PC-RGNN: Point Cloud Completion and Graph Neural Network for 3D Object Detection

Yanan Zhang[1, 2, 3], **Di Huang**[1, 2, 3*], **Yunhong Wang**[1, 3]

[1]Beijing Advanced Innovation Center for Big Data and Brain Computing, Beihang University
[2]State Key Laboratory of Software Development Environment, Beihang University
[3]School of Computer Science and Engineering, Beihang University, Beijing 100191, China
{zhangyanan, dhuang, yhwang}@buaa.edu.cn

Abstract

LiDAR-based 3D object detection is an important task for autonomous driving and current approaches suffer from sparse and partial point clouds of distant and occluded objects. In this paper, we propose a novel two-stage approach, namely PC-RGNN, dealing with such challenges by two specific solutions. On the one hand, we introduce a point cloud completion module to recover high-quality proposals of dense points and entire views with original structures preserved. On the other hand, a graph neural network module is designed, which comprehensively captures relations among points through a local-global attention mechanism as well as multi-scale graph based context aggregation, substantially strengthening encoded features. Extensive experiments on the KITTI benchmark show that the proposed approach outperforms the previous state-of-the-art baselines by remarkable margins, highlighting its effectiveness.

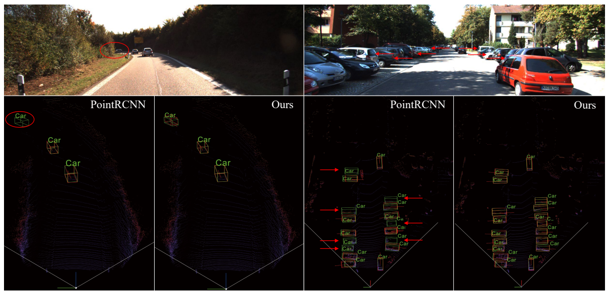

Figure 1: Illustration of the two major challenges in LiDAR-based 3D object detection (best viewed with zoom-in). The left case shows a sparse point cloud for a car far away, while the right case shows the extremely incomplete point clouds for occluded cars. The red and green boxes indicate the predicted results and ground-truths respectively.

Introduction

3D object detection in point clouds is eagerly in demand in autonomous driving, and LiDAR laser scanners are the most common instruments to collect such data. Compared to 2D images, LiDAR point clouds convey real 3D geometric structures and spatial locations of objects and are less sensitive to illumination variations, which enables more reliable detection results.

In recent years, several approaches have been proposed for 3D object detection and they follow either the one-stage framework or the two-stage one as in the 2D domain, where how to learn effective shape features is an essential issue. For instance, MV3D (Chen et al. 2017) and AVOD (Ku et al. 2018) transform point clouds to bird's eye view or front view as initial representation and apply 2D convolutions for feature map computation; VoxelNet (Zhou and Tuzel 2018) and SECOND (Yan, Mao, and Li 2018) voxelize the 3D space into regular cells and employ 3D convolutions to extract features; F-Pointnet (Qi et al. 2018) and PointRCNN (Shi, Wang, and Li 2019) take raw point clouds as input and encode features by PointNets (Qi et al. 2017a,b).

Those approaches indeed show great potentials and have consistently improved the performance of major benchmarks. Unfortunately, they tend to fail in the presence of low-quality input point clouds, *i.e.* sparse and partial data due to distant and occluded objects, which often occur in the real world. As illustrated in Fig. 1, PointRCNN, the state of the art representative, misses many objects marked with the red ellipse and arrows for long-distance and severe occlusions. Such a limitation derives from two main aspects: (1) point cloud representation in current 3D object detectors does not work well on large variations in sampling density and sampling integrity and (2) the PointNet-like networks adopted as the backbones in the leading 3D object detectors are not so powerful which make insufficient use of given point clouds.

Motivated by the analysis above, this paper proposes a novel two-stage approach, named PC-RGNN, to 3D object detection from LiDAR based point clouds. Specifically, the 3D proposal generation (3D PG) module first suggests bounding box candidates in a bottom-up manner via segmenting the whole scene into foreground and background. Since objects are usually sparsely and partially sampled, a point cloud completion (PC) module is then introduced to recover high-quality proposals with dense points and entire views. Further, we model each refined proposal as a graph and design a graph neural network module, called AMS-GNN, to capture its shape characteristics. AMS-GNN ag-

[*]Corresponding author: Di Huang.

