 IEEE
[23] A. Garivier and E. Moulines, “On upper-confidence bound policies for non-stationary bandit problems,” vol. abs/0805.3415, 2008. [Online]. Available: http://arxiv.org/abs/0805.3415
[24] https://www.lasse.ufpa.br/raymobtime/
[25] A. Klautau, P. Batista, N. González-Prelcic, Y.Wang, and R.W. Heath, “5G MIMO data for machine learning: Application to beam-selection using deep learning,” in Proc. Inf. Theory Appl. Workshop, 2018, pp. 1–9.
[26] https://genesys-lab.org/multimodal-fusion-nextgv2x-communications
[27] S. Bubeck and N. Cesa-Bianchi, “Regret analysis of stochastic and non stochastic multi-armed bandit problems,” Foundations and Trends in Machine Learning, vol. 5, no. 1, pp. 1–122, 2012.
[28] R. Jain, D. Chiu, and W. Hawe, “A quantitative measure of fairness and discrimination for resource allocation in shared computer systems,” vol. cs.NI/9809099, 1998. [Online]. Available:https://arxiv.org/abs/cs/9809099
[29] R. Heinzler, P. Schindler, J. Seekircher, W. Ritter, and W. Stork, “Weather influence and classification with automotive lidar sensors,” in Proc. IEEE Intell. Veh. Symp., 2019, pp. 1527–1534

A Deep Learning Network Architecture

This section covers the architecture of the deep learning network used to develop the DL model's parameterized layers along with its properties. The blueprints for the parameterized functions $f_{C,}$ f_L f_I and f_F respectively for GPS, LiDAR, camera unimodal networks and fusion network is illustrated. Before we discuss the configuration of each of these networks, let us discuss the properties of different layers used in the DL model.

- Input Layer: This layer specifies the shape of the input data. For GPS inputs (X_c), it is (2,1) matrix giving the coordinates of the location as latitude and longitude The input form the LiDAR sensors (X_L) is a matrix of dimensions (20,20,20). For Camera Images as inputs (X_I), it is (90,160,3) matrix giving the dimension of images in RBG
- Conv Layer: These layers are designed to extract features from the input sequence by sliding filters across sequence performing a convolution operation. They are responsible for extracting features from the input data through learnable filters. The Conv 1D layer works on 1D data and only used for GPS unimodal. The Conv 2D layer works on 2D LiDAR and camera data.
- Max Pool Layer: These layers down sample the input nodes along its spatial dimensions by taking the maximum value over a window. The windows can be either 1D or 2D based on the input data. 1D windows takes strides over adjacent data points and return the maximum value within the window. 2D windows comprises windows with a height and width dimension returning the maximum value within the window. This is used to reduce the dimensionality and to extract the most significant features, making the network more efficient and reducing overfitting. These layers are marked in yellow color and there are two such layers in the network.
- Add Layer: These layers perform element-wise addition of the input tensors. This operation is part of constructing residual connections in the network, which help in alleviating the vanishing gradient problem specially for deeper networks by enabling direct paths for the gradient during backpropagation. The element wise addition is performed with corresponding tensors form these preceding input layers.
- Flatten Layer: This layer flattens the multi-dimensional output of the previous layers into a vector. It is necessary to transition from convolutional layers (which have 3D outputs) to dense layers (which have 1D outputs) and these layers help in reshaping the nodes.
- Dense Layer: These are fully connected layers that apply a linear transformation followed by a ReLU activation function. The network uses these layers to learn non-linear combinations of the high-level features extracted by the convolutional layers.
- Dropout Layer: These layers randomly set a portion of the input units to '0' during training, with a probability factor indicating the probability of an input unit being zeroed. This helps prevent overfitting by ensuring that the network cannot rely on any single feature too much. They are generally placed in-between dense layers.
- Output Layer: The output layer is generally a fully connected dense layer. The output layer of the fusion network produces the set S in Equation (14) and the number of nodes is configured to be the size of set (β). In case of unimodal networks, the output layer produces the set $Z_{c,}$ Z_L and $Z_{I.}$

A.1 GPS Unimodal Network Architecture

The Figure 57 illustrates the architecture and connections of each of the network layers of the GPS unimodal. The configurations of each of the layers are discussed below in Table 16.

