BAŞKENT UNIVERSITY
INSTITUTE OF SCIENCE AND ENGINEERING
DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING
DOCTOR OF PHILOSOPHY IN ELECTRICAL AND ELECTRONICS ENGINEERING

GRAPH-BASED OBJECT CLASSIFICATION TECHNIQUES FOR AUTONOMOUS VEHICLE RADAR SENSORS

BY

RASİM AKIN SEVİMLİ

DOCTOR OF PHILOSOPHY THESIS

ANKARA – 2023

BAŞKENT UNIVERSITY
INSTITUTE OF SCIENCE AND ENGINEERING
DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING
DOCTOR OF PHILOSOPHY IN ELECTRICAL AND ELECTRONICS ENGINEERING

GRAPH-BASED OBJECT CLASSIFICATION TECHNIQUES FOR AUTONOMOUS VEHICLE RADAR SENSORS

BY

RASİM AKIN SEVİMLİ

DOCTOR OF PHILOSOPHY THESIS

ADVISOR

ASST. PROF. MURAT ÜÇÜNCÜ

CO-ADVISOR

ASSOC. PROF. AYKUT KOÇ

ANKARA – 2023

BAŞKENT UNIVERSITY
INSTITUTE OF SCIENCE AND ENGINEERING

This study, which was prepared by Rasim Akın SEVİMLİ, for the program of Electrical and Electronics Engineering, has been approved in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Electrical and Electronics Engineering Department by the following committee.

Date of Thesis Defense: 04 / 09 / 2023

Thesis Title: Graph-based Object Classification Techniques for Autonomous Vehicle Radar Sensors

Examining Committee Member	Signature
Prof. Dr. Sinan GEZİCİ, Bilkent Üniversitesi Elektrik-Elektronik Müh. Böl	……………..
Doç.Dr. (Eş Dan) Aykut KOÇ, Bilkent Üniversitesi Elektrik-Elektronik Müh. Böl	………………
Doç.Dr. Selda GÜNEY, Başkent Üniversitesi Elektrik-Elektronik Müh. Böl	……………..
Dr. Öğretim Üyesi Hayrullah YILDIZ, Başkent Üniversitesi Elektrik-Elektronik Müh. Böl	……………...
Dr. Öğretim Üyesi Atila Bostan , Ankara Bilim Üniversitesi Bilgisayar Müh. Böl. Bşk.lığı	………………
Dr. Öğretim Üyesi (Dan) Murat ÜÇÜNCÜ, Başkent Üniversitesi Elektrik-Elektronik Müh. Böl.Bşk.lığı	....

APPROVAL

Prof.Dr. Faruk ELALDI
Director, Institute of Science and Engineering

Date: … / … / ……..

BAŞKENT ÜNİVERSİTESİ
FEN BİLİMLER ENSTİTÜSÜ
DOKTORA TEZ ÇALIŞMASI ORİJİNALLİK RAPORU

Tarih: 04 / 09 / 2023

Öğrencinin Adı, Soyadı: Rasim Akın SEVİMLİ

Öğrencinin Numarası: 21820419

Anabilim Dalı: Elektrik-Elektronik Mühendisliği Ana Bilim Dalı

Programı: Doktora Programı

Danışmanın Unvanı/Adı, Soyadı: Dr. Öğretim Üyesi Murat ÜÇÜNCÜ

Tez Başlığı: Graph-based Object Classification Techniques for Autonomous Vehicle Radar Sensors

Yukarıda başlığı belirtilen Doktora tez çalışmamın; Giriş, Ana Bölümler ve Sonuç Bölümünden oluşan, toplam 76 sayfalık kısmına ilişkin, 03/09/2023 tarihinde tez danışmanım tarafından adlı intihal tespit programından aşağıda belirtilen filtrelemeler uygulanarak alınmış olan orijinallik raporuna göre, tezimin benzerlik oranı % 17 'dir. Benzerlik oranının Ek'teki Turnitin raporunda da görüldüğü üzere %13'ü tezden türetilmiş makalemiz nedeniyledir. Diğer kısımlar için benzerlik oranı %4'tür.

Uygulanan filtrelemeler:

1. Kaynakça hariç
2. Alıntılar hariç
3. Beş (5) kelimeden daha az örtüşme içeren metin kısımları hariç

"Başkent Üniversitesi Enstitüleri Tez Çalışması Orijinallik Raporu Alınması ve Kullanılması Usul ve Esaslarını" inceledim ve bu uygulama esaslarında belirtilen azami benzerlik oranlarına tez çalışmamın herhangi bir intihal içermediğini; aksinin tespit edileceği muhtemel durumda doğabilecek her türlü hukuki sorumluluğu kabul ettiğimi ve yukarıda vermiş olduğum bilgilerin doğru olduğunu beyan ederim.

Öğrenci İmzası:………………….

ONAY

04 / 09 / 2023

Öğrenci Danışmanı

Dr. Öğretim Üyesi Murat ÜÇÜNCÜ

To my Sevimli family & son H. Enes

ACKNOWLEDGEMENTS

My deep gratitude goes first to Asst. Prof. Murat Üçüncü and Assoc. Prof. Aykut Koç who expertly guided me through this thesis and shared the excitement for four years of the research. Their unwavering wisdoms kept me constantly engaged with my research. I would also like to thank Prof. Sinan Gezici and Asst. Prof. Hayrullah Yıldız as my Thesis Tracking Committee (TİK) members for their supervision, support, encouragement and suggestions for the completion of my thesis from the beginning to the end. Also, I would like to thank Asst. Prof. Atilla Bostan and Assoc. Prof. Selda Güney for serving on my dissertation committee and their guidance on my dissertation.

I owe special thanks to my family, whose unwavering support and love have been the bedrock of my journey through the challenging terrain of doctoral research. I would like to express my deepest gratitude to my wife M. Ebru Sevimli. I owe a deal of gratitude to her for the support and motivation she provided without which this thesis would not have come to fruition. I would like to congratulate my son H. Enes Sevimli for being insightful throughout the entire thesis. I cannot find the words to adequately convey how grateful I am, to my family and relatives. Arif and Zühal, who have always been supportive and understanding, a brother and sister—Mert and Melek, who have always helpful and motivate, grandparents—Rasim and Melek, who are unforgettable people for me, mother and father in law—Davut and Nihal, who have always been supportive and understanding.

I wish to extend my thanks to all of my friends and colleagues for their valuable help in the development of this thesis. I would like to thank all my friends at BMC Automotive Comp., for their supportive and understanding approaches. I especially would like to thank my friends Serkan Sarıtaş, Ahmet Dündar Sezer, Ömer Ahıskalı, Abdullah Gümüş, Ufuk Uyan, who have been on my side during my happy and difficult days.

ÖZET

Rasim Akın SEVİMLİ

OTONOM ARAÇ RADAR SENSÖRLERİ İÇİN ÇİZGE TABANLI NESNE SINIFLANDIRMA TEKNİKLERİ

Başkent Üniversitesi Fen Bilimleri Enstitüsü

Elektrik ve Elektronik Mühendisliği Bölümü

2023

Otomotiv endüstrisi geliştikçe, otonom araçlara durumsal farkındalık sağlayan görsel algılama sistemlerinin önemi hayati hale gelmiştir. Geleneksel derin sinir ağları son on yılda 2D Öklid problemleri çözmede başarılı olmuş olsa da, nokta bulutlarının analizi, özellikle RADAR verisi, düzensiz yapısı ve karmaşık 3D geometrisi nedeniyle 2D sinyal işleme için uygun değildir ve önemli zorluklar çıkarmaktadır. Bu sorunu çözmek için, bu tezde RADAR nokta bulutları için Çizge ve Dönüştürücü tabanlı sınıflandırma yöntemleri kullanarak iki yeni metot önerilmektedir. Bu çalışmaların yenilikçi yönü, Çizge ve/veya Dönüştürücü tabanlı yöntemler kullanılarak nesne nokta tespiti yapılabilen bir yapı geliştirmekte yatmaktadır. Oluşturulan her bir model, baştan sona derin öğrenme yapısını kullanırken RADAR noktaları ve öznitelik vektörleri üzerinde çizge evreşimlerini içermektedir. Öznitelik vektörleri, sensör verilerinin bağlamsal bir temsili oluşturmak amacıyla GSP çatısı altında üretilmektedir. Kapsamlı derin öğrenme yöntemlerine yönelik eğilimin popülaritesi artmaya devam ederken, her ikisi de çizge tabanlı algoritmalar kullanılarak oluşturulan RADAR-DGCNN ve RADAR-PointNGCNN yöntemler önerilmektedir. Başlangıçta önerdiğimiz yaklaşımlarımız çoğunlukla çizge ile ilişkilendirilen yöntemleri kullanırken, sınıflandırma sonuçlarını GSP çatısı altında bir Dönüştürücü ağına uyarlama ile iyileştirilmiştir. Önerilen yöntemlerimizin etkinliğini doğrulamak için halka açık nuScenes ve RadarScenes nokta bulutu veri kümelerini kullanarak deneyler gerçekleştirilmiştir. Bu zorlu karşılaştırmalı veri kümeleri üzerinde yapılan geniş kapsamlı deneylerle, önerilen yöntemimizin RADAR nokta bulutu üzerindeki performans açısından mevcut en iyi temel yöntemleri geride bıraktığı görülmüştür.

ANAHTAR KELİMELER: *Çizge Sinyal İşleme (ÇSİ), Otomotiv Radarı, Nokta Bulutu İşleme, Hassas Karayolu Kullanıcısı (HKK) Sınıflandırması, Nokta Bulutu Dönüştürücüleri, Radar Nokta Bulutu İşleme.*

ABSTRACT

Rasim Akın SEVİMLİ
GRAPH-BASED OBJECT CLASSIFICATION TECHNIQUES FOR AUTONOMOUS VEHICLE RADAR SENSORS
Başkent University Institute of Science
Department of Electrical-Electronics Engineering
2023

As the automotive industry continues to evolve, the importance of visual perception systems that provide situational awareness to autonomous vehicles has become crucial. While traditional deep neural networks have been successful in solving 2D Euclidean problems over the past decade, the analysis of point clouds, especially RADAR data, presents significant challenges owing to its irregular structure and intricate 3D geometry, which are not well-suited for 2D signal processing. To address this issue, we propose two novel approaches by using Graph and Transformer based classification methods for RADAR point clouds in this thesis. The novel aspect of these studies lie in the development of an object point detection pipeline utilizing Graph and/or Transformer based methods. Each of the created models utilizes a deep learning structure from start to finish, incorporating graph convolutions on both RADAR points and feature vectors. Feature vectors are generated within the GSP framework to create a contextualized representation of the sensor data. As the movement towards comprehensive deep learning methods continues to grow in popularity, we initially present the RADAR-DGCNN and RADAR-PointNGCNN methods, both of which are constructed using graph-oriented algorithms. While our initial suggested approaches predominantly utilize graph-related methods, we improve the classification outcomes by integrating adaptations into a Transformer network within the GSP framework. To validate the effectiveness of our proposed methods, we conduct experiments using publicly available nuScenes and RadarScenes point cloud datasets. Through extensive experimentation on these challenging benchmark datasets, we demonstrate that our proposed method outperforms state-of-the-art baselines studied on the RADAR point cloud in terms of performance.

KEYWORDS: *Graph Signal Processing (GSP), Automotive RADAR, Point Cloud Processing, Vulnerable Road User (VRU) Classification, Point Cloud Transformers, RADAR Point Cloud Processing*.

TABLE OF CONTENTS

APPENDIX

APPENDIX A: Datasets Class Mapping Tables

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ABBREVIATIONS

2D/3D	Two-Dimensional / Three-Dimensional
AAF	Anti-Aliasing Filter
AI	Artificial Intelligence
ACC	Adaptive Cruise Control
AEBS	Advanced Emergency Braking System
ADAS	Advanced Driver-Assistance Systems
ADC	Analog-to-Digital Converter
AV	Autonomous Vehicle
BSD	Blind Spot Detection
CFAR	Constant False Alarm Rate
CNN	Convolutional Neural Network
DBSCAN	Density-Based Spatial Clustering of Applications with Noise
DGCNN	Dynamic Graph Convolutional Neural Network
DSP	Digital Signal Processing
F_1-Macro	Macro-Averaged F1 Score
FMCW	Frequency Modulated Continuous Wave
GCN	Graph Convolutional Network
GNN	Graph Neural Network
GSP	Graph Signal Processing
GFT	Graph Fourier Transform
Hz	Hertz
k-NN	k-Nearest Neighbour
LDGCNN	Linked Dynamic Graph CNN
LiDAR	Light Detection And Ranging
LNA	Low Noise Amplifier
LRR	Long Range Radar
OGM	Occupancy Grid Mapping
MLP	Multi-Layer Perceptron
MMPoint-GNN	Graph Neural Network with Dynamic Edges for Human Activity Recognition through a Millimetre-Wave Radar
MRR	Medium Range Radar
PointNGCNN	Deep convolutional networks on 3D point clouds with neighbourhood graph filters
RADAR	Range Detection And Ranging
RCS	Radar Cross Section
RF	Radio Frequency
RSP	Radar Signal Processing
SA	Self-Attention
SAR	Synthetic Aperture Radar
SIFT	Scale Invariant Feature Transform
SNR	Signal-to-Noise Ratio
SSL	Self-Supervised Learning
SRR	Short Range Radar
VCO	Voltage-Controlled Oscillator
VRU	Vulnerable Road User

1. INTRODUCTION

Autonomous driving is revolutionizing the automotive industry and transforming the way we think about transportation. This cutting-edge technology enables vehicles to work without human intervention, which utilizes artificial intelligence (AI) [1]. The concept of autonomous driving has gained significant attention and importance in recent years, and it holds immense potential for many applications and industries [2].

Autonomous driving has the potential to revolutionize road safety, enhance transportation efficiency, help solve the traffic problem, and offer solutions for the mobility of people who cannot drive. This transformative technology has received interest from leading automotive manufacturers, technology companies, policymakers, and researchers worldwide. Autonomous driving systems have the potential for reshaping our cities, transportation networks, and daily lives [3].

Some well- known capabilities of autonomous driving can be listed as: Road safety, transportation efficiency, accessibility and mobility, environmental impact.

Road Safety: Autonomous driving can eliminate human error and reduce accidents. The potential to prevent up to 94% of accidents caused by human error is a key benefit [4].

Transportation efficiency: Autonomous driving technology has the potential to greatly enhance transportation efficiency. Self-driving vehicles can communicate with each other and with transportation infrastructure, enabling them to optimize routes, reduce congestion, and minimize travel times [5].

Accessibility and Mobility: Another critical aspect of autonomous driving technology is its potential to provide newfound mobility options for people with disabilities [6]. *Environmental Impact:* Autonomous driving reduces the environmental impact of transportation by using clean energy and optimizing fuel consumption [7].

Autonomous driving systems employ a range of sensors that work together to gather and process data from the surrounding environment [8]. These sensors can be listed as RADAR (Radio Detection and Ranging), LiDAR (Light Detection and Ranging), ultrasonic sensors, cameras, Global Positioning System (GPS). The capability of sensors changes the strengths of accurate sensing environment. Autonomous vehicles employ sensors both independently and in combination by using the sensor fusion technique [9]. The complete block diagram of the multi-sensor perception systems is depicted in Figure 1.1. The data collected by these sensors is then analysed and used to make informed decisions in real-time, ensuring precise control and navigation. The integration of these sensors enables autonomous vehicles to navigate safely, make informed decisions, and revolutionize the way we travel. Continued advancements in sensor technology and Artificial Intelligence (AI) algorithms will drive further improvements in the safety and reliability of autonomous driving systems. In an autonomous vehicle, all perception sensors send individual object detections to the sensor fusion module. While only the camera sensor is used in some autonomous vehicles, some vehicles use Camera-Radar, some use Camera-LiDAR, and some use all of them. However, the structure created using only one sensor is not very feasible. In this thesis, the classification methods of automotive radars are emphasized in terms of helping the camera or LiDAR point object detection information.

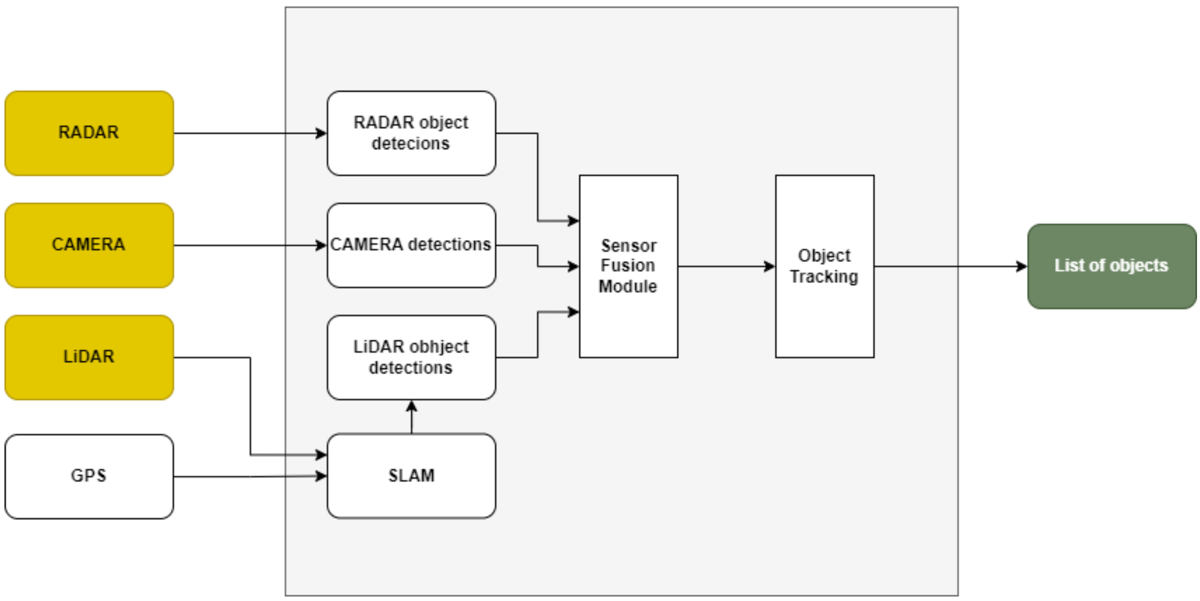

Figure 1.1 Multiple sensor perception system in autonomous driving.

Automotive RADARs are integral components of autonomous driving systems, providing crucial capabilities for safe and efficient navigation. By leveraging RADAR technology, autonomous vehicles can accurately perceive their surroundings, anticipate hazards, and make informed decisions in real-time, ensuring a higher level of safety and reliability [10], [11]. These RADAR systems utilize radio waves to detect objects, enabling autonomous vehicles to precisely determine distance, angle, and speed of objects. If we compare RADAR and LiDAR in the view of autonomous applications, we see some advantages over the other. The work with principles, but each benefits different kinds of sources. RADAR has an antenna to transmit radio signals, but LiDAR utilised specialized optics and lasers for receiving and transmitting signals. For autonomous vehicle applications, LiDAR may be a good choice. However, for rough environments, RADAR is a good choice [12]. In Figure 1.2, both RADAR and LiDAR point clouds are illustrated in an environment. RADAR point cloud are more sparse as compared to the LiDAR point cloud. The vehicle only has seven RADAR points, whereas the number of LiDAR points is significantly higher.

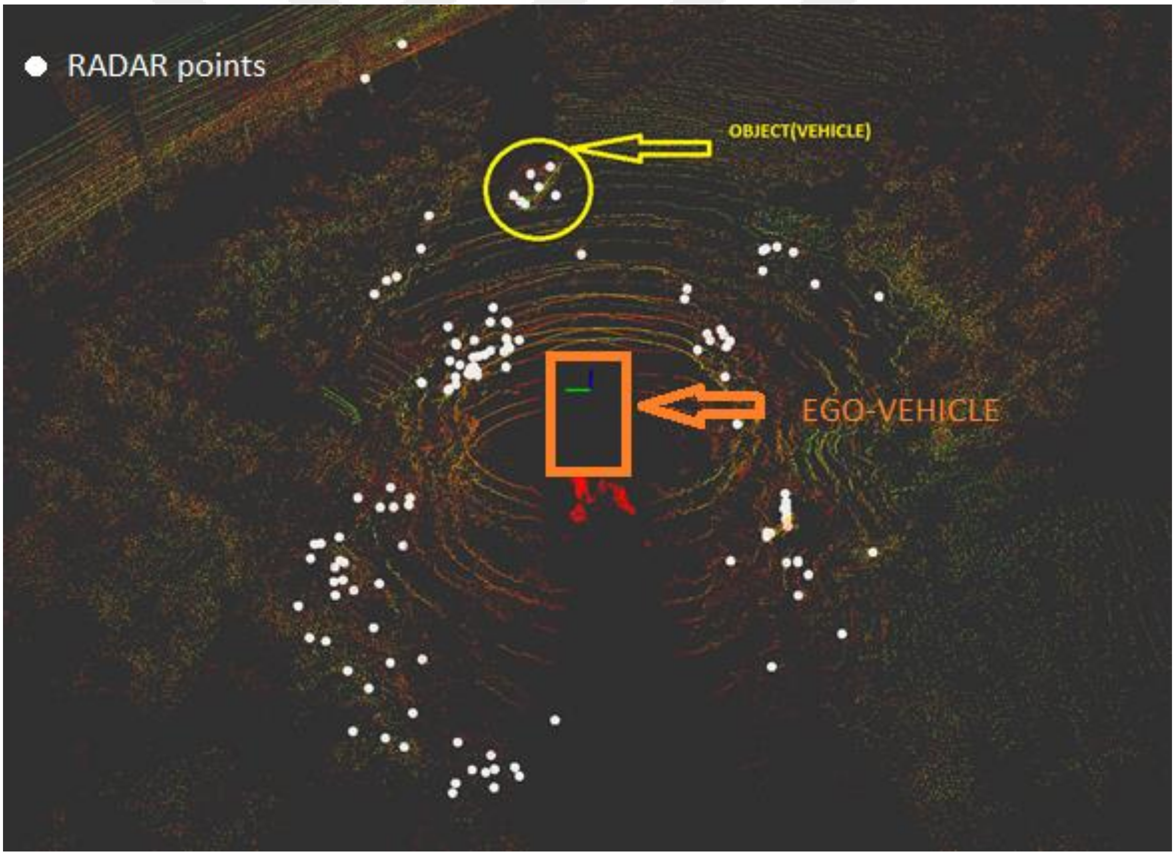

Figure 1.2 An example LiDAR and RADAR point clouds in the same environment.

Table 1.1 displays the strengths and weaknesses of LiDAR compared to RADAR, and vice versa, using a dot scale (Maximum: ●●●●●, Minimum: ○○○○○). In addition, we give the detailed comparison by explaining in the following. Remember that these comparisons are generalizations, and the performance of LiDAR and RADAR can vary based on specific technologies, implementations, and use cases.

Table 1.1 The advantages and disadvantages of LiDAR over RADAR and vice versa.

	Sensitivity to Weather Conditions	**Precision in Object Detection**	**Spatial Resolution**	**Penetration of Materials**	**Complexity of Data Processing**	**Cost-effectiveness**	**Environmental Friendliness**	**Use in Autonomous Vehicles**
RADAR	●●●○○	●●○○○	●○○○○	●●●●○	●●○○○	●●●●○	●●●●●	●●●●○
LiDAR	●●○○○	●●●●●	●●●●○	●○○○○	●●●○○	●○○○○	●○○○○	●●●○○

Sensitivity to Weather Conditions:

LiDAR is somewhat sensitive to adverse weather conditions like rain and snow, while RADAR is more resilient in such situations.

Precision in Object Detection:

LiDAR excels in precise object detection due to its high-resolution point cloud, while RADAR might struggle with detailed detection.

Spatial Resolution:

LiDAR provides better spatial resolution, leading to more accurate 3D mapping compared to RADAR.

Penetration of Materials:

RADAR can penetrate various materials, making it suitable for applications where object detection through obstacles is important.

Complexity of Data Processing:

LiDAR data processing is somewhat complex due to the detailed point cloud, while RADAR data processing is comparatively simpler.

Cost-effectiveness:

RADAR technology tends to be more cost-effective compared to LiDAR, making it a preferred choice in some applications.

Environmental Friendliness:

RADAR is more environmentally friendly as it doesn't emit potentially harmful light, unlike LiDAR.

Use in Autonomous Vehicles:

Both LiDAR and RADAR play essential roles in autonomous vehicles, with LiDAR providing detailed perception and RADAR offering very good object detection.

The advanced capabilities of automotive RADARs provide a comprehensive understanding of the surroundings, empowering autonomous vehicles to navigate securely and execute crucial functions detailed in [11], Additionally, RADARs find applications in diverse domains covered in [13], [14]. These functions rely on automotive RADARs to detect and classify objects, such as pedestrians and cars, ensuring the safe operation of autonomous vehicles [15], [16].

Various point cloud models have been developed to represent and process 3D data. These models provide valuable insights into the structure, geometry, and semantics of objects in a scene. Regarding to the model used in neural networks for point cloud processing, we can look at four popular point cloud models: *volumetric, point-based, graph-based, and transformer-based* models. These models employ different approaches and techniques to analyze and understand 3D data, each with its strengths and applications [17].

Volumetric-based models:

Volumetric models represent point clouds by discretizing the 3D space into a regular grid or voxel grid. Each voxel in the grid is associated with properties such as occupancy, color, or density, providing a structured representation of the point cloud data shown in Figure 1.3. Volumetric models are commonly employed in several applications [18].

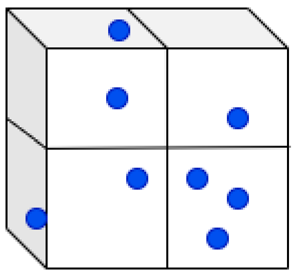

Figure 1.3 The volumetric-based point cloud representation.

Point-based models:

Point-based models directly operate on the individual points in the point cloud without explicit grid-based representations. They preserve the spatial irregularity of the data and can be more memory-efficient compared to volumetric models. As an illustration, point-based representation of point cloud is given in Figure 1.4. PointNet and its variants are popular point-based models that have shown success in tasks like object classification, part segmentation, and point cloud generation [19], [20].

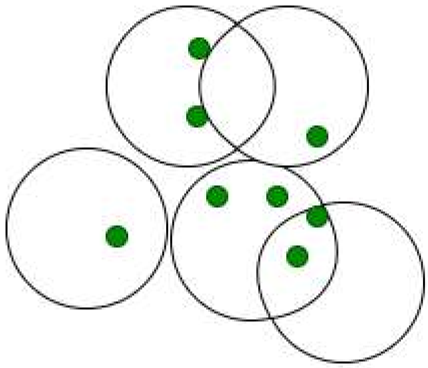

Figure 1.4 The point-based point cloud representation.

Graph-based models:

Graph-based models represent the point cloud as a graph. Each point in the graph corresponds to a node, and edges capture the relationships between neighboring points. An example graph-based representation is shown in Figure 1.5. Graph neural networks (GNNs) are then employed to process the graph structure and learn representations for various tasks, including object classification, segmentation, and scene understanding [21].