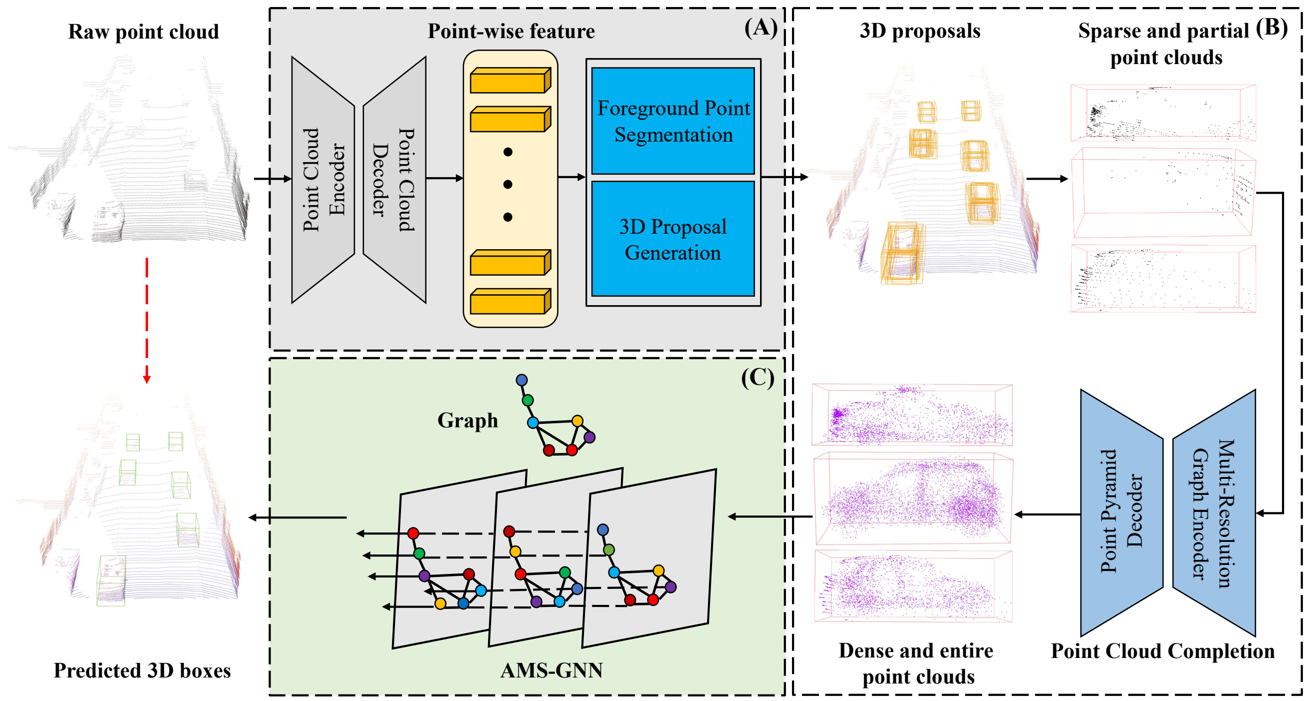

Figure 2: Framework overview. The whole PC-RGNN network consists of three main modules: (A) 3D proposal generation, (B) point cloud completion, and (C) attention based multi-scale GNN representation.

gregates contextual clues by combining multi-scale graphs and learns different weights of neighboring nodes through a local-global attention mechanism, and geometric relations among points can thus be comprehensively exploited, leading to enhanced features for decision making. Thanks to these modules, PC-RGNN reaches very competitive scores on the KITTI database, and in particular, it facilitates the detection of 3D objects in very difficult scenes, as depicted in Fig. 1.

In summary, the main contributions of this paper are:

- We highlight the challenge of low-quality input point clouds in LiDAR-based 3D object detection and propose a novel two-stage detection approach (PC-RGNN), which significantly boosts the performance.
- We present a new point cloud completion (PC) module to improve proposals of sparse and partial points. To the best of our knowledge, this is the first study that considers point cloud refinement in 3D object detection.
- We design a new graph neural network (AMS-GNN) module, which strengths the structure features by encoding geometric relations among points through attention based multi-scale graph aggregation.

Related Work

This section briefly reviews the major approaches to 3D object detection as well as the ones of point cloud completion.

Grid-based detectors. A number of methods initially convert point clouds to regular grids by projecting them to the planes of specific views (Chen et al. 2017; Engelcke et al. 2017; Ku et al. 2018; Liang et al. 2018, 2019) or subdividing them to equally distributed voxels (Wang and Posner 2015; Zhou and Tuzel 2018; Yan, Mao, and Li 2018; Lang et al. 2019) so that they can be processed by 2D or 3D CNNs to compute detection features. Although grid-based methods are generally straightforward and efficient, they inevitably incur much information loss and thus limit the performance because of the coarse quantization process.

Point-based detectors. Many methods directly take the raw unordered and irregular points as input and apply point cloud deep learning networks, such as PointNet (Qi et al. 2017a) and PonintNet++ (Qi et al. 2017b), to encode structure features for detection (Qi et al. 2018; Chen et al. 2019; Qi et al. 2019; Shi, Wang, and Li 2019; Yang et al. 2019; Shi et al. 2020; Yang et al. 2020). These methods outperform grid-based ones. However, without explicit modeling of point relations, they are not so competent at discriminative geometry representation.

Graph-based detectors. Recently, inspired by the success of graph convolutions in point cloud segmentation and classification tasks, Point-GNN (Shi and Rajkumar 2020) investigates the graph neural network to extract shape features for 3D object detection, which proves a promising way. Nevertheless, in their model, each node is regarded to equally contribute in local and global representation and the single-scale graph does not make full use of the contextual information. Both the facts leave space for improvement.

Point cloud completion methods. Point cloud completion aims to estimate entire 3D shapes from partial point cloud inputs. L-GAN (Achlioptas et al. 2018) introduces the first deep learning model with an Encoder-Decoder architecture. PCN (Yuan et al. 2018) presents a coarse-to-fine procedure to synthesize dense and complete data by a specially designed decoder. RL-GAN-Net (Sarmad, Lee, and Kim 2019) proposes a reinforcement learning agent controlled GAN to speed up the inference phase. PF-Net (Huang et al. 2020) hierarchically recovers the point cloud by a feature-point based multi-scale generation network. Different from the research aforementioned, for the first time, point cloud completion is attempted to ameliorate LiDAR-based 3D object detection, as objects are often very far away or seriously occluded, leading to sparse and partial sampling.

PC-RGNN

In this section, we describe the proposed PC-RGNN in detail, including the entire framework as well as the modules of 3D proposal generation, point cloud completion, and attention based multi-scale graph feature aggregation.

Framework

The framework overview of our proposed PC-RGNN is illustrated in Fig. 2. The whole network consists of three main modules: (A) 3D proposal generation, (B) point cloud completion, and (C) attention based multi-scale graph neural network representation. Given a raw point cloud, the proposal generation module segments the foreground from background and generates 3D bounding box candidates simultaneously. The point cloud completion module then recovers dense and entire 3D shapes from sparse and/or partial proposal point clouds. Finally, the GNN module comprehensively encodes the structure characteristics to predict detection results.