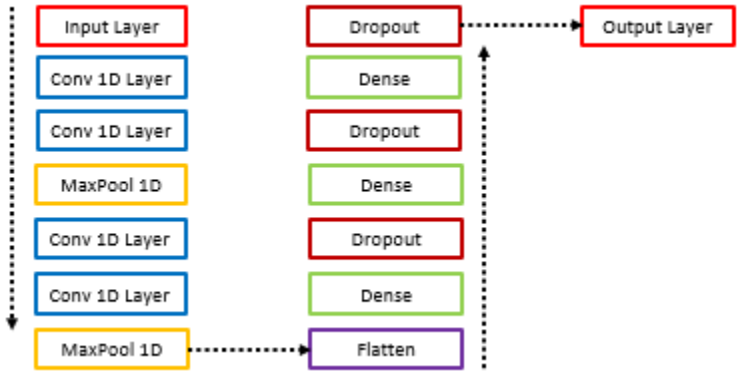

Figure 57:GPS Unimodal Network Architecture

Table 16: GPS Unimodal Network Settings

Layer #	**Layer Type**	**Output Size**	**Comments**
1	Input Layer	(2,1)	Input Layer with dimensions of (X_c)
2,3,5,6	Conv 1D Layer	(2,20), (2,20), (1,20),(1,20)	ReLU activation function and "Same" padding is used. Each layer uses 20 kernels with a size of 2.
4,7	MaxPool 1D Layer	(1,20),(1,20)	Pool Size as 2 and "Same" padding is used
8	Flatten Layer	20	
9,11,13	Dense Layer	1024,512,256	ReLU activation function is used
10,12,14	Dropout Layer	1024,512,256	Dropout Probability of 0.25 is assigned
15	Output Layer	64	TanH activation function

A.2 GPS Unimodal Network Architecture after Pruning

The need for pruning the GPS unimodal was discussed before. The overall network architecture remains the same as shown in Figure 57 but the size of the dense layers is downsized. The configurations of each of the layers after pruning are discussed below in Table 17.

Table 17: Pruned GPS Unimodal Network Settings

Layer #	Layer Type	Output Size	Comments
1	Input Layer	(2,1)	Input Layer with dimensions of (X_C)
2,3,5,6	Conv 1D Layer	(2,20), (2,20), (1,20),(1,20)	ReLU activation function and "Same" padding is used. Each layer uses 20 kernels with a size of 2.
4,7	MaxPool 1D Layer	(1,20),(1,20)	Pool Size as 2 and "Same" padding is used
8	Flatten Layer	20	
9,11,13	Dense Layer	32,16,8	ReLU activation function is used
10,12,14	Dropout Layer	32,16,8	Dropout Probability of 0.25 is assigned
15	Output Layer	2	TanH activation function

A.3 LiDAR Unimodal Network Architecture

The input form the LiDAR sensors (X_L) are of the dimensions 20×20×20. Figure 58 illustrates the architecture and connections of each of the network layers of the LiDAR unimodal. The configurations of each of the layers are discussed below in Table 18.