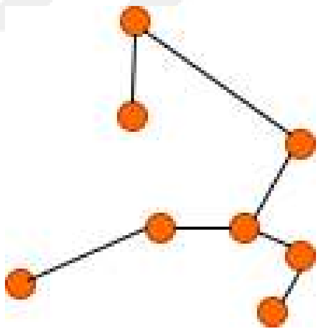

Figure 1.5 The graph-based point cloud representation.

Transformer-based models:

Transformer-based models have recently gained attention for point cloud processing. They adapt the transformer architecture, originally developed for sequential data such as natural language processing, to handle the spatial relationships in point clouds. A sample image is illustrated in Figure 1.6. Transformer-based models capture long-range dependencies and have achieved excellent performance in point cloud tasks like object detection, segmentation, and generation [22].

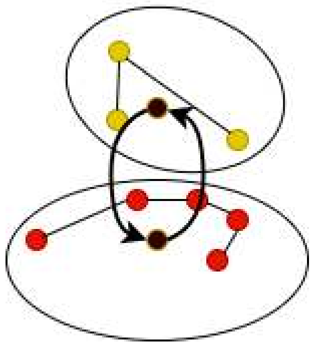

Figure 1.6 The Transformer-based point cloud representation.

Apart from point cloud classification models that operate on a point-wise basis, clustering-based RADAR object detection can be introduced as an alternative approach. Clustering algorithms group data points based on their spatial proximity or similarity in features, enabling the identification of distinct objects or regions within the point cloud. While clustering-based algorithms may not be as commonly preferred or practical as point-wise methods, they find significant usage in the autonomous driving domain.

In the light of this much information, we propose two comprehensive methods applied on automotive RADAR point cloud. The inherent irregularity and complexity of RADAR point cloud data call for sophisticated approaches that can effectively capture the spatial relationships and dependencies among the data points. Graph-based models leverage the graph structure to represent the interconnections between points, allowing for efficient and accurate classification. On the other hand, transformer-based models adapt the powerful self-attention mechanism to retrieve long-range dependencies and capture complex patterns in the point cloud. These methods have shown promising results in RADAR point cloud classification, offering improved performance and robustness in handling diverse and challenging scenarios. By employing graph-based and transformer-based methods, we can enhance the understanding and interpretation of RADAR point cloud data, enabling more accurate and reliable classification for various applications such as target recognition, object detection, and situational awareness. Furthermore, we present an innovative application of clustering-based RADAR object detection.

1.1. Thesis Outline and Contribution

This thesis starts with a background review for autonomous driving, automotive RADARs, point cloud processing models in introduction part. The principles and millimetre wave RADAR are introduced in Chapter 2, including general hardware architecture and RADAR signal processing background. In Chapter 3, Graph Signal Processing (GSP) theory is summarised. In Chapter 4, a comprehensive overview of the datasets employed in this thesis is provided, along with sample scenes for illustration. Chapter 5 introduces two innovative approaches. One method is based on GSP technique and the other method is a novel method which we called RadGT. RadGT leverages both graph and Transformer-based techniques. Additionally, Chapter 4 explores clustering-based methods that operate within the graph framework. Chapter 5 encompasses all essential sections, including related work, model explanation, experimental results, and conclusion. In Chapter 6, the thesis concludes by highlighting significant findings derived from the experimental results. Chapter 7 presents several implications that warrant careful consideration, along with a range of potential future ideas.

2. PRINCIPLES OF MILLIMETER WAVE (mmWave) RADAR

RADAR, short for "RAdio Detection And Ranging," is an active detection system. RADAR benefits from radio frequency (RF) signals to determine the range, angle, and velocity for objects in the surrounding environment. Unlike passive sensing systems, such as human vision that relies on receiving reflected light or signals, RADAR actively illuminates the environment with modulated RF signals and receives the returning echoes from objects. A basic RADAR configuration consists of two main elements: a transmitter and a receiver. When the receiver picks up a signal that has been reflected back, it is considered as a detection. It's worth noting that a single transmitted signal can result in multiple detections, which may be caused by various objects on the ground and in the surrounding area. In RADAR terminology, an object of particular interest is commonly referred to as a "target," and it can produce several RADAR detections, as depicted in Figure 2.1. Figure 2.1 presents a host vehicle equipped with a RADAR system positioned at its front. As we see in the figure, the RADAR identifies four detections, with three of them being associated with a target, represented here as a pedestrian. The forth does not belong to target. The significant detections are highlighted in orange. By analyzing the transmitted reference signals and the received echoes, RADAR extracts valuable information about the detected objects.

Figure 2.1 Depicting a host vehicle equipped with a RADAR system positioned at its front.

The frequency range of modulated RF signals varies depending on the specific RADAR application, as electromagnetic waves exhibit different carrier frequencies. Until recently, low-frequency RADAR systems have had a long history of development, while the mmWave RADAR, operating at frequencies between 60-64 GHz or 76-81 GHz and featuring wavelengths in the millimeter range, has gained popularity in various commercial applications [23]. The 60 GHz RADAR is mainly suitable for indoor use due to significant energy loss caused by atmospheric attenuation. On the other hand, the 76/77 GHz RADAR is specifically designated for automotive applications. One of the major advantages of mmWave RADAR is its compact RF components and antenna design, thanks to its high carrier frequency. Additionally, mmWave RADAR can achieve a signal bandwidth of up to 4 GHz, resulting in an impressive range resolution of up to 3.375 cm.

In mmWave RADAR technology, different types of systems are used by considering their range and application requirements. Classification of RADARs in terms of ranges include Long Range Radar (LRR), Medium Range Radar (MRR), and Short Range Radar (SRR) [24]. LRR systems operate in the millimeter wave frequency range and are designed for long-range detection and tracking of objects. They typically have a high transmit power and can cover a distance of several hundred meters. LRR RADARs are commonly used in applications such as highway surveillance, collision avoidance systems for autonomous vehicles, and perimeter security. MRR systems are optimized for medium-range applications and provide a balance between range and resolution. They are often employed in scenarios where a range of tens to a few hundred meters is required. MRRs find applications in adaptive cruise control, blind-spot detection, and traffic monitoring systems. SRR systems are designed for close-range detection and are commonly used in applications where precise object detection and localization are crucial. SRRs typically have a shorter detection range, often ranging from a few centimetres to tens of meters. They are widely deployed in parking assistance systems, pedestrian detection, and collision warning systems.

The Pulse-Doppler RADAR requires a digital replica of the transmitted signal to extract echo information. However, this increases system costs as it necessitates a high sampling rate Analog-to-Digital Converter (ADC) to digitize the wideband RF pulses. Conversely, in cost-sensitive commercial applications with detection ranges typically within a few hundred meters, mmWave RADAR employs a full-duplex Frequency Modulated Continuous Wave (FMCW) architecture. This approach allows simultaneous operation of the transmitter and receiver, albeit with a limit on maximum transmitting power due to imperfect isolation between the two components. Nonetheless, it ensures a continuous replica of the transmitted signal, enabling the extraction of echo information using an RF mixer before digitalization. Consequently, the mmWave FMCW RADAR is a preferred choice for such commercial applications.

2.1. Hardware Architecture

The hardware architecture of mmWave RADAR systems plays a crucial role in enabling their advanced sensing capabilities. The hardware architecture of a typical mmWave RADAR consists of several key components, including antennas, transceivers, digital signal processors (DSPs), and power management units. These components work in tandem to transmit and receive mmWave signals, process the received data, and extract valuable information about the surrounding environment. Figure 2.2 depicts the hardware configuration of a standard mmWave FMCW RADAR incorporating a quadrature receiver.

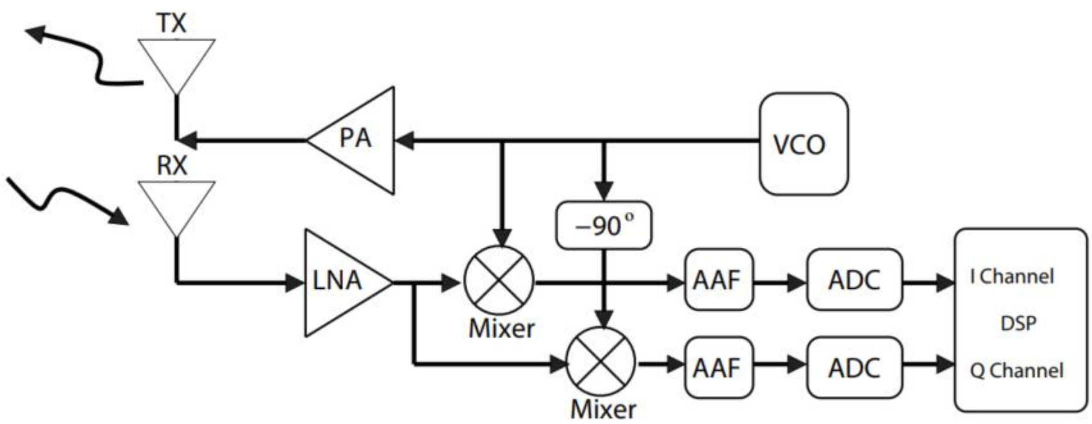

Figure 2.2 Hardware design of standard mmWave RADAR [25].

The design of the hardware architecture involves considerations such as antenna design, RF circuitry, analog-to-digital converters (ADCs), digital signal processing algorithms, and power management techniques. The voltage-controlled oscillator (VCO) is responsible for sweeping the linear frequency modulated signal, also known as the chirp signal. This is followed by an RF power amplifier that increases the transmitted power. The chirp signal is then propagated through space using antennas. Upon encountering objects, the signal reflects back to receiving antennas. To have a good SNR, LNA boosts the received signal's strength. The received echoes are then input to the quadrature mixer. Before passing through the anti-aliasing filter (AAF), both the in-phase and quadrature components of the echoes are combined with the local reference from the transmitted LFM signal. Subsequently, the analog-to-digital converter (ADC) changes the filtered signal into digital form. Finally, the

DSP implements standard RADAR signal processing algorithms to ascertain the range, angle, and velocity of objects within the surroundings.

2.2. RADAR Signal Processing (RSP)

RSP involves the analysis and manipulation of the signals received by a RADAR system to extract meaningful information about the surrounding environment. Various algorithms and techniques are employed to enhance the RADAR's performance in terms of range resolution, target detection, and tracking. In the context of FMCW RADARs, a typical signal processing chain produces a collection of points in a two-dimensional spatial coordinate system denoted as $x \in \mathbb{R}^2$. These points represent RADAR detection information and are associated with object velocity in the direction the sensor is facing ($v \in \mathbb{R}$) and the RADAR Cross Section (RCS) ($\sigma \in \mathbb{R}$). To prevent any confusion, these points, conventionally referred to as RADAR detection points, will be referred to as RADAR points in this thesis. It is important to note that a single object instance can give rise to multiple RADAR points depending on factors such as object size, material properties, and the distance from the RADAR system.

A RADAR has the ability to calculate both the distance to a detection and a target by analysing the time for the transmitted signal to be received. This distance, denoted as "r," can be easily calculated by using a straightforward equation in the following:

$$r = \frac{ct}{2}, \tag{2.1}$$

where t represents the time for the RADAR signal to return after echoing, while c denotes the velocity of electromagnetic waves in the medium, specifically the speed of light in the air.

The radial velocity of a target is its velocity along the line-of-sight direction from the RADAR. It can be determined by using the Doppler frequency shift of the received RADAR signal [26]. The Doppler frequency shift (Δf) can be described by Eq. (2.2).

$$\Delta f = \frac{2v}{\lambda} f, \tag{2.2}$$

where Δf is the Doppler frequency shift in hertz (Hz), v is the radial velocity in meters per second (m/s), λ is the wavelength in meters, f is the carrier frequency of the RADAR signal in hertz (Hz). RADAR cross section (RCS) is a parameter which indicates how well it reflects the RADAR signal back to the RADAR system. It depends on the size, shape, and material properties of the target. RCS is typically calculated by the Eq. (2.3).

$$\mathrm{RCS} = \frac{4\pi A\sigma}{\lambda^2}, \tag{2.3}$$

where A is the effective illuminated area of the target (m²), σ is the scattering cross section (m²), λ is the wavelength (m) .

2.2.1. Frequency Modulated Continuous Wave (FMCW) RADAR

FMCW RADAR utilizes continuous transmission and frequency modulation to measure the range and relative velocity, range, and angular position of objects in its field of view. This technology is particularly effective in automotive applications due to its high range resolution, robustness against multi-path interference, and ability to accurately estimate target velocity. In FMCW RADAR, the transmitted signal is modulated with a linearly increasing or decreasing frequency. The distance to the target is easily determined by using the transmitted and received signals' frequency. The resolution for the distance of an FMCW RADAR can be determined through the sweep bandwidth and the frequency sweep the duration. The FMCW RADAR transmits a continuous signal with a modulated frequency, while the received signal is processed by using Fast Fourier Transform (FFT) to extract information about the target's range and velocity [27]. The basic equation defining the received signal in terms of the transmitted signal is given by Eq. (2.4).

$$\mathrm{Rx}(t) = \mathrm{Tx}(t - \tau)\exp(j2\pi\Delta f_r(t - \tau)), \tag{2.4}$$

where Rx(t) represents the received signal at time t, Tx(t - τ) is the transmitted signal, τ is the round-trip delay, Δf_r is the frequency modulation rate of the transmitted signal. Figure 2.3 illustrates the plot of the received and transmitted signals, along with the utilization of FFT calculation to generate a Range-Doppler (RD) map. RD map provides a comprehensive visualization of the RADAR echoes in a two-dimensional plot, with the range displayed along one axis and the Doppler shift along the other axis.

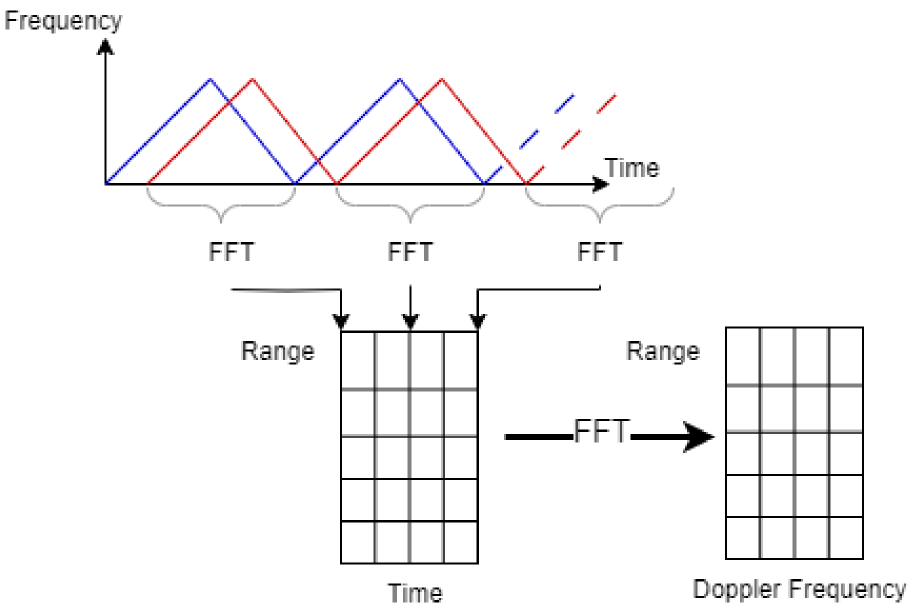

Figure 2.3 FMCW RADAR Rx/Tx signals along with range-Doppler map [28].

The frequency content can be analysed to obtain the distance and velocity of the targets in the RADAR's field of view. The analysis is carried out by applying FFT on the received signal. The range is easily determined by using the delay τ defined in Eq.(2.4), while the Doppler shift extracted from the FFT provides the velocity information. The use of FFT allows for efficient processing of the received signals, enabling accurate range and velocity estimation in FMCW RADAR systems.

2.2.2. Point Cloud Generation and Constant False Alarm Rate (CFAR)

RADAR point generation is another important issue which involves the process of creating points in a coordinate system to represent the position and characteristics of RADAR targets. The RADAR data contains a significant level of noise, primarily originating from multiple surface reflections, background electromagnetic radiation, and additional sources like reflections caused by rain particles. CFAR algorithm is frequently used in RSP signal processing to mitigate the effects of clutter and noise.

In scenarios where noise levels are sufficiently low, a straightforward approach to solve this problem involves establishing a specific threshold for the signal strength. This threshold effectively filters out all signals below its defined value. When opting for a lower threshold, a greater number of false alarms may arise, yet it reduces the risk of missing genuine targets. Conversely, a higher threshold would result in fewer false alarms, but it bears a higher risk of erroneously filtering out legitimate detections. Therefore, finding the appropriate balance is crucial to achieving optimal performance in RADAR signal processing with minimal false alarms while ensuring real targets are accurately identified.

3. THE THEORY OF GRAPH SIGNAL PROCESSING (GSP)

GSP is a branch of signal processing with signals defined on graphs [29]. It extends traditional signal processing concepts to non-Euclidean domains, enabling the analysis, processing, and understanding of data on irregular structures. GSP provides powerful tools for capturing the inherent structures and dependencies present in graph data, making it applicable to various fields.

Graph signal processing involves simultaneous exploration of the inherent geometric structure of data space and the analysis of discrete data signals within the identified domain. This interdisciplinary field combines knowledge from algebraic and spectral graph theory, enabling a comprehensive approach to understanding and processing signals on graphs [30].

The exploration of the theory of Graph Signal Processing (GSP) begins by grasping the concept of DSP. DSP has demonstrated significant efficacy in handling signals that are time-dependent (e.g., speech, communication, RADAR, or econometric time series), spatially-dependent (e.g., images, multidimensional signals) and signals that combine both time and space components (e.g., video). GSP expands the traditional theory of signal processing to encompass arbitrary graphs [30], [31], [32]. In essence, both DSP and GSP share common areas of study, including the analysis of signals and their representations, the design of systems such as filters, the application of signal transforms like the Fourier transform and z-transform, as well as the sampling of signals.

The signal is typically sampled at evenly spaced time points. The time intervals between samples is a crucial parameter in signal processing. This relationship between sampling points can be treated in terms of a graph. Here, the graph consists of vertices representing the sampled data values and edges connecting these vertices. The graph leverages the intrinsic connections among the data based on their relevant properties. The vertices correspond to the instances where sampling occurs, while the edges establish the ordering of the these points. In the case of uniform sampling instants, all edges can be assigned the same weights, as depicted in Figure 3.1.

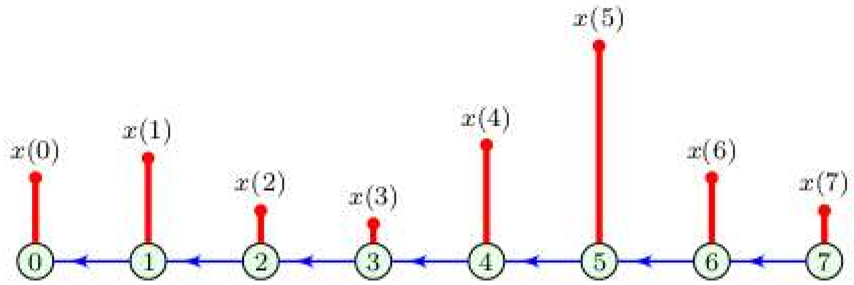

Figure 3.1 A classical time-domain signal graph depiction.

Another important issue about DSP is shifting or delaying. We can represent signals by defining polynomials in z^{-1} and build filters as in z^{-1}. These concepts can also be extended to graph signals whose samples are indexed by the nodes of arbitrary graphs. In digital signal processing, periodicity is the case that sample $x(\mathrm{N}-1)$ is succeeded by $x(0)$ as depicted by **Figure 3.2**. This model is employed by several signal processing algorithms [31].

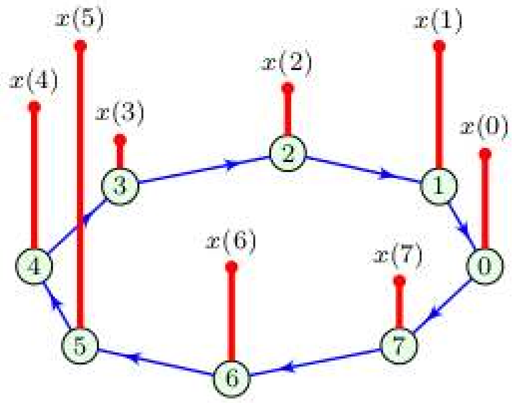

Figure 3.2 A periodic signal in the form of a graph

The key concepts in GSP include graph representation, Graph Fourier transform (GFT), graph filtering, and graph sampling. Graph representation involves modelling data as a graph, where nodes represent elements and edges represent relationships or connections. The Graph Fourier transform allows the analysis of graph signals in the frequency domain, analogous to the Fourier transform in traditional signal processing. Graph filtering involves in applying filters to graph signals to extract desired information or remove noise. Graph sampling addresses the challenge of acquiring and reconstructing graph signals from a limited set of samples.

A weighted and undirected graph is represented by the tuple $\mathcal{G}(\boldsymbol{\mathcal{V}},\ \boldsymbol{\mathcal{E}},\ W)$, where $\boldsymbol{\mathcal{V}}$ indicates a finite set of N nodes or vertices [33]. The edge set $\boldsymbol{\mathcal{E}}$ defines the connections between nodes in the graph topology. The weight matrix W assigns weights to the similarities between two vertices connected by an edge, obtained using a weight function equation that establishes connections between data points. An example of an undirected graph with six vertices and six edges is illustrated in Figure 3.3. The example matrices $\boldsymbol{\mathcal{V}}$, $\boldsymbol{\mathcal{E}}$, and W can be represented as follows: $\boldsymbol{\mathcal{V}} = \{\boldsymbol{v_1}, \boldsymbol{v_2}, \boldsymbol{v_3}, \boldsymbol{v_4}, \boldsymbol{v_5}, \boldsymbol{v_6}\}$, $\boldsymbol{\mathcal{E}} = \{(\boldsymbol{v_1}, \boldsymbol{v_2}), (\boldsymbol{v_1}, \boldsymbol{v_3}), (\boldsymbol{v_1}, \boldsymbol{v_4}), \ldots, (\boldsymbol{v_5}, \boldsymbol{v_6})\}$, and $\boldsymbol{W} = \{\boldsymbol{w_1}, \boldsymbol{w_2}, \boldsymbol{w_3}, \boldsymbol{w_4}, \boldsymbol{w_5}, \boldsymbol{w_6}\}$. Here, we have six vertices and six edges. The edge set corresponds to the connection between nodes.

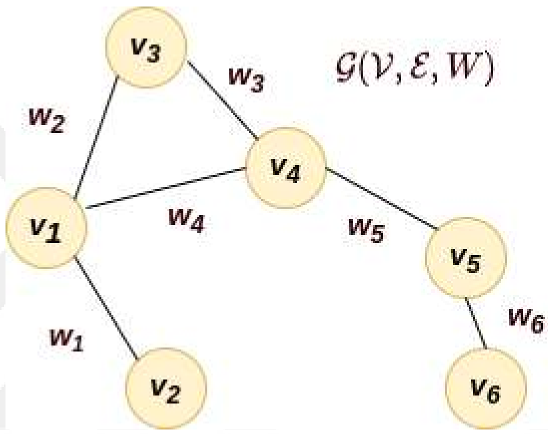

Figure 3.3 An example of undirected graph.

3.1. Graph Fourier Transform (GFT)

GFT is a fundamental operation within GSP that facilitates the examination of graph signals in the frequency domain, analogous to the traditional Fourier Transform used in classical signal processing. The GFT facilitates the decomposition of graph signals into graph frequency components, revealing the underlying graph structure and signal characteristics.

A *graph signal* is a function that assigns a real value to each vertex in a graph. Consequently, it can be represented as a vector, denoted as $x \in \mathbb{R}^N$, where N is the total number of vertices in the graph. To assess the smoothness of the signal x across the graph's vertices, the calculation in Eq.(3.1) can be employed,

$$(\mathbf{Lx}) = \sum_{(i,j)\in\mathcal{E}}^{N} \mathbf{W}_{ij}|x(i) - x(j)|^2, \tag{3.1}$$

This equation can be regarded as a quadratic form denoted by $\mathbf{x^T L x}$**,** where $\mathbf{L}$ represents the Laplacian matrix of the graph. A graph's Laplacian matrix $\mathbf{L}$ is defined as $\mathbf{L} = \mathbf{D} - \mathbf{W}$. Here, the diagonal matrix $\mathbf{D}$ is a degree matrix of an undirected graph, where the diagonal elements are obtained by summing the weights of all edges connected to node i as $\mathbf{D}_{ii} = \sum_{j=1}^{N} \mathbf{W}_{ij}$. In other words, the Laplacian matrix, denoted as **L**, can be regarded as a difference operator, offering a generalization of smoothness concept on the graph. As shown in Eq. 3.1, a signal that demonstrates gentle variations along the graph's vertices will yield a low value for the quadratic form of the graph Laplacian. This observation can be interpreted as revealing the frequency characteristics of the signal on the given graph. Consequently, the Laplacian matrix unveils the spectral properties of the graph through its eigen-decomposition, as elucidated by Chung [34] in the variational method of spectral graph theory. The Laplacian matrix, **L**, possesses essential properties such as being real, symmetric, and positive semidefinite. These characteristics ensure that **L** consists of a complete set of eigenvectors, represented as $\{\mathrm{e_l}\}_{\mathrm{l}=0,1,\ldots \mathrm{N}-1}$, and non-negative, real eigenvalues. In the case of a connected graph, the eigenvalues of **L** are guaranteed to be arranged in ascending order : $0 = \lambda_0 \leq \lambda_1 \leq \ldots \leq \lambda_{N-1}$. The eigenvectors of **L** form a Fourier basis for the given graph, and the eigenvalues correspond to different frequency components. Consequently, one can conveniently take the Fourier transform of a graph signal by merely computing the inner product between the signal and the Fourier basis vectors as:

$$\mathcal{GF}[\mathbf{x}](\lambda_l) = \hat{\mathbf{x}}(\lambda_l) = \langle \mathbf{x}, \mathbf{e}_l \rangle = \sum_{i=0}^{N-1} x(\mathrm{i})\, \mathrm{e}_l^*(\mathrm{i}) \;, \tag{3.2}$$

where $\hat{\mathbf{x}}(\lambda_l)$ is the GFT of the **x**. To provide a comprehensive definition, Inverse GFT can be expressed as:

$$\mathbf{x}(\mathrm{i}) = \sum_{i=0}^{N-1} \hat{\boldsymbol{x}}(\lambda_l)\mathrm{e_l}(\mathrm{i}). \tag{3.3}$$

4. DATASETS

The limited availability of suitable RADAR sensors and the substantial effort required for accurate labelling have contributed to the relatively low visibility of automotive RADAR sets until recently. While popular automotive datasets like KITTI include LiDAR data alongside camera sensors, significant advancements have been made in Machine Learning-based RADAR perception, particularly in classification, object detection, and tracking [35]. Although the number of publicly available RADAR datasets has increased, many of them have limitations such as small sizes, specific sensor requirements, or other restrictions. Several open RADAR datasets, including Astyx [36], Oxford Radar RobotCar [37], Zendar [38], MulRan [39], RADIATE [40], CARRADA [41], NLOS-Radar [42], and the CRUW data set [43] have emerged, but not all of them are applicable to every problem due to various reasons. The Zendar dataset employs synthetic aperture RADAR to enhance the sensitivity by gathering multiple measurements from various positions of the self-driving vehicle. Although the advantages of this technique are demonstrated in the linked article, uncertainties regarding its overall reliability in autonomous driving scenarios remain. Furthermore, it's worth noting that SAR primarily addresses static environments, whereas the dataset in question emphasizes moving objects. In the Astyx dataset, they utilize a sensor with significantly greater inherent detail. Yet, like Zendar [38], this dataset predominantly comprises labeled vehicles, and the accuracy of the bounding box dimensions is influenced by skewed effects. Additionally, there's a scarcity of scenarios, and the sporadic measurements hinder the application of algorithms that consider temporal details. A comparable RADAR setup to Astyx's is employed in the NLOSRadar dataset [42]. However, this dataset only covers pedestrians and cyclists within highly distinct non-line-of-sight situations. The Oxford Radar RobotCar, MulRan, and RADIATE datasets take a distinct strategy. They employ a rotating RADAR, typically found on ships, to generate RADAR images that are more intricate and simpler to understand due to their increased density. Finally, it's worth noting that CARRADA and CRUW also rely on traditional automotive radar setups. However, they provide annotations solely for radar spectra, which are essentially 2D representations of the radar data cube. Regrettably, these datasets do not encompass complete point cloud details, and in the case of the CRUW dataset, labels are present for only a limited portion of all the sequences. The available comparison of publicly accessible RADAR datasets can be found in Table 4.1. Here, we introduce two datasets: the first dataset, named "nuScenes," encompasses a comprehensive sensor configuration comprising cameras, LiDARs, and RADARs in a compact setup. The second dataset, referred to as "RadarScenes," offers extensive measurements and point-wise annotations gathered over a duration of more than four hours of driving. In Table 4.1, we provide a set of characteristics for the purpose of contrasting the mentioned datasets. These characteristics encompass aspects such as dimensions, alternative modalities, sequential information involving Doppler measurements, types of object categories, equitable distribution of classes, annotations pertaining to objects, and diverse situational contexts. As you can see from the table, the nuScenes and RadarScenes datasets share the characteristic of containing Doppler data, encompassing velocity information. This results in these datasets being larger in size compared to others. An additional shared significant aspect is the presence of sequential data in both sets. This implies a sequential extraction of information from recorded traces, allowing data accumulation as required. While the RadarScenes dataset employs point-wise object annotation, the nuScenes dataset utilizes 3D box annotation. However, it's worth noting that this variance necessitates some post-processing efforts. Another noteworthy common feature is the incorporation of diverse scenes and environmental data, contributing to an augmented potential for achieving successful outcomes. A disadvantage common to both is the presence of an uneven distribution of classes, a concern addressed in the subsequent sections.