3D Proposal Generation

As described in (Shi, Wang, and Li 2019), objects in 3D scenes are naturally separated, and the segmentation masks can be directly acquired from their 3D bounding box annotations. Therefore, we follow PointRCNN and build a sub-network to learn point-wise features to simultaneously locate the foreground areas and generate 3D proposals. Based on this bottom-up mechanism, we avoid using a large number of predefined 3D anchors and thus dramatically reduce the searching space in this phase.

Concretely, PointNet++ (Qi et al. 2017b) with multi-scale grouping is adopted as the backbone, and a segmentation head and a regression head are added to estimate the foreground mask and generate bounding box candidates respectively. For large outdoor scenes, the number of background points is usually much larger than that of foreground, and we therefore use the focal loss (Lin et al. 2017b) in segmentation to deal with the class imbalance problem as:

$$L_{\text{seg}}(p_t) = -\alpha_t(1-p_t)^{\gamma}\log(p_t),$$
$$\text{where } p_t = \begin{cases} p & \text{for foreground points,} \\ 1-p & \text{otherwise.} \end{cases} \tag{1}$$

During training, we set $\alpha_t = 0.25$ and $\gamma = 2$. For proposal generation, box locations are only regressed from foreground points. Here, a 3D bounding box is described as (x, y, z, h, w, l, θ) in the LiDAR coordinate, where (x, y, z) is the object center, (h, w, l) is the object size, and θ is the object orientation from the bird's eye view. For (z, h, w, l), the smooth L1 loss is utilized and for (x, y, θ), the bin-based loss (Shi et al. 2020) is exploited.

Point Cloud Completion

To address the challenges of sparse and partial data caused by distant and occluded objects, we propose a point cloud completion (PC) module for 3D detection. Unlike the existing 3D shape generation or reconstruction methods that output totally new point clouds, the proposed module only renders additional points as supplement with input data unchanged, aiming to preserve original spatial arrangement. As shown in Fig. 3, the overall architecture of the PC module is composed of three fundamental building blocks, *i.e.* Multi-Resolution Graph Encoder (MRGE), Point Pyramid Decoder (PPD), and Discriminator Network.

The input to MRGE is an $N \times 3$ unordered point cloud. It is first down sampled to obtain two more views of smaller resolutions by farthest point sampling (FPS). Three independent GCN layers (Wang et al. 2019) are then used to map those resolutions into individual latent vectors F_i. Compared with the PointNet-like models, GCN captures extra geometry clues from connection relations of points, which is expected to facilitate low-quality point cloud refinement. All the F_i are further concatenated to form a stronger feature map M in the size of 1920×3 and the feature maps are integrated into a final vector V. Inspired by Feature Pyramid Networks (Lin et al. 2017a), PPD conducts in a coarse to fine fashion. Three feature layers FC$_1$, FC$_2$, FC$_3$ (size: 1024, 512, 256) are computed by passing V through fully-connected layers. Each feature layer is responsible for recovering point clouds in a specific resolution. The coarse center points Y_{coarse} are predicted from FC$_3$, which are of the size of $M_1 \times 3$. The relative coordinates of middle center points Y_{middle} are predicted from FC$_2$. Each point in Y_{coarse} serves as the center to generate M_2/M_1 points of Y_{coarse}. Thus, the size of Y_{middle} is $M_2 \times 3$. Fine points Y_{fine} are finally predicted by PPD, with the size of $M \times 3$.

The loss function is made up of two parts: the multi-level completion loss and the adversarial loss. The Chamfer Distance (CD) is chosen as the completion loss:

$$\begin{aligned} d_{CD}(S_1, S_2) = \tfrac{1}{S_1}\sum_{x\in S_1}\min_{y\in S_2}\|x-y\|_2^2 \\ + \tfrac{1}{S_2}\sum_{y\in S_2}\min_{x\in S_1}\|y-x\|_2^2 \end{aligned} \tag{2}$$

It measures the average nearest squared distance between the predicted point set S_1 and the ground truth S_2. Since PPD estimates three point clouds at different levels, the multi-level completion loss is calculated in (3), where d_{CD1}, d_{CD2}, and d_{CD3} are weighted by a hyperparameter α:

$$\begin{aligned} L_{com} = \alpha d_{CD1}(Y_{coarse}, Y'_{gt}) + 2\alpha d_{CD2}(Y_{middle}, Y''_{gt}) \\ + d_{CD3}(Y_{fine}, Y_{gt}) \end{aligned} \tag{3}$$

