HRNet: 3D Object Detection Network for Point Cloud with Hierarchical Refinement

Bin Lu[a,*], Yang Sun[a], Zhenyu Yang[a], Ran Song[b], Haiyan Jiang[c], Yonghuai Liu[d]

[a]*Department of Computer, North China Electric Power University, Baoding, Hebei 071003, China*
[b]*School of Control Science and Engineering, Shandong University, Jinan, Shandong 250100, China*
[c]*College of Artificial Intelligence, Nanjing Agricultural University, Nanjing, Jiangsu 210095, China*
[d]*Edge Hill University, St Helens Road, Ormskirk, Lancashire L39 4QP, United Kingdom*

Abstract

Recently, 3D object detection from LiDAR point clouds has advanced rapidly. Although the second stage can improve the detection performance significantly, prior works concern little about the essential differences among different stages for the performance enhancement. To address this, this paper proposes a Hierarchical Refinement Network (HRNet) with two novel strategies. Firstly, we build the detection head on multi-scale voxel features to optimize the regression branch progressively with an effective Scale-aware Attentive Propagation (SAP) module. Then, we propose a Dynamic Sample Selection (DSS) module for the recalculation of the IoU during each stage to obtain more balanced positive and negative sample selections. Experiments over benchmark datasets show the effectiveness of our HRNet, particularly for car detection in the sparse point clouds.

Keywords: 3D object detection, LiDAR point clouds, multi-scale features, transformer, dynamic sample selection, hierarchical refinement.

*Corresponding author
Email address: lubin@ncepu.edu.cn (Bin Lu)

1. Introduction

3D object detection is considered as a critical task to provide environment perception for autonomous driving and industrial robots. Accurate position estimation is fundamental to prevent collisions. Point clouds from LiDAR can provide accurate pose information of objects, which is essential to accurate localization for object detection. 3D object detection from point clouds aims at detecting the classes, sizes and positions of objects in point cloud scenes [1]. Due to the intrinsic properties of LiDAR sensors, point cloud data is often highly incomplete and varied due to truncation, occlusion, sparsity, or different distances to the sensor, leading to the inability of delivering complete contextual information.

The task of 3D object detection is fundamentally formulated as a combination of classification and regression tasks. The former is used to identify the categories of the objects, while the latter is employed to pinpoint their locations. Both tasks require enhanced accuracy for an overall improvement in detection performance.

Single-stage methods [2–4] directly detect objects from raw point cloud scenes. They extract features from the whole point clouds only once, and then perform box classfication and regression without generating proposals. While they have high inference speed due to the unsophisticated feature extraction schemes, they usually produce lower accuracy than two-stage methods. The raw point clouds contain too much background information, which has a negative impact on extracting object features. Current two-stage methods [5–7] thus firstly generate proposals from raw point clouds by using efficient feature extraction methods [8–10], and then refine them with more complex algorithms to balance computational efficiency and accuracy.

To verify the impact of the second stage on IoU and confidence scores, we conducted experiments on the KITTI dataset with SECOND [11] and Voxel R-CNN [6] to calculate the IoUs

and confidence scores of all the predicted boxes and generate historgrams. All the experiments followed the default settings of OpenPCDet [1] toolbox. SECOND [11] is a classic single-stage method, and Voxel R-CNN [6] is a two-stage method that outperforms SECOND with an improvement of nearly 7%(Table 1). Moreover, Voxel R-CNN employs SECOND in its first stage and refines the detected object boxes in the second stage, making the experiment more convincing in demonstrating the effectiveness of the second stage. The median value is employed to verify the improvement across different stages. The median, denoted as median($\mathbb{O}$), is defined as:

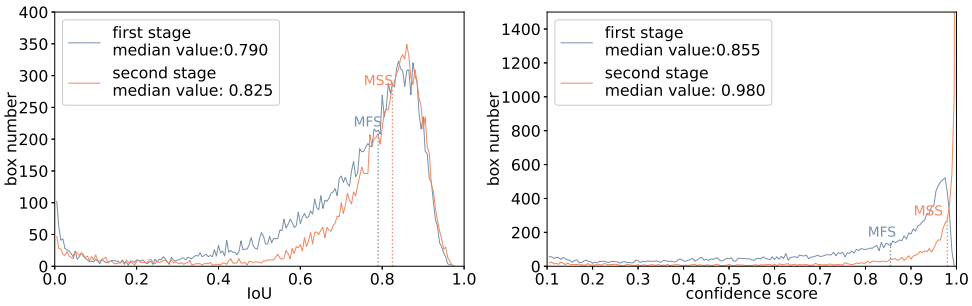
$$\text{median}(\mathbb{O}) = \begin{cases} o_{\frac{n+1}{2}} & \text{if } n \text{ is odd} \\ \frac{o_{\frac{n}{2}} + o_{\frac{n+1}{2}}}{2} & \text{if } n \text{ is even} \end{cases} \tag{1}$$

where $\mathbb{O} = \{o_1, o_2, ..., o_n\}$ be a set of observations (e.g., IoU scores, confidence scores, etc.), where each o_i is an individual observation, and n is the total number of observations. The observations are ordered from the smallest to the largest value. If the number n of observations is odd, the median is the middle number; otherwise, the median is the average of the two middle numbers. Unlike the mean value, which can be influenced by extreme outliers, the median provides a more representative measure of the central tendency.

As shown in Fig. 1, the median value of the confidence score is greatly improved after the second stage refinement, but the IoU gains rare improvements. This demonstrates that the refinement in the second stage proves to be more effective in enhancing the confidence score.

For further study of this phenomenon, we performed the second stage repeatedly and generated historgrams, as shown in Fig. 2. The experimental setup was the same as Fig. 1. As shown in Fig. 2(a), when the second stage is repeated twice, the median value of the IoU shifts slightly from 0.815 to 0.810, indicating a minimal drop. However, the median confidence score

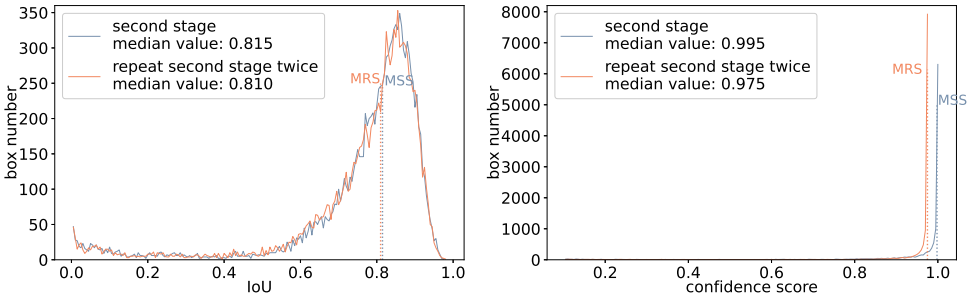
[1] https://github.com/open-mmlab/OpenPCDet



(a) IoU distribution in different stages. (b) Conf. distribution in different stages.

Figure 1: IoU and confidence distributions in different stages for object detection. MFS denotes the median value of the first stage, and MSS for the second stage.

decreases more significantly, from 0.995 to 0.975 as shown in Fig. 2(b). This demonstrates that repeatedly refining the proposals negatively impacts their qualities.



(a) IoU distribution of multi stages. (b) Conf. distribution of multi stages.

Figure 2: Comparison of IoU and confidence distributions in multi stages. MRS denotes the median value of the repeated stage.

To address this issue, we propose a multi-stage detector with two novel strategies. The Scale-aware Attentive Propagation (SAP) module is used for feature extraction and aggregation with different scale features, and the dynamic sample selection (DSS) module is used to dynamically balance the sample selection in each stage. The experimental results show that the proposed modules can improve the detection performance. We believe that our work will shed light on the

future development of 3D object detection methods and inspire further studies in this field.

2. Related works

According to the criteria whether the detected object proposals are refined or not, 3D object detectors can be divided into two categories: single-stage and multi-stage detectors.