Table 4.1 The comparison of available RADAR datasets [44], C: Camera, L: LiDAR.

Data Set	Size	Other Modalities	Sequential Data	Includes Doppler	Object Categories	Class Balancing	Object Annotations	Varying Scenarios
Oxford Radar RobotCar	+++	C,L	Yes	No	0	N/A	No	+
MulRan	++	L	Yes	No	0	N/A	No	+
RADIATE	+++	C,L	Yes	No	8	+	2D Boxes	++
Zendar	+++	C,L	Yes	Yes	1	N/A	2D Boxes (2.5% Manual Labels)	++
Astyx	+	C,L	No	Yes	7	-	3D Boxes	-
NLOS-Radar	+	L	Yes	Yes	2	++	Point-Wise	-
CARRADA	+	C	Yes	Yes	3	++	Spectral Annotations	-
CRUW	+++	C	Yes	No	3	+	Spectral Boxes (Only 19%)	++
nuScenes [45]	+++	C,L	Yes	Yes	23	+	3D Boxes	++
RadarScenes [44]	+++	C	Yes	Yes	11	+	Point-Wise	++

4.1. RadarScenes Dataset

The RadarScenes dataset is a valuable resource in the field of RADAR sensing and perception [44]. This dataset provides a comprehensive collection of annotated point-wise 2D RADAR point cloud detections, offering valuable insights into the detection and localization of objects using RADAR technology.

The RadarScenes dataset was collected using an experimental vehicle with four 77 GHz automotive RADAR sensors. These sensors offer a wide field of view of ±60° shown in Figure 4.1, enabling the detection and tracking of objects in the surrounding environment.

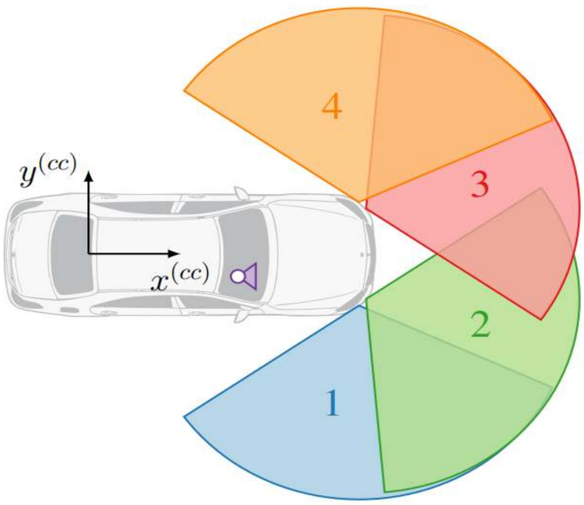

Figure 4.1 Field of View (FOV) for four RADAR settlements [44].

In addition to the 2D information, the dataset includes essential measurements such as the RCS and compensated velocity of the detected points. These measurements provide detailed information about the detected objects' properties and motion, enhancing the understanding and analysis of RADAR-based perception tasks.

Derived from over four hours of driving recordings, the RadarScenes dataset is a recently introduced automotive RADAR point cloud dataset that comprises 158 unique sequences of measurements and 2D point-wise annotations. The RadarScenes dataset encompasses point cloud annotations for eleven distinct categories as given in Appendix A and encompasses a substantial collection of recordings from diverse weather conditions, traffic scenarios, and days. This dataset comprises 118.9 million RADAR points captured along 100.1 kilometres of routes. The data collection process encompassed a range of scenarios with varying durations, 13 seconds to 4 minutes, resulting in a collection time of 4.3 hours. In this thesis, the dataset includes annotations for five object classes: Vehicle, Pedestrian, Truck, Bike, and Pedestrian-Group. Any remaining points that do not fall into these classes are initially labelled as Static. However, to align with the nuScenes dataset, the Static class label is renamed as None, without any technical impact. The distribution of total number RADAR reflections for both RadarScenes and nuScenes datasets is shown in Table 4.2. As can be seen, the number of None class element is relatively more than the others.

The RadarScenes dataset serves as a valuable resource for developing and evaluating algorithms and models related to RADAR perception, including object detection, tracking, and classification. It facilitates research and advancements in areas such as autonomous driving, ADAS, and environmental sensing. Figure 4.2 displays an aerial perspective and its corresponding visual representation. In Figure 4.2b, a corner is formed by the intersection of two streets. As you can see, there are pedestrians, vehicles and cyclists. They are all detected by human specialist by the aid of image processing. Then, Figure 4.2b depicts the point cloud produced by the RADAR system installed on the self-driving vehicle. This image of the point cloud is captured using the tools provided by the dataset provider. It is obvious that we can see the accumulated points on the targets with their labels.

Figure 4.2a The example bird eye’s view point cloud data

Figure 4.2b Corresponding camera image for point cloud scene above.

Figure 4.2 The example scene from RadarScenes dataset [44].

4.2. nuScenes Dataset

The nuScenes dataset has emerged as a significant resource in the field of autonomous driving and perception systems. Developed by nuTonomy [45], the nuScenes dataset provides a comprehensive and diverse collection of sensor data captured from real-world urban driving environments. This dataset is designed to address the challenges of perception, prediction, and planning in autonomous driving.

The dataset was gathered during driving sessions in both Singapore and Boston, amounting to a total of 5.5 hours of driving time. The dataset comprises 1000 scenes, each lasting for 20 seconds, encompassing a variety of sensor data including RADAR, LiDAR, camera, and global positioning information. Additionally, the dataset includes details about the motion of the ego vehicle. The sensor setup employed in the nuScenes data collection platform, which captures the complete range of sensor data, is visualized in Figure 4.3.

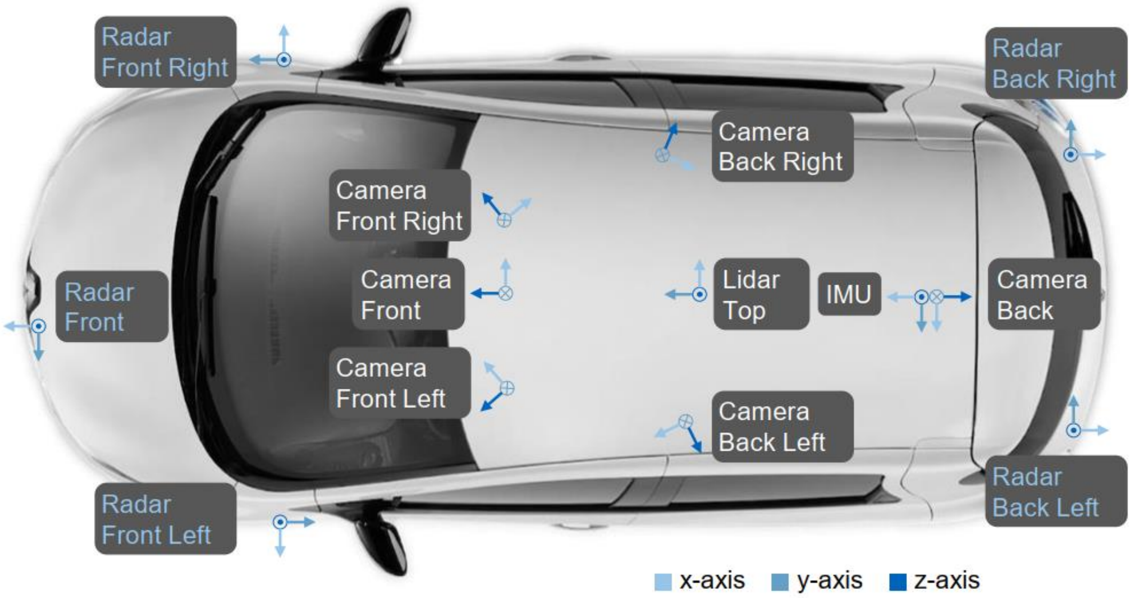

Figure 4.3 The configuration of sensors in the nuScenes data collection platform is organized in accordance with [45].

Apart from the sensor data, the nuScenes dataset includes annotated data for all 34,149 key frames. These annotations provide ground truth information in the form of three-dimensional bounding boxes, which are defined by their centre coordinates, rotation quaternion, and spatial extension. The annotations also provide additional details such as object visibility and category labels, which categorize the objects into one of 23 different categories as given in Appendix A. These annotations greatly enhance the spatial understanding and classification of objects within the dataset. The sample visualization of dataset output can be seen in Figure 4.4. Rather than RadarScenes dataset, nuScenes dataset includes annotations in the form of three-dimensional bounding boxes that encompass the actual size of an object. Each bounding box has LiDAR and RADAR points as well as camera images. By post-processing, RADAR points are extracted by labelled bounding boxes.

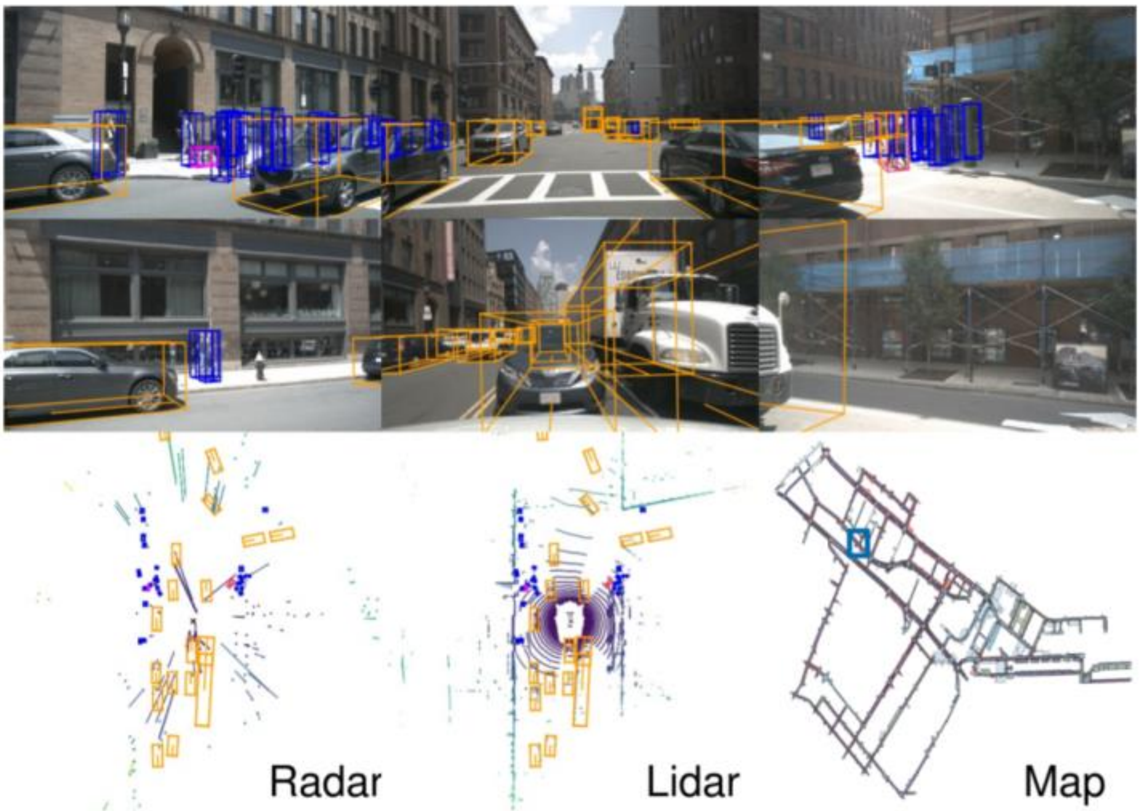

Figure 4.4 The example bird eye's view and corresponding image for nuScenes dataset [45].

In this thesis, the 23 classes have been grouped into four class compositions referred to as Vehicle, Pedestrian, Cycle, and None. Any points that do not belong to a specific annotation are classified as background and labeled under the None class. Similarly, Table 4.2 illustrates the comparison of the annotation quantities. The dataset distribution of both datasets clearly shows an imbalance. In particular, the None class is notably overrepresented in both datasets. Although the number of vehicle class instances is comparable at 1954700 and 1238849, there is a substantial discrepancy in the counts for the pedestrian and cycle classes.

The provided distributions exhibit significant class imbalances, leading to two impacts on the training of the model. Initially, the effectiveness of the model on the less common categories is diminished due to the limited data accessible [46]. While the precise data volume necessary for effectively training a model remains a subject of ongoing research in statistical power analysis, data trends demonstrate that improved outcomes are typically associated with larger datasets. Next, the uneven distribution of classes directly impacts how the model is trained, as the loss value calculation is more influenced by instances from the dominant class. Consequently, conventional machine learning methods tend to exhibit bias toward the majority class. To address this concern, a weighted loss function based on the focal loss function [60] is employed. Despite this effort, the issue of class imbalances remains a significant hurdle during the process of fine-tuning hyperparameters.

Table 4.2 Total number of annotations for nuScenes and RadarScenes datasets.

	Vehicle	**Pedestrian**	**Ped. Group**	**Truck**	**Cycle/Bike**	**None**
nuScenes	1954700	70300	-	-	23400	14601100
RadarScenes	1238849	315268	739690	603476	114153	97714654

5. AUTOMOTIVE RADAR VULNERABLE ROAD USER (VRU) CLASSIFICATION

Vulnerable road user (VRU) classification is a crucial and rapidly advancing field within the realm of computer vision and transportation safety. With the ever-increasing prevalence of vehicles on roads, pedestrian safety has become a paramount concern. Vulnerable road users, which include pedestrians, cyclists, and motorcyclists, are particularly exposed to the risks associated with traffic accidents. To enhance road safety and reduce the number of accidents involving these users, advanced computer vision techniques are being harnessed to develop intelligent systems capable of accurately classifying and identifying vulnerable road users in real-time.

Here, we concentrate on the specific problem of mmWave RADAR point cloud classification for VRUs, aiming to leverage the capabilities of deep learning technique. The classification of VRUs in RADAR point clouds is a challenging task. Traditional methods mainly try to find the fine-grained features and spatial relationships among the data points, limiting their accuracy and robustness.

To overcome these challenges, we investigated the realm of deep learning and machine learning extensively to seek effective solutions. By employing advanced neural network architectures and sophisticated learning algorithms, we aim to harness the power of mmWave RADAR point clouds for accurate and reliable VRU classification. This approach offers a promising opportunity to significantly improve the safety of VRUs on the road, by enabling ADAS to better detect and respond to potential collision risks.

Throughout this chapter, we present the design and implementation of our proposed deep learning models for automotive RADAR VRU classification. In Section 5.6, we propose two novel approaches which we named as RADAR-DGCNN and Radar-PointNGCNN, both falling under the category of graph-based methods. Additionally, we developed a novel hybrid method which incorporates the recently popular “Transformer techniques” in Section 5.7. Lastly, we present clustering-based approaches integrated within the graph framework, along with the corresponding baseline methods in this context. We extensively evaluate the performance of these methods by using real-world RADAR data collected from diverse driving scenarios and road environments.

5.1. Pre-processing

Prior to utilizing graph signal processing for automotive RADAR point cloud object classification, it is imperative to conduct pre-processing procedures to ensure the quality of the data and enable effective analysis. To begin with, the measurements obtained from the RADAR sensor suite are transformed into a unified coordinate system centered on the measurement vehicle for both datasets. This alignment allows the information captured by different sensors to collectively contribute to the representation of a RADAR point.

Prior to deploying our models, it is crucial to generate the necessary data from raw datasets to ensure fair comparisons with previous outcomes. As we inspired from [47] , the inherent sparsity of RADAR point clouds, characterized by fewer data points, was addressed by aggregating points within time windows of T=500 ms. This allowed the point cloud to increase in density, enabling the consideration of more reflections per object. In this thesis, we designed RadarScenes dataset's methods are designed by using up to 3072 RADAR points, whereas the nuScenes dataset is constrained to only 625 points. If the 500 ms long time window yields more than 3072 reflections, reflections belonging to the static class are eliminated. Conversely, if the number of measured reflections is less than 3072, one reflection is repeatedly sampled to meet the required amount. Such an approximation has been carried out in nuScenes dataset with T=500 ms.

5.2. Data augmentation

Throughout the development process, two primary challenges encountered were the sparsity of the provided RADAR point cloud data and the class imbalance. To address these issues and enhance the existing data, data augmentation methods were employed. It is worth noting that while implementing all baseline methods, their original design and data processing techniques were preserved. Gaussian noise with $\mathcal{N}(0,0.3)$ was introduced to all feature dimensions, with additional scaling using a random factor $\alpha \sim unif_1(0.8,1.25)$ within the range [0.8-1.25]. In this thesis, we only apply into point cloud of both datasets for point-wise addition and scaling. For this reason, the number of data in datasets does not change.

The class imbalance issue also had a direct impact on the model training process, as the majority classes had a higher frequency in the computation of the loss value. To tackle this problem, class weights were introduced to all classes in each dataset to penalize classes with more samples compared to those with fewer samples. These class weights were applied to the loss value, scaling it with a vector of class weights $w \in \mathbb{R}^k$. The vector of class weights was computed based on the underlying class distribution of the datasets, using a general heuristic definition introduced by Dai et al. [48]. As a result, the weight of the i^{th} class was determined as given in Eq. 5.1. below:

$$w_i = 1/\log(1.2 + \mathrm{h}_\mathrm{i}'), \tag{5.1}$$

where $\mathrm{h}_\mathrm{i}' = \mathrm{h}_\mathrm{i} / ||\mathrm{h}_\mathrm{i}||$ and h_i is the histogram of class labels. The allocation of class weights is decided by considering the inherent distribution of classes within the used dataset. This helps counteract the impact of imbalanced datasets during the training of the model. As explained in Section 7, we could use the feature selection algorithms to choose the optimum features in both datasets. In this case, it may help to the accuracy while decreasing the imbalance in addition to increase the computational cost in pre-processing.

5.3. Feature extraction from graphs

Feature extraction is a crucial step in our proposed methods that aim to transform the raw graph data into informative and discriminative feature representations. In addition to normal RADAR point cloud features, we employ GSP techniques to extract informative features from the constructed graph representation. What sets it apart from other research is the unique approach we employ in this thesis [49].

5.3.1. Initialization of graphs

In graph-based methods, the first step in many applications involves initializing the graph and constructing its structure. Graph initialization involves defining the set of nodes (vertices) and the connections between them (edges) based on the data or problem at hand. In this thesis, we create a graph with respect to the spatial coordinates of points in point cloud. This step involves capturing the inherent patterns and dependencies present in the graph structure. First, we establish the definition of an *N*-point point cloud, represented as a set $\mathbf{P} = \{\mathbf{p}_1, \mathbf{p}_2, \dots \mathbf{p}_N\}$. Each point $\mathbf{p}_i = (\mathbf{c}_i, \mathbf{s}_i)$ within the cloud comprises a 3-D Cartesian coordinate vector $\mathbf{c}_i \in \mathbb{R}^3$, where $\mathbf{c}_i = (\mathbf{x}_i, \mathbf{y}_i, \mathbf{z}_i)$, and a state vector $\mathbf{s}_i \in \mathbb{R}^k$, with k denoting the dimension of additional RADAR point characteristics. These characteristics could include various features naturally produced by RADAR, such as RCS and velocity, as well as newly generated attributes, like the number of edges connected to each node. The inclusion of a more comprehensive feature vector positively impacts the model's performance, enhancing its ability to capture and interpret essential information from the point cloud data.

5.3.2. Building feature vectors

To convert the point cloud challenge into a graph-oriented task, we initiate the process by establishing a weighted and undirected graph represented as $\mathcal{G}(\mathcal{V}, \mathcal{E}, \mathbf{W})$. Given a point cloud **P**, we construct the fully connected graph with the following:

$$\mathcal{E} = \{(i,j) | \, d(i,j) < r\}, \tag{5.2}$$

where *r* is 1 meter in our letter. In addition, we employ the Gaussian function as a weight function to establish the connection strength between vertex *i* and vertex *j*, represented by the Eq. (5.3).

$$\mathbf{W}_{\mathrm{ij}} = \begin{cases} \exp\left({-d(i,j)^2}/{2\sigma^2} \right), & \text{when } (\mathrm{i},\mathrm{j}) \in \mathcal{E}, \\ 0, & \text{otherwise.} \end{cases} \tag{5.3}$$

where the weight $\mathbf{W}_{\mathrm{ij}}$ connecting vertex *i* and vertex *j* is non-zero only when there is an edge between these vertices, as dictated by the graph topology. Here, the $d(i,j)$ is the distance between point *i* and point *j* .

Additionally, we leverage graph Fourier transforms for the examination of the graph signal's frequency characteristics. GFT serves as a representation of the signal in the spectral domain, allowing us to scrutinize its frequency components and extract pertinent features as discussed in Section 3.1. Through the analysis of the spectrum of the graph signal, we can discern significant patterns and structural attributes within the data.

Since we are focusing on classifying objects individually based on their reflection points, the labeled points in the mmWave RADAR point cloud represent the RADAR reflection points. To gain a deeper understanding of these labelled points, we utilize a feature vector, $\mathbf{F}_{e,i} \in \mathbb{R}^8$, which contains relevant features. We introduce our proposed feature vector, $\mathbf{F}_{e,i}$ new, which is presented as follows:

$$\mathbf{F}_{e,i} = \{c_i, \sigma_i, v_i, deg(\mathbf{p}_i), \mathcal{GF}[\mathbf{x}](\sigma_i), \mathcal{GF}[\mathbf{x}](v_i)\}, \tag{5.4}$$

where c_i represents the spatial coordinates, σ_i represents the Radar Cross Section (RCS), v_i represents the velocity measured by the sensor, $deg(\mathbf{p}_i)$ denotes the number of edges connected to $\mathbf{p}_i$ within a radius of 1 meter. Additionally, $\mathcal{GF}[\mathbf{x}](\sigma_i)$ refers to the GFT of the RCS, while $\mathcal{GF}[\mathbf{x}](v_i)$ represents the GFT of the velocity. The purpose of the feature $deg(\mathbf{p}_i)$ is to determine the number of connections within the local neighborhood. Through the utilization of this method, we can retrieve data on the quantity of connections within a one-meter radius, consequently enabling the creation of a potential object network. As an example, consider pedestrians and trucks. The quantity of connections within a 1-meter radius will vary significantly between pedestrians and trucks. Pedestrians have fewer connected points, whereas trucks have a considerably larger number of connections to each point, primarily due to their larger size. Regarding $\mathcal{GF}[\mathbf{x}](\sigma_i)$,$\mathcal{GF}[\mathbf{x}](v_i)$ features, we assume every RCS and velocity values as a signal on a created graph. The process of generating graphs varies from case to case. In each instance, the graphs are uniquely formed based on the spatial arrangement of points within the point cloud. Through the extraction of features from the raw RADAR point cloud, we are able to transform the input point cloud data into a higher-dimensional representation denoted as $\mathbf{F}_{e,i}$.

To illustrate, Figure 5.1 presents a frame sample from the RadarScenes dataset, demonstrating the graph frequency components $\mathcal{GF}[\mathbf{x}](\sigma_i)$ and $\mathcal{GF}[\mathbf{x}](v_i)$. Notably, in Figure 5a, high gain values are primarily observed at low graph frequencies, while in Figure 5b, high gain values are apparent at certain high graph frequencies. Evidently, specific graph frequency components exhibit distinct features and significantly contribute to the characterization of class labels.

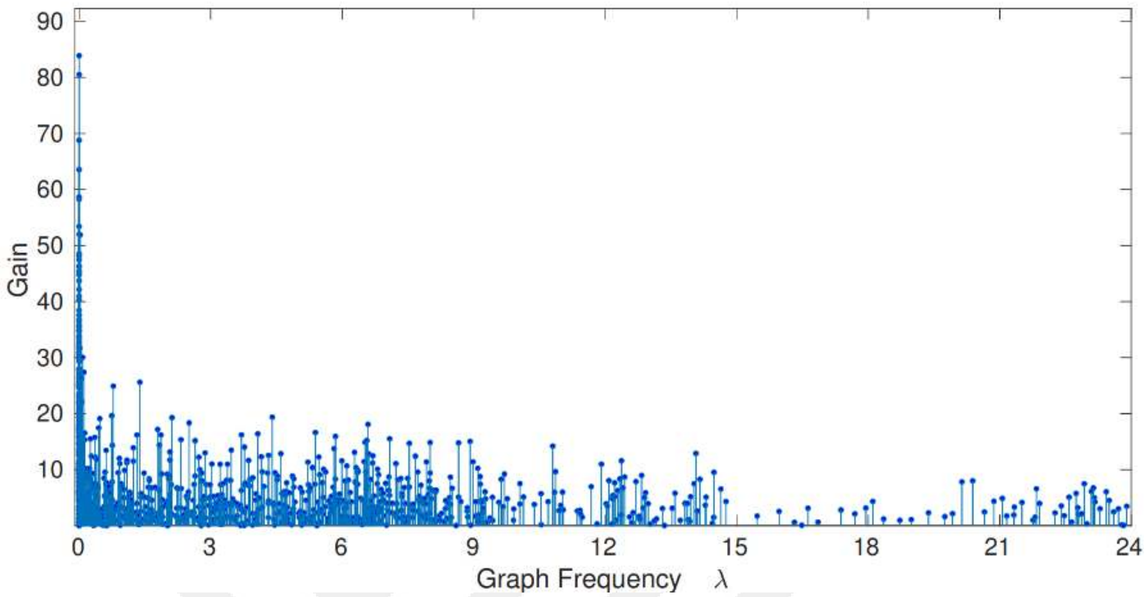

Figure 5.1.a GFT of σ_i.

