Multi-Scale Keypoints Feature Fusion Network for 3D Object Detection from Point Clouds

Xu Zhang¹*, Linjuan Bai¹, Zuyu Zhang², and Yan Li²

Abstract

Lidar based object detection is utilized in numerous applications. The point-based and voxel-based methods employ furthest point sampling (FPS) algorithms and feature up-sampling to reduce the high computation cost respectively. In addition, aggregating all feature vectors for prediction leads to a large cost, the object confidence and location estimate are affected. Therefore, we propose a novel multi-scale keypoints feature fusion framework for 3D object detection, which take advantage of the 3D voxel convolutional neural network and PointNet-based set abstraction to learn more discriminative point features. A feature FPS set abstraction module is proposed to aggregate 3D voxel-wise features which can handle point loss and redundancy with learning complex features. A multi-scale feature fusion strategy is used to acquire context information and location information with multiple receptive fields, which reduces the computation cost. Finally, a refine prediction head is designed to improve box refinement and confidence prediction. We evaluate our model on benchmark KITTI which exhibit good performance and achieved 2.09% improvement in car hard difficulty. The feature FPS set abstraction module and fusion strategy outperform the state-of-the-arts work by 2.13% in all categories at different difficulties.

Keywords
Point Cloud, 3D Object Detection, Multi-Scale, Keypoints, Feature Fusion, Furthest Point Sampling

1. Introduction

In recent years, lidar-based object detection, the same as RGB-D based and Lidar-RGB based 3D object detection, has been receiving increasing attention in numerous applications such as autonomous driving and robotics. Lidar sensors are widely adopted in autonomous driving vehicles and robots for capturing 3D scene information as sparse and irregular point clouds [1]. Decreasing the occurrence of accidents with the assistance of science and technology has become a significant research topic in computer vision and intelligent vehicle infrastructure cooperative systems (IVICS). With the widespread popularity of 3D object detection in autonomous driving, augmented reality, and robotics. Lidar sensors gradually enter people's horizons to capture much richer spatial information and structural information of the surrounding environment which provides vital clues for 3D scene perception and understanding [2].
Recently, 2D object detection has been made huge breakthroughs [3]. Applications for object detection are expanding [4]. However, it is impossible to directly transform 2D approaches to 3D object detection on account of point clouds is unstructured distribution, sparse, irregular, and locality sensitive [5]. Most existing 3D detection methods could be classified into two categories in terms of sensor amount: multi-sensor methods [6–11] and lidar-only methods [12–23]. The lidar-RGB-based method, which belongs to multi-sensor methods, aims at seeking an efficient solution from images to promote high-performance object detection. The lidar-only method throws away complicated matches between the point clouds and images by using the point directly. Three mainstream methods to deal with point cloud are slowly derived: voxel-based method [12–17], point-based method [18–20], and graph-based method [21–23]. Point-based methods take raw data as input and predict bounding boxes. Voxel-based methods conduct voxelization on the entire point cloud to abstract each single voxel feature by 3D or 2D convolutional neural networks (CNN). The graph-based methods apply spatial information by establishing the relationship between vertex (point) and edge (the relationship between points pair). Subsequently, graph neural network (GNN) is used to learn point features for 3D detection.

The aforementioned methods such as voxel-based methods are more computationally efficient suffer from inevitable information loss during voxelization and encounter a performance bottleneck. The point-based method considers the spatial distance to select keypoints and ignores local feature correlation. As results, the height of the bounding box, occlusion, and truncation levels have a significant impact on the recognition. The reason is that these methods lose information in the downsampling process. Furthest point sampling (FPS) based on 3D Euclidean distance, not only ground-truth with few interior points lose points [20], but the survival sampling points also have a lot of redundancy. The subsequent classification and prediction are based on the feature extraction of the points, which causes the problem of fewer effective keypoints. For small objects with high occlusion and truncation levels, the recognition performance is affected. Selection algorithms are used to remove the irrelevant features from the data set. Kishor and Chakraborty [24] used the fast correlation-based filter feature (FCBF) selection technique to find the best features. Grape leaf disease detection network (GLDDN) [25] is proposed that utilizes dual attention mechanisms for feature evaluation, detection, and classification. X-view [26] proposed a non-egocentric multi-view detection method to overcome the grid partition coarse of the multi-view.

To address the above issues, we propose a novel two-stage 3D object detection network to predict 3D bounding boxes and object categories for each instance in the point cloud scene. The multi-scale keypoints feature fusion network combines the advantages from both the 3D voxel convolutional neural network and PointNet-based set abstraction to learn more discriminative point cloud features. Firstly, we voxelization the whole point cloud with 3D sparse convolution for voxel-wise feature learning and proposal generation. A small set of keypoints are selected by the 3D Euclidean distance FPS to summarize the overall 3D location information from the voxel-wise features in the parallel time. Secondly, a subset of keypoints is picked up by using 3D Euclidean feature FPS. The subset of the keypoints feature comes from 3D sparse convolution by grouping the neighboring voxel-wise features. As a result, it can effectively preserve interior points of various instances. Different from previous work. We choose the number of different measured keypoints in the sampling process for the minimum representative points. We abandon concatenating entire 3D sparse convolution features or other multi-perspective mapping features as before. Instead, we present a fusion strategy to better use different receptive fields. In the end, we conduct keypoints feature set abstraction to aggregate the features to object classification and location regression.

