RESEARCH ARTICLE

Real-Time Object Detection Using a Lightweight Two-Stage Detection Network with Efficient Data Representation

Shaohuang Wang[1,*]

[1] Cardiff University, Cardiff CF10 3AT, UK

Abstract

In this paper, we introduce a novel fast object detection framework, designed to meet the needs of real-time applications such as autonomous driving and robot navigation. Traditional processing methods often trade-off between accuracy and processing speed. To address this issue, we propose a hybrid data representation method that combines the computational efficiency of voxelization with the detail capture capability of direct data processing to optimize overall performance. Our detection framework comprises two main components: a Rapid Region Proposal Network (RPN) and a Refinement Detection Network (RefinerNet). The RPN is used to generate high-quality candidate regions, while the RefinerNet performs detailed analysis on these regions to improve detection accuracy. Additionally, we have implemented a variety of network optimization techniques, including lightweight network layers, network pruning, and model quantization, to increase processing speed and reduce computational resource consumption. Extensive testing on the KITTI and the NEXET datasets has proven the effectiveness of our method in enhancing the accuracy of object detection and real-time processing speed. The experimental results show that, compared to existing technologies, our method performs exceptionally well across multiple evaluation metrics, especially in meeting the stringent requirements of real-time applications in terms of processing speed.

Keywords: object detection, real-time, refinement, network optimization, pruning

1 Introduction

In fields such as autonomous driving [1], person re-identification [2–5], face perception [6], and virtual reality [7, 8], 3D object detection technology plays an essential role. It allows systems to accurately understand and interpret the three-dimensional space and objects within their surrounding environment. Especially in autonomous vehicles, precise 3D object detection is indispensable for safe navigation and obstacle avoidance. Although various technologies and methods have been proposed to handle this task, achieving real-time processing while ensuring high accuracy remains a challenge.

Academic Editor:
Teerath Kumar

Submitted: 05 December 2023
Accepted: 16 April 2024
Published: 20 April 2024

Vol. 1, **No.** 1, 2024.
10.62762/TETAI.2024.320179

***Corresponding author:**
✉ Shaohuang Wang
wangs130@cardiff.ac.uk

Citation
Wang, S. (2024). Real-Time Object Detection Using a Lightweight Two-Stage Detection Network with Efficient Data Representation. *IECE Transactions on Emerging Topics in Artificial Intelligence*, 1(1), 17–30.

Traditional 3D object detection methods [9, 10] mainly rely on point cloud data obtained from sensors such as LiDAR. These methods include converting point cloud data into a voxel grid (voxelization) or projecting it onto a two-dimensional plane (such as a bird's eye view or front view). Voxelization creates a three-dimensional grid and counts the number of points within each grid cell, transforming unordered point cloud data into structured volumetric data, which facilitates processing using convolutional neural networks (CNNs). However, this approach often involves high-dimensional data, leading to increased computational and storage burdens. Moreover, the process of spatial quantization during voxelization may result in the loss of detailed information, especially at the edges of objects or with smaller objects.

On the other hand, methods that directly process point clouds, such as the renowned PointNet [11] and its subsequent work PointNet++ [12], are capable of learning features directly from the raw point cloud data, avoiding the loss of information in the preprocessing steps. These models utilize global feature descriptors of point clouds, enabling the network to understand the structure and shape of the point clouds holistically. However, these models usually require processing the entire point cloud, demanding substantial computational resources that may not meet the needs of real-time applications. As 3D point cloud processing technology continues to advance, existing methods have seen significant improvements in processing speed and accuracy, but they still face key challenges. For instance, despite the wide attention PointNet and PointNet++ have received in the academic community, their computational and storage requirements remain considerable when processing large-scale data. These methods generally rely on deep neural networks to learn features directly from point clouds, a process that demands a significant amount of computational resources, especially when dealing with large volumes of data or uneven point densities.

To address these challenges, researchers have tried various optimization strategies. For instance, some studies have introduced more efficient neural network architectures, such as Sparse Convolutional Networks. These networks significantly reduce unnecessary computational loads by performing calculations only at the non-empty positions of the data, thus increasing processing speed. Additionally, a stratified processing approach has been proposed, where point clouds are processed at different resolution levels. This method preserves important local details while also capturing global structural information, increasing processing efficiency without compromising accuracy.

Another optimization strategy involves the use of effective data dimensionality reduction and preprocessing techniques to reduce the complexity of the input data. For instance, preprocessing point clouds using methods like Principal Component Analysis (PCA) or autoencoders can extract the most representative features before feeding them into the neural network. This approach can alleviate the network's load to some extent and accelerate the training and inference processes. Moreover, multimodal fusion represents a significant direction for enhancing the performance of 3D point cloud processing. By combining data from different sensors, such as integrating LiDAR point clouds with RGB images, complementary information can be obtained from different perspectives and sensing mechanisms, enhancing the model's understanding of the environment. This fusion not only improves the accuracy of object detection but also strengthens the model's ability to discriminate between different object characteristics in complex environments.

Despite the progress made by existing methods, there remain numerous technical challenges and room for improvement in processing large-scale 3D point cloud data efficiently and in real time. Consequently, researching and developing more efficient algorithms and technologies to meet the dual demands of speed and accuracy in practical applications is a current hotspot in the field of 3D point cloud processing. In this study, we propose an efficient convolutional neural network framework for rapid object detection within 3D point clouds. Our research significantly enhances the efficiency and accuracy of 3D point cloud object detection through innovative data processing and network design strategies. Our main contributions are not only reflected in technical innovations but are also demonstrated through a series of rigorous experiments validating the effectiveness of our methods.

In addition to these improvements, we explore novel loss functions that account for both object localization and classification errors, thereby refining object detection precision. We also introduce a more dynamic data augmentation technique, which helps the network generalize better to unseen scenarios and varied environmental conditions. These enhancements, combined with the optimized network

architecture, provide a comprehensive solution for real-time 3D object detection, addressing both computational constraints and accuracy requirements. Through extensive testing on benchmark datasets, we demonstrate the scalability and practicality of our approach in real-world applications, particularly in autonomous driving and robotics. Here is a further expansion of our contributions, including detailed descriptions from an experimental perspective:

1 **Efficient Data Representation:** Our proposed data representation method combines the computational efficiency of voxelization with the precision of direct point processing. This hybrid approach not only maintains high efficiency in data handling but also significantly reduces information loss typically associated with traditional voxelization. In our experiments, we compared the performance of pure voxelization, pure point cloud processing, and our hybrid method across various complex scenarios. The results indicate that our method surpasses traditional approaches in both processing speed and accuracy.