We define $F : X \rightarrow Y'$, which maps the low-quality in-

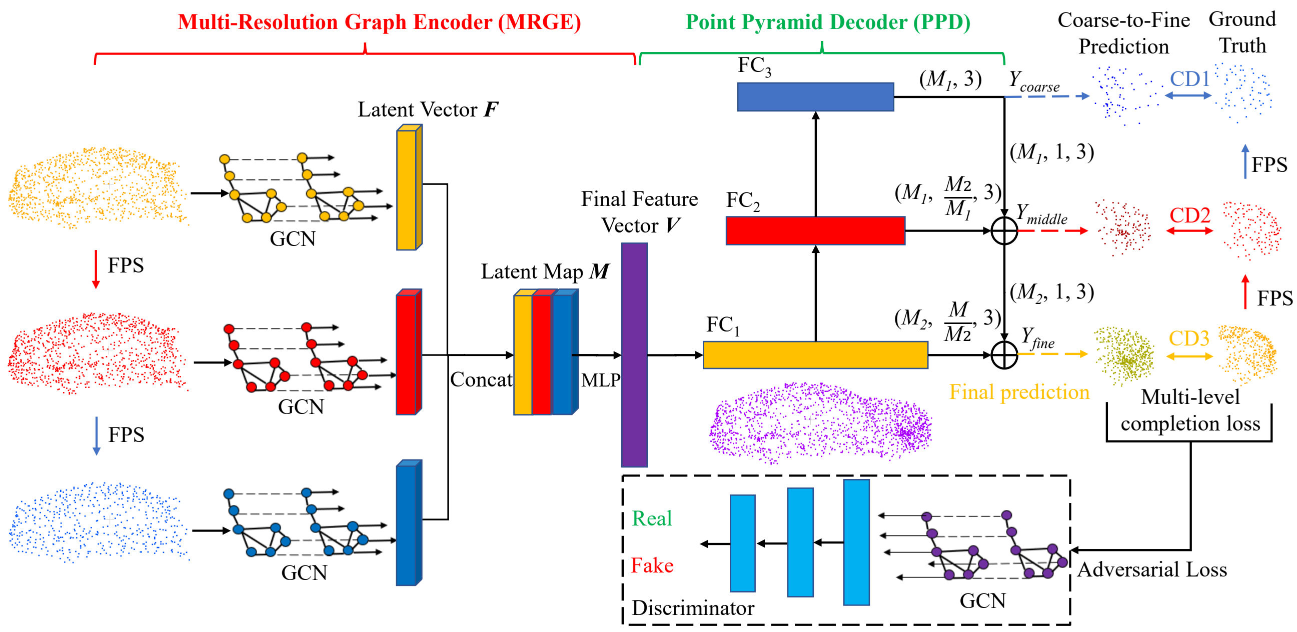

Figure 3: Architecture of the PC module. With input point clouds, it predicts additional parts of sparse and partial data by a Multi-Resolution Graph Encoder (MRGE) and a Point Pyramid Decoder (PPD). The Discriminator tries to distinguish the predicted regions from the real ones.

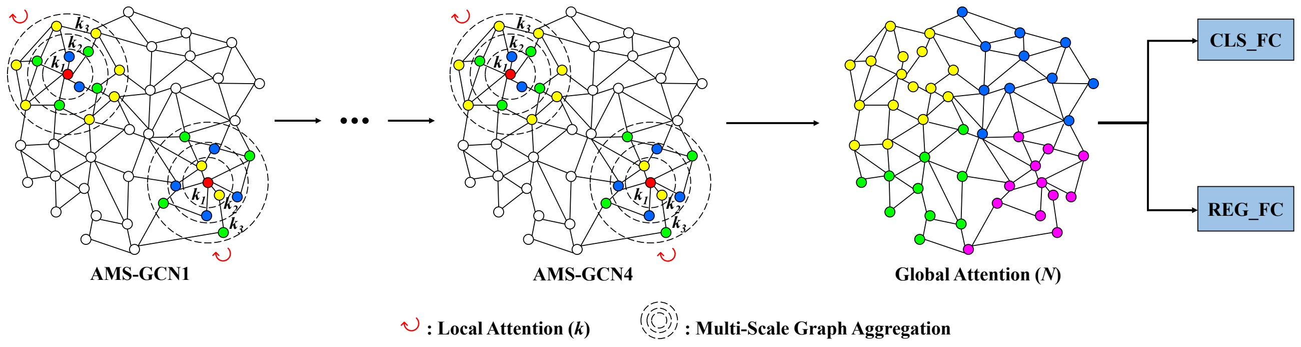

Figure 4: Architecture of the Attention based Multi-Scale Graph Neural Network.

put X into the predicted additional point set Y'. Discriminator D tries to distinguish Y' from the real point set Y. We first obtain a latent vector F after two GCN layers and F is then passed through fully-connected layers [256, 128, 16, 1] followed by a sigmoid-classifier to calculate the predicted value. In this case, the adversarial loss is defined as:

$$L_{adv} = \sum_{1 \le i \le S} \log(D(y_i)) + \sum_{1 \le i \le S} \log(1 - D(F(x_i))) \tag{4}$$

where $x_i \in X$, $y_i \in Y$, and S is the dataset size. The total loss is thus formulated as:

$$L_{total} = \lambda_{com} L_{com} + \lambda_{adv} L_{adv} \tag{5}$$

where λ_{com} and λ_{adv} are the weights of the completion loss and the adversarial loss.

Attention Based Multi-Scale GNN

To comprehensively encode shape characteristics of point clouds in proposals refined by the PC module, we design a novel graph neural network module. It strengths the features delivered by the previous GNN counterpart by multi-scale contextual clue extraction and attention based discriminative point highlighting, thus named AMS-GNN. As demonstrated in Fig. 4, it is composed of four attention based multi-scale graph convolution (AMS-GCN) layers and a global attention (GA) layer. Each AMS-GCN layer contains a multi-scale graph aggregation operation and a local attention (LA) operation.

Specifically, we encode each proposal point cloud in a graph by regarding the points as vertices and the connection between points as edges, which makes features flow between neighbors. We define a point cloud as a set $V = \{v_1, ..., v_i, ..., v_N\}$, where $v_i = (p_i, s_i)$ is a point with both the 3D coordi-

