*Computer Systems Science & Engineering*
DOI: 10.32604/csse.2023.034230
***Article***

Tech Science Press

# 3D Object Detection with Attention: Shell-Based Modeling

**Xiaorui Zhang[1,2,3,4,*], Ziquan Zhao[1], Wei Sun[4,5] and Qi Cui[6]**

[1]School of Computer and Software, Nanjing University of Information Science & Technology, Nanjing, 210044, China
[2]Wuxi Research Institute, Nanjing University of Information Science & Technology, Wuxi, 214100, China
[3]Engineering Research Center of Digital Forensics, Ministry of Education, Jiangsu Engineering Center of Network Monitoring, Nanjing, 210044, China
[4]Jiangsu Collaborative Innovation Center of Atmospheric Environment and Equipment Technology (CICAEET), Nanjing University of Information Science & Technology, Nanjing, 210044, China
[5]School of Automation, Nanjing University of Information Science & Technology, Nanjing, 210044, China
[6]Department of Electrical and Computer Engineering, University of Windsor, Windsor, N9B 3P4, Canada
*Corresponding Author: Xiaorui Zhang. Email: zxr365@126.com
Received: 10 July 2022; Accepted: 10 October 2022

**Abstract:** LIDAR point cloud-based 3D object detection aims to sense the surrounding environment by anchoring objects with the Bounding Box (BBox). However, under the three-dimensional space of autonomous driving scenes, the previous object detection methods, due to the pre-processing of the original LIDAR point cloud into voxels or pillars, lose the coordinate information of the original point cloud, slow detection speed, and gain inaccurate bounding box positioning. To address the issues above, this study proposes a new two-stage network structure to extract point cloud features directly by PointNet++, which effectively preserves the original point cloud coordinate information. To improve the detection accuracy, a shell-based modeling method is proposed. It roughly determines which spherical shell the coordinates belong to. Then, the results are refined to ground truth, thereby narrowing the localization range and improving the detection accuracy. To improve the recall of 3D object detection with bounding boxes, this paper designs a self-attention module for 3D object detection with a skip connection structure. Some of these features are highlighted by weighting them on the feature dimensions. After training, it makes the feature weights that are favorable for object detection get larger. Thus, the extracted features are more adapted to the object detection task. Extensive comparison experiments and ablation experiments conducted on the KITTI dataset verify the effectiveness of our proposed method in improving recall and precision.

**Keywords:** 3D object detection; autonomous driving; point cloud; shell-based modeling; self-attention mechanism

## 1 Introduction

3D object detection technology based on LIDAR point cloud data is increasingly used in autonomous driving and mobile robotics. The large-scale implementation of artificial intelligence technology provides the potential for L2-L5 level autonomous driving. 3D object detection takes LIDAR point cloud data as input, which provides detailed geometric and semantic information about objects in 3D space. It is crucial to extract effective features and apply the extracted features for 3D object detection.

Early 3D object detection methods, such as VoxelNet [1], PV-RCNN [2], and PointPillar [3], predict object centers by regression calculation, which gradually approximates the ground truth of the extracted features. However, the challenges in autonomous driving scenes can be summarized as follows: due to the large distribution span of point clouds, taking the KITTI [4,5] dataset as an example, a single object is about 1 to 5 meters, the angular resolution is 0.09°, and the accuracy is 2 cm. While the object detection is based on LIDAR point cloud data with a wide search range, therefore, the stepwise approximation based on regression is relatively slow to calculate the true coordinates. It often takes many epochs of regression to obtain the geometric center, and it is also difficult to achieve high recall and accuracy [6]. In addition, effective features are crucial for object detection tasks. The 3D object detection methods, typically such as Voxelnet [1] and PointPillar [3], often require the original point cloud is firstly divided into structured forms such as voxels. For example, Voxelnet [1] divides the original point cloud into voxels firstly, and PointPillar seeks to repackage the original point cloud data into several pillars. They extracted features from processed voxels and pillars rather than from the original point cloud, thereby losing the coordinate information of the original point cloud. In addition to the manner of extracting features, the effective selection of the extracted features also affects the effectiveness of the 3D object detection. Self-attention assignment in feature dimensions allows feature selection and its application in 2D image segmentation, which effectively improves the precision of object detection [7] as well as the recall and precision of image recognition [8]. Although the self-attention mechanism has achieved impressive results in various 2D computer vision tasks, it is difficult to accommodate 3D point cloud object detection due to different modalities between 2D and 3D data.

To address the above issues, this study designs a shell-based modeling method. The localization range of the object is divided into several spherical shell intervals firstly, accordingly, the object detection task is formulated as a classification problem of which interval the object belongs to. Then a fine-grained regression is used to obtain the exact object coordinates after a suitable interval has been classified. To deal with the problem of geometric information loss of 3D object detection methods in extracting the original point cloud features, our proposed network framework uses PointNet++ [9] to extract the original point cloud features, in which a plug-and-play point self-attention module is designed to bridge the gap between the 2D vision algorithms and 3D object detection. Extensive experiments are conducted to illustrate the effectiveness of the self-attention module for 3D object detection in terms of recall and precision.

Our contributions can be summarized as follows.

- Using the strategy of classification followed by regression, a shell-based loss model is designed for regression of the object parameters, which accurately and efficiently classifies the object centers into certain shells to narrow the regression range in the input original point cloud. Moreover, further regression is calculated to obtain the geometric centers.
- The proposed method subtly adds a point-attention module for 3D object detection between the encoder and decoder. These features favorable for object detection are highlighted by assigning larger weights to them, thereby improving the recall rate of object detection.
- This method extracts features from the original 3D point cloud by PointNet++ and generates proposals directly without excessive processing for the original point cloud, which can retain the coordinate information of the original point cloud and obtain richer features.

The remainder of this paper is organized as follows: Section 2 presents the related work with attentional mechanism and object detection; Section 3 explains the principle of shell-based 3D object detection with attention and details the 3D self-attention mechanism; Section 4 describes the specific implementation of this method and extensive experiments; Section 5 discusses the effectiveness of the proposed method and concludes the work.