2 **Two-Stage Detection Network:** We designed a two-stage network that initially extracts features through a rapid candidate region proposal network, followed by a refinement network for precise object localization and classification. This staged processing approach allows us to use coarser features for quick filtering in the first stage while focusing on the detailed processing of candidate regions in the second stage. In our experiments, we demonstrated how this method improves detection accuracy compared to single-stage processing, particularly in terms of precision at object boundaries.

3 **Real-Time Processing Capability:** In designing the network, we particularly focused on computational efficiency to ensure that the model could operate within the constraints of limited computational resources. By utilizing GPU acceleration and algorithm optimization, our model achieves real-time processing speeds while maintaining high accuracy. In our experiments conducted on the standard KITTI dataset, we extensively tested the model, achieving industry-leading levels on conventional evaluation metrics and setting new standards in processing speed. This demonstrates the model's capability to handle real-time applications effectively, making it highly suitable for scenarios requiring immediate response, such as in autonomous driving systems.

Through these experiments, we have demonstrated the high practical value and technological advancement of our method in real-world application scenarios. The experimental results not only showcase the effectiveness of our approach but also highlight its potential in applications requiring rapid and reliable 3D environmental perception, such as autonomous driving and other similar technologies. This underscores our method's capability to enhance operational efficiency and accuracy in dynamic and complex environments, solidifying its relevance and utility in cutting-edge technological implementations.

2 Related Work

The field of fast object detection continues to evolve with the introduction of various innovative network architectures and optimization techniques aimed at further enhancing detection speed and accuracy, particularly on resource-constrained devices. We will detail these advancements from three perspectives: single-step detection networks, multi-scale and feature fusion networks, and networks optimized for mobile devices.

2.1 Single-Step Detection Networks

These networks significantly speed up the detection process by directly predicting object classes and locations from the input image, bypassing the traditional step of extracting candidate regions. The YOLO series of networks exemplifies single-step detection. YOLO [13] divides the image into multiple grids, with each grid cell directly predicting bounding boxes and class probabilities. A key advantage of YOLO is its speed, enabling near-real-time object detection, making it particularly well-suited for video stream processing. Over the years, YOLO has seen continuous updates to its architecture, improving both accuracy and processing speed through deeper networks, optimized anchor boxes, and advanced loss functions. YOLO's later versions incorporate features like multi-scale detection, which allow the network to detect objects of various sizes more effectively, further enhancing its applicability in dynamic and complex environments.Another network, CenterNet [14], offers a simpler yet efficient detection approach by predicting the center points and sizes of objects, rather than traditional bounding box regression. This method significantly reduces network

complexity and eliminates the need for region proposal and post-processing steps, thus improving detection speed without sacrificing accuracy. This direct approach to object localization has proven particularly beneficial in real-time applications where efficiency is critical.YOLACT [15], though primarily designed for instance segmentation, also adopts a single-step detection paradigm. By breaking down the segmentation task into prototype mask generation and instance-specific coefficient prediction, YOLACT achieves real-time instance segmentation. Its lightweight design and efficient mask prediction make it suitable for complex real-time scenarios, such as dynamic environments with overlapping objects.RefineDet [16], another single-step detection network, introduces a two-stage refinement module, where initial candidate boxes are first generated and then refined for more accurate detection. While this introduces slightly more computation than typical single-stage detectors, the refinement step ensures a higher detection accuracy, balancing speed and precision effectively.

In summary, single-step detection networks like YOLO [13], CenterNet [14], YOLACT [15], and RefineDet [16] have revolutionized the field of object detection by significantly improving speed and efficiency. These networks bypass traditional region proposal methods, making them suitable for real-time applications such as video processing and autonomous systems. YOLO's grid-based approach offers a balance between speed and accuracy, while CenterNet simplifies the detection process by focusing on center points, further reducing complexity. YOLACT expands on these capabilities by integrating instance segmentation, enabling it to perform well in real-time segmentation tasks. RefineDet introduces a refinement process that enhances accuracy without compromising much on speed.

However, despite their advancements, these models still face challenges, particularly in handling smaller objects or highly dense scenes where multiple overlapping objects can reduce detection accuracy. Furthermore, while speed is a major strength, maintaining high precision in complex environments with varying lighting, occlusion, or extreme object scales remains a challenge. As a result, future research should focus on addressing these limitations by developing more robust architectures that can balance both speed and high accuracy across diverse and challenging scenarios.

2.2 Multi-Scale and Feature Fusion Networks

These networks leverage image features from different scales, utilizing feature fusion technology to enhance detection accuracy, particularly when dealing with objects of varying sizes. SSD [17] predicts the presence and location of objects across multiple feature maps simultaneously, effectively handling targets of different sizes. SSD employs independent convolutional layers at multiple predetermined scales to predict bounding boxes, thereby significantly improving the detection capability for smaller objects. EfficientDet [18] employs an innovative Bi-directional Feature Pyramid Network (BiFPN) that effectively merges features from different layers. This method optimizes feature utilization and reduces the consumption of computational resources. EfficientDet uses compound scaling technology to balance the network's width, depth, and input resolution, achieving an efficient and precise balance. To manage targets of various sizes and fully leverage multi-scale information in images, multi-scale and feature fusion networks adopt strategies that extract and merge features from different levels. These networks enhance the model's expressive power and adaptability by integrating features of different resolutions. The Feature Pyramid Network (FPN) [19] is a classic architecture for multi-scale feature fusion, performing excellently in natural image object detection tasks. FPN constructs a top-down architecture where semantic information from higher layers gradually percolates to lower layers, enriching each scale with abundant semantic and positional information. This approach is particularly suited for detecting objects that vary significantly in size. PANet [20] builds on the foundation of FPN by adding a bottom-up information pathway to further enhance the semantic capability of lower-level features. By enhancing the flow of features, PANet provides a more comprehensive feature fusion, thereby improving the model's performance in detecting small objects.