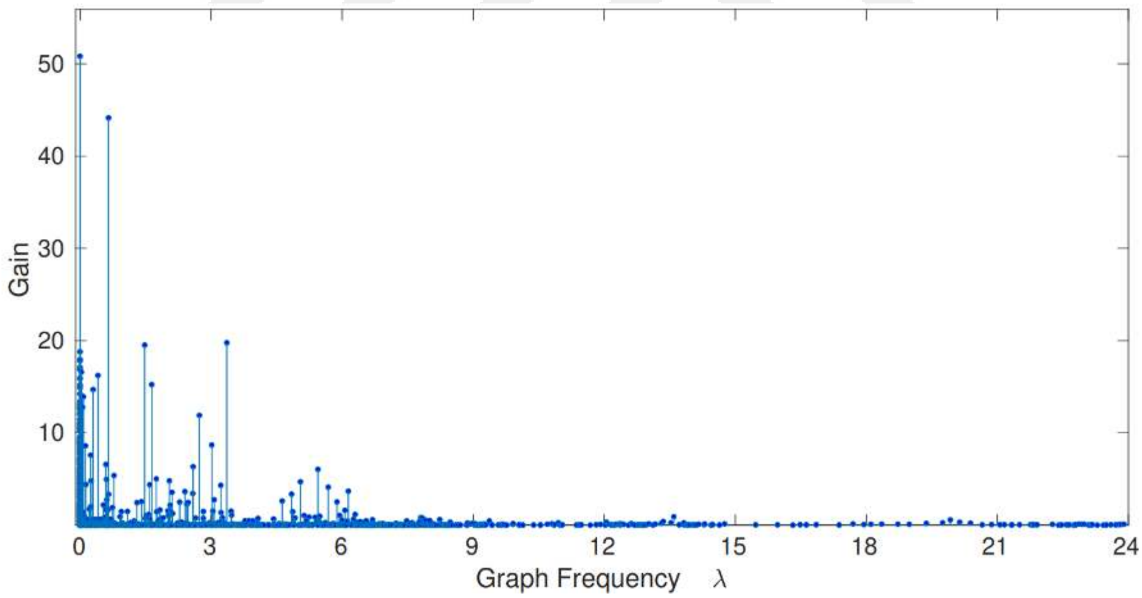

Figure 5.1.b GFT of v_i.

Figure 5.1 Illustrative examples of features $\mathcal{GF}[\mathbf{x}](\sigma_i)$ and $\mathcal{GF}[\mathbf{x}](v_i)$ are presented.

5.4. Loss Function

In the context of classification, the choice of an appropriate loss function plays crucial role in the gradient-based optimization process, where the objective is to discover the most optimal set of model parameters that minimizes the loss value [50]. The optimization process, known as model training, involves assessing the model's predictions y' in comparison to the target values y using the loss function. The aim of this training is to optimize the model. Here, the loss function can be considered as the objective function for this optimization process [51].

In order to accomplish this objective, the selection of the loss function must take into account both the underlying data distribution and the specific project goals. For this purpose, the Cross Entropy loss function can be extended to handle imbalanced datasets by incorporating class weights and multi-class classification challenges during model training. The objective is to penalize misclassifications more heavily for underrepresented classes, effectively mitigating the impact of class imbalance on the training process. The Cross Entropy loss function for multi-label classification can be defined as in Eq. (5.5).

$$\mathcal{L}\,(y_i, \widehat{y_i}) = -\sum_{c=1}^{C} w_c\, y_i^c \,\log \widehat{y_i}^{\,c}\,, \tag{5.5}$$

where y_i is the ground truth binary label vector for the i-th sample, with $y_i^c = 1$ if the sample belongs to class c, and $y_i^c = 0$ otherwise. $\widehat{y_i}^{\,c}$ is the predicted probability vector for the i-th sample, with $\widehat{y_i}^{\,c}$denoting the predicted probability of the i-th sample of class c. C is the number of classes in the dataset. w_c is the weight belonging to class c. The class weights w_c are calculated based on the idea stated in Section 5.2. The modified Cross Entropy loss function with class weights provides a robust solution for multi-label classification tasks with imbalanced datasets, facilitating more effective learning and better model performance in real-world scenarios.

5.5. Performance Metrics

In object classification, the validation of the performance of classification models is essential to assess their effectiveness and robustness. Several performance metrics are commonly used to quantify the classification accuracy and provide valuable insights into the model's behavior such as Accuracy, Recall, F_1 score. The objective of the performance metrics introduced in this section is to quantify the degree of resemblance between the set of predicted objects generated by the model and the set of annotated objects present in the dataset.

Precision:

Precision measures the ratio of correctly predicted positive observations to the total predicted positives. In other words, it focuses on how many of the predicted positive instances were actually correct.

Recall (Sensitivity or True Positive Rate):

Recall calculates the ratio of correctly predicted positive observations to all actual positive instances. It emphasizes the model's ability to correctly identify positive cases.

F_1- Score:

The F_1 score is the harmonic mean of precision and recall. It is used to balance the trade-off between precision and recall, providing a single value that considers both aspects. The F_1 score is particularly useful when dealing with imbalanced datasets, where one class might dominate the other.

The choice of the evaluation metric should be tailored to both the problem and the the dataset characteristics. In the case of both RadarScenes and nuScenes datasets, significant class imbalances are evident, as depicted in Table 4.2. Consequently, evaluation metrics like accuracy are unsuitable for this particular problem since they tend to favor the majority class [52]. To address this issue and provide a fair assessment, the main evaluation metric selected is the macro-averaged F_1 (F_1-Macro) score in this thesis. This metric takes into account the performance across all classes and is better suited to handle imbalanced datasets, ensuring a comprehensive evaluation of the classification model. The macro-averaged F_1 score is well-suited for assessing models trained on imbalanced datasets. The F_1 score is the harmonic mean of precision and recall and gives a balanced measure of the classifier's performance as given in Eq. (5.4)-(5.5) below:

$$F_1 = 2\,\frac{precision \times recall}{precision + recall}, \tag{5.4}$$

and

$$precision = \frac{t_p}{t_p + f_p}\,, \qquad \text{recall} = \frac{t_p}{t_p + f_{n}}, \tag{5.5}$$

where t_p, f_p, and $f_{n'}$ are the numbers of true positives, false positives, and false negatives, respectively. In multi-class classification tasks, where there are multiple classes to be predicted, the macro-average F_1 score is used to compute the overall F_1 score for the model. The macro-average F_1 score calculates F_1 score for each class independently and then takes the average across all classes. This ensures that each class is equally weighted in the evaluation, regardless of its class size. The F_1-Macro score is computed by the Eq.(5.6)

$$F_1\text{-Macro} = (\mathrm{F}_{1,1} + \mathrm{F}_{1,2} + \ldots + \mathrm{F}_{1,\mathrm{C}}) \,/\, \mathrm{C} \;, \tag{5.6}$$

where C is the number of classes in the dataset, $\mathrm{F}_{1,1} + \mathrm{F}_{1,2} + \ldots + \mathrm{F}_{1,\mathrm{C}}$ correspond F_1 scores for each individual class.

5.6. GSP Based Object Classification for Automotive Radar Point Clouds

5.6.1. Introduction

RADAR point clouds provide valuable information about the environment, enabling object detection and classification. On the other hand, the sparsity, complexity, and irregularity of RADAR data present significant obstacles for accurate and reliable object classification. RADAR is one example of devices that generate unstructured 3D point clouds through scanning the surroundings in autonomous driving scenarios. Among the vital tasks applied to RADAR-acquired point clouds is classification, playing a critical role in identifying class-labelled points, such as vehicles, pedestrians, and obstacles. In this study, we focus on tackling the classification challenge for unstructured RADAR point clouds by using graph-based algorithms within the GSP framework.

Among various application domains, automotive RADAR sensors present an ideal use case where data is naturally represented in irregular grids. The signals obtained from RADAR sensors are inherently collected and processed in irregular grids. In the context of irregular data structures, the inherent challenge lies in not having readily available information about the data domain's topology and the relationships between data points. This limitation often poses a significant obstacle when constructing graph representations [30]. To begin with, there is no definitive method to determine the best connection topology for the graph.

Additionally, the weight function and its associated parameters may hold significant importance based on the specific operations applied to the graph. Consequently, inferring the weights and, particularly, the edges of the graph becomes essential by investigating the physical characteristics of the data. Nonetheless, utilizing graph-based techniques proves highly advantageous when working directly with unstructured data. As a result, RADAR sensors are well-suited for using graph-based techniques, given their ability to handle irregular data representations effectively.

In this study, we introduce novel multi-class automotive RADAR point cloud classification techniques called RADAR-DGCNN and RADAR-PointNGCNN for VRUs by using GSP. Our methods capitalize on critical target-level information, including precise 3D locations, compensated velocity, and RCS, alongside high-level features derived through GSP. To prove the validity of our approach, we carried out experiments on two worldwide used datasets: RadarScenes (RS) and nuScenes (NS), comparing our proposed methods with state-of-the-art baseline techniques.

5.6.2. Related Work

Earlier research in the automotive context has extensively examined the classification of VRUs by using 2D or 3D point cloud data, exploring various aspects such as shape classification, object detection, and point cloud segmentation[53]. In particular, machine learning and deep learning algorithms have been prominently employed to tackle these tasks with point clouds, as they possess the capability to learn intricate hierarchical patterns and distinctive features [54], [55]. Starting from the initial raw input, every layer within these networks progressively converts the inputs into more abstract representations, optimizing them for deep learning algorithms. The use of LiDAR for VRU classification has been extensively explored, and automotive RADAR is becoming increasingly significant in this context. Notably, deep learning algorithms that were originally developed for LiDAR applications are now being adapted and applied to the domain of RADAR data analysis [56], [57], [14].

Earlier research endeavours have investigated the categorization of VRUs within the RADAR context by employing clustering techniques to group RADAR detection points and subsequently classifying these clusters according to their statistical properties [58], [59], [60]. Zhao et al. [61] proposed a clustering-based approach for point cloud classification. However, the effectiveness of their technique heavily relies on the Density-Based Spatial Clustering of Applications with Noise (DBSCAN) algorithm used for clustering moving targets. Moreover, real-life RADAR data often contains clutter, making it challenging to achieve clutter-free data. Some RADAR data models, as discussed in [62], are designed by using either low-level information, target-level information, or a combination of both. In [62], both target-level information and low-level RADAR data are utilized. However, this reliance on low-level data poses limitations on the performance of clustered-based algorithms and CFAR techniques.

A ground-breaking advancement in point-based techniques is PointNet [19], where individual point-wise features are learned independently through multiple MLP layers, and global features are extracted using a max-pooling layer. Nonetheless, this method lacks the capability to capture local information among data points. To address this, Qi et al. [20] introduced PointNet++, a hierarchical network that collects subtle geometric features from each point's neighbourhood. In the context of RADAR point cloud applications, deep learning-based methods PointNet [19] and PointNet++ [20] have been implemented in [63] and [47], respectively. These studies utilize x-y plane, Doppler velocity and RCS point cloud data instead of 3D (x, y, z) data for semantic segmentation, leading to enhanced performance on RADAR point clouds.

GSP-oriented approaches are designed to address VRU classification challenges using LiDAR point clouds [55]. For instance, Zhang et al. [64] introduced a GSP-based technique called PointGCN, which focuses on classifying 3-D spatial point cloud data. To construct a graph, an edge-weighted Gaussian kernel is employed, considering *k*-closest neighbours from the point cloud. Convolutional filters are created using Chebyshev polynomials in the graph spectral domain. To capture both global and local characteristics of the point cloud, the method employs global pooling and multi-resolution pooling techniques.

In [21] and [65], novel neural network modules named Dynamic Graph CNN (DGCNN) and Linked Dynamic Graph CNN (LDGCNN) are introduced, respectively, for learning on point clouds. These methods establish graphs in the feature space, and the graphs are dynamically updated after each network layer. Unlike PointNet++, DGCNN takes local features into account. Additionally, LDGCNN enhances the network design of DGCNN to improve performance and reduce the network's model size [66]. Another method based on LiDAR introduces PointNGCNN, an end-to-end deep learning architecture that utilizes Chebyshev polynomials to construct neighbourhood graphs [67]. By employing graph filters of Chebyshev polynomial, PointNGCNN can extract richer neighbourhood geometric information compared to the MLPs used in PointNet++ and DGCNN. Consequently, PointNGCNN effectively captures high-dimensional feature space information, showcasing its performance in extracting neighbourhood characteristics and spatial distribution information.

Recently, novel approaches have emerged in the field of graph-based RADAR point cloud classification. Drawing inspiration from the Point-GNN model originally designed for LiDAR point cloud data [17], Svenningsson et al. introduced Radar-PointGNN tailored explicitly for RADAR data [56]. This innovative method incorporates not only spatial coordinates but also additional features such as velocity, RCS, age of measurement, and point density proxy for each data point. However, despite the theoretical feasibility of object detection on low-quality RADAR data, Radar-PointGNN falls short in achieving sufficiently accurate results. Similarly, Gong et al. [57] utilized the Point-GNN model as a foundation to propose MMPoint-GNN, targeting human activity recognition. MMPoint-GNN effectively handles sparse point clouds collected by mmWave RADAR, showcasing the superior performance of GNNs in mmWave RADAR-related tasks. Moreover, mmWave RADAR data is leveraged for gait recognition in [68], where spatial and temporal graph convolutional networks are employed. Furthermore, gesture recognition is addressed in [69] by developing an approach based on a variant of the DGCNN model. These graph-based techniques represent promising avenues for enhancing the classification of RADAR point clouds, thereby contributing to the advancement of RADAR-related applications in various domains.

In conclusion, there is a limited number of deep learning-based RADAR point cloud classification algorithms in the automotive domain. The current approaches either focus solely on LiDAR point clouds or lack support for leveraging the GSP framework to effectively handle the inherent irregularity of point clouds.

5.6.3. DGCNN (Dynamic Graph CNN for Learning on Point Clouds Architecture)

DGCNN is an innovative architecture that combines PointNet and a graph CNN approach as its backbone. It adopts a point-wise neural network design for tasks like classification, part segmentation, and semantic segmentation. Our focus in this study is on the semantic segmentation branch, where the model takes an entire point set as input and predicts a semantic label for each point.

In DGCNN, point clouds are represented as dynamically constructed linked graphs in the latent space, utilizing k-NN to establish connections. Besides leveraging PointNet, DGCNN enhances the model by incorporating local features through an edge convolution technique called *EdgeConv*. This enables the processing of data based on the graph structure. The architecture also ensures the exchange and rotation invariances of its structure, enabling efficient feature extraction from point cloud data. The corresponding algorithm structure is given in Algorithm 1.

Algorithm 1 RADAR-DGCNN

Input: Raw RADAR point clouds and their class labels, $\mathbf{P} = \{\mathbf{p}_1, \mathbf{p}_2, ...\mathbf{p}_N\}$, where $\mathbf{p}_i = (\mathbf{c}_i, \mathbf{s}_i)$, $\mathbf{c}_i = (x_i, y_i, z_i)$ and $\mathbf{s}_i = (\sigma_i, v_i)$

Output: Predicted labels, F_1-Macro

Data: Training, Validation and Test Sets for RadarScenes and nuScenes Datasets

Graph construction: $G = (\mathcal{V}, \mathcal{E}, W)$

Feature extraction: $s_{i,new} = \{c_i, \sigma_i, v_i, deg(\mathbf{p}_i), \mathcal{GF}[\sigma_i], \mathcal{GF}[v_i]\}$

procedure TRAINING PROCEDURE

  $epochs \leftarrow 30$, $learningrate \leftarrow 0.0001$, $batchsize \leftarrow 4$, $momentum \leftarrow 0.9$, $m \leftarrow 3072$

  $\mathcal{N}(0, 0.3)$

  while $epochs \neq 0$ **do**

    $N \leftarrow 3$, $LS(layersizes) \leftarrow [128, 128, 64]$

    for $k \leftarrow 1$ to N **do**

      EdgeConv($\mathbf{p}_i = (c_i, s_{i,new})$)

      MLP (LS[k])

    end for

    $epochs \leftarrow epochs - 1$

  end while

end procedure

5.6.3.1. Edge Convolution (EdgeConv)

The DGCNN architecture incorporates a specialized module known as EdgeConv, designed to capture features from the local neighbourhood graph, describing the connections between a point and its neighbouring points. By constructing a graph $\mathcal{G}(\boldsymbol{\mathcal{V}},\ \boldsymbol{\mathcal{E}},\ \boldsymbol{W})$, we can represent the local structures of a given point cloud. Essentially, $\mathcal{G}$ represents the k-NN graph of points $\mathbf{P} = \{\mathbf{p}_1, \mathbf{p}_2, \dots \mathbf{p}_N\}$ in $\Re^N$, where k edge features are computed for each point b_i by evaluating the function $e_{ij} = h_{\mathbf{\Theta}}\left(\mathbf{p}_i, \mathbf{p}_j\right)$ where $\mathbf{p}_j$ is one of the neighbours of $\mathbf{p}_i$, and $h_{\mathbf{\Theta}}$ is a nonlinear function with learnable parameters $\mathbf{\Theta}$. Notably, the function $h_{\mathbf{\Theta}}\left(\mathbf{p}_{i}, \mathbf{p}_{j}\right)$ is asymmetric, using $h_{\mathbf{\Theta}}\left(\mathbf{p}_i, \mathbf{p}_j\right) = \overline{h}_{\mathbf{\Theta}}\left(\mathbf{p}_i, \mathbf{p}_j - \mathbf{p}_i\right)$ to account for both global shape structure and local neighborhood information. The EdgeConv module is then completed with a channel-wise symmetric operation applied to each vertex. The output $\mathrm{p}_i{}'$ is determined by aggregating all edge features associated with the vertex p_i as follows:

$$\mathbf{p}_i' = \max_{j:(i,j)\in\mathcal{E}} \mathrm{ReLU}\left(\mathbf{\theta_M} \cdot \left(\mathbf{p}_j - \mathbf{p}_i\right) + \boldsymbol{\phi}_{M} \cdot \mathbf{p}_i\right). \tag{5.7}$$

In EdgeConv, the parameter set Θ comprises $\mathbf{\Theta} = \{\mathbf{\theta_1}, \mathbf{\theta_2}, \dots, \mathbf{\theta_M}, \boldsymbol{\phi}_\mathbf{1}, \boldsymbol{\phi}_\mathbf{2}, \dots, \boldsymbol{\phi}_\mathbf{M}\}$, and the Euclidean inner product is denoted by $(\cdot)$. The process of graph generation and convolution in EdgeConv is depicted in Figure 5.2, involving three primary blocks: Neighbourhood Selection, Neighbourhood Graph Construction, and Neighbourhood Graph Convolution.

The Neighbourhood Selection module's primary objective is to choose k-nearest points for the central point through the application of the k-nearest neighbours (k-NN) graph algorithm. The Neighbourhood Graph Construction module adopts dynamic graph construction to create edge features, defining the connections between a point and its neighbouring points. To update the center of the feature vector, graph convolution operations are performed using Multi-Layer Perceptrons (MLPs). Figure 5.2 illustrates how these components collectively contribute to the EdgeConv process.

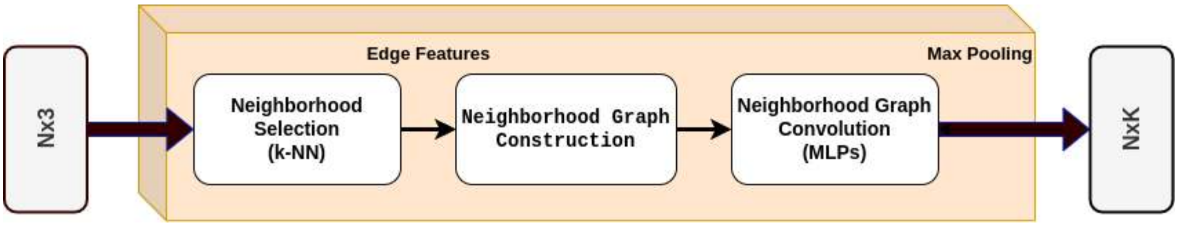

Figure 5.2 Structure of EdgeConv

5.6.4. PointNGCNN (Deep convolutional networks on 3D point clouds with neighborhood graph filters)

PointNGCNN [67] is an end-to-end deep learning network with applications in classification, part segmentation, and semantic segmentation. Focusing solely on semantic segmentation, PointNGCNN utilizes graph convolutions to leverage local geometric structures. The network introduces a novel architecture based on neighbourhood graph filters, enabling direct processing of point clouds. The process begins by selecting the k-NN around each center, which are then transformed into the local coordinates of the center. PointNGCNN employs the PointSIFT algorithm to address potential issues arising from the k-NN method used in DGCNN, ensuring that neighbourhood points are effectively captured in eight orientations. By treating the points' features as graph signals, the Laplacian Matrix L is derived. Notably, PointNGCNN improves upon MLPs by defining a graph filter using the Chebyshev polynomial approximation, allowing for the extraction of additional geometric information from the neighbourhood while minimizing the complexity of graph convolution. After applying graph filters, the Laplacian and feature matrices are combined within the model to obtain the features of each center. The algorithm for RADAR-PointNGCNN is elaborated in detail in Algorithm 2.

Algorithm 2 RADAR-PointNGCNN

Input: Raw RADAR point clouds and their class labels, $\mathbf{P} = \{\mathbf{p}_1, \mathbf{p}_2, ...\mathbf{p}_N\}$, where $\mathbf{p}_i = (\mathbf{c}_i, \mathbf{s}_i)$, $\mathbf{c}_i = (x_i, y_i, z_i)$ and $\mathbf{s}_i = (\sigma_i, v_i)$

Output: Predicted labels, F_1-Macro

Data: Training, Validation, and Test Sets for RadarScenes and nuScenes Datasets

Graph construction: $G = (\mathcal{V}, \mathcal{E}, W)$
Feature extraction: $s_{i,new} = \{c_i, \sigma_i, v_i, deg(\mathbf{p}_i), \mathcal{GF}[\sigma_i], \mathcal{GF}[v_i]\}$
procedure TRAINING PROCEDURE
  $epochs \leftarrow 30$, $learningrate \leftarrow 0.0001$, $batchsize \leftarrow 4$, $momentum \leftarrow 0.9$, $m \leftarrow 625$
  $\mathcal{N}(0, 0.3)$
  while $epochs \neq 0$ **do**
    $N \leftarrow 3$, $LS(layersizes) \leftarrow [64, 64, 128]$
    for $k \leftarrow 1$ to N **do**
      FeatureFusionBlock($\mathbf{p}_i = (c_i, s_{i,new})$)
      MLP (LS[k])
    end for
    $epochs \leftarrow epochs - 1$
  end while
end procedure

5.6.4.1. Feature Fusion Block

Similarly, just as DGCNN, PointNGCNN employs the graph convolution operation to capture local neighbourhood details. PointNGCNN utilizes Chebyshev polynomials to approximate spectral filtering on graph signals, represented by the spectral filtering expression on the graph signal $\mathbf{u}'$defined by Eq.(5.8).

$$\mathbf{u}' = \mathbf{U}\boldsymbol{\Theta}'\mathbf{U}^T\mathbf{u}, \tag{5.8}$$

where $\mathbf{u}'$ is the output signal, $\boldsymbol{\Theta}' = \boldsymbol{\Theta}'(\boldsymbol{\Lambda})$ is the diagonal matrix of the learnable filter, and Λ is the eigenvector of $\mathbf{L}$. The output $\mathbf{u}'$ is determined by the learnable diagonal matrix $\boldsymbol{\Theta}$', which corresponds to the filter with $\boldsymbol{\Lambda}$ representing the eigenvector of L. Defferrard et al. introduced the utilization of the Chebyshev polynomial filter in the following manner:

$$\boldsymbol{\Theta}'(\Lambda) = \sum_{k=0}^{K-1} \varphi_k \,\Theta^k = \sum_{k=0}^{K-1} \varphi_k \,\mathrm{T_k}(\overline{\boldsymbol{\Lambda}}), \tag{5.9}$$

where $\varphi_0, \ldots, \varphi_{K-1}$ are learnable parameters, K is the order of the polynomials, I is the degree matrix, $\overline{\mathbf{\Lambda}} = 2\mathbf{\Lambda}/\lambda_{max} - \mathbf{I}$ is the re-adjusted eigenvalue, λ_{max} is the largest eigenvalue, and $\mathrm{T_k}(\overline{x})$ is the Chebyshev polynomial of order k. As polynomials of the Laplacian function similarly to polynomials of its eigenvectors, the Eq. 5.8 can be expressed as follows:

$$\mathbf{u}' = \mathbf{U}\Theta'(\mathbf{\Lambda})\mathbf{U}^T = \sum_{k=0}^{K-1} \varphi_k \, \mathbf{U}\mathrm{T_k}(\overline{\mathbf{\Lambda}})\mathbf{U}^{\mathbf{T}}\mathbf{u} = \sum_{k=0}^{K-1} \varphi_k \, \mathrm{T_k}(\overline{\mathbf{L}})\mathbf{u}, \tag{5.10}$$

where $\overline{\mathbf{L}} = 2\mathbf{L}/\lambda_{max} - \mathbf{I}$. Then, one can obtain the output y as follows:

$$\mathbf{y} = \mathrm{ReLU}(\mathbf{u}'\mathbf{\Omega} + \mathbf{b}), \tag{5.11}$$

where $\mathbf{\Omega}$ is the weight parameter matrix, $\mathbf{b}$ is the bias, $\mathbf{y}$ is the last output, and ReLU (Rectified Linear Unit) is an activation function.

Figure 5.3 illustrates the complete design of the Feature Fusion Block. The process is depicted similarly to Figure 1, with two notable distinctions from the EdgeConv approach. Firstly, it employs a different method for neighbourhood selection known as PointSIFT [70]. Secondly, it incorporates a graph filter over the neighbourhood using Chebyshev polynomial approximation.

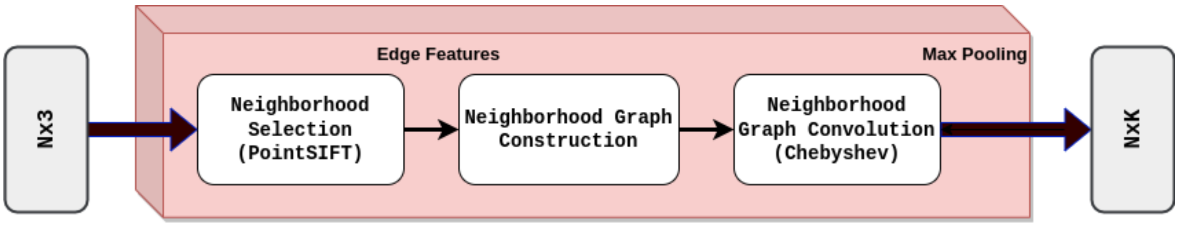

Figure 5.3 Structure of Feature Fusion Block.