## 2 Related Work

In this section, we first review the work related to attention mechanisms in solving other challenges, such as Natural Language Processing (NLP), 2D image recognition, and object detection, which is a key module in our framework. Then, we review other modeling methods for 3D object detection, where we have made important innovations.

### *2.1 Attention Mechanism*

Attention Mechanisms have been a hot topic in the field of machine learning for years and have achieved impressive results in the field of NLP and 2D image processing with improved accuracy and recall.

In June 2017, Vaswani et al. [10] gave a solution idea of using attention mechanism in NLP, and self-attention mechanism started to become a hot research topic of attention neural network. However, the attention mechanism is still only in the research of NLP.

In 2018, Tan et al. [11] from Xiamen University proposed a deep attentional neural network for semantic segmentation and semantic role labeling. In the same year, Woo et al. [12] proposed a Convolutional Block Attention Module for 2D image classification. In 2022, Yan et al. [13] proposed to capture the contextual information of 2D images from the sequence of facial features. These features are extracted by the backbone network using multi-headed attention to improve the classification performance of the network. All methods above significantly improved the accuracy of the baseline and achieved better results in terms of accuracy and recall. However, the above methods are only for the segmentation and classification of 2D images.

In 2020, Huang et al. [14] proposed an efficient and fine-grained attention mechanism to enhance the accuracy of a 2D object detector. In the same year, Cao et al. [15] designed the attention-guided module (AM). This module adaptively captures significant object dependencies using an attention mechanism. Many experiments on 2D object detection and instance segmentation showed that the accuracy of the existing AM-based model was significantly better than that of the model without AM. However, the above approaches with attention mechanisms only focus on the 2D object detection rather than on the 3D object detection.

Therefore, in this work, we design a point-attention module adapted to 3D object detection between the encoder and decoder and creatively introduce the skip connection structure [16], where the network is back-propagating if self-attention has no positive effect on the network. The skip connection structure becomes the back-propagation “shortcut”, thus blocking out such self-attention modules that have no positive incentive. This enables the network to robustly add deeper self-attention modules, avoiding the reduction of network adaptability due to too many network parameters and effectively extracting features that are more adapted to the 3D object detection.

### *2.2 Object Detection*

The object detection task aims to locate objects [17–19]. As far as localization is concerned, 3D objects can be modeled in a variety of ways.

In 2018, Zhou et al. proposed VoxelNet [1], in 2021 Wang et al. proposed FCOS3D [20], and Deng et al. proposed Voxel R-CNN [21], which modeled the 3D BBox as a 7-dimensional vector representation. These methods use 7 variables to perform regression training by smooth L1 loss. However, these methods have disadvantages such as low prediction accuracy, slow convergence of the network, and too much GPU resource occupied.

In 2022, Shi et al. [22] proposed Center-based 3d object detection, in which objects are modeled as vertex coordinates and centroid coordinates, and direct regression is performed on the down-sampled offsets of every vertex. Vertex offsets are used in a direct regression on L1 loss.

In 2018, Yan et al. proposed SECOND, where the most important innovation is the re-modeling of the object orientation estimation [23]. In this work, this significant innovation in classifying the continuous orientation parameters inspired our shell-based modeling of classification followed by regression in this work.

The study adopts the idea of classification followed by regression and designs a shell-based model: firstly, the parameters of the object are classified as belonging to a shell, and then the residual is regressed in the obtained shell.

## 3 3D Object Detection with Attention Mechanism

In this section, we introduce the proposed framework for 3D object detection with attention mechanism. Specifically, we will introduce the feature extraction and semantic segmentation using PointNet++ in Section 3.1. Then describe the details of the point-wise semantic features obtained through the self-attention mechanism for 3D point cloud in Section 3.2, and the shell-based modeling to generate proposals and BBox in Section 3.3. The specific system framework is shown in Fig. 1. Using the strategy of classification first and then regression, the shell-based loss model is designed to replace the previous method.

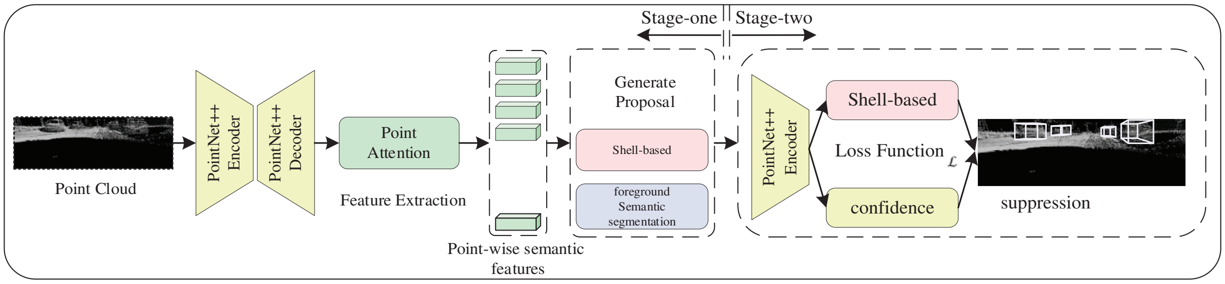

**Figure 1:** 3D object detection with attention mechanism

### *3.1 Feature Extraction and Semantic Segmentation*

Input the original point cloud directly into the PointNet++ feature extractor. The proposed features are in one-to-one correspondence with the input points, called point-wise extraction. Point-wise extraction of point cloud features allows the original geometric information and semantic information to be preserved [24].

The extracted semantic features contain rich contextual information. The segmentation of the original point cloud into foreground points is beneficial to improve the accuracy of 3D BBox by adding semantic information before the prediction of 3D BBox. In this work, the semantic features serve two purposes. On the one hand, the semantic segmentation can divide the foreground points and the background points. In the 3D object detection dataset, the labels of semantic segmentation can be generated by Ground-truth. On the other hand, the network helps the understanding by adding semantic information, such as environmental information like car wheels.

Unlike 2D semantic segmentation, which tends to be ambiguous between object-critical pixels, 3D semantic segmentation can provide accurate semantic information more effectively due to the loose nature of point clouds discrete from each other. Hence, 3D semantic segmentation outperforms 2D semantic segmentation.

