BEAMFORMING FOR V2V COMMUNICATION UTILIZING DEEP LEARNING

A PROJECT REPORT

Submitted by

ARAVINDAN S M	**2020504515**
RAMANA SRIVATS S	**2020504568**
SHRI HARISH S	**2020504583**

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

in

ELECTRONICS AND COMMUNICATIONS ENGINEERING

DEPARTMENT OF ELECTRONICS ENGINEERING

MADRAS INSTITUTE OF TECHNOLOGY, CHROMPET

ANNA UNIVERSITY: CHENNAI 600 044

MAY 2024

ANNA UNIVERSITY: CHENNAI 600 044

BONAFIDE CERTIFICATE

Certified that this project report **"BEAMFORMING FOR V2V COMMUNICATIONS UTILIZING DEEP LEARNING"** is the bonafide work of

ARAVINDAN S M	**2020504515**
RAMANA SRIVATS S	**2020504568**
SHRI HARISH S	**2020504583**

who carried out the project work under my supervision.

SIGNATURE

Dr. P. INDUMATHI

HEAD OF THE DEPARTMENT

Professor

Department of Electronics Engineering,

Madras Institute of Technology,

Anna University

Chennai 600 044

SIGNATURE

Mrs. P. KASTHURI

SUPERVISOR

Teaching Fellow

Department of Electronics Engineering,

Madras Institute of Technology,

Anna University,

Chennai – 600 044

ACKNOWLEDGEMENT

We consider it as our privilege and our primary duty to express our gratitude and respect to all those who guided and inspired us in the successful completion of the project.

We owe solemn gratitude to **Dr. K. RAVICHANDRAN**, Dean, Madras Institute of Technology, for having given consent to carry out the project work at MIT Campus, Anna University.

We wish to express our sincere appreciation and gratitude to **Dr. P. INDUMATHI**, Professor and Head of the Department of Electronics Engineering, who has encouraged and motivated us in our endeavours.

We are extremely grateful to our project guide **Mrs. P. KASTHURI**, Teaching Fellow, Department of Electronics Engineering, for his timely and thoughtful guidance and encouragement for the completion of the project.

We sincerely thank all our panel members **Dr. M. VASIM BABU**, Assistant Professor, **Dr. S. RAMPRABHU**, Assistant Professor and **Dr. G. BALAMURUGAN,** Assistant Professor of the Department of Electronics Engineering, for their valuable suggestions.

We sincerely thank our project coordinator **Mrs**. **U. Vijayalakshmi** for her valuable suggestions. We also thank all the teaching and non-teaching staff members of the Department of Electronics Engineering for their support in all aspects.

PLACE : Chennai	**ARAVINDAN S M**	**2020504008**
DATE :	**RAMANA SRIVATS S**	**2020504012**
	SHRI HARISH S	**2020504027**

TABLE OF CONTENTS

ABSTRACT

Vehicle-to-Vehicle (V2V) communication systems hold high significance for enhancing road safety and efficiency. In this work, we utilized an existing dataset comprising GPS latitude and longitude coordinates and corresponding ideal beams which gives least power loss to train a machine learning model. By extracting features such as distance between the vehicles and the angle of arrival between the two cars from the dataset, six machine learning models were trained which are ANN, SVM, KNN, Random Forest, XGBoost and decision tree to predict the optimal beam which gives the lowest average power loss and highest accuracy. Here, the technique used for evaluating the performance of the models is top-k metric. Our analysis reveals that Random Forest emerged as the most accurate model with a Top-5 accuracy of 85.37% and an average power loss of -1.45dB followed by the Decision Tree Model which has a Top-5 accuracy of 74.9% with an average power loss of -2.82dB. While Random Forest exhibits superior performance in terms of accuracy and power loss reduction, the Decision Tree model also presents as a good alternative, particularly because it has relatively lower computational complexity. Hence by using machine learning model ideally Random Forest we efficiently predict optimal beams and reduce training overhead and latency unlike traditional methods.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATION

ABBREVIATION	EXPANSION
V2V	Vehicle-to-Vehicle
GPS	Global Positioning System
RADAR	Radio Detection And Ranging
LiDAR	Light Detection and Ranging
AI	Artificial Intelligence
SVM	Support Vector Machines
KNN	K-Nearest Neighbors
ANN	Artificial Neural Network
mm-Wave	Millimeter Wave
APL	Average Power Loss
FMCW	Frequency-modulated continuous wave
AoA	Angle of Arrival
SRE	Smart Radio Environment
ML	Machine Learning

CHAPTER 1

INTRODUCTION

Vehicle-to-Vehicle (V2V) communication is a groundbreaking technology that revolutionizes the way vehicles interact with each other on the road. Unlike traditional communication methods that rely on centralized infrastructure such as roadside towers or satellite networks, V2V communication enables vehicles to directly exchange information with nearby vehicles in real-time.

One of the key benefits of V2V communication is its potential to significantly improve road safety. By sharing critical data like vehicle speed, direction, and braking status, neighbouring vehicles can anticipate and react to potential hazards much faster than human drivers alone. This proactive approach to safety can help prevent accidents and reduce the severity of collisions, ultimately saving lives and reducing the economic costs associated with traffic accidents.

Moreover, V2V communication has the capacity to optimize traffic flow and alleviate congestion on roadways. Vehicles equipped with V2V technology can share traffic information such as road conditions, congestion levels, and optimal routes with each other. This allows drivers to make informed decisions about their travel routes in real-time, leading to more efficient use of road infrastructure and smoother traffic flow.

In addition to enhancing safety and efficiency, V2V communication paves the way for the development of intelligent transportation systems (ITS). These systems leverage advanced technologies like artificial intelligence, machine learning, and big data analytics to further improve the performance and reliability of transportation networks.

1.1. DATA SHARING IN V2V COMMUNICATION

Data sharing plays a crucial role in Vehicle-to-Vehicle (V2V) communication, essential for enabling efficient and safe navigation on roadways. It encompasses the exchange of various types of information among vehicles engaged in V2V communication, facilitating seamless interaction and collaboration on the road. At its core, data sharing entails the transmission and reception of critical data such as speed, position, acceleration, and other relevant factors in real-time. This data is gathered through onboard sensors like GPS, radar, and LiDAR, which are integrated into modern vehicles. These sensors provide vehicles with accurate and up-to-date information about their surroundings, enabling them to understand their relative positions, detect obstacles, and anticipate changes in traffic conditions. By sharing this information with other vehicles, V2V communication systems empower collaborative decision-making and proactive safety measures, ultimately contributing to enhanced road safety and efficiency.

1.2. APPLICATIONS OF V2V COMMUNICATION

V2V communication opens the door to a wide array of impactful applications that revolutionize the way vehicles interact on the road. These applications range from enhancing individual vehicle safety to optimizing traffic flow and management.

One such application is cooperative adaptive cruise control, where vehicles communicate with each other to maintain safe following distances and adjust speeds accordingly. This technology can significantly reduce the risk of rear-end collisions and improve overall traffic flow on highways and congested roadways.

Additionally, intersection collision warning systems leverage V2V communication to alert drivers of potential collisions at intersections, helping to prevent accidents and improve intersection safety. Another crucial application is emergency vehicle notification, where nearby vehicles are alerted when an emergency vehicle approaches, allowing them to make way and clear a path for the emergency vehicle to pass safely and swiftly. These are just a few examples of the diverse applications enabled by V2V communication, all of which contribute to safer roads, more efficient traffic management, and ultimately, a better transportation experience for everyone.

1.3. IMPROVING BEAMFORMING EFFICIENCY

Significant efforts have been dedicated to improving beamforming efficiency in V2V communication systems through the integration of position-related information. Advanced methodologies, including machine learning and deep learning techniques, have been extensively explored to optimize beamforming processes and mitigate complexity. These initiatives are driven by the objective of maximizing the efficiency and reliability of V2V communication systems in practical, real-world scenarios. By harnessing sophisticated algorithms and leveraging spatial data, researchers aim to enhance the precision and speed of beamforming operations, ultimately contributing to the seamless exchange of information among connected vehicles. Some of the ways by which Beamforming efficiency can be improved is:

1.3.1 INTEGRATION OF POSITION-RELATED INFORMATION

Incorporating position-related information, such as GPS coordinates or relative vehicle positions, enhances beamforming efficiency in V2V communication systems. This spatial data optimizes beam directionality, reducing interference and improving signal strength. Understanding vehicle locations allows for better beam alignment, minimizing multipath effects and maximizing signal quality. Dynamic adaptation to changing conditions ensures robust communication links, enhancing reliability in V2V scenarios.

1.3.2 OPTIMIZING BEAMFORMING WITH ADVANCED MACHINE LEARNING TECHNIQUES

Machine Learning and Deep Learning offer transformative potential for optimizing beamforming processes in V2V communication systems. ML algorithms, including reinforcement learning, enable adaptive beamforming strategies that dynamically adjust to changing environmental conditions and user requirements. Additionally, DL techniques, such as convolutional neural networks, excel at spatial feature extraction, facilitating accurate beam directionality determination. By harnessing these advanced methodologies, V2V systems can achieve enhanced performance, robustness, and adaptability in diverse operating scenarios.

1.3.3 EFFICIENCY ENHANCEMENT THROUGH COMPLEXITY MITIGATION

Complexity poses a significant challenge in beamforming algorithms, particularly in V2V communication systems where real-time processing is essential. Advanced methodologies strive to mitigate this challenge by employing various strategies to reduce computational complexity while maintaining beamforming performance. Techniques such as compressed sensing enable the reconstruction of sparse signals from fewer samples, thereby reducing computational burden without sacrificing accuracy.

1.4. MACHINE LEARNING IN V2V COMMUNICATION

In Vehicle-to-Vehicle (V2V) communication, machine learning involves creating models and algorithms that allow for autonomous learning and decision-making without the need for explicit programming. These strategies use statistical and computational approaches to improve performance using already existing data. Machine learning has applications in several sectors related to V2V communication which includes are:

Beamforming Optimization: In V2V communication systems, ML algorithms dynamically modify beam directions to boost signal strength and lower interference, therefore increasing communication dependability.

Channel Prediction: By using past data to forecast channel characteristics in real-time, machine learning models allow vehicle-to-vehicle (V2V) systems to modify their communication tactics in response to shifting environmental conditions.

Predicting The Future Position of Vehicles: Machine learning in V2V communication can also predict the future position of vehicles. By analysing historical movement patterns and contextual data such as speed, acceleration, and direction, machine learning models can forecast the trajectory of vehicles in real-time. This predictive capability enables V2V systems to anticipate potential collision scenarios, optimize routing decisions.

1.5. SUPERVISED MACHINE LEARNING FOR V2V COMMUNICATION

Supervised machine learning algorithms encompass a range of techniques designed to learn patterns and relationships from labelled data. Linear regression, a fundamental algorithm, models the relationship between dependent and independent variables by minimizing the sum of squared errors. Logistic regression, on the other hand, is tailored for binary classification tasks, estimating the probability of a sample belonging to a particular class. Decision trees offer versatility, capable of handling both classification and regression tasks by recursively partitioning the feature space based on simple decision rules. However, their susceptibility to overfitting is mitigated by ensemble methods like Random Forests, which aggregate multiple decision trees to improve robustness. Support Vector Machines (SVM) excel in finding optimal hyperplanes for separating classes in high-dimensional spaces, while K-Nearest Neighbours (KNN) relies on the majority vote of neighbouring points to make predictions.

Each algorithm has its own set of strengths and weaknesses, and selecting the most suitable one depends on factors such as dataset characteristics, problem complexity, and computational resources. Through experimentation and evaluation, practitioners can identify the most effective algorithm for a given task.