Single-Stage 3D object detectors. Single-stage detectors directly generate object boxes from raw point clouds. VoxelNet [12] first divides the point clouds into regular 3D voxels and extracts voxel-wise features with 3D convolutional neural network (CNN), and then projects the 3D voxel features to the bird's eye view (BEV) for the generation of 3D object proposals. SECOND [11] proposes 3D sparse convolution for the improvement of the 3D convolution efficiency. PointPillars [3] encodes the point clouds into pseudo-images for the preservation of the spatial information, and then uses 2D CNN for feature extraction. 3DSSD [13] proposes furthest feature distance sampling to preserve more foreground points in downsampling, and then generates proposals at each candidate point. 3D-CenterNet [4] proposes a center regression module for improving the accuracy of the center point estimation.

Multi-Stage 3D object detectors. Multi-stage detectors generate proposals in the first stage, and then refine them in the later stages. Based on SECOND [11], which is usually used as a baseline method due to its simple structure, the subsequent voxel-based two-stage works investigate how to improve feature extraction methods for the refinement of the proposals from the first stage. PV-RCNN [5] proposes voxel-set abstract and generates proposals with 3D voxel features, and then performs refinement with raw point features. Voxel R-CNN [6] aggregates features from multi-scale 3D voxel features and acquires a good balance between accuracy and inference speed. Some works [14, 15] integrate attention mechanisms for enhancing the robustness of voxel features. PointRCNN [16] samples points with Farthest Point Sampling (FPS), and

then learns the semantic features of each point for the refinement of the proposals. STD [17] converts regions of interest (RoI) into a normalized coordinate system by the proposed PointsPool layer for more robust features. SVGA-Net [18] proposes a sparse voxel graph attention network for the construction of graphs over local and global voxels in order to enhance the extracted features. Graph R-CNN [19] performs point sampling in the second stage and learns features with graph-based feature aggregation. The point-based method is usually computationally expensive and usually requires a long inference time. BADet [20] takes each proposal as a node to construct local graphs, so that the boundary correlations can be explicitly exploited to refine them. CT3D [14] uses 3D voxel features to generate proposals, and then performs channel-wise Transformer [21] for the extraction of the features with raw points. Point Density-Aware Voxels (PDV) [22] introduces density information to the points, and then encodes points with Transformer.

3. Method

In this section, we detail the design of our HRNet, as shown in Fig. 3. It is an end-to-end multi-stage 3D object detection framework consisting of a 3D backbone, a 2D backbone, and a hierarchical refinement network. Firstly, the point clouds are voxelized via quantization and fed into the 3D backbone for feature extraction. Then, the 3D sparse features are mapped to the BEV for the generation of object proposals through 2D backbone. Subsequently, the proposed SAP is used to capture the subtle context information of multi-scale voxels. The DSS module is employed to balance sample selection across different refinements. Each module is described in detail in the subsequent sections.

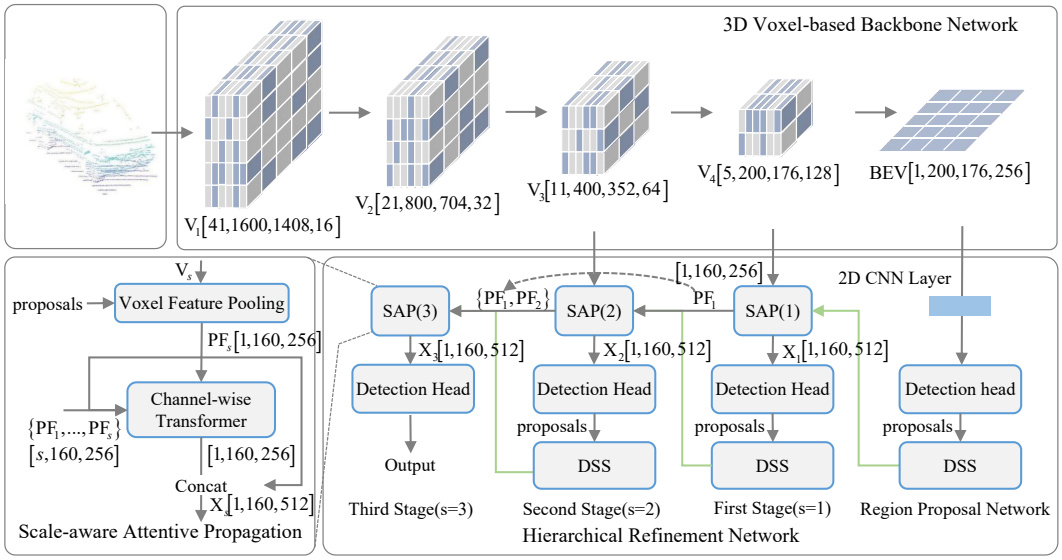


Figure 3: Network Structure of our proposed HRNet.

3.1. Preliminary

Before delving into the details, we first revisit the objectives of the object detection task. 3D object detection is to estimate a set of parameterized bounding boxes $\{x, y, z, l, w, h, \theta, c\}_{j=1}^{M} \in \mathbb{R}^7$, x, y, and z denote the center coordinates, l, w, and h for the length, width and height, θ for the orientation, and c represents the category. M denotes the total number of objects detected. This task involves two primary goals: classification and localization. The classification aims to identify the confidence scores $C = \{c_1, c_2, ..., c_P\}$ of the objects and the localization aims to estimate the positions of the proposals $R = \{r_1, r_2, ..., r_P\}$, where P represents the number of proposals.

C and R play crucial roles in the detection process, working together to filter the proposals. Firstly, the proposals are sorted by C as R_{sorted} and the top$-t$ proposals are selected as $R_{top-t} = \{r_1, r_2, ..., r_t\} \in R_{sorted}$. Then, given the ground truth $G = \{g_1, g_2, ..., g_K\}$, for each proposal r_j $(j \leq t)$, the IoU is calculated with each ground truth g_i $(i \leq K)$, and the highest IoU is determined:

$$IoU_{max,j} = \max_i(\text{IoU}(g_i, r_j)) \tag{2}$$

Each proposal is assigned to the ground truth corresponding to $IoU_{max,j}$. Consequently, each ground truth may correspond to multiple proposals. Proposals with an IoU greater than τ (IoU threshold) are selected as positive samples for the calculation of the regression loss and all the proposals are used to calculate the classification loss. Finally, these losses are summed up and used in backpropagation to train the model.

3.2. 3D Voxel-based backbone network

A point cloud is defined as $P = \{p_i\}_{i=1}^{N}$, where $p_i = \{x, y, z, r\}$ is the representation of each point consisting of 3-dimension coordinates x, y,and z and other features r. The point cloud is voxelized as $\mathbb{R}^{L\times W\times H}$, where L, W, and H denote the number of the voxels along x, y and z axes. We utilize four downsampling blocks consisting of sub-manifold convolution and 3D sparse convolution [11] to learn multi-scale voxel features. The multi-scale voxel features can be denoted as $V = \{V_1, V_2, V_3, V_4\}$, where the subscript number refers multi-scale voxel features with downsampling factors of 1×, 2×, 4×, and 8×. Subsequently, following [6], we compress $f_v^{8\times}$ onto the BEV for the generation of the object proposals. As shown in Fig. 3, the shape of output from each block of the model is denoted as $[z, x, y, d]$, where z, x, and y represent the size of the tensor along the z, x, and y axes, respectively, and d denotes the feature dimensions.

3.3. Hierarchical refinement network

Most existing two-stage detectors refine the 3D proposals by fine-grained features. The predicted boxes are firstly sorted by classification scores, and then go through Non-Maximum Suppression (NMS) by an IoU threshold. The regression branch needs to predict more variables than the classification branch and the latter can usually be optimized to approach the target label faster

than the former. This probably explains why the single-stage regressor performs usually not well due to the inconsistent optimization speed.

Inspired by a multi-stage 2D detector [23], we find that performing multi-stage refinement can improve the performance. However, the calculation cost is unbearable, which makes it difficult for further practical applications. So, we introduce the Hierarchical Refinement Network for achieving high efficiency and less computation with scale-aware refinements.

We propose our Hierarchical Refinement Network to refine the proposals generated by the first stage gradually. Specifically, we generate proposals at each scale voxel layer and the new proposals are used for further refinement with higher resolution voxel features, and then recalculate the IoU thresholds dynamically in each stage for more appropriate positive samples. We emphasize that HRNet can significantly enhance the efficiency of detection without adding excessive costs. It essentially uses the object position at higher scales with richer semantic and context information to guide its search at lower scales with more data information and thus propagates the object position information from low resolution to high resolution for higher accuracy.