Specifically, the point cloud is fed to the backbone network, and point-wise features are obtained. Then, the foreground point probability score is obtained by sigmoid to separate the foreground points. In the point cloud distribution in outdoor scenes such as autonomous driving, the number of foreground points of interest and background points is often unbalanced, and to solve this problem, the focal loss function [25] is used in this paper, as shown in Eq. (1).

$$\mathcal{L}_{local}(P) = -\alpha(1-P)^{\gamma}\log(P) \tag{1}$$

$$P = \begin{cases} P & , Positive\ Point \\ 1-P & , Negative\ Point \end{cases}$$

where $\alpha$ and $\gamma$ denote hyperparameters. Multiplying by the hyperparameter $\alpha$ can solve the imbalance between the foreground and the background point sample. The focal loss parameters of the sample balance retain the settings of $\alpha = 0.25$ and $\gamma = 2$. $P$ is the predicted probability of foreground or background points when semantically segmenting.

### *3.2 3D Point Cloud Self-Attention Mechanism*

Original point cloud is encoded and decoded by PointNet++. Then the obtained vector $X$ is a rich feature vectors, but these features have no focus, so they cannot be well applied to object detection, therefore, we designed the 3D point cloud self-attention module, as shown in Fig. 2. Using the 3D point cloud self-attention module, it is possible to continuously speculate on which local point should be focused on and to continuously increase the focus on the key region when extracting information. Prediction of the regions and channels that should be focused on makes the predicted proposal more accurate and more conducive to improving the accuracy of object detection.

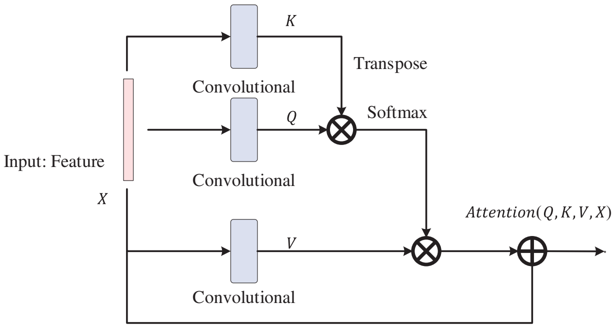

**Figure 2:** Self-attention with skip connection module

The vector $X$ goes through three convolutional layers to obtain the query vector ($Q$), the key vector ($K$), and the content vector ($V$), respectively [10]. The skip connection structure keeps the vector $X$ unchanged to compensate the output from the attention module. The stacking of attention modules will undoubtedly deepen the structure of the network, and the resulting possible gradient explosion problem is not conducive to the robustness of the network [26]. Therefore, in this paper, this skip connection is added to the design of the self-attention module [23]. The self-attentive mechanism with a skip connection structure proposed in this work can be expressed by Eq. (2).

$$Attention(Q,\ K,\ V,\ X) = softmax\left(\frac{QK^T}{\sqrt{d_K}}\right)V + X \qquad (2)$$

where, $Q$, $K$, and $V$ denote the query vector, the vector to be matched, and the content vector, respectively. $d_k$ denotes the dimension of vector $K$, and vector $X$ denotes the input vector of the module. $K^T$ represents the transposition of $K$. The similarity score is calculated by SoftMax function, and then the similarity score is multiplied by the content vector to obtain the weighted value of the content vector. These three vectors are learned through their respective convolutional layers, and the $Q$, $K$, and $V$ that best characterize the object are obtained by back-propagation to enhance the network's capability [27].

### *3.3 Shell-Based Modeling*

To solve the problem that the distribution of point cloud data is large, and the direct regression leads to low object detection accuracy, we propose shell-based modeling, which introduces the modeling process through spherical space division, transformation of spherical and polar coordinates, and shell-based loss definition respectively. For a regression problem, we first divide the true value range into several intervals, so that the regression problem becomes a classification problem of "which interval to belong to", and then use one-hot encoding and cross-entropy loss function to regress to a correct interval, and then use smooth L1 loss parametrization to do fine-grained regression.

#### *3.3.1 Spherical Space Division*

As shown in Fig. 3, the shell-based model for shell classification shows a spherical shape overall. The closer the object is to the center coordinates of the sphere, the smaller the classification shell is. Therefore, the regression range of the shell is smaller, and the regression is finer. Conversely, the farther the object is from the center coordinates of the sphere, the larger the classification shell is. Therefore, the regression range of the shell is larger, and the regression is coarser.

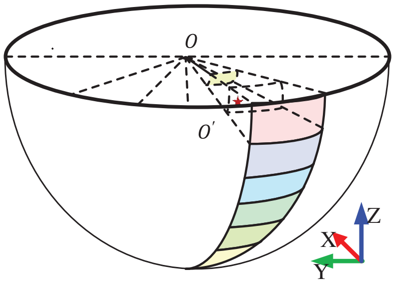

**Figure 3:** Shell-based classification regression

So, the division is as follows, the former point cloud $O$ is the center of the spherical space, and the spherical space is divided equally along the radius $R$ of the regression with $\hat{r}$ as the smallest unit, at which time the spherical space is presented as $R/\hat{r}$ shells. The zenith angle of the spherical space is uniformly distributed with $\hat{\varphi}$ as the smallest unit. The direction angle of the spherical space is uniformly distributed with $\hat{\omega}$ as the smallest unit.

#### *3.3.2 Coordinate System Conversion*

Our shell-based spherical modeling is easier in the spherical coordinate system. However, the labels of the original KITTI dataset are collected in the Cartesian coordinate system, so the original dataset needs to be pre-processed and converted to spherical coordinates before performing network training. Before drawing the bounding box by getting the prediction results, we also need to convert the spherical coordinates to

Cartesian coordinates since the regression is obtained in spherical coordinates. The mapping relations are shown in Eqs. (3) and (4).