1.6. OBJECTIVE OF THE PROJECT

The project aims to revolutionize Vehicle-to-Vehicle (V2V) communication systems by developing a machine learning-based model to predict optimal beam pairs solely using GPS data, thus reducing computational overhead, and selecting beams with minimal average power loss. Leveraging deep learning techniques like convolutional and recurrent neural networks, alongside ensemble learning methods, the project seeks to identify the most effective approach for accurate and efficient beam prediction. By analysing spatial correlations in GPS data and integrating information such as vehicle speed, acceleration, and angle of arrival, the model aims to capture factors influencing beamforming performance, this project also aims in exploring diverse driving scenarios, including interstate and intercity conditions to ensure the model's robustness and generalization across various environments. Ultimately, the goal is to advance V2V communication networks, enhancing road safety and efficiency through predictive beam selection in real-world driving contexts.

CHAPTER 2

LITERATURE SURVEY

Muhammad Baqer Mollah, Honggang Wang, and Hua Fang [1] (2024) presented their work titled "Position Aware 60 GHz mmWave Beamforming for V2V Communications Utilizing Deep Learning" at the IEEE International Conference on Communications (ICC) in Denver, CO, USA. This work introduces an innovative solution designed to bolster downlink received power in vehicle-to-vehicle (V2V) connectivity by harnessing the capabilities of 60 GHz mmWave communications. At the core of their solution lies a sophisticated deep learning model, strategically crafted to predict a subset of beams based on precise vehicular position information. By leveraging this approach, the research team aims to significantly reduce beam search space and associated overheads, thus optimizing the beamforming process in mmWave V2V communications. Through meticulous experimentation conducted on real-world datasets, the proposed approach showcases remarkable advancements in key performance metrics, notably accuracies and received power ratios. In direct comparison to traditional baseline methods, the deep learning model emerges as a frontrunner, outperforming its counterparts and underscoring the tangible benefits of incorporating position information for beamforming in mmWave V2V communications. These findings not only address the critical challenge of minimizing beam search overheads but also offer invaluable insights into streamlining V2V communication systems for heightened connectivity and operational efficiency. The results of their research present a compelling narrative, with the proposed solution achieving an average of 84.58% of the received power of link status. This remarkable outcome underscores the transformative potential of deep learning methodologies in revolutionizing the landscape of high-frequency mmWave transmissions.

Muhammad Alrabeiah and Ahmed Alkhateeb [2] (2019) authored "DeepSense 6G: A Large-Scale Real-World Multi-Modal Sensing and Communication Dataset" published in IEEE Communications Magazine. This research's primary objective is of addressing and mitigating prevalent challenges through the ingenious application of vision-aided deep learning models. In a landscape where the reliability and efficiency of mmWave communications are often compromised by factors such as blockages and dynamic environmental conditions, this study emerges as a beacon of innovation and resilience. By harnessing the powerful capabilities inherent in cameras and RGB images, these sophisticated deep learning models are meticulously crafted to predict optimal beam selection and swiftly detect blockages in real-time scenarios. The envisioned solutions offered by these models are engineered to navigate through the complexities of interference, seamlessly mitigating its effects and steadfastly upholding robust communication links even amidst the most challenging environmental fluctuations. Through rigorous and comprehensive experimental evaluations conducted with meticulous attention to detail, the research community unveils the extraordinary accuracy and efficacy inherent in these vision-aided deep learning models. Moreover, the findings underscore the myriad potential benefits that these models bring forth, particularly in dynamic environments characterized by the presence of multiple users and ever-evolving conditions. In essence, by seamlessly integrating sophisticated computer vision techniques into the intricate fabric of mmWave systems, these visionary models contribute significantly to enhancing the overall robustness and reliability of communications. The transformative impact of these endeavors reverberates across various dimensions, manifesting in tangible improvements in link quality, the attainment of higher data rates, and the implementation of optimized resource allocation strategies.

Muhammad Alrabeiah, Andrew Hredzak, and Ahmed Alkhateeb [3] (2019) authored "Millimeter Wave Base Stations with Cameras: Vision Aided Beam and Blockage Prediction" presented at the IEEE Vehicular Technology Conference. This paper explores how vision technology can assist in solving challenges in wireless communication, especially in millimeter wave (mmWave) systems used in 5G and beyond. These systems face issues like beam selection and link blockages. Since many devices using mmWave arrays also have cameras, the paper investigates how cameras at mmWave base stations can predict beams and blockages using computer vision and deep learning. The system model considers a base station communicating with a single user. For beam prediction, the goal is to select the best beamforming vector to maximize signal quality at the receiver. For blockage prediction, it's essential to determine if the user's line-of-sight (LOS) link is blocked. The paper proposes using RGB images and sub-6 GHz channels to predict beams and detect blockages. The proposed solutions involve deep learning, particularly using a ResNet-18 model trained on ImageNet data and fine-tuned for the specific tasks. For beam prediction, the model learns to map images to beam indices, while for blockage prediction, it detects the presence of users and assesses link status using sub-6 GHz channels.Simulation results show promising performance, with high accuracy achieved even with relatively small training datasets. The paper concludes by highlighting the potential of vision-based approaches in improving the mobility and reliability of mmWave systems, suggesting future work to explore dynamic environments and multiple users.

Hao Luo, Umut Demirhan, and Ahmed Alkhateeb [4] (2023) presented "Millimeter Wave V2V Beam Tracking using Radar: Algorithms and Real-World Demonstration" at the 31st European Signal Processing Conference (EUSIPCO). This paper explores the utilization of radar sensing to assist communication in dynamic environments, particularly focusing on vehicle-to-vehicle (V2V) millimeter wave and terahertz communication scenarios. These scenarios require accurate alignment of narrow beams for high data transfer rates, posing challenges in highly-mobile vehicular settings.To address this problem, the authors develop a radar-aided beam-tracking framework where a single initial beam and a set of radar measurements over time are used to predict future beams. They propose two approaches combining radar signal processing and machine learning: one involving classical radar signal processing to identify radar states and predict beams through a machine learning model, and another end-to-end machine learning approach using radar maps directly as input to predict beams.The paper formalizes the radar-aided beam tracking problem considering practical communication and radar models, and describes the communication model between vehicles and the radar model aiding communication. It defines the problem of predicting the optimal beam based on radar measurements and the initial beam index, aiming to optimize a mapping function and parameters to maximize beam prediction accuracy.The proposed solutions are evaluated using a real-world V2V dataset collected from the DeepSense 6G framework. The dataset includes measurements from mmWave antenna arrays and FMCW radars placed on vehicles. The evaluation involves training neural network models for transmitter identification and end-to-end beam prediction, and testing their performance on real-world data.Results indicate that while the transmitter identification-based approach shows limitations due to low radar angular resolution, the end-to-end machine learning approach outperforms baseline methods, demonstrating the potential of radar-aided beam tracking for V2V communication in challenging real-world scenarios.

Hao Ye, Geoffrey Ye Li, and Biing-Hwang Fred Juang [5] (2019) published "Deep Reinforcement Learning Based Resource Allocation for V2V Communications" in the IEEE Transactions on Vehicular Technology. This paper introduces a communication approach based on deep reinforcement learning, suitable for both unicast and broadcast scenarios in vehicle-to-vehicle (V2V) communications. The proposed decentralized resource allocation mechanism involves autonomous "agents," representing V2V links or vehicles, making decisions to optimize sub-band selection and power allocation without requiring global information. As a decentralized method, it incurs minimal transmission overhead. Simulation results demonstrate the effectiveness of the approach in meeting stringent latency constraints on V2V links while minimizing interference with vehicle-to-infrastructure communications.

The authors explore various innovative approaches to address challenges in millimeter-wave (mmWave) communications, specifically focusing on vehicle-to-vehicle (V2V) scenarios. Alrabeiah, Alkhateeb, and Hredzak (2019) investigate the use of base station cameras for beam selection and blockage prediction, achieving high accuracy with minimal overhead. Luo, Demirhan, and Alkhateeb (2023) develop a radar-aided beam-tracking framework, demonstrating the potential of radar in V2V beam management. Ye, Li, and Juang (2019) propose a decentralized resource allocation mechanism based on deep reinforcement learning for V2V communications, optimizing transmission parameters autonomously while minimizing interference. Finally, Baqer Mollah, Wang, and Fang (2024) introduce a position-aware mmWave beamforming solution using deep learning, reducing latency in V2V communication. These studies collectively advance mmWave V2V communication capabilities, offering insights and solutions to overcome challenges and drive innovation in the field.

CHAPTER 3
EXISTING METHODOLOGY

In real-world scenarios, using conventional beam selection methods, vehicles typically choose the optimal beam pairs through a detailed beam measurement process. This process can add significant computing load and latency, particularly in highly dynamic vehicular environments. This overhead arises because of the limited time frame within which the receiver can receive correct packets, frequent beam realignments, and the changing state of the channel, all of which require beam recalculations. Consequently, to harness the full benefits of mmWave communications, it is essential to reduce beam misalignments and lower overheads.

In response to these challenges, particularly in vehicular contexts, recent studies have proposed innovative methods to establish communication links by utilizing out-of-band contextual information.

3.1. BASE LINE SOLUTION

The baseline solution involves the manually calculating the beam strength and the average power loss for all the beams individually and selecting the best beams for each configuration of the two cars.

It can be split in three simple phases:

1. Use input positions to estimate car1 and car2 positions in the new timestamp (via linear extrapolation)

2. Use estimated positions to estimate angle of arrival

3. Use estimated angle of arrival to estimate optimal beam (assuming the beams are uniformly distributed in angular domain)

The baseline was applied to the four scenarios (scenarios 36-39). The results include the average power loss (APL) and top-k accuracy.

Regarding the performance of this baseline, the baseline is limited in several ways; for example, it considerably underperforms when both cars are static, which is common in traffic lights.

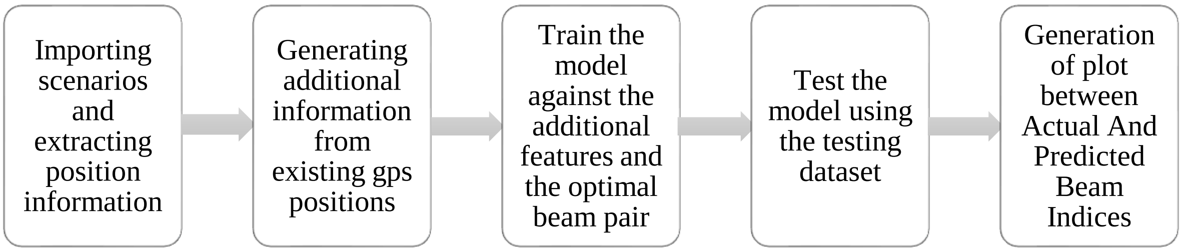

Figure 3.1 Flowchart for Optimal Beam Prediction Using ML

3.2. DEEPSENSE 6G DATASET

The DeepSense 6G dataset is a collection of real-world scenarios that includes multi-modal sensing and communication data. The dataset is intended to help advance deep learning research in a variety of applications that combine multi-modal sensing, communication, and positioning.

Each scenario in the DeepSense 6G dataset is a standalone dataset that's created from a single, usually lengthy, data collection session. The data is formatted in a generic way that's consistent across all scenarios. The data collection for each scenario is intended to cover a significant deployment scenario and enable one or more applications. In this work scenario 36-39 has been used.

