Hindawi
Scientific Programming
Volume 2021, Article ID 4650660, 11 pages
https://doi.org/10.1155/2021/4650660

Research Article

Enhancing Point Features with Spatial Information for Point-Based 3D Object Detection

Huaijin Liu,[1] **Jixiang Du**,[2,3] **Yong Zhang**,[1] **and Hongbo Zhang**[2,3]

[1]*College of Mechanical Engineering and Automation, Huaqiao University, Xiamen 361021, China*
[2]*College of Computer Science and Technology, Huaqiao University, Xiamen 361021, China*
[3]*Fujian Key Laboratory of Big Data Intelligence and Security, Huaqiao University, Xiamen 361021, China*

Correspondence should be addressed to Huaijin Liu; lhjhqdx@163.com and Jixiang Du; jxdu@hqu.edu.cn

Received 7 September 2021; Revised 27 October 2021; Accepted 3 December 2021; Published 21 December 2021

Academic Editor: Jianping Gou

Currently, there are many kinds of voxel-based multisensor 3D object detectors, while point-based multisensor 3D object detectors have not been fully studied. In this paper, we propose a new 3D two-stage object detection method based on point cloud and image fusion to improve the detection accuracy. To address the problem of insufficient semantic information of point cloud, we perform multiscale deep fusion of LiDAR point and camera image in a point-wise manner to enhance point features. Due to the imbalance of LiDAR points, the object point cloud in the long-distance area is sparse. We design a point cloud completion module to predict the spatial shape of objects in the candidate boxes and extract the structural information to improve the feature representation ability to further refine the boxes. The framework is evaluated on widely used KITTI and SUN-RGBD dataset. Experimental results show that our method outperforms all state-of-the-art point-based 3D object detection methods and has comparable performance to voxel-based methods as well.

1. Introduction

3D object detection is particularly useful in autonomous driving applications, because various types of dynamic objects must be recognized in the driving environment, such as surrounding vehicles, pedestrians, and cyclists. In recent years, various 3D detectors using LiDAR point clouds have been proposed, including PointRCNN [1], Part- A^2 [2], PV-RCNN++ [3], 3DSSD [4], and CIA-SSD [5]. Although LiDAR points can capture the three-dimensional structure of an object and contain accurate depth information, they do not have sufficient semantic information and have the problem of point sparsity. Compared with LiDAR point clouds, RGB images have more regular and dense data format and have richer semantic information to distinguish between vehicles and backgrounds. Therefore, some research works [6, 7] try to estimate the position and size of objects through monocular or stereo images. However, the biggest challenge of 3D object detection based on camera image is that it cannot get accurate depth information, which is very important for 3D object detection. Considering that the representation under different sensor views have their own shortcomings, and for the 3D object detector of automatic driving, only one view input is not enough. This prompts us to design an effective framework to integrate features from different perspectives to achieve accurate 3D object detection. Early multisensor feature fusion methods take RGB image, front view, and bird's eye view (BEV) as input and then directly combine and merge the features by cropping and resizing to generate 3D candidate boxes, such as MVF [8] and AVOD [9], but they ignore the different perspectives of image and BEV. In order to reduce the accuracy loss caused by different viewing angles, ContFuse [10] uses continuous convolution to improve feature fusion, and MVAF-Net [11] uses bilinear interpolation to correct features. Although continuous convolution or bilinear interpolation is used to modify alignment to overcome the challenges of different perspectives, quantifying point cloud 3D structures into BEV pseudoimages to fusion image features will inevitably suffer a loss of accuracy. There are

also some research works [12, 13] using 3D frustum projected by 2D bounding boxes to estimate 3D bounding boxes, but these methods require additional 2D annotations and their performance is limited by 2D detectors. The above multisensor feature fusion methods all transform point clouds from sparse formation to compact representation by projecting them into images or subdividing them into uniformly distributed voxel. We call these methods voxel-based multimodal feature fusion methods, which voxelize the entire point cloud. However, the voxel-based feature fusion method will inevitably lose some information and is relatively sensitive to voxel parameters. There are also some methods that directly perform image feature fusion on LiDAR point cloud, instead of performing image feature fusion with BEV of the point cloud or voxelized pseudoimage of the point cloud. These methods are called point-based multimodal feature fusion methods. For example, PI-RCNN [14] directly fuses image features and point features, and EPNet [15] and MOT [16] perform deep fusion between point features and image features. In addition, since object detection serves the perception system of autonomous vehicle, the farther the object detected is, the more the time left for the decision planning system is, and the safer the autonomous vehicle will be. However, due to the imbalance of point clouds, the point clouds of the short-distance object are denser, and the point clouds of the long-distance object are sparse, which contains less spatial information, thus increasing the difficulty of detecting the distant object. In order to improve the detection accuracy of difficult cases, SIENet [17] predicts the shape of distant objects through point completion network to enhance the spatial structure information. Inspired by some multitask work (EPNet and SIENet), this paper proposes a point-based multimodal fusion 3D object detection method with enhanced spatial structure.

The main contributions of this paper are as follows: (1) we design a new backbone network for multimodal feature fusion, which combines LiDAR points and camera images in a point-wise manner to enhance point features without point cloud voxelization and image annotation. (2) A spatial structure enhancement module is proposed to predict the shape of object in the candidate box and learn structural information to further refine box. (3) We propose a new two-stage 3D object detection framework based on point cloud and image fusion. The test results on the KITTI benchmark show that the accuracy of our method is higher than all the current multisensor-based 3D object detection methods.

2. Related Work

3D object detection based on LiDAR: due to the sparsity and irregularity of LIDAR point cloud, traditional convolutional neural networks (CNN) cannot be directly applied to LIDAR point cloud. Many algorithms have tried various point cloud representation methods to solve this problem. Currently, there are three types of point cloud representation for the input of the 3D detector. (1) Based on the voxel representation, this method converts point clouds into regular grids through voxel transformation, so that 3D CNN can directly apply this representation. SECOND [18] divides the point cloud into voxel representations and uses sparse convolution to learn voxel features to generate 3D bounding boxes. PointPillars [19] converts point clouds into pseudoimages, eliminating the time-consuming 3D convolution operations. Fast-PointRCNN [20] introduces the attention mechanism to enhance the positioning ability of the network. The ROI-aware pooling proposed by Part- A^2 [2] refines the candidate box and improves the 3D detection accuracy. The voxel-based method has high perceptual ability, but it will cause information loss during the voxelization process of point cloud. And the storage and computing efficiency of 3D CNN are very low. (2) Based on the point representation, this method does not need to transform the original point cloud and directly uses PointNet++ [21] to process the original point cloud to obtain global features, thus retaining the original geometric information as much as possible. F-PointNet [12] proposes the application of PointNet++ [21] to 3D detection based on the cropped point cloud of 2D image bounding box. Point-RCNN [1] is the first point-based 3D object detection method that only uses point cloud as network input. 3DSSD [4] proposed a lightweight and efficient point-based single-stage 3D object detection framework, which has a good balance between accuracy and speed. (3) Point-voxel joint representation method takes points and voxels as inputs and fuses the features of points and voxels at different stages of the network for 3D object detection, such as Part- A^2 [2] and PV-RCNN++ [3]. These methods can use voxel-based perception capabilities (i.e., 3D sparse convolution) and point-based geometric structure capabilities (i.e., set abstraction) to achieve high computational efficiency and flexible receiving field, thereby improving 3D detection performance.