In the spherical coordinate system, the spherical coordinates of any foreground point P $(x_O,\ y_O,\ z_O)$ are denoted as P $(r_O,\ \omega_O,\ \varphi_O)$, and the mapping relationship between them is denoted as $\mathcal{F}$, which is expressed by Eq. (3).

$$\begin{cases} x_O = r_O \sin \omega_O \cos \varphi_O \\ y_O = r_O \sin \omega_O \sin \varphi_O \\ z_O = r_O \cos \omega_O \end{cases} \tag{3}$$

The mapping relationship from Cartesian to the spherical coordinate system is denoted as $\vec{\mathcal{F}}$ and is expressed through the following Eq. (4).

$$\begin{cases} r_O = \sqrt{x_O^2 + y_O^2 + z_O^2} \\ \omega_O = arccos \dfrac{z_O}{r_O} \\ \varphi_O = arctan \dfrac{y_O}{x_O} \end{cases} \tag{4}$$

where $\varphi_O$ denotes the orientation angle of the foreground point cloud $O$ in spherical coordinates, $\omega_O$ denotes the zenith angle of the foreground point cloud $O$ in the spherical coordinate system, and $r_O$ denotes the radius of the foreground point cloud $O$ in spherical coordinates. $x_O$, $y_O$, and $z_O$ represent the coordinates of point $O$ in the cartesian coordinate system. As shown in Fig. 3, For the divided spherical regression space, as different classification categories, in the designed deep neural network, after calculation, back propagation, classify which divided shell the geometric center $O'$ is belonged to. Next, through the regression method, $r$, $\varphi$, $\omega$ are calculated. Then, through the mapping relation $\mathcal{F}$, the $x$, $y$, $z$ coordinates of the corresponding geometric center $O'$ under the Cartesian coordinate system are calculated $(x_{O'},\ y_{O'},\ z_{O'})$.

### 3.3.3 Shell-Based Loss

The shell-based loss is designed as follows: optimize the regression problem of the original regression center coordinates. We optimize it as a classification problem of "which interval to belong to". Then, fine-grained regression is performed to obtain the centroid coordinates. Therefore, the shell-based loss consists of two parts, the first part is the shell interval classification loss, and the second part is the coordinate fine regression loss. This can solve the problem of using direct regression modeling in a 3D space, which leads to low accuracy due to the large span of data distribution.

The first part is the classification loss, through the constraint of the classification loss, which shell the center coordinates belong to can be obtained, and the classification $\underset{u \in \{x,\ y,\ z\}}{\mathrm{Shell}_u^{(O')}}$ of the shell to which the geometric center $O'$ belongs, as shown in Eq. (5).

$$\underset{u \in \{x,\ y,\ z\},\ \sigma \in \{\hat{r},\ \hat{\varphi},\ \hat{\omega}\}}{\mathrm{Shell}_u^{(O')}} = \left\lfloor \frac{\vec{\mathcal{F}}_u^{O'} - \vec{\mathcal{F}}_u^{(O')} + \hat{D}}{\sigma} \right\rfloor \tag{5}$$

where $\hat{r}$ is the radius gain unit size of shell, $\hat{\varphi}$ is the directional angle gain unit size of shell, $\hat{\omega}$ is the zenith angle directional gain unit size of shell, $\vec{\mathcal{F}}_u^{O'}$ is the result after mapping the predicted value of its corresponding object's geometric center $O'$ coordinates, $\vec{\mathcal{F}}_u^{(O')}$ is the result after mapping the true value of the $O'$ coordinate of the geometric center of its corresponding object, $\hat{D}$ denotes the corresponding search range search domain so that regression around the domain, $u$ generalizes the $x$, $y$ and $z$ coordinates, and $\sigma$ generalizes $\hat{r}$, $\hat{\varphi}$, and $\hat{\omega}$.

The second part is the regression loss, in the shell obtained by classification, further position refinement in the residuals along the $x$, $y$ and $z$ axes with the true values $\underset{u \in \{x,\ y,\ z\}}{res_u^{(O')}}$, as shown in Eq. (6).

$$\underset{u \in \{x,\ y,\ z\},\ \sigma \in \{\hat{r},\ \hat{\varphi},\ \hat{\omega}\}}{res_u^{(O')}} = \vec{\mathcal{F}}_u^{O'} - \vec{\mathcal{F}}_u^{(O')} + \hat{D} - \left( Shell_u^{(O')} * \sigma + \frac{\sigma}{2} \right) \tag{6}$$

where $Shell_x^{(O')}$, $Shell_y^{(O')}$ and $Shell_z^{(O')}$ are the conversion values of the true values of the geometric center $O'$ coordinates $(x,\ \ y,\ \ z)$ in the shell division, and $R$ denotes the regression radius of the spherical regression space.

From the above two components of losses, the total loss $\mathcal{L}_{Shell}$ for shell-based modeling can be calculated as shown in Eq. (7).

$$\mathcal{L}_{Shell} = \sum_{u \in \{x,y,z\}} \left( \mathcal{F}_{cls} \left( \overset{\wedge}{shell}{}_u^{(O')},\ shell_u^{(O')} \right) + \mathcal{F}_{reg} \left( \overset{\wedge}{\mathrm{res}}{}_u^{(O')},\ res_u^{(O')} \right) \right) \tag{7}$$

where $\mathcal{F}_{cls}$ is the binary cross-entropy loss, $\overset{\wedge}{shell}{}_u^{(O')}$ denotes the true value of which shell the $O'$ coordinate of the geometric center belongs to, $shell_u^{(O')}$ is the predicted value of which shell the $O'$ coordinate of the geometric center belongs to, $\mathcal{F}_{reg}$ denotes the smooth L1 loss, $\overset{\wedge}{\mathrm{res}}{}_u^{(O')}$ denotes the true value of the residual part, and $res_u^{(O')}$ denotes the predicted value of the residual part.

## 4 Experiments and Discussion

We evaluate the performance of the different modules on the KITTI dataset, a realistic autonomous driving scenario using LIDAR and camera acquisition.

In Section 4.1, the experimental details are presented; in Section 4.2, a comparison experiment with existing 3D object detection methods is launched; and in Section 4.3, an ablation experiment is conducted to demonstrate the impact of the attention module on the object detection accuracy.