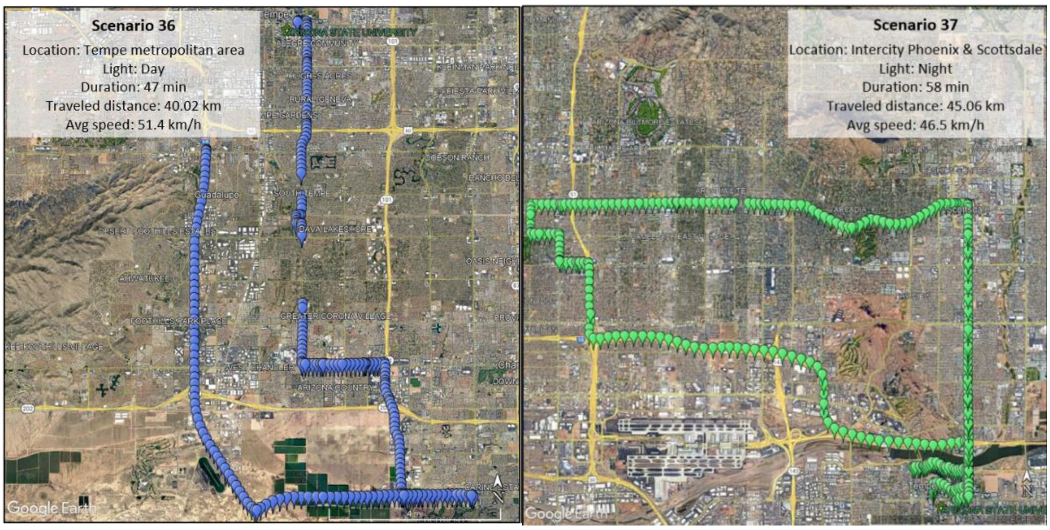

Figure 3.2 InterCity Scenarios

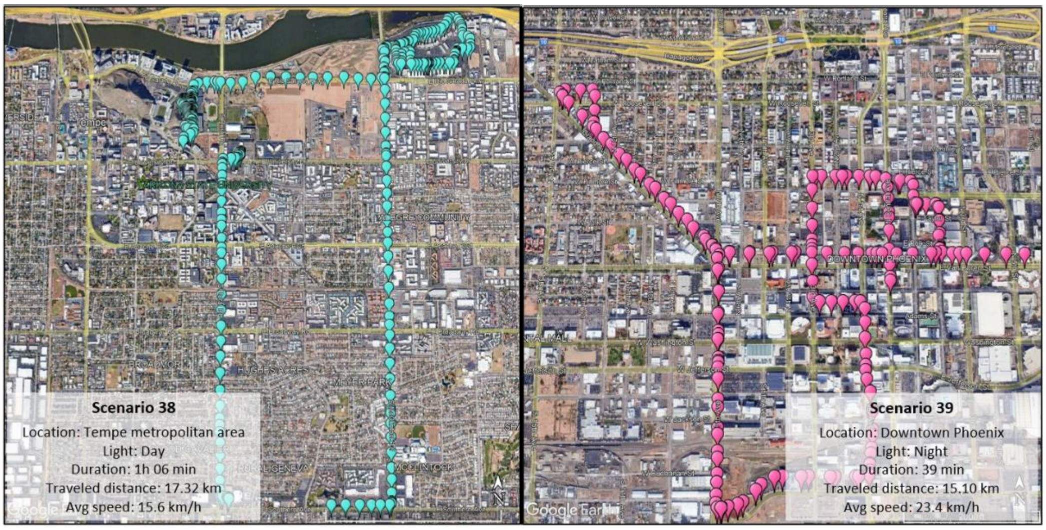

Figure 3.3 Long Distance Interstate Scenarios

The dataset is collected using DeepSense 6G. The testbed comprises two units: Unit 1, a mobile receiver (vehicle) equipped with advanced sensing capabilities, and Unit 2, a mobile transmitter.

Figure 3.2 shows scenarios 38 and 39 which are predominantly focused on emulating short urban commutes, capturing data primarily within city boundaries.

Figure 3.3 shows scenarios 36 and 37 which are collected in long drives between cities, targeting long travels, and are referred to as inter-city scenarios.

The distinction between these scenarios is supported by the variations in traveled distances and average speeds of the vehicles. Scenarios 36 involve traveling long distances at relatively high average speeds, while Scenarios 38 covers shorter distances at lower speeds due to speed limits imposed within cities.

abs_index	timestamp	seq_index	unit1_gps1	unit1_gps1_lat	unit1_gps1_lon	unit1_gps1_altitude	unit1_gps1_hdop	unit1_gps1_pdop	unit1_overall-beam
2674	11-46-31.214536	1	unit1/gps1/gps_8869_11-46-31.233334.txt	33.42170563	-111.9301714	354.8673333	0.5	1.07	162
2675	11-46-31.314452	1	unit1/gps1/gps_8870_11-46-31.300000.txt	33.42170564	-111.9301714	354.863	0.5	1.07	161
2676	11-46-31.414397	1	unit1/gps1/gps_8871_11-46-31.400000.txt	33.42170565	-111.9301714	354.863	0.5	1.07	162
2677	11-46-31.514787	1	unit1/gps1/gps_8872_11-46-31.500000.txt	33.42170565	-111.9301714	354.858	0.5	1.07	162
2678	11-46-31.615223	1	unit1/gps1/gps_8873_11-46-31.600000.txt	33.42170565	-111.9301714	354.851	0.5	1.07	161
2679	11-46-31.714998	1	unit1/gps1/gps_8874_11-46-31.700000.txt	33.42170565	-111.9301714	354.854	0.5	1.07	161
2680	11-46-31.814664	1	unit1/gps1/gps_8875_11-46-31.800000.txt	33.42170565	-111.9301714	354.853	0.5	1.07	162
2681	11-46-31.914686	1	unit1/gps1/gps_8876_11-46-31.900000.txt	33.42170566	-111.9301715	354.851	0.5	1.07	161
2682	11-46-32.015005	1	unit1/gps1/gps_8877_11-46-32.000000.txt	33.42170568	-111.9301714	354.844	0.5	1.07	162
2683	11-46-32.114994	1	unit1/gps1/gps_8879_11-46-32.133334.txt	33.42170567	-111.9301714	354.8426667	0.5	1.07	161

Figure 3.4 Sample GPS DeepSense 6G Dataset

3.3. EXPLANATION OF DATASET

abs_index: Represents the sample absolute index. This is the sample index that is displayed in the video. This index remains unchanged from the sample collection all the way to the processed scenario. The advantage is a perfect backtrace of information across processing phases. Moreover, if problems with the data exist, mentioning this sample index allows easier interactions with the DeepSense team.

Timestamp: This column represents the time of data capture in “hr-mins-secs.us” format with respect to the current time zone. It not only tells the time of the data collection but also tells the time difference between sample captures.

seq_index: The index of a sequence. Samples have different sequence indices for two reasons. The main reason is to separate continuous sampling intervals. Samples with the same index have been collected at precisely the same sampling rate (10 Hz in this case). As such, when the sequence index changes, it is because some circumstances of the data collection lead to a sample loss or similar, thus disrupting the continuity. However, it is possible that adjacent samples with different sequence indices have a 100 ms interval between them, thus actually respecting the continuity. In this case, the second reason for changing sequences applies. The second reason for changing sequences is to create 20-second chunks of data that can be separated for training and testing purposes. No sequence is larger than 20 s (200 samples) because of this reason. If the last sample of a sequence and the first sample of the next have consecutive absolute indices (abs_index column), then the sequence changed because of the 20 s rule, not because of a break in continuity.

Sensors can be gps1, pwr2, lidar1, radar3, etc. Sensors are associated with units. unit1_gps1 refers to the first GPS of unit 1. unit2_gps1 refers to the first GPS of unit 2. In these scenarios, only radar and power have more than one sensor per unit.. Therefore, unit1_pwr1 will have information about the phased array in the front, and unit1_radar4 will have the raw radar samples from the radar pointing to the left. Sensors may additionally have labels. Labels are metadata acquired from that sensor. In these scenarios, only two sensors have metadata, and the labels are as follows:

gps1_lat / gps1_lon / gps1_altitude: Latitude, longitude, and altitude.

gps1_pdop / gps1_hdop / gps1_vdop: Position, horizontal and vertical, dilutions of precision. Measures of GPS confidence.

unit1_overall-beam: a value between 0 and 255 indicating the best beam across all phased arrays. Beam 0 corresponds to the first beam of the first array (unit1_pwr1) and beam 255 corresponds to the last beam of the last phased array (unit1_pwr4). Note further that the phased arrays sweep 64 beams in azimuth and the first beam is the leftmost beam. As such, one can imagine beam 0 pointing to 45º to the left of the direction the vehicle is moving, beam 63 pointing 45º to the right of this direction, and the remaining beams pseudo-uniformly covering the angular domain clockwise.

Table 3.1 Metadata of the DeepSense 6G Dataset for Scenario 36,

PARAMETER	RANGE OF VALUES
Sample Absolute Index	2674-32466
Sequential Index	1-137
Unit-1 Overall Beam	0-255

Table 3.1 contains the range of values of all the important parameters such as sample absolute index, sequential index and overall beam for scenario 36.

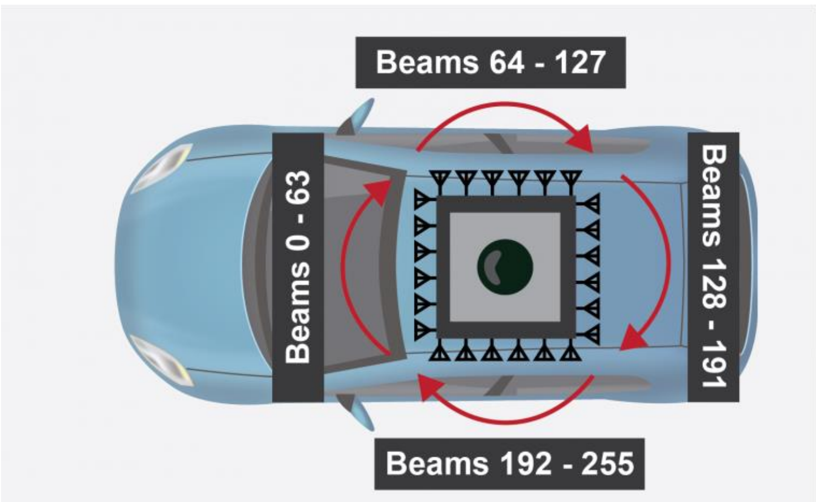

Figure 3.5 Transmitter Car Model

The first unit, designated as the receiver, is equipped with an array of sensors, including four mmWave phased arrays facing different directions, a 360-degree RGB camera, four mmWave FMCW radars, a 3D LiDAR, and a GPS RTK kit. The second unit acts as the transmitter, possessing a mmWave quasi-omnidirectional antenna that remains oriented towards the receiver and a GPS receiver for capturing real-time position data.

CHAPTER 4

DATASET PREPROCESSING

The dataset has been modified such that new parameters such as angle of arrival, velocity and distance between cars have been added. Since the latitude and longitude values are very specific we consider relative positions. Using haversine formula the distance between the cars is calculated using the latitude and longitude.

$$hav(\theta) = hav(\varphi_2 - \varphi_1) + \cos(\varphi_1)\cos(\varphi_2) hav(\lambda_2 - \lambda_1) \qquad (4.1)$$

Where,

φ1, φ2 are the latitude of point 1 and latitude of point 2,

λ1, λ2 are the longitude of point 1 and longitude of point 2.