3D object detection based on multiple sensors: in recent years, great progress has been made in the research of multisensors such as camera image and LiDAR. AVOD [9] uses RGB image and BEV as input, proposes a feature pyramid skeleton to extract features in BEV, and combines features from BEV feature map and RGB feature map through cropping and resizing operations. ContFuse [10] applies continuous convolution to overcome the problem of different viewing angles between image and BEV. MVAF-Net [11] proposes a multiview adaptive fusion module to enhance feature fusion among image, front view, and BEV. The above methods all try to fuse the features of image and BEV, but quantifying the point cloud 3D structure into BEV pseudoimage to fuse image features will inevitably suffer accuracy loss. F-PointNet [12] uses 3D frustum projected from 2D bounding boxes to estimate 3D bounding boxes, but this method requires additional 2D annotations, and their performance is limited by 2D detectors. There are also some methods that directly perform image feature fusion on the LiDAR point cloud rather than the LiDAR BEV or the voxelized pseudoimage of the point cloud. PI-RCNN [14] directly attaches the image semantic segmentation information to the LiDAR point cloud through the transformation matrix and then uses the LiDAR detector for 3D object detection. EPNet [15] and MOT [16] establish a deep

fusion between the point cloud feature extractor and the image feature extractor to enhance the point cloud features. Although various sensor fusion networks have been proposed, they are not easily superior to LiDAR detectors because the fusion of multiview features will bring interference and noise.

3. Our Framework

In this section, we introduce a new two-stage 3D object detection framework based on point cloud and image fusion. Firstly, we describe our proposed multiscale deep fusion strategy and proposal generation layers. Next, we propose a spatial structure prediction network, including point cloud region pooling, spatial structure enhancement, and refined regression head. Finally, the loss function is discussed. Our overall framework is shown in Figure 1.

3.1. Multiscale Feature Fusion RPN. As shown in Figure 2, our multiscale feature fusion RPN consists of a point branch and an image branch. Specifically, we first use a four-layer four-scale PointNet++ to extract point features from the point cloud. Meanwhile, the image branch extracts semantic features from the image through a four-layer four-scale Unet [22] segmentation network. Finally, the proposed adaptive attention fusion (AAF) module is used to fuse the point features at different scales with corresponding image semantic features to enhance the point features.

3.1.1. Point Branch. The point branch takes LiDAR point cloud as input and generates 3D candidate boxes. The point branch is composed of four paired set abstraction (SA) and feature propagation (FP) layers for extracting point cloud features. SA consists of farthest point sampling (FPS) layer, multiscale grouping (MSG) layer, and PointNet layer, which are used for downsampling points to improve efficiency and expand the receptive field. FP consists of bilinear interpolation and multilayer perception (MLP), which is applied to broadcast feature for dropped points during the downsampling process to recover all points. Due to insufficient semantic information of LiDAR point cloud, we use LI-Fusion module [18] to fuse rich image semantic features and point features. In addition, multiscale deep fusion of point clouds and images can further enrich the point semantic features and obtain compact and discriminative feature representations. The multiscale feature fusion method is shown in Figure 2.

3.1.2. Image Branch. In order to perform multiscale semantic feature fusion, we choose the lightweight semantic segmentation network Unet that also has an encoder and decoder for image semantic feature extraction. Unet consists of four convolution blocks and four upsampling layers. Each convolution block has two repeated 3×3 convolution layers and a 2×2 maximum pooling layer. In order to obtain strong semantic features and balance GPU memory, we fine-tuned the convolution block of Unet. Our convolution block consists of two repeated 3×3 convolution layers (stride 1, padding 1) and one 3×3 convolution layer (stride 2, padding 1). Each of the first two convolution layers is followed by a batch normalization layer and a ReLU activation function, as shown in Figure 3.

3.1.3. Adaptive Attention Fusion Module. In order to fuse data from two different views, we first use the projection method to establish the relationship between LiDAR points and image pixels. Then, we obtain the semantic features of each point through grid sampling. Finally, the proposed adaptive attention fusion (AAF) module is used to perform feature fusion. Specifically, we take each point coordinate $p(x, y, z)$ through the projection matrix M to generate the corresponding image coordinate $p'(x', y')$, which can be written as

$$p' = M \cdot p, \tag{1}$$

where M is the internal parameters of the camera and the size is 3×4. Note that we convert p and p' into four-dimensional and three-dimensional vectors in homogeneous coordinates in projection formula (1). After establishing the corresponding relationship, we use the grid sample function of Pytorch framework to obtain the semantic features of each point on the image. Because the projection point may fall between adjacent pixels, the bilinear interpolation method needs to be used to obtain the image feature at the continuous coordinates, which can be written as

$$F^{(p)} = \mathscr{B}\left(F^{(N(p'))}\right), \tag{2}$$

where $F^{(p)}$ is the corresponding image feature for point p, $\mathscr{B}$ is the bilinear interpolation function, and $F^{(N(p'))}$ is the image feature of the adjacent pixels of the projection point p'. Finally, in order to better integrate point cloud features and image features, we design an adaptive attention fusion module to suppress the interference of noninterested areas and extract effective information for fusion, as shown in Figure 4. The adaptive attention fusion module can be expressed as follows:

$$\begin{cases} F_E = \tau\left(\mathrm{FC}_1\left(F_P\right) \oplus \mathrm{FC}_2\left(F_I\right)\right), \\ F_{\mathrm{PA}} = F_P \otimes \sigma\left(\mathrm{FC}_3\left(F_E\right)\right), \\ F_{\mathrm{IA}} = F_I \otimes \sigma\left(\mathrm{FC}_4\left(F_E\right)\right), \\ F_{\mathrm{PI}} = \mathrm{Concat}\left(F_{\mathrm{PA}}, F_{IA}\right), \end{cases} \tag{3}$$

where F_P and F_I represent point cloud features and point-wise image semantic features, F_E represents extended features, F_{PA} and F_{IA} represent two-branch attention features, F_{PI} represents fusion features, FC represents fully connected layer, $\oplus$ represents element-wise addition, $\otimes$ represents element-wise multiplication, σ represents the Sigmoid activation function, τ represents the Tanh activation function, and Concat represents the concatenation operation.

3.2. Spatial Structure Enhancement Module. For each candidate box generated in RPN stage, the denser the foreground point set is, the more spatial the information

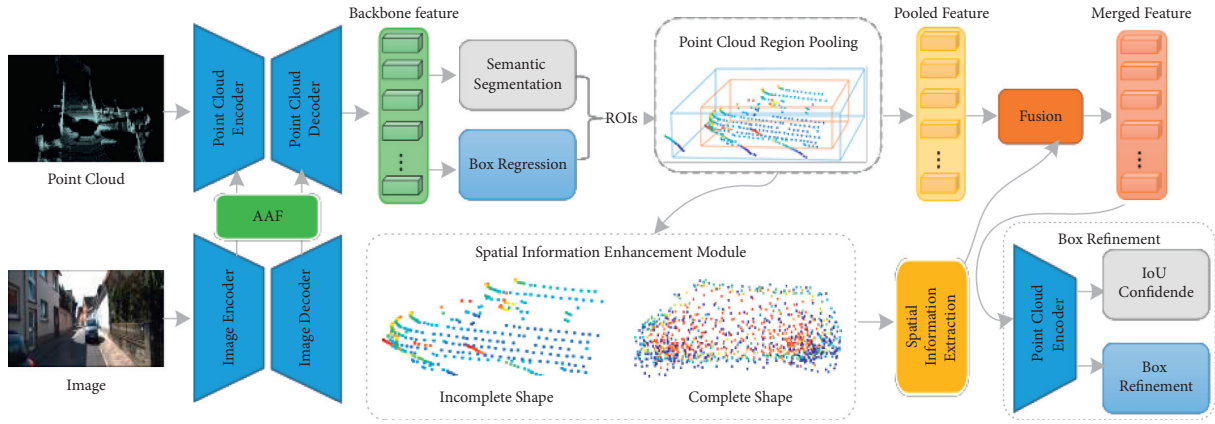

Figure 1: Description of the 3D object detection framework based on point cloud and image fusion. The whole framework consists of two stages. Stage 1 uses two-stream deep fusion RPN to extract backbone features and generate proposal boxes. Stage 2 generates high-quality 3D boxes through the spatial structure enhancement module.

