Scientific Research Publishing

Open Journal of Applied Sciences, 2025, 15(3), 627-637
https://www.scirp.org/journal/ojapps
ISSN Online: 2165-3925
ISSN Print: 2165-3917

A Novel Pillar Feature Encoder for Pillar-Based 3D Object Detection in Point Clouds

Huijun Yu, Shuai Kang*, Zhihao Zou

College of Railway Transportation, Hunan University of Technology, Zhuzhou, China
Email: *fq1948316719@163.com

How to cite this paper: Yu, H.J., Kang, S. and Zou, Z.H. (2025) A Novel Pillar Feature Encoder for Pillar-Based 3D Object Detection in Point Clouds. *Open Journal of Applied Sciences*, **15**, 627-637.
https://doi.org/10.4236/ojapps.2025.153041

Received: February 26, 2025
Accepted: March 15, 2025
Published: March 18, 2025

Abstract

Point cloud-based 3D object detection is a key technology in autonomous driving and mobile robot perception systems. However, the sparsity and irregularity of point cloud data result in poor performance of existing methods in detecting small objects at long distances and occluded objects. This paper proposes a Novel Pillar Feature Encoder to address feature encoding challenges in Pillar-based 3D point cloud object detection, improving the detection of occluded and small objects, especially at long distances. This method converts point cloud data into pillar features through voxelization and designs two convolutional neural network branches: Point Feature Encoding and Pillar Feature Encoding. The former extracts point features within local pillars, while the latter extracts global pillar features, which are then fused to resolve the problem of occlusion-related information loss, thus enhancing the detection accuracy of occluded objects. A Multi-attention mechanism is introduced to enhance the focus on key point features and learn optimal channel weights, thus improving the detection of small objects at long distances. We conducted experiments on the PointPillars network framework using the KITTI dataset for training and testing. The results show that the improved algorithm significantly enhances the average precision (AP) for 3D detection of Cars, Pedestrians, and Cyclists on the KITTI dataset, demonstrating exceptional performance in detecting occluded objects and small targets at long distances, thus validating the effectiveness of the proposed method.

Keywords

Point Cloud, 3D Object Detection, Pillar Feature Encoder, Multi-Attention

1. Introduction

3D Object Detection in point clouds plays a crucial role in environmental percep-

tion for autonomous driving systems [1] and robotics [2]. As the primary data form collected by LiDAR, point cloud data provides precise three-dimensional geometric information, demonstrating unique advantages in 3D object detection tasks. However, the sparsity, disorderliness, and complexity of point cloud data, as well as challenging scenarios such as distant small objects and occluded targets, pose significant challenges to 3D object detection. The performance of existing methods in these scenarios still faces notable bottlenecks.

In recent years, significant progress has been made in point cloud-based 3D object detection algorithms. Based on the processing methods of point cloud data, existing approaches can be primarily categorized into three types: Point-based methods, Voxel-based methods, and Pillar-based methods. Point-based methods directly process raw point clouds, maximally preserving the geometric information of the point cloud. Pioneering works such as PointNet [3] and PointNet++ [4] successfully addressed the disorderliness of point cloud data by designing point-based convolutional networks and effectively extracting local geometric features. However, these methods exhibit limitations in handling global information and perform poorly in scenarios involving sparse point clouds, occlusions, and distant small object detection, leading to reduced detection accuracy. Subsequent studies such as PointCNN [5] and DGCNN [6] further improved the feature extraction mechanisms but still face challenges in balancing computational efficiency and detection accuracy.

Voxel-based methods rasterize point clouds into regular voxels and utilize 3D convolutional neural networks for feature extraction and 3D object detection. VoxelNet [7] first proposed the Voxel Feature Extractor (VFE) module, laying the foundation for voxel-based processing of point cloud data. However, the high computational complexity of 3D convolution limits its application in sparse point cloud scenarios. SECOND [8] significantly reduced computational overhead by performing calculations only on non-empty voxels, but issues such as information loss and low computational efficiency remain.

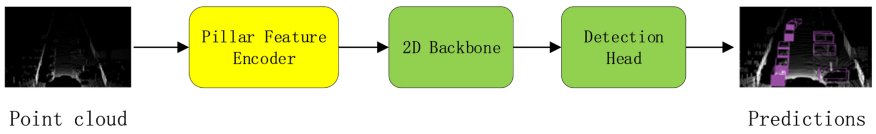

Figure 1. Structure of PointPillars.

To achieve a better balance between detection accuracy and computational efficiency, Pillar-based methods have emerged. PointPillars [9] achieves this by rasterizing point cloud data into Pillar features and utilizing a 2D convolutional network for feature extraction. This approach not only significantly reduces computational complexity but also effectively retains key information from the point cloud data. The network structure, shown in **Figure 1**, consists of three main components: the Pillar Feature Encoder, the 2D Backbone, and the Detection Head. However, the information loss during the feature encoding process in the Pillar

Feature Encoder still limits its performance in detecting occluded objects and small targets [10]-[12].

To address this issue, PillarNet [13] proposed a novel point cloud grid encoder structure, which improves 3D object detection performance by decoupling point cloud feature extraction, backbone, and neck modules. Meanwhile, FastPillars [14] introduced a lightweight Max-and-Attention Pillar encoding module, focusing on enhancing the detection capability for small targets, but it still lacks sufficient global feature fusion.

Although the aforementioned methods have made significant progress in the field of point cloud-based 3D object detection, the detection performance in complex scenarios such as distant small objects and occluded targets still needs improvement. This paper addresses the issue of information loss in the feature encoding process of existing Pillar-based methods by proposing an improved Pillar Feature Encoder (Novel Pillar Feature Encoder). This network effectively enhances the detection accuracy of occluded objects and small targets by introducing global feature fusion and attention mechanisms. The main contributions of this paper are as follows:

1) Novel Pillar Feature Encoder (NPFE): This paper designs a Novel Pillar Feature Encoder network, which structures unordered point clouds into Pillars and fully leverages the correlations among points within each Pillar. By efficiently extracting and fusing both local and global features within Pillars, the proposed method significantly enriches the feature representation capability of point cloud data. This approach effectively enhances the detection performance of Pillar-based 3D object detection algorithms for small and subtle objects.

2) Multi-Attention Mechanism for Pillar Feature Extraction: A multi-attention mechanism is introduced to optimize pillar feature extraction. By focusing on regions in the feature map that contribute significantly to key pillar information while suppressing irrelevant information, the mechanism enhances the network's ability to identify critical features. This significantly improves the robustness of 3D detection for occluded objects in complex scenes.

3) NPFE-improved PointPillars algorithm and Experimental Validation: Based on the NPFE method, this paper improves the PointPillars algorithm and validates its effectiveness through training and testing on the KITTI [15] dataset. Experimental results demonstrate that the proposed method achieves significant improvements in 3D object detection accuracy, particularly for occluded objects and distant small targets.

2. The Design of Novel Pillar Feature Encoder

The structure of the Novel Pillar Feature Encoder (NPFE) proposed in this paper is shown in **Figure 2**. For an input frame of point cloud data, the feature of any given point i can be represented as $p_i = [x_i, y_i, z_i, r_i]$, where $p_i \in \mathrm{R}^{1\times 4}$, $[x_i, y_i, z_i]$ represents the point's coordinates in 3D space, and r_i represents the point's reflectance intensity. In the point cloud feature encoding module, the

point cloud data is first divided into equally sized pillars. The Z-coordinate height information of each point is ignored, and the point cloud is voxelized in the X-Y plane using a fixed-size grid, assigning each point to the corresponding pillar. The number of points in each pillar is set to N, which is set to 64 in this paper. When the number of points in a pillar exceeds the set value, points with Z-coordinates close to zero (ground points) are first removed, and the remaining points are randomly downsampled to ensure that the number of points in each pillar is maintained at 64, thus suppressing potential ground interference during the sampling process. After voxelization, the point cloud pillar features $\boldsymbol{S}=\{\boldsymbol{P}_1,\boldsymbol{P}_2,\cdots,\boldsymbol{P}_k\}$ are obtained, where $\boldsymbol{P}_k=\{p_1,p_2,\cdots,p_N\}$ represents the k-th pillar feature after voxelization, and C denotes the feature dimension of each point in a pillar, *i.e.*, the feature channel size is 4. These pillar features $\boldsymbol{P}_k$ are then fed into two branches: the Point Feature Encoding branch and the Global Pillar Feature Encoding branch, which respectively extract the point features within each pillar and the global association features between each pillar and the entire set of pillars, followed by fusion to obtain the encoded pillar features for the entire point cloud. Next, this paper will introduce the design of the Point Feature Encoding branch and the Global Pillar Feature Encoding branch in detail.

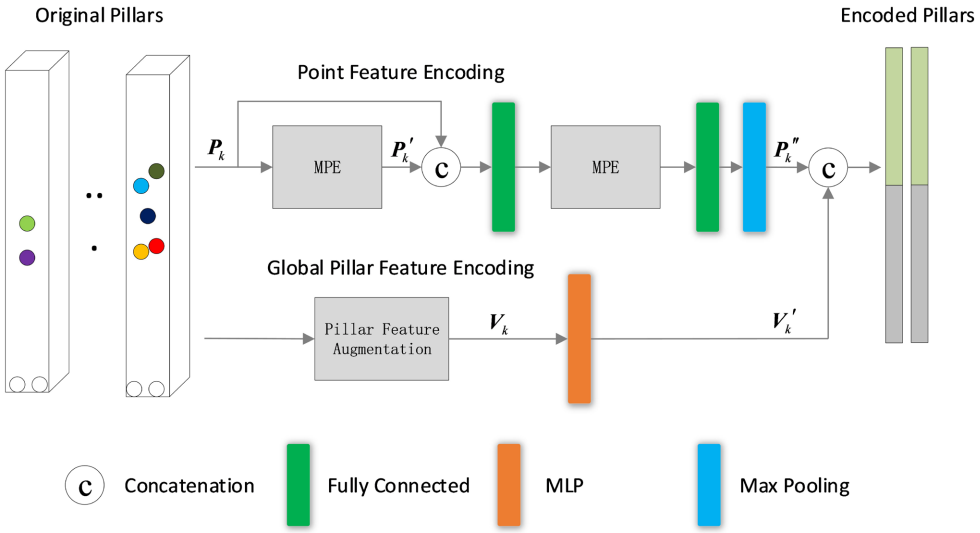

Figure 2. The structure of the novel pillar feature encoder.

2.1. Point Feature Encoding Branch

As shown in **Figure 3**, to obtain the point-wise and channel-wise features of the point cloud, this paper refers to the hybrid attention network of CBAM [16] and designs a Multi-attention Point Encoding (MPE) module in the Point Feature Encoding branch as the basic unit for point feature extraction. In the Point Feature Encoding branch, to enhance the network's generalization ability, the MPE module is stacked twice sequentially. First, the original pillar features $\boldsymbol{P}_k$ are fed into the first MPE module for processing, resulting in the primary feature $\boldsymbol{P}_k'$. Then, $\boldsymbol{P}_k'$ is concatenated with the original feature $\boldsymbol{P}_k$ and passed through a fully con-

nected layer to map it into a higher-dimensional space, which is then forwarded to the next MPE module. The final output is further processed by a fully connected layer and a max-pooling layer to aggregate the features of all points within the pillars, thus obtaining the final output feature P_k''.

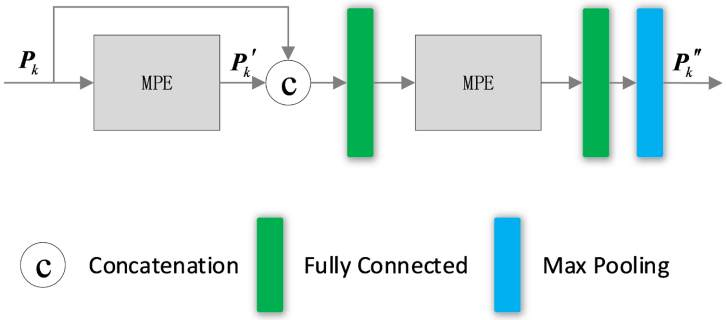

Figure 3. Structure of point feature encoding.

The structure of the MPE module is shown in Figure 4. First, channel attention [17] and point attention are applied to calculate the attention weights for each point within the pillars. Then, these two attention weights are fused, and the fused weights are applied to the point cloud's pillar feature matrix P_k, allowing the encoding network to adaptively extract pillar features based on the importance of both points and channels.

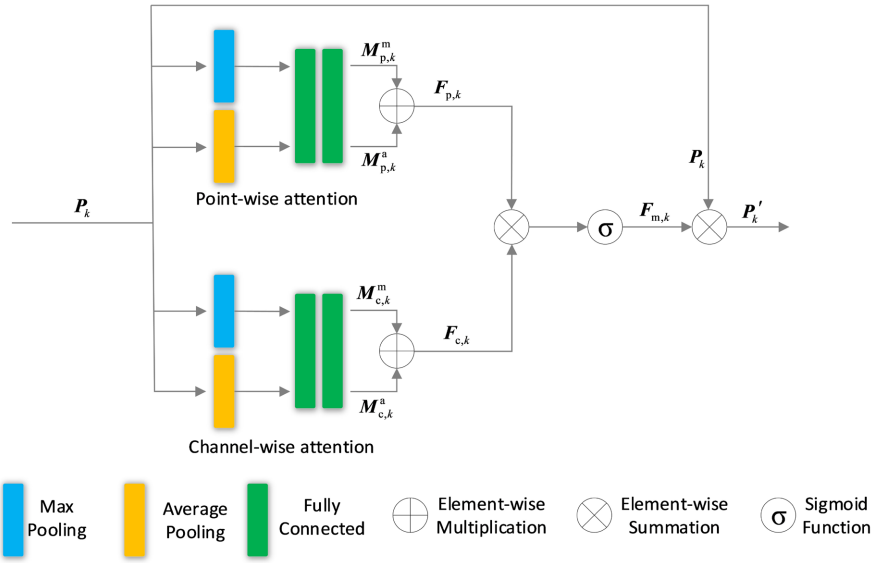

Figure 4. The structure of the MPE.

In the Point-wise attention part of the MPE, the input pillar feature matrix P_k is processed to fully capture the feature relationships between each channel. Max pooling and average pooling operations are applied to the input feature map for point-wise feature aggregation. Then, the two resulting weight matrices are fed into a shared network that includes fully connected layers and a nonlinear activa-

tion function. Finally, the point attention weight $\boldsymbol{F}_{\mathrm{p},k}$ is obtained by element-wise addition of the weight matrices $\boldsymbol{M}_{\mathrm{p},k}^{\mathrm{m}}$ and $\boldsymbol{M}_{\mathrm{p},k}^{\mathrm{a}}$ from the max pooling and average pooling operations, respectively. This step primarily describes the spatial correlation between points within the pillars. The corresponding computational formulas for this process are provided in Equation (1), Equation (2) and Equation (3) below:

$$\boldsymbol{M}_{\mathrm{p},k}^{\mathrm{m}} = \boldsymbol{W}_2\left(\sigma \boldsymbol{W}_1\left(\max\_\mathrm{c}\left(\boldsymbol{P}_k\right)\right)\right) \tag{1}$$

$$\boldsymbol{M}_{\mathrm{p},k}^{\mathrm{a}} = \boldsymbol{W}_2\left(\sigma \boldsymbol{W}_1\left(\mathrm{avg\_c}\left(\boldsymbol{P}_k\right)\right)\right) \tag{2}$$

$$\boldsymbol{F}_{\mathrm{p},k} = \boldsymbol{M}_{\mathrm{p},k}^{\mathrm{m}} + \boldsymbol{M}_{\mathrm{p},k}^{\mathrm{a}} \tag{3}$$

Here, $\max\_\mathrm{c}(\bullet)$ represents the max pooling operation along the channel dimension of the feature map, $\mathrm{avg\_c}(\bullet)$ represents the average pooling operation along the channel dimension of the feature map, $\sigma(\bullet)$ represents the ReLU activation function, and $\boldsymbol{W}_1$ and $\boldsymbol{W}_2$ represent the weight matrices of the two fully connected layers in the point attention module shown above.

In the Channel-wise attention part, it is similar to the Point-wise attention part. To more comprehensively aggregate the channel features across the point dimension, max pooling and average pooling operations are applied to aggregate the point features of the input feature matrix $\boldsymbol{P}_k$, as shown in the lower part of the channel-wise attention section in Figure 4. Then, the two weight matrices are fed into a shared network that includes fully connected layers and a nonlinear activation function. Finally, the channel attention weight $\boldsymbol{F}_{\mathrm{c},k}$ is obtained by element-wise addition of the weight matrices $\boldsymbol{M}_{\mathrm{c},k}^{\mathrm{m}}$ and $\boldsymbol{M}_{\mathrm{c},k}^{\mathrm{a}}$ from the max pooling and average pooling operations, respectively. This step is primarily used to describe the spatial correlation between points within the pillars. The corresponding computational formulas for this process are provided in Equation (4), Equation (5) and Equation (6) below:

$$\boldsymbol{M}_{\mathrm{c},k}^{\mathrm{m}} = \boldsymbol{W}_2'\left(\sigma \boldsymbol{W}_1'\left(\max\_\mathrm{p}\left(\boldsymbol{P}_k\right)\right)\right) \tag{4}$$

$$\boldsymbol{M}_{\mathrm{c},k}^{\mathrm{a}} = \boldsymbol{W}_2'\left(\sigma \boldsymbol{W}_1'\left(\mathrm{avg\_p}\left(\boldsymbol{P}_k\right)\right)\right) \tag{5}$$

$$\boldsymbol{F}_{\mathrm{c},k} = \boldsymbol{M}_{\mathrm{c},k}^{\mathrm{m}} + \boldsymbol{M}_{\mathrm{c},k}^{\mathrm{a}} \tag{6}$$

Here, $\max\_\mathrm{p}(\bullet)$ represents the max pooling operation along the point dimension of the feature map, $\mathrm{avg\_c}(\bullet)$ represents the average pooling operation along the point dimension of the feature map, and $\boldsymbol{W}_1'$ and $\boldsymbol{W}_2'$ represent the weight matrices of the two fully connected layers in the channel attention module shown above.

Finally, the point attention weight $\boldsymbol{F}_{\mathrm{p},k}$ and the channel attention weight $\boldsymbol{F}_{\mathrm{c},k}$ are element-wise multiplied, and the result is passed through the ReLU activation function to obtain the mixed attention weight $\boldsymbol{F}_{\mathrm{m},k}$. The mixed attention weight is then applied to the original pillar features $\boldsymbol{P}_k$ by performing an element-wise multiplication, resulting in the final output feature $\boldsymbol{P}_k'$. The corre-

sponding computational formulas for this process are provided in Equation (7) and Equation (8) below:

$$\boldsymbol{F}_{\mathrm{m},k} = \sigma\left(\boldsymbol{F}_{\mathrm{c},k} \times \boldsymbol{F}_{\mathrm{p},k}\right) \tag{7}$$

$$\boldsymbol{P}_k' = \boldsymbol{F}_{\mathrm{m},k} \odot \boldsymbol{P}_k \tag{8}$$

2.2. Global Pillar Feature Encoding Branch

As shown in the Global Pillar Feature Encoding branch in Figure 2, to encode the relationship features between each pillar and the entire point cloud, this paper designs an extension for the pillar features (Pillar Feature Augmentation) in the Global Pillar Feature Encoding branch, constructing pillar features $\boldsymbol{V}_k = \left[x_k^{\mathrm{c}}, y_k^{\mathrm{c}}, z_k^{\mathrm{c}}, x_k^{\mathrm{p}}, y_k^{\mathrm{p}}, z_k^{\mathrm{p}}, \Delta x_k^{\mathrm{c}}, \Delta y_k^{\mathrm{c}}, \Delta z_k^{\mathrm{c}}, \Delta x_k^{\mathrm{p}}, \Delta y_k^{\mathrm{p}}, \Delta z_k^{\mathrm{p}}\right]$, $\boldsymbol{V}_k \in \mathrm{R}^{1\times 12}$ that include global association information. Here, $\left[x_k^{\mathrm{c}}, y_k^{\mathrm{c}}, z_k^{\mathrm{c}}\right]$ represents the centroid coordinates of all points in the *k*-th pillar (the arithmetic average of all point coordinates), $\left[x_k^{\mathrm{p}}, y_k^{\mathrm{p}}, z_k^{\mathrm{p}}\right]$ represents the center coordinates of the *k*-th pillar, $\left[\Delta x_k^{\mathrm{c}}, \Delta y_k^{\mathrm{c}}, \Delta z_k^{\mathrm{c}}\right]$ represents the coordinate offset of the centroid of all points in the *k*-th pillar relative to the centroid of the entire point cloud, and $\left[\Delta x_k^{\mathrm{p}}, \Delta y_k^{\mathrm{p}}, \Delta z_k^{\mathrm{p}}\right]$ represents the coordinate offset of the center of the *k*-th pillar relative to the center of the entire point cloud.

In the encoding of the *k*-th pillar's features, the input pillar feature vector is first processed by a Multi-Layer Perceptron (MLP) for feature extraction and mapping. Since the MLP layer includes operations such as Batch Normalization and ReLU activation functions, these steps effectively enhance the expressive ability of the features, resulting in an expanded pillar feature vector $\boldsymbol{V}_k'$, as expressed in Equation (9):

$$\boldsymbol{V}_k' = f\left(\boldsymbol{V}_k; \boldsymbol{W}_3\right) \tag{9}$$

The $f(\bullet)$ represents the result of applying the MLP operation to $\boldsymbol{V}_k$, where $\boldsymbol{V}_k$ represents the weight parameters of the MLP.

3. Experiments and Comparisons

PointPillars is the first state-of-the-art (SOTA) model to propose a Pillar-based approach. To validate the effectiveness of the proposed feature encoding network, this paper conducts experimental improvements based on the PointPillars algorithm. Specifically, the Pillar Feature Encoder network of PointPillars is replaced with the proposed NPFE, while retaining the backbone and detection head of the PointPillars algorithm. Finally, training and testing experiments are conducted on the KITTI dataset, and the effect of NPFE in 3D object detection tasks is analyzed.

3.1. Experimental Dataset

This paper utilizes the open-source KITTI dataset for algorithm validation, which includes diverse traffic scenarios, such as urban, suburban, and highway environments, and provides rich image and LiDAR data. The dataset consists of 7481

training samples and 7518 testing samples. Following the methodology outlined in the PointPillars paper, the training samples of KITTI are further split into 3712 samples for the training set and 3769 samples for the validation set, which are used during the network training process. The KITTI dataset includes three types of objects: cars, pedestrians, and cyclists. Based on the height, occlusion, and truncation levels of the object bounding boxes in the dataset, the object detection task is classified into three difficulty levels: Easy, Moderate, and Hard. The performance of the proposed algorithm is evaluated at these three difficulty levels.

3.2. Experimental Environment

The experiments are conducted using the PyTorch deep learning framework, with the operating system being Ubuntu 18.04. The computer processor is an Intel Core i5-11600 CPU, with 64GB of memory. An NVIDIA GeForce RTX 3090 GPU is used to accelerate the training process. The batch size during training is set to 1, the learning rate is set to 0.00075, and the Adam optimizer is used for training. The maximum number of iterations is set to 180.

3.3. Comparative Experiments

This paper employs average precision (AP) to evaluate the experimental results of the NPFE-improved PointPillars algorithm (Ours), which outperforms both the original baseline network and other methods in terms of 3D detection AP for the Car, Pedestrian, and Cyclist categories on the KITTI dataset. The results are compared with those of the original PointPillars algorithm, as well as several other prominent point cloud 3D object detection algorithms, including VoxelNet, SECOND, and PointRCNN. The comparison results are presented in Table 1.

Table 1. Comparison of AP for our algorithm and other algorithms on the KITTI validation set.

Algorithm	Car (IoU = 0.7)			Pedestrian (IoU = 0.5)			Cyclist (IoU = 0.5)		
	Easy	Mode.	Hard	Easy	Mode.	Hard	Easy	Mode.	Hard
VoxelNet	81.97	65.46	62.85	57.86	53.42	48.87	67.17	47.65	45.11
SECOND	83.13	73.66	66.20	51.07	42.56	37.29	70.51	53.85	46.90
PointRCNN	85.94	75.76	68.32	49.43	41.78	38.63	73.84	59.86	53.49
PointPillars	83.02	73.98	68.20	51.48	42.05	38.90	76.78	56.85	52.04
Ours	86.26	77.05	74.09	55.71	49.84	44.61	77.82	60.21	53.42

In the same experimental setup, qualitative evaluations of the aforementioned models were conducted, with the results summarized in Table 1. The NPFE-improved PointPillars outperforms the baseline and other methods in 3D detection AP for cars, pedestrians, and cyclists due to its ability to better preserve global contextual information and focus on critical point features. Specifically, under the Moderate difficulty level, the proposed algorithm achieves a 3.07% increase in car detection AP, a 3.35% increase in cyclist detection AP, and a notable 7.79% im-

provement in the detection of smaller targets such as pedestrians. Furthermore, as indicated in Table 1, the NPFE-enhanced PointPillars also outperforms several other classic algorithms.

3.4. Visualization of Experimental Results

This section presents a qualitative evaluation of the NPFE-improved PointPillars algorithm (Ours) through visual analysis. For this experiment, two complex scenes from the KITTI dataset validation set were selected to conduct a comparative 3D object detection experiment, as shown in Figure 5. Specifically, Figure 5(a) depicts a scene with occlusion, while Figure 5(b) illustrates a scene containing numerous small targets and occlusions. In Figure 5, the RGB images corresponding to each scene are shown at the top, while the point cloud data processed for visualization is displayed at the bottom. The detection results of both algorithms are marked in the figure, with the NPFE-improved PointPillars algorithm on the left and the original PointPillars algorithm on the right. The purple boxes indicate cars, the red boxes represent cyclists, and the blue boxes highlight pedestrians. Additionally, the yellow ellipses emphasize misidentified targets found in the improved algorithm's results.

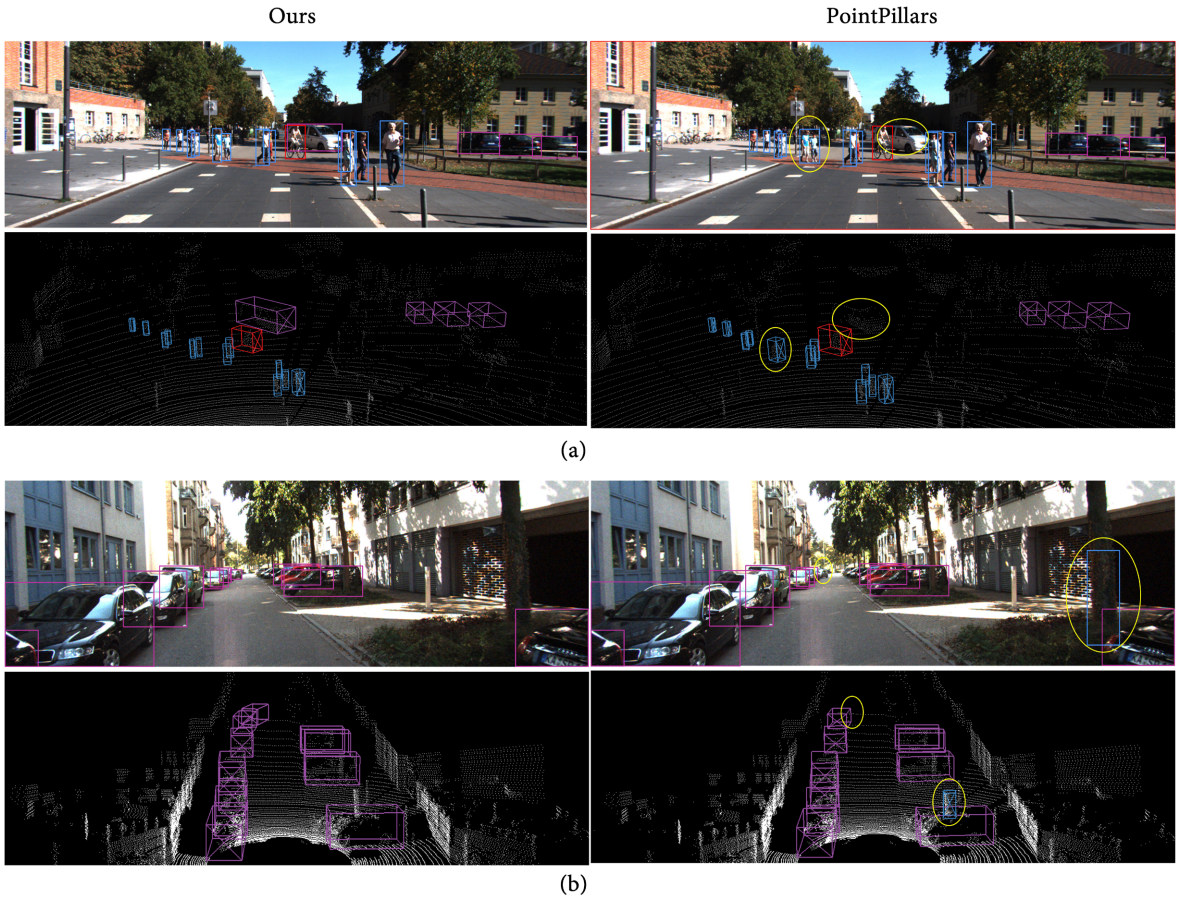

Figure 5. Point cloud and image results of 3D object detection performance of our algorithm and the PointPillars algorithm in different scenes. (a) Scene 1; (b) Scene 2.

As shown in Figure 5, the NPFE-improved PointPillars algorithm has superior detection performance in both occluded and small target scenarios. In scene 1, which features a diverse and densely distributed set of object types, the original PointPillars algorithm fails to detect a vehicle occluded by a crowd (highlighted by the yellow elliptical frame), whereas the improved algorithm successfully identifies the target, showcasing its advantage in handling occlusion. Similarly, in scene 2, which contains sparse and small distant targets, the improved algorithm outperforms the original in detecting densely packed vehicles and avoids the issue of misidentifying small roadside trees as pedestrians, thereby significantly enhancing detection accuracy and reliability, particularly under occlusion conditions and for small target 3D detection.

4. Conclusion

This paper presents a novel Pillar Feature Encoder (NPFE) aimed at enhancing the performance of Pillar-based 3D point cloud object detection algorithms. By introducing a multi-attention mechanism, NPFE can more effectively extract key features from point clouds, combining global and local information to further improve detection performance for small and distant objects. Experimental results demonstrate that NPFE significantly outperforms traditional PointPillars, particularly in scenarios involving occlusion and small object detection, on the KITTI dataset. Additionally, the visual results demonstrate that the improved algorithm has significantly improved detection performance in small target and occluded scenes, further confirming its significant advantages in enhancing the robustness and accuracy of Pillar-based point cloud 3D object detection algorithms. Future work will focus on extending NPFE to other point cloud-based detection frameworks and exploring its scalability in large-scale real-time systems. We also plan to validate the robustness and performance of this method in more complex dynamic environments, particularly for applications in autonomous driving and robotic perception systems.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Song, Z., Liu, L., Jia, F., Luo, Y., Jia, C., Zhang, G., *et al.* (2024) Robustness-Aware 3D Object Detection in Autonomous Driving: A Review and Outlook. *IEEE Transactions on Intelligent Transportation Systems*, **25**, 15407-15436. https://doi.org/10.1109/tits.2024.3439557

[2] Wang, X., Mizukami, Y., Tada, M. and Matsuno, F. (2020) Navigation of a Mobile Robot in a Dynamic Environment Using a Point Cloud Map. *Artificial Life and Robotics*, **26**, 10-20. https://doi.org/10.1007/s10015-020-00617-3

[3] Charles, R.Q., Su, H., Kaichun, M. and Guibas, L.J. (2017) Pointnet: Deep Learning on Point Sets for 3D Classification and Segmentation. 2017 *IEEE Conference on Computer Vision and Pattern Recognition* (*CVPR*), Honolulu, 21-26 July 2017, 77-

85. https://doi.org/10.1109/cvpr.2017.16

[4] Qi, C.R., Yi, L., Su, H. and Guibas, L.J. (2017) Pointnet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. *Advances in Neural Information Processing Systems*, Long Beach, 4-9 December 2017, 5099-5108.

[5] Li, Y., Bu, R., Sun, M., Wu, W., Di, X. and Chen, B. (2018) PointCNN: Convolution on X-Transformed Points. *Neural Information Processing Systems*, Long Beach, 16-20 June 2019, 770-779.

[6] Phan, A.V., Nguyen, M.L., Nguyen, Y.L.H. and Bui, L.T. (2018) DGCNN: A Convolutional Neural Network over Large-Scale Labeled Graphs. *Neural Networks*, **108**, 533-543. https://doi.org/10.1016/j.neunet.2018.09.001

[7] Zhou, Y. and Tuzel, O. (2018) Voxelnet: End-to-End Learning for Point Cloud Based 3D Object Detection. 2018 *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Salt Lake City, 18-22 June 2018, 4490-4499. https://doi.org/10.1109/cvpr.2018.00472

[8] Yan, Y., Mao, Y. and Li, B. (2018) SECOND: Sparsely Embedded Convolutional Detection. *Sensors*, **18**, Article No. 3337. https://doi.org/10.3390/s18103337

[9] Lang, A.H., Vora, S., Caesar, H., Zhou, L., Yang, J. and Beijbom, O. (2019) Pointpillars: Fast Encoders for Object Detection from Point Clouds. 2019 *IEEE/CVF Conference on Computer Vision and Pattern Recognition* (*CVPR*), Long Beach, 15-20 June 2019, 12697-12705. https://doi.org/10.1109/cvpr.2019.01298

[10] He, C., Zeng, H., Huang, J., Hua, X. and Zhang, L. (2020) Structure Aware Single-Stage 3D Object Detection from Point Cloud. 2020 *IEEE/CVF Conference on Computer Vision and Pattern Recognition* (*CVPR*), Seattle, 14-19 June 2020, 11873-11882. https://doi.org/10.1109/cvpr42600.2020.01189

[11] Shi, S., Guo, C., Jiang, L., Wang, Z., Shi, J., Wang, X., *et al.* (2020) PV-RCNN: Point-Voxel Feature Set Abstraction for 3D Object Detection. 2020 *IEEE/CVF Conference on Computer Vision and Pattern Recognition* (*CVPR*), Seattle, 14-19 June 2020, 10529-10538. https://doi.org/10.1109/cvpr42600.2020.01054

[12] Yang, Z., Sun, Y., Liu, S. and Jia, J. (2020) 3DSSD: Point-Based 3D Single Stage Object Detector. 2020 *IEEE/CVF Conference on Computer Vision and Pattern Recognition* (*CVPR*), Seattle, 14-19 June 2020, 11040-11048. https://doi.org/10.1109/cvpr42600.2020.01105

[13] Shi, G., Li, R. and Ma, C. (2022) Pillarnet: Real-Time and High-Performance Pillar-Based 3D Object Detection. *Computer Vision—ECCV* 2022 17*th European Conference*, Tel Aviv, 23-27 October 2022, 35-52. https://doi.org/10.1007/978-3-031-20080-9_3

[14] Zhou, S., Tian, Z., Chu, X., Zhang, X., Zhang, B., Lu, X., *et al.* (2023) FastPillars: A Deployment-Friendly Pillar-Based 3D Detector. https://doi.org/10.48550/arXiv.2302.02367

[15] Geiger, A., Lenz, P., Stiller, C. and Urtasun, R. (2013) Vision Meets Robotics: The KITTI Dataset. *The International Journal of Robotics Research*, **32**, 1231-1237. https://doi.org/10.1177/0278364913491297

[16] Woo, S., Park, J., Lee, J. and Kweon, I.S. (2018) CBAM: Convolutional Block Attention Module. *Computer Vision—ECCV* 2018 15*th European Conference*, Munich, 8-14 September 2018, 3-19. https://doi.org/10.1007/978-3-030-01234-2_1

[17] Wang, Q., Wu, B., Zhu, P., Li, P., Zuo, W. and Hu, Q. (2020) Eca-Net: Efficient Channel Attention for Deep Convolutional Neural Networks. 2020 *IEEE/CVF Conference on Computer Vision and Pattern Recognition* (*CVPR*), Seattle, 14-19 June 2020, 11534-11542. https://doi.org/10.1109/cvpr42600.2020.01155