The haversine function hav(θ), applied above to both the central angle θ and the differences in latitude and longitude, is

$$hav(\theta) = \sin^2(\frac{\theta}{2}) = \frac{1 - \cos\theta}{2} \qquad (4.2)$$

The haversine function computes half a versine of the angle θ, or the squares of half chord of the angle on a unit circle (sphere). To solve for the distance d, apply the archaversine (invers haversine) to h = hav(θ) or use the arcsine (inverse sine) function:

$$d = r.archav(h) = 2r.\arcsin\left(\sqrt{h}\right) \qquad (4.3)$$

4.1. SAMPLE CALCULATION

Convert latitude and longitude from degrees to radians:

lat1_rad = 33.35564402 * π / 180

lon1_rad = -111.9288873 * π / 180

lat2_rad = 33.35564598 * π / 180

lon2_rad = -111.9290033 * π / 180

Calculate the differences between the latitudes and longitudes:

dlat = 0.58216584 - 0.58216581 =3.420845329227262e-08

dlon = -1.95352963 - (-1.95352761) = -2.024581932320203e-06

Apply the Haversine formula:

a = sin(dlat/2)**2 + cos(lat1) * cos(lat2) * sin(dlon/2)**2

a = $\sin^2$(3.420845329227262e-08/2) + (cos(33.35564402) * cos(33.35564598) * $\sin^2$(-2.024581932320203e-06 /2))

c = 2 * asin(sqrt(a))

c = 2 * asin(sqrt(7.152305885735995e-13))

c = 1.6914261303098874e-06

Calculate the distance:

distance ≈ diameter of earth * C

distance = 6371 * 1.6914261303098874e-06

distance = 0.010776

4.2. DERIVATION OF VELOCITY

The velocity is calculated using the distance moved by the car between two samples by time, that is 500ms. 500ms since the model is going to predict the position of the cars 500ms in the future.

4.3. DERIVATION OF ANGLE OF ARRIVAL

The Angle of Arrival (AoA) refers to the direction from which a signal arrives at a receiver antenna. It's a fundamental parameter in wireless communication systems and radar systems, among other applications.

First the relative position of car2 with respect to car1 is calculated. Using that the angle of arrival is estimated.

Angle of arrival of the previous sample is also calculated and added as a column.

With the calculated AoA the rate of change of AoA is determined.

Table 4.1 Derived Metrics from GPS Data

PARAMETER	VALUE
Lat1	33.35564402
Long1	111.9288873
Lat2	33.35564598
Long2	-111.9290033
Distance	0.010776166
Velocity of Car1	0.0000384
Velocity of Car2	0.0008557
Rate of Change of AoA	-0.03130794

Table 4.1 contains calculated parameters based on latitude and longitude coordinates of the 2 cars. These parameters include the distance between these cars (Distance), the velocities of two vehicles (Velocity of Car1, Velocity of Car2), and the rate of change of the angle of arrival between the two cars (Rate of Change of AoA).

	dist_bw_c	car1_dista	car2_dista	aoa_estim	car1_veloc	car2_veloc	aoa_estim	rate_aoa_change
0	0.031154	0.001917	0.001901	-2.38219	0.003834	0.003801	-2.38046	-0.00346
1	0.030657	0.001322	0.001891	0.980609	0.002644	0.003782	1.03935	-0.11748
2	0.030704	0.001383	0.00189	0.970009	0.002765	0.00378	1.024392	-0.10877
3	0.031756	0.002166	0.001865	0.966288	0.004331	0.003731	1.019641	-0.10671
4	0.03191	0.002219	0.001838	0.957396	0.004438	0.003676	1.0126	-0.11041
5	0.031343	0.00155	0.001817	0.951109	0.003101	0.003634	1.005383	-0.10855
6	0.031405	0.00155	0.001792	0.951427	0.003101	0.003583	1.001282	-0.09971
7	0.031545	0.001605	0.001768	0.943528	0.003211	0.003535	0.991619	-0.09618
8	0.032825	0.002491	0.001746	0.93679	0.004981	0.003493	0.983539	-0.0935
9	0.032939	0.002539	0.001734	0.925168	0.005079	0.003469	0.973723	-0.09711
10	0.032215	0.001758	0.00174	0.919274	0.003515	0.00348	0.966085	-0.09362
11	0.032308	0.001758	0.001738	0.917561	0.003515	0.003477	0.959707	-0.08429
12	0.032469	0.001775	0.001713	0.910076	0.00355	0.003427	0.951312	-0.08247
13	0.032608	0.001793	0.001696	0.901871	0.003586	0.003391	0.942262	-0.08078
14	0.033916	0.002687	0.001701	0.894101	0.005373	0.003402	0.933461	-0.07872
15	0.033956	0.002684	0.001719	0.884419	0.005368	0.003438	0.924808	-0.08078
16	0.033116	0.001787	0.00173	0.87892	0.003575	0.00346	0.918012	-0.07818
17	0.033157	0.001788	0.001734	0.875984	0.003575	0.003468	0.911477	-0.07099
18	0.034454	0.002681	0.001721	0.871371	0.005363	0.003442	0.905526	-0.06831
19	0.034545	0.002681	0.001692	0.864545	0.005363	0.003384	0.89874	-0.06839
20	0.033727	0.001787	0.00169	0.857405	0.003575	0.00338	0.889867	-0.06492
21	0.033761	0.001787	0.001716	0.84874	0.003575	0.003431	0.878317	-0.05915
22	0.033761	0.001781	0.001738	0.843389	0.003562	0.003475	0.872459	-0.05814
23	0.034999	0.002661	0.001737	0.84042	0.005323	0.003474	0.868435	-0.05603
24	0.034997	0.002636	0.001715	0.8369	0.005272	0.00343	0.86405	-0.0543
25	0.034132	0.001724	0.001693	0.833255	0.003448	0.003385	0.858165	-0.04982
26	0.034157	0.001724	0.0017	0.828784	0.003448	0.0034	0.851008	-0.04445
27	0.034118	0.001687	0.001727	0.824844	0.003373	0.003454	0.846633	-0.04358

Figure 4.1 Modified Dataset with derived parameters

Figure 4.1 contains a section of the dataset that was used in training the ML model which has additional parameters such as velocity and rate of change of Angle of arrival which is derived from the GPS data present in the original dataset

CHAPTER 5

PROPOSED METHODOLOGY

The proposed methodology involves training different machine learning models such as ANN, KNN, SVM, Random Forest, XGBoost, and Decision Tree using the modified dataset which has additional features such as velocity, acceleration, and the angle of arrival between two cars. These models are trained to predict the optimal beam pair for V2V communication which has the lowest average power loss. By leveraging these diverse models and incorporating additional features, the methodology aims to enhance the accuracy and efficiency of beam prediction in V2V communication systems, ultimately improving safety and performance on the roads.

5.1. ANN MODEL

Artificial Neural Networks (ANNs) are computational models inspired by the structure and functioning of the human brain's neural networks. They consist of interconnected nodes organized into layers. Information flows from the input layer through one or more hidden layers to the output layer, with each layer performing complex transformations on the input data. ANNs are commonly used for various tasks such as classification, regression, pattern recognition, and more, due to their ability to learn complex patterns and relationships within data. Training an ANN involves adjusting the connections (weights) between nodes to minimize the difference between the model's predictions and the actual outputs. This process, often done using optimization algorithms like gradient descent, enables ANNs to adapt and improve their performance over time, making them powerful tools in machine learning and artificial intelligence applications.

ANN MODEL SETUP

The ANN model used follows a sequential architecture, comprising 3 densely connected layers.

The input layer consists of 1024 units with Rectified Linear Unit (ReLU) activation function, accepting input data of shape 6.

Subsequent hidden layers include:

1.A layer with 768 units, activated by ReLU.

2.Another layer with 512 units, also activated by ReLU.

3.A final hidden layer with 256 units, utilizing ReLU activation.

The output layer consists of 256 units, employing the softmax activation function to output probabilities for each class.

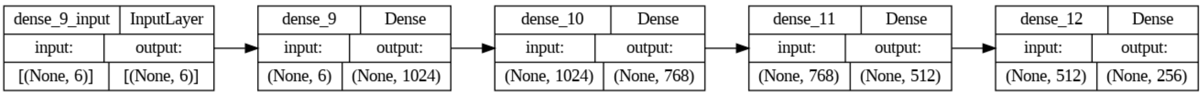

Figure 5.1 ANN Model Architecture

Figure 5.1 contains the architecture for the proposed ANN model which contains only the layers of the model, None represents a dimension that is not fixed or determined until the model is run with real time data.

5.2. KNN MODEL

The k-nearest neighbors (KNN) algorithm is a non-parametric, supervised learning classifier, which uses proximity to make classifications or predictions about the grouping of an individual data point.

KNN MODEL SETUP

The model setup involves several steps to prepare the data, instantiate the KNN classifier, train the model, and evaluate its performance:

Data Preparation: Data from multiple scenarios are loaded, concatenated, and split into training and testing sets using the train_test_split function from sklearn.model_selection.

Classifier Instantiation: The KNN classifier is instantiated using KNeighborsClassifier from sklearn.neighbors. Here, n_neighbors=5 is chosen, meaning that the class label or predicted value for a data point will be determined based on the majority class or average value among its five nearest neighbors.

Model Training: The KNN classifier is trained on the standardized training data using the fit method.

Model Evaluation: After training, the model is evaluated on the testing data to assess its performance. This is done by making predictions on the testing data using the predict method.

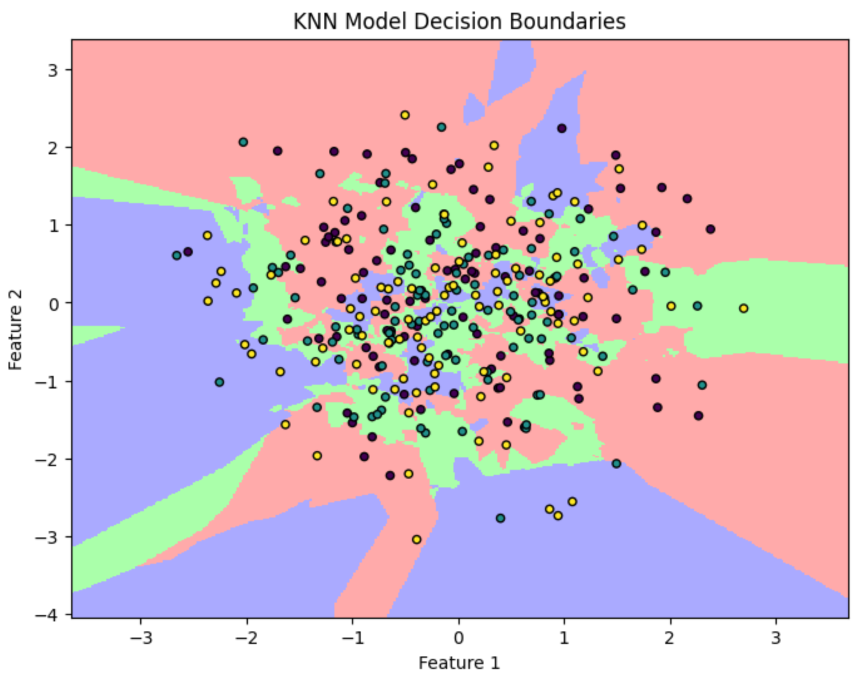

Figure 5.2 KNN Model Decision Boundaries

Figure 5.2 represents the decision boundaries of the KNN Model which are the lines, curves, or surfaces that separate different classes in the feature space.

Here feature 1 and feature 2 are input features which are used to predict the target variable that is the optimal beam index.