In summary, our primary contribution is manifold:

- We propose a feature FPS set abstraction module to aggregate 3D voxel-wise features above surrounding voxel volume, several minimum representative points will be selected.
- A multi-scale feature fusion strategy is designed in set abstraction layers after the point cloud voxelization, which can extract accurate context information and location information.
- We design a refine prediction head, making the proposed framework use different keypoints feature to improve box refinement and confidence prediction respectively.
The remaining parts of this paper are organized as follows. In Section 2, related works of 3D object detection are reviewed. Section 3 presents our framework and experiments. Finally, Section 4 gives the conclusion of this paper.

2. Related Work

Existing work can be classified into two categories in terms of sensor type: multi-sensor methods and single-sensor methods. In this paper, we focus on lidar-only method that belongs to Single-sensor methods. Point clouds can be scanned directly from light detection and ranging (LIDAR) technique [27]. The lidar-only object detection methods can be categorized as voxel-based method, point-based method, and graph-based method. All those can be grouped into two categories according to the pipeline: one-stage and two-stage pipeline for 3D object detection. The technical routes for lidar-only method is shown in Fig. 1.

![Fig. 1. Technical routes for 3D object detection.](image)

2.1 Multi-Sensor based 3D Object Detection

As a consequence of sensor development, quantity transportations are equipped with multiple sensors, such as lidar and camera. Laser scanners have accurate depth information, coming with the character of sparse unordered, and locality sensitive. Cameras provide more semantic information with weak depth and location information. The combination of two different sensors is perhaps an elegant solution to achieve high performance. Series works [6–11] take advantage of the image to detect 2D bounding box previous or project point cloud into multiple views and extract the corresponding features. Multi-view 3D network (MV3D) [6] generates candidate boxes in the bird’s eye view (BEV) representation with a fixed height from the point cloud. Aggregate view object detection network (AVOD) [9] improves MV3D by introducing image features into the candidate boxes generation module to improve MV3D using fixed height. Frustum-PointNet [7] attempts to generate 2D detection boxes from the given images and stretch the 2D detection box to the frustum plane of the point cloud to cut off the non-frustum plane to reduce calculation. Multi-task multi-sensor fusion network (MMF) [11] integrates depth map, point cloud, and image information to complete multiple tasks including depth information detection, 2D object detection, and 3D object detection. PFF3D [28] presents a fusion method to exploit both LIDAR and camera data. Those aforementioned tasks need to be calibrated and synchronized.

2.2 Lidar-Only based 3D Object Detection
Combining image and point cloud for object detection has bottlenecks with data alignment. The use of a truncated point cloud under the frustum plane for object detection is limited by the performance of the 2D detection boxes. To avoid this circumstance, there are three mainstream typical skills used for feature learning: voxel-based, point-based, and graph-based networks.

**Voxel-based:** On the one hand, in the 2D object detection domain, CNNs are used to extract features. On the other hand, the point cloud irregular format and non-uniformity introduces a challenge for feature learning. Numerous researchers subdivide raw point cloud to equally distributed which can be processed by 3D CNN like as 2D CNN. VoxelNet [12] uses voxelization to tackle point cloud firstly. SECOND [13] introduced an efficient 3D voxel process using 3D sparse convolution. Consequently, feature learning in dense data can be generalized to sparsely uniformly distributed sampled regions thanks to voxelization.

**Point-based:** Although using voxelization can embed the entire scene with voxels. Nevertheless, renders data unnecessary information loss and causes issues. Rather than passing to an intermediate regular representation. PointNet [29] provides non-trivial thinking by using neural networks to extract unordered points features directly. They take raw point as input and use multi-layer perceptron (MLP) to map low dimensional feature to high dimensional feature space to ensure the network invariance under transformations. PointNet++ [30] is a variant of PointNet which proposes hierarchical aggregation of point features and uses iterative FPS to choose a subset of points to cover the entire point set for further feature extraction. Many works take PointNet and PointNet++ as a backbone and design different models to achieve better performance. STD [19] proposes spherical anchors aiming to retain location information and use PointNet++ as the backbone to extract semantic context features. 3DSSD [20] uses PointNet to sample the keypoints with the feature distance on the point cloud.