5.6.5. Model Architecture

Our proposed method draws inspiration from PointNGCNN [67], DGCNN [21], and convolution operations. These techniques share similarities, aiming to enhance the PointNet++ [20] model. However, instead of utilizing LiDAR point cloud data solely based on spatial coordinates, we leverage RADAR point cloud data. Additionally, we incorporate specific feature vectors derived from additional RADAR properties and GSP methodology.

Our proposed approaches, RADAR-DGCNN, and RADAR-PointNGCNN share similar structures inspired by original DGCNN and PointNGCNN methods overall. To demonstrate the improved performance achieved with the new feature vector, we applied both graph-based methods to RADAR point cloud data. As a result, we have illustrated main components of the proposed methods with a concise diagram that highlights their distinctions in Figure 5.4. Both of our introduced approaches employ a feature extraction module, consisting of MLPs and related graph convolution blocks. This shared feature extraction module operates on the RADAR outputs, extracting higher-dimensional features based on GSP principles. After the feature extraction step, the RADAR-DGCNN method incorporates three EdgeConv and MLP modules, while the RADAR-PointNGCNN method adopts three Feature Fusion Block and MLP modules. In both methods, the point features are accumulated globally, resulting in the creation of a 1D global descriptor. Consequently, classification scores for semantic labels of each point are produced.

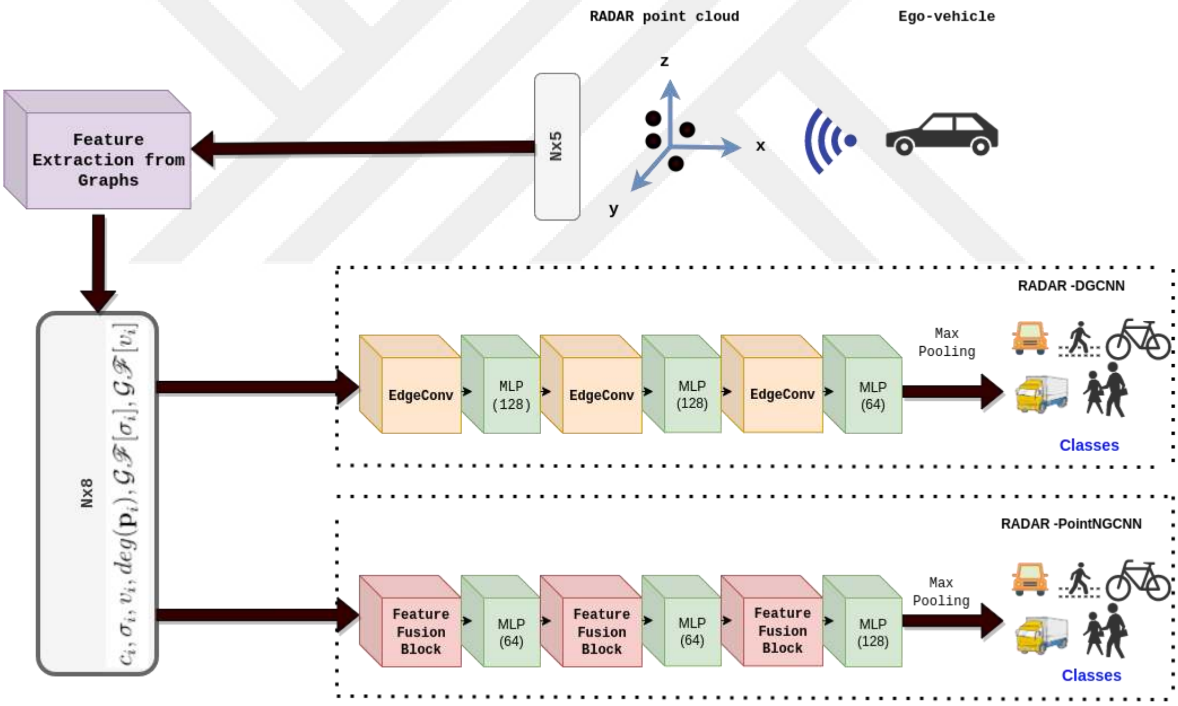

Figure 5.4 The primary structures of the proposed approaches.

5.6.6. Experiments

5.6.6.1. Implementation Details

The convolution blocks in DGCNN and PointNGCNN utilize Shared-MLP layers with sizes (128, 128, 64) and (64, 64, 128) respectively. These MLP layers' sizes are obtained by inference from original studies. Following the convolutional layers, global max/sum pooling is applied, followed by two fully connected layers with sizes (512, 256). Additionally, we employed the specific feature vector described in the study [45] to implement its respective baseline. Random hyper-parameter settings were explored, including learning rates of 0.0001, a batch size of 4, and momentum of 0.9. Training was performed for 30 epochs on a Linux laptop equipped with an Nvidia GeForce GTX 3060 GPU.

5.6.6.2. Results

In this chapter, we propose two novel approaches which we called RADAR-DGCNN and RADAR-PointNGCNN, specifically tailored for RADAR point clouds. To evaluate their performance, we compare our proposed methodology against various state-of-the-art and baseline methods. The baselines encompass Schumann et al. [47], Zhao et al. [61], DGCNN [21], PointNGCNN [67], and Radar-PointGNN [56]. Schumann et al. [47] and Zhao et al. [61] present methods applied in the conventional mmWave RADAR domain, with Schumann et al. [47] implemented solely on the RadarScenes dataset, and Zhao et al. [61] lacking specific dataset information. DGCNN [21] and PointNGCNN [67] are GSP-based techniques used for LiDAR data; however, they have not been directly applied to RADAR point clouds. Radar-PointGNN [56], on the other hand, is a GSP-based method specifically designed for RADAR point clouds but has only been reported on the nuScenes dataset in its original paper, while we assess its performance on both datasets. For evaluating the methods' effectiveness, we utilize macro-averaged Precision, macro-averaged Recall, Accuracy (Acc.), and F_1-Macro metrics in our comparisons. The comprehensive results of our proposed methods and all other baseline techniques are presented in Table 5.1 for both the nuScenes and RadarScenes datasets.

Table 5.1 The outcomes of the classification task for Vulnerable Road Users on the nuScenes and RadarScenes datasets.

Methods	nuScenes				RadarScenes			
	Precision	Recall	Acc.	F_1-Macro	Precision	Recall	Acc.	F_1-Macro
Schumann et al. [47]	0.45	0.43	0.25	0.26	0.77	0.75	0.99	0.75
Zhao et al. [61]	0.27	0.36	0.20	0.18	0.34	0.34	0.40	0.32
DGCNN [21]	0.44	0.57	0.32	0.31	0.81	0.86	0.98	0.83

PointNGCNN [67]	0.47	0.59	0.37	0.37	0.82	0.83	0.98	0.82
Radar-PointGNN [56]	0.44	0.48	**0.66**	0.44	0.22	0.50	0.23	0.14
RADAR-PointNGCNN	**0.52**	**0.63**	0.60	**0.49**	0.79	0.84	0.98	0.81
RADAR-DGCNN	0.48	0.61	0.52	0.44	**0.83**	**0.86**	**0.99**	**0.84**

Based on the data presented in Table 5.1, both of the proposed methods exhibit superior performance compared to the baseline VRU classification techniques concerning Precision, Recall, and F_1-Macro on the nuScenes dataset. Although the Radar-PointGNN method outperforms the proposed methods in terms of Accuracy on the nuScenes dataset, which is the same dataset used in the original Radar-PointGNN paper, our proposed methods significantly outperform the Radar-PointGNN across all metrics on the RadarScenes dataset. These results suggest that our proposed methods surpass all baselines and demonstrate higher generalizability to different datasets.

The RADAR-DGCNN method achieves the highest F_1-Macro score of 0.84 on the RadarScenes dataset, while the RADAR-PointNGCNN achieves the highest F1-Macro score of 0.49 on the nuScenes dataset. The RADAR-DGCNN method also secures the second-highest score on the RadarScenes dataset. We further compare the RADAR-DGCNN and RADAR-PointNGCNN methods using confusion matrices for both datasets in Figure 5.5 and Figure 5.6. The confusion matrices illustrate that our proposed methods exhibit high true positive values for all classes, except for the Pedestrian class. It appears that the Pedestrian class in the RadarScenes dataset is often misclassified as the Pedestrian-Group class (Figure 5.5), whereas in the nuScenes dataset, it is frequently misclassified as the Cycle class (Figure 5.6). This misinterpretation could be due to the proximity of pedestrians in the datasets or the presence of pedestrians using bicycles, leading to false negative values. Notably, the None class demonstrates the highest accuracy in the RADAR-DGCNN method, primarily because the number of elements in the None class in the RadarScenes dataset is relatively larger compared to other classes. The second-highest accuracy is observed for the Cycle class, followed by Truck, Pedestrian-Group, Car, and Pedestrian in both the RADAR-DGCNN and RADAR-PointNGCNN methods on the nuScenes dataset.

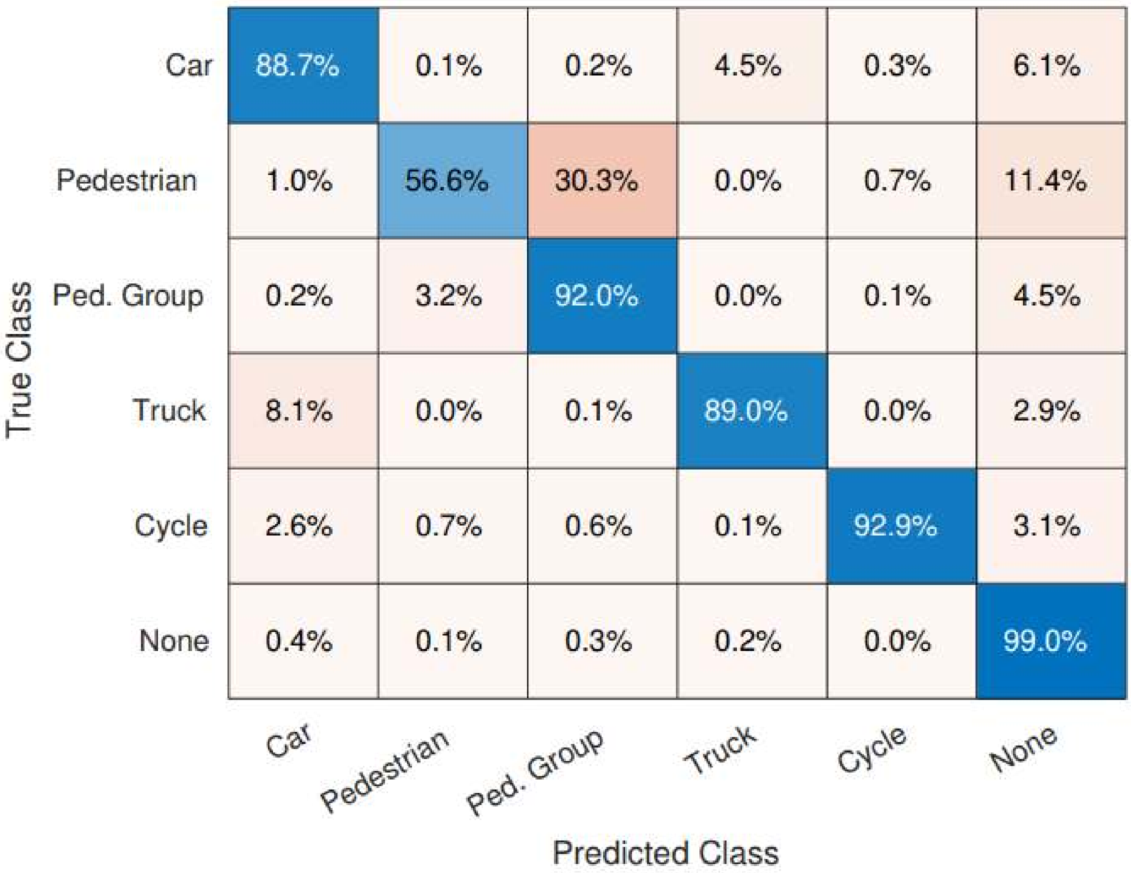

Figure 5.5 Confusion matrix for RADAR-DGCNN method on the RadarScenes dataset.

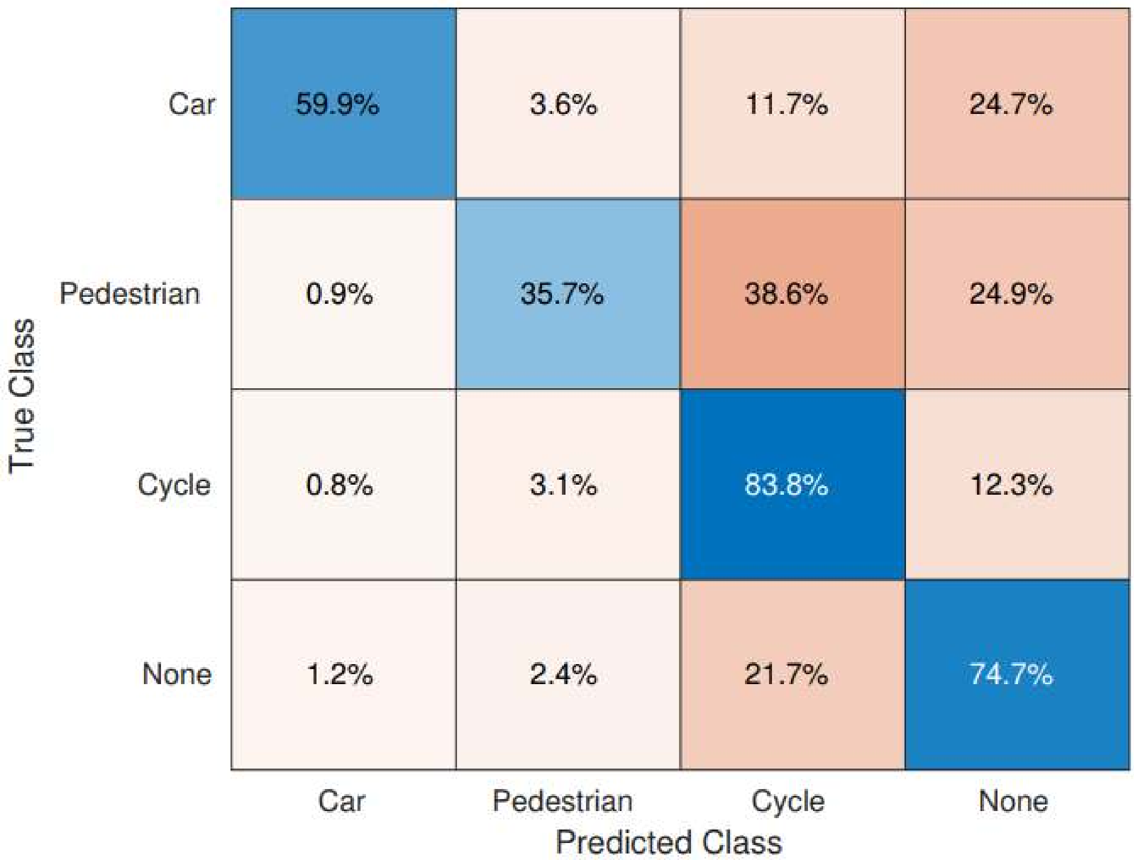

Figure 5.6 Confusion matrix for RADAR-PointNGCNN method on the nuScenes dataset

Figure 5.7 and Figure 5.8 shows the cumulative F_1-Macro performance across epochs for the nuScenes and RadarScenes datasets. These graphs are generated by training models on validation data. They showcase the results of baseline methods as well as both of our proposed methods. These graphs provide a visual representation of how the F_1 scores evolve on the validation set throughout the training process. In Figure 5.7, our proposed RADAR-PointNGCNN model showcases a superior F_1-Macro score compared to other models on the nuScenes dataset. Conversely, in Figure 5.8, both DGCNN and our proposed RADAR-DGCNN methods achieve the highest scores on the RadarScenes dataset. Subsequently, when evaluating our proposed RADAR-DGCNN on the RadarScenes dataset, it attains the highest overall score.

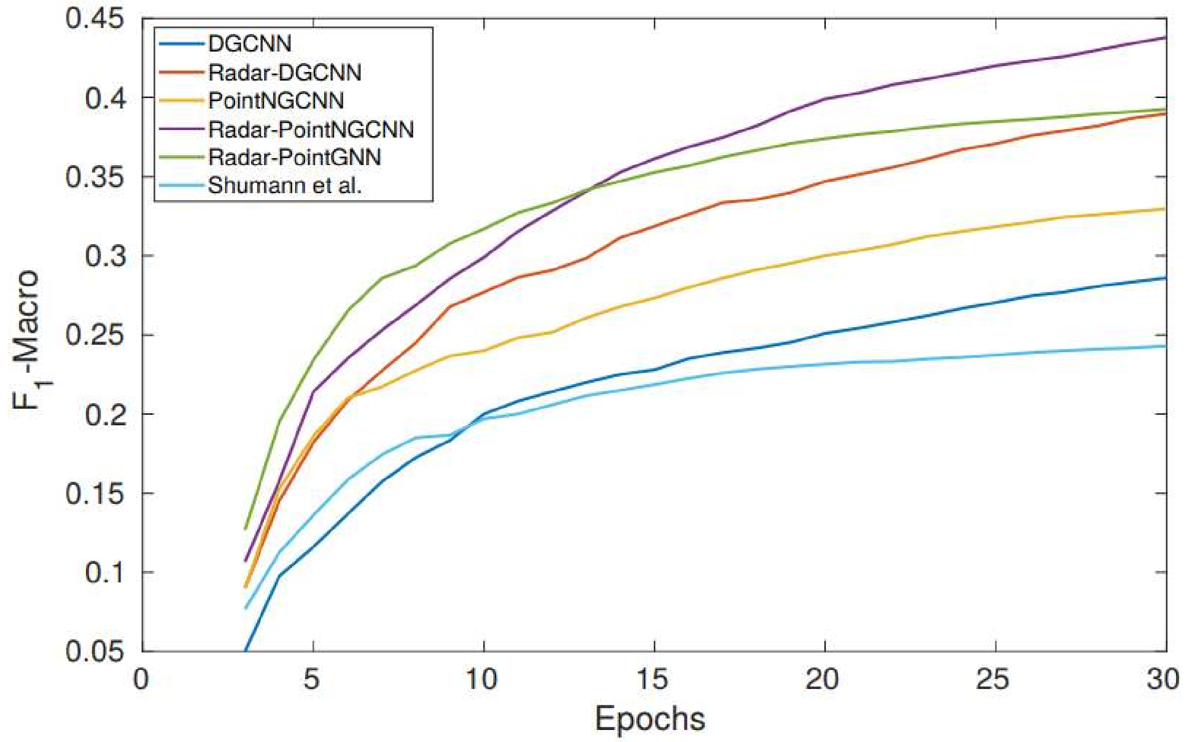

Figure 5.7 The graphs depict the F_1-Macro scores against the number of epochs on the nuScenes dataset.

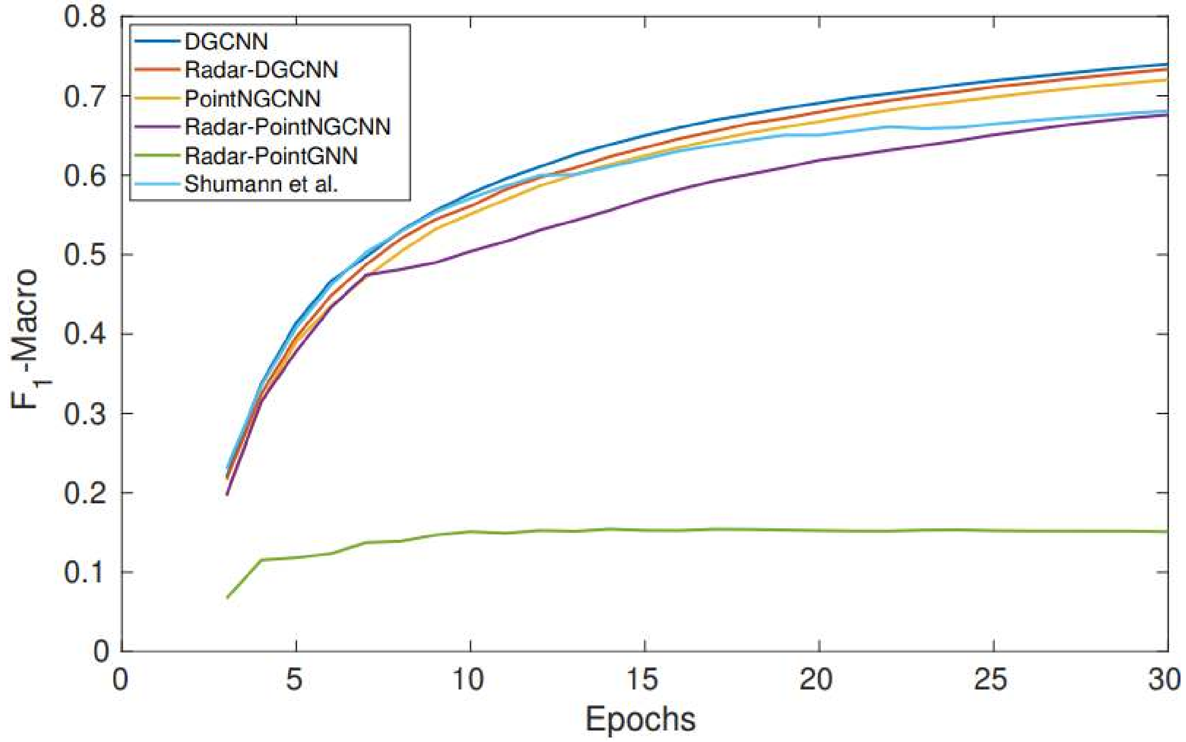

Figure 5.8 The graphs depict the F_1-Macro scores against the number of epochs on the RadarScenes dataset.

Figure 5.9 displays visual results for the RadarScenes dataset, displaying camera images of a sample scene, its corresponding ground truth, and the outcomes obtained using the proposed methods. In Figure 5.9a, we observe the sample scene camera image, while Figure 5.9b illustrates the ground truth point cloud, featuring four cars and points belonging to the "None" class. The results achieved by the RADAR-DGCNN and RADAR-PointNGCNN methods are presented in Figures 5.9c and 5.9d, respectively, with accuracy rates of 99.38% and 98.67% compared to the ground truth. Notably, the cars detected by RADAR-DGCNN are more discernible than those detected by RADAR-PointNGCNN.

For the nuScenes dataset, Figure 5.10 displays a sample scene, with Figure 10a exhibiting the camera image and Figure 5.10b representing the ground truth point cloud, containing two cars and points classified as pedestrians. The accuracy of the RADAR-DGCNN and RADAR-PointNGCNN methods concerning the ground truth is 94.88% and 93.12%, respectively, as depicted in Figures 5.10c and 5.10d. Due to the sparser nature of the nuScenes dataset, it contains less data than the RadarScenes data. Despite the absence of a detailed segmentation, the RADAR-PointNGCNN approach successfully detects both the automobile crossing the street and the parked car behind it, as demonstrated in Figure 5.10d.

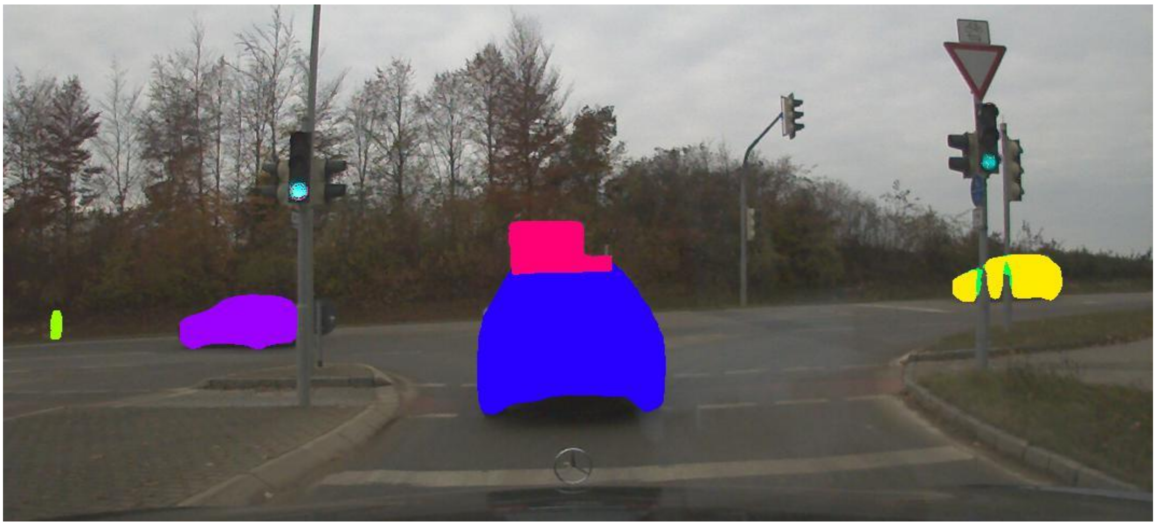

Figure 5.9a An example scene from the RadarScenes dataset.

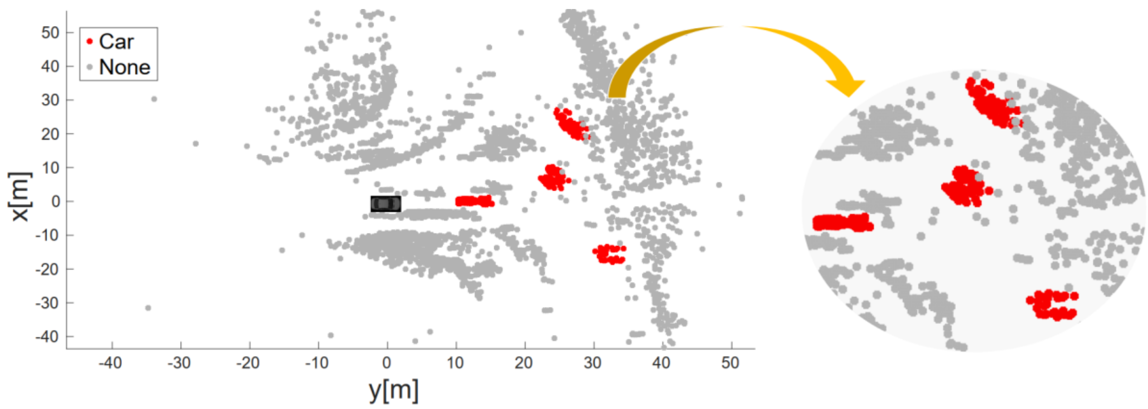

Figure 5.9b An example ground truth point cloud of the scene from the RadarScenes dataset.