The Hierarchical Refinement Network is built on the multi-scale voxel features from the 3D backbone. As illustrated in Fig. 3, every scale voxel feature is used to generate proposals in the Hierarchical Refinement Network. We have totally 4 proposal blocks.

The hierarchical refinement processes can be represented as:

$$PF_s = F(V_{5-s}, R_{s-1}) \tag{3}$$

$$PFS = \{PF_1, PF_2, ..., PF_s\} \tag{4}$$

$$X_s = \mathrm{SAP}_s(PFS, PF_s) \tag{5}$$

$$R_s = \mathrm{DH}(X_s) \tag{6}$$

where $s = \{1, 2, 3\}$ denotes the s-th refinement stage. F indicates the feature extraction method,

which is used for feature pooling within each proposal. PF denotes the pooling feature. X is the output from SAP, which is used to generate new proposals. When $s = 1$, the R_0 is the proposals output from Region Proposal Network (RPN). SAP() represents our proposed SAP module. DH() denotes the detection head, which consists of two separate linear layers to generate C for classification and R for localization.

SAP module consists of two encoders, where a channel-wise Transformer is used to perform feature propagation and the Voxel Set Abstract (VSA) [6] is used to aggregate 3D voxel features within the proposal. As a commonly used method for processing voxel features, VSA first divides RoI into smaller grids. Then, features are pooled with a PointNet [24] module within each grid. Finally, all the pooled features are concatenated and processed through an MLP to ensure a consistent feature dimension.

The shape of X_s is $[1, M, D]$, where M is the number of proposals, D is the feature dimensions. The shape of PF is $[1, M, N]$. N is the feature dimension, as shown in Fig. 3. For H-head self-attention, the calculation of $\mathbf{Q}$, $\mathbf{K}$, $\mathbf{V}$ can be summarized as $\mathbf{Q} = [\mathbf{Q}_1, \mathbf{Q}_2, ..., \mathbf{Q}_H]$, $\mathbf{K} = [\mathbf{K}_1, \mathbf{K}_2, ..., \mathbf{K}_H]$, and $\mathbf{V} = [\mathbf{V}_1, \mathbf{V}_2, ..., \mathbf{V}_H]$, $\mathbf{Q}_h$, $\mathbf{K}_h$, $\mathbf{V}_h \in \mathcal{R}^{M \times \frac{D}{H}}$, where $h \in \{1, 2, ..., H\}$. Denote $\mathbf{Q}_h = W_q^h PF_i$, $\mathbf{K}_h = W_k^h PF_i$, and $\mathbf{V}_h = W_v^h PF_i$, where W_q^h, W_k^h and $W_v^h \in \mathcal{R}^{\frac{M}{H} \times M}$. Sine and cosine are applied for positional encoding [21]. The output is:

$$\mathrm{A}_h = \mathrm{Softmax}\left(\frac{\mathbf{Q}_h \mathbf{K}_h^T}{\sqrt{d_k}}\right)\mathbf{V}_h, h = 1, ..., H \tag{7}$$

$$\mathrm{A} = \mathrm{Concat}(A_1, A_2, ..., A_h) \tag{8}$$

$$X = \mathrm{MLP}(A) \tag{9}$$

where $d_k = \sqrt{\frac{D}{H}}$, and MLP() is an Multilayer Perceptron (MLP) with one hidden layer.

3.4. Dynamic sample selection

Existing methods [6, 22] typically use a fixed IoU threshold as a hyperparameter for classifying positive and negative samples. During the multi-stage refinements, the qualities of the proposals progressively change. A fixed threshold is hard to account for this situation. For example, some low-quality proposals might still be treated as positive samples in the later stages. This weakens the model to learn features from high-quality samples. ATSS [25] introduces an innovative statistics-based sampling strategy for object detection, offering potential research directions in multi-stage sample selection. This inspired us to dynamically set IoU thresholds at each stage to filter high-quality positive samples effectively. Our DSS module adaptively selects samples using the statistical characteristics of IoUs, allowing the model to assign different IoU thresholds at each stage. It is both simple and effective, bringing no additional computations.

The number of the proposals is usually much larger than the ground truth, and thus we find the k-nearest proposals for each ground truth, where k is a sensitive hyperparameter. The distance d is calculated by the center coordinates of the proposal and ground truth, which can be calculated as:

$$d = \sqrt{(x_R - x_G)^2 + (y_R - y_G)^2 + (z_R - z_G)^2}, \tag{10}$$

where x, y, and z represent the center coordinate of a bounding box, subscript R denotes the proposal and G denotes the ground truth.

The proposed DSS first calculates the 3D IoU between the ground truth and its corresponding k-nearest proposals. Then, their mean and standard deviation are calculated as μ and σ for each ground truth in each point cloud. Therefore, the DSS can learn a specific IoU threshold τ for each ground truth.

After that, we take $\tau = \mu + \sigma$ as the final IoU threshold of each ground truth in each stage to

select positive samples for each point cloud. When the proposals are closer to the ground truth, τ dynamically adjusts based on the distribution of IoUs. Finally, we select proposals $\{R\}_{j=1}^{M}$ with larger IoUs than τ as positive samples $\{\mathcal{R}\}_{\mathcal{J}=1}^{\mathcal{M}}$ for regression, where the latter is a subset of the former and $M \geq \mathcal{M}$.

The intention of introducing IoU statistical characteristics for the selection of positive samples is that if the proposals have high quality, most IoUs are larger than the average μ. This does not perform well for the selection of those proposals with varied IoU qualities relative to each ground truth. For those objects with sparse points, the proposals commonly have wide divergence in IoUs, and a high IoU threshold makes it hard for the model to focus on them. So, τ is used to adjust the proposals with different qualities in IoU. In addition, the proposals of objects with sparse points often have large differences in location. σ is used to raise the threshold for improving the quality of positive samples. This leads to a balance in selecting positive samples of objects with different sparsities in LiDAR data.

3.5. Detection head and training loss

Our HRNet can be trained in an end-to-end way without any pre-trained operations by the multi-task loss composed of RPN loss L_{rpn} and hierarchical proposal refinement loss L_{HR}. The total loss is calculated as:

$$L_{total} = L_{rpn} + L_{HR}. \tag{11}$$

Specifically, we have 3 stages in Hierarchical Refinement Network. The RPN head consists of two fully connected layers with nonlinear activation functions for confidence prediction loss L_{cls} and box regression loss L_{reg}. L_{rpn} is thus composed of:

$$L_{rpn} = L_{cls} + L_{reg}. \tag{12}$$

In the remainder of this section, c_R and r_R denote the classification label and regression target

of the object proposals R, and c_R^* and r_R^* denote the predicted values, respectively.

Following [11], L_{cls} is calculated over all the proposals $\{R\}_{j=1}^{M}$ and we use the Focal Loss [26] to balance the loss of positive and negative samples:

$$L_{cls} = -\alpha \sum_{j=1}^{M} (1 - c_{R_j}^*)^{\gamma} log(c_{R_j}^*), \tag{13}$$

where M denotes the number of proposals. α and γ are hyperparameters of the Focal Loss, which are empirically set as 0.25 and 2 in the training process, respectively [5].

L_{reg} is calculated by the Smooth-L1 function over the positive samples $\{\mathcal{R}\}_{\mathcal{J}=1}^{\mathcal{M}}$ only:

$$L_{reg} = \sum_{\mathcal{J}=1}^{\mathcal{M}} L_{Smooth-L1}(r_{\mathcal{R}_\mathcal{J}}, r_{\mathcal{R}_\mathcal{J}}^*), \tag{14}$$

where $\mathcal{M}$ denotes the number of positive samples.