**Graph-based:** The common problem of using PointNet++ as a backbone network or set abstraction module is the grouping and sampling points repeatedly. GNN was first proposed by Gori et al. [31]. It is a neural network that directly acts on the graph structure. It is a natural generalization of CNNs in non-Euclidean space. Dynamic graph CNN (DGCNN) [32] seems the first module adopts GNN to feature extraction on point clouds including classification and segmentation in our knowledge. EdgeConv was proposed to act on graphs dynamically computed in each layer of the network. SawNet [33] combines the cogitation of PointNet based on DGCNN to learn local and global features in point cloud classification and segmentation tasks. Point-GNN [22] uses voxel down sample point cloud for the graph construction, they regard the point as vertex and use fixed radius reconstruction a graph, the neural network on sets to extract features, an auto-registration mechanism was proposed to ensure invariance under transformations. Sparse voxel-graph attention network (SVGA-Net) [23] constructs the local complete graph within each spherical voxel and the global KNN graph to learn the discriminative feature representation. S-AT GCN [34] proposes a spatial-attention graph convolution to form the feature enhancement layer.

As for the two-stage pipeline, they employ images and point clouds, or only point as input to generate 2D bounding boxes. Point cloud converts to point-wise or voxel-wise representation at the same time. Set abstraction (SA) layers are used on point features representation for extracting context information. 3D region proposal network (RPN) is adopted for generating a series of rough proposal boxes. Based on those proposal boxes, a refinement module is developed to find definitive predictions at the second stage.

As for the one-stage pipeline, they provide a straightforward solution to category classification and location regression, accompanying by enduring a serious imbalance of positive and negative proposal boxes. Notably, be different from the two-stage pipeline, FP layers and the refinement stage are figured that consumes half of the inference time. 3DSSD [20] removes FP layers and the refinement stage to prediction bounding box directly.

Our work differs from previous work by using distance farthest point to represent the difference between the points in the whole scene for the two-stage model. The motivation lies in the fact that using distance farthest point contains more location information and feature farthest point contains more semantic information. These two different kinds of points possibly affect the classification and regression of objects respectively. Effective coding representation of multi-scale features based on voxel
operation creates favorable conditions for selecting different kinds of keypoints under multi-scale features. We reckon that combining the advantages from both the 3D voxel CNN with different sampling strategies and PointNet-based set abstraction is the right procedure to classification and regression objects, respectively.

3. Framework

In this section, we analyze and introduce the main modules of the 3D point cloud object detection model. First, we introduce “distance FPS voxel set abstraction module” and “feature FPS voxel set abstraction module”; then we propose a box prediction module for proposal generation. Finally, we present a multi-scale keypoints feature fusion strategy and refine prediction head (Fig. 2).

Fig. 2. The overall architecture contains four main components. (1) The backbone network includes 3D voxelization, 3D sparse convolution, and the BEV projection. (2) Distance FPS voxel set abstraction module. (3) Feature FPS voxel set abstraction module. (4) Refine prediction head.

3.1 Distance FPS Voxel Set Abstraction Module

To encode voxel features, we need to obtain keypoints. Specifically, we use the distance FPS (D-FPS) method to select a point cloud sequence from the point cloud $P$.

The spatial distance measurement is formulated as.

$$D - Distance = \|X - Y\|_2 = \left\{\sqrt{\sum_{\mathbb{R}} X_i^2 \mathbb{R} + Y_j^2 \mathbb{R} - 2X_i Y_j \mathbb{R}} \right\}_{X_i, Y_j \in P} \quad (1)$$

$$d_{\text{Distance}} = \min_{\mathbb{R} \in \mathbb{R}_3} \left\{\sqrt{\sum_{\mathbb{R}} X_i^2 \mathbb{R} + Y_j^2 \mathbb{R} - 2X_i Y_j \mathbb{R}} \right\}$$

D-Distance represents the L2 distance between two points. $X$ and $Y$ represent coordinates and reflection intensity from different points. A small number of representative points d-keypoint sequence $dp = \{p_1, p_2, p_3 \ldots p_n\}$ is generated with D-FPS, which represent the distribution of position information in the point cloud space. A 3D voxel sparse CNN is frequently used as the backbone network for point cloud feature extraction because of its efficiency and accuracy and have the ability to generate high-quality 3D propose boxes. Use the 3D CNN to divide the points into $L \times W \times H$ voxels with the same intervals. Then calculate the points in the voxel as the average value of 3D coordinates and reflection intensity $x, y, z, r$ of all internal points or use other calculations methods to obtain unique eigenvalues for the voxel directly. We adopt the mean value of each voxel to encode the features in this paper. Different
convolution kernels and step sizes for CNNs are used to \{1x, 2x, 4x, 8x\} downsample. The position information of the keypoints obtained by sampling is mapped to the voxel feature space of the corresponding position to ensure that the keypoints have and only one corresponding voxel. Treat each voxel as a point, the set abstraction proposed by PointNet++ [30] is adopted for the aggregation of the voxel-wise feature.