Method	Modality	3D Object Detection (%)			Bird's Eye View Detection (%)			Orientation Estimation (%)		
		Easy	Moderate	Hard	Easy	Moderate	Hard	Easy	Moderate	Hard
MV3D (Chen et al. 2017)	RGB+LiDAR	74.97	63.63	54.00	86.62	78.93	69.80	–	–	–
ContFuse (Liang et al. 2018)	RGB+LiDAR	83.68	68.78	61.67	94.07	85.35	75.88	–	–	–
AVOD-FPN (Ku et al. 2018)	RGB+LiDAR	83.07	71.76	65.73	90.99	84.82	79.62	94.65	88.61	83.71
F-PointNet (Qi et al. 2018)	RGB+LiDAR	82.19	69.79	60.59	91.17	84.67	74.77	–	–	–
MMF (Liang et al. 2019)	RGB+LiDAR	88.40	77.43	70.22	93.67	88.21	81.99	–	–	–
VoxelNet (Zhou and Tuzel 2018)	LiDAR only	77.47	65.11	57.73	89.35	79.26	77.39	–	–	–
SECOND (Yan, Mao, and Li 2018)	LiDAR only	83.13	73.66	66.20	88.07	79.37	77.95	87.84	81.31	71.95
PointPillars (Lang et al. 2019)	LiDAR only	82.58	74.31	68.99	90.07	86.56	82.81	93.84	90.70	87.47
Fast Point R-CNN (Chen et al. 2019)	LiDAR only	85.29	77.40	70.24	90.87	87.84	80.52	–	–	–
STD (Yang et al. 2019)	LiDAR only	87.95	79.71	75.09	94.74	89.19	86.42	–	–	–
Part-A^2 (Shi et al. 2020)	LiDAR only	87.81	78.49	73.51	91.70	87.79	84.61	95.00	91.73	88.86
3DSSD (Yang et al. 2020)	LiDAR only	88.36	79.57	74.55	92.66	89.02	85.86	–	–	–
Point-GNN (Shi and Rajkumar 2020)	LiDAR only	88.33	79.47	72.29	93.11	89.17	83.90	–	–	–
Improvement	LiDAR only	***+0.80***	***+0.43***	***+3.25***	***+1.80***	***+0.45***	***+2.67***	–	–	–
PointRCNN (Shi, Wang, and Li 2019)	LiDAR only	86.96	75.64	70.70	92.13	87.39	82.72	95.90	91.77	86.92
Improvement	LiDAR only	***+2.17***	***+4.26***	***+4.84***	***+2.78***	***+2.23***	***+3.85***	***+0.67***	***+1.46***	***+2.12***
PC-RGNN (**Ours**)	LiDAR only	**89.13**	**79.90**	**75.54**	**94.91**	**89.62**	**86.57**	**96.57**	**93.23**	**89.04**

Table 1: Performance comparison in 3D object detection with previous state-of-the-art methods in terms of the car class on the KITTI test split (the results are computed by the official test server). 3D object detection and bird's eye view detection are evaluated by Average Precision (AP) with IoU threshold 0.7, while orientation estimation is validated by Average Orientation Similarity (AOS) as mentioned in (Geiger, Lenz, and Urtasun 2012).

nates $p_i \in R^3$ and the state value $s_i \in R^c$, a c-length vector that represents the point property. Given *V*, we construct a graph *G* = (*V*, *E*) by taking *V* as the vertices and connecting each point to its k neighbors as the edges *E*.

Compared to the point cloud classification and segmentation tasks, point cloud detection is more complex as it regresses object positions and the points in different locations non-equally contribute to the results. Therefore, unlike the typical graph model preliminarily attempted by (Wang et al. 2019), to dynamically adapt to the geometric structure of the object, we automatically learn the weight of each neighbor node when aggregating edge features. To this end, we design a GNN to extend the states of the vertices to include explicit position information and introduce an attention mechanism to assign individual weights to different nodes:

$$\begin{aligned}
&\Delta p_i{}^t = MLP_1^t(s_i^t) \\
&e_{ij}^t = MLP_2^t(concat(p_j - p_i + \Delta p_i{}^t, s_j^t - s_i^t, s_i^t)) \\
&\alpha_{ij}^t = softmax(MLP_3^t(e_{ij}^t)) = \frac{exp(MLP_3^t(e_{ij}^t))}{\sum_{k \in N_i} exp(MLP_3^t(e_{ik}^t))} \\
&s_i^{t+1} = sum(\alpha_{ij}^t e_{ij}^t \,|(i,j) \in E)
\end{aligned} \tag{6}$$

where s_i^t, s_j^t are the vertex features of the current node and its connecting node, respectively. MLP$_1$, MLP$_2$ and MLP$_3$ are used to learn position offset $\Delta p_i{}^t$, edge feature e_{ij}^t and edge weight α_{ij}^t respectively.

In addition, for 3D object detection, multiple instances belonging to the same category often have different point cloud discretization distributions, which makes the features learned by graph nodes sensitive to graph resolution and connection relationship. To alleviate this interference, we aggregate multi-scale graph contextual features, and besides the local attention applied to the neighborhood, we also employ an attention to weight the global feature after four AMS-GCNs, which is called global attention. This local-global attention mechanism substantially makes the feature more powerful in detection.

Experiments

In this section, we subsequently present experimental evaluation, containing datasets and implementation details, results, and ablation studies.

Datasets and Implementation Details

We evaluate our PC-RGNN on the KITTI 3D object detection benchmark (Geiger et al. 2013) which contains 7481 training point clouds and 7518 testing ones. For fair comparison, we follow the previous studies (Chen et al. 2017; Qi et al. 2018) to subdivide the original training data into a training set and a validation set, resulting in 3712 samples for training and 3769 for validation. Since the point cloud in KITTI does not have a complete object shape, we first use ShapeNetCars with 1824 samples derived from (Yuan et al. 2018) to train our point cloud completion module. In this way, we incorporate prior knowledge of the car class. Considering the variations in point cloud distribution, we extract 2000 object point clouds located in the ground-truth boxes with more than 2048 points from the KITTI training split to finetune the PC module. The ground-truth point cloud is created by uniformly sampling 2048 points on each shape. The low-quality point cloud is generated by randomly selecting a viewpoint and removing the points within a certain radius from original data. During training, each point cloud is transformed into the bounding box's coordinates and projected back to the world frame after completion.