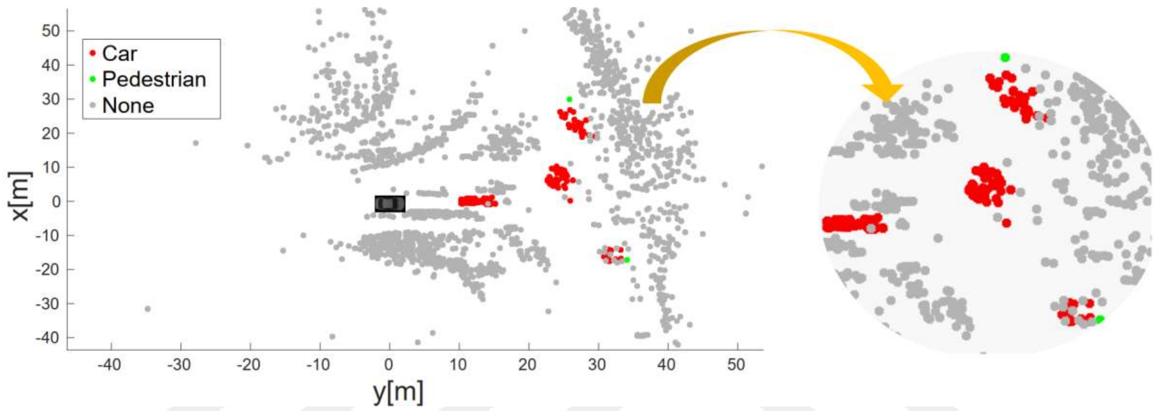

Figure 5.9c Results of RADAR-DGCNN method on the RadarScenes dataset.

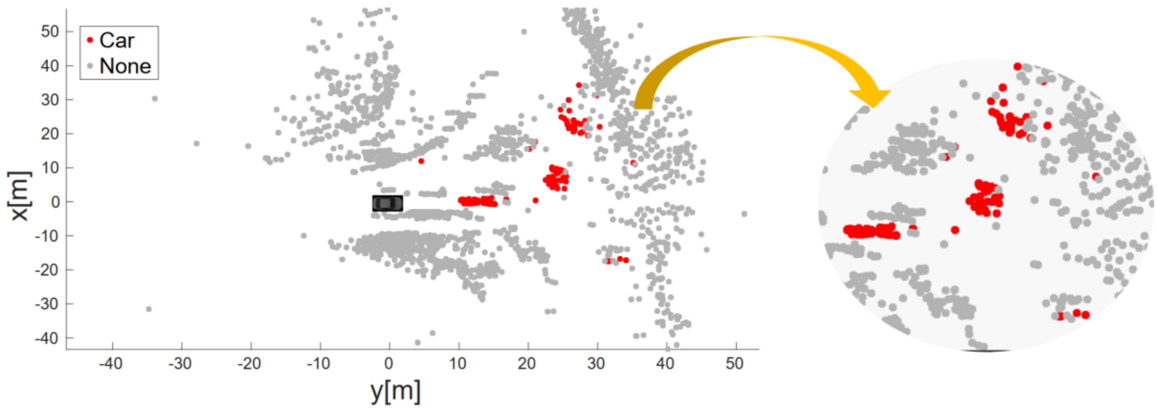

Figure 5.9d Results of RADAR-PointNGCNN method on the RadarScenes dataset.

Figure 5.9 The first figure displays a camera image capturing a scene, along with the corresponding ground truth RADAR point cloud data from the RadarScenes dataset. In the lower section, the proposed method's outcomes are showcased, depicting the actual scene from the RadarScenes dataset. The image featuring a black car represents the ego vehicle.

Figure 5.10a An example scene from the nuScenes dataset.

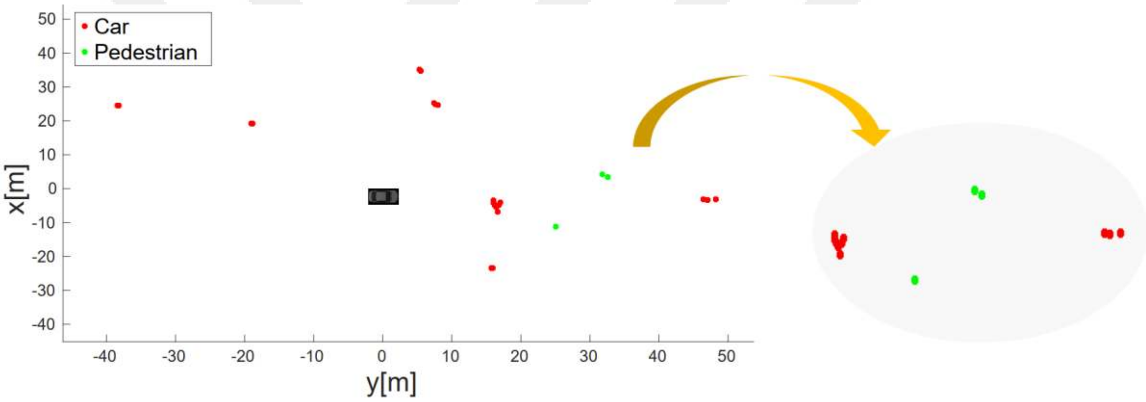

Figure 5.10b An example ground truth point cloud of the scene from the nuScenes dataset.

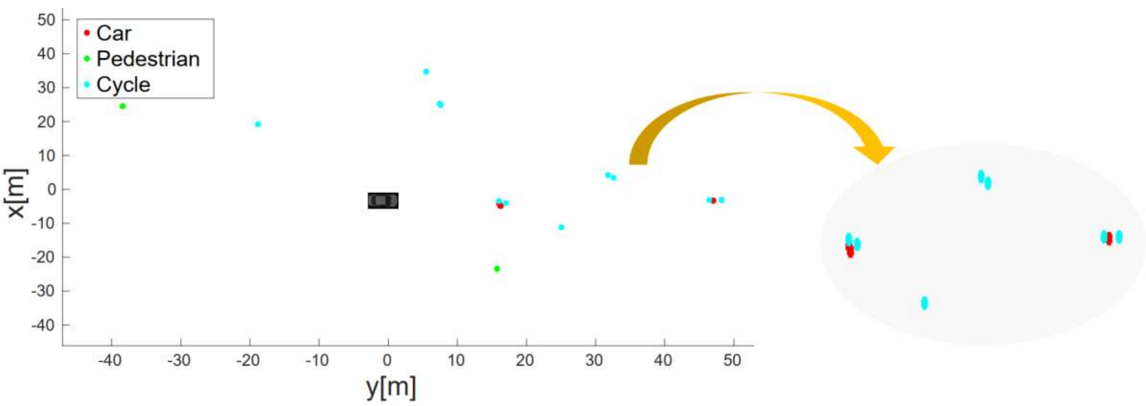

Figure 5.10c Results of RADAR-DGCNN method on the nuScenes dataset.

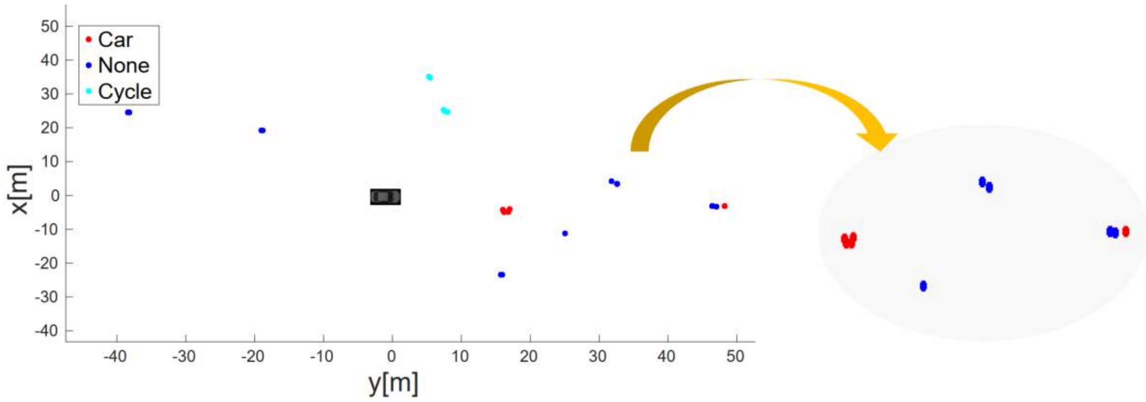

Figure 5.10d Results of RADAR-PointNGCNN method on the nuScenes dataset.

Figure 5.10 The first figure displays a camera image capturing a scene, along with the corresponding ground truth RADAR point cloud data from the nuScenes dataset. the proposed method's outcomes (In the lower section) , are showcased, depicting the actual scene from the nuScenes dataset. The image featuring a black car represents the ego vehicle.

In Table 5.2, we present a comparison of the complexity between our proposed methods and the baseline models. This comparison includes model sizes, computation times, and the resulting F_1-Macro scores. The model size indicates the total file size containing all variable values and the graph structure. Computation time represents the duration required to process a single batch.

Analyzing the results on the nuScenes dataset, we find that the Radar-PointGNN method, which has the second-highest F_1-Macro score, exhibits a smaller model size than our proposed methods. However, our proposed methods outperform Radar-PointGNN in terms of computational time. On the RadarScenes dataset, DGCNN ranks second in F_1-Macro compared to RadarDGCNN. The difference in model size and time between DGCNN and our proposed method is 0.05 MB and 0.09 ms, respectively, which is relatively minor compared to the F_1-Macro score. Although Zhao et al.[61] and Schumann et al. [47] methods demonstrate advantages in model sizes and computational time, our proposed methods outperform in terms of F_1-Macro scores.

In summary, upon examining the results in Table 5.2, our proposed methods achieve the best trade-off compared to the baseline models. They strike a balance between model complexity and computational efficiency while maintaining superior F_1-Macro performance.

Table 5.2 Complexity, Time, and F_1-Macro of different methods

	nuScenes			RadarScenes		
Methods	**Model Size (MB)**	**Time (ms)**	**F_1-Macro**	**Model Size (MB)**	**Time (ms)**	**F_1-Macro**
Schumann et al.	5.53	40.3	0.26	5.67	71.0	0.75
Zhao et al.	0.0007	104.2	0.18	0.23	1236.0	0.32
DGCNN	12.51	45.8	0.31	12.52	180.4	0.83
PointNGCNN	13.91	67.1	0.37	13.92	159.3	0.82
Radar-PointGNN	3.2	231.5	0.44	3.29	229.6	0.14
RADAR-PointNGCNN	13.93	68.9	**0.49**	13.93	159.1	0.81
RADAR-DGCNN	12.52	48.4	0.44	12.57	181.3	**0.84**

5.6.7. Conclusion

The novelty in our study the introduction of a new GSP-based method for Vehicle-to-VRU classification, specifically tailored for RADAR point cloud data. As RADAR point clouds exhibit irregular spatial structure and geometry, representing them as graphs preserves their irregularity without the need for converting the data into regular voxel grids. Current approaches within this domain are either tailored solely for LiDAR point clouds or do not possess the capability to effectively utilize the irregular characteristics of point clouds within the context of the GSP framework.

This research bridges the gap between LiDAR and GSP-based applications, enabling the effective use of GSP in the RADAR domain. Our proposed VRU classification method operates directly on raw point cloud data, eliminating the necessity for pre-processing steps. To evaluate our approach's performance, we conduct experiments on well-known publicly available datasets, including RadarScenes and NuScenes. Notably, our methods are the first VRU classification models evaluated on both datasets.

The experimental results validates the superiority of our proposed methods, RADAR-PointNGCNN on the nuScenes dataset, and RADAR-DGCNN on the RadarScenes dataset, outperforming the baseline studies in VRU classification. The F_1-Macro scores achieved are 0.49 and 0.84, respectively. Moreover, from a computational perspective, our proposed methods strike an optimal balance between model size and processing time when compared to the baseline approaches. This showcases the effectiveness and efficiency of our GSP-based VRU classification approach for RADAR point cloud data.

5.7. RadGT: Graph and Transformer-Based Automotive Radar Point Cloud Segmentation

5.7.1. Introduction

In the recent years, the integration of graph and Transformer architectures has gained significant traction as a potent and promising strategy for point cloud classification tasks. In Sections 1 and 3, we elaborated on the evolution and significance of graph-based methods. Furthermore, we explored graph-based methods through the introduction of two techniques operating within the GSP framework. We have seen that if we consider point clouds as graph nodes and make feature vectors more diverse with the GSP infrastructure, we can achieve good results. However, we think that there are still open points to be solved such as long-range dependency handling, positional encoding, and scalability.

A novel strategy in exploring the semantic relationship learning between objects incorporates the Transformers architecture [71]. The Transformers architecture has gained significant recognition for its impressive accomplishments in Natural Language Processing (NLP) [72], [73], [74] and image analysis [75]. Transformers offer a host of advantages, including their ability to capture long-range dependencies and support parallel processing, leading to state-of-the-art performance in diverse domains and tasks. Their adaptability to handle various modalities and facilitate transfer learning has further contributed to their wide adoption and profound impact in the field of artificial intelligence. Recently, several methodologies have adopted Transformer-based deep learning models for point clouds, demonstrating remarkable results [22], [76].

In our study, we introduce a new multi-class automotive RADAR point cloud classification techniques which we called as RadGT for VRUs by using GSP and Transformer framework. Our approach involves a fusion of graph-based techniques and Transformer models to achieve accurate and efficient multi-class classification. Leveraging essential target-level information as we used in Section 5.6, our proposed method effectively captures point dependencies within local neighbourhoods. Simultaneously, the Transformer-based component assigns distinct weights to neighbouring points and learns comprehensive global features. To validate our approach, we again carried out experiments on the RadarScenes (RS) and nuScenes (NS) datasets.

5.7.2. Related Work

Here, we summarise the related works focusing on Transformer-based approaches, extending the discussion from Section 5.6.2, where we already highlighted important works in the RADAR point cloud domain.

Point Cloud Transformer (PCT) [22] was the pioneering local feature extraction module that introduced an intra-domain self-attention (SA) mechanism to obtain centroid features. Point Transformer [77] followed suit by applying SA to the local range of each point and embedding location encoding in the input. To tackle the high computational cost of Point Transformer, PatchFormer [78] devised a solution by estimating a set of patches as bases in point clouds, replacing the key vector with these bases, significantly reducing complexity from $O(N^2)$ to $O(MN)$, where N is the original input points2 number and M is the number of bases ($M << N$).

Cloud Transformer [79] introduced spatial Transformers combined with translation, rotation, and scaling invariance. Additionally, it integrated 2D/3D mesh features to address Transformer's poor timeliness, considerably improving model efficiency. PVT [76] comprises a voxel branch and a point branch. The voxel branch obtains coarse-grained local features using Sparse Window Attention, while the point branch extracts fine-grained global features using Relative Attention or External Attention.

Some recent Transformer-based models adopted a self-supervised learning (SSL) strategy to establish point cloud representations from unlabeled data [80], [81], [82]. Among them, MV-JAR [81], proposed by Chen et al., adopts a Reversed-Furthest-Voxel-Sampling strategy to address the uneven distribution of LiDAR points. Voxel-MAE [80], another SSL model, employs a Transformer-based 3D object detector as a pre-trained backbone to process voxel data. While SSL methods eliminate the need for extensive manual annotations, their results may have lower performance compared to supervised models [83], [79], [76], [84], which require labelled data and achieve higher classification results in testing.

5.7.3. Model Architecture

In this thesis, we propose RadGT, an innovative deep learning model that capitalizes on the strengths of graph-based and Transformer-based methods to perform RADAR point cloud segmentation. By integrating these approaches, RadGT effectively captures both local and global features from the input data. This fusion enhances the representation of features, leading to improved performance in the RADAR point cloud segmentation task. The overall architecture of our method is illustrated in Figure 5.11. The RadGT model commences by extracting distinctive features obtained through the GSP approach, along with geometric coordinates. Our proposed model consists of three subsequent modules: Feature Alignment Block (FAB), the Graph Feature Space Module (GFSM) and the Transformer-based Segmentation Module (TSM). RadGT initially constructs graph representations from input point clouds, which are then employed by the FAB. The FAB is introduced with the purpose of producing enhanced point cloud data and deriving features from the graphs. In the GFSM, Feature Fusion blocks are utilized, playing a critical role in addressing the challenge of integrating multi-scale and multi-level features in point cloud analysis, as elucidated in [67]. This integration enables the model to capture both intricate details and contextual relationships within the point cloud data. The subsequent significant block is the Transformer module, which incorporates SA blocks. The initial output from the Graph Feature Space becomes the input to the first SA block. The SA blocks empower the model to assign varying levels of importance to different parts of the input data, enhancing its capability to comprehend complex relationships and make well-informed decisions.

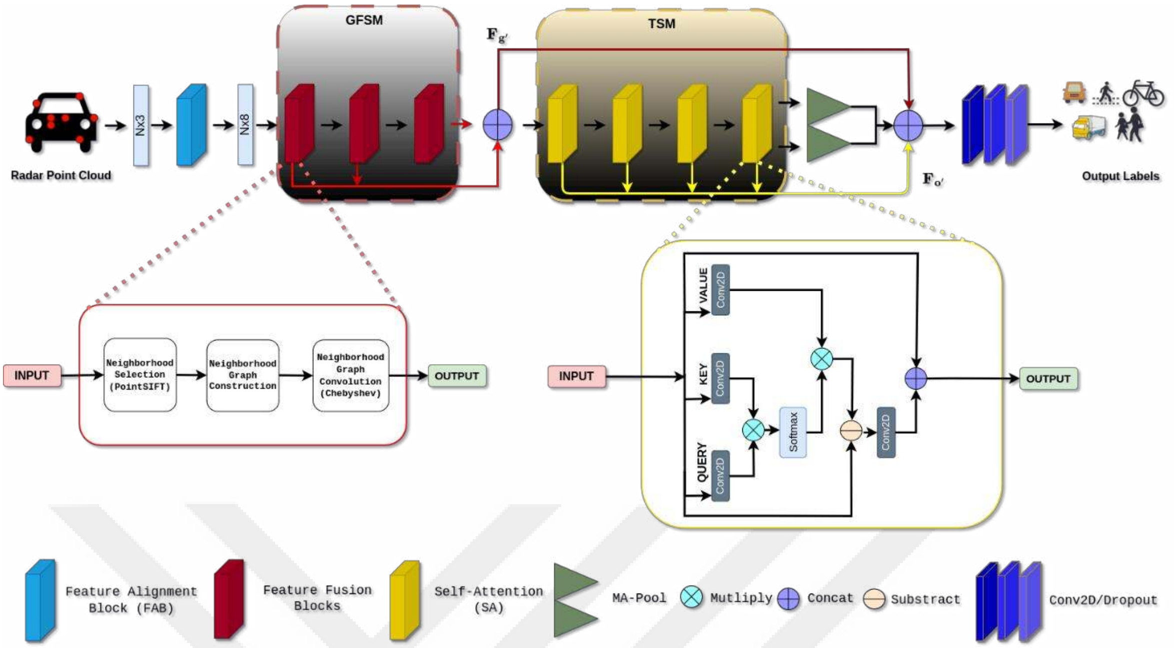

Figure 5.11 The proposed RadGT architecture.

5.7.3.1. Graph Feature Space Module (GFSM)

Our novel method presents a deep neural network architecture designed to handle point clouds directly by employing neighbourhood graph filters through graph convolutions, effectively leveraging local geometric structures. The module comprises three Feature Fusion Blocks (FFBs) that take high-dimensional feature vectors inspired by our proposed work in Section 5.6.4, denoted as $\mathbf{F}_e$, as their input. FFB architecture is depicted with red rectangular in Figure 5.11. The output of this module is $\mathbf{F}_g$.

5.7.3.2. Transformer Module (TSM)

We implement the Transformer module as the core of our architecture, depicted in Figure 5.11. This module aims to capture long-range dependencies and global context information. The SA block within the Transformer allows for the computation of relationships between points independently of their positions, irrespective of their specific order. The initial input to the first SA block is derived from the concatenated output of the Graph Feature Space module, denoted $\mathbf{F}_{g'}$. The SA mechanism in the Transformer architecture consists of three main components: Queries, Keys, and Values. These components are utilized to compute attention scores, capturing the relationships between different elements in the input sequence. To accomplish this, three distinct matrices, namely *query* (**Q**), *key* (**K**), and *value* (**V**), are generated through linear transformations of the original input, $\mathbf{F_g}$.

The generation of attention weights involves computing the dot product between the query (**Q**) and key (**K**) vectors, resulting in a matrix that represents the importance of each pair of elements in the sequence. Subsequently, these attention weights are normalized to reflect the weighted relevance of each element pair. The final SA mechanism is obtained by multiplying the attention weights with the value (**V**) matrix, creating a weighted combination of the values. To facilitate these operations, we utilize Conv2D layers for linear transformations as given in Eq. 5.1, which enable efficient and effective computation within the Transformer module.

$$\mathbf{Q}, \mathbf{K}, \mathbf{V} = \mathbf{F_{g'}} \left(\widetilde{\mathbf{W}}_{\mathbf{q}}, \widetilde{\mathbf{W}}_{\mathbf{k}}, \widetilde{\mathbf{W}}_{\mathbf{v}}\right) \tag{5.1}$$

where $\mathbf{Q}, \mathbf{K} \in \mathbb{R}^{N\times d_{in}}$ and $\mathbf{V} \in \mathbb{R}^{N\times d_{out}}$. Here, $d_{in} = d_{out}/4$. The trainable linear coefficients $\widetilde{\mathbf{W}}_{\mathbf{q}}, \widetilde{\mathbf{W}}_{\mathbf{k}}, \widetilde{\mathbf{W}}_{\mathbf{v}}$ are matrices incorporated into the convolutional operation, which can be adjusted during the training process. The attention weights of the matrix A are computed by performing matrix multiplication between **Q** and the transpose of **K** as in Eq.5.2 below:

$$\mathbf{A} = \text{Softmax}(\mathbf{Q}\mathbf{K}^T)/N \tag{5.2}$$

where N is the dimension of the matrix **A** to normalize. The self-attention F_{sa} is obtained by multiplying the attention weights with the Value matrix, **K**:

$$\mathbf{F_{sa}} = \mathbf{AV} \tag{5.3}$$

where $\mathbf{F_{sa}} \in \mathbb{R}^{N\times d_{out}}$. The difference between $\mathbf{F_{g'}}$ and $\mathbf{F_{sa}}$ called offset-attention is passed through a Conv2D and then the result is added to initial inputs. The output $\mathbf{F_o}$ is calculated in Eq. 5.4 as follows:

$$\mathbf{F_0} = \text{Conv2D}\left(\mathbf{F_{g'}}\text{-}\mathbf{F_{sa}}\right) + \mathbf{F_{g'}}. \tag{5.4}$$

To form the overall architecture, the four SA layers are merged by straightforwardly concatenating them along the feature dimension. Furthermore, we calculate the Max (MP) and Average (AP) Pooling called MA-Pool on the output of the last SA block. At the end, all four outputs including $\mathbf{F_{o'}}$, $\mathbf{F_{g'}}$, MA-Pool results are concatenated. Next, the output of concatenation is followed by convolution and dropout layers.

5.7.4. Experiments

5.7.4.1. Implementation Details

Our model comprises two primary modules: The Graph Feature Space and the Transformer modules. The Graph Feature Space is composed of three Feature Fusion blocks, with predetermined input and output dimensions of (8, 64), (64, 64), and (64, 128) respectively. Similarly, the Transformer module consists of four SA blocks, each with input and output dimensions set as (128, 128). To determine essential hyper-parameters, we conducted thorough investigations using random sampling. Among the parameters, we set the learning rates to 0.001, the batch size to 4, and the momentum to 0.9. The model underwent 50 epochs of training and evaluation on a Linux laptop equipped with an Nvidia GeForce GTX 3060 GPU. We implemented RadGT using Tensorflow v1.15.

To assess the model's performance, we adopted a five-fold cross-validation approach. This involved splitting the dataset into five equal-sized folds, with each fold containing 20% of the data. In each iteration, one fold served as the test set, while the remaining four folds were merged to form the training set.

5.7.4.2. Results

Here, we introduce an analysis of the experimental outcomes and performance evaluation of the proposed method. We conduct a comprehensive comparison between our approach and a diverse set of state-of-the-art and baseline methods, including Schumann et al. [47], DGCNN [21], PointNGCNN [67], Radar-PointGNN [16], RADAR-PointNGCNN [49], and

RADAR-DGCNN [49]. These methods have been extensively studied and are widely recognized in the field.

By including these baselines, we aim to establish a robust evaluation framework that facilitates a thorough comparison of our proposed approach with established techniques. To gauge the models' performance, we adopt the F_1-Macro, which is derived through macro-averaging. This evaluation metric offers a reliable measure of the overall effectiveness of the models in achieving the research objectives.

The results of our proposed approach are presented in Table 5.3, showcasing the evaluation outcomes for both the nuScenes and RadarScenes datasets. This comprehensive evaluation allows us to draw meaningful conclusions about the superiority and efficacy of our proposed approach in RADAR point cloud classification as compared to current state-of-the-art methods.

Table 5.3 Results of semantic segmentation task on the nuScenes and RadarScenes datasets

Methods	nuScenes				RadarScenes			
	Precision	Recall	Acc.	F_1-Macro	Precision	Recall	Acc.	F_1-Macro
Schumann et al. [47]	0.44± 0.02	0.41± 0.01	0.23± 0.03	0.25 ± 0.02	0.76± 0.01	0.74± 0.01	0.99± 0.01	0.74 ± 0.02
DGCNN [21]	0.44± 0.01	0.56± 0.02	0.31± 0.01	0.30 ± 0.03	0.80± 0.03	0.85± 0.02	0.97± 0.03	0.82 ± 0.02
PointNGCNN [67]	0.46± 0.01	0.59± 0.03	0.37± 0.01	0.37 ± 0.04	0.81± 0.02	0.82± 0.01	0.98± 0.02	0.81 ± 0.01
Radar-PointGNN [56]	0.45± 0.02	0.48± 0.02	0.66± 0.01	0.45 ± 0.02	0.23± 0.01	0.51± 0.03	0.23± 0.02	0.15 ± 0.02
RADAR-PointNGCNN [49]	0.52± 0.03	0.63± 0.02	0.61± 0.03	0.50 ± 0.01	0.79± 0.01	0.85± 0.02	0.99± 0.01	0.82 ± 0.01
RADAR-DGCNN [49]	0.50± 0.01	0.62± 0.02	0.53± 0.01	0.46 ± 0.01	0.84± 0.02	0.87± 0.01	0.99± 0.01	0.84 ± 0.01
RadGT	**0.53± 0.01**	**0.64± 0.01**	**0.63± 0.02**	**0.52 ± 0.02**	**0.85± 0.01**	**0.87± 0.02**	**0.98± 0.02**	**0.85 ± 0.01**

The outcomes presented in Table 5.3 indicate that the proposed method showcases superior performance compared to the baseline methods in terms of F_1-Macro, on both the nuScenes and RadarScenes datasets. In contrast to prior research findings [49], we observed a noticeable enhancement in the F_1-Macro results after incorporating the Transformer module. Specifically, our model exhibits a substantial 52% improvement in F_1-Macro compared to the RADAR-PointNGCNN approach when assessed on the nuScenes dataset. On the RadarScenes dataset, our model, named *RadGT*, achieves the highest performance, showcasing an impressive 0.85% improvement. The 5-fold cross-validation results are illustrated in Figure 5.12. It can be observed that the distribution of results from the proposed method on both datasets is relatively lower, indicating its superiority in object classification accuracy.