The voxel-wise feature vectors formulated reference PV-RCNN [1] as

\[
S_i^{(l_k)} = \left\{ \begin{array}{l}
\left[ f_i^{(l_k)}; v_i^{(l_k)} - P_l \right]^T \\
\forall v_j^{(l_k)} \in V^{(l_k)} \\
\forall f_j^{(l_k)} \in F^{(l_k)} \\
P_l \in dp
\end{array} \right| \left\| v_j^{(l_k)} - P_l \right\|^2 < r_k
\]

where \(S_i^{(l_k)}\) indicate the \(k^{th}\) layer feature of the number \(i\) distance-measured keypoint in voxel space. \(F^{(l_k)} = \{f_1^{(l_k)}, f_2^{(l_k)}, \ldots, f_{F_k}^{(l_k)}\}\) denote as the set of voxel-wise feature vectors in the \(k^{th}\) layer of 3D voxel CNN, \(V^{(l_k)} = \{v_1^{(l_k)}, v_2^{(l_k)}, \ldots, v_{V_k}^{(l_k)}\}\) denote as the d-keypoint sequence 3D coordinates calculated by the voxel indices and actual voxel sizes of the \(k^{th}\) layer. \(r_k\) representative non-empty voxels at the \(k^{th}\) layer within a fixed radius. PointNet [29] unit will be adopted as

\[
df_i^{(pv_k)} = \max \left\{ G \left( M \left( S_i^{(l_k)} \right) \right) \right\}
\]

where \(df_i^{(pv_k)}\) indicate the \(k^{th}\) layer feature of the number \(i\) distance-measured keypoint after set abstraction.

### 3.2 Feature FPS Voxel Set Abstraction Module

Determine the position distribution of the points through the points obtained by D-FPS, which usually contains a large number of background points and invalid points. This is explained by the fact that D-FPS only considers 3D location characteristics. The sampled points are scattered in the scene, which is beneficial to object regression but not conducive to the object classification, because the sampled point is a tiny subset sequence of the raw point cloud. A few points in this subset sequence fall in the ground-truth, which greatly weakens the points that can be referred to for object classification. To deal with the dilemma, we extend the “distance FPS voxel set abstraction module” to a more general feature extraction module. Specifically, the distance between points is used as a 3D feature sequence, and we extend the 3D distance feature to higher-dimensional features space \(N\) to obtain semantic information. Thinking of distance sampling as a special variant of feature sampling. When \(N\) is equal to 4 and the feature contains \(x, y, z, r\), “feature FPS voxel set abstraction module” turns to “distance FPS voxel set abstraction module.”

The spatial feature measurement is formulated as

\[
F - Distance = \|X - Y\|_2 = \sqrt{\sum_{j} X_j^2 + Y_j^2 - 2X_j Y_j} \left( X_i, Y_j \in dp \ \ \ \ \ \ \ \ \ | \mathbb{R} \in N \right)
\]

F-Distance represents L2 feature distance between two distance-measured keypoints. \(X\) and \(Y\) represent \(N\) dimensional features from different distance-measured keypoints. Otherwise, \(X\) and \(Y\) belong to \(dp = \{p_1, p_2, p_3 \ldots p_N\}\). After feature FPS (F-FPS), we could get the f-keypoint sequence \(fp = \{p_1, p_2, p_3 \ldots p_N\}\) and satisfies the constraint \(fp \subseteq dp\). In this way, we can receive different semantic information under the coverage of the scene by sampling points after D-FPS. A smaller subset
sequence of keypoints is picked up to represent the distribution of the semantic information of the point cloud. The same as the distance FPS voxel set abstract module, the keypoints at this time are replaced with a smaller subset sequence of feature keypoints.

The voxel-wise feature vectors formulated as

\[
B_i^{(l_k)} = \left\{ \begin{array}{l}
\forall \| v_j^{(l_k)} - f_{p_i} \|^2 < r_k \\
\forall f_j^{(l_k)} \in F^{(l_k)} \\
\end{array} \right. \right\}
\]

(5)

where \(B_i^{(l_k)}\) indicate the \(k^{th}\) layer feature of the number \(i\) feature-measured keypoint in voxel space. \(F^{(l_k)} = \{f_1^{(l_k)}, f_2^{(l_k)} \ldots f_n^{(l_k)}\}\) denote as the set of voxel-wise feature vectors in the \(k^{th}\) layer of 3D voxel CNN, \(V^{(l_k)} = \{v_1^{(l_k)}, v_2^{(l_k)} \ldots v_n^{(l_k)}\}\) denote as the f-keypoint sequence 3D coordinates calculated by the voxel indices and actual voxel sizes of the \(k^{th}\) layer. \(r_k\) representative non-empty voxels at the \(k^{th}\) layer within a fixed radius. PointNet [29] unit will be used as

\[
ff_i^{(pvk)} = \max \left\{ G \left( M \left( B_i^{(l_k)} \right) \right) \right\}
\]

(6)

where \(ff_i^{(pvk)}\) indicate the \(k^{th}\) layer feature of the number \(i\) feature-measured keypoint after set abstraction.

### 3.3 BEV Projection and Proposal Generation

After the 3D sparse CNN implemented by several SA layers intertwined with key point sampling. The 3D voxel features after 8x downsampling are projected to the 2D BEV perspective through the Z-axis. Bilinear interpolation is utilized to obtain the dense features from the bird-view feature. Anchor-based approach [13] is used for high-quality 3D proposal generation.