5.3. SVM MODEL

A Support Vector Machine (SVM) is defined as a machine learning algorithm that uses supervised learning models to solve complex classification, regression, and outlier detection problems by performing optimal data transformations that determine boundaries between data points based on predefined classes, labels, or outputs.

The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.

The hyperplane tries that the margin between the closest points of different classes should be as maximum as possible. The dimension of the hyperplane depends upon the number of features.

SVM MODEL SETUP

The model setup for Support Vector Machine (SVM) involves the following steps:

Data Preparation: The dataset is loaded and split into training and testing sets using the train_test_split function from sklearn.model_selection.

Classifier Instantiation: The SVM classifier is instantiated using the SVC class from sklearn.svm. By default, the SVM classifier uses a Radial Basis Function (RBF) kernel.

Model Training: The instantiated SVM classifier is trained on the standardized training data using the fit method. During training, the classifier learns the optimal hyperplane that best separates the classes in the feature space.

Model Evaluation: After training, the performance of the SVM classifier is evaluated on the testing data.

5.4. RANDOM FOREST MODEL

Random Forest is a powerful ensemble learning technique that enhances predictive accuracy by aggregating the outputs of multiple decision trees.

It employs bootstrap sampling to create diverse subsets of the training data and introduces randomness by considering only a random subset of features at each split in each tree.

Through recursive construction, decision trees are built to maximize predictive performance, selecting the best features for splitting at each node.

The final prediction is then determined by averaging (for regression) or voting (for classification) across all individual trees. Random Forest is prized for its robustness against overfitting and its ability to handle missing data.

RANDOM FOREST MODEL SETUP

The setup of a Random Forest model involves several key steps:

Data Preparation: The dataset is preprocessed, including handling missing values, encoding categorical variables if necessary, and splitting the data into training and testing sets using the train_test_split function from sklearn.model_selection.

Classifier Instantiation: The Random Forest classifier is instantiated using RandomForestClassifier from sklearn.ensemble. Parameters such as the number of trees in the forest (n_estimators) and the maximum depth of the trees (max_depth) can be specified

Model Training: The Random Forest classifier is trained on the training data using the fit method. During training, multiple decision trees are constructed based on bootstrap samples of the training data and feature randomness.

Model Evaluation: Once trained, the model is evaluated on the testing data to assess its performance.

Table 5.1 Parameter Values for Random Forest Model

PARAMETER	VALUE
Number of Estimators	50
criterion	'entropy'
random_state	0
verbose	22
Oob_score	True
Class Weight	Balanced Subsample

The table 5.1 provides information about the parameters and settings used in this Random Forest model. It comprises 50 decision trees, employing the entropy criterion to measure split quality. The random state is set to 0 for reproducibility, with a verbosity level of 22 during fitting. Out-of-bag scoring is enabled to estimate generalization accuracy, and class weights are balanced across subsamples.

5.5. XGBOOST MODEL

XGBoost stands as a leading machine learning algorithm renowned for its exceptional performance in handling structured and tabular data across various predictive modeling tasks. With a plethora of tunable hyperparameters, XGBoost facilitates meticulous model optimization, empowering users to achieve optimal performance across a wide array of machine learning applications including classification and regression algorithms.

XGBOOST MODEL SETUP

Setting up an XGBoost model involves the following key steps:

Data Preparation: The dataset is prepared, including handling missing values, encoding categorical variables if necessary, and splitting the data into training and testing sets using sklearn.model_selection's train_test_split function.

Model Configuration: The XGBoost model is configured by specifying hyperparameters such as the learning rate, maximum depth of trees, number of boosting rounds, and regularization parameters.

Model Training: The XGBoost model is trained on the training data using the fit method from the XGBoost library. During training, an ensemble of decision trees is built sequentially, with each subsequent tree correcting the errors of the previous ones.

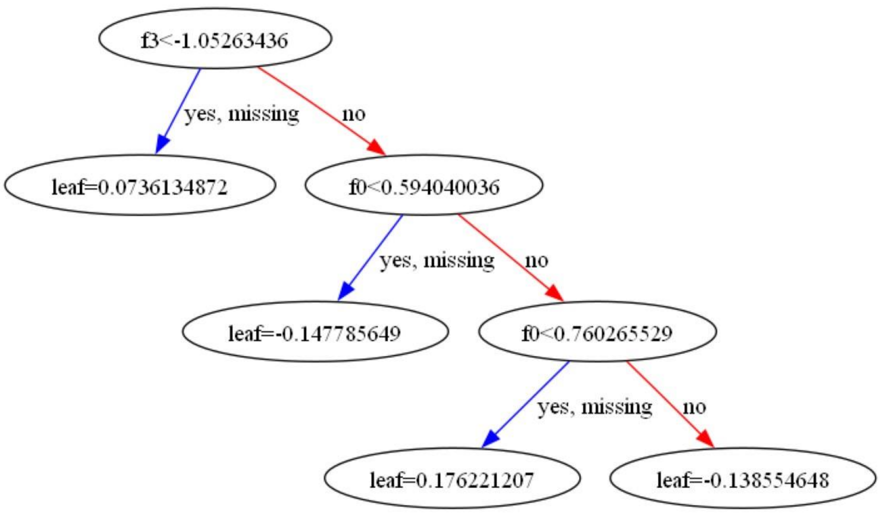

Figure 5.3 Visualization of Single Decision Tree in XGboost

The Figure 5.3 represents a decision tree from an XGBoost model, illustrating the sequence of decision rules used for prediction. Each node in the tree represents a decision based on a specific feature value, guiding the traversal down the tree branches. At each decision node, a condition is evaluated to determine the next path, and at the leaf nodes, the final prediction value is provided.

5.6. DECISION TREE MODEL

The decision tree model is a versatile and interpretable machine learning algorithm used for both classification and regression tasks.

It operates by recursively partitioning the dataset into subsets based on feature values, forming a tree-like structure where each internal node represents a feature and a split point, while each leaf node represents a predicted class label or value. Decision trees also provide valuable insight into feature importance, aiding in feature selection and interpretation.

DECISION TREE MODEL SETUP

Setting up a decision tree model involves the following key steps:

Data Preparation: Begin by preparing the dataset for training the decision tree model. This involves tasks such as handling missing values, encoding categorical variables if necessary, and splitting the data into training and testing sets using the `train_test_split` function from `sklearn.model_selection`.

Model Instantiation: Next, instantiate the decision tree classifier using the `DecisionTreeClassifier` class from `sklearn.tree`. You can specify hyperparameters such as the criterion for splitting, the maximum depth of the tree, minimum samples required to split a node, etc.

Model Training: Train the decision tree classifier on the training data using the `fit` method. During training, the decision tree algorithm recursively splits the data based on the selected criterion to minimize impurity or maximize information gain.

Model Evaluation: After training, evaluate the performance of the decision tree model on the testing data to assess its accuracy and generalization ability.

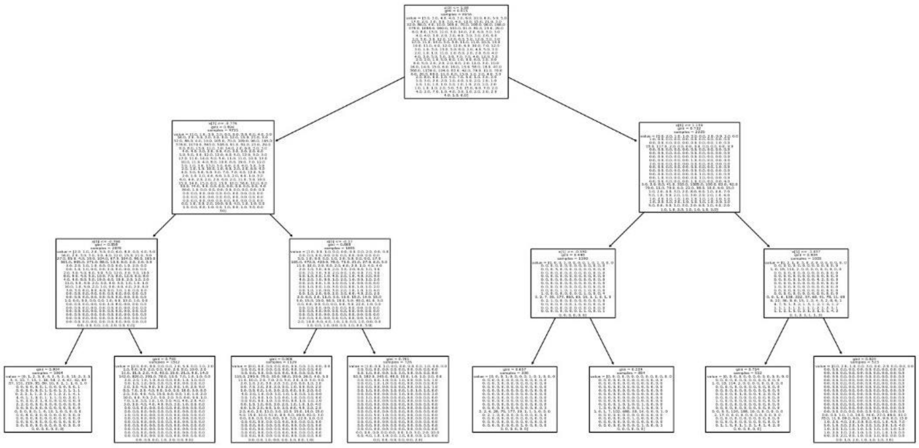

Figure 5.4 Visualization Of Nodes In Decision Tree Model

Figure 5.4 contains the nodes used to represent the decision tree model with a max depth limited to 3. Each node represents a split based on a feature and threshold value, with the Gini impurity indicating the node's homogeneity. The number of samples at each node and the class distribution illustrate how the dataset is partitioned. Visualizing the decision tree with a depth of 3 offers a concise overview of the classification process, showing key splits and decision points in the data.

CHAPTER 6
RESULTS AND DISCUSSION

6.1. BEAM PREDICTION USING ANN MODEL

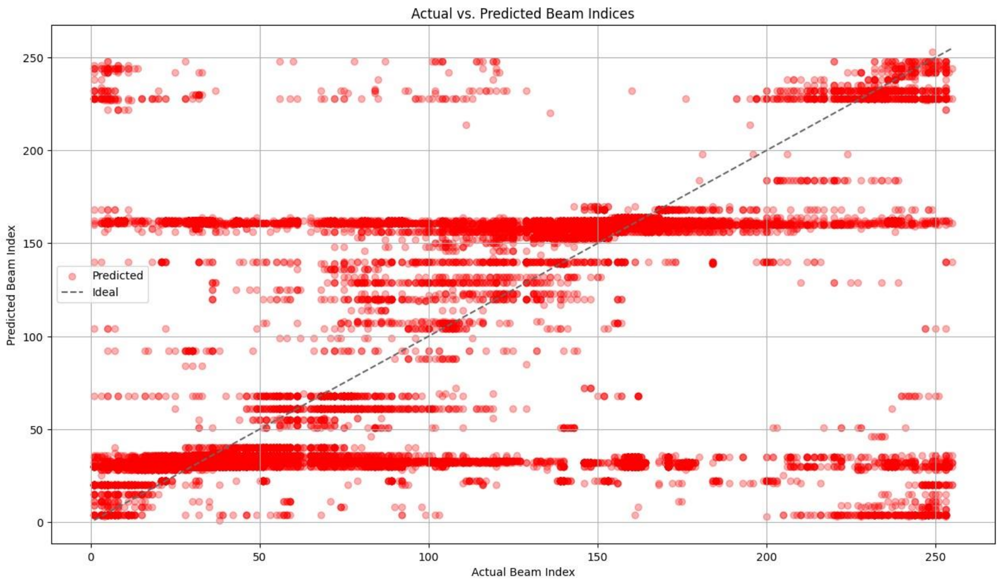

Figure 6.1 Scatter Plot Of Actual Vs Predicted Values For ANN Model

The scatter plot presented in Figure 6.1 showcases the performance evaluation of the ANN model, contrasting actual beam indices with their corresponding predicted values. Here, the red data points signify the model's predictions, while the dotted black line represents the true indices. As observed, there are noticeable deviations between actual and predicted values, particularly where points diverge significantly. These deviations may be attributed to various factors, including an imbalanced dataset where certain scenarios are underrepresented, leading to limitations in the model's predictive accuracy.

