AGPNet: Augmented Graph Pointpillars for Object Detection from Point Clouds

Xiaohua Wang *, Jian Liu, Zhonghe Liao

Posted Date: 27 October 2023

doi: 10.20944/preprints202310.1729.v1

Keywords: 3D object detection; point cloud; voxel; deep learning

Article

AGPNet: Augmented Graph Pointpillars for Object Detection from Point Clouds

Xiaohua Wang [1,*], Jian Liu [2] and Zhonghe Liao [2]

[1] School of Mechatronic Engineering and Automation, Shanghai University, China
[2] x.wang@shu.edu.cn; jianl@shu.edu.cn; zhongheliao@shu.edu.cn;
* Correspondence: x.wang@shu.edu.cn

Abstract: In the realm of autonomous vehicle environment perception, the primary objective of point cloud target detection is to swiftly and accurately identify three-dimensional objects from point cloud data. To meet this requirement, the prevalent network architecture employed in the industry is the voxel-based PointPillars model. Nonetheless, this model faces challenges in maintaining detection accuracy when objects are obscured or diminutive in size. In response to this issue, we introduce AGPNet, a novel model that seamlessly integrates four key modules: Data Augmentation, Dynamic Graph CNN, Pillar Feature Net, and Detection Head (SSD). The Data Augmentation module enhances the adaptability of point cloud data to complex and ever-changing real-world environments. The Dynamic Graph CNN module endows the network structure with geometric features, which encapsulate not only the point itself but also its adjacent points. The Pillar Feature Net module translates three-dimensional point cloud data into pseudo-image data through the utilization of voxels. Subsequently, the Detection Head (SSD) module leverages this pseudo-image data to conduct target detection of three-dimensional objects. Our experiments, conducted on the KITTI dataset, demonstrate that our proposed method boosts object detection accuracy by 6-7 percentage points compared to the PointPillars model, while maintaining similar detection times.

Keywords: 3D object detection; point cloud; voxel; deep learning

1. Introduction

The detection of three-dimensional objects constitutes a vital component within the autonomous driving environment perception system. It offers crucial reference information about objects in the vehicle's vicinity for subsequent processes. Common methods for three-dimensional object detection rely on both image data and three-dimensional point cloud data. Owing to its robust resistance to interference and its capability to capture the three-dimensional nature and spatial characteristics of the object's environment, three-dimensional point cloud data has emerged as a fundamental approach for achieving high-quality 3D object detection.

In recent years, with deep learning yielding remarkable outcomes in the field of two-dimensional image analysis in computer vision, an increasing number of researchers have turned their attention to the integration of deep learning techniques with three-dimensional point clouds. This integration capitalizes on the wealth of spatial information provided by point clouds to enhance the computer's capacity to process and understand complex three-dimensional data, ultimately elevating its environmental awareness. Nevertheless, owing to the irregular and disordered nature of point cloud data, researchers often face challenges when attempting to extract relevant local features through convolution, unlike the relatively structured two-dimensional images. Current three-dimensional object detection methods employing point cloud deep learning can be categorized into three main approaches: multi-view fusion, point-based methods, and voxel-based methods. Within the multi-view fusion category, notable examples include MV3D [1], AVOD [2], and MVF [3]. These methods involve projecting 3D point clouds into multi-view images, enabling the use of 2D object detection techniques to perform 3D object detection tasks. However, the process of converting from three-dimensional to two-dimensional representation results in dimension reduction, limiting the ability of multi-view fusion methods to capture fine-grained point cloud features, which can affect

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 27 October 2023 doi:10.20944/preprints202310.1729.v1

their overall detection accuracy. In contrast, point-based methods, such as PointRCNN [4], Frustum PointNet [5], and Point-GNN [6], utilize PointNet and PointNet++ [7,8] networks for direct feature learning from the original point cloud data. They predict object frame positions and sizes for selected points in the object point cloud and employ the non-maximum suppression technique used in two-dimensional object detection models like SSD (Single Shot MultiBox Detector, SSD) [9] to obtain final three-dimensional object frame regression and classification results. While point-based methods offer high detection accuracy without issues like resolution loss resulting from point cloud grouping, they are computationally intensive and have slower detection speeds, making them less suitable for real-time detection in autonomous driving scenarios. On the other hand, voxel-based methods, including VoxelNet [10], SECOND [11], and PointPillars [12], employ a uniform voxelization process to encode irregular point cloud data into a regular, grid-like format akin to images. The encoded data is subsequently fed into traditional convolutional neural networks for feature extraction, which leads to the acquisition of voxel-based features. Finally, these voxel features are utilized to generate object frame prediction results through two-dimensional target detection algorithms. Voxel-based methods strike a balance between detection accuracy and speed, making them a prominent choice for three-dimensional object detection in this study.