### 3.4 Multi-Scale Keypoint Feature Fusion

The point cloud is divided into voxels to calculate the features of a certain range of points. The semantic information at different scales can be better captured by adopting sparse convolution. After feature FPS voxel set abstraction module, we can obtain the f-keypoints sequence at different layers defined as follows:

\[
fP = \begin{cases} 
fp_{conv1} = \{p_1, p_2, p_3 \ldots p_t\} \\
fp_{conv2} = \{p_1, p_2, p_3 \ldots p_t\} \\
fp_{conv3} = \{p_1, p_2, p_3 \ldots p_t\} \\
fp_{conv4} = \{p_1, p_2, p_3 \ldots p_t\} \\
fp_{bev} = \{p_1, p_2, p_3 \ldots p_t\}
\end{cases}
\]

(7)

\[
fp_{conv1} \neq fp_{conv2} \neq fp_{conv3} \neq fp_{conv4} \neq fp_{bev}
\]

(8)

More specifically, denote \(fp_{conv1}, fp_{conv2}, fp_{conv3}, fp_{conv4}, fp_{bev}\) as the set of the f-keypoint sequence after conv1; conv2; conv3; conv4 and BEV feature abstract. The constraint (8) is satisfied due to using F-FPS at different feature spaces. \(ff_{fp_i}^{(l)}\) indicate the \(l^{th}\) layer f-keypoint sequence feature.
Based on the difference between sampling points, how to effectively use those keypoints at different layers has become a problem that must be considered:

(i) Whether the f-keypoint feature of each layer is effective for the point cloud task;

To observe this phenomenon more clearly, the quotient is proposed as \( \frac{\text{fpingt}}{\text{fp}} \). \( \text{fpingt} \) denote the number of the total f-keypoint sequence comes from ground-truth. \( \text{fp} \) indicate the number of the f-keypoint sequences equals to \( q \). Calculating the distribution of the points obtained by F-FPS for each layer in the ground-truth is illustrated in Fig. 3. The abscissa indicates the sampling points using different convolutional layers. It should be noted that “raw” represents the original point that falls in the ground-truth after using laser scanners. “fusionfeature” represents the feature that is downsampled to 128 dimensional after concatenating the features of all layers.

(ii) How to pick up different layers of the f-keypoint sequence;

Fig. 3. Different layers distribution of f-keypoint sequence in ground-truth (from left to right and top to bottom). The percentage and number of the f-keypoint falling in the ground-truth box of different objects. All the distance sampling points are 2048.
For different layers of sampling points, using F-FPS after 3D sparse convolution generally obeys the same principle. As illustrated in Fig. 3, under different categories, the number and percentage of the f-keypoint sequence sampled at BEV, conv1, and conv4 layers falling in the ground-truth are greater than other layers. On the contrary, the number of the f-keypoint sequence sampled after “fusionfeature” falling in the ground-truth is the lowest among conv1, conv2, conv3, conv4, and Bev layers. Based on this phenomenon, it is an unwise choice to concatenate each layer’s features to object classification. Does it mean that the points in these layers contribute more to classification and regression? To solve this problem, we propose four different point selection strategies and test them on the validation set which is given in the next consideration.

(iii) How to merge the characteristics of different layers of the f-keypoint sequence;

Albeit an intuitive solution is to directly combine the features of each layer to provide support for the subsequent object classification and regression as shown in formula (9). However, this solution is not consistent with our original intention of using the minimum number of representative points to represent the semantic information. Based on the above reasons, we consider using the sampling points on BEV, conv1, and conv4 layers. The four fusion strategies are defined as follows:

The spatial f-keypoint fusion strategy is formulated as

\[ fp = fp_{\text{conv1}} \cup fp_{\text{conv2}} \cup fp_{\text{conv3}} \cup fp_{\text{conv4}} \cup fp_{\text{bev}} \]  \hspace{1cm} (9)

The second choice of the f-keypoint fusion strategy is

\[ fp = fp_{\text{conv1}} \cap fp_{\text{conv4}} \cap fp_{\text{bev}} \]  \hspace{1cm} (10)

The third choice of the f-keypoint fusion strategy is

\[ fp = fp_{\text{conv1}} \cup fp_{\text{conv4}} \cup fp_{\text{bev}} \]  \hspace{1cm} (11)

Finally, a fourth option that we adopt in this paper is

\[ fp = (fp_{\text{conv1}} \cup fp_{\text{conv4}}) \cap fp_{\text{bev}} \]  \hspace{1cm} (12)

The corresponding feature fusion strategy is defined as

\[ ff = (ff^e_{\text{conv1}} \cup ff^e_{\text{conv4}}) \cap ff^e_{\text{bev}} \]  \hspace{1cm} (13)

3.5 Refine Prediction Head

With multi-scale keypoints feature fusion, the whole scene is summarized into two different keypoints sequences with multiscale semantic features. ROI feature abstraction [1] is adopted between two different keypoints sequences for proposal refinement. The refine prediction head adopts MLP from ff and df for confidence prediction and box refinement, respectively.