### *4.1 Experimental Details*

Dataset: The KITTI dataset is an important dataset for evaluating computer vision algorithms in autonomous driving scenarios. In this dataset, there are 7,481 training images and 7,518 test images and the corresponding point cloud data, where the easy sample accounts for 15%, the moderate sample accounts for 30%, and the hard sample accounts for 50% [4,5]. According to the camera parameters provided, the data from one LIDAR point cloud acquisition, the color images from two RGB color cameras, and the grayscale images from one grayscale camera are synchronized under the same coordinate system.

Experimental platform: The implementations were implemented on a tower server with Intel(R) Xeon(R) Gold 6126 CPU @ 2.60 GHz CPU, 256G RAM, 3.6 TB SATA, and four NVIDIA GeForce 3090 GPUs. The implementations were implemented in a software environment with CentOS Linux release 7.9.2009 (Core), CUDA 11.4, Pytorch 1.9.0, and Python 3.7, all using the same hardware and software platform.

Specific training parameter settings: A total of 200 training epochs, a batch size of 16, and an initial learning rate of 0.02 were used. 7481 training samples were input, and 7518 test samples were used for evaluation according to the default configuration of the KITTI dataset.

Evaluation metrics: Using only LIDAR point cloud data as input, we tested the average precision (AP) and recall for the category of cars, pedestrians with differentiation, and the overall metrics of average mean precision (mAP) and average recall (AR). Accuracy can algorithmically detect accuracy and recall can evaluate the mining ability of the proposal, which is shown in more detail in Eqs. (8) and (9) [28].

$$Pre = \frac{TP}{TP + FP} \times 100\% \tag{8}$$

$$Rec = \frac{TP}{TP + FN} \times 100\% \tag{9}$$

where P (positive) and N (negative) denote the results of the model, and T (True) and F (False) denote to evaluate whether the results of the model are correct or not. Precision denotes the proportion of the number of true case samples to the samples predicted by the model as positive cases (retrieved positive case samples). Recall denotes the number of true case samples and represents the number of positive cases in the original data set.

### *4.2 Comparison Experiments*

For the sub-network in the first stage, all points within the 3D ground-truth box are considered foreground points, and other points are considered as background points. During the training process, we ignore the background points near the object bounding by appropriately zooming the 3D ground-truth box by 0.1 m on every side of the object for robust segmentation, since the 3D ground-truth box may have small variations. For shell-based proposal generation, the hyperparameters are set as follows: search domain $\hat{D} = 3$ m, shell size $\delta = 0.5$ m.

To train the sub-network in the second stage, we randomly add 3D proposals with small variations to increase the diversity of proposals. For the training of the proposal, if the maximum 3D IoU (with ground truth box) of the proposal is higher than 0.6, the proposal is considered a positive proposal, and if the maximum 3D IoU of a proposal is lower than 0.45, the proposal is considered as negative. We use 3D IoU = 0.55 as the minimum threshold for training proposals with BBox regression head. For shell-based proposal refinement, the search range search domain $\hat{D} = 1.5$ m, the radius gain size of the shell is $\Delta r =$ 0.5 m, the directional angle gain unit size is $\varphi = 30°$, and the zenith angle directional gain size is $\omega = 30\circ$. The focal loss parameters of the sample balance retain the default settings of $\alpha = 0.25$ and $\gamma = 2$.

We examine the effect of the 3D object detection algorithm on the same experimental platform and conduct comparative experiments on the current mainstream 3D object detection framework, and the experimental results are shown in Table 1.

Table 1 shows the performance comparison on the KITTI test set. The method compares the optimal results under all conditions in the table. The font of the optimal results is bold. The result is evaluated by the mAP with 40 recall positions. From the experimental results, our detection results are more accurate than previous advanced algorithms, and for the hard sample of pedestrians, our detection framework is more advantageous with a 1.54% improvement.

The performance of this 3D object detection is compared with previous methods on the same experimental platform using KITTI's validation set, as shown in Table 2. The evaluation metric is the average accuracy (AP) with an IoU threshold of 0.7 for cars and 0.5 for pedestrians/bicycles. From the experimental results, the object detection using a shell-based model has higher recall than previous methods based on regression alone. The superiority of our method was proved by comparative experiments.

**Table 1:** Performance comparison on the KITTI test set

| Method | Modality | Cars | | | Pedestrians | | | Cyclists | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Easy | Mod. | Hard | Easy | Mod. | Hard | Easy | Mod. | Hard |
| F-ConvNet [29] | RGB + LIDAR | 87.36 | 76.39 | 66.69 | 52.16 | 43.38 | 38.80 | 81.98 | **65.07** | 56.54 |
| F-PointNet [30] | RGB + LIDAR | 82.19 | 69.79 | 60.59 | 50.53 | 42.15 | 38.08 | 72.27 | 56.12 | 49.01 |
| AVOD-FPN [31] | RGB + LIDAR | 83.07 | 71.76 | 65.73 | 50.46 | 42.27 | 39.04 | 63.76 | 50.55 | 49.93 |
| MV3D [32] | RGB + LIDAR | 74.97 | 63.63 | 54.00 | - | - | - | - | - | - |
| PointPainting [33] | RGB + LIDAR | 82.11 | 71.70 | 67.08 | 50.32 | 40.97 | 37.87 | 77.63 | 63.78 | 55.89 |
| ContFuse [34] | RGB + LIDAR | 83.68 | 68.78 | 61.67 | - | - | - | - | - | - |
| 3D IoU Loss [35] | LIDAR only | 81.16 | 75.50 | 71.39 | - | - | - | - | - | - |
| STD [36] | LIDAR only | 87.95 | **79.71** | 75.09 | 53.29 | 42.47 | 38.55 | 78.69 | 61.59 | 55.30 |
| PointPillars [3] | LIDAR only | 82.58 | 74.31 | 68.99 | 51.45 | 41.92 | 38.89 | 77.10 | 58.65 | 51.92 |
| Fast Point RCNN [37] | LIDAR only | 85.29 | 77.40 | 70.24 | - | - | - | - | - | - |
| PointRCNN [24] | LIDAR only | 86.96 | 75.64 | 70.70 | 47.98 | 39.37 | 36.01 | 74.96 | 58.82 | 52.53 |
| 3D SSD [38] | LIDAR only | **88.36** | 79.57 | **74.55** | **54.64** | **44.27** | **40.23** | **82.48** | 64.10 | 56.90 |
| Ours | LIDAR only | 88.72 | 79.66 | 74.63 | 55.47 | 45.23 | 41.77 | 83.02 | 65.51 | 56.26 |
| Improvement | - | 0.36 | - | 0.08 | 0.83 | 0.96 | 1.54 | 0.54 | 0.44 | - |