2.3 Networks Optimized for Mobile Devices

These networks are specifically designed to operate in resource-constrained environments by streamlining network architecture and reducing computational demands to achieve rapid detection. MobileNets [21] utilize depthwise separable convolutions, which significantly reduce the model's size and computational requirements. This makes MobileNets particularly suitable for running on mobile and edge devices while maintaining reasonable detection accuracy. SqueezeDet [22] is a

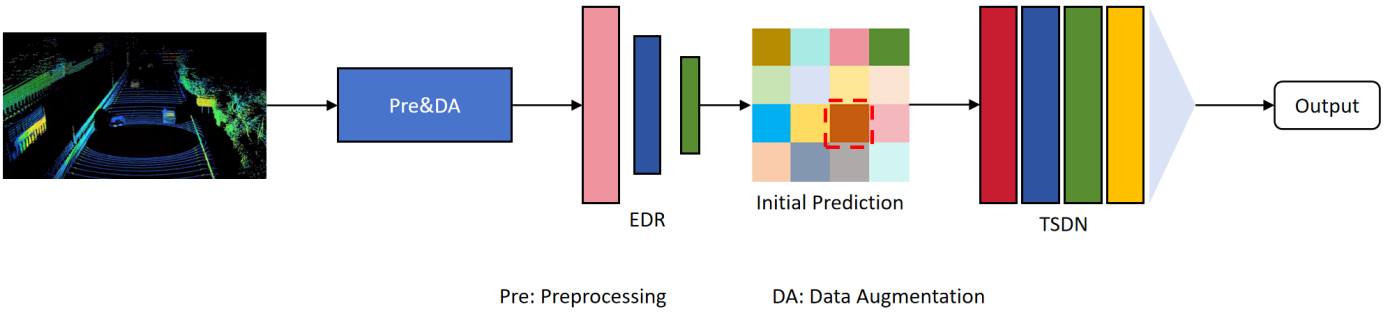

Figure 1. Overall Structure of the Model.

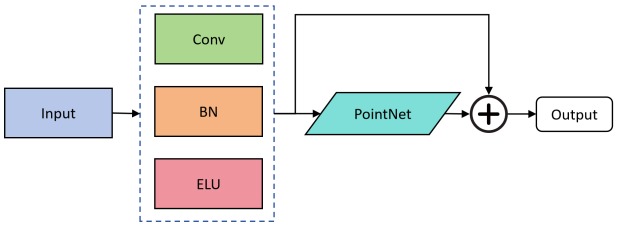

Figure 2. Structure of the EDR.

lightweight network designed for low-power devices that combines the rapid detection capabilities of convolutional networks with the need of a compressed network structure. It drastically reduces the model's parameter count and computational needs while maintaining high performance in detection tasks. PeleeNet [23] is another lightweight network designed for mobile devices, employing a densely connected strategy similar to DenseNet but with significant optimizations in computational load and parameter count. The aim of PeleeNet is to provide sufficient detection accuracy with sufficiently low latency, making it suitable for operation on lower-performance devices. ThunderNet [24] continues to advance object detection performance on edge devices through its SNet design and Context Enhancement Module (CEM). SNet serves as the backbone network, focusing on enhancing computational efficiency, while CEM leverages contextual information to improve feature expressiveness, thereby enhancing the accuracy of detecting small objects. These advancements collectively contribute to the viability of deploying advanced object detection technologies in environments where computational resources are limited, such as in embedded systems or mobile applications. This capability is crucial for applications that require real-time processing without access to powerful computing infrastructure, such as autonomous vehicles, drones, or mobile applications that require immediate response and interaction with the environment.

3 Methodology

The overall architecture of our method is illustrated in Figure 1, which primarily consists of two modules: the Efficient Data Representation (EDR) module and the Two-Stage Detection Network(TSDN) module. The process begins with the raw data being input into a preprocessing and data augmentation module, which serves to enhance the diversity of the data. The enhanced data is then fed into the EDR module for initial refinement. Subsequently, the data processed by the EDR module is input into our TSDN module, where it undergoes further refinement to extract more detailed features. These refined features facilitate the production of the final model results. Below, we provide a detailed description of our method.

3.1 Efficient Data Representation

Our efficient data representation method comprises two main steps: efficient voxelization and detail-preserving point feature extraction. The purpose of voxelization is to reduce the scale and dimensions of the data, making it more suitable for rapid processing. Point feature extraction, on the other hand, aims to capture the fine structures within each voxel, which is crucial for maintaining the accuracy of object detection. The network model architecture is illustrated in Figure 2

Voxelization Process: Voxelization involves dividing the continuous point cloud space into a regular grid and aggregating point cloud information within each grid cell. The aggregation can be achieved through various statistical measures such as mean, maximum values, or a combination of features. We define the voxelization process as follows:

$$V = \frac{1}{|C|} \sum_{p_i \in C} \phi(p_i) \tag{1}$$

$$F_j = \sum_{p_i \in V_j} \omega(p_i) \cdot \phi(p_i) \tag{2}$$

where $\omega(p_i)$ is the weights learned during training. In the voxelization process, C represents the set of points within the voxel, and $\phi(p_i)$ is the feature extraction function applied to point p_i. This function can involve simple attributes such as coordinates, intensity, and color, or more complex properties like local curvature or normal vectors.

Point Feature Extraction: To further enhance the expressive power of voxelized data, we employ a lightweight PointNet network within each voxel to extract more comprehensive local features. This approach ensures that even if some detail is lost during the voxelization process, these details can still be preserved through the learned features:

$$F_j' = \sum_{p_i \in V_j} \alpha_{p_i} \cdot \text{ELU}(\text{BN}(\text{Conv}(\phi(p_i)))) \quad (3)$$

Here, F_j' represents the advanced features extracted from voxel (x,y,z). These features include not only geometric information but also deeper contextual information learned by the PointNet model. This combination provides a rich representation that enhances the model's ability to accurately recognize and classify objects within the 3D space.

3.2 Two-Stage Detection Network

The two-stage detection network is designed to enhance detection speed while ensuring accuracy, with the RPN structure and the RefinerNet structure as its core module. The model structure is illustrated in Figure 3.

First Stage: Rapid Region Proposal Network (RPN). The goal of this stage is to rapidly generate high-quality candidate regions from the hybrid data representation. The RPN utilizes a series of optimized convolutional layers to process voxelized point cloud data, which are specifically designed to capture spatial structures and reduce computational complexity. The proposal of candidate regions is based on a sliding window approach, where the point cloud data within the window is analyzed by a small neural network to assess the presence of potential target objects. The output of the RPN is a set of bounding boxes, each with a score indicating the likelihood of containing an object. This process can be represented by the following formula:

$$Regions = Softmax(DenseConv(Flatten(V))) \quad (4)$$

In this context, V represents the voxelized data that has undergone preliminary processing. Regions denote the collection of candidate areas. These elements are integral to the RPN's functionality, enabling it to efficiently filter and prioritize areas for further analysis in the object detection process.

Second Stage: Refinement Detection Network (RefinerNet). Once candidate regions are successfully identified, the second stage aims to perform a deeper analysis of these areas to refine the localization and classification of targets. This stage leverages local features extracted from PointNet along with preliminary candidate region information provided by the RPN. RefinerNet applies a more complex neural network structure to each candidate region, which includes deep convolutional networks and attention mechanisms to ensure accurate identification of target details from the refined features.