Our PC-RGNN is trained on 4 GTX 1080Ti GPUs using PyTorch. The stage-1 sub-network is trained for 200 epochs with the batch size at 16 and the learning rate of 0.002. The PC module is first pretrained for 60 epochs by using the Adam optimizer with an initial learning rate of 0.0001 and a batch size of 32. Combining the PC and AMS-GNN modules, the stage-2 sub-network is trained for 80 epochs with

Method	AP_{3D} (IoU=0.7) (%)		
	Easy	Moderate	Hard
MV3D (Chen et al. 2017)	71.29	62.68	56.56
ContFuse (Liang et al. 2018)	86.32	73.25	67.81
AVOD-FPN (Ku et al. 2018)	84.41	74.44	68.65
F-PointNet (Qi et al. 2018)	83.76	70.92	63.65
VoxelNet (Zhou and Tuzel 2018)	81.98	65.46	62.85
SECOND (Yan, Mao, and Li 2018)	87.43	76.48	69.10
Fast Point R-CNN (Chen et al. 2019)	89.12	79.00	77.48
STD (Yang et al. 2019)	89.70	79.80	79.30
Part-A^2 (Shi et al. 2020)	89.47	79.47	78.54
3DSSD (Yang et al. 2020)	89.71	79.45	78.67
PointRCNN (Shi, Wang, and Li 2019)	88.88	78.63	77.38
PC-RGNN (**Ours**)	**90.94**	**81.43**	**80.45**
Improvement	***+2.06***	***+2.80***	***+3.07***

Table 2: Performance comparison with previous state-of-the-art methods in terms of the car class on the KITTI val split.

Module		AP_{3D} (IoU=0.7) (%)				
PC	AMS-GNN	Easy	Moderate	Hard	3D mAP	Gain
		88.88	78.63	77.38	81.63	–
	✓	89.62	79.56	78.43	82.54	↑**0.91**
✓		89.97	80.28	79.29	83.18	↑**1.55**
✓	✓	90.94	81.43	80.45	84.27	↑**2.64**

Table 3: Ablation study on the proposed PC and AMS-GNN modules.

Module			AP_{3D} (IoU=0.7) (%)				
MS	LA	GA	Easy	Moderate	Hard	3D mAP	Gain
			90.02	80.39	79.44	83.28	–
✓			90.16	80.57	79.73	83.49	↑**0.21**
✓	✓		90.78	81.25	80.29	84.11	↑**0.83**
✓	✓	✓	90.94	81.43	80.45	84.27	↑**0.99**

Table 4: Ablation experiments on the AMS-GNN module.

the batch size at 8 and the learning rate of 0.002 in an end-to-end manner.

Results

In the KITTI dataset, all the samples are divided into three sets with increasing difficulties, *i.e.* Easy, Moderate and Hard, according to different bounding box heights and occlusion/truncation levels. For example, the occlusion levels in the three difficulties are 'Fully visible', 'Partly occluded', and 'Difficult to see', respectively.

Firstly, we evaluate PC-RGNN on the test set by submitting detection results to the official server. The results are summarized in Table 1. PC-RGNN significantly outperforms the previously published state-of-the-art counterparts in all the tasks and difficulties. Point-GNN (Shi and Rajkumar 2020) is the pioneer graph-based detector, and PointRCNN (Shi, Wang, and Li 2019) is a representative two-stage approach. They are related to our method and compared to them, PC-RGNN makes two improvements at the second stage, *i.e.* point cloud refinement by the PC module and feature enhancement by the AMS-GNN module. Therefore, we choose the two methods as the baselines. The performance gains delivered by PC-RGNN are emphasized in slanted bold font which indicate that our method is effective in LiDAR-based 3D object detection in the challenging scenes.

We then report the performance achieved on the KITTI validation set in Table 2. We follow the official KITTI evaluation protocol, where the IoU threshold is 0.7 for the car class. The proposed PC-RGNN also outperforms all the other approaches by remarkable margins, especially in the Moderate and Hard difficulties. We present several challenging 3D detection examples in Fig. 5. We can observe that even in very difficult cases with distant and occluded objects, our network still reaches decent results, thanks to point cloud completion based data refining and attention multi-scale GNN based feature strengthening.

Ablation Study

To better verify our contributions, we conduct ablation studies on the KITTI validation set. We primarily investigate the impacts of our proposed PC module as well as the AMS-GNN module to the final results. We remove the PC module from PC-RGNN and replace our AMS-GNN with PointNet++ as our baseline, which achieves a 3D mAP of 81.63%. We then add the PC and AMS-GNN modules separately on the baseline for comparison. Finally, we combine both the two modules to update the scores.

The results are shown in Table 3. Only with either the PC module or the AMS-GNN module, the performance is boosted to 82.54% and 83.18%, respectively. Further, when combining both of them, it yields an mAP of 84.27%, largely superior to that of the baseline model by 2.64%. It clearly demonstrates the credits of the PC and AMS-GNN modules in PC-RGNN. Additionally, we visualize the object proposals before and after point cloud completion. As shown in Fig. 6, the original cars are barely recognizable due to the low-quality of the input data (*i.e.* long-distance and heavy occlusion). In contrast, the refined point clouds display much more reasonable geometric shapes. It confirms the fact that the point cloud completion operation significantly increases the accuracy of 3D object detection in difficult scenes.

To further analyze the effectiveness of AMS-GNN, we take a graph neural network with four typical graph convolution layers derived from (Wang et al. 2019) as our baseline. We successively add the multi-scale (MS) graph aggregation, the local attention (LA), and the global attention (GA) to the baseline. All the results are listed in Table 4. The improvements of the three parts are 0.21%, 0.62%, 0.16%, respectively. This shows that both the local-global attention mechanism and the multi-scale graph feature aggregation improve the discriminative power of GNN in capturing geometric characteristics of point clouds.