6.2. BEAM PREDICTION USING SVM MODEL

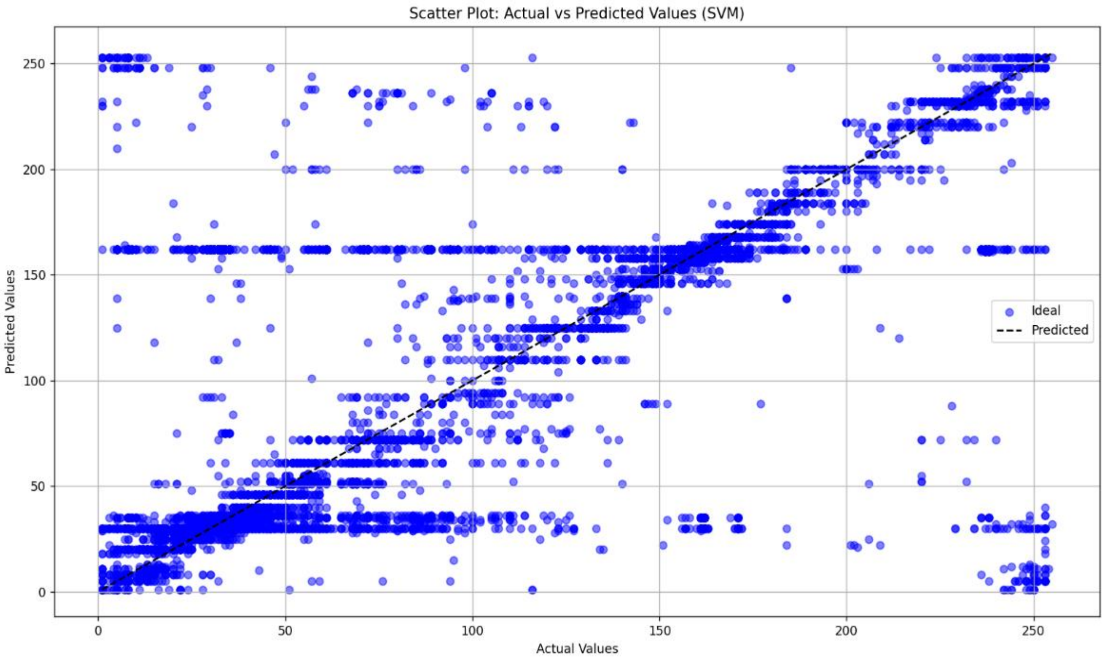

Figure 6.2 Scatter Plot of Actual Vs Predicted Values for SVM Model

The scatter plot depicted in Figure 6.2 illustrates the performance of the SVM model by showing actual and predicted beam indices. The blue data points correspond to the predicted values produced by the model, while the dotted black line represents the actual values. This visual comparison enables an assessment of the model's accuracy. However, discrepancies between actual and predicted values are observed, particularly in instances where points are significantly distant. These deviations may be attributed to an unbalanced dataset, where certain scenarios are underrepresented, leading to limitations in the model's predictive capabilities. Additionally, challenges arise in predicting beam indices during complex moving of the cars such as turning at junctions.

6.3. BEAM PREDICTION USING KNN MODEL

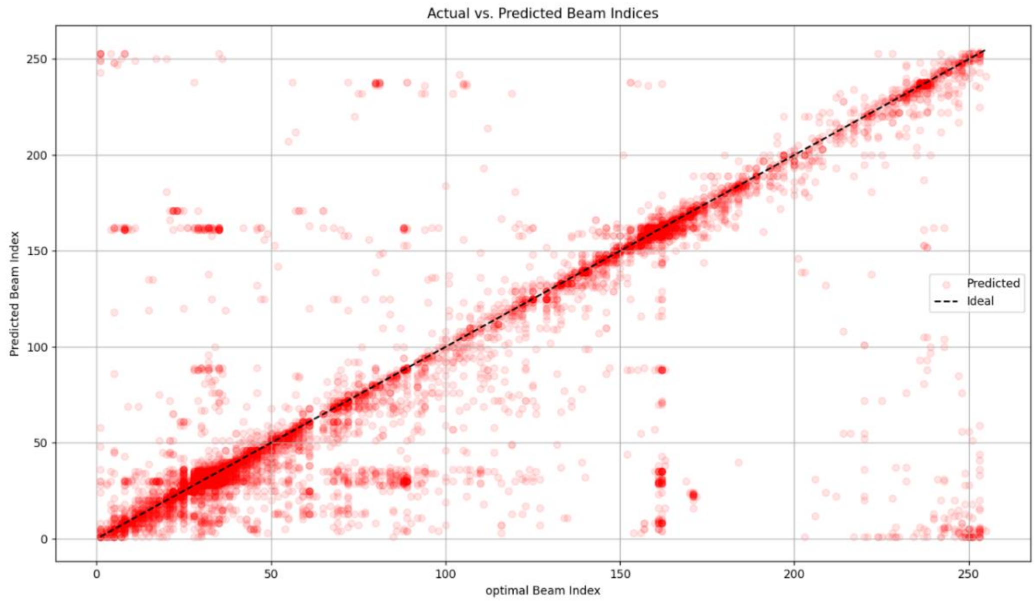

Figure 6.3 Scatter Plot of Actual Vs Predicted Values for KNN Model

The Scatter plot in figure 6.3 illustrates a comparison between the actual and predicted beam indices from a KNN model, which achieves higher accuracy compared to SVM and the ANN Models. Here, the red points represent the predicted values generated by the KNN model, while the dotted black line represents the actual values. As given by table 6.1 below this model gives a higher accuracy compared to KNN model. The top-k classification accuracy is one of the core metrics in machine learning. Here, k is conventionally a positive integer, such as 1 or 5, leading to top-1 or top-5 training objectives. This metric computes the number of times where the correct label is among the top k labels predicted (ranked by predicted scores).

TABLE 6.1 Comparison of Top-K Accuracy for ANN, KNN and SVM

MODEL	**Top-1**	**Top-3**	**Top-5**
ANN MODEL	17.21%	36.12%	52.01%
K-NEAREST NEIGHBORS	33.73%	58.96%	68.95%

SUPPORT VECTOR MACHINE	23.29%	52.8%	62.42%

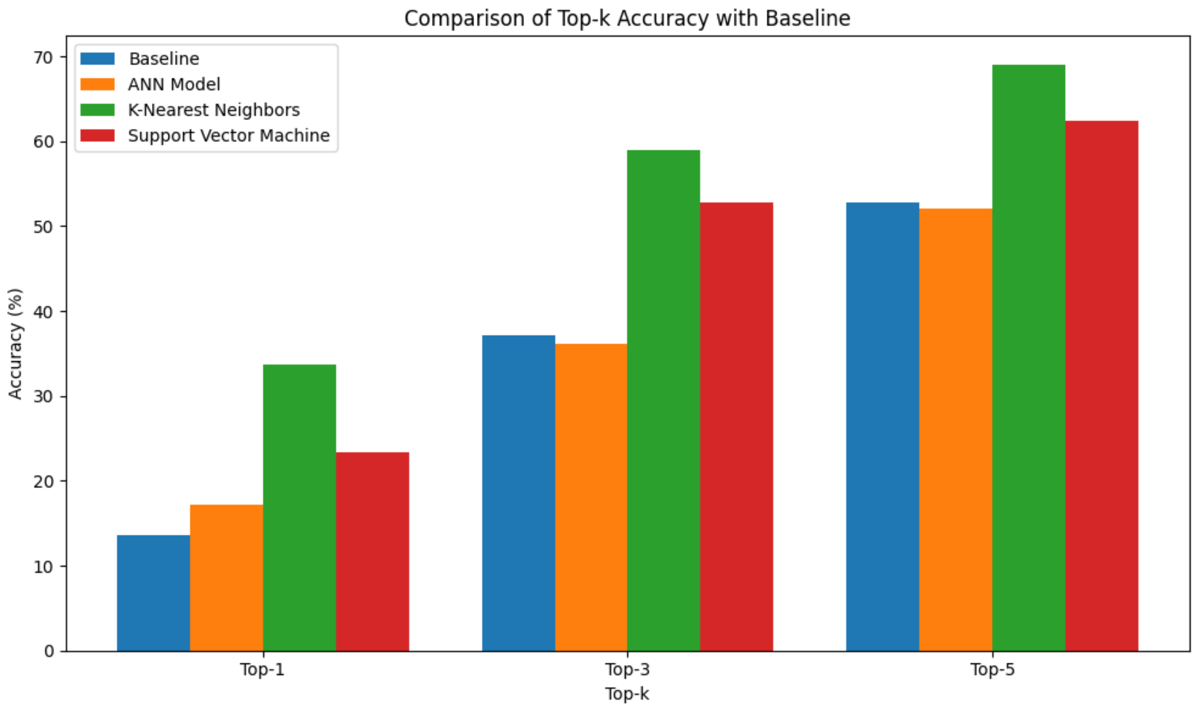

Figure 6.4 Top-K Accuracies for Different ML Models Against Baseline Model

As shown 6.4 in the bar graph in figure the K-Nearest Neighbors (KNN) model consistently demonstrates superior performance compared to the Artificial Neural Network (ANN) and Support Vector Machine (SVM) models. With higher accuracy in predicting the optimal beam index within the top 1, top 3, and top 5 guesses, the KNN algorithm establishes itself as the most reliable choice for all scenarios combined. This unified outcome underscores the effectiveness of the KNN model across varied scenarios, reaffirming its efficacy in beam index prediction tasks.

6.4. BEAM PREDICTION USING RANDOM FOREST

Random Forest is a powerful ensemble learning technique that enhances predictive accuracy by aggregating the outputs of multiple decision trees.

It employs bootstrap sampling to create diverse subsets of the training data and introduces randomness by considering only a random subset of features at each split in each tree.

Through recursive construction, decision trees are built to maximize predictive performance, selecting the best features for splitting at each node.

The final prediction is then determined by averaging (for regression) or voting (for classification) across all individual trees. Random Forest is prized for its robustness against overfitting and its ability to handle missing data.

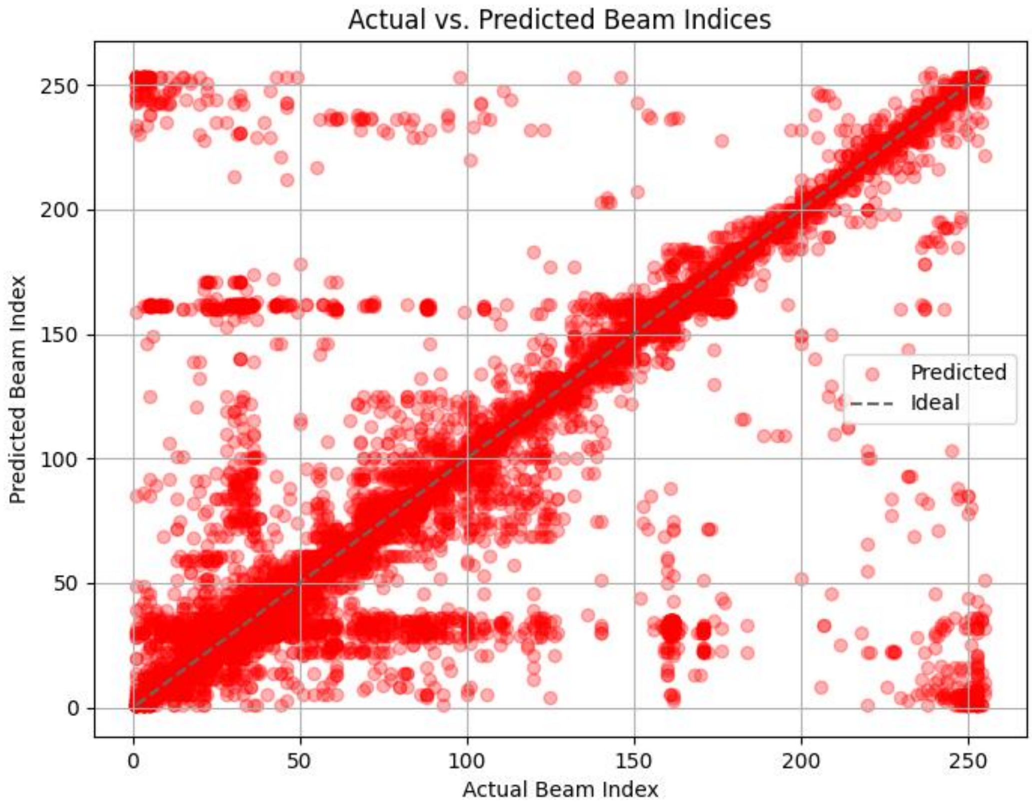

Figure 6.5 Scatter Plot of Actual Vs Predicted Values For Random Forest

The figure 6.5 represents the scatter plot for the predicted vs the actual beam indices for the random forest model. As shown this model attains higher accuracy compared to the previous model and this model performs optimally in the case where the data is unbalanced. Table 6.2 represents the top-k accuracy and average power loss for each scenario and then all the scenarios combined.