The total loss L_{rpn} can then be calculated as:

$$L_{rpn} = -\alpha \sum_{j=1}^{M} (1 - c_{R_j}^*)^{\gamma} log(c_{R_j}^*) + \sum_{\mathcal{J}=1}^{\mathcal{M}} L_{Smooth-L1}(r_{\mathcal{R}_\mathcal{J}}, r_{\mathcal{R}_\mathcal{J}}^*). \tag{15}$$

L_{HR} is similar to L_{RPN}, which consists of classification loss $L_{H\text{-}cls}$ and regression loss $L_{H\text{-}reg}$. L_{HR} can be calculated as:

$$L_{HR} = L_{H\text{-}cls} + L_{H\text{-}reg}. \tag{16}$$

$L_{H\text{-}cls}$ can be calculated as:

$$\begin{aligned} L_{H\text{-}cls} &= \sum_{s} L_{H\text{-}cls_s} \\ &= \sum_{s} (\sum_{j=1}^{M} (-c_{R_j} log(c_{H_{s,j}}^*) - (1 - c_{R_j}) log(1 - c_{H_{s,j}}^*))) \end{aligned} \tag{17}$$

where $c_{H_{s,j}}^*$ denotes the classification score of proposals R_j generated and $L_{H\text{-}cls_s}$ is the classification loss at the s-th stage.

We take the binary cross entropy to calculate $L_{H\text{-}cls_s}$, and then sum all the stage losses $L_{H\text{-}cls_s}$

together as the final classification loss of $L_{H\text{-}cls}$. $L_{H\text{-}reg}$ consists of all the stage regression losses $L_{H\text{-}reg_s}$, and their calculations are similar to L_{reg} in L_{rpn} above.

4. Experiments

In this section, we first describe the experimental setup and training details of the proposed method, then validate and compare it over the KITTI [27] and the Waymo [28] datasets with the SOTA ones, and finally conduct ablation studies on the KITTI benchmark for the demonstration of the validity of the proposed modules.

4.1. Experimental setup

4.1.1. Datasets

The KITTI dataset includes 7,481 training images/point clouds and 7,518 test images/point clouds. The training and validation sets contain 3,769 and 3,712 point clouds, respectively. The KITTI dataset contains several categories such as car, pedestrian, cyclist, etc. Three difficulty levels are classified as easy, moderate, and hard. The KITTI test set does not provide publicly available ground truth labels, and the detection results need to be submitted to the KITTI official server for evaluation, presenting a challenge to the generalization ability of the model.

We test our model with the official KITTI evaluation metrics in both the validation and test sets. Currently, the most commonly used evaluation criterion is 40 recall positions, but some previous works used 11 instead. A numerical difference exists between them, and we explicitly mark the evaluation criteria used in our experiments. The Waymo dataset is one of the largest public datasets for autonomous driving, which contains 158K point clouds as the training set and 40K point clouds as the validation set with full 360° fields of view.

4.1.2. Network architecture and hyperparameter settings

The 3D backbone network contains four 3D sparse convolution layers, and each layer contains a series of submanifold convolutions, batch normalization, and a nonlinear activation function. The number of convolution kernels is 16, 32, 64, and 64, respectively, which represent the dimension of the 3D voxel feature vector at each scale. The 2D backbone network consists of two channels that perform convolution operations with $3 * 3$ convolution kernels at different resolutions. Finally, we concatenate the outputs of the two channels to generate region proposals.

In the second stage, each RoI is divided into $6 * 6 * 6$ regular grids [6], and the grid center aggregates features with two types of range for each scale features: $[0.4, 0.4]$, $[0.4, 0.8]$, and $[0.8, 1.6]$, respectively. The Manhattan distances of each range are thus $[2, 2, 2]$ and $[4, 4, 4]$ along x, y, and z axes, respectively. 16 points are sampled in each range. For memory efficiency, the voxel feature at each scale is unified to 32 dimensions.

4.1.3. Implementation detail

For the KITTI dataset, the x-axis of the point cloud scene points towards the front of the vehicle, while the y-axis points to the left, and the z-axis points upwards. All detectors use the same standard detection range: $[0, 70.4]$m for the x-axis, $[-40.0, 40.0]$m for the y-axis, and $[-3.0, 1.0]$m for the z-axis, with a voxel size of $[0.05, 0.05, 0.05]m$ for the three axes. For the Waymo dataset, the coordinate system aligns with that of the KITTI dataset. All detectors use the same standard detection range: $[-75.2, 75.2]$m for the x-axis and y-axis, and $[-2.0, 4.0]$m for the z-axis, with a voxel size of $[0.1, 0.1, 0.15]$m) for the three axes.

We use the same data augmentation methods as [11], which include: 1) increasing the ground truth instances in each scene; 2) scaling the points with random factors in the range of $[0.95, 1.05]$ and rotating the scene around the z-axis with random angles in $[-\pi/4, +\pi/4]$; 3) rotating all

ground truth instances around the z-axis with random angles in $[-\pi/2, +\pi/2]$ to simulate steering; and 4) randomly flipping the scene along the x-axis.

During the training process, each stage generates 128 proposals and we use adam_onecycle optimizer with a learning rate of 0.001 for the KITTI dataset and 0.01 for the Waymo dataset, a momentum of 0.9, and a weight decay of 0.01. We use 8 GTX3090 GPUs to train our model with a batch size of 2, 80 epochs for the KITTI dataset and 30 epochs for the Waymo dataset. For more detailed training configurations, please refer to OpenPCDet. k was empirically set as 12 for selecting the nearest proposals to calculate the threshold for sample selection.

4.1.4. Inference

During the inference, the predicted boxes are sorted by IoU-guided confidence score, and the top 100 boxes are selected for NMS with an IoU threshold of 0.1. In the post-processing, the confidence threshold is set as 0.7 to filter the boxes.

4.2. Results on the KITTI dataset

For the KITTI dataset, the methods are typically evaluated on both the validation and test sets. We classify the methods with different stages (single-stage and multi-stage) and types (point based, voxel based, and point-voxel based). Our model is categorized as a multi-stage, and voxel-based method. We report the 3D average precision (3D AP) of the selected SOTA methods on the KITTI test set, as shown in Table 1. All results are reported over 40 recall positions and the best ones are marked in bold black, the second are marked with underscore. The results are reported from the KITTI official benchmark. From Table 1, we can observe that the multi-stage methods are generally better than the single-stage ones. This confirms our motivation for the development of our proposed HRNet as a multi-stage one.

Table 1: 3D AP of various methods for car detection on the KITTI test set over 40 recall positions. All results are reported from the KITTI official benchmark. The top result is in bold, and the second-best is underscored.

Modality	Type	Method	3D AP(%)			
			Easy	Mod.	Hard	mAP
Single-stage	Point based	3DSSD[13]	88.36	79.57	74.55	80.83
		SA-SSD[30]	88.75	79.79	74.16	80.90
		SE-SSD[31]	91.49	82.54	77.15	83.73
		CIA-SSD[32]	89.59	80.28	72.87	80.91
		3D-CenterNet[4]	86.83	80.17	75.96	80.99
		IA-SSD[33]	88.87	80.32	75.10	81.43
	Point-Voxel based	VIC-Net[34]	88.25	80.61	75.83	81.56
		HVPR[35]	86.38	77.92	73.04	79.11
	Voxel based	SECOND[11]	84.65	75.96	68.71	76.44
		PointPillars[3]	82.58	74.31	68.99	75.29
		3D IoU Loss[36]	86.16	76.50	71.39	78.02
		Associate-3Ddet[37]	85.99	77.40	70.53	77.97
Multi-stage	Point based	PointRCNN[16]	86.96	75.64	70.70	77.77
		CT3D[14]	87.83	81.77	77.16	82.25
		Graph-Po[19]	91.79	**83.18**	77.98	83.45
	Point-Voxel based	Fast Point R-CNN[38]	85.29	77.40	70.24	77.64
		STD[17]	87.95	79.71	75.09	80.92
		PV-RCNN[5]	90.25	81.43	76.82	82.83
		FV2P[39]	88.53	81.58	77.37	82.49
		BADet[20]	89.28	81.61	76.58	82.49
		SIENet[7]	88.22	81.71	77.22	82.38
	Voxel based	Part-A2[40]	87.81	78.49	73.51	79.94
		Voxel R-CNN[6]	90.90	81.62	77.06	83.19
		PDV[22]	90.43	81.86	77.36	83.22
		Focals Conv[41]	90.20	82.12	77.50	83.27
		GraR-Vo[19]	91.29	82.77	77.20	83.75
		3D Cascade RCNN[42]	90.46	82.16	77.31	83.31
		CasA+V[29]	91.58	83.06	**80.08**	**84.91**
		Ours	**91.82**	82.91	78.07	84.27

Our HRNet outperforms other multi-stage methods in car detection, achieving 91.82%, 82.91%, and 78.07% for the easy, moderate, and hard difficulty levels, respectively. Particularly for the easy difficulty level, we have achieved the best performance. For the hard difficulty level and the mean average precision (mAP) metric, our HRNet ranks second, demonstrating strong competitiveness. The experimental results indicate that multi-stage refinements by the features at different resolutions enhance the performance of the regression branch. Additionally, the proposed SAP effectively extracts more contextual information for localization. Furthermore, the proposed DSS balances positive and negative samples, which reduces the ambiguity in calculating regression loss.