Besides, we validate the PC module. We take the complete PC-RGNN as our baseline, and its point cloud completion module (PC-M) only predicts additional points for refinement according to low-quality input. We then replace the graph encoder in PC-M with PointNet and remove the multi-resolution branch, generating two new models named PC-M without GE and PC-M without MR, respectively. Meanwhile, we change PC-M to the module which generates totally new shapes from low-quality input as existing

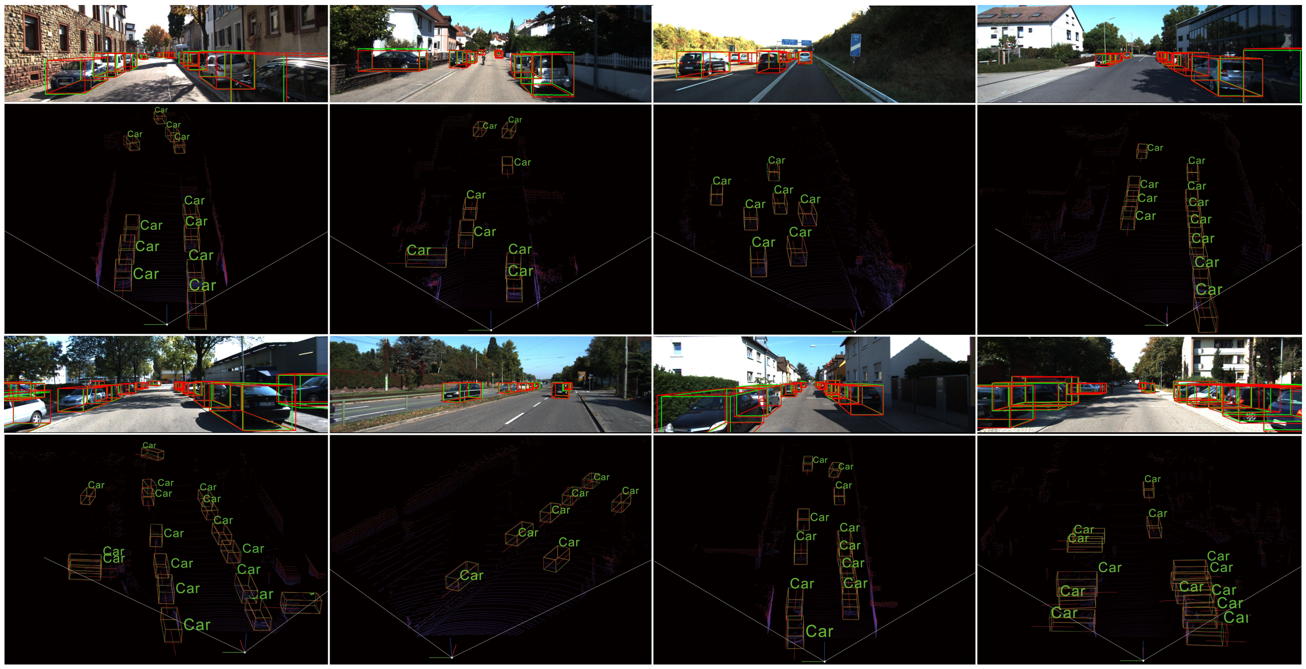

Figure 5: Qualitative results of PC-RGNN on the KITTI val split (best viewed with zoom-in). For each sample, the upper part is the image and the lower part is a representative view of the corresponding point cloud. The red boxes indicate the predicted results while the green ones denote the ground-truths.

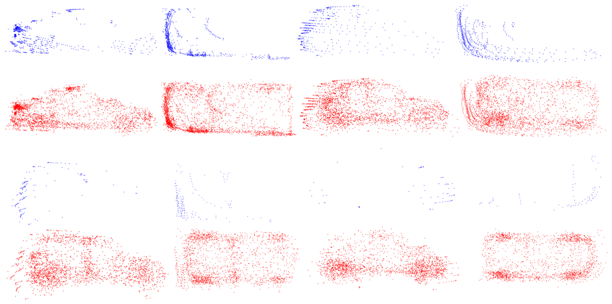

Figure 6: Point cloud completion results delivered by the PC module. Blue and red point clouds represent the data before and after this phase, respectively.

Method	AP_{3D} (IoU=0.7) (%)				
	Easy	Moderate	Hard	3D mAP	Gain
PC-M	90.94	81.43	80.45	84.27	–
PC-M without GE	90.72	80.48	79.81	83.67	↓**0.60**
PC-M without MR	90.55	80.60	79.66	83.60	↓**0.67**
PC-O	90.42	80.51	79.29	83.41	↓**0.86**

Table 5: Ablation experiments on the PC module.

point cloud completion networks do and name it as PC-O. As shown in Table 5, compared with our baseline, PC-M without GE and PC-M without MR decrease by 0.60% and 0.67%, respectively. This proves that the multi-resolution graph encoding strategy indeed captures additional geometric features. Compared with the baseline, PC-O encounters a drop of 0.86%. It suggests that despite the low-quality, the original points are critical to regress the object locations, and the way proposed in this study, *i.e.* PC-M, well handles this problem.

Conclusion

This paper proposes a novel two-stage approach, namely PC-RGNN, to LiDAR-based 3D object detection. It aims to address low-quality inputs, *i.e.* sparsely and partially sampled point clouds, caused by distant and/or occluded objects in challenging scenarios. To this end, we introduce a point cloud completion module for data refinement, and to the best of our knowledge, this is the first attempt to integrate this technique into 3D object detection framework. Furthermore, we design a new graph neural network for feature enhancement, which comprehensively captures the geometric relations among points by a local-global attention mechanism and multi-scale graph based contextual information aggregation. Extensive experiments are carried out on the KITTI dataset and state of the art results are reached, which demonstrate the effectiveness of the proposed PC-RGNN.