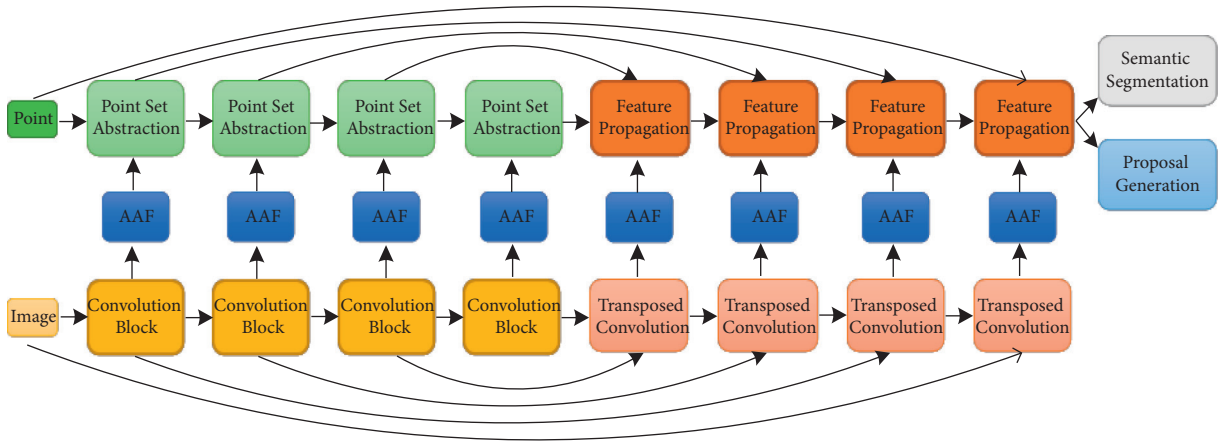

Figure 2: Description of multiscale feature fusion RPN composed of a point branch and an image branch. We use multiple AAF modules to enhance LiDAR point features with corresponding image semantic features at multiple scales.

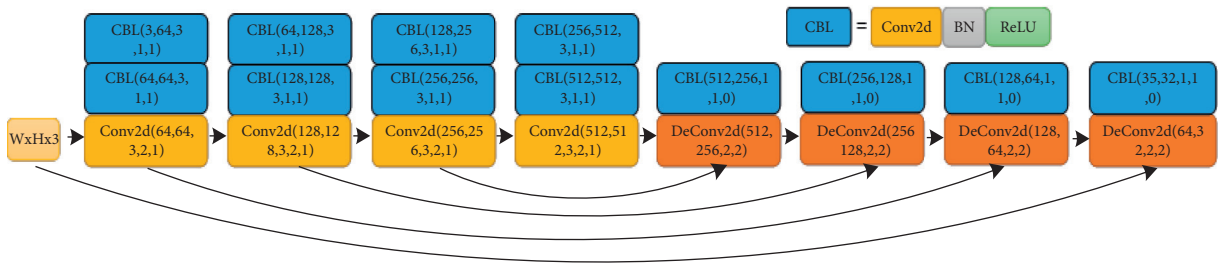

Figure 3: Details of the image backbone network. Conv2d (cin, cout, k, s, p) represents 2D convolution, and DeConv2d (cin, cout, k, s) represents 2D deconvolution, where cin, cout, k, s, and p represent the number of input channels, the number of output channels, kernel size, stride, and padding, respectively. Each convolution block consists of Convolution, BatchNorm, and ReLU.

retained is. Therefore, the central idea of our spatial information enhancement module is to predict the complete shapes of candidate objects and extract structural information to enhance feature representation. To this end, we need to solve two subtasks, namely, how to predict the spatial shape, and how to extract the spatial structural information and integrate it into the model to further refine the candidate box.

3.2.1. Point Cloud Region Pooling. After obtaining 3D bounding box proposals, we use RoI Pooling [1] to optimize the box locations and orientations. Specifically, 512 candidate regions of RPN are sampled through NMS to obtain 64 candidate regions of RCNN. For each 3D box $b_i = (x_i, y_i, z_i, h_i, w_i, l_i, \theta_i)$, we slightly enlarge it to create a new 3D box $b_i' = (x_i, y_i, z_i, h_i + \mu, w_i + \mu, l_i + \mu, \theta_i)$, so as to obtain additional context information, where (x_i, y_i, z_i) is the center

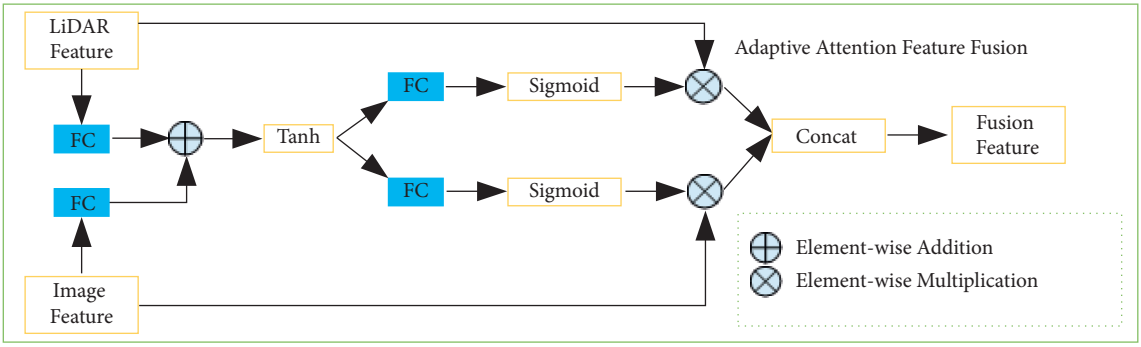

Figure 4: Illustration of the AAF module. FC represents fully connected layer, ⊕ represents element-wise addition, and ⊗ represents element-wise multiplication.

location of object b_i, (h_i, w_i, l_i) is the size of object b_i, θ_i is the object orientation of the bird's view, and μ is a constant used to expand the box size. For each point, through the segmentation mask we perform an internal/external testing, to determine whether the point is within the expanded bounding box proposal b_i'. If it is an internal point, the point and its features would be retained to refine the box b_i. Finally, we will get 512 points for each candidate box and encode them to get the pooling feature $F^p \in \Re^{512\times C_1}$, where C_1 represents the number of channels.