For the moderate difficulty level, our model is slightly behind Graph-Po[19] and CasA+V[29]. The Graph-Po constructs graph attention directly on the points, which potentially enables it to

capture more detailed spatial information from the point clouds. CasA+V applies voxel features across more scales than HRNet for refinement, yielding competitive performance. However, this advantage of CasA+V comes with significantly slower training and inference times. Table 3 presents the inference speeds of various methods. Each model was evaluated with a single GTX3090 GPU and AMD EPYC 7543 32-Core Processor, with the batch size set to 1. The experimental results demonstrate that our HRNet achieves a better trade-off between accuracy and efficiency, thus enhancing its practical applicability.

Table 2: 3D AP of various methods over the KITTI validation set in 11 and 40 recall positions for car category.

Modality	Type	Method	3D AP recall 11(%)				3D AP recall 40(%)			
			Easy	Mod.	Hard	mAP	Easy	Mod.	Hard	mAP
single-stage	Point based	SA-SSD[30]	90.15	79.91	78.78	82.95	93.23	84.30	81.36	86.30
		SE-SSD[31]	**90.21**	85.71	79.22	85.05	93.19	86.12	83.31	87.54
		CIA-SSD[32]	90.04	79.81	78.80	82.88	-	-	-	-
		3D-CenterNet[4]	86.83	80.17	75.96	80.99	-	-	-	-
		IA-SSD[33]	-	-	-	79.57	-	-	-	-
	Point-Voxel based	VIC-Net[34]	89.58	84.40	78.86	84.28	-	-	-	-
		HVPR[35]	-	-	-	-	91.14	82.05	79.49	84.23
	Voxel based	Associate-3Ddet[37]	-	-	-	-	89.29	79.17	77.76	82.07
multi-stage	Point based	CT3D[14]	89.54	86.06	78.99	84.86	92.85	85.82	83.46	87.38
		Graph-Po[19]	-	-	-	-	93.27	**86.50**	83.87	87.88
	Point-Voxel based	PV-RCNN[5]	89.35	83.69	78.70	83.91	-	-	-	-
		FV2P[39]	-	-	-	-	93.00	85.61	83.43	87.35
		BADet[20]	90.06	85.77	79.00	84.94	-	-	-	-
		SIENet[7]	-	-	-	-	92.49	85.43	83.05	86.99
	Voxel based	Part-A2[40]	89.47	79.47	78.54	82.49	-	-	-	-
		Voxel R-CNN[6]	89.41	84.52	78.93	84.29	92.38	85.29	82.86	86.84
		PDV[22]	-	-	-	-	92.56	85.29	83.05	86.97
		Focals Conv[41]	89.52	84.93	79.18	84.54	-	-	-	-
		GraR-Vo[19]	-	-	-	-	**93.33**	86.12	83.29	87.58
		3D Cascade RCNN[42]	90.05	86.02	79.27	85.11	93.20	86.19	83.48	87.62
		CasA+V[29]	89.88	**86.58**	79.38	85.28	93.21	86.37	83.93	87.84
		Ours	90.01	86.57	**79.46**	**85.35**	93.31	86.43	**84.00**	**87.91**

Considering that the 3D object detection accuracy has been significantly improved in the past three years, we choose the representative methods in the last three years for comparison. Since 08.10.2019, KITTI has modified its AP calculation criteria from 11 recall positions to 40, and many previous methods reported their results with different recall positions. For a fair comparison, we present the results for both 11 and 40 recall positions in Table 2. Our model performs again among the best over both 11 and 40 recall positions, showing a significant improvement

Table 3: Inference speeds in milliseconds (ms) of various methods.

Method	Speed (ms)	mAP(%)
PDV[22]	183	83.22
CT3D[14]	130	82.25
Voxel R-CNN[6]	155	83.19
PV-RCNN[5]	200	82.83
Graph-Po[19]	152	83.45
CasA+V[29]	298	**84.91**
Ours	185	84.27

in mAP. The superior performance on both the validation and test sets further demonstrates the remarkable generalization ability of our model.

We also conduct experiments in the pedestrian and cyclist categories to further evaluate the performance of our model. Recent methods rarely report the detection results for pedestrian and cyclist categories due to a higher probability of false detections. For comparison, we select representative two-stage detectors of different types (point based, voxel based, and point-voxel based) and multi-stage detector (CasA+V). Table 4 shows the detection performance of different methods over 40 recall positions. HRNet outperforms other methods with an mAP of 67.15% and 81.15% in 3D detection of pedestrian and cyclist, respectively.

Table 4: 3D AP of various methods on the KITTI validation set calculated over 40 recall positions for pedestrian and cyclist.

Method	Pedestrian 3D AP(%)				Cyclist 3D AP(%)			
	Easy	Mod.	Hard	mAP	Easy	Mod.	Hard	mAP
CT3D[14]	65.73	58.56	53.04	59.11	91.99	71.60	67.34	76.98
Voxel R-CNN[6]	68.22	60.97	55.63	61.61	91.28	72.54	68.46	77.43
PV-RCNN[5]	64.26	56.67	51.91	57.61	88.88	71.95	66.78	75.87
PDV[22]	66.90	60.80	55.85	61.18	92.72	74.23	69.60	78.85
CasA+V[29]	**73.95**	**66.62**	59.97	66.85	92.78	73.94	69.37	78.70
Ours	73.23	66.32	**61.91**	**67.15**	**93.55**	**76.65**	**73.25**	**81.15**

As shown in Table 5, our model has 39.5M parameters. Compared to other SOTA methods, the model size of HRNet is not significantly larger.

Fig. 4 shows the recall-precision curves of HRNet compared with other advanced methods. We select four classic methods for comparison. CT3D is the first method to introduce Transformer for the extraction of point-wise features. Voxel R-CNN is the most classic voxel based

Table 5: Parameters of various methods.

Method	Params
Part-A2[40]	63.8M
PV-RCNN[5]	13.1M
Voxel R-CNN[6]	7.6M
Graph-Po[19]	19.1M
CasA+V[29]	44.3M
Ours	33.2M

Table 6: mAP of various methods for the vehicle detection at two difficulty levels on the Waymo validation set. '-' indicates no public data.

Methods	LEVEL_1				LEVEL_2			
	overall	[0,30)m	[30,50)m	[50,inf)m	overall	[0,30)m	[30,50)m	[50,inf)m
MVF[43]	62.93	86.30	60.02	36.02	-	-	-	-
PointPillars[3]	56.62	81.01	51.75	27.94	-	-	-	-
PV-RCNN[5]	70.30	91.92	69.21	42.17	65.36	91.58	65.13	36.46
VoTr-TSD[44]	74.95	92.28	73.36	51.09	65.91	-	-	-
Voxel R-CNN[6]	76.08	92.44	74.67	54.69	68.06	91.56	69.62	42.80
CT3D[14]	76.30	92.51	75.07	55.36	69.04	91.76	68.93	42.60
Ours	**77.85**	**93.29**	**76.62**	**56.87**	**69.76**	**92.22**	**70.03**	**43.69**

method with simple network architecture and high performance. PV-RCNN is the first method to combine the advantage of points and voxels for the generation of high quality proposals. PDV takes density information of point clouds to refine the proposals, which is a representative work of introducing additional information for object detection. The solid line denotes our HRNet and the dashed line denotes the others. The curve closer to the up-right corner indicates thta the detector has both higher accuracy and recall rate, which represents better performance. HRNet performs best among the selected detectors.