Acknowledgments

This work is partly supported by the National Natural Science Foundation of China (No. 62022011), the Research Program of State Key Laboratory of Software Development Environment (SKLSDE-2019ZX-03), and the Fundamental Research Funds for the Central Universities.

References

Achlioptas, P.; Diamanti, O.; Mitliagkas, I.; and Guibas, L. 2018. Learning representations and generative models for 3d point clouds. In *International Conference on Machine Learning*, 40–49.

Chen, X.; Ma, H.; Wan, J.; Li, B.; and Xia, T. 2017. Multi-view 3d object detection network for autonomous driving. In *IEEE Conference on Computer Vision and Pattern Recognition*, 1907–1915.

Chen, Y.; Liu, S.; Shen, X.; and Jia, J. 2019. Fast point r-cnn. In *IEEE International Conference on Computer Vision*, 9775–9784.

Engelcke, M.; Rao, D.; Wang, D. Z.; Tong, C. H.; and Posner, I. 2017. Vote3deep: Fast object detection in 3d point clouds using efficient convolutional neural networks. In *IEEE International Conference on Robotics and Automation*, 1355–1361.

Geiger, A.; Lenz, P.; Stiller, C.; and Urtasun, R. 2013. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research* 32(11): 1231–1237.

Geiger, A.; Lenz, P.; and Urtasun, R. 2012. Are we ready for autonomous driving? the kitti vision benchmark suite. In *IEEE Conference on Computer Vision and Pattern Recognition*, 3354–3361.

Huang, Z.; Yu, Y.; Xu, J.; Ni, F.; and Le, X. 2020. PF-Net: Point fractal network for 3D point cloud completion. In *IEEE Conference on Computer Vision and Pattern Recognition*, 7662–7670.

Ku, J.; Mozifian, M.; Lee, J.; Harakeh, A.; and Waslander, S. L. 2018. Joint 3d proposal generation and object detection from view aggregation. In *IEEE International Conference on Intelligent Robots and Systems*, 1–8.

Lang, A. H.; Vora, S.; Caesar, H.; Zhou, L.; Yang, J.; and Beijbom, O. 2019. Pointpillars: Fast encoders for object detection from point clouds. In *IEEE Conference on Computer Vision and Pattern Recognition*, 12697–12705.

Liang, M.; Yang, B.; Chen, Y.; Hu, R.; and Urtasun, R. 2019. Multi-task multi-sensor fusion for 3d object detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, 7345–7353.

Liang, M.; Yang, B.; Wang, S.; and Urtasun, R. 2018. Deep continuous fusion for multi-sensor 3d object detection. In *European Conference on Computer Vision*, 641–656.

Lin, T.-Y.; Dollár, P.; Girshick, R.; He, K.; Hariharan, B.; and Belongie, S. 2017a. Feature pyramid networks for object detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2117–2125.

Lin, T.-Y.; Goyal, P.; Girshick, R.; He, K.; and Dollár, P. 2017b. Focal loss for dense object detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2980–2988.

Qi, C. R.; Litany, O.; He, K.; and Guibas, L. J. 2019. Deep hough voting for 3d object detection in point clouds. In *IEEE International Conference on Computer Vision*, 9277–9286.

Qi, C. R.; Liu, W.; Wu, C.; Su, H.; and Guibas, L. J. 2018. Frustum pointnets for 3d object detection from rgb-d data. In *IEEE Conference on Computer Vision and Pattern Recognition*, 918–927.

Qi, C. R.; Su, H.; Mo, K.; and Guibas, L. J. 2017a. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 652–660.

Qi, C. R.; Yi, L.; Su, H.; and Guibas, L. J. 2017b. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in Neural Information Processing Systems*, 5099–5108.

Sarmad, M.; Lee, H. J.; and Kim, Y. M. 2019. Rl-gan-net: A reinforcement learning agent controlled gan network for real-time point cloud shape completion. In *IEEE Conference on Computer Vision and Pattern Recognition*, 5898–5907.

Shi, S.; Wang, X.; and Li, H. 2019. Pointrcnn: 3d object proposal generation and detection from point cloud. In *IEEE Conference on Computer Vision and Pattern Recognition*, 770–779.

Shi, S.; Wang, Z.; Shi, J.; Wang, X.; and Li, H. 2020. 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* .

Shi, W.; and Rajkumar, R. 2020. Point-gnn: Graph neural network for 3d object detection in a point cloud. In *IEEE Conference on Computer Vision and Pattern Recognition*, 1711–1719.

Wang, D. Z.; and Posner, I. 2015. Voting for voting in online point cloud object detection. In *Robotics: Science and Systems*, volume 1, 10–15607.

Wang, Y.; Sun, Y.; Liu, Z.; Sarma, S. E.; Bronstein, M. M.; and Solomon, J. M. 2019. Dynamic graph cnn for learning on point clouds. *ACM Transactions on Graphics* 38(5): 1–12.

Yan, Y.; Mao, Y.; and Li, B. 2018. Second: Sparsely embedded convolutional detection. *Sensors* 18(10): 3337.

Yang, Z.; Sun, Y.; Liu, S.; and Jia, J. 2020. 3dssd: Point-based 3d single stage object detector. In *IEEE Conference on Computer Vision and Pattern Recognition*, 11040–11048.

Yang, Z.; Sun, Y.; Liu, S.; Shen, X.; and Jia, J. 2019. Std: Sparse-to-dense 3d object detector for point cloud. In *IEEE International Conference on Computer Vision*, 1951–1960.

Yuan, W.; Khot, T.; Held, D.; Mertz, C.; and Hebert, M. 2018. Pcn: Point completion network. In *IEEE Conference on 3D Vision*, 728–737.

Zhou, Y.; and Tuzel, O. 2018. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, 4490–4499.