3.6 Experiments

3.6.1 Datasets

KITTI dataset, released in 2012, is the widely used computer vision algorithm evaluation dataset in the field of autonomous driving. This data set contains multiple tasks such as 3D object detection and multi-object tracking and segmentation. The 3D object detection benchmark consists of 7,481 training images and 7,518 test images as well as the corresponding point clouds. The training samples are widely divided into the train set (3,712 samples) and the validation set (3,769 samples).
3.6.2 Implementation Details

We conduct the experiment on a GTX 1080Ti GPU. The 3D sparse voxel CNN has four layers with feature dimensions 16, 32, 64, 64, respectively. The end-to-end network train with a learning rate 0.01 for 80 epochs with batch size 1. For the survival of sampling points at different layers in the ground-truth box, we amplify the absolute coordinate position of the ground-truth box in the scene based on the principle: $2 \times L, 2 \times W,$ and $2 \times H$. The reason is that we should not only focus on the points that fall in the ground-truth box, but we should also pay attention to the points around the ground-truth box to fine-tune the bounding box. For the refine prediction head stage, we use 2-layer MLP for confidence prediction and box regression respectively. Furthermore, we set the number of sampling points as follows $p = 2,048$ and $q = 960$ for the KITTI dataset. By using such numbers, the scene can be better covered with the fewest points and better detection results can be obtained on multiple objects such as cars, pedestrians, and cyclists.

3.6.3 Results

To evaluate the performance, we submitted the experimental results on the KITTI 3D Object Detection Benchmark and the KITTI Bird’s eye view (BEV) Object Detection Benchmark. We conducted 3D object detection experiments on cars, pedestrians, and cyclists as indicated in Table 1. Besides, we conducted the BEV object detection experiments on cars and cyclists as illustrated in Table 2. Before submitting the results to the official website, we tested the performance of the most important car class on the KITTI validation split with mean average precision (mAP) with R40 as illustrated in Table 3. More table attribute details are as follows, “Modality” means data sources, “R” means using pictures, “L” means using point clouds. According to the format “Ours-number-method” which means to use the specified number points from F-FPS sampling to evaluate with specified fusion strategy. “U” is the union of mathematics, “\“ is a mathematical intersection. The mean average precision with 40 recall positions (mAP) is used to evaluate model performance on three difficulty levels: easy, moderate, and hard. To evaluate the overlap of the boxes, we use the same indicators as the official evaluation. Specifically, for cars, the overlap of the bounding box on easy, moderate, and hard objects requires 70%, 50%, and 50% respectively. For pedestrians and cyclists, the overlap of the bounding box on easy, moderate, and hard objects requires 50%, 25%, and 25% respectively.

### Table 1. mAP about 3D object detection performance comparison on the test set

<table>
<thead>
<tr>
<th>Method</th>
<th>Modality</th>
<th>3DCar (%)</th>
<th>3DPedestrian (%)</th>
<th>3DCyclist (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Easy</td>
<td>Moderate</td>
<td>Hard</td>
</tr>
<tr>
<td>MV3D [6]</td>
<td>R+L</td>
<td>74.97</td>
<td>63.63</td>
<td>54.00</td>
</tr>
<tr>
<td>F-PointNet [7]</td>
<td>R+L</td>
<td>82.19</td>
<td>69.79</td>
<td>60.59</td>
</tr>
<tr>
<td>AVOD-FPN [9]</td>
<td>R+L</td>
<td>83.07</td>
<td>71.76</td>
<td>65.73</td>
</tr>
<tr>
<td>UberATG-MMF [11]</td>
<td>R+L</td>
<td>88.40</td>
<td>77.43</td>
<td>70.22</td>
</tr>
<tr>
<td>PFF3D [28]</td>
<td>R+L</td>
<td>81.11</td>
<td>72.93</td>
<td>67.24</td>
</tr>
<tr>
<td>PointPillars [15]</td>
<td>L</td>
<td>82.58</td>
<td>74.31</td>
<td>68.99</td>
</tr>
<tr>
<td>PointRCNN [18]</td>
<td>L</td>
<td>86.96</td>
<td>75.64</td>
<td>70.70</td>
</tr>
<tr>
<td>Point-GNN [22]</td>
<td>L</td>
<td>88.33</td>
<td>79.47</td>
<td>72.29</td>
</tr>
<tr>
<td>S-AT GCN [34]</td>
<td>L</td>
<td>83.20</td>
<td>76.04</td>
<td>71.17</td>
</tr>
<tr>
<td>Ours-896-bevUConv1UConv4</td>
<td>L</td>
<td>86.86</td>
<td>78.30</td>
<td>73.80</td>
</tr>
<tr>
<td>Ours-960-bevUConv1UConv4</td>
<td>L</td>
<td>87.25</td>
<td>78.30</td>
<td>73.66</td>
</tr>
</tbody>
</table>

The bold number indicates the best result.
We compare the model with other baselines. Tables 1 and 2 show the test set performance of the proposed model and the other baseline models. Table 3 exhibits the validation set performance of the proposed model and the other baseline models. We can see that our model has a better performance in the moderate and hard difficulties. The details are as follows.

As it is shown in Table 1, the proposed method achieved 1.9%–2.09% improvement by using strategy in the maximum occlusion level, and maximum truncation. The hard difficulty which uses the union of keypoints set has a 2.09% improvement. As it is shown in Table 2, the BEV object detection performance are compared in car and cyclist category. For the car category, our method accuracy has 0.53%–0.74% improved by using strategy in the hard difficulty. For the cyclist category, our method improves 0.1%–0.49% in moderate and hard difficulties. Table 3 exhibits 3D object detection performance comparison on the validation set. Our model achieves better performance compared with the state-of-the-art methods in easy/moderate/hard difficulties of car category, which indicates it is a potential solution to moderate and hard difficulties. It is frustrating that pedestrians are difficult to correctly detect. The reason could be that the number of sampling points is too little and the pedestrian bounding box is small. In the scene, pedestrian points often coincide with other semantics.

### Table 2. mAP about BEV object detection performance comparison on the test set

<table>
<thead>
<tr>
<th>Method</th>
<th>Modality</th>
<th>BEVCar (%)</th>
<th>BEVCyclist (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Easy</td>
<td>Moderate</td>
</tr>
<tr>
<td>MV3D [6]</td>
<td>R+L</td>
<td>86.02</td>
<td>76.90</td>
</tr>
<tr>
<td>F-PointNet [7]</td>
<td>R+L</td>
<td>88.70</td>
<td>84.00</td>
</tr>
<tr>
<td>AVOD-FPN [9]</td>
<td>R+L</td>
<td>88.53</td>
<td>83.79</td>
</tr>
<tr>
<td>UberATG-MMF [11]</td>
<td>R+L</td>
<td>89.49</td>
<td>87.47</td>
</tr>
<tr>
<td>PFF3D [28]</td>
<td>R+L</td>
<td>89.61</td>
<td>85.08</td>
</tr>
<tr>
<td>PointPillars [15]</td>
<td>L</td>
<td>88.35</td>
<td>86.10</td>
</tr>
<tr>
<td>PointRCNN [18]</td>
<td>L</td>
<td>89.47</td>
<td>85.68</td>
</tr>
<tr>
<td>Point-GNN [22]</td>
<td>L</td>
<td>93.11</td>
<td>89.17</td>
</tr>
<tr>
<td>S-AT GCN [34]</td>
<td>L</td>
<td>90.85</td>
<td>87.68</td>
</tr>
<tr>
<td>X-view [26]</td>
<td>L</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Ours-896-bevUConv1UConv4</td>
<td>L</td>
<td>91.62</td>
<td>87.41</td>
</tr>
<tr>
<td>Ours-960-bevUConv1^Conv4</td>
<td>L</td>
<td>91.93</td>
<td>87.41</td>
</tr>
</tbody>
</table>

The bold number indicates the best result.

### Table 3. mAP about 3D object detection performance comparison on the validation set

<table>
<thead>
<tr>
<th>Method</th>
<th>Modality</th>
<th>Label</th>
<th>3DCar (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Easy</td>
<td>Moderate</td>
</tr>
<tr>
<td>MV3D [6]</td>
<td>R+L</td>
<td>two-stage</td>
<td>71.29</td>
</tr>
<tr>
<td>F-PointNet [7]</td>
<td>R+L</td>
<td>two-stage</td>
<td>83.76</td>
</tr>
<tr>
<td>AVOD-FPN [9]</td>
<td>R+L</td>
<td>two-stage</td>
<td>84.41</td>
</tr>
<tr>
<td>PFF3D [28]</td>
<td>R+L</td>
<td>one-stage</td>
<td>88.04</td>
</tr>
<tr>
<td>Voxelnet [12]</td>
<td>L</td>
<td>two-stage</td>
<td>81.97</td>
</tr>
<tr>
<td>PointRCNN [18]</td>
<td>L</td>
<td>two-stage</td>
<td>88.88</td>
</tr>
<tr>
<td>Point-GNN [22]</td>
<td>L</td>
<td>one-stage</td>
<td>87.89</td>
</tr>
<tr>
<td>Ours</td>
<td>L</td>
<td>two-stage</td>
<td><strong>89.10</strong></td>
</tr>
</tbody>
</table>

The bold number indicates the best result.
3.6.4 Ablation Studies

Two sets of ablation experiments were performed to determine the number of points selected on F-FPS and fusion strategy. The training samples are widely divided into the train set (3,712 samples) and the validation set (3,769 samples). Firstly, we test different feature fusion strategies for the feature furthest point \( fp \) as illustrated in Table 4. Then the influence of the different sample numbers of feature furthest points \( fp \) is tested as illustrated in Table 5.

**Table 4.** mAP about 3D object detection performance comparison on the validation set for different fusion strategies

<table>
<thead>
<tr>
<th>Fusion strategy</th>
<th>3DCar (%)</th>
<th>3DPedestrian (%)</th>
<th>3DCyclist (%)</th>
<th>Sample number</th>
</tr>
</thead>
<tbody>
<tr>
<td>PVRCNN [1] (without F-FPS)</td>
<td>88.96</td>
<td>78.74</td>
<td>78.11</td>
<td>60.83</td>
</tr>
<tr>
<td>bev(^{\text{conv1}})(^{\text{conv4}})</td>
<td>88.77</td>
<td>78.87</td>
<td>78.29</td>
<td>60.33</td>
</tr>
<tr>
<td>bev(^{\text{Uconv1}})(^{\text{Uconv4}})</td>
<td>89.02</td>
<td>78.82</td>
<td>78.30</td>
<td>64.71</td>
</tr>
<tr>
<td>Improvement</td>
<td>0.06</td>
<td>0.07</td>
<td>0.19</td>
<td>3.89</td>
</tr>
</tbody>
</table>

The bold number indicates the best result.

As it is shown in Table 4, to compare the effect of the feature FPS voxel set abstraction module, we first test the methods without F-FPS and compare them on various feature fusion methods. It is worth noting that when using all the calculated intersections of different layers features, the final points can survive around 300 points. The number of sampling points is too small, thus impossible to cover the scene well. For the above reasons, this fusion method is directly abandoned. Compared with the methods without F-FPS (PV-RCNN), the proposed model improve 2.13% between different difficulties.

The most intuitive way to select the number of feature keypoints \( fp \) is half of the distance feature points \( dp \). We test different number of selected points as it is shown in Table 5, the number of F-FPS points set as 1,024, 960, and 896. Comprehensive consideration of the two indicators of difficulties and accuracy, we choose 960 sampling points for F-FPS.

**Table 5.** mAP about 3D object detection performance comparison on the validation set for different numbers

<table>
<thead>
<tr>
<th>Sample number</th>
<th>3DCar (%)</th>
<th>3DPedestrian (%)</th>
<th>3DCyclist (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ours-1024</td>
<td>72.62</td>
<td>64.06</td>
<td>62.23</td>
</tr>
<tr>
<td>Ours-896</td>
<td>88.77</td>
<td>78.87</td>
<td>78.29</td>
</tr>
<tr>
<td>Ours-960</td>
<td>88.96</td>
<td>78.89</td>
<td>78.28</td>
</tr>
<tr>
<td>Ours-960-(^{\text{bev}})^{\text{Uconv1}}(^{\text{Uconv4}})</td>
<td>89.10</td>
<td>79.13</td>
<td>78.46</td>
</tr>
</tbody>
</table>

The bold number indicates the best result.

As it is shown in Table 4, to compare the effect of the feature FPS voxel set abstraction module, we first test the methods without F-FPS and compare them on various feature fusion methods. It is worth noting that when using all the calculated intersections of different layers features, the final points can survive around 300 points. The number of sampling points is too small, thus impossible to cover the scene well. For the above reasons, this fusion method is directly abandoned. Compared with the methods without F-FPS (PV-RCNN), the proposed model improve 2.13% between different difficulties.

The most intuitive way to select the number of feature keypoints \( fp \) is half of the distance feature points \( dp \). We test different number of selected points as it is shown in Table 5, the number of F-FPS points set as 1,024, 960, and 896. Comprehensive consideration of the two indicators of difficulties and accuracy, we choose 960 sampling points for F-FPS.

### 4. Conclusion

In this paper, we present a multi-scale keypoints feature fusion object detection model for the point clouds. We introduce a feature FPS set abstraction module to aggregate 3D voxel-wise features above surrounding voxel volume to preserve minimum representative points. After that, a novel multi-scale
feature fusion strategy is adopted to get more accurate semantic information. All the above designs enable our model to achieve the leading accuracy in both the 3D object detection and BEV object detection of the KITTI benchmark. For the car category, our method in the hard difficulty (maximum occlusion and truncation) has 1.9%–2.09% improved. For the cyclist category, our method improved 0.1%–0.49% in the moderate and hard difficulties for the BEV object detection. Combine feature FPS set abstraction module and fusion strategy outperform the previous works by at least 2.13%. In the future, it is hopeful to extend the model for better prediction tasks based on the feature fusion structure design.

Acknowledgements
We thank lab members at Chongqing University of Posts and Telecommunications for providing the GPU and technical support. We thank lab members at Inha University for their technical support. We would also like to thank the editors, reviewers, and editorial staff who participated in the publication process of this paper.

Author’s Contributions
Conceptualization, XZ, LB. Funding acquisition, XZ. Investigation and methodology, XZ, LB. Writing of the original draft, LB. Writing of the review and editing, XZ, ZZ, YL. All the authors have proofread the final version.

Funding
This work was supported in part by the Key Cooperation Project of Chongqing Municipal Education Commission (No. HZ2021008), in part by the Natural Science Foundation of Chongqing, China (No. cstc2019jscx-mbdxX00021 and cstc2014jrc-qnrc40002), and in part by the Major Industrial Technology Research and Development Project of Chongqing High-tech Industry (No. D2018-82).

Competing Interests
The authors declare that they have no competing interests.

References