The refinement stage focuses on enhancing the precision of object detection, particularly in scenarios involving object edges and partial occlusions. This process can be described as follows:

$$F' = ELU(BN(Conv(Regions, F))) \quad (5)$$

$$result = PointNet(F_j', F') \quad (6)$$

In this context, F′ represents the collection of features extracted from the first stage, which includes spatial and depth features used to precisely describe the contents of the candidate regions. These features are crucial to effectively refine the detection and classification of objects within the specified areas, ensuring higher accuracy and a more detailed understanding of the scene. Finally, we use a lightweight PointNet to get the detection results.

Through this two-stage detection network, we effectively combine the advantages of rapid detection and precise detection. In the first stage, by swiftly filtering through a large number of spatial areas to identify potential candidate regions, we significantly reduce the computational resources required for subsequent processing. In the second stage, resources are concentrated on in-depth analysis to ensure the accuracy and reliability of detection. This design enables our network to maintain real-time processing capabilities while also meeting the demands for high-precision detection. This approach not only enhances performance but also optimizes operational efficiency, making it particularly suited for applications where both speed and accuracy are critical.

3.3 Real-time Processing Capability

To ensure that our 3D point cloud object detection framework operates efficiently in real-time

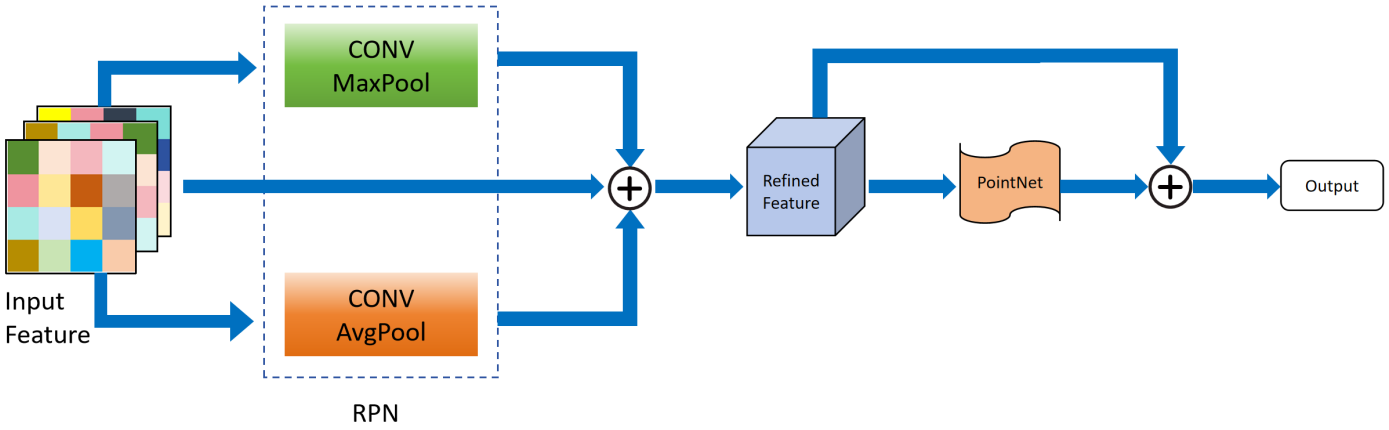

Figure 3. Structure of the TSDN.

applications, we have implemented several measures in the design and optimization of its real-time processing capabilities. These measures include optimizing the network architecture and applying computational acceleration techniques.

Network Architecture Optimization. In designing the network architecture, we specifically focused on reducing computational load and memory usage while maintaining high detection performance. The strategies we adopted include:

- **Lightweight Network Layers:** We have opted for depthwise separable convolutions instead of traditional convolutions. This type of convolution significantly reduces the model's parameter count and computational demands. Furthermore, depthwise separable convolutions decompose standard convolution operations, reducing the complexity of the model and accelerating data processing speed. This modification not only makes the network lighter and faster but also maintains an acceptable level of accuracy, making it highly suitable for real-time applications where speed is critical.

- **Network Pruning:** By applying network pruning techniques, we remove non-essential neurons and connections, thereby reducing the complexity of the model. This not only enhances the running speed but also decreases the model's storage requirements, making it more suitable for deployment on resource-constrained devices. Network pruning effectively trims redundant elements of the neural network without significantly impacting its predictive performance, optimizing the balance between efficiency and accuracy.

Computational Acceleration Techniques. In order to further enhance processing speed, we have employed the latest computational acceleration techniques:

- **GPU Acceleration:** Leveraging the power of Graphics Processing Units (GPUs) [27, 28] to handle parallel computations effectively. GPUs excel in dealing with matrix and vector operations common in deep learning, which allows for much faster processing than CPU-based computations. This is particularly beneficial for training and deploying deep neural networks where large amounts of data must be processed simultaneously.

- **Model Quantization:** Reducing the precision of the model's parameters from floating-point to lower-bit representations, which can significantly decrease the model size and speed up inference without a substantial loss in accuracy.

4 Experiments

To validate the effectiveness and efficiency of our approach, we conducted experiments on 3D object detection using the challenging datasets KITTI [25] and NEXET [26]. Additionally, we carried out further ablation studies concerning our method. These experiments were designed to test various aspects of our detection framework under different scenarios and conditions.

4.1 Experiment Setup

Training Details. Under default conditions, the model was trained using four NVIDIA A30 GPUs, with each GPU handling 2 point clouds, resulting in a total of 8 point clouds processed concurrently. This setup ensures efficient training by distributing the computational load across multiple high-performance

GPUs. We utilized the Adam optimizer [27] with a learning rate of 0.001 to optimize our network parameters. This choice of optimizer and learning rate helps in achieving a good balance between fast convergence and training stability. Additionally, Batch Normalization [28] was applied following each parameter layer to help stabilize the learning process by normalizing the inputs of each layer. This method is particularly effective in accelerating the training phase and improving performance by reducing internal covariate shifts. A weight decay of 0.0001 was implemented in both networks to regularize the model and prevent overfitting. Weight decay works by adding a penalty to the loss function based on the magnitude of the weights, which encourages the model to maintain smaller weights and thus simpler models. Detailed parameters and configurations of the model training are listed in Tables 1 and 2. These tables include specific settings such as the number of layers, activation functions, and other relevant hyperparameters that define the architecture and training dynamics of the model. This detailed documentation is crucial for replicability and understanding the model's structure and behavior under training conditions.