3.2.2. Spatial Shape Prediction. The foreground points of the candidate box constitute a shape describing semantic clues; however, this shape is usually incomplete. Therefore, based on the point completion framework PCN [23], we design a spatial structure prediction network to complete the missing part of the object in the candidate box. As shown in Figure 5, the network takes incomplete points as input and predicts the corresponding dense shape through the encoder-decoder. The encoder consists of two simple PointNet units (SharedMLP + Maxpool), each SharedMLP consisting of a 1×1 convolution layer, a BN layer, a ReLU layer, and a 1×1 convolution layer. The number of convolution output channels for the first SharedMLP is (128, 256), and the number of convolution output channels for the second SharedMLP is (512, 1024). The decoder consists of two stacked fully connected layers (Linear + BN + ReLU) and one fully connected layer (Linear), and the output is a 1024×3 matrix. The number of output channels for the three fully connected layers is (1024, 1024, $3 * 1023$). Unlike the coarse-to-fine pipeline in PCN, we believe that the coarse output is effective for subsequent processing, so we remove the fine output branch, thus saving GPU memory. In order to reduce the burden of training, we download the KITTI [23] car data set and trained our spatial shape prediction network in advance.

Figure 6 shows part of the visualization results of our spatial structure prediction model. It can be seen from the figure that our spatial structure prediction model performs well on automobiles and has a good generalization prospect.

3.2.3. Structure Information Extraction and Fusion. To obtain the local and global context from the predicted spatial shapes, we use a PointNet++ [21] module to extract the structural information. First, we use the FPS algorithm to select 512 points from the predicted shape. Then for each point, we use the Ball Query algorithm to generate a local area. Finally, the PointNet units are applied to capture the local area feature C_2 of each point, thereby obtaining the enhanced features $F^s \in \Re^{512\times C_2}$. In the refinement subnetwork part, we use a similar 3D box refinement network of PointRCNN [1] to further refine the box and confidence. The input of refining subnetwork consists of the canonical transformation coordinates of each pooling point, the pooling features, and the extracted spatial structure features. Since the pooling features and the spatial structure features come from different patterns, connecting them without any additional processing may cause interference. In order to better fusion spatial structure features and pooling features, we adopt the perspective-channel attention fusion [24] to obtain merged feature $F^m \in \Re^{512\times(C_1+C_2)}$.

3.3. Loss Function. The proposed network is trained in an end-to-end manner. Our overall losses L_{total} include the two-stream RPN loss L_{rpn} in stage 1 and the box refining network loss L_{rcnn} in stage 2 as follows:

$$L_{\text{total}} = \omega_{\text{rpn}} L_{\text{rpn}} + \omega_{\text{rcnn}} L_{\text{rcnn}}, \tag{4}$$

where ω_{rpn} and ω_{rcnn} are the coefficients that control the balance weight; we set the parameters $\omega_{\text{rpn}} = 1.0$ and $\omega_{\text{rcnn}} = 1.0$. L_{rpn} and L_{rcnn} adopt similar optimization objectives, including classification loss, regression loss, and consistency enhancement loss. For classification loss at the RPN stage, we use focal loss similar to [25] to balance positive and negative samples:

$$L_{\text{cls}}^{\text{rpn}} = \frac{-1}{N} \sum_{i=1}^{N} \begin{cases} \alpha(1-\hat{y}_i)^{\gamma} \log \hat{y}_i, & \text{if } y_i = 1, \\ (\alpha y_i + (1-\alpha)(1-y_i))(\hat{y}_i)^{\gamma} \log(1-\hat{y}_i), & \text{otherwise}, \end{cases} \tag{5}$$

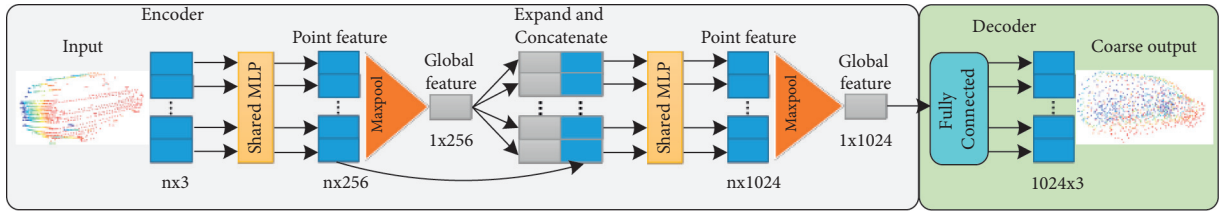

Figure 5: Graphical spatial structure prediction network. The network takes the incomplete points as input and predicts the corresponding complete shape.

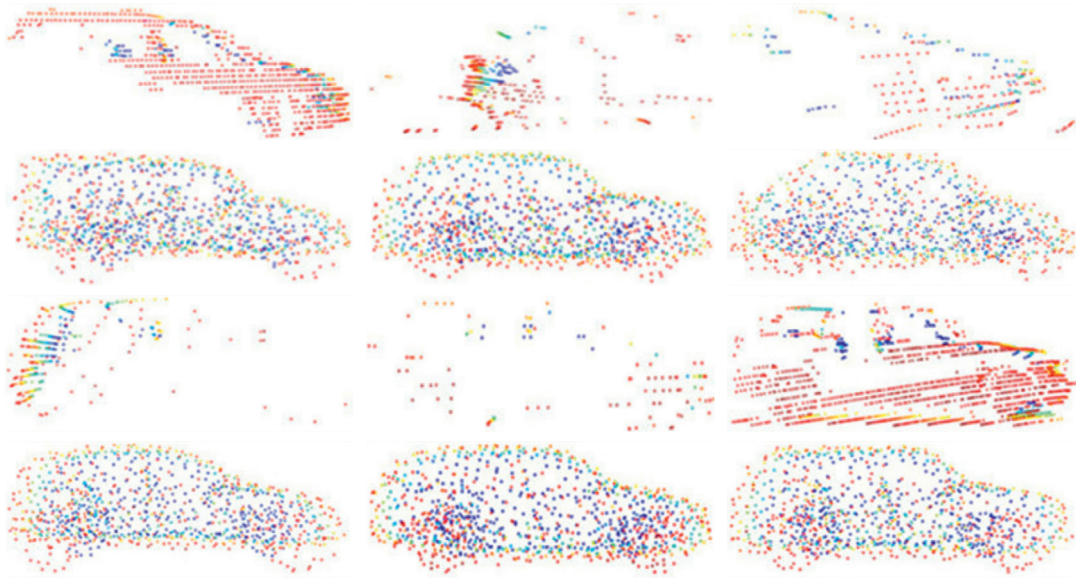

Figure 6: Visualization of spatial shape prediction of candidate objects. The original LiDAR point cloud (top) and the corresponding prediction results (bottom) for each object.

where y_i is the target classification label, $\hat{y}_i$ is the positive sample prediction probability, N represents the number of targets, and α and γ are focal loss hyperparameters. For the regression loss in the RPN stage, we adopt a bin-based regression loss similar to [1] to regress the center point (x, y, z), size (l, h, w), and orientation θ:

$$L_{\text{res}}^{\text{rpn}} = \frac{-1}{N}\sum_{i=1}^{N}\left(\sum_{u\in\{x,z,\theta\}} F_{\text{cls}}\left(\widehat{\text{bin}}_u, \text{bin}_u\right) + \sum_{u\in\{x,y,z,h,w,l,\theta\}} F_{\text{reg}}\left(\widehat{\text{res}}_u, \text{res}_u\right)\right), \tag{6}$$

where F_{cls} denotes the cross-entropy classification loss, F_{reg} denotes the smooth 1 loss, bin_u and res_u denote the bins and residuals of the ground truth, and $\widehat{\text{bin}}_u$ and $\widehat{\text{res}}_u$ denote the predicted bins and residuals of the ground truth. In addition, in order to improve the consistency of localization confidence and classification confidence, we add a consistency enhancement loss:

$$L_{\text{ce}}^{\text{rpn}} = \frac{-1}{N}\sum_{i=1}^{N}\left(\log\left(c_i \times \frac{B_i \cap B_i^{\text{gt}}}{B_i \cup B_i^{\text{gt}}}\right)\right), \tag{7}$$

where B_i represents the predicted bounding box, B_i^{gt} represents the ground truth, and c_i represents the classification confidence of the predicted box. In summary, L_{rpn} is a weighted sum of the three loss functions:

$$L_{\text{rpn}} = \omega_{\text{cls}}L_{\text{cls}}^{\text{rpn}} + \omega_{\text{reg}}L_{\text{reg}}^{\text{rpn}} + \omega_{\text{ce}}L_{\text{ce}}^{\text{rpn}}, \tag{8}$$

where ω_{cls}, ω_{reg}, and ω_{ce} are used to control the balance coefficient of the importance degree of loss. We set the parameters $\omega_{\text{cls}} = 1.0$, $\omega_{\text{reg}} = 1.0$, and $\omega_{\text{ce}} = 5.0$. Similarly, RCNN loss also includes classification loss, regression loss, and consistency enhancement loss. For RCNN classification loss, we adopt binary cross entropy loss:

$$L_{\text{cls}}^{\text{rcnn}} = -\frac{1}{N}\sum_{i=1}^{N}\left(y_i \log\,\hat{y}_i + (1-y_i)\log\left(1-\hat{y}_i\right)\right), \tag{9}$$

where y_i is the target classification label, $\hat{y}_i$ is the target prediction probability, and N is the number of targets. RCNN regression loss $L_{\text{reg}}^{\text{rcnn}}$ and consistency enhancement loss $L_{\text{ce}}^{\text{rcnn}}$ are defined in the same way as RPN. The weighted sum of the three loss functions of rcnn:

$$L_{\text{rcnn}} = \omega_{\text{cls}} L_{\text{cls}}^{\text{rcnn}} + \omega_{\text{reg}} L_{\text{reg}}^{\text{rcnn}} + \omega_{\text{ce}} L_{\text{ce}}^{\text{rcnn}}. \tag{10}$$

4. Experiments

In this section, we evaluate our method on two common 3D object detection datasets, including the outdoor dataset KITTI [26] and the indoor dataset SUN-RGBD [27]. In Section 4.1, we introduce these datasets and evaluation metrics. In Section 4.2, we provide the implementation details of the experiment. In Section 4.3 and Section 4.4, we, respectively, show the comparison results of indoor and outdoor datasets. Finally, we conducted an extensive ablation study to analyze our proposed 3D target detection model in Section 4.5.

4.1. Datasets and Evaluation Metric. KITTI is the most popular standard benchmark dataset for autonomous driving, consisting of 7,481 samples for training and 7,518 samples for testing. As a common practice, the training samples are divided into a train set with 3,712 samples and a val set with 3,769 samples. The KITTI 3D object detection benchmark uses an average accuracy (AP) with a bounding box overlap of 0.7 as the evaluation indicator for cars, where three difficulty levels (easy, moderate, and hard) are taken into consideration. SUN-RGBD is a benchmark dataset for indoor 3D target detection. The dataset consists of 10,335 images and directional 3D bounding boxes with 37 target categories, including 5,285 images for training and 5,050 images for testing. We follow the same settings in VoteNet [28] and report the performance of 10 classes on SUN-RGBD. We use the average accuracy (AP) with a 3D overlap of 0.25 as the evaluation index of SUN-RGBD. We compare our method with the state-of-the-art methods in the KITTI and SUN-RGBD test set.

4.2. Implementation Details. Two-stream RPN takes LiDAR point clouds and camera images as input. We select 1,6384 points from the raw LiDAR point cloud as the input of the point stream and take the image with a resolution of 1280×384 as the input of the image stream, which is the same as EPNet [15]. We use four SA layers (4096, 1024, 256, and 64) to subsample the input LiDAR point cloud and use four FP layers to recover the size of the point cloud for foreground segmentation and candidate box generation. Similarly, we use four convolutional blocks to downsample the input image and four transposed convolutional layers to restore the size of the image. In the NMS process, we select 8000 proposals generated by the two-stream RPN based on the classification confidence and then filter the redundant proposals with the NMS threshold of 0.8 to obtain 64 proposals for the refinement network. In the process of refining candidate boxes, we train a spatial structure prediction model in advance and then initialize the spatial shape prediction network with the weights. In the ablation experiment, we refer to the two-stage image classification strategy [29] to analyze the speed of our method. We train the model in an end-to-end manner on GeForce RTX 3090, the optimizer is ADAM [30], the initial learning rate is 0.002, and the weight attenuation is 0.001. The minibatch size is set to 2 and the model is trained for 40 epochs.

4.3. Experimental Results in KITTI. We compare the proposed two-stage detector with other state-of-the-art methods and submitted the results to the KITTI server for evaluation. As shown in Table 1, we evaluate our method on the BEV detection benchmark and 3D object detection benchmark of the KITTI test data set. It can be seen that our method is significantly ahead of the advanced single-stage multisensor methods ContFuse [10], MAFF [31], MVX-Net [32], and MVAF-Net [11] in terms of 3D mAP by 10.32%, 5.64%, 4.18%, and 1.01%, respectively. It should be pointed out that our method is a point-based two-stage multisensor method, so we focus on the performance comparison with the point-based multisensor methods. It can be seen that our method outperforms all advanced point-based multisensor methods F-PointNet [12], IDMOD [33], PI-RCNN [14], and EPNet [15] by 10.84%, 5.45%, 5.29%, and 0.47%, respectively. At the same time, our method is also superior to most voxel-based methods.

The visualization results of our method on KITTI are shown in Figure 7. For better visualization, we project the 3D bounding box of LiDAR coordinates to the RGB image. The upper part is the image 3D detection result, and the lower part is the point cloud scene detection result. It can be seen that our method performs well in capturing distant cars, although these objects are difficult to identify in RGB images and are susceptible to sparse point clouds.

4.4. Experimental Results in SUN-RGBD. We further perform experiments on SUN-RGBD data sets to verify the effectiveness of our method in indoor scenarios. Table 2 shows the results compared with the most advanced methods. Our method achieves excellent detection performance, outperforming PointFusion [35], F-PointNet [12], VoteNet [28], MBDF-NET [36], and EPNet [15] by 16.1%, 6.2%, 2.5%, 0.7%, and 0.4%, respectively. Specifically, F-Pointnet and VoteNet both estimate 3D boundary boxes of point clouds based on 2D boundary box projections of images. PointFusion combines point cloud features and image features in a concatenation fashion. Different from them, our method establishes a correspondence between image features and point features, thus providing a clearer representation. In addition, comparing with multisensor-based methods, EPNet and MBDF-NET are particularly valuable. Because they also establish the mapping relationship between image features and point features, EPNet and MBDF-NET do not consider the point cloud sparse problem, and MBDF-NET is a three-branch detector.

Table 1: Comparison with state-of-the-art methods on the KITTI test server.