In this paper, we introduce the AGPNet model, a network structure that combines Data Augmentation, Dynamic Graph CNN, Pillars Feature Net, and Detection Head (SSD). As illustrated in Figure 1, the AGPNet model comprises four stages: (i) Data Augmentation Stage: Through occlusion, exchange, and sparsity operations, the point cloud data can be brought closer to the real-world point cloud data in a genuine environment. (ii) Dynamic Graph CNN Stage: Utilizing point cloud convolution network operations, we can extract pertinent geometric information for each point within the point cloud and its neighboring points, and integrate it into the dimension of each point. (iii) Pillars Feature Net Stage: By employing PointPillars-related processing operations, we can transform relevant point cloud data into pseudo-image information. (iv) Detection Head (SSD) Stage: The SSD detection module processes the pseudo-image information from the previous stage to obtain pertinent information for object detection. Experiments demonstrate that, compared to the commonly used PointPillars model architecture in the industry, our AGPNet model enhances 3D object detection accuracy by approximately 6-7%, while maintaining detection times within the same order of magnitude as PointPillars. This meets the real-time 3D object detection requirements for autonomous vehicles.

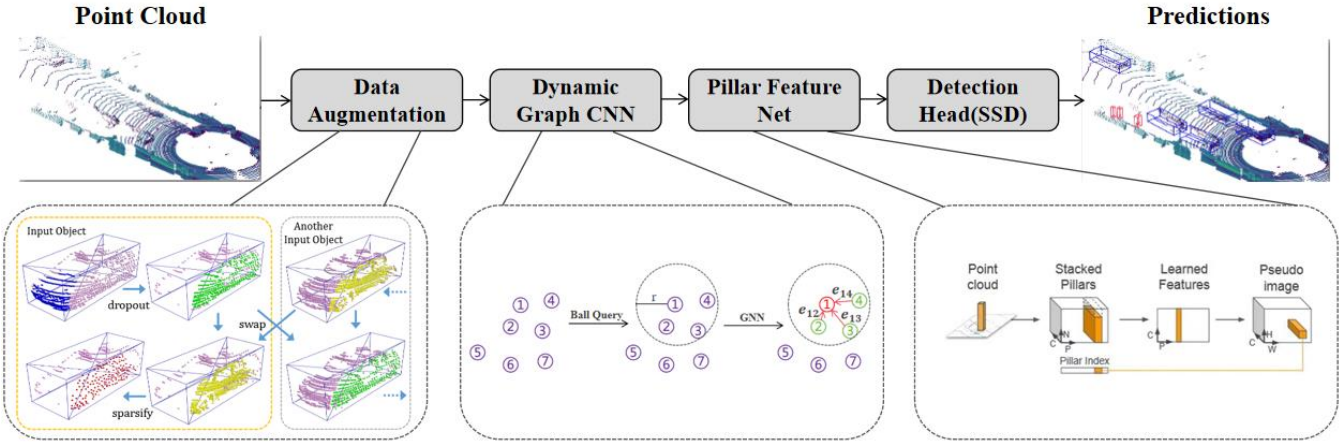

Figure 1. AGPNet Module Overview: The primary components of the network encompass Data Augmentation, Dynamic Graph CNN, Pillar Feature Net, and Detection Head (SSD) . Data Augmentation serves to enhance data diversity, while Dynamic Graph CNN conduct high-dimensional operations on point cloud data. The Pillar Feature Net transforms the initial elevated point cloud into a two-dimensional pseudo-image, and the Detection Head (SSD) utilizes the pseudo-image features to predict the three-dimensional bounding box of the object.

 doi:10.20944/preprints202310.1729.v1

1.1. Related Work

1.1.1. Data Augmentation

To address challenges such as occlusion, proximity, varying shapes of objects within the same category encountered in real road conditions, and to enhance the model's capacity to understand the unique circumstances of objects, as well as improve the common detection results of current 3D object detection models that often exhibit limited accuracy, particularly for distant or sparse object point clouds, we employed data augmentation techniques. In this study, we drew inspiration from the data enhancement operations introduced in [12]. However, our approach, embodied in the AGPNet model's Data Augmentation module, diverges from the original method in an important way. Rather than using the farthest point