**Table 2:** Recall comparison

| Network | Car (IoU = 0.7) recall (%) | | | Pedestrian (IoU = 0.5) recall (%) | | | Cyclists (IoU = 0.5) recall (%) | | |
|---|---|---|---|---|---|---|---|---|---|
| | Easy | Moderate | Hard | Easy | Moderate | Hard | Easy | Moderate | Hard |
| VoxelNet | 77.47 | 65.11 | 57.73 | 39.48 | 33.69 | 31.51 | 61.22 | 48.36 | 44.37 |
| SECOND | 83.13 | 73.66 | 66.20 | 51.07 | 42.56 | 37.29 | 70.51 | 53.85 | 46.90 |
| PointRCNN | 85.94 | 75.76 | 68.32 | 49.43 | 41.78 | 38.63 | 73.93 | 59.60 | 53.59 |
| Ours | 86.55 | 76.23 | 67.86 | 52.07 | 42.85 | 39.29 | 75.46 | 61.78 | 54.59 |

### *4.3 Ablation Experiments*

The impact of the attention module on the mAP and recall of the object detection framework is verified separately compared to the baseline without any attention module added.

Table 3 shows the performance of skip connection structure in self-attention. The result is evaluated by the mAP on the KITTI test set. From the experimental results, self-attention with the added skip connection structure can be added more when the corresponding module of the attention mechanism is added, making the network pay more attention to the desired features and further improving the object detection accuracy. However, when the self-attention module of the skip connection structure is not added, the object detection accuracy decreases as expected when the number is increased to a certain number.

Table 4 shows the effect of self-attention on recall, counting the recall at epochs 50, 150, and 200 for the cars tested. The experiment shows that the recall result of adding three self-attentions at epoch 50 is already close to the recall of epoch 200. However, the baseline comparison experimental group continues to converge until epoch 150, and recall has room for improvement. This shows that the self-attention module can accelerate the convergence of the network; in addition, the recall of the comparison group with the addition of self-attention module achieves the optimal results when the IoU is 0.1, 0.3, and 0.5 corresponding to 95.4%, 93.8%, and 92.0%, respectively. The effectiveness of our proposed method was proved by ablation experiments.

**Table 3:** Performance of skip connection structure in self-attention

| Method | Num. | Cars | | | Pedestrians | | | Cyclists | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Easy | Mod. | Hard | Easy | Mod. | Hard | Easy | Mod. | Hard |
| Self-attention | 1 | 87.36 | 76.39 | 66.69 | 52.16 | 43.38 | 38.80 | 81.98 | 65.07 | 56.54 |
| | 2 | 87.68 | 78.78 | 66.67 | 50.53 | 42.15 | 38.08 | 72.27 | 56.12 | 49.01 |
| | 4 | 87.07 | 77.76 | 65.73 | 50.46 | 42.27 | 39.04 | 63.76 | 50.55 | 49.93 |
| | 8 | 83.07 | 71.76 | 65.73 | - | - | - | - | - | - |
| | 16 | 82.11 | 71.70 | 67.08 | 50.32 | 40.97 | 37.87 | 77.63 | 63.78 | 55.89 |
| | 32 | 77.68 | 68.78 | 61.67 | - | - | - | - | - | - |
| Self-attention with skip connection | 1 | 81.16 | 75.50 | 71.39 | - | - | - | - | - | - |
| | 2 | 87.95 | 79.71 | 75.09 | 53.29 | 42.47 | 38.55 | 78.69 | 61.59 | 55.30 |
| | 4 | 82.58 | 74.31 | 68.99 | 51.45 | 41.92 | 38.89 | 77.10 | 58.65 | 51.92 |
| | 8 | 85.29 | 77.40 | 70.24 | - | - | - | - | - | - |
| | 16 | 86.96 | 75.64 | 70.70 | 47.98 | 39.37 | 36.01 | 74.96 | 58.82 | 52.53 |
| | 32 | 88.72 | 79.26 | 72.63 | 54.64 | 44.27 | 40.23 | 82.48 | 64.10 | 56.90 |

**Table 4:** Effect of attention mechanism on recall

| Network | Epoch50 Recall (%) | | | Epoch150 Recall (%) | | | Epoch200 Recall (%) | | |
|---|---|---|---|---|---|---|---|---|---|
| IoU threshold | 0.1 | 0.3 | 0.5 | 0.1 | 0.3 | 0.5 | 0.1 | 0.3 | 0.5 |
| Baseline | 90.2 | 85.7 | 62.4 | 93.2 | 92.5 | 90.2 | 94.2 | 93.1 | 91.2 |
| Baseline + attention | 91.1 | 90.6 | 83.5 | 94.2 | 93.3 | 91.4 | 95.1 | 93.5 | 91.8 |
| Baseline + attention*2 | 92.5 | 91.6 | 89.3 | 94.7 | 93.6 | 91.7 | 95.3 | 93.7 | 91.8 |
| Baseline + attention*3 | 93.7 | 92.2 | 90.5 | 95.2 | 93.9 | 91.8 | 95.4 | 93.8 | 92.0 |

The green box indicates the labeled information, and the red box indicates the test result. As shown in Fig. 4, this network uses the 3D raw point cloud as input, and for the heavily occluded region, this object detection framework can still detect accurately, and the discrete shape of the 3D point cloud in the depth direction is incomparable to the 2D image detection. In addition, many hard-to-detect objects are not labeled in the KITTI dataset because of the long distances. Thanks to the self-attention module, our detection framework can detect these unlabeled samples, showing the superiority of the present object detection framework.