TABLE 6.2 Random Forest Top-K Comparision for All Scenarios

Scenario	TOP 1	TOP 3	TOP 5	Avg. power loss
36	58.83	84.68	88.49	-0.77
37	55.38	88.65	93.62	-0.45
38	43.26	73.65	82.42	-1.78
39	49.16	85.99	93.53	-0.67
Combined (36,37,38,39)	46.93	77.27	85.37	-1.45

6.5. BEAM PREDICTION USING XGBOOST MODEL

XGBoost stands as a leading machine learning algorithm renowned for its exceptional performance in handling structured and tabular data across various predictive modelling tasks.

XGBoost employs a gradient boosting framework, where decision trees are sequentially constructed to iteratively correct the errors of previous trees. This approach ensures superior predictive accuracy while mitigating overfitting through the integration of regularization techniques such as L1 and L2 regularization.

With a plethora of tuneable hyperparameters, XGBoost facilitates meticulous model optimization, empowering users to achieve optimal performance across a wide array of machine learning applications.

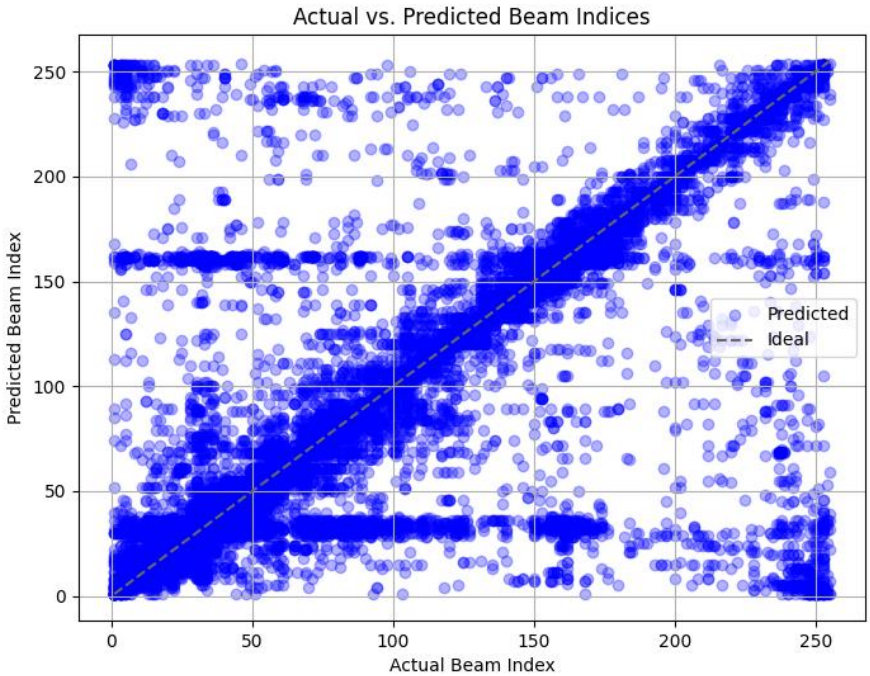

Figure 6.6 XGBOOST MODEL SCATTER PLOT

The figure 6.6 represents the scatter plot for the predicted vs the actual beam indices for the XGBoost model. As shown this model attains higher accuracy compared to the ANN, SVM and KNN model but lower compared to Random Forest Model. Table 6.3 represents the top-k accuracy and average power loss for each scenario and then all the scenarios combined.

Table 6.3 XGBoost Top-K Comparision for All Scenarios

Scenario	TOP1	TOP3	TOP5	Avg. power loss
36	48.21	75.67	80.98	-2.24
37	43.6	77.76	85.34	-1.69
38	34.95	62.62	72.46	-3.24
39	43.68	80.92	90.69	-1.13
combined (36,37,38,39)	33.54	62.5	74.77	-2.81

In the combined analysis of scenarios 36, 37, 38, and 39, the top-performing model achieves an average top-1 accuracy of 46.93%, top-3 accuracy of 77.27%, and top-5 accuracy of 85.37%. Additionally, the average power loss across all scenarios is recorded at -1.45 dB. These results indicate the effectiveness of the models in predicting the optimal beam index, with varying levels of accuracy and power loss across different scenarios.

6.6. BEAM PREDICTION USING DECISION TREE

The decision tree model is a versatile and interpretable machine learning algorithm used for both classification and regression tasks.

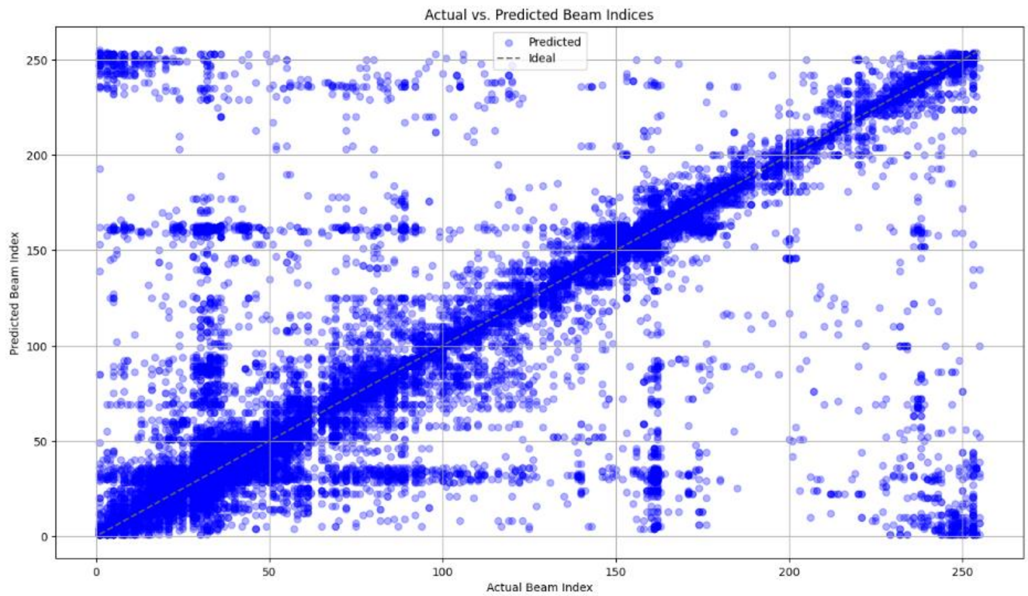

Figure 6.7 Decision Tree Model Scatter Plot

The figure 6.7 represents the scatter plot for the predicted vs the actual beam indices for the Decision Tree model. This model attains similar accuracy XGBoost Model and the points which are deviated away from the from the ideal straight line occurs when the car experiences turning at roads which are unpredictable at times. Table 6.4 represents the top-k accuracy and average power loss for each scenario and then all the scenarios combined for decision tree model.

Table 6.4 Decision Tree Top-K Comparison For All Scenarios

Scenario	TOP1	TOP3	TOP5	Avg. power loss
36	50.17	79.86	85.18	-1.19
37	45.75	80.07	87.78	-1.10
38	30.57	57.55	67.25	-4.13
39	39.75	77.16	87.87	-1.46
Combined (36,37,38,39)	35.27	64.52	74.9	-2.82

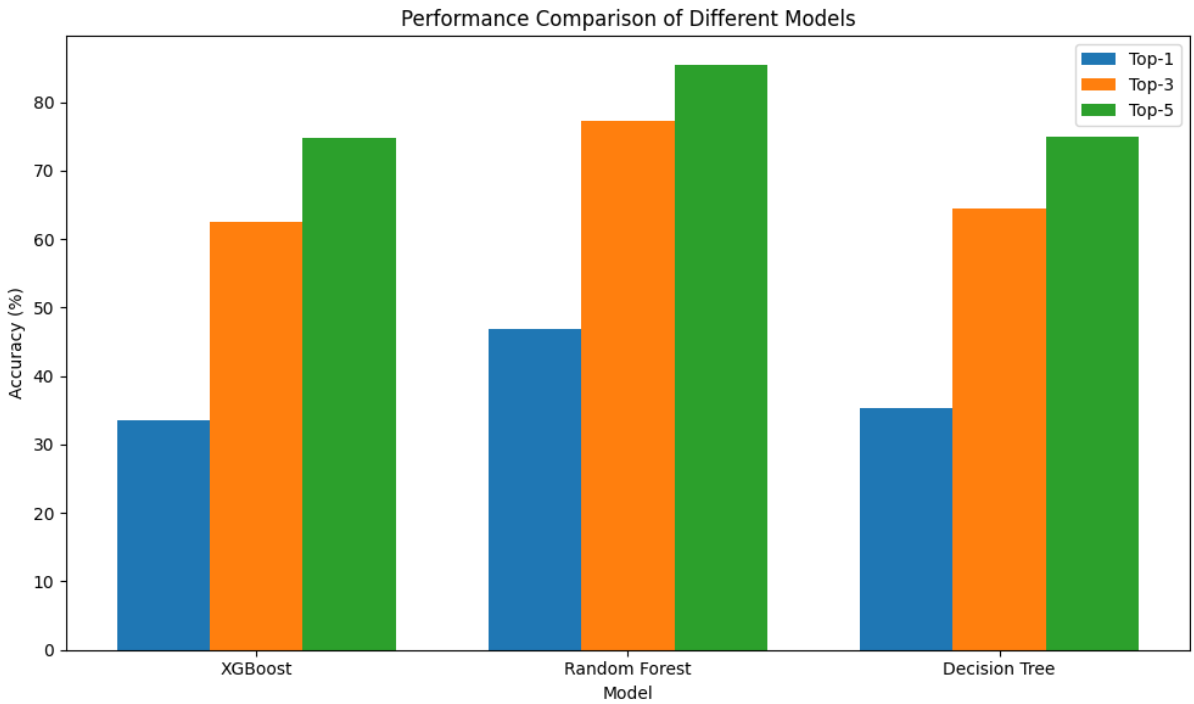

Figure 6.8 Top-K Accuracy For Top Performing Ml Models

The bar graph in figure 6.8 illustrates the performance of three models – Random Forest, XGBoost, and Decision Tree – in predicting the optimal beam index. Among these models, Random Forest exhibits the highest accuracy. This outcome is attributed to Random Forest's robustness in handling data imbalance, a common challenge in this predictive task. Despite variations in model complexity and optimization techniques, Random Forest consistently outperforms XGBoost and Decision Tree in accurately identifying the optimal beam index. This highlights the importance of employing techniques that mitigate the effects of data imbalance when developing predictive models for similar tasks.