Type	Method	Modality	3D detection (car)				Bev detection (car)			
			Easy	Mod.	Hard	3D mAP	Easy	Mod.	Hard	Bev mAP
Stage 1	SECOND [18]	LiDAR	84.65	75.96	68.71	76.44	91.81	86.37	81.04	86.41
	3DSSD [4]	LiDAR	88.36	79.57	74.55	80.83	92.66	89.02	**85.86**	89.18
	MAFF [31]	LiDAR & img.	85.52	75.04	67.61	76.06	90.79	87.34	77.66	85.26
	MVX-Net [32]	LiDAR & img.	85.99	75.86	70.70	77.52	91.86	86.53	81.41	86.60
	MVAF-Net [11]	LiDAR & img.	87.87	78.71	**75.48**	80.69	91.95	87.73	85.00	88.23
	PointRCNN [1]	LiDAR	86.96	75.64	70.70	77.77	92.13	87.36	82.72	87.41
	Fast PointRCNN [20]	LiDAR	85.29	77.40	70.24	77.64	90.87	87.84	80.52	86.41
	Part-A^2 [2]	LiDAR	87.81	78.49	73.51	79.94	91.70	87.79	84.61	88.03
Stage 2	F-PointNet [12]	LiDAR & img.	82.19	69.79	60.59	70.86	91.17	84.67	74.77	84.54
	PI-RCNN [14]	LiDAR & img.	84.37	74.82	70.03	76.41	91.44	85.81	81.00	86.08
	EPNet [15]	LiDAR & img.	89.81	79.28	74.59	81.23	94.22	88.47	83.69	88.79
	IDMOD [33]	LiDAR & img.	84.50	75.41	68.83	76.25	89.43	86.46	78.93	84.94
	F-PointPillars [34]	LiDAR & img.	88.90	79.28	78.07	82.08	90.20	89.43	88.77	89.47
	Our	LiDAR & img.	**89.94**	**79.89**	75.24	**81.70**	**94.39**	**88.84**	84.39	**89.21**

The bold value indicates the highest performance.

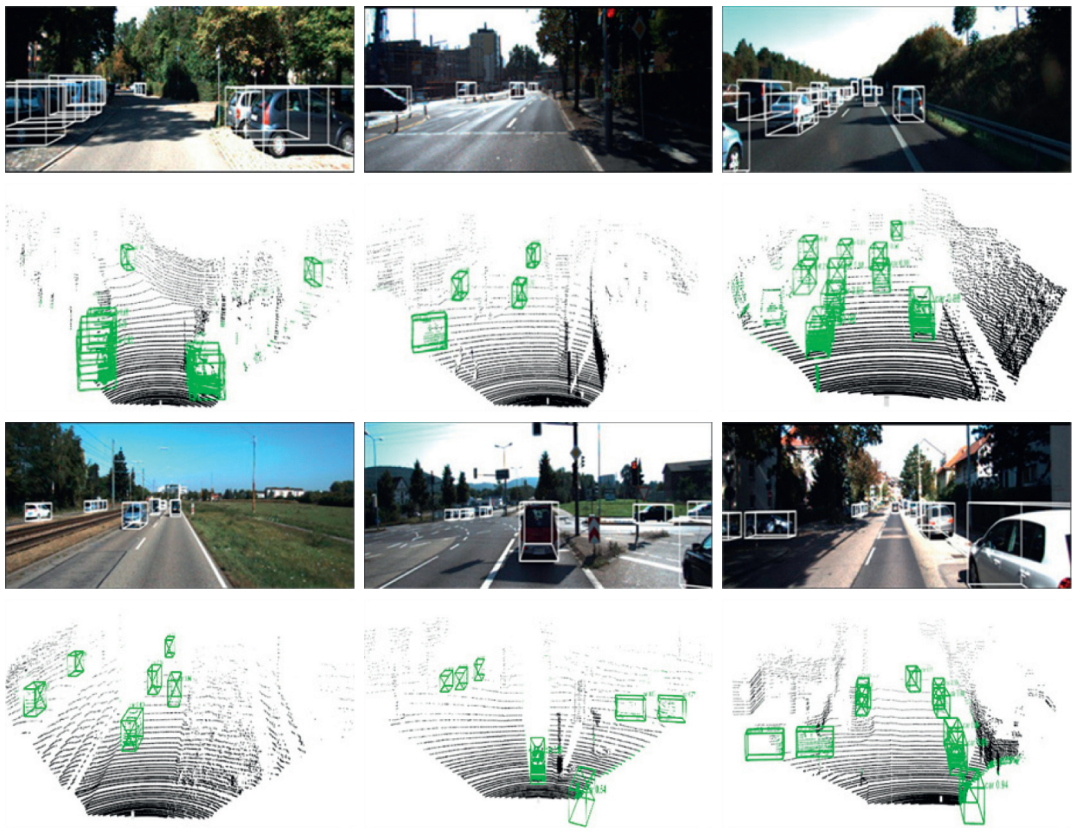

Figure 7: Qualitative results of our method on the KITTI dataset. The detection results are shown in the image (upper) and the corresponding point cloud (lower).

Table 2: Quantitative comparison with advanced methods on the SUN-RGBD test set.

Method	Modality	Bathtub	Bed	Bookshelf	Chair	Desk	Dresser	Nightstand	Sofa	Table	Toilet	3D mAP
PointFusion [35]	L & I	37.3	68.6	**37.7**	55.1	17.2	24.0	32.3	53.8	31.0	83.8	44.1
F-PointNet [12]	L & I	43.3	81.1	33.3	64.2	24.7	**32.0**	58.1	61.1	51.1	**90.9**	54.0
VoteNet [28]	L	74.4	83.0	28.8	75.3	22.0	29.8	62.2	64.0	47.3	90.1	57.7
EPNet [15]	L & I	75.4	85.2	35.4	75.0	26.1	31.3	62.0	67.2	52.1	88.2	59.8
MBDF-Net [36]	L & I	**81.5**	84.7	33.0	**77.3**	**31.2**	29.0	57.7	65.6	49.9	85.5	59.5
Our	L & I	75.6	**85.4**	35.5	75.6	26.4	31.6	**62.5**	**67.7**	**52.8**	88.6	**60.2**

L and I represent the LiDAR point cloud and camera image.