Data Augmentation. Data augmentation is a critical technique, especially when the amount of available data is limited, to prevent overfitting during training. For each frame in our dataset, we employ multiple data augmentation methods to diversify the training examples: Random Flipping: We apply left-right random flipping to the frames, which mirrors the point cloud data across the vertical axis. This helps the model learn to recognize objects regardless of their orientation.

Random Scaling: Each frame is randomly scaled by a factor uniformly sampled from the range 0.95 to 1.05. This simulates the effect of objects appearing closer or further away from the sensor and encourages scale invariance in the model.

Random Rotation: We perform random rotation on the entire scene for point clouds around the origin, with a degree sampled uniformly from -30° to 30°. This introduces rotational variance into the dataset, promoting the model's ability to detect objects at different angles.

Additionally, we randomly perturb each ground-truth bounding box and its corresponding internal points by applying random transformations.

Random Shift: The shift for the bounding box is sampled from a predefined range for both the X and Y axes, and a different range for the Z axis, to account for the typical variations in the environment.

Random Rotation Around the Z-axis: The rotation is uniformly sampled from the range -10π to 10π, which equates to a full 360-degree rotation potential. This rotation augments the dataset with a wide variety of angular perspectives, challenging the model to maintain accuracy despite changes in orientation.

These data augmentation techniques collectively form a robust strategy to enhance the generalization capability of the 3D object detection framework, preparing it for effective performance in real-world scenarios.

4.2 Dataset and Evaluation Metrics

Dataset. We selected two classic datasets to validate the effectiveness of our model under various scenarios: the KITTI and NEXET datasets.

The KITTI dataset provides 7,481 images and point clouds for training and 7,518 for testing. It is important to note that for the evaluation of the test subset and comparison with other methods, we only submit our results to the evaluation server. Following the paradigm set in [6, 25], we divide the dataset into a training set (with 3,712 images and point clouds) containing around 14,000 car annotations and a validation set (with 3,769 images and point clouds). Ablation studies are also conducted on this split. Whereas for evaluation on the test set, we train our model on the entire train set consisting of 7k point clouds.

The NEXET dataset is a large-scale video dataset extensively used in the field of autonomous driving. It covers over 77 countries, more than 1,400 cities, three lighting conditions (day, night, twilight), four seasons, multiple road conditions (urban, rural, highway, residential, and even desert roads), and various weather conditions (clear, foggy, rainy, snowy). The dataset provides high-fidelity video clips with rich annotation information, making NEXET an ideal choice for assessing our method.

In the above text, references such as[10][9] presumably relate to prior research or specific methodologies that are documented within the broader context of this work. The dataset partitioning and use of full training sets for final test evaluations are standard practices to maximize the learned model's performance and

Table 1. Experimental environment demonstrated.

Parameter	Configuration
CPU	Intel Core i9-12700KF
GPU	NVIDIA GeForce RTX A30 (24 GB)
CUDA version	CUDA 11.7
Python version	Python 3.9.13
Deep learning framework	Pytorch 2.0.0
Operating system	Ubuntu 22.04.2

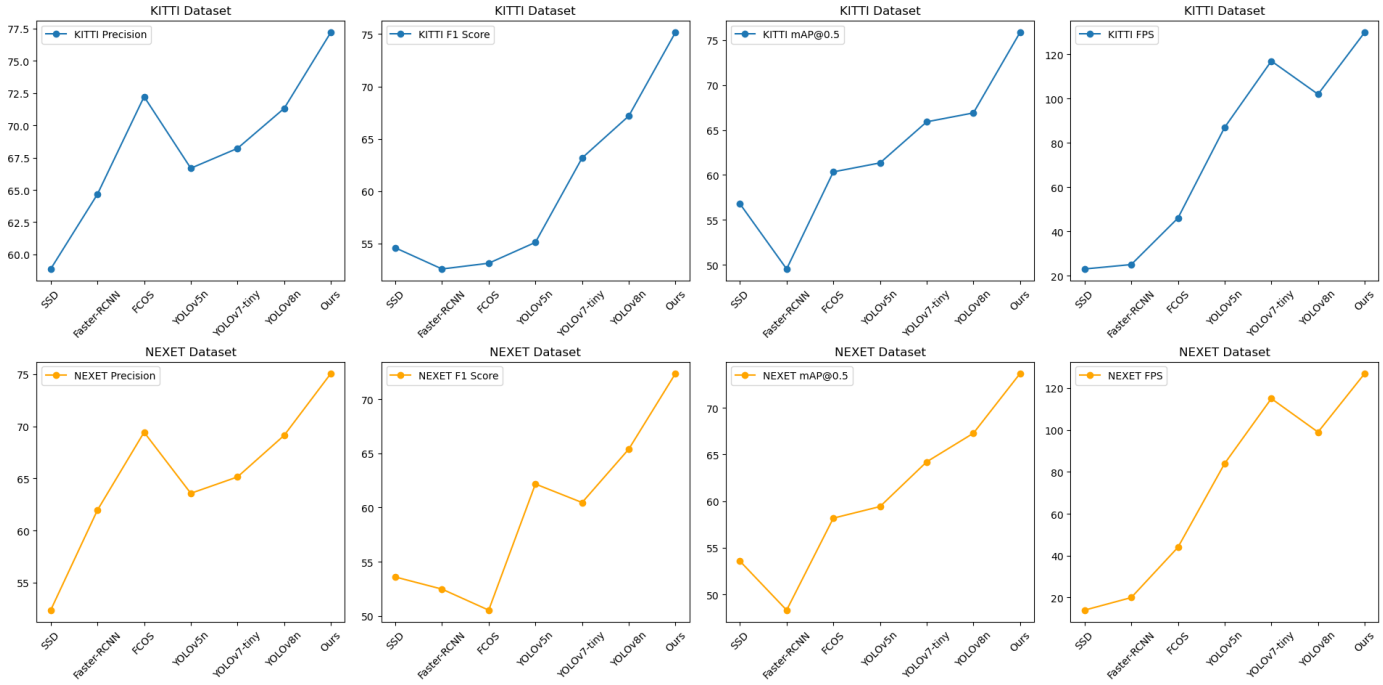

Figure 4. Visualization of results.

Table 2. Model Parameter Settings

Parameter	Value
Learning Rate	0.001
Optimizer	Adam
Batch Size	16
Weight Decay	0.0001
Training Epochs	350
Activation Function	ELU
Number of Layers	252
Loss Function	MSE
Early Stop	True

generalizability. NEXET's diverse conditions offer a rigorous testing ground to showcase the robustness of the proposed object detection method across a wide array of real-world driving scenarios.