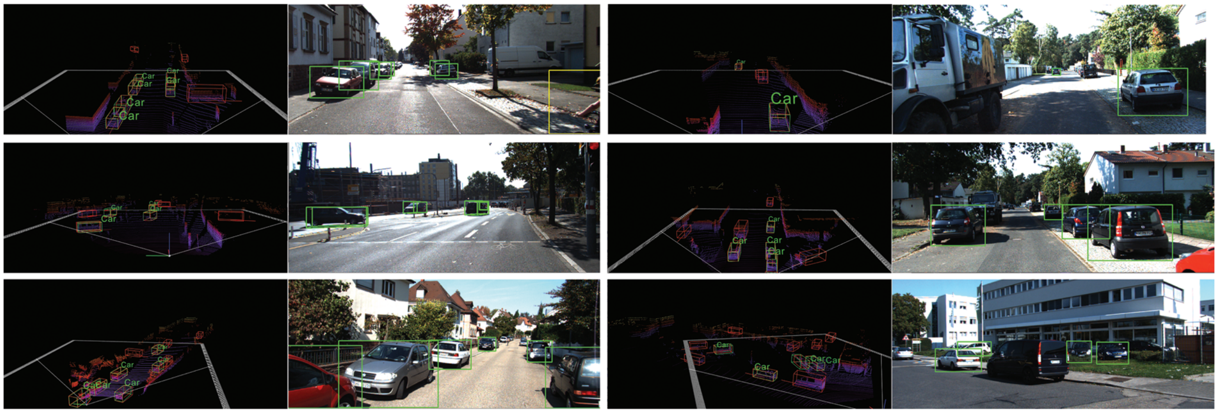

**Figure 4:** Visualization results on the KITTI test set

## 5 Conclusion

We propose a 3D object detection framework with attention mechanism, which is a new 3D object detector from the original point cloud. The stage-one sub-network uses PointNet++ to extract the original point cloud features directly from the point cloud to generate 3D proposals. After adding the attention mechanism, it makes the network learn more beneficial features for object detection, resulting in higher recall rates. The second stage network refines the proposal in the spherical coordinate system by combining semantic features and local spatial features. In addition, the proposed shell-based model first roughly classifies which shell the coordinates belong to, and then refines the regression localization coordinates, narrows the localization range, and improves the detection accuracy, proving its effects for 3D bounding box regression. The effectiveness of this model for 3D bounding box regression is demonstrated. Experiments show that the performance of the framework proposed in this work outperforms a group of previous state-of-the-art methods by a significant margin on the challenging 3D detection baseline of the KITTI dataset.

This proposed framework only uses the original point cloud collected by LIDAR as input, even though the original point cloud has many excellent features, it also naturally lacks texture and color information. In future, we will further explore the integration of image features into the object detection framework. The introduction of multimodal inputs, including LiDAR and image data, can enrich the perceptual features. It is the direction of our efforts to study efficient algorithms that integrate two-dimensional and three-dimensional features. It will make up for the lack of texture information in the LiDAR point cloud data.

**Acknowledgement:** We would like to thank the support of the School of Nanjing University of Information Science & Technology.

**Funding Statement:** This work was supported, in part, by the National Nature Science Foundation of China under grant numbers 62272236; in part, by the Natural Science Foundation of Jiangsu Province under grant numbers BK20201136, BK20191401; in part, by the Priority Academic Program Development of Jiangsu Higher Education Institutions (PAPD) fund.

**Conflicts of Interest:** The authors declare that they have no conflicts of interest to report regarding the present study.

## References

[1] Y. Zhou and O. Tuzel, “Voxelnet: End-to-end learning for point cloud based 3d object detection,” in *Proc. CVPR*, Salt Lake City, UT, USA, pp. 4490–4499, 2018.

[2] S. S. Shi, C. X. Guo, L. Jiang, Z. Wang, J. Shi *et al.,* “PV-RCNN: Point-voxel feature set abstraction for 3d object detection,” in *Proc. CVPR*, Seattle, WA, USA, pp. 10526–10535, 2020.

[3] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang *et al.,* “Fast encoders for object detection from point clouds,” in *Proc. CVPR*, Long Beach, CA, USA, pp. 12697–12705, 2019.

[4] A. Geiger, P. Lenz and R. Urtasun, “Are we ready for autonomous driving? The kitti vision benchmark suite,” in *Proc. CVPR*, Providence, RI, USA, pp. 3354–3361, 2012.

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

[6] S. Qiu, Y. F. Wu, S. Anwar and C. Y. Li, “Investigating attention mechanism in 3d point cloud object detection,” in *Proc. 3DV*, Xia Men, China, pp. 403–412, 2021.

[7] L. W. Ye, M. Rochan, Z. Liu and Y. Wang, “Cross-modal self-attention network for referring image segmentation,” in *Proc. CVPR*, Long Beach, CA, USA, pp. 10502–10511, 2019.

[8] H. Zhao, J. Y. Jia and V. Koltun, “Exploring self-attention for image recognition,” in *Proc. CVPR*, Seattle, WA, USA, pp. 10076–10085, 2020.

[9] C. R. Qi, L. Yi, H. SU and L. J. Guibas, “Pointnet++: Deep hierarchical feature learning on point sets in a metric space,” *Advances in Neural Information Processing Systems*, vol. 30, pp. 5099–5108, 2017.

[10] A. Vaswani, N. Shazeer, N. Parmar, N. Uszkoreit, J. Jones *et al.,* “Attention is all you need,” *Advances in Neural Information Processing Systems*, vol. 30, pp. 5998–6008, 2017.

[11] Z. Tan, M. Wang, J. Xie, Y. Chen and X. Shi, “Deep semantic role labeling with self-attention,” in *Proc. AAAI*, Louisiana, LA, USA, pp. 30548–30559, 2018.

[12] S. Woo, J. Park, J. Y. Lee and I. S. Kweon, “Cbam: Convolutional block attention module,” in *Proc. ECCV*, Munich, Germany, pp. 3–19, 2018.