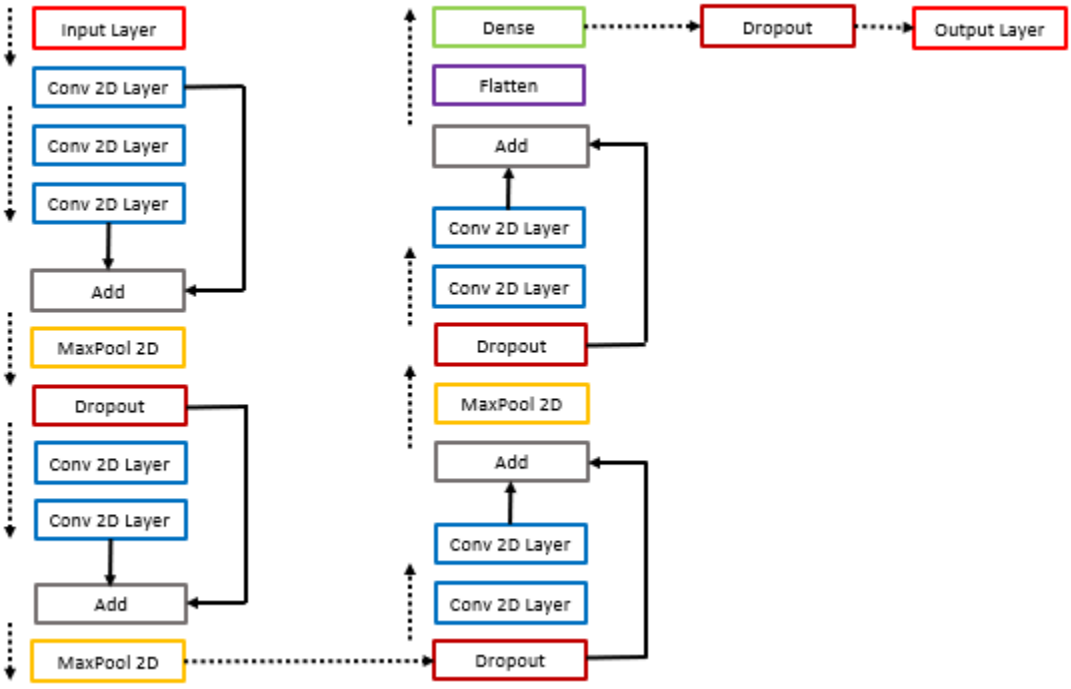

Figure 58: LiDAR Unimodal Network Architecture

Table 18: LiDAR Unimodal Network Settings

Layer #	Layer Type	Output Size	Comments
1	Input Layer	(20,20,20)	Input Layer with dimensions of (XL)
2,3,4,8,9,13, 14,18,19	Conv 2D Layer	(20,20,32),(20,20,32), (20,20,32),(10,10,32), (10,10,32), (5,5,32),(5,5,32), (5,2,32),(5,2,32)	ReLU activation function, "Same" padding and 32 Kernels of size 3×3 is used
6,11,16	MaxPool 2D Layer	(10,10,32),(5,5,32), (5,2,32)	Pool Size of (2,2) for first two layers and (1,2) for the final layer and "Same" padding is used
21	Flatten Layer	320	
22	Dense Layer	1024	ReLU activation function is used. Elastic Net with both L1 (=e-5) and L2 (=e-4) regularizer is used
7,12,17,23	Dropout Layer	(10,10,32),(5,5,32), (5,2,32),1024	Dropout Probability of 0.3 is used
5,10,15,20	Add Layer	(20,20,32),(10,10,32) (5,5,32),(5,2,32)	
Layer #	**Layer Type**	**Output Size**	**Comments**
15	Output Layer	512	ReLU activation function is used

A.4 Camera Unimodal Network Architecture

The input form the camera images are of the dimensions 90×160×3. Figure 59 illustrates the architecture and connections of each of the network layers of the Images unimodal. The configurations of each of the layers are discussed below in Table 19.