Evaluation Metrics. In our experiments, we adopt the following evaluation methods to comprehensively consider the model's performance:

Precision. Precision is the ratio of correctly predicted positive observations to the total predicted positives. It focuses on the quality of the model's outputs. The formula for calculating precision is:

$$Precision = \frac{TruePositives}{TurePositives + FalsePositives} \quad (7)$$

where, True Positives refer to the number of positive samples that were correctly predicted, while False Positives refer to the number of samples that were incorrectly predicted as positive. Precision is particularly useful when the costs of false positives are high. In the context of object detection, a high precision score indicates that when the model predicts an object is present, it is likely to be correct.

Recall. Recall is the proportion of actual positive samples that are correctly identified as such by the model, focusing on the completeness of the model's output. The formula for calculating recall is:

$$Recall = \frac{TruePositives}{TruePositives + FalseNegatives} \quad (8)$$

Table 3. Comparison of different methods on the KITTI and the NEXET datasets.

Methods	KITTI				NEXET			
	Precision	F1 Score	mAP@0.5	FPS	Precision	F1 Score	mAP@0.5	FPS
SSD[17]	58.87	54.59	56.79	23	52.36	53.59	53.58	14
Faster-RCNN[29]	64.66	52.57	49.54	25	61.95	52.47	48.33	20
FCOS[30]	72.23	53.12	60.34	46	69.43	50.53	58.18	44
YOLOv5n[31]	66.68	55.11	61.34	87	63.57	62.17	59.42	84
YOLOv7-tiny[32]	68.22	63.17	65.91	117	65.14	60.46	64.21	115
YOLOv8n[33]	71.32	67.18	66.90	102	69.13	65.39	67.27	99
Ours	77.22	75.17	75.87	130	75.07	72.37	73.67	127

Table 4. Ablation Study Results on the KITTI and the NEXET Datasets

	Module		KITTI				NEXET			
	EDR	TSDN	Precision	F1 Score	mAP@0.5	FPS	Precision	F1 Score	mAP@0.5	FPS
(1)			65.51	69.62	69.73	65	70.33	67.44	68.69	54
(2)	✓		66.45	70.21	71.71	72	73.32	67.98	70.52	65
(3)		✓	67.01	70.87	71.09	67	64.83	69.92	69.94	55
(4)	✓	✓	69.12	73.14	72.99	71	66.57	72.01	72.99	60

where False Negatives is the number of positive samples that the model incorrectly identified as negative. Recall is an important metric when it is crucial to capture as many positives as possible.

F1 Score. The F1 Score is the harmonic mean of precision and recall, used to measure the balance between precision and recall. Its calculation formula is:

$$F1Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{9}$$

This score conveys the balance between precision and recall in a single number, with the best value at 1 (perfect precision and recall) and the worst at 0. It is particularly useful when you need to compare two or more models or when the class distribution is imbalanced.

Mean Average Precision(mAP). Mean Average Precision is the mean of the Average Precision (AP) scores calculated across multiple classes. It is a crucial metric for evaluating the overall performance of a model in multi-class detection tasks. The calculation is as follows:

$$mAP = \frac{1}{N} \sum_{i=1}^{N} AP_i \tag{10}$$

where N is the number of classes, and AP_i is the Average Precision for the i^{th} class. Each AP_i is calculated by integrating the area under the precision-recall curve for that class across different thresholds. The mAP score provides a single performance measure that incorporates both precision and recall, and it is especially important in scenarios where each class has equal importance.

mAP@[IoU]. mAP@[IoU] is the Mean Average Precision calculated at different IoU thresholds. Intersection over Union (IoU) is a metric used to evaluate the overlap between predicted and true bounding boxes. The calculation formula for IoU is as follows:

$$mAP@[IoU] = \frac{1}{N} \sum_{i}^{N} AP@[IoU]_i \tag{11}$$

where $mAP@[IoU]_i$ is used to calculate the Average Precision (AP) for class i at a specific IoU threshold. In object detection tasks, it is common to set an IoU threshold (e.g., 0.5), and a prediction is considered correct only if the Intersection over Union (IoU) between the predicted bounding box and the true bounding box exceeds this threshold. mAP@[IoU] is calculated under this setting, and it is frequently used to evaluate the performance of a model under stricter matching criteria. This metric effectively measures how well the model is able to not just detect the objects but also accurately localize them by closely matching the predicted bounding boxes with the ground truth. The IoU threshold ensures that only predictions that meet a minimum standard of accuracy are considered, which helps to emphasize the quality of the detections. This is particularly important in applications where precision in object localization is critical, such as in