[13] C. Yan, L. Meng, L. Li, J. Zhang, Z. Wang *et al.,* “Age-invariant face recognition by multi-feature fusion and decomposition with self-attention,” *ACM Transactions on Multimedia Computing, Communications, and Applications*, vol. 18, no. 1, pp. 1–18, 2022.

[14] Z. Huang, W. Ke and D. Huang, “Improving object detection with inverted attention,” in *Proc. WACV*, Snowmass Village, C, USA, pp. 1294–1302, 2020.

[15] J. Cao, Q. Chen, J. Guo and R. Shi, “Attention-guided context feature pyramid network for object detection,” *arXiv Preprint*, vol. 15, no. 1, pp. 63–72, 2020.

[16] K. He, X. Zhang, X. Ren and J. Sun, “Deep residual learning for image recognition,” in *Proc. CVPR*, Las Vegas, NV, USA, pp. 770–778, 2016.

[17] D. Zhang, J. Hu, F. Li, X. Ding, A. K. Sangaiah *et al.,* “Small object detection via precise region-based fully convolutional networks,” *Computers, Materials & Continua*, vol. 69, no. 2, pp. 1503–1517, 2021.

[18] J. Wang, T. Zhang, Y. Cheng and N. Al-Nabhan, “Deep learning for object detection: A survey,” *Computer Systems Science and Engineering*, vol. 38, no. 2, pp. 165–182, 2021.

[19] W. Sun, L. Dai, X. R. Zhang, P. S. Chang and X. Z. He, “RSOD: Real-time small object detection algorithm in UAV-based traffic monitoring,” *Applied Intelligence*, pp. 1–16, 2021. https://doi.org/10.1007/s10489-021-02893-3.

[20] T. Wang, X. Zhu, J. Pang and D. Lin, “Fcos3d: Fully convolutional one-stage monocular 3d object detection,” in *Proc. CVPR*, Online, pp. 913–922, 2021.

[21] J. Deng, S. Shi, P. Li, W. Zhou, W. Zhang *et al.,* “Voxel r-cnn: Towards high performance voxel-based 3d object detection,” in *Proc. AAAI*, Online, pp. 1201–1209, 2021.

[22] Y. Shi, Y. Guo, Z. Mi and X. Li, “Stereo CenterNet-based 3d object detection for autonomous driving,” *Neurocomputing*, vol. 471, no. 4, pp. 219–229, 2022.

[23] Y. Yan, Y. X. Mao and B. Li, “Second: Sparsely embedded convolutional detection,” *Sensors*, vol. 18, no. 10, pp. 3337–3353, 2018.

[24] S. S. Shi, X. G. Wang and H. S. Li, “Pointrcnn: 3d object proposal generation and detection from point cloud,” in *Proc. CVPR*, Long Beach, CA, USA, pp. 770–779, 2019.

[25] T. Y. Lin, P. Goyal, P. Girshick, K. He and P. Dollár, “Focal loss for dense object detection,” in *Proc. CVPR*, Honolulu, HI, USA, pp. 2980–2988, 2017.

[26] W. Sun, X. Chen, X. R. Zhang, G. Z. Dai, P. S. Chang *et al.,* “A multi-feature learning model with enhanced local attention for vehicle re-identification,” *Computers, Materials & Continua*, vol. 69, no. 3, pp. 3549–3560, 2021.

[27] X. R. Zhang, X. Sun, W. Sun, T. Xu and P. P. Wang, “Deformation expression of soft tissue based on BP neural network,” *Intelligent Automation & Soft Computing*, vol. 32, no. 2, pp. 1041–1053, 2022.

[28] X. R. Zhang, J. Zhou, W. Sun and S. K. Jha, “A lightweight CNN based on transfer learning for COVID-19 diagnosis,” *Computers, Materials & Continua*, vol. 72, no. 1, pp. 1123–1137, 2022.

[29] Z. Wang and K. Jia, “Frustum convnet: Sliding frustums to aggregate local point-wise features for amodal 3d object detection,” in *Proc. IROS*, Macao, MAC, China, pp. 1742–1749, 2019.

[30] C. R. Qi, W. Liu, C. Wu, H. Su and L. J. Guibas, “Frustum pointnets for 3d object detection from rgb-d data,” in *Proc. CVPR*, Salt Lake City, UT, USA, pp. 918–927, 2018.

[31] J. Ku, M. Mozifian, M. Lee, A. Harakeh and S. L. Waslander Cai, “Joint 3d proposal generation and object detection from view aggregation,” in *Proc. IROS*, Madrid, MAD, Spain, pp. 1–8, 2018.

[32] X. Chen, H. Ma, J. Wan, B. Li and T. Xia, “Multi-view 3d object detection network for autonomous driving,” in *Proc. CVPR*, Honolulu, HI, USA, pp. 1907–1915, 2017.

[33] S. Vora, A. H. Lang, B. Helou and O. Beijbom, “Pointpainting: Sequential fusion for 3d object detection,” in *Proc. CVPR*, Seattle, WA, USA, pp. 4604–4612, 2020.

[34] M. Liang, B. Yang, S. Wang and R. Urtasun, “Deep continuous fusion for multi-sensor 3d object detection,” in *Proc. ECCV*, Munich, MUC, Germany, pp. 641–656, 2018.

[35] D. Zhou, J. Fang, X. Song, C. Guan, J. Yin *et al.,* “Iou loss for 2d/3d object detection,” in *Proc. 3DV*, Quebec City, QC, Canda, pp. 85–94, 2019.

[36] Z. T. Yang, Y. Sun, S. Liu, X. Shen and J. Jia, “Std: Sparse-to-dense 3d object detector for point cloud,” in *Proc. ICCV*, Seoul, SEL, Korea (south), pp. 1951–1960, 2019.

[37] Y. L. Che, S. Liu, X. Y. Shen and J. Jia, “Fast point r-cnn,” in *Proc. ICCV*, Seoul, SEL, Korea (south), pp. 9775–9784, 2019.

[38] Z. T. Yang, Y. Sun, S. Liu and J. Jia, “3dssd: Point-based 3d single stage object detector,” in *Proc. CVPR*, Seattle, WA, USA, pp. 11040–11048, 2020.