6.7 CONTRIBUTION OF EACH FEATURE ON MODEL PERFORMACE

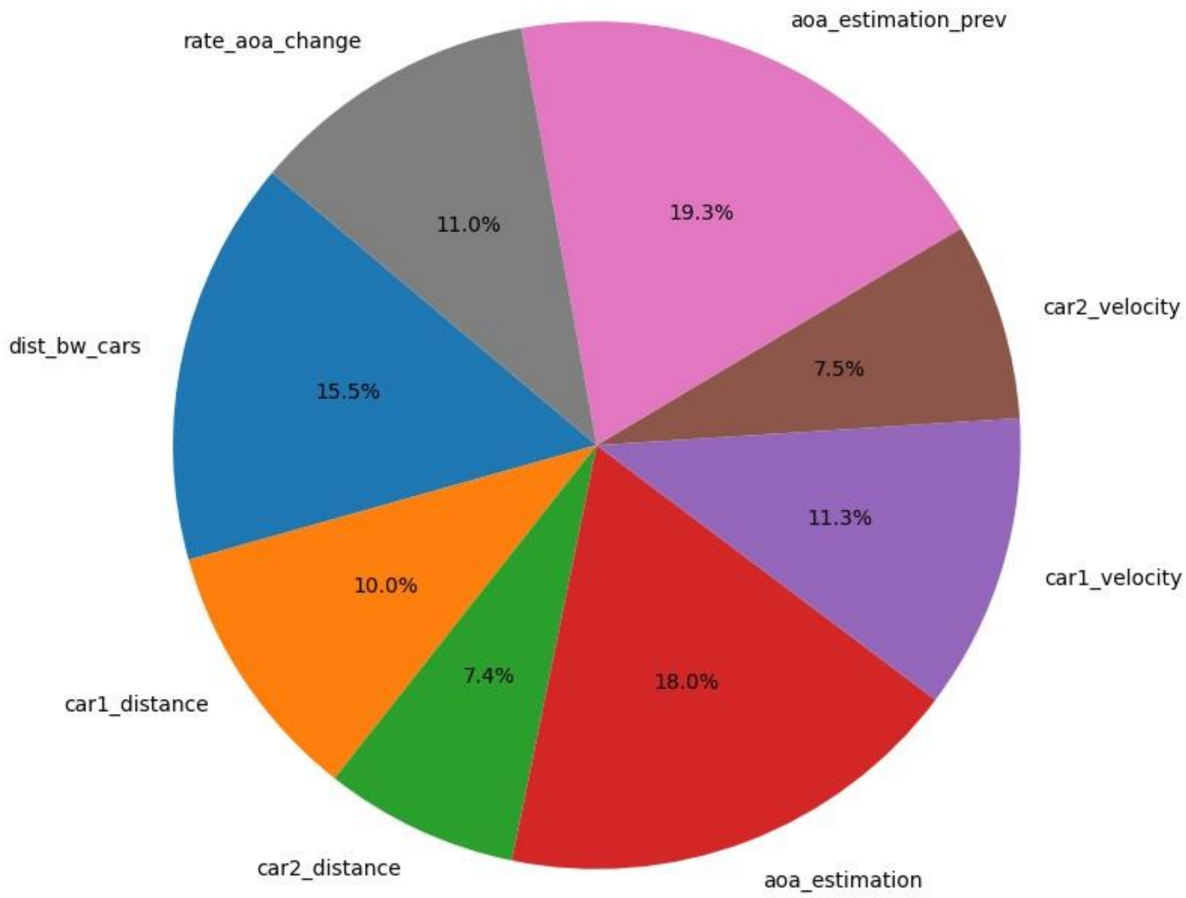

FIGURE 6.9 Contribution of Each Feature Towards Accuracy

Figure 6.9 represents the contribution of each feature that has been derived from the original dataset towards the model performance. Each of the parameters are explained below

- dist_bw_cars: This feature represents the distance between two cars. It could be a crucial factor in determining the optimal beam index, as the relative position of cars can affect the quality of communication signals.

- car1_distance: This feature represent the distance of the first car from its previous position in the last time sample. This is a crucial parameter in determining the velocity of the car
- car2_distance: This feature represent the distance of the second car from its previous position in the last time sample. This is a crucial parameter in determining the velocity of the car
- aoa_estimation: This feature stands for the estimation of the Angle of Arrival (AoA) of a signal which is the angle between the transmitter and the receiver car. AoA estimation is vital in beamforming, where antennas are directed towards the source of incoming signals for better reception.
- car1_velocity: This feature is the velocity or speed of the first car. Car velocity can influence the propagation characteristics of wireless signals, affecting signal strength and reception quality.
- car2_velocity: Similar to car1_velocity, this feature represents the velocity or speed of the second car. The velocity of both cars plays a role in determining the dynamics of the communication environment.
- aoa_estimation_prev: This feature gives the previous estimation of the Angle of Arrival (AoA). Historical AoA data could provide insights into signal patterns and variations over time.
- rate_aoa_change: This feature indicates the rate of change in the Angle of Arrival (AoA) over time. Rapid changes in AoA could signify dynamic movement or environmental conditions affecting signal propagation.

CHAPTER 7

CONCLUSION AND FUTURE WORKS

This project is focused on developing an efficient approach for predicting optimal beam configurations in Vehicle-to-Vehicle (V2V) communication systems, crucial for enhancing road safety and efficiency. Leveraging an existing dataset comprising GPS coordinates and corresponding optimal beams with minimal power loss, we trained six machine learning models—ANN, SVM, KNN, Random Forest, XGBoost, and Decision Tree. The models were trained using features such as inter-vehicle distance and angle of arrival which are extracted from the original dataset and the performance evaluation metric used was the top-k metric. Our analysis indicates Random Forest as the top-performing model, achieving a Top-5 accuracy of 85.37% and an average power loss of -1.45dB. The model that performs the second best is the Decision Tree model, exhibiting a Top-5 accuracy of 74.9% with an average power loss of -2.82dB. These different models have been tested in different scenarios such as interstate and intercity scenarios where the distance travelled and speed of the cars vary. Thus, Random Forest stands out for its higher accuracy and power loss reduction, while the Decision Tree model is a good alternative with lower computational complexity. Thus, by leveraging machine learning—specifically Random Forest—we efficiently forecast optimal beams, mitigating training overhead and latency associated with traditional methods.

REFERENCES

[1] Muhammad Baqer Mollah, Honggang Wang, and Hua Fang, "mmWave enabled connected autonomous vehicles: A use case with V2V cooperative perception," IEEE Network, pp. 1–7, 2023.

[2] G. Ciaramitaro, M. Brambilla, M. Nicoli, and U. Spagnolini, "Signalling design in sensor-assisted mmWave communications for cooperative driving," IEEE Open Journal of Intelligent Transportation Systems, 2023.

[3] H. Ngo, H. Fang, and H. Wang, "Cooperative perception with V2V communication for autonomous vehicles," IEEE Transactions on Vehicular Technology, 2023.

[4] "Propagation models and performance evaluation for 5G millimeterwave bands," IEEE Transactions on Vehicular Technology, vol. 67, no. 9, pp. 8422–8439, 2018.

[5] P. Kyöst, P. Zhang, A. Pärssinen, K. Haneda, P. Koivumäki, and W. Fan, "On the feasibility of out-of-band spatial channel information for millimeter-wave beam search," IEEE Transactions on Antennas and Propagation, 2023.

[6] Muhammad Alrabeiah and Ahmed Alkhateeb, "Deep learning for mmwave beam and blockage prediction using sub-6 GHz channels," IEEE Transactions on Communications, vol. 68, no. 9, pp. 5504–5518, 2020.

[7] A. Graff, Y. Chen, N. González-Prelcic, and T. Shimizu, "Deep learning-based link configuration for radar-aided multiuser mmwave vehicle-to-infrastructure communication," IEEE Transactions on Vehicular Technology, 2023.

[8] H. Luo, U. Demirhan, and A. Alkhateeb, "Millimeter wave V2V beam tracking using radar: Algorithms and real-world demonstration," arXiv preprint arXiv:2308.01558, 2023.

[9] U. Demirhan and A. Alkhateeb, "Radar aided 6G beam prediction: Deep learning algorithms and real-world demonstration," in 2022 IEEE Wireless Communications and Networking Conference (WCNC). IEEE, 2022, pp. 2655–2660.

[10] J. Gu, L. Collins, D. Roy, A. Mokhtari, S. Shakkottai, and K. R. Chowdhury, "Meta-learning for image-guided millimeter-wave beam selection in unseen environments," in ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2023, pp. 1–5.

[11] J. A. Zhang, F. Liu, C. Masouros, R. W. Heath, Z. Feng, L. Zheng, and A. Petropulu, “An overview of signal processing techniques for joint communication and radar sensing,” IEEE Journal of Selected Topics in Signal Processing, vol. 15, no. 6, pp. 1295–1315, 2021.

[12] F. Liu, Y. Cui, C. Masouros, J. Xu, T. X. Han, Y. C. Eldar, and S. Buzzi, “Integrated sensing and communications: Toward dual-functional wireless networks for 6g and beyond,” IEEE Journal on Selected Areas in Communications, vol. 40, no. 6, pp. 1728–1767, 2022.

[13] A. Liu, Z. Huang, M. Li, Y. Wan, W. Li, T. X. Han, C. Liu, R. Du, D. K. P. Tan, J. Lu, Y. Shen, F. Colone, and K. Chetty, “A survey on fundamental limits of integrated sensing and communication,” IEEE Communications Surveys and Tutorials, vol. 24, no. 2, pp. 994–1034, 2022.

[14] G. Charan, M. Alrabeiah, and A. Alkhateeb, “Vision-aided 6g wireless communications: Blockage prediction and proactive handoff,” IEEE Transactions on Vehicular Technology, vol. 70, no. 10, pp. 1019310208, 2021.

[15] M. Zecchin, M. B. Mashhadi, M. Jankowski, D. Gündüz, M. Kountouris, and D. Gesbert, “Lidar and position-aided mmwave beam

selection with non-local cnns and curriculum training," IEEE Transactions on Vehicular Technology, vol. 71, no. 3, pp. 2979–2990, 2022.

[16] G. Charan, U. Demirhan, J. Morais, A. Behboodi, H. Pezeshki, and A. Alkhateeb, "Multi-modal beam prediction challenge 2022: Towards generalization," 2022. [Online]. Available: https://arxiv.org/abs/2209. 07519

[17] U. Demirhan and A. Alkhateeb, "Radar aided 6g beam prediction: Deep learning algorithms and real-world demonstration," in 2022 IEEE Wireless Communications and Networking Conference (WCNC), 2022, pp. 2655–2660.

[18] S. Jiang, G. Charan, and A. Alkhateeb, "Lidar aided future beam prediction in real-world millimeter wave v2i communications," IEEE Wireless Communications Letters, pp. 1–1, 2022.

[19] A. Ali, N. González-Prelcic, and A. Ghosh, "Passive radar at the roadside unit to configure millimeter wave vehicle-to-infrastructure links," IEEE Trans. Veh. Technol., vol. 69, no. 12, pp. 14903–14917, 2020.

[20] H. Kim, K. Granström, L. Gao, G. Battistelli, S. Kim, and H. Wymeersch, "5g mmwave cooperative positioning and mapping using multimodel phd filter and map fusion," IEEE Transactions on Wireless Communications, vol. 19, no. 6, pp. 3782–3795, 2020.

[21] U. Demirhan and A. Alkhateeb, "Integrated sensing and communication for 6G: Ten key machine learning roles," 2022. [Online]. Available: https://arxiv.org/abs/2208.02157

[22] A. Alkhateeb, "DeepMIMO: A generic deep learning dataset for millimeter wave and massive MIMO applications," in Proc. of Information Theory and Applications Workshop (ITA), San Diego, CA, Feb 2019, pp. 1–8.