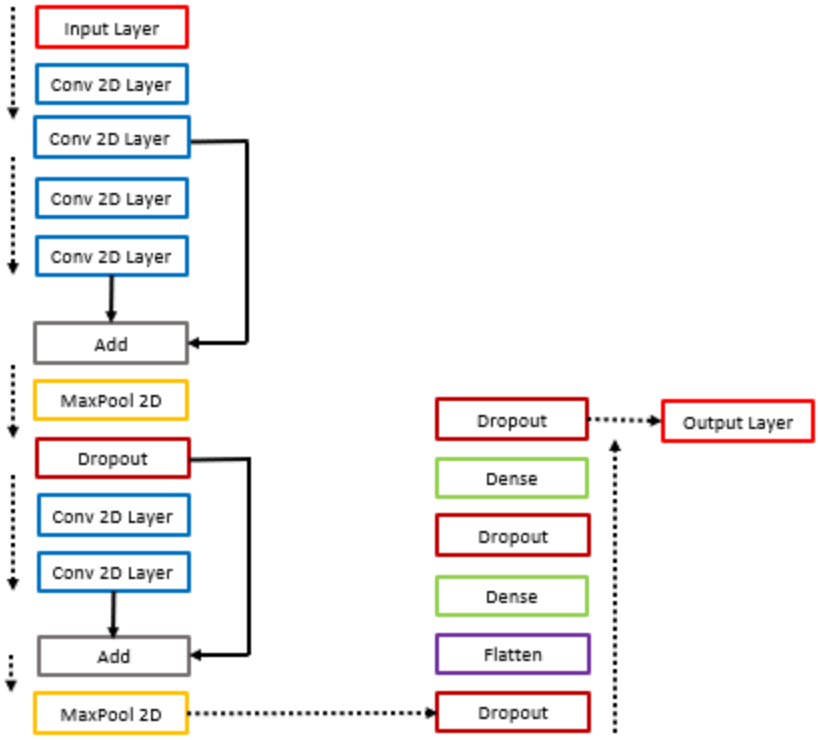

Figure 59: Camera Images Unimodal Network Architecture

Table 19: Camera Images Unimodal Network Settings

Layer #	**Layer Type**	**Output Size**	**Comments**
1	Input Layer	(90,160,3)	Input Layer with dimensions of (X_I)
2,3,4,5,8,9,10	Conv 2D Layer	(90,160,32),(90,160,32), (90,160,32),(90,160,32), (90,160,32), (45,80,32),(45,80,32)	ReLU activation function, "Same" padding and 32 Kernels of size 3×3 is used
7,12	MaxPool 2D Layer	(45,80,32),(15,26,32)	Pool Size of (2,2) for first layer and (3,3) for the second layer and "Same" padding is used
Layer #	**Layer Type**	**Output Size**	**Comments**
14	Flatten Layer	12480	
15,17	Dense Layer	6390272,131328	ReLU activation function is used
8,13,16,18	Dropout Layer	(45,80,32),(15,26,32), 512,256	Dropout Probability of 0.25 is used
6.11	Add Layer	(90,160,32),(45,80,32)	

15	Output Layer	256	TanH activation function is used

A.5 Fusion Network

The high-level feature vectors extracted by each unimodal network is concatenated to provide input to the fusion network. Considering all three modalities working, there are 832 inputs to the network. Considering only LiDAR and GPS working, there are 576 inputs to the network. The architecture of the network remains the same for both these configurations. Figure 60 illustrates the architecture and connections of each of the network layers of the fusion network. The configurations of each of the layers are discussed below in Table 20.

Table 20: Fusion Network Settings

Layer #	**Layer Type**	**Output Size**	**Comments**
1 (LiDAR, Camera, GPS concatenated)	Input Layer	576/832	Input Layer with dimensions of (X_I)
(2,4,6,8)	Dense Layer	1024,512,256,128	ReLU activation function is used. L2 (=e- regularizer is used
(3,5,7,9)	Batch Normalization	1024,512,256,128	
10	Output	64	SoftMax activation function is used