4.3. Results on the Waymo Dataset

We report the detection result on the Waymo dataset for further validation of our proposed HRNet. The Waymo dataset provides two levels of difficulty, i.e., LEVEL_1 for the ground truth objects with at least five LiDAR points and LEVEL_2 for the ground-truth objects with at least one LiDAR point.

The results were calculated by the official released tools with the mAP. The IoU threshold was set as 0.7. The detection results are presented in two modalities. The first modality is based

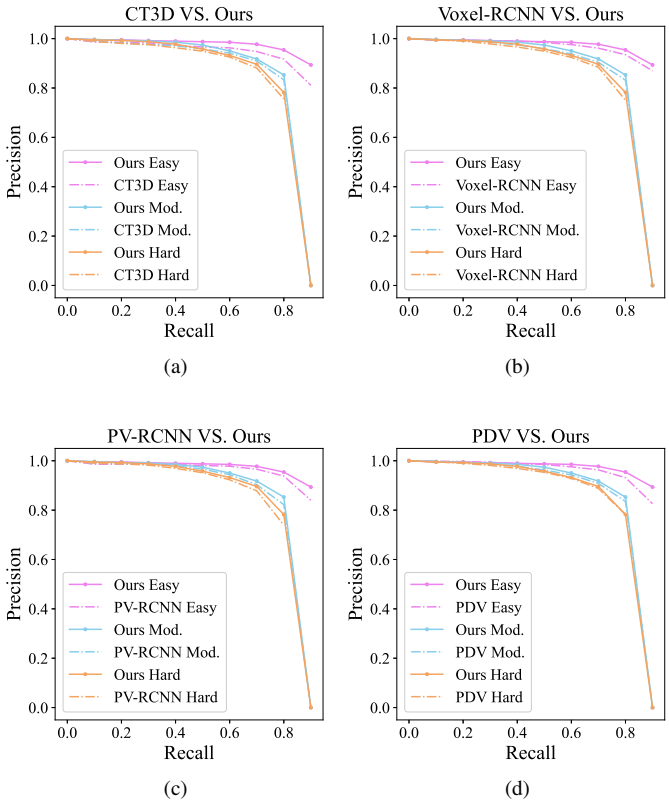


Figure 4: The recall-precision curves of different methods over the KITTI test set.

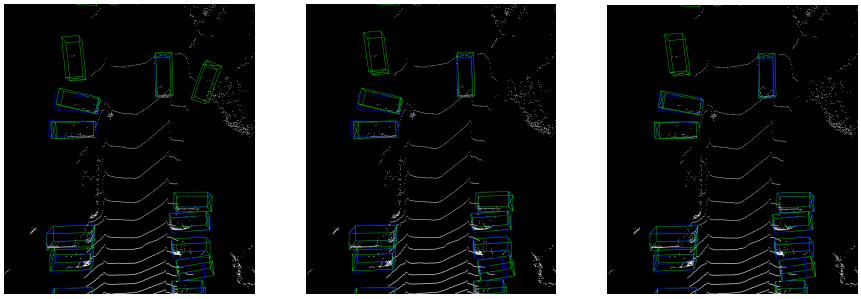
on the distance between objects and the LiDAR sensor: 0-30m, 30-50m, and 50-inf. The second modality considers two difficulty levels: LEVEL_1 and LEVEL_2. We report the detection results subject to two difficulty levels with different distances in Table 6. Our HRNet achieves the best performance compared with other SOTA detectors.

Specifically, our model achieves 77.85% at LEVEL_1 and 69.76% at LEVEL_2, respectively, which indicates that our HRNet is powerful in detecting objects from point clouds. For vehicles far away from the sensor, particularly in the $[50, inf)$m range, our HRNet achieves 56.87% and 43.69% at LEVEL_1 and LEVEL_2, respectively, outperforming others by as much as 1.51% and 1.09%. These results clearly demonstrate that the propagation of semantic and context in-

formation from the higher scale to the lower scale in conjunction with the DSS are powerful for the refinement of the object proposals detected in the first stage.

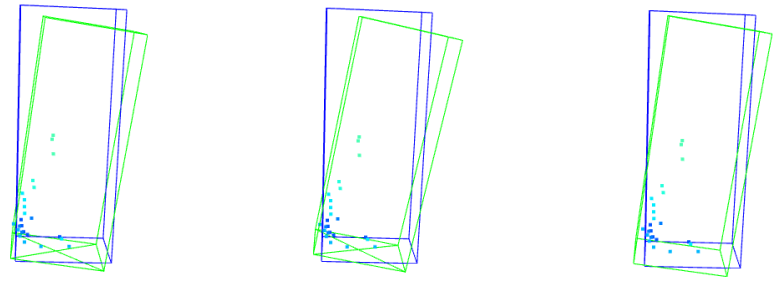
4.4. Visualization

Qualitative experimental results are presented in Fig. 5 and Fig. 6 for the KITTI dataset. The ground truths and predictions are marked in blue and green, respectively. Fig. 5 presents the results from different stages of the HR module. In Fig. 5(a), there exists a false detection in the upper right corner. In the subsequent stage, the model corrects this misdetection, as shown in Fig. 5(b). In Fig. 5(c), the predictions get closer to the ground truths compared with Fig. 5(b), which means that the model has improved the localization of the detected objects.



(a) second stage (b) third stage (c) forth stage

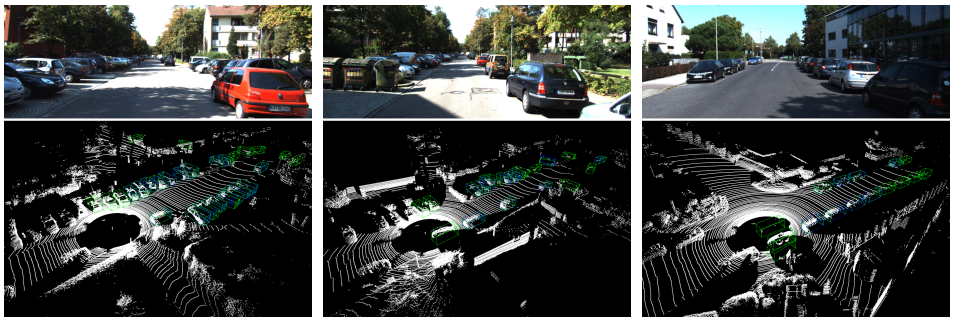
Figure 5: Improvement in object detection at each stage of the HR module.



(a) second stage (b) third stage (c) forth stage

Figure 6: Detailed improvement in object detection at each stage of the HR module.

Fig. 6 presents the details of each stage. All figures are in the bird eye view. The green boxes are the predictions and the blue ones indicate the ground truths. Gradually, the green boxes align more closely with the ground truths after each refinement stage, which demonstrates the effectiveness of each stage.



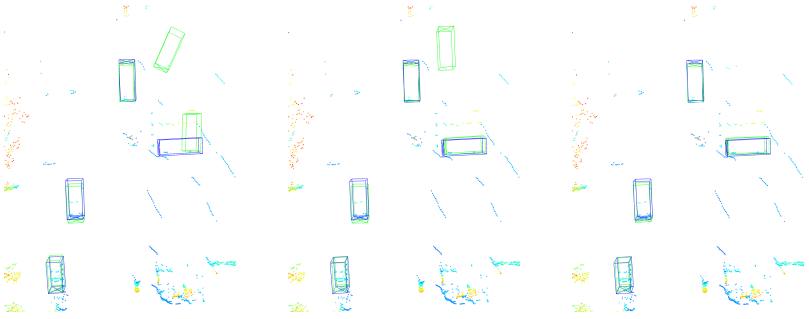
(a) (b) (c)

Figure 7: Qualitative results of HRNet for car detection.

Fig. 7 presents the detection results in three scenes. The RGB images related to the point clouds are presented above. The predicted boxes have a high overlap with ground truths, which indicates that the positioning is accurate. Furthermore, HRNet is capable of detecting cars with sparse points and far away from the sensor.

Fig. 8 displays the qualitative results from various methods. We selected a scene with sparse point clouds. In Fig. 8(a), Voxel R-CNN has a false detection and detects a car with wrong direction. PV-RCNN performs better than Voxel R-CNN as shown in Fig. 8(b), but it still has false detections when the point clouds are sparse. Our HRNet can stably detect cars accurately without any false detections, as shown in Fig. 8(c).