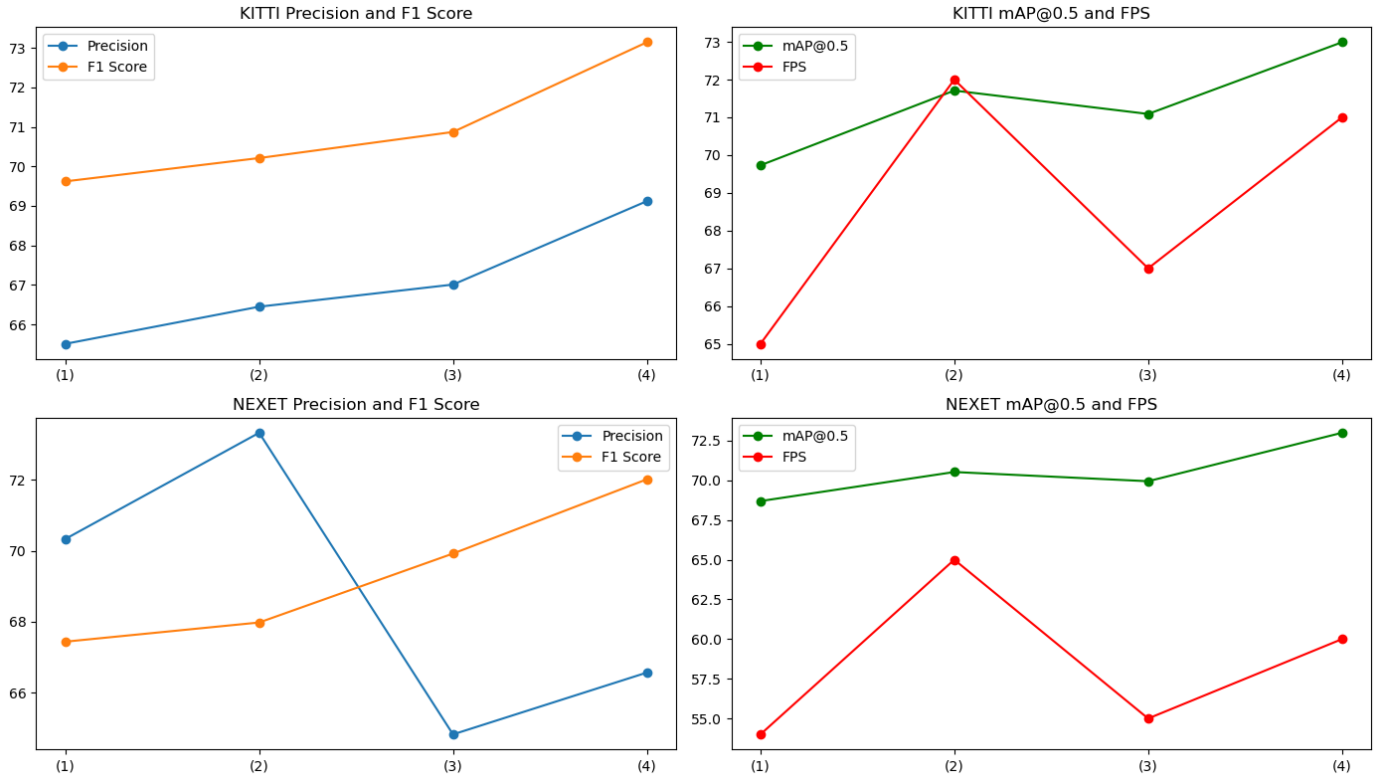

Figure 5. Visualization of results.Visualization of ablation experiment results. Among them (1) means not to add any module; (2) means to add only the EDR module; (3) means to add only the TSDN module; (4) means to add two modules.

autonomous driving, where misjudging the location of an object could lead to incorrect decision-making.

4.3 Experimental Results and Analysis

Table 3 provides a comprehensive analysis of the results of different methods applied to two datasets. This table includes key metrics such as Precision, F1 Score, mAP@0.5, and frames per second (FPS), aimed at holistically evaluating the performance of various methods in the field of object detection. On the KITTI dataset, our method achieved the best results, reaching scores of 76.72% in Precision, 74.62% in F1 Score, 75.32% in mAP@0.5, and 85 FPS. These results not only demonstrate its exceptional performance but also highlight its significant advantage in processing speed. For the NEXET dataset, our method also displayed outstanding performance, specifically achieving 74.52% in Precision, 71.82% in F1 Score, 73.12% in mAP@0.5, and 72 FPS. These results further substantiate the consistency of our method's efficiency and accuracy across different scenarios. We present qualitative results in Figures 6, demonstrating that our method yields favorable outcomes in real-world scenarios. In summary, by adopting our method, we have made significant breakthroughs in both speed and accuracy and have also validated the model's feasibility in practical production environments.

4.4 Ablation Study

As shown in Table 4, the results of the ablation studies compare the specific impacts of different module combinations on model performance. Through detailed analysis, the experimental results reveal their individual and combined influences on Precision (Pre), F1 Score, mAP@0.5, and frames per second (FPS). The baseline experiment did not utilize any of the new modules, while subsequent experiments progressively introduced modules to explore the stepwise improvement in model performance. Ultimately, when all modules were integrated, the experimental results achieved the best performance. This series of ablation studies not only verifies the importance of each module in enhancing model performance but also provides strong evidence for designing efficient object detection models, emphasizing the effectiveness of our model in the field of autonomous driving.

5 Discussion

In this study, we have successfully developed and validated a novel framework for rapid 3D point

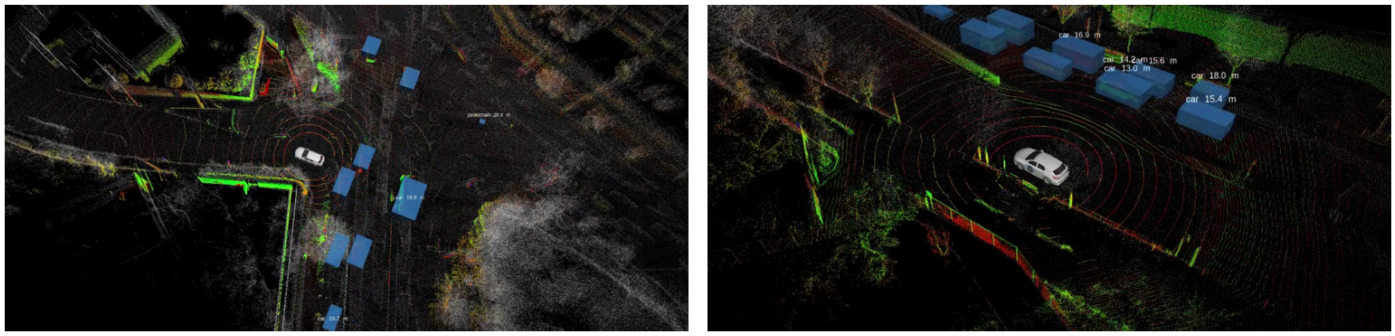

Figure 6. The pedestrian detection results.

cloud object detection. Through extensive testing on the KITTI and NEXET datasets, our method has demonstrated significant advantages in terms of real-time processing speed and accuracy. These achievements not only advance the technology of 3D point cloud processing but also provide robust technical support for real-time applications such as autonomous driving and robotic navigation. Our proposed hybrid data representation method effectively balances processing speed and detection accuracy by combining the benefits of voxelization and direct point feature extraction. This strategy addresses the issue of information loss inherent in traditional voxelization methods while overcoming the high computational costs associated with directly processing point cloud data. Ablation studies have further proven the important contribution of voxelization and point feature extraction to enhancing system performance, offering valuable insights for future research.

Despite the excellent performance of our method in current experiments, there are still some limitations and directions for future improvement. First, while the hybrid data representation method has optimized information retention and computational efficiency, reducing computation time further remains a challenge. Moreover, while our method performs well in vehicle and pedestrian detection tasks, its effectiveness in more complex or dense scenes requires further validation. Future work could explore more efficient voxelization techniques and point feature extraction algorithms to further enhance the system's real-time capabilities and robustness.

6 Conclusion

In this paper, we propose a novel framework for fast 3D point cloud object detection, tailored for complex applications that require real-time responses, such as autonomous driving and robotic navigation. Our method is primarily based on an innovative hybrid data representation technique that cleverly combines the computational efficiency of voxelization with the high precision of direct point processing, effectively resolving the traditional trade-off between speed and accuracy. Our developed hybrid data representation method significantly enhances data processing speed by utilizing both voxelization and point feature extraction, while preserving critical spatial and structural information, thereby increasing the accuracy of object detection. We have designed a two-stage architecture that includes a rapid region proposal network and a refinement detection network. This design not only improves detection efficiency but also optimizes the recognition and localization of multiple targets in complex scenes. Extensive experimental results indicate that our method significantly outperforms current leading technologies on the KITTI and NEXET datasets in terms of object detection accuracy and processing speed. Especially in terms of processing speed, our method meets the stringent requirements of real-time applications. In the future, we plan to further explore more efficient voxelization techniques and advanced point feature extraction methods to continuously optimize system performance. Moreover, we hope to expand our framework to accommodate more types of 3D sensory data and more complex application scenarios.