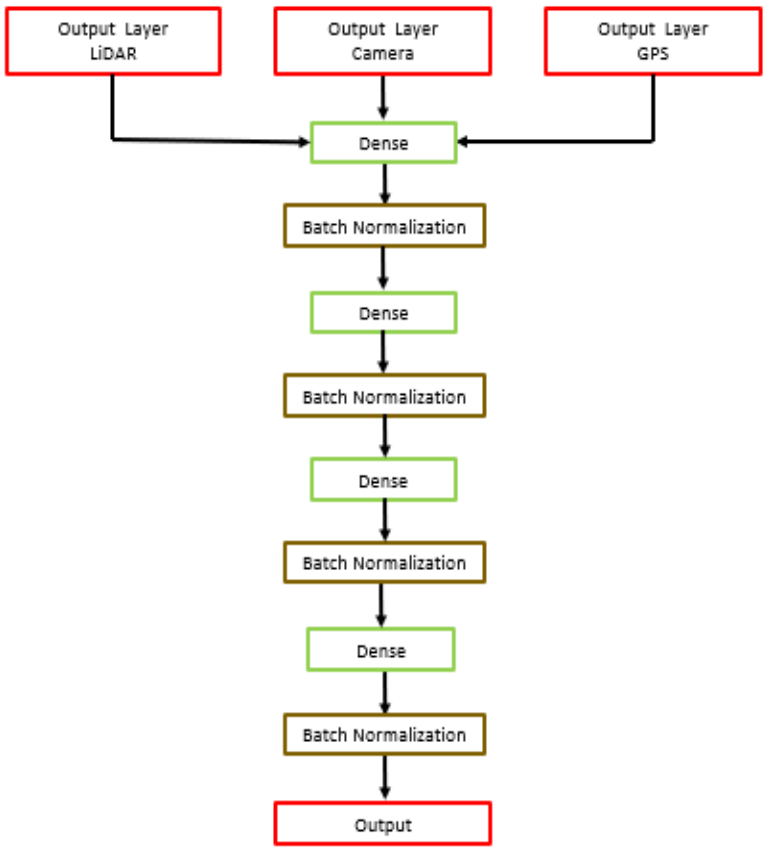

Figure 60: Fusion Network Architecture

B Distributed Architecture Implementation

B.1 Python Package Dependencies

Table 21 shows all the python library dependencies along with the version needed to run the python implementation.

Table 21: Python Package Dependencies

Package Name	Version
absl-py	1.4.0
astunparse	1.6.3
ca-certificates	2023.01.10
cachetools	5.3.0
certifi	2022.12.7
charset-normalizer	3.0.1
colorama	0.4.6
flatbuffers	23.1.21
gast	0.4.0
google-auth	2.16.1
google-auth-oauthlib	0.4.6
google-pasta	0.2.0
grpcio	1.51.3
h5py	3.8.0
idna	3.4
importlib-metadata	6.0.0
joblib	1.2.0
keras	2.11.0
libclang	15.0.6.1
markdown	3.4.1
markupsafe	2.1.2
numpy	1.21.6
oauthlib	3.2.2
openssl	1.1.1t
opt-einsum	3.3.0
packaging	23
pandas	1.3.5
pip	22.3.1
protobuf	3.19.6
pyasn1	0.4.8
pyasn1-modules	0.2.8
python	3.7.16
python-dateutil	2.8.2

pytz	2023.3
requests	2.28.2
Package Name	**Version**
requests-oauthlib	1.3.1
rsa	4.9
scikit-learn	1.0.2
scipy	1.7.3
setuptools	65.6.3
six	1.16.0
sqlite	3.40.1
tensorboard	2.11.2
tensorboard-data-server	0.6.1
tensorboard-plugin-wit	1.8.1
tensorflow	2.11.0
tensorflow-estimator	2.11.0
tensorflow-intel	2.11.0
tensorflow-io-gcs-filesystem	0.30.0
termcolor	2.2.0
threadpoolctl	3.1.0
tqdm	4.64.1
typing-extensions	4.5.0
urllib3	1.26.14
vc	14.2
vs2015_runtime	14.27.29016
werkzeug	2.2.3
wheel	0.38.4
wincertstore	0.2
wrapt	1.14.1
zipp	3.14.0

B.2 Steps to run the Distributed Implementation

1. Download the pre-processed Infocom FLASH dataset.
2. Create a python environment with the above packages.
3. Clone the repository from : https://github.com/ImAbishekSubramanian/Federated-Learning-MMWave-Beamforming.
4. Copy the downloaded data in “empty” data folder within the repository.
5. Run generate_randperm.py file. (Command: python generate_randperm.py) Make sure the path is pointing to the data folder in Step 4. For example : *data_path = r"C:/Thesis/FDL1/data/"*.
6. Run all_train_validate.py file. (Command: python all_train_validate.py). Make sure the data path and save path is pointing to the right folder. An “empty” folder

all_train_val which is added to the repository can be used to save. For example: *data_path = r"C:/Thesis/FDL1/data/" ,save_path = 'C:/Thesis/FDL1/all_train_val/'* .