However, HRNet still has certain limitations when facing scenarios with extreme sparsity, occlusions, and truncation, as shown in Fig. 9. Specifically, cars A and B are not correctly detected. The point clouds of A and B are extremely sparse, and the voxel-based pooling strategy



(a) Voxel R-CNN (b) PV-RCNN (c) Ours

Figure 8: Qualitative experiments of various methods for car detection: (a) Voxel R-CNN, (b) PV-RCNN, and (c) Our model.

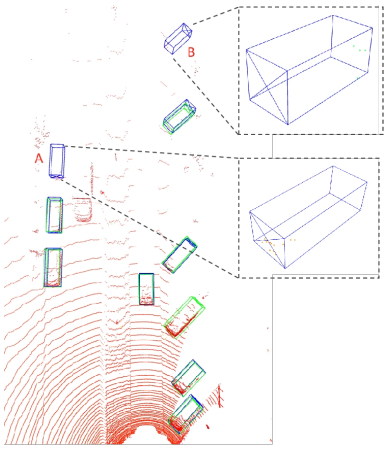


Figure 9: Visualization of the limitations of our model in detecting cars in occlusion scenarios.

might further result in the loss of spatial information. Future work could focus on reducing the loss of spatial information in sparse point clouds, for example, by incorporating original point cloud data to learn more effective features.

Fig. 10 illustrates the improvements in the quality of bounding boxes at different refinement stages. For the car category, Fig. 10(a) shows that the median IoU of the proposals progressively increases across the three stages, with values of 0.815, 0.821, and 0.825, respectively. This indicates an improvement in localization. Fig. 10(b) displays the distribution of confidence

scores for the predicted boxes at each of the three stages, with scores being 0.996, 0.996, and 0.996, respectively. For cyclist and pedestrian categories, the object contains a relatively small number of points due to inherent limitations, making it difficult to detect. This leads to some proposals with low IoUs on the left side of Fig. 10(c) and Fig. 10(e). However, the model can still accurately predict most of the boxes and progressively refine the proposals to improve their localization performance. Fig. 10(d) and Fig. 10(f) illustrate the improvement in confidence scores across different stages. This demonstrates that HRNet can improve localization accuracy without reducing the confidence scores, as presented in Fig. 2(b).

4.5. Ablation study

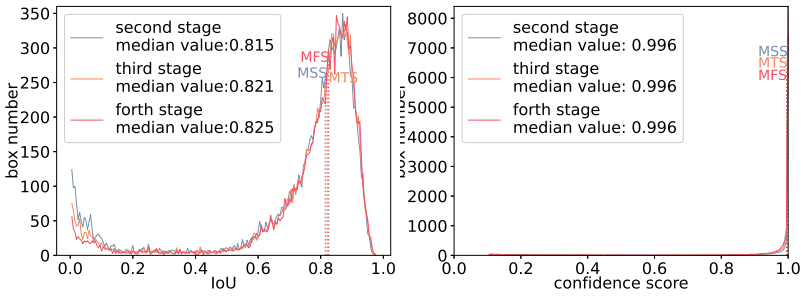
We conduct extensive experiments on the validation set of the KITTI dataset to analyze each module. We have two main modules: HR and DSS. Table 7 demonstrates the impact of each module on car detection in the proposed HRNet.

Table 7: The impact of different modules of the proposed HRNet over the KITTI validation set with AP calculated over 40 recall positions for the car category.

HR	DSS	3D AP(%)		
		Easy	Mod.	Hard
✓		92.24	85.61	83.40
	✓	92.67	85.99	83.74
✓	✓	93.45	86.48	84.07

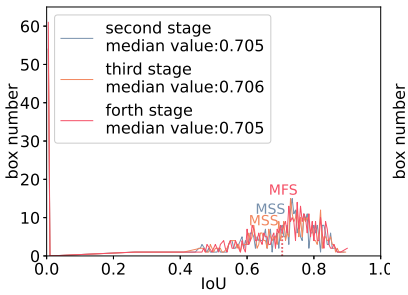
We verify the effectiveness of the proposed HR network. We use V_2, V_3 and V_4 voxel features to gradually refine the proposals, as shown in Fig. 3. We use V_4 as the first stage of our HRNet. The results are shown in Table 8. By using hierarchical refinement with multi-scale voxel features, the performance increases progressively.

In Table 9, we investigate the inference time of each stage. Due to the high efficiency of feature aggregation at a single scale, our HRNet achieves good performance without too much computational cost. Each stage accounts for approximately 10% of the total inference time.

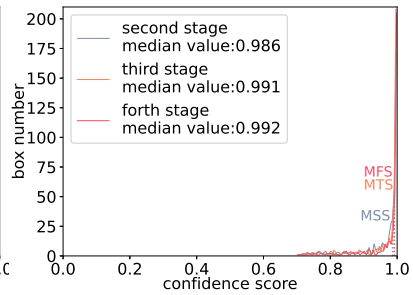


(a) Car IoU

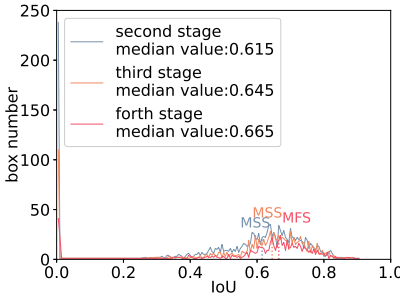
(b) Car conf



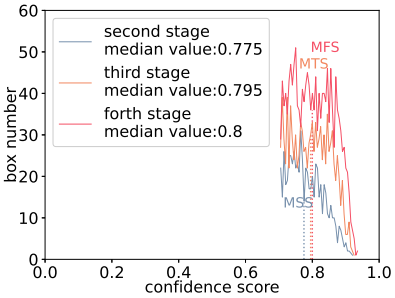
(c) Cyclist IoU



(d) Cyclist conf



(e) Pedestrian IoU



(f) Pedestrian conf

Figure 10: Distributions of IoU and confidence scores for detected boxes across different stages of refinement. MTS denotes the median value of the third stage, MFS denotes the median value of the forth stage.

In our DSS module, we used the k-nearest proposals to calculate the IoU threshold. To find the best value, we conduct experiments on k. As shown in Table 10, the results indicate that $k = 12$ achieves better results, thereby justifying our decision.

Table 8: The 3D AP of HRNet with different scales of voxel features on the KITTI validation set over 40 recall positions for the car category.

Forth stage	Third stage	Second stage	3D AP(%)			
			Easy	Mod.	Hard	mAP
✓			92.24	85.61	83.40	87.09
✓	✓		92.67	85.99	83.74	87.47
✓	✓	✓	93.45	86.48	84.07	88.00

Table 9: The impact of the different stages in the proposed HRNet on its inference time.

Second stage	Third stage	Forth stage	time(ms)
✓			146
✓	✓		165
✓	✓	✓	185

Table 10: 3D AP of HRNet on the KITTI validation set with different values of k.

k	3D AP(%)		
	Easy	Mod.	Hard
3	93.48	85.94	83.66
6	92.67	85.99	83.74
9	94.88	86.00	83.75
12	93.45	86.48	84.07
15	93.43	86.42	84.10

5. Conclusion

This paper has proposed a high-performance multi-stage detector for point clouds, which improves model performance by hierarchical refinement. We develop two novel strategies to achieve efficient multi-stage refinement. The Scale-aware Attentive Propagation was used to extract of more contextual information for localization, and the Dynamic Sample Selection was used to balance the selection of the positive and negative samples in each stage. We have conducted a series of experiments to demonstrate the effectiveness of the proposed modules. Although our method makes progress in implementing a high-performance multi-stage detector, there are still many potential research directions to explore in terms of enhancing the efficiency and performance of multi-stage detectors. For example, designing the multi-stage refinement process as an auxiliary training network, which does not participate in the inference stage, might balance accuracy and efficiency. In addition, incorporating confidence scores into the sample

selection strategy is also a promising research direction. Research in these areas could lead to more effective models and algorithms in the future for object detection tasks.