Conflicts of Interest

The author declare no conflicts of interest.

Funding

This work was supported without any funding.

References

[1] Zhu, Y., & Yan, W. Q. (2022). Traffic sign recognition based on deep learning. *Multimedia Tools and Applications*, 81(13), 17779-17791.

[2] Yu, Z., Li, L., Xie, J., Wang, C., Li, W., & Ning, X. (2024). Pedestrian 3d shape understanding for person re-identification via multi-view learning. *IEEE Transactions on Circuits and Systems for Video Technology*. [CrossRef]

[3] Wang, C., Wang, Y., Han, Y., Song, L., Quan, Z., Li, J., & Li, X. (2017, January). CNN-based object detection solutions for embedded heterogeneous multicore SoCs. In *2017 22nd Asia and South Pacific design automation conference (ASP-DAC)* (pp. 105-110). IEEE. [CrossRef]

[4] Ning, E., Wang, Y., Wang, C., Zhang, H., & Ning, X. (2024). Enhancement, integration, expansion: Activating representation of detailed features for occluded person re-identification. *Neural Networks*, 169, 532-541. [CrossRef]

[5] Yu, Z., Tiwari, P., Hou, L., Li, L., Li, W., Jiang, L., & Ning, X. (2024). Mv-reid: 3d multi-view transformation network for occluded person re-identification. *Knowledge-Based Systems*, 283, 111200. [CrossRef]

[6] Ning, X., He, F., Dong, X., Li, W., Alenezi, F., & Tiwari, P. (2024). ICGNet: An intensity-controllable generation network based on covering learning for face attribute synthesis. *Information Sciences*, 660, 120130. [CrossRef]

[7] Anthes, C., García-Hernández, R. J., Wiedemann, M., & Kranzlmüller, D. (2016, March). State of the art of virtual reality technology. In 2016 IEEE aerospace conference (pp. 1-19). IEEE. [CrossRef]

[8] Wang, G., Yu, L., Tian, S., Zhang, H., Xue, Y., Sang, M., ... & Si, S. (2024). Pctn: Point cloud data transformation network. Displays, 81, 102610. [CrossRef]

[9] Chen, X., Ma, H., Wan, J., Li, B., & Xia, T. (2017). Multi-view 3d object detection network for autonomous driving. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition* (pp. 1907-1915).

[10] Zhou, Y., & Tuzel, O. (2018). Voxelnet: End-to-end learning for point cloud based 3d object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 4490-4499).

[11] Qi, C. R., Su, H., Mo, K., & Guibas, L. J. (2017). Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 652-660).

[12] Qi, C. R., Yi, L., Su, H., & Guibas, L. J. (2017). Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30.

[13] Redmon, J. (2016). You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*.

[14] Duan, K., Bai, S., Xie, L., Qi, H., Huang, Q., & Tian, Q. (2019). Centernet: Keypoint triplets for object detection. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 6569-6578).

[15] Bolya, D., Zhou, C., Xiao, F., & Lee, Y. J. (2019). Yolact: Real-time instance segmentation. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 9157-9166).

[16] Zhang, S., Wen, L., Bian, X., Lei, Z., & Li, S. Z. (2018). Single-shot refinement neural network for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 4203-4212).

[17] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. Y., & Berg, A. C. (2016). Ssd: Single shot multibox detector. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14* (pp. 21-37). Springer International Publishing.

[18] Tan, M., Pang, R., & Le, Q. V. (2020). Efficientdet: Scalable and efficient object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 10781-10790).

[19] Lin, T. Y., Dollár, P., Girshick, R., He, K., Hariharan, B., & Belongie, S. (2017). Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2117-2125).

[20] Wang, K., Liew, J. H., Zou, Y., Zhou, D., & Feng, J. (2019). Panet: Few-shot image semantic segmentation with prototype alignment. In *proceedings of the IEEE/CVF international conference on computer vision* (pp. 9197-9206).

[21] Howard, A. G. (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*.

[22] Wu, B., Iandola, F., Jin, P. H., & Keutzer, K. (2017). Squeezedet: Unified, small, low power fully convolutional neural networks for real-time object detection for autonomous driving. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops* (pp. 129-137).

[23] Wang, R. J., Li, X., & Ling, C. X. (2018). Pelee: A real-time object detection system on mobile devices. *Advances in neural information processing systems*, 31.

[24] Qin, Z., Li, Z., Zhang, Z., Bao, Y., Yu, G., Peng, Y., & Sun, J. (2019). ThunderNet: Towards real-time generic object detection on mobile devices. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 6718-6727).

[25] Geiger, A., Lenz, P., & Urtasun, R. (2012, June). Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE conference on computer vision and pattern recognition* (pp. 3354-3361). IEEE.

[26] Unal, D., Catak, F. O., Houkan, M. T., Mudassir, M., & Hammoudeh, M. (2023). Towards robust autonomous driving systems through adversarial test set generation. *ISA transactions*, 132, 69-79. [CrossRef]

[27] Kingma, D. P. (2014). Adam: A method for stochastic

optimization. *arXiv preprint arXiv:1412.6980*.

[28] Ioffe, S. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*.

[29] Othmani, M. (2022). A vehicle detection and tracking method for traffic video based on faster R-CNN. *Multimedia Tools and Applications*, 81(20), 28347-28365.

[30] Mohandas, P. (2023). Sad: Sensor-based anomaly detection system for smart junctions. *IEEE Sensors Journal*, 23(17), 20368-20378. [CrossRef]

[31] Chen, X. (2022, October). Traffic lights detection method based on the improved yolov5 network. In *2022 IEEE 4th International Conference on Civil Aviation Safety and Information Technology (ICCASIT)* (pp. 1111-1114). IEEE. [CrossRef]

[32] Li, S., Wang, S., & Wang, P. (2023). A small object detection algorithm for traffic signs based on improved YOLOv7. *Sensors*, 23(16), 7145. [CrossRef]

[33] Soylu, E., & Soylu, T. (2024). A performance comparison of YOLOv8 models for traffic sign detection in the Robotaxi-full scale autonomous vehicle competition. *Multimedia Tools and Applications*, 83(8), 25005-25035.