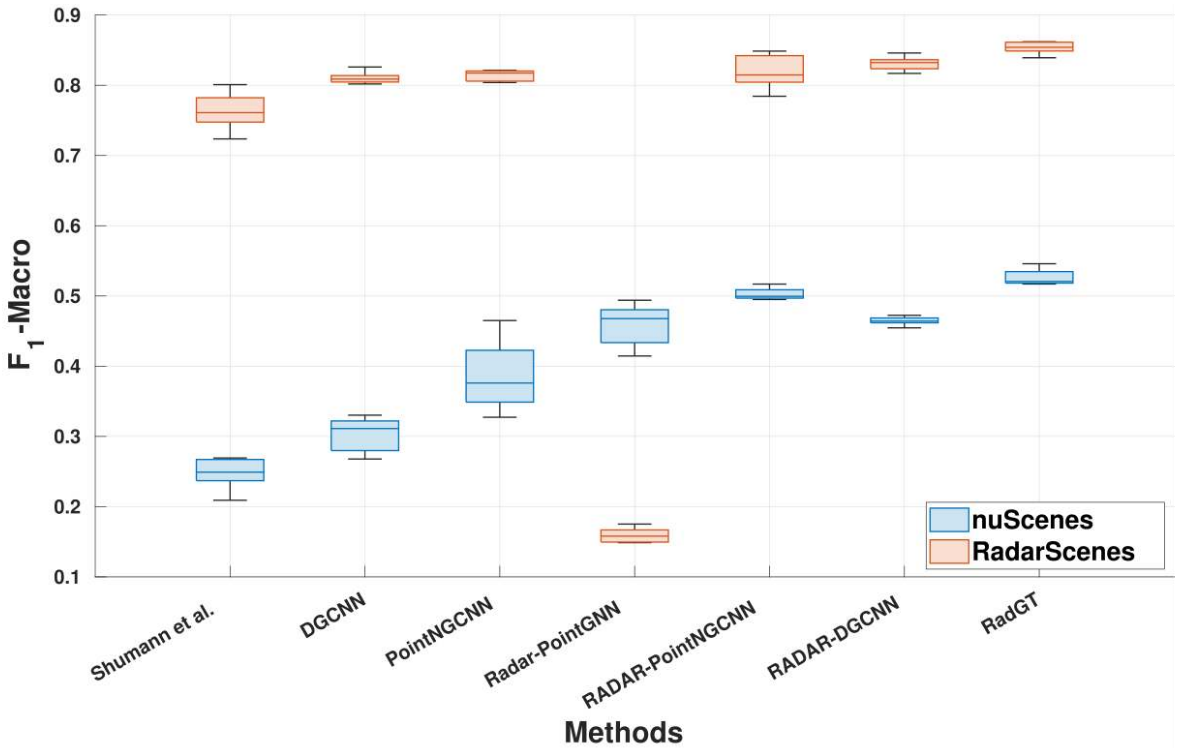

Figure 5.12 The $F1$-Macro results for both RadarScenes and nuScenes datasets were obtained by using five-fold cross validation

Through a comparison between the RADAR-PointNGCNN and RadGT models, both of which share certain components, we observe significantly improved performance in our proposed method. We attribute this enhancement to the superior capability of our approach in extracting neighborhood characteristics and spatial distribution information. As a result, our method exhibits higher efficiency and outperforms the EdgeConv method.

To demonstrate the practical application of our method, we present visual results in Figure 5.13, showcasing camera images of a sample scene, its corresponding ground truth, and the outcomes of the RadGT method for the RadarScenes dataset. In the sample scene (Figure 5.13a), three cars, one truck, and one cycle are marked in red. The corresponding ground truth point cloud for this scene is depicted in Figure 5.13b.

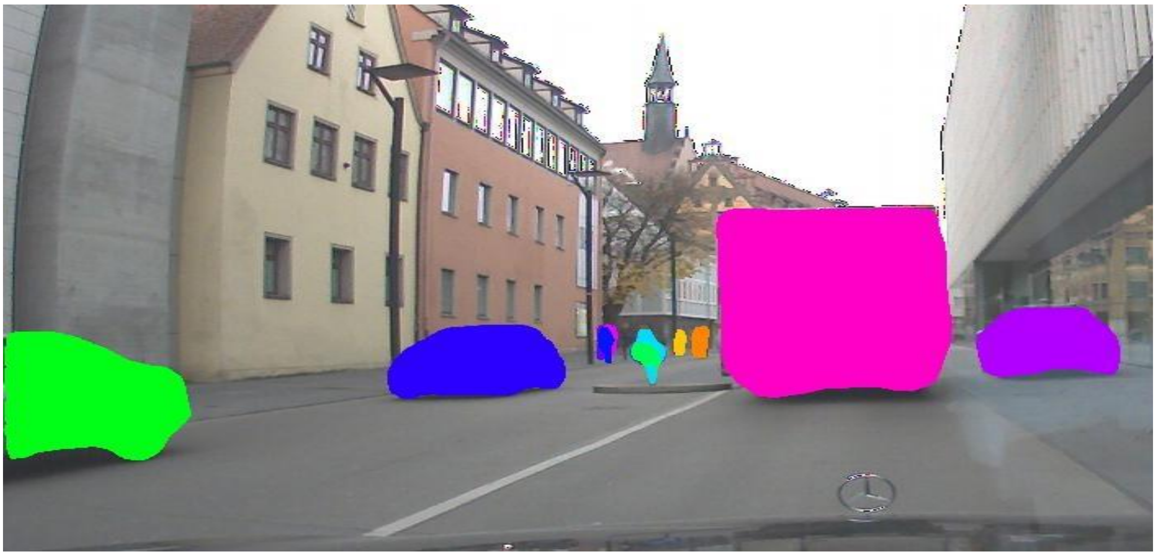

Figure 5.13a An example scene from the RadarScenes dataset.

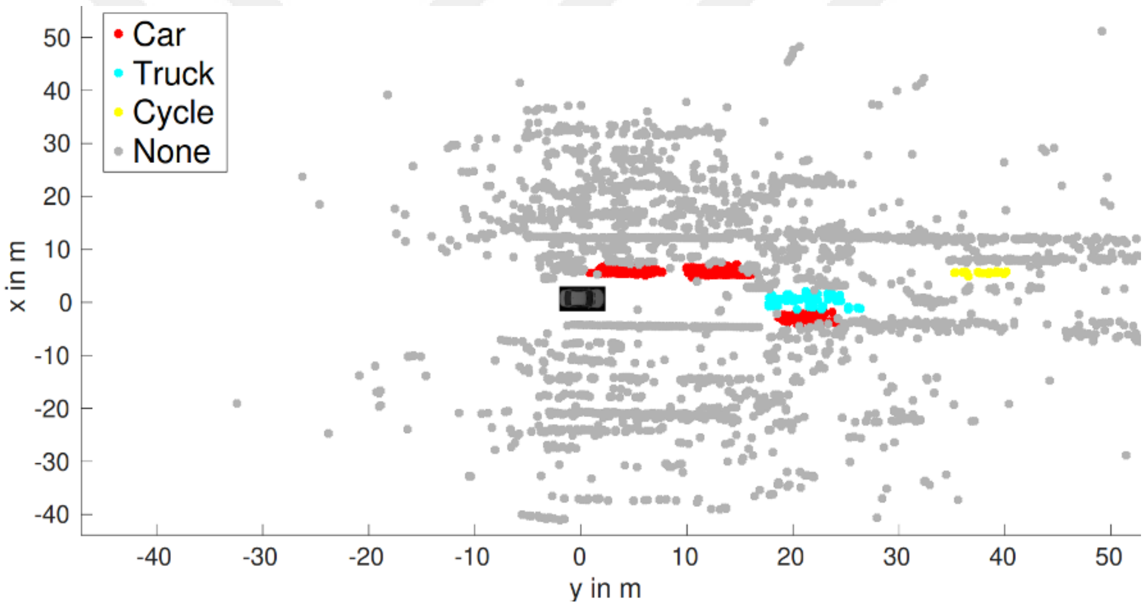

Figure 5.13b An example ground truth point cloud of the scene from the RadarScenes dataset.

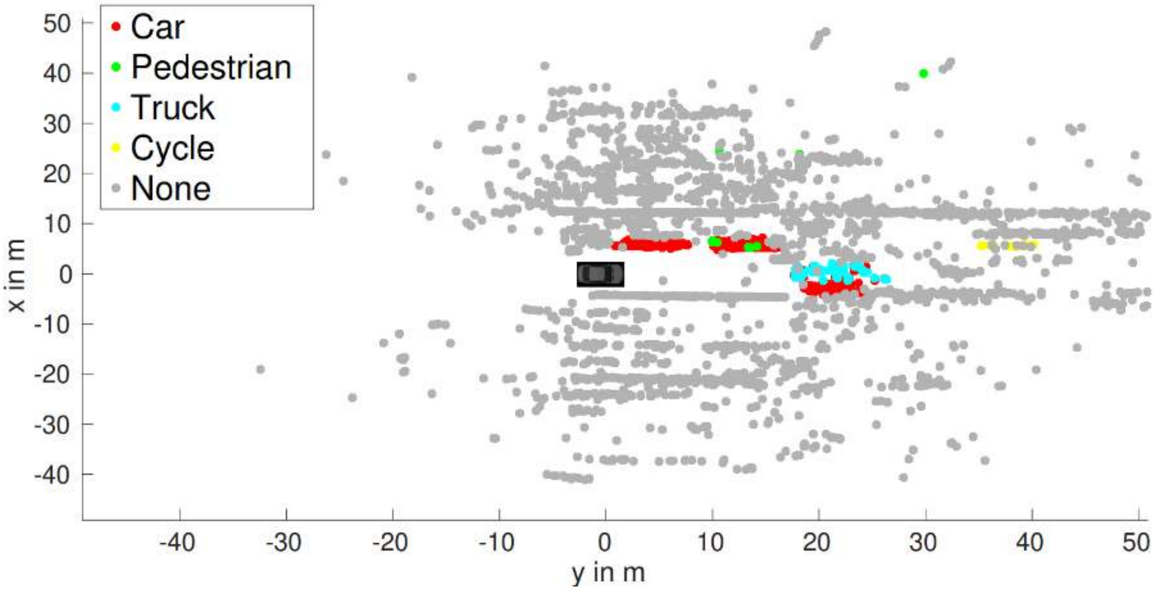

Figure 5.13c Results of RadGT method on the RadarScenes dataset.

Figure 5.13 Sample scene camera image, its ground truth RADAR

Despite the presence of a few false positive points, the objects in the resultant point cloud remain clearly discernible in Figure 5.13.c, indicating the effectiveness of our RadGT method. On the left side of the car, there are a few pedestrian class points that are incorrectly labeled, and at the back, some points are misclassified as cycle class. Nonetheless, these errors do not impact the overall outcomes.

We carried out ablation experiments to validate the effectiveness of different components within our model. It's important to mention that these ablation experiments are trained using the training subset and subsequently evaluated on the validation subset. As indicated in Table 5.4, we employ GFSM and TSM to compare the performance of the baseline and RadGT across various usage scenarios. In the initial row, we select RADAR-DGCNN [49] as the baseline model. By incorporating both modules alongside the baseline, we achieve the best detection performance, outperforming the baseline by 0.5%, 1.2%, 0.3%, 0.6%, 0.6%, 0% for the vehicle, pedestrian, truck, cycle, ped. group, and none categories respectively. This confirms the effectiveness of both GFSM and TSM. Since none class has too many elements, the result for none class doesn't change.s

Table 5.4 Ablation Studies of RadGT with different components on RadarScenes dataset.

Exp	GFSM	TSM	Vehicle	Pedestrian	Truck	Cycle	Ped. Group	None
1	✗	✗	88.7%	56.6%	89.0%	92.9%	92.0%	99.0%
2	✗	✓	87.4%	55.9%	89.2%	88.1%	92.4%	99.0%
3	✓	✗	88.9%	56.5%	88.7%	93.2%	91.5%	98.3%
4	✓	✓	**89.2%**	**57.8%**	**89.3%**	**93.5%**	**92.6%**	**99.0%**

5.7.5. Conclusion

In this section of thesis, we present RadGT, an innovative deep learning model that seamlessly combines the Transformer and graph frameworks for processing RADAR point clouds. By incorporating a SA mechanism from the Transformer methodology and leveraging graph-based applications, RadGT efficiently captures semantic relationships among the points. To evaluate its performance, we conducted experiments on widely recognized and publicly accessible RADAR datasets, nuScenes and RadarScenes. While previous evaluations were limited to graph-based methods, our novel approach outperforms these methods, achieving superior results.

The fusion of graph-based methods and Transformer in RadGT unlocks new possibilities for point cloud analysis and applications in the automotive RADAR domain. Our model demonstrates significant advancements over traditional approaches, demonstrating the potential of this synergistic integration. By harnessing the strengths of both frameworks, RadGT provides valuable contributions to deep learning and 3D data analysis, offering enhanced capabilities for processing and comprehending complex point cloud data. These promising outcomes encourage further exploration and research in the field of graph and Transformer-based techniques, facilitating the development of sophisticated and reliable solutions for diverse real-world applications.

5.8. Clustering-Based Object Detection for RADAR Point Clouds

5.8.1. Introduction

Several methods have been developed for classifying RADAR point clouds, as shown in previous sections. In contrast to the approaches discussed earlier, we introduce clustering-based methods for RADAR object classification, which diverges from the conventional point-based methods. This clustering-based technique is an integral part of the traditional RADAR detection pipeline. In this section, we present both the conventional detection pipeline and recent innovations in clustering and feature extraction. The classical RADAR detection pipeline comprises four main stages: Clustering, feature extraction, data augmentation and classification. Various research studies employ diverse methodologies at each stage.

At the earliest step as discussed in Section 2.2.2, a CFAR detector is employed to identify peaks in the Range-Doppler (RD) heat map, resulting in a list of detected targets. Subsequently, the moving targets are transformed and projected into Cartesian coordinates. After generating raw point clouds through CFAR, the data is subjected to DBSCAN (Density-Based Spatial Clustering of Applications with Noise) clustering [85]. In this process, static targets are typically removed since they cannot be differentiated from environmental clutter. Subsequently, after assigning labels, data augmentation is carried out. Lastly, feature extraction is executed to implement the desired process. Each of these four steps presents an opportunity for enhancement.

5.8.2. Related Work

Radar-based object classification has been gaining more interest, but the majority of research in this area has mainly focused on two-dimensional (2D) RADAR point clouds. Clustering plays a pivotal role in the RADAR detection process, particularly for advanced high-resolution RADAR systems [86]. DBSCAN is a preferred choice for several reasons: it eliminates the need for specifying a fixed number of clusters, accommodates arbitrary shapes, and exhibits fast processing capabilities [87]. Some researchers have enhanced DBSCAN by taking the properties of RADAR point clouds into account. Grid-based DBSCAN [88] suggests clustering RADAR points in a RD map to mitigate range-dependent resolution variations typically present in Cartesian coordinates. Additionally, Multi-stage clustering [60] introduces a two-stage framework to address clutter-related challenges. This approach employs a coarse-to-fine strategy, incorporating a second cluster merging step based on the velocity and spatial trajectory of clusters identified in the first stage.

As automotive RADAR resolution continues to advance, RADAR target classification has started to be a prominent area of research. For moving objects, the classification process benefits from considering micro-Doppler velocity, which captures subtle motions like wheels and arms. By analysing micro-Doppler signatures, various types of VRUs can be accurately classified. In [89], the classification of humans and vehicles was based on the Doppler spectrum. Molchanov et al. [90] sought to enhance real-time performance by utilizing image features derived from the distance Doppler matrix, achieving an accuracy of up to 88%. Another approach in [91] involved in using custom root RADAR cross-section features to extract three key features from the RADAR signal for classification purposes. Moreover, RADAR point clouds have been explored for object classification, as demonstrated by [61], who proposed a human-vehicle binary classification of 2D RADAR point clouds using the support vector machine (SVM) algorithm. For static objects, the classification focus shifts to statistical Radar Cross Section (RCS) and time-domain RCS, proposed by Cai et al. [92], as valuable features for distinguishing between vehicles and pedestrians. Some studies have explored leveraging a large number of features for improved classification accuracy. For instance, Scheiner et al. [93] considered an extensive set of 98 features and used heuristic-guided backward elimination for feature selection. They observed that range and Doppler features hold the most significance for classification, while angle and shape features are often disregarded due to lower angular resolution.

Moreover, the comparison between random forest and LSTM for RADAR classification was investigated by Schumann et al. [94]. Their experiments revealed that LSTM, when fed with eight-frame sequences, exhibited slightly superior performance compared to random forests, particularly for classes with similar shapes, such as pedestrians and pedestrian groups, as well as for false alarms. The convolutional neural network is not optimized for sparse data structures. To achieve improved performance, it becomes essential to enhance the input density of the RADAR sensors' point cloud. In their study, Scheiner et al. [95] conducted a comparison of the performance of three different methods for 2D detection: the two-stage clustering method, the occupancy grid mapping (OGM) -based method, and the PointNet-based method. The experimental results revealed that the OGM-based method outperformed the other two approaches, while the PointNet-based method performed notably worse, possibly attributed to issues related to sparsity in the data.

5.8.3. Model Architecture

In this section, we present two baseline methods working on RADAR point clouds although there are many methods using clustering-based algorithms. The fundamental idea of this approach corresponds to the description outlined Figure 5.14. The variations distinguishing the baselines primarily stem from differences in feature extraction methods or the types of classifiers employed. Zhao et al. [61] introduced a baseline method for point cloud classification, implementing a series of informative techniques. Initially, raw point clouds undergo noise suppression and clustering using the DBSCAN algorithm, resulting in clusters without category information. Subsequently, eleven-element feature vectors denoted as FV1 are extracted from these clusters to capture objects' characteristics as given in [61]. Finally, the kernel SVM classifier is employed to classify the feature vectors and provide the category information of objects. Furthermore, the method offers flexibility by enabling the utilization of various SVM kernels, such as linear, Gaussian, or polynomial, along with different feature vectors.

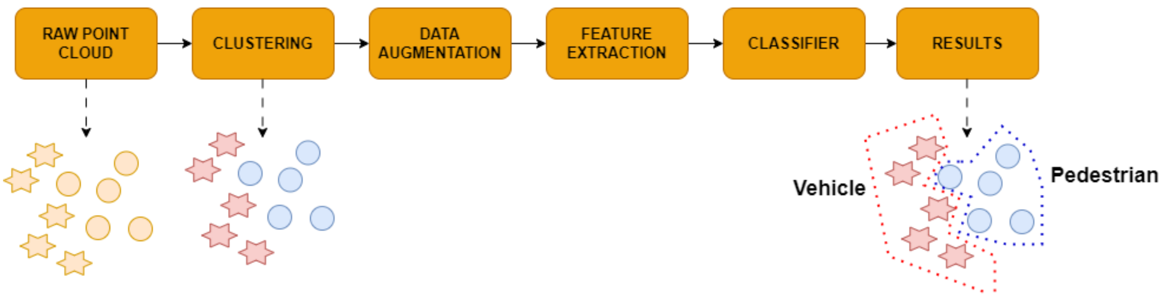

Figure 5.14 The sequence of operations for a multi-objective classification algorithm utilizing automotive RADAR.

For the other baseline method [94], Schumann et al. employed DBSCAN as the other baseline method and conducted a comparison between Random Forest and LSTM networks. Given LSTM's superior performance, the results obtained with LSTM were selected for further consideration and analysis. Distinct feature vectors, labelled as FV2 detailed in work [61] and [94] are employed for analysis.

To mention the DBSCAN algorithm which forms the basis of the common clustering section in this thesis, it is a popular density-based clustering algorithm used for identifying clusters in spatial data. It works by defining neighbourhoods based on density and is particularly effective in handling data with varying density and irregular shapes.
The key components of the DBSCAN algorithm include:

- Epsilon (ε) Neighbourhood: The algorithm defines a neighbourhood around each data point based on a specified distance threshold, denoted as ε. The ε-neighbourhood of a data point p includes all other data points within a distance of ε from p.
- MinPts: The minimum number of data points required to form a dense region or cluster is denoted as MinPts. In other words, a data point must have at least MinPts neighbours within its ε-neighbourhood to be considered as a core point.
- Core Points: Data points that have at least MinPts neighbours within their ε-neighbourhood are classified as core points. Core points are the foundation of clusters and act as the starting points for forming clusters.
- Border Points: Data points that do not meet the MinPts requirement for being core points but lie within the ε-neighbourhood of a core point are classified as border points. Border points can be part of a cluster but do not initiate new clusters.
- Noise Points: Data points that do not belong to any cluster, i.e., they are not core points and do not have core points within their ε-neighbourhood, are considered as noise points or outliers.

The algorithm's steps of DBSCAN can be summarized as follows:

1) For each data point, determine its ε-neighbourhood.
2) Identify core points that have at least MinPts neighbours within their ε-neighbourhood.
3) Form clusters by connecting directly reachable core points and their border points.
4) Assign noise points to clusters, if possible, based on their proximity to core points.

In this thesis, we utilize a simple GCN (Graph Convolutional Network) [96] classifier to analyze the provided feature vectors listed in work [61] and [94], in addition to our novel feature vector developed by using the GSP framework. The objective of a GCN classifier is to predict the class label or category of the entire graph based on its node features and the graph's structural information. The classifier typically employs message passing and aggregation operations to gather information from neighbouring nodes and iteratively update the node features. The final output is a probability distribution over the possible classes, indicating the likelihood of the graph belonging to each class.

The core idea behind GCNs is to learn node embeddings that capture both the node's features and its interactions with neighbouring nodes. This is achieved through graph convolutional layers, which are fundamental building blocks of GCNs. Graph convolutional layers enable information aggregation and propagation across the graph, allowing nodes to incorporate information from their neighbours to make informed decisions. The overall GCN architecture is depicted in Figure 5.15. In each graph convolutional layer, the process of updating node representations involves three steps:

- Feature propagation, where information is passed between neighbouring nodes.
- Linear transformation, where the aggregated information is combined with the node's own features using a learnable weight matrix.
- Pointwise nonlinear activation, where a nonlinearity, such as ReLU, is applied to the combined information to produce the updated node representation.

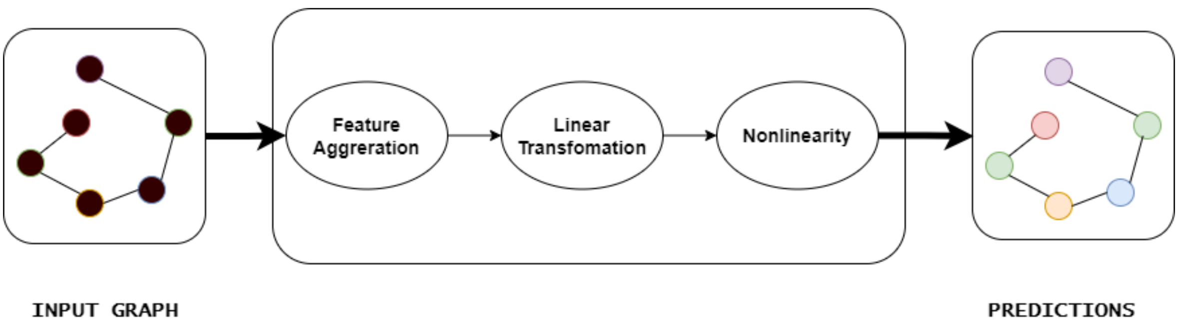

Figure 5.15 Schematic layout of a GCN

5.8.4. Experiments

In this section, we provide an evaluation of the experimental results and performance assessment of our graph-based approach. We perform an extensive comparison between our proposed approach and state-of-the-art and baseline methods. To validate the effectiveness of these methods, we employ the RadarScenes dataset as our testbed. The outcomes of our proposed method, as well as the baseline techniques, are showcased in Table 8. Apart from the outcomes, we furnish the F_1-Macro scores achieved by GCN using FV1 and FV2 feature vectors.

Table 8 The F_1-Macro scores obtained from the clustering-based techniques as well as GCN classifier on the RadarScenes dataset.

Methods	**Feature Vector**	**F_1-Macro**
Schumann et al. [85] (*LSTM*)	FV2	**0.88 ± 0.003**
Zhao et al. [61]	FV1	0.34± 0.001
GCN	FV2	0.68± 0.002
GCN	FV1	0.52± 0.001
GCN	mean($\mathbf{F_e}$)	0.64± 0.004

We explore a wide range of potential feature vectors, including the one proposed in Section 5.3.2, where we calculate the mean value of each element.

In the experimental results, we haven't seen significant improvements in the model's performance. Schumann et al. [85] with LSTM classifier achieved a higher accuracy compared to traditional graph-based models and Zhao et al. [61]. Specifically, on the test dataset, the method proposed by Schumann et al. [85] achieved an F_1-Macro score of 0.88, outperforming the baseline GCN by a huge margin.

5.8.5. Conclusion

In this section, we demonstrate the application of graph-based models and feature vectors for clustering-based object detection. We conduct experiments by using two baseline methods and GCN with various feature vectors. While we did not achieve higher scores, our results indicate the feasibility of using these approaches for clustering-based object detection on RADAR point clouds. As a future direction, addressing this challenge presents a significant demand and potential for further exploration, making it an important and promising topic for research.

6. SUMMARY

This thesis focuses on addressing the challenges of segmentation and edge detection in 3D RADAR point clouds using a signal processing approach. To effectively handle the unorganized nature of point cloud data as signals, we have harnessed the power of graph representations. Graphs offer a convenient and efficient means of encoding irregular data, especially those existing in high-dimensional geometric spaces. By incorporating techniques from spectral graph theory and graph signal processing, we have successfully extracted valuable information through the graph representations of 3D point clouds. These approaches enable us to analyse the data not only in spatial fashion but also in a spectral manner, providing comprehensive insights into the underlying structures and characteristics of the point clouds. Apart from graph-based techniques, we also directed our attention towards employing Transformer-based methods for RADAR point cloud processing. Transformer-based models utilize the potent SA mechanism to effectively capture long-range dependencies and intricate patterns within the point cloud. These techniques have exhibited encouraging outcomes in RADAR point cloud classification, showcasing enhanced performance and resilience in tackling diverse and demanding scenarios. By amalgamating graph-based and transformer-based methods, we can elevate the comprehension and interpretation of RADAR point cloud data, facilitating precise and dependable classification for a multitude of applications, including target recognition, object detection, and situational awareness.

RADAR point clouds are valuable for object detection and classification in autonomous driving scenarios. However, their sparsity, complexity, and irregularity pose challenges for accurate classification. This study addresses the classification challenge by utilizing graph-based algorithms within the GSP framework. Automotive RADAR sensors, generating unstructured 3D point clouds, naturally represent data in irregular grids, making them well-suited for graph-based techniques. The challenge lies in determining the graph topology and edge weights, which can be inferred based on the physical characteristics of the data. Overall, leveraging graph-based techniques proves advantageous for handling unstructured RADAR point cloud data effectively.