7. Run all_test.py file. (Command: python all_test.py Make sure the data path and save path is pointing to the right folder. An "empty" folder all_test which is added to the repository can be used to save. For example: *data_path = r"C:/Thesis/FDL1/data/" ,save_path = 'C:/Thesis/FDL1/all_test/'* .
8. Steps 7 and 8 generate the local train, local test and global test data.
9. Run distributed_learning.py (Command: python distributed_learning.py).
10. The file distaributed_learning.py has configuration parameters as listed in Table 22. Refer them to make appropriate configuration changes to the model.
11. Additionally, you can run distributed_learning_PD.py (Command: python distributed_learning_PD.py) for training the model post deployment.

Table 22: Python Configuration Parameters

Configuration Parameter	**Use**	**Default values**
id_gpu	GPU id to be used if GPU is present. The model can run without GPU.	1
data_folder	Path to data folder	Point it to the data folder (Step 5)
input	To select the modalities to use	"coord", "lidar"
epochs	Number of epochs to train local models	100
lr	Learning Rate	0.0001
bs	Batch size	32
shuffle	To enable shuffling while training	True
strategy	Labeling strategy to use	"one_hot"
model_folder	Path to save trained model	Point to the model folder (Available in repository)
image_feature_to_use	Feature images to use	"raw"
experiment categories	Infocom FLASH data categories to use. Cat 1 is LOS samples and the	'Cat1','Cat2','Cat3','Cat4'

	other are NLOS samples	
Configuration Parameter	**Use**	**Default values**
experiment_episodes	Clients to be used for training	'0','1','2','3','4','5','6','7','8','9'
test_all_path	Path to global test data in Step 7	
train_val_path	Path to local train and local test data in Step 6	
latest_step	To resume training previously started	0
aggregating_method	Biased/Unbiased client selection	“biased”
biased_method	MaxLoss or Heuristic MAB	“MAB”
m	Subset of client size for training	4
gamma	Hyperparameter “ϒ”	0.3
training_round	Federated Training Rounds	300

C Portage (HPC) Boilerplate Description

Michigan Tech's shared high-performance computing infrastructure, *Superior*, is available to all researchers. It has the following computing and storage components:

1. Generation 1.0 (acquired between 2013/06 - 2015/10)
   a) 92 CPU compute nodes - each having 16 CPU cores (Intel Xeon E5-2670 2.60 GHz) and 64 GB RAM - providing 30 TFLOPS
   b) 4 CPU compute nodes - each having 24 CPU cores (Intel Xeon E4-2680 2.50 GHz) and 256 GB RAM - providing 2 TFLOPS
   c) 1 storage node with 32 TB shared usable space
2. Generation 2.0 (acquired between 2017/06 - 2018/08)
   a) 85 CPU compute nodes - each having 32 CPU cores (Intel Xeon E5-2683 2.10 GHz) and 256 GB RAM - providing 91 TFLOPS

Portage is another shared high-performance computing infrastructure and a miniature version of *Superior*. Intended primarily for testing, educational (course work and/or senior design projects) and gateway/preliminary research projects involving non-confidential/non-sponsored data, *Portage* has 3 TFLOPS of CPU computing capacity with hardware identical to *Superior*'s generations 1.0 and 2.0. The specification of "Portage" HPC is listed in Table 23.

Table 23: "Portage" HPC specification

Specification	**Configuration**
OS flavor	CentOS Linux release 7.9.2009 (Core)
Hardware generation	core_avx2
Processor type	x86_64
Kernel release	3.10.0-1160.el7.x86_64
RAM (total)	251 GB
CPU model	Intel(R) Xeon(R) CPU E5-2683 v4 @ 2.10GHz
CPU count	32