References

[1] R. Qian, X. Lai, X. Li, 3D Object Detection for Autonomous Driving: A Survey, in: Pattern Recognition, vol. 130, 2021, p. 108796.

[2] G. Shi, R. Li, C. Ma, PillarNet: Real-Time and High-Performance Pillar-Based 3D Object Detection, in: *Proc. ECCV*, 2022, vol.13670, pp. 35–52.

[3] A. H. Lang, et al. PointPillars: Fast encoders for object detection from point clouds, in: *Proc. CVPR*, 2019, pp. 12697-12705.

[4] Q. Wang, J. Chen, J. Deng, et al. 3D-CenterNet: 3D object detection network for point clouds with center estimation priority, in: Pattern Recognition, vol. 115, 2021, p. 107884.

[5] S. Shi, et al. PV-RCNN: point-Voxel feature set abstraction for 3D object detection, in: *Proc. CVPR*, 2020, pp. 10526-10535.

[6] J. Deng, et al. Voxel R-CNN: towards high performance voxel-based 3D object detection, in: *Proc. AAAI*, 2021, pp. 1201-1209.

[7] Z. Li, Y. Yao, Z. Quan, et al. Spatial information enhancement network for 3D object detection from point cloud, in: Pattern Recognition, vol. 128, 2022, p. 108684.

[8] M.H. Guo, et al. PCT: point cloud transformer, in: Comput. Vis. Media, vol. 7, no. 2, 2021, pp. 187-199.

[9] D. Wang, L. Tang, X. Wang, et al. Improving deep learning on point cloud by maximizing mutual information across layers, in: Pattern Recognition, vol. 131, 2022, p. 108892.

[10] M. Tao, C. Zhao, M. Tang, et al. Objformer: Boosting 3D object detection via instance-wise interaction, in: Pattern Recognition, vol. 146, 2024, p. 110061.

[11] Y. Yan, Y. Mao, B. Li, SECOND: Sparsely embedded convolutional detection, in: Sensors, vol. 18, 2018, p. 3337.

[12] Y. Zhou, O. Tuzel, Voxelnet: End-to-end learning for point cloud based 3d object detection, in: *Proc. CVPR*, pp. 4490-4499, Jun. 2018.

[13] Z. Yang, Y. Sun, S. Liu, et al. 3DSSD: Point-based 3D single stage object detector, in: *Proc. CVPR*, 2020, pp. 11037-11045.

[14] H. Shenga, et al. Improving 3D Object Detection with Channel-wise Transformer, in: *Proc. ICCV*, 2021, pp. 2723-2732.

[15] C. R. Qi, O. Litany, K. He, et al. Deep Hough Voting for 3D Object Detection in Point Clouds, in: *Proc. ICCV*, 2019, pp. 9277-9286.

[16] S. Shi, X. Wang, H. Li, PointRCNN: 3D Object Proposal Generation and Detection From Point Cloud, in: *Proc. CVPR*, 2019, pp. 770-779.

[17] Z. Yang, Y. Sun, S. Liu, X. Shen, J. Jia, STD: Sparse-to-dense 3D object detector for point cloud, in: *Proc. CVPR*, 2019, pp. 1951-1960.

[18] Q. He, et al. SVGA-Net: Sparse Voxel-Graph Attention Network for 3D Object Detection from Point Clouds, in: *Proc. AAAI*, 2022, pp. 870-878.

[19] H. Yang, et al. Graph R-CNN: Towards Accurate 3D Object Detection with Semantic-Decorated Local Graph, in: *Proc. ECCV*, 2022, pp. 662-679.

[20] R. Qian, X. Lai, X. Li, BADet: Boundary-Aware 3D Object Detection from Point Clouds, in: Pattern Recognition, vol. 125, 2022, p. 108524.

[21] A. Vaswani, et al. Attention is All you Need, in: *Proc. NIPS*, 2017, pp. 5998-6008.

[22] J. S. K. Hu, T. Kuai, S. L. Waslander, Point Density-Aware Voxels for LiDAR 3D Object Detection, in: *Proc. CVPR*, 2022, pp. 8469-8478.

[23] Z. Cai, N. Vasconcelos, Cascade R-CNN: Delving Into High Quality Object Detection, in: *Proc. CVPR*, 2018, pp. 6154-6162.

[24] C. R. Qi, H. Su, K. Mo, et al. Pointnet: Deep learning on point sets for 3d classification and segmentation, in: *Proc. CVPR*, 652–660.

[25] S. Zhang,C. Chi,Y. Yao, et al. Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection, in: *Proc. CVPR*, 2020, pp. 9759-9768.

[26] T. Y. Lin, P. Goyal, R. Girshick, K. He, P. Dollár, Focal loss for dense object detection, in: *Proc. ICCV*, 2017, pp. 2980-2988.

[27] A. Geiger, et al. Are we ready for autonomous driving? the KITTI vision benchmark suite, in: *Proc. CVPR*, 2012, pp. 3354-3361.

[28] P. Sun, H. Kretzschmar, X. Dotiwalla, et al. Scalability in perception for autonomous driving: Waymo open dataset, in: *Proc. CVPR*, 2020, pp. 2443–2451.

[29] H. Wu, et al. CasA: A Cascade Attention Network for 3-D Object Detection From LiDAR Point Clouds, IEEE

Trans. Geoscience and Remote Sensing, vol. 60, 2022, pp. 1-11.

[30] C. He, et al. Structure Aware Single-Stage 3D Object Detection From Point Cloud, in: *Proc. CVPR*, 2020, pp. 11873-11882.

[31] W. Zheng, et al. SE-SSD: Self-Ensembling Single-Stage Object Detector From Point Cloud, in: *Proc. CVPR*, 2021, pp. 14494-14503.

[32] W. Zheng, et al. CIA-SSD: Confident IoU-Aware Single-Stage Object Detector From Point Cloud, in: *Proc. AAAI*, 2021, pp. 3555-3562.

[33] Y. Zhang, Q. Hu, G. Xu, Y. et al. Not All Points Are Equal: Learning Highly Efficient Point-Based Detectors for 3D LiDAR Point Clouds, in: *Proc. CVPR*, 2022, pp. 18953-18962.

[34] T. Jiang, N. Song, H. Liu, R. et al. VIC-Net: Voxelization Information Compensation Network for Point Cloud 3D Object Detection, in: *Proc. ICRA*, 2021, pp. 13408-13414.

[35] J. Noh, et al. HVPR: Hybrid Voxel-Point Representation for Single-Stage 3D Object Detection, in: *Proc. CVPR*, 2021, pp. 14605-14614.

[36] D. Zhou, J. Fang, X. Song, et al. IoU Loss for 2D/3D Object Detection, in: *Proc. 3DV*, 2019, pp. 85-94.

[37] L. Du, et al. Associate-3Ddet: Perceptual-to-Conceptual Association for 3D Point Cloud Object Detection, in: *Proc. CVPR*, 2020, pp. 13329-13338.

[38] Y. Chen, et al. Fast Point R-CNN, in: *Proc. ICCV*, 2019, pp. 9775-9784.

[39] J. Li, H. Dai, L. Shao, Y. Ding, From Voxel to Point: IoU-guided 3D Object Detection for Point Cloud with Voxel-to-Point Decoder, in: *Proc. ACM*, 2021, pp. 4622-4631.

[40] S. Shi, et al. From Points to Parts: 3D Object Detection From Point Cloud With Part-Aware and Part-Aggregation Network, in: IEEE Trans. PAMI, vol. 43, no. 8, 2020, pp. 2647-2664.

[41] Y. Chen, et al. Focal Sparse Convolutional Networks for 3D Object Detection, in: *Proc. CVPR*, 2022, pp. 5428-5437.

[42] Q. Cai, et al. 3D Cascade RCNN: High Quality Object Detection in Point Clouds, in: IEEE TIP, vol. 31 , 2022, pp. 5706-5719.

[43] Y. Zhou, et al. End-to-End Multi-View Fusion for 3D Object Detection in LiDAR Point Clouds, in: *Proc. CoRL*, 2020, pp. 923-932.

[44] J. Mao, et al. Voxel Transformer for 3D Object Detection, in: *Proc. ICCV*, 2021, pp. 3164-3173.