In the upcoming chapters, we explore research focused on RADAR data segmentation, where a combination of graph and Transformer-based methodologies is employed. The integration of graph and Transformer architectures has gained popularity for point cloud classification tasks. Despite achieving good results by considering point clouds as graph nodes, challenges remain, such as long-range dependency handling, positional encoding, and scalability. Transformers, renowned for their accomplishments in NLP and image analysis, offer advantages like capturing long-range dependencies and supporting parallel processing. Their adaptability and effectiveness in various domains have led to their widespread use in AI. Recent methodologies have applied Transformer-based models to point clouds, showing impressive results. In this thesis, a novel multi-class automotive RADAR point cloud classification technique called RadGT for VRUs is introduced. It combines graph-based techniques and Transformers to achieve accurate and efficient multi-class classification.

In the last chapter, we investigated the application of GCNs in clustering-based object detection on RADAR point clouds. The primary objective was to investigate the effectiveness of graph-based models and feature vectors for this specific task. We utilized two baseline methods as a point of comparison and tested various feature vectors with the GCN framework. Although our results did not show significantly higher scores compared to the baselines, we demonstrated the adaptability of GCNs for clustering-based object detection on RADAR point clouds. This finding highlights the potential of leveraging graph-based approaches to capture spatial relationships and interdependencies among points in the point cloud data.

As a concluding remark for this summary, after thoroughly examining the state-of-the-art techniques in RADAR point cloud, we explore novel approaches to tackle the problem more effectively. Additionally, we extend the algorithms to ensure computational efficiency. The experimental results demonstrate the promising potential of studies, not only for RADAR point clouds but also in other data domains characterized by high-dimensional geometry and irregular structures.

7. DISCUSSIONS AND FUTURE WORK

In our experiments on common RADAR benchmark datasets, we have demonstrated state-of-the-art results for VRU classification using machine learning-based signal processing methods. However, as we extend these techniques to real-world scenarios, it is essential to consider some restrictions and limitations stemming from sensor-level properties, as discussed comprehensively in [97].

One limitation is the sparsity and clutter present in RADAR data. Classical RADAR detectors like Constant False Alarm Rate (CFAR) can exacerbate these issues [98]. While advanced methods like Cell Averaging (CA)-CFAR and Ordered Statistic (OS)-CFAR have been proposed to address this, a trade-off between clutter and sparsity remains. Additionally, ghost detection due to multipath propagation poses challenges, requiring expert knowledge for labelling datasets with such samples to develop algorithms for ghost object detection. High-quality datasets, such as nuScenes and RadarScenes, labelled by boundary box and human experts, respectively, are employed in these studies.

Another challenge is the lower angular resolution in the horizontal view when using conventional 3D RADAR datasets, which contributes to sparser RADAR point clouds. 4D mmWave RADARs offer higher angular resolution in both azimuth and elevation, enabling the inclusion of 4D data in feature vectors. Expanding our proposed methods to accommodate 4D point clouds can be achieved by incorporating height information and calculating the degree of connected edges in the feature vector.

Moreover, the number of equivalent virtual receiving antennas can frequently limit azimuth and elevation resolutions, even with 4D RADARs. Increasing the number of virtual receiving antennas incurs higher hardware costs and sizes. Therefore, careful consideration of hardware constraints is necessary when aiming to enhance angular resolution.

In addition, the evaluation of the proposed methods is carried out on the nuScenes and RadarScenes datasets, both of which are based on 3D mmWave RADAR technology. In future research, the RADAR-DGCNN and RADAR-PointNGCNN methods could be further assessed on 4D mmWave RADAR datasets. These 4D RADARs have the advantage of capturing higher resolution and denser point clouds, including height information, which leads to the generation of high-quality datasets. By adapting the developed feature vectors, our proposed methods can be enhanced by incorporating 4D RADAR datasets.

Additionally, the suitability of classification algorithms and the corresponding datasets for their intended use in specific environments should be taken into account. RADAR is often referred to as an all-weather sensor; however, the effects of adverse weather on classification performance have not been extensively explored. Future research should investigate data collected in controlled areas, such as airfields, for various autonomous vehicle (AV) applications in urban and rural settings.

In summary, while our proposed methods have demonstrated significant advancements in VRU classification using RADAR point clouds, various challenges related to data sparsity, clutter, ghost detection, angular resolution, hardware constraints, and environmental factors warrant further investigation and consideration in future research to enhance the reliability and applicability of RADAR-based classification algorithms. Additionally, there exist techniques designed to enhance the quality of point clouds by reducing the impact of background noise and producing point clouds with higher density. Future research could focus on investigating more efficient RADAR detectors to achieve these improvements. Feature selection is an another topic that affect the processing and analyzing 3D spatial data. It entails recognizing and selecting from a broader collection of accessible features a subset of the most relevant and informative features. Utilizing feature selection algorithms can be advantageous for addressing data imbalance and selecting more appropriate features. This can be challenging new work. In addition, we have analysed our methods on six classes for RadarScenes dataset and four classes for nuScenes dataset. As a future work, some classes can be combined such as pedestrian and ped. group classes and better comparability can be achieved. Moreover, there is valuable potential in applying the GSP-based methodology along with clustering-based algorithms on a RADAR dataset, as demonstrated in Section 5.8. This opens up promising avenues for future research. Enhancing feature vectors and graph-based classifiers and experimenting with diverse datasets, such as a 4D RADAR dataset, can further explore their capabilities in point cloud classification and object detection tasks.

REFERENCES

[1] R. Ishrat, “Global Status Report on Road Safety 2018: Summary - WHO | World Health Organization,” *World Heal. Organ.*, 2018.

[2] J. Levinson, J. Askeland, J. Becker, J. Dolson, D. Held, S. Kammel, J. Z. Kolter, D. Langer, O. Pink, V. Pratt, M. Sokolsky, G. Stanek, D. Stavens, A. Teichman, M. Werling, and S. Thrun, “Towards fully autonomous driving: Systems and algorithms,” in *IEEE Intelligent Vehicles Symposium, Proceedings*, 2011.

[3] NHTSA, “Automated Vehicles for Safety,” *National Highway Traffic Safety Administration, Technology and Innovation*, 2020. .

[4] O. Egbue and S. Long, “Barriers to widespread adoption of electric vehicles: An analysis of consumer attitudes and perceptions,” *Energy Policy*, 2012.

[5] S. A. Cohen and D. Hopkins, “Autonomous vehicles and the future of urban tourism,” *Ann. Tour. Res.*, 2019.

[6] K. Bimbraw, “Autonomous cars: Past, present and future: A review of the developments in the last century, the present scenario and the expected future of autonomous vehicle technology,” in *ICINCO 2015 - 12th International Conference on Informatics in Control, Automation and Robotics, Proceedings*, 2015.

[7] B. Sivak, M., & Schoettle, “Road safety with self-driving vehicles: general limitations and road sharing with conventional vehicles,” *J. Safety Res.*, pp. 67–74, 2019.

[8] J. Wang, J. Liu, and N. Kato, “Networking and Communications in Autonomous Driving: A Survey,” *IEEE Commun. Surv. Tutorials*, 2019.

[9] J. Kocic, N. Jovicic, and V. Drndarevic, “Sensors and Sensor Fusion in Autonomous Vehicles,” in *2018 26th Telecommunications Forum, TELFOR 2018 - Proceedings*, 2018.

[10] C. Waldschmidt, J. Hasch, and W. Menzel, "Automotive Radar — From First Efforts to Future Systems," *IEEE J. Microwaves*, 2021.

[11] S. M. Patole, M. Torlak, D. Wang, and M. Ali, "Automotive radars: A review of signal processing techniques," *IEEE Signal Process. Mag.*, vol. 34, no. 2, pp. 22–35, 2017.

[12] I. Bilik, O. Longman, S. Villeval, and J. Tabrikian, "The rise of radar for autonomous vehicles: Signal processing solutions and future research directions," *IEEE Signal Process. Mag.*, vol. 36, no. 5, pp. 20–31, 2019.

[13] R. A. Krofli and R. D. Kelly, "Meteorological research applications of mm-wave radar," *Meteorol. Atmos. Phys.*, vol. 59, no. 1, pp. 105–121, 1996.

[14] A. D. Singh, S. S. Sandha, L. Garcia, and M. Srivastava, "Radhar: Human activity recognition from point clouds generated through a millimeter-wave radar," in *Proceedings of the Annual International Conference on Mobile Computing and Networking, MOBICOM*, 2019.

[15] A. Smith, J., Johnson, "Advances in Automotive Radar Technology for Autonomous Vehicles," *Int. J. Auton. Syst.*, vol. 5(2), pp. 100–115, 2022.

[16] S. Rajendar and V. K. Kaliappan, "Recent Advancements in Autonomous Emergency Braking: A Survey," in *Proceedings of the 5th International Conference on Electronics, Communication and Aerospace Technology, ICECA 2021*, 2021.

[17] W. Shi and R. Rajkumar, "Point-GNN: Graph neural network for 3D object detection in a point cloud," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2020.

[18] Y. Zhou and O. Tuzel, "VoxelNet: End-to-End Learning for Point Cloud Based 3D Object Detection," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2018.

[19] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, “PointNet: Deep learning on point sets for 3D classification and segmentation,” in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, 2017.

[20] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, “PointNet++: Deep hierarchical feature learning on point sets in a metric space,” in *Advances in Neural Information Processing Systems*, 2017.

[21] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, “Dynamic graph Cnn for learning on point clouds,” *ACM Trans. Graph.*, 2019.

[22] M. H. Guo, J. X. Cai, Z. N. Liu, T. J. Mu, R. R. Martin, and S. M. Hu, “PCT: Point cloud transformer,” *Comput. Vis. Media*, 2021.

[23] S. M. Patole, M. Torlak, D. Wang, and M. Ali, “Automotive Radars: A review of signal processing techniques,” *IEEE Signal Process. Mag.*, 2017.

[24] A. Shafew, E. J. Kim, and D. Kim, “A Study on mm Wave Sensor Technology and its Applications,” in *International Conference on ICT Convergence*, 2022.

[25] F. Jin, “Millimeter-wave Radar Point Cloud Classification and Interference Mitigation,” The University of Arizona, 2020.

[26] M. A. Richards, J. A. Scheer, and W. A. Holm, *Principles of modern radar: Basic principles*. 2010.

[27] R. A. Sevimli, “TARGET DETECTION AND IMAGING ON PASSIVE BISTATIC RADAR SYSTEMS,” Bilkent University, 2014.

[28] F. D. W. Enggar, A. M. Muthiah, O. D. Winarko, O. N. Samijayani, and S. Rahmatia, “Performance comparison of various windowing on FMCW radar signal processing,” in *2016 International Symposium on Electronics and Smart Devices, ISESD 2016*, 2017.

[29] A. Ortega, P. Frossard, J. Kovacevic, J. M. F. Moura, and P. Vandergheynst, “Graph

Signal Processing: Overview, Challenges, and Applications," *Proc. IEEE*, vol. 106, no. 5, pp. 808–828, 2018.

[30] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," *IEEE Signal Process. Mag.*, vol. 30, no. 3, pp. 83–98, 2013.

[31] A. Sandryhaila and J. M. F. Moura, "Discrete signal processing on graphs: Frequency analysis," *IEEE Trans. Signal Process.*, vol. 62, no. 12, pp. 3042–3054, 2014.

[32] A. Sandryhaila and J. M. F. Moura, "Big data analysis with signal processing on graphs: Representation and processing of massive data sets with irregular structure," *IEEE Signal Process. Mag.*, 2014.

[33] A. Sandryhaila and J. M. F. Moura, "Big data analysis with signal processing on graphs: Representation and processing of massive data sets with irregular structure," *IEEE Signal Process. Mag.*, vol. 31, no. 5, pp. 80–90, 2014.

[34] F. R. K. Chung, "Lectures on Spectral Graph Theory," *Lecture Notes*. 2001.

[35] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the KITTI vision benchmark suite," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2012.

[36] M. Meyer and G. Kuschk, "Automotive radar dataset for deep learning based 3D object detection," in *EuRAD 2019 - 2019 16th European Radar Conference*, 2019.

[37] D. Barnes, M. Gadd, P. Murcutt, P. Newman, and I. Posner, "The Oxford Radar RobotCar Dataset: A Radar Extension to the Oxford RobotCar Dataset," in *Proceedings - IEEE International Conference on Robotics and Automation*, 2020.

[38] M. Mostajabi, C. M. Wang, D. Ranjan, and G. Hsyu, "High resolution radar dataset for semi-supervised learning of dynamic objects," in *IEEE Computer Society*

Conference on Computer Vision and Pattern Recognition Workshops, 2020.

[39] G. Kim, Y. S. Park, Y. Cho, J. Jeong, and A. Kim, "MulRan: Multimodal Range Dataset for Urban Place Recognition," in *Proceedings - IEEE International Conference on Robotics and Automation*, 2020.

[40] M. Sheeny, E. de Pellegrin, S. Mukherjee, A. Ahrabian, S. Wang, and A. Wallace, "Radiate: A Radar Dataset for Automotive Perception in Bad Weather," in *Proceedings - IEEE International Conference on Robotics and Automation*, 2021.

[41] A. Ouaknine, A. Newson, J. Rebut, F. Tupin, and P. Pérez, "CarraDA dataset: Camera and automotive radar with range-angle-doppler annotations," in *Proceedings - International Conference on Pattern Recognition*, 2020.

[42] N. Scheiner, F. Kraus, F. Wei, B. Phan, F. Mannan, N. Appenrodt, W. Ritter, J. DIckmann, K. DIetmayer, B. Sick, and F. Heide, "Seeing around Street Corners: Non-Line-of-Sight Detection and Tracking In-the-Wild Using Doppler Radar," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2020.

[43] Y. Wang, Z. Jiang, X. Gao, J. N. Hwang, G. Xing, and H. Liu, "RODNet: Radar object detection using cross-modal supervision," in *Proceedings - 2021 IEEE Winter Conference on Applications of Computer Vision, WACV 2021*, 2021.

[44] O. Schumann∗, M. Hahn, N. Scheiner∗, F. Weishaupt∗, J. F. Tilly∗, J. D. ¨, ∗, and and Christian Wohler, "RadarScenes: A Real-World Radar Point Cloud Data Set for Automotive Applications," *arXiv*, 2021.

[45] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "Nuscenes: A multimodal dataset for autonomous driving," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2020.

[46] R. L. Figueroa, Q. Zeng-Treitler, S. Kandula, and L. H. Ngo, "Predicting sample size

required for classification performance," *BMC Med. Inform. Decis. Mak.*, 2012.

[47] O. Schumann, M. Hahn, J. Dickmann, and C. Wöhler, "Semantic Segmentation on Radar Point Clouds," in *2018 21st International Conference on Information Fusion, FUSION 2018*, 2018.

[48] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner, "ScanNet: Richly-annotated 3D reconstructions of indoor scenes," in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, 2017.

[49] R. A. Sevimli, M. Üçüncü, and A. Koç, "Graph signal processing based object classification for automotive RADAR point clouds," *Digit. Signal Process. A Rev. J.*, vol. 137, p. 104045, 2023.

[50] C. M. Bishop, "Bishop - Pattern Recognition And Machine Learning - Springer 2006," *Antimicrob. Agents Chemother.*, 2014.

[51] F. Fent, "Machine Learning-Based Radar Point Cloud Segmentation," 2020.

[52] P. Soda, "A hybrid approach handling imbalanced datasets," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2009.

[53] S. Chen, B. Liu, C. Feng, C. Vallespi-Gonzalez, and C. Wellington, "3D point cloud processing and learning for autonomous driving: Impacting map creation, localization, and perception," *IEEE Signal Process. Mag.*, vol. 38, no. 1, pp. 68–86, 2020.

[54] A. Ioannidou, E. Chatzilari, S. Nikolopoulos, and I. Kompatsiaris, "Deep learning advances in computer vision with 3D data: A survey," *ACM Computing Surveys*. 2017.

[55] Y. Guo, H. Wang, Q. Hu, H. Liu, L. Liu, and M. Bennamoun, "Deep Learning for 3D Point Clouds: A Survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2020.

[56] P. Svenningsson, F. Fioranelli, and A. Yarovoy, "Radar-PointGNN: Graph Based Object Recognition for Unstructured Radar Point-cloud Data," in *IEEE National Radar Conference - Proceedings*, 2021.

[57] P. Gong, C. Wang, and L. Zhang, "MMPoint-GNN: Graph Neural Network with Dynamic Edges for Human Activity Recognition through a Millimeter-Wave Radar," in *Proceedings of the International Joint Conference on Neural Networks*, 2021.

[58] R. Prophet, M. Hoffmann, M. Vossiek, C. Sturm, A. Ossowska, W. Malik, and U. Lubbert, "Pedestrian classification with a 79GHz automotive radar sensor," in *Proceedings International Radar Symposium*, 2018.

[59] N. Scheiner, N. Appenrodt, J. DIckmann, and B. Sick, "Radar-based Feature Design and Multiclass Classification for Road User Recognition," in *IEEE Intelligent Vehicles Symposium, Proceedings*, 2018.

[60] N. Scheiner, N. Appenrodt, J. Dickmann, and B. Sick, "A Multi-Stage Clustering Framework for Automotive Radar Data," in *2019 IEEE Intelligent Transportation Systems Conference, ITSC 2019*, 2019.

[61] Z. Zhao, Y. Song, F. Cui, J. Zhu, C. Song, Z. Xu, and K. Ding, "Point Cloud Features-Based Kernel SVM for Human-Vehicle Classification in Millimeter Wave Radar," *IEEE Access*, 2020.

[62] A. Palffy, J. Dong, J. F. P. Kooij, and D. M. Gavrila, "CNN Based Road User Detection Using the 3D Radar Cube," *IEEE Robot. Autom. Lett.*, 2020.

[63] A. Danzer, T. Griebel, M. Bach, and K. Dietmayer, "2D Car Detection in Radar Data with PointNets," in *2019 IEEE Intelligent Transportation Systems Conference, ITSC 2019*, 2019.

[64] Y. Zhang and M. Rabbat, "A Graph-CNN for 3D Point Cloud Classification," *ICASSP, IEEE Int. Conf. Acoust. Speech Signal Process. - Proc.*, vol. 2018–April, pp.

6279–6283, 2018.

[65] K. Zhang, M. Hao, J. Wang, C. W. de Silva, and C. Fu, “Linked dynamic graph CNN: Learning on point cloud via linking hierarchical features,” *arXiv Prepr. arXiv1904.10014*, 2019.

[66] A. Gamal, A. Wibisono, S. B. Wicaksono, M. A. Abyan, N. Hamid, H. A. Wisesa, W. Jatmiko, and R. Ardhianto, “Automatic LIDAR building segmentation based on DGCNN and euclidean clustering,” *J. Big Data*, 2020.

[67] Q. Lu, C. Chen, W. Xie, and Y. Luo, “PointNGCNN: Deep convolutional networks on 3D point clouds with neighborhood graph filters,” *Comput. Graph.*, 2020.

[68] C. Wang, P. Gong, and L. Zhang, “STPOINTGCN: Spatial Temporal Graph Convolutional Network for Multiple People Recognition Using Millimeter-Wave Radar,” in *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, 2022.

[69] S. Hazra, H. Feng, G. N. Kiprit, M. Stephan, L. Servadei, R. Wille, R. Weigel, and A. Santra, “Cross-modal Learning of Graph Representations using Radar Point Cloud for Long-Range Gesture Recognition,” in *Proceedings of the IEEE Sensor Array and Multichannel Signal Processing Workshop*, 2022.

[70] S. Wang, Y. Liu, L. Wang, Y. Sun, and B. Yin, “PASIFTNet: Scale-and-Directional-Aware Semantic Segmentation of Point Clouds,” *CAD Comput. Aided Des.*, 2023.

[71] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, “Attention is all you need,” in *Advances in Neural Information Processing Systems*, 2017.

[72] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. Le Scao, S. Gugger, M. Drame, Q. Lhoest, and A. Rush, “Transformers: State-of-the-Art Natural Language Processing,” 2020.

[73] A. Y. Yldz, E. Koc, and A. Koc, "Multivariate Time Series Imputation With Transformers," *IEEE Signal Process. Lett.*, 2022.

[74] B. Jiang, K. Zhao, and J. Tang, "RGTransformer: Region-Graph Transformer for Image Representation and Few-Shot Classification," *IEEE Signal Process. Lett.*, 2022.

[75] P. Wu, Bichen and Xu, Chenfeng and Dai, Xiaoliang and Wan, Alvin and Zhang, Peizhao and Yan, Zhicheng and Tomizuka, Masayoshi and Gonzalez, Joseph and Keutzer, Kurt and Vajda, "Visual transformers: Token-based image representation and processing for computer vision," *arXiv Prepr. arXiv2006.03677*, 2020.

[76] C. Zhang, H. Wan, X. Shen, and Z. Wu, "PVT: Point-voxel transformer for point cloud learning," *Int. J. Intell. Syst.*, 2022.

[77] H. Zhao, L. Jiang, J. Jia, P. Torr, and V. Koltun, "Point Transformer," in *Proceedings of the IEEE International Conference on Computer Vision*, 2021.

[78] C. Zhang, H. Wan, X. Shen, and Z. Wu, "PatchFormer: An Efficient Point Transformer with Patch Attention," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2022.

[79] K. Mazur and V. Lempitsky, "Cloud Transformers: A Universal Approach To Point Cloud Processing Tasks," in *Proceedings of the IEEE International Conference on Computer Vision*, 2021.

[80] G. Hess, J. Jaxing, E. Svensson, D. Hagerman, C. Petersson, and L. Svensson, "Masked Autoencoder for Self-Supervised Pre-training on Lidar Point Clouds," in *Proceedings - 2023 IEEE/CVF Winter Conference on Applications of Computer Vision Workshops, WACVW 2023*, 2023.

[81] D. Xu, Runsen and Wang, Tai and Zhang, Wenwei and Chen, Runjian and Cao, Jinkun and Pang, Jiangmiao and Lin, "MV-JAR: Masked Voxel Jigsaw and

Reconstruction for LiDAR-Based Self-Supervised Pre-Training," *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, pp. 13445--13454, 2023.

[82] Y. Pang, W. Wang, F. E. H. Tay, W. Liu, Y. Tian, and L. Yuan, "Masked Autoencoders for Point Cloud Self-supervised Learning," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2022.

[83] M.-H. Guo, J.-X. Cai, Z.-N. Liu, T.-J. Mu, R. R. Martin, and S.-M. Hu, "Pct: Point cloud transformer," *Comput. Vis. Media*, vol. 7, no. 2, pp. 187–199, 2021.

[84] Z. Huang, Z. Zhao, B. Li, and J. Han, "LCPFormer: Towards Effective 3D Point Cloud Analysis via Local Context Propagation in Transformers," *IEEE Trans. Circuits Syst. Video Technol.*, 2023.

[85] E. Schubert, J. Sander, M. Ester, H. P. Kriegel, and X. Xu, "DBSCAN revisited, revisited: Why and how you should (still) use DBSCAN," *ACM Trans. Database Syst.*, 2017.

[86] N. Scheiner, O. Schumann, F. Kraus, N. Appenrodt, J. Dickmann, and B. Sick, "Off-the-shelf sensor vs. experimental radar - How much resolution is necessary in automotive radar classification?," in *Proceedings of 2020 23rd International Conference on Information Fusion, FUSION 2020*, 2020.

[87] V. and others Pedregosa, F and Varoquaux, G and Gramfort, A and Michel, V and Thirion, B and Grisel, O and Blondel, M and Prettenhofer, P and Weiss, R and Dubourg, "Comparing Different Clustering Algorithms on Toy Datasets," 2022.

[88] D. Kellner, J. Klappstein, and K. Dietmayer, "Grid-based DBSCAN for clustering extended objects in radar data," in *IEEE Intelligent Vehicles Symposium, Proceedings*, 2012.

[89] A. Bartsch, F. Fitzek, and R. H. Rasshofer, "Pedestrian recognition using automotive radar sensors," *Adv. Radio Sci.*, 2012.

[90] R. Prophet, M. Hoffmann, A. Ossowska, W. Malik, C. Sturm, and M. Vossiek, "Pedestrian Classification for 79 GHz Automotive Radar Systems," in *IEEE Intelligent Vehicles Symposium, Proceedings*, 2018.

[91] S. Lee, Y. J. Yoon, J. E. Lee, and S. C. Kim, "Human-vehicle classification using feature-based SVM in 77-GHz automotive FMCW radar," *IET Radar, Sonar Navig.*, 2017.

[92] X. Cai, M. Giallorenzo, and K. Sarabandi, "Machine Learning-Based Target Classification for MMW Radar in Autonomous Driving," *IEEE Trans. Intell. Veh.*, 2021.

[93] N. Scheiner, N. Appenrodt, J. Dickmann, and B. Sick, "Radar-based road user classification and novelty detection with recurrent neural network ensembles," in *IEEE Intelligent Vehicles Symposium, Proceedings*, 2019.

[94] O. Schumann, C. Wöhler, M. Hahn, and J. Dickmann, "Comparison of random forest and long short-term memory network performances in classification tasks using radar," in *2017 Sensor Data Fusion: Trends, Solutions, Applications (SDF)*, 2017, pp. 1–6.

[95] N. Scheiner, F. Kraus, N. Appenrodt, J. Dickmann, and B. Sick, "Object detection for automotive radar point clouds – a comparison," *AI Perspect.*, 2021.

[96] S. Zhang, H. Tong, J. Xu, and R. Maciejewski, "Graph convolutional networks: a comprehensive review," *Comput. Soc. Networks*, 2019.

[97] Y. Zhou, Yi and Liu, Lulu and Zhao, Haocheng and Lopez-Benitez, Miguel and Yu, Limin and Yue, "Towards deep radar perception for autonomous driving: Datasets, methods, and challenges," *Sensors*, vol. 22, p. 4208, 2022.

[98] Y. Cheng, J. Su, H. Chen, and Y. Liu, "A new automotive radar 4D point clouds detector by using deep learning," in *ICASSP, IEEE International Conference on*

Acoustics, Speech and Signal Processing - Proceedings, 2021.

APPENDIX

APPENDIX A: Datasets Class Mapping Tables

Table A.1 Proposed dataset arrangements involve combining various categories from the RadarScenes dataset to create classes customized by the user.

Category	**Benchmark dataset**
Car	Vehicle
Large vehicle	-
Truck	Large Vehicle
Bus	-
Train	-
Bicycle	Cycle
Motorized two-wheeler,	-
Pedestrian,	Pedestrian
Pedestrian group	Pedestrian Group
Animal	-
Other	None

Table A.2 Proposed dataset arrangements involve combining various categories from the nuScenes dataset to create classes customized by the user.

Category	**Benchmark dataset**
None	None
animal	Pedestrian
human.pedestrian.adult	Pedestrian
human.pedestrian.child	Pedestrian
human.pedestrian.construction_wo	Pedestrian
human.pedestrian.personal_mobili	Cycle
human.pedestrian.police_officer	Pedestrian
human.pedestrian.stroller	Pedestrian
human.pedestrian.wheelchair	Pedestrian
movable_object.barrier	None
movable_object.debris	None

movable_object.pushable_pullable	None
movable_object.trafficcone	None
static_object.bicycle_rack	None
vehicle.bicycle	Cycle
vehicle.bus.bendy	Vehicle
vehicle.bus.rigid	Vehicle
vehicle.car	Vehicle
vehicle.construction	Vehicle
vehicle.emergency.ambulance	Vehicle
vehicle.emergency.police	Vehicle
vehicle.motorcycle	Cycle
vehicle.trailer	Vehicle