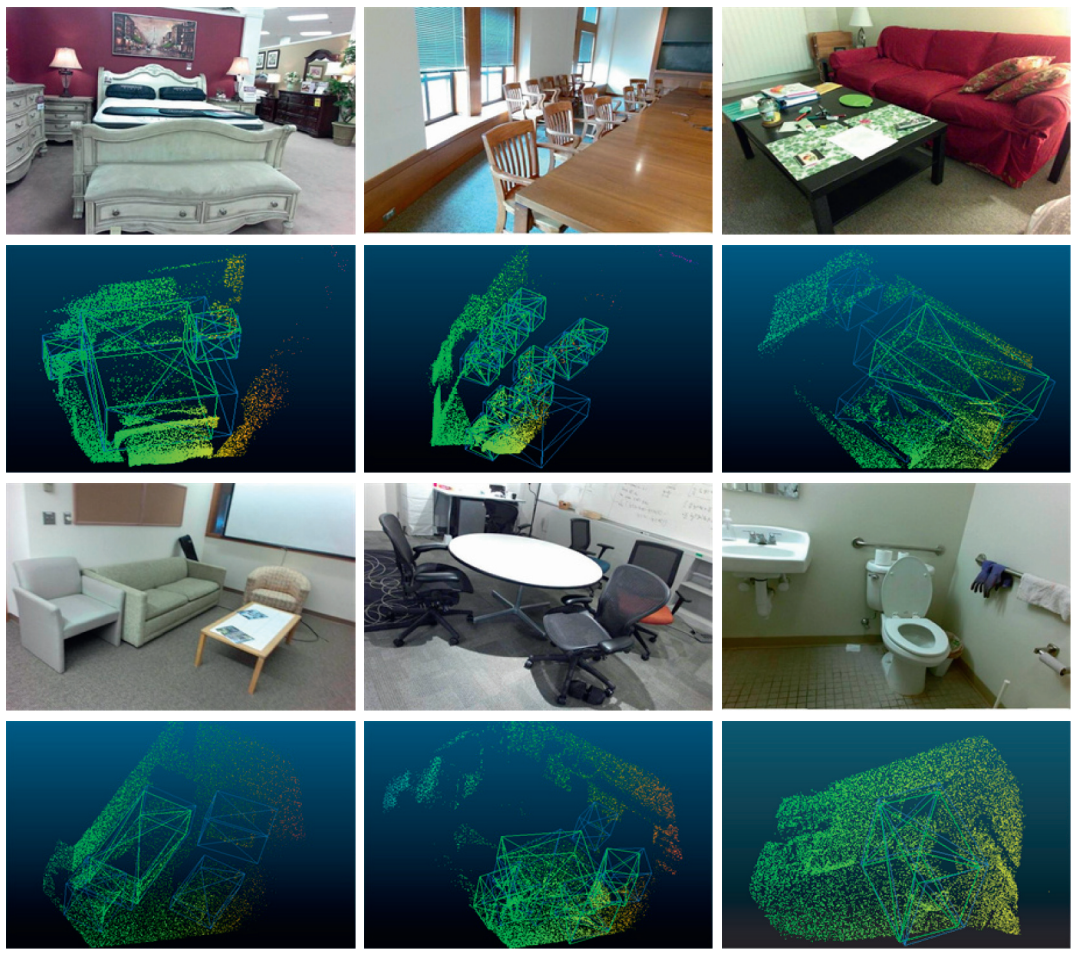

Figure 8: Qualitative results of our method on the SUN-RGND dataset. For each pair, the camera image is shown above and the corresponding point cloud detection result is shown below. The ground truth and detected boxes are highlighted with green and blue boxes, respectively.

The visualization results of our method on SUN-RGBD are shown in Figure 8. Unlike the KITTI dataset, the SUN-RGBD dataset contains objects of multiple categories and different scales. It can be seen from Figure 8 that our method can better detect a variety of objects with obvious scale changes, including small objects (such as chair and dressing table) and large objects (such as sofa and bed).

4.5. Ablation Studies. We conduct a series of ablation studies on the KITTI dataset to analyze multiscale fusion RPN and spatial structure enhancement modules. All models are trained on the training set and evaluated on the validation set of the KITTI dataset for car detection. All evaluations on the validation set are conducted through 40 recall positions.

4.5.1. Effect of Multiscale Fusion RPN. In Table 3, we investigate the effectiveness of different structures in multiscale fusion RPN. We analyze the effect of each structure on stage 1 by removing one structure while leaving the others unchanged. To be fair for comparison, all the experiments shared the same fixed state 2. In the first row, we remove the image semantic branch, and the performance decreases significantly,

Table 3: Effect of multiscale fusion module.

Method	Easy	Moderate	Hard	3D mAP	Time (ms)
RPN baseline	85.66	76.48	76.05	79.40	80
SFP	91.59	82.32	79.89	84.6	93
MFP	**92.2**	**82.49**	**79.87**	**84.85**	**105**

SFP: single-scale feature propagation layer fusion; MFP: multiscale feature propagation layer fusion.

which demonstrates the advantage of semantic segmentation. Then we compare two different fusion schemes. One is the single-scale feature propagation layer (SFP) fusion, which is similar to the multisensor feature fusion backbone network of EPNet [15], and the image semantic features are fused with the last feature propagation layer. The other is multiscale feature propagation layer (MFP) fusion, where image semantic features are fused with each feature propagation layer (see Figure 2). The results show that MFP is better than SFP by 0.25% in 3D mAP. This shows that the application of semantic features on multiscale feature propagation layer is effective. At the same time, we also give the inference time in Table 3. It can be seen that the inference time of SFP is similar to the baseline, and the time consumption of MFP does not increase much.

Table 4: Performance with the number of different convolution layers.

Layer	Easy	Moderate	Hard	3D mAP	Time (ms)
2	92.2	82.49	79.87	84.85	105
3	**92.21**	**82.54**	**80.22**	**84.99**	**119**
5	91.38	82.33	80.19	84.63	137

Table 5: Performance of point cloud region pooling with different context widths.

Context width	Easy	Moderate	Hard	3D mAP
0	92.12	82.12	79.67	84.64
0.2	**92.21**	**82.54**	**80.22**	**84.99**
0.4	91.70	82.45	79.93	84.69

4.5.2. Effect of Convolution Layer. Table 4 shows the effects of different convolution layers on the performance of image semantic segmentation. We take the convolution layer number of Unet convolution block as the baseline. When the number of convolutional layers of the convolution block is increased appropriately, the AP is slightly increased, but excessively increasing the number of convolution layers of convolution blocks will reduce AP. This is because a reasonable depth of convolutional neural network can extract more image semantic features, but too deep network will lead to overfitting, which is not good for convergence. At the same time, it can be seen from Table 4 that the inference time increases slightly with the increase of the number of convolutional layers.

4.5.3. Effect of Point Cloud Region Pooling. Table 5 shows the effects of different pool context widths on performance. When no context information is pooled, the accuracy of 3D object detection, especially for those difficult instances, decreases significantly. Because the object might be obscured or far away from the sensor, difficult cases often have fewer points in the candidate box, which requires more contextual information to classify and refine the candidate box. As shown in Table 5, too large pooling context width can also result in performance drops because the pooled region of the current candidate box may include noisy foreground points for other objects.

4.5.4. Effect of Spatial Structure Enhancement. We explore the effects of the spatial information enhancement module in Table 6. In the first row, we do not use the spatial information enhancement module. In the second row, we add the spatial information enhancement module and only use the simplest connection fusion, which reduces AP. This is because the pooling features and the spatial structure features come from different patterns, and connecting them without any additional processing produces interference. In the third row, we use perspective-channel attention fusion to fuse the spatial information enhancement module, and the gain of mAP is 0.42%. This is because the spatial information enhancement module promotes the model to better obtain spatial information. In addition, the inference time of our spatial information enhancement module is only increased by 11 ms compared with the RCNN baseline.

Table 6: Effect of spatial information enhancement module.

Method	Easy	Moderate	Hard	3D mAP	Time (ms)
RCNN baseline	92.21	82.54	80.22	84.99	119
SSE-con	91.85	82.28	80.00	84.71	127
SSE-att	**92.61**	**83.00**	**80.63**	**85.41**	**130**

SSE-con represents the spatial information enhancement module of simple connection. SSE-att represents the spatial information enhancement module based on attention fusion.

5. Conclusion

In this paper, we introduce a multiscale fusion RPN for features extraction and proposals generation. Besides, we also propose a novel spatial information enhancement module for detecting 3D objects from point clouds with the imbalanced density. Specifically, we design a spatial structure enhancement module to generate the complete shape of the candidate box and learn the structural information to enhance the features for box refinement. A large number of experiments verify the effectiveness of our proposed framework.

Data Availability

The data used to support the findings of this study are available at https://github.com/liuhuaijjin/EPFSI/.

Conflicts of Interest

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

Acknowledgments

This work was supported by the Natural Science Foundation of China (nos. 61673186 and 61871196), the Natural Science Foundation of Fujian Province of China (no. 2019J01082), and the Promotion Program for Young and Middle-Aged Teachers in Science and Technology Research of Huaqiao University (ZQN-YX601).

References

[1] S. Shi, X. Wang, and H. Li, "Pointrcnn: 3d object proposal generation and detection from point cloud," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 770–779, Long Beach, CA, USA, June 2019.

[2] S. Shi, Z. Wang, J. Shi, X. Wang, and H. Li, "From points to parts: 3d object detection from point cloud with part-aware and part-aggregation network," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, pp. 2647–2664, 2021.

[3] S. Shi, L. Jiang, J. Deng et al., "Pv-rcnn++: Point-voxel Feature Set Abstraction with Local Vector Representation for 3d Object Detection," 2021, https://arxiv.org/abs/2102.00463.

[4] Z. Yang, Y. Sun, S. Liu, and J. Jia, "3DSSD: Point-based 3d single stage object detector," in *Proceedings of the IEEE/CVF*

Conference on Computer Vision and Pattern Recognition, Article ID 11040, Seattle, WA, USA, June 2020.

[5] W. Zheng, W. Tang, S. Chen, L. Jiang, and C. W. Fu, "Cia-ssd: Confident Iou-Aware Single-Stage Object Detector from point Cloud," 2020, https://arxiv.org/abs/2012.03015.

[6] W. Bao, B. Xu, and Z. Chen, "Monofenet: monocular 3d object detection with feature enhancement networks," *IEEE Transactions on Image Processing*, vol. 29, pp. 2753–2765, 2019.

[7] X. Mo, U. Sajid, and G. Wang, "Stereo frustums: a siamese pipeline for 3d object detection," *Journal of Intelligent and Robotic Systems*, vol. 101, no. 1, pp. 1–15, 2021.

[8] Y. Zhou, P. Sun, Y. Zhang et al., "End-to-end multi-view fusion for 3d object detection in lidar point clouds," in *Proceedings of the Conference on Robot Learning, PMLR*, pp. 923–932, Cambridge MA, USA, November 2020.

[9] J. Ku, M. Mozifian, J. Lee, A. Harakeh, and S. L. Waslander, "Joint 3d proposal generation and object detection from view aggregation," in *Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1–8, IEEE, Madrid, Spain, October 2018.

[10] M. Liang, B. Yang, S. Wang, and R. Urtasun, "Deep continuous fusion for multi-sensor 3d object detection," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 641–656, Munich, Germany, September 2018.

[11] G. Wang, B. Tian, Y. Zhang, L. Chen, D. Cao, and J. Wu, "Multi-view adaptive fusion network for 3d object detection," 2020, https://arxiv.org/abs/2011.00652.

[12] C. R. Qi, W. Liu, C. Wu, H. Su, and L. J. Guibas, "Frustum pointnets for 3d object detection from rgb-d data," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 918–927, Salt Lake City, UT, USA, June 2018.

[13] H. Zhang, D. Yang, E. Yurtsever, K. A. Redmill, and Ü. Özgüner, "Faraway-frustum: Dealing with Lidar Sparsity for 3d Object Detection Using Fusion," 2020, https://arxiv.org/abs/2011.01404.

[14] L. Xie, C. Xiang, Z. Yu et al., "PI-RCNN: an efficient multi-sensor 3D object detector with point-based attentive cont-conv fusion module," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 7, Article ID 12460, NY, USA, February 2020.

[15] T. Huang, Z. Liu, X. Chen, and X. Bai, "Epnet: enhancing point features with image semantics for 3d object detection," in *Proceedings of the 2020 European Conference on Computer Vision*, pp. 35–52, Springer, Glasgow, UK, August 2020.

[16] K. Huang and Q. Hao, "Joint Multi-Object Detection and Tracking with Camera-Lidar Fusion for Autonomous Driving," 2021, https://arxiv.org/abs/2108.04602.

[17] Z. Li, Y. Yao, Z. Quan, W. Yang, and J. Xie, "Sienet: Spatial Information Enhancement Network for 3d Object Detection from point Cloud," 2021, https://arxiv.org/abs/2103.15396.

[18] Y. Yan, Y. Mao, and B. Li, "Second: sparsely embedded convolutional detection," *Sensors*, vol. 18, no. 10, 2018.

[19] J. Stanisz, K. Lis, T. Kryjak, and M. Gorgon, "Optimisation of the pointpillars network for 3d object detection in point clouds," in *Proceedings of the 2020 Signal Processing: Algorithms, Architectures, Arrangements, and Applications (SPA)*, pp. 122–127, IEEE, Poznan, Poland, September 2020.

[20] Y. Chen, S. Liu, X. Shen, and J. Jia, "Fast point r-cnn," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9775–9784, Seoul, Republic of Korea, August 2019.

[21] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep Hierarchical Feature Learning on point Sets in a Metric Space," 2017, https://arxiv.org/abs/1706.02413.

[22] O. Ronneberger, P. Fischer, and T. Brox, "U-net: convolutional networks for biomedical image segmentation," in *Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 234–241, Springer, Munich, Germany, October 2015.

[23] W. Yuan, T. Khot, D. Held, C. Mertz, and M. Hebert, "Pcn: point completion network," in *Proceedings of the 2018 International Conference on 3D Vision (3DV)*, pp. 728–737, IEEE, Verona, Italy, September 2018.

[24] B. Yang, J. Wang, R. Clark et al., "Learning Object Bounding Boxes for 3d Instance Segmentation on point Clouds," 2019, https://arxiv.org/abs/1906.01140.

[25] T. Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2980–2988, Venice, Italy, October 2017.

[26] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: the kitti dataset," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.

[27] K. Chen, Y. K. Lai, and S. M. Hu, "3d indoor scene modeling from rgb-d data: a survey," *Computational Visual Media*, vol. 1, no. 4, pp. 267–278, 2015.

[28] C. R. Qi, O. Litany, K. He, and L. J. Guibas, "Deep hough voting for 3d object detection in point clouds," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9277–9286, Seoul, Republic of Korea, November 2019.

[29] J. Zhou, S. Zeng, and B. Zhang, "Two-stage knowledge transfer framework for image classification," *Pattern Recognition*, vol. 107, Article ID 107529, 2020.

[30] D. P. Kingma and J. Ba, "A Method for Stochastic Optimization," 2014, https://arxiv.org/abs/1412.6980.

[31] Z. Zhang, M. Zhang, Z. Liang et al., "Maff-net: Filter False Positive for 3d Vehicle Detection with Multi-Modal Adaptive Feature Fusion," 2020, https://arxiv.org/abs/2009.10945.

[32] V. A. Sindagi, Y. Zhou, and O. Tuzel, "Mvx-net: multimodal voxelnet for 3d object detection," in *Proceedings of the 2019 International Conference on Robotics and Automation (ICRA)*, pp. 7276–7282, IEEE, Montreal, QC, Canada, May 2019.

[33] R. Khamsehashari and K. Schill, "Improving deep multimodal 3d object detection for autonomous driving," in *Proceedings of the 2021 Seventh International Conference on Automation, Robotics and Applications (ICARA)*, pp. 263–267, IEEE, Prague, Czech Republic, February 2021.

[34] A. Paigwar, D. S. Gonzalez, O. Erkent, and C. Laugier, "Frustum-pointpillars: a multi-stage approach for 3d object detection using rgb camera and lidar," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 2926–2933, Montreal, BC, Canada, October 2021.

[35] D. Xu, D. Anguelov, and A. Jain, "Pointfusion: Deep sensor fusion for 3d bounding box estimation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 244–253, Salt Lake City, UT, USA, June 2018.

[36] X. Tan, X. Chen, G. Zhang, J. Ding, and X. Lan, "Mbdf-net: Multi-branch deep fusion network for 3d object detection," 2021, https://arxiv.org/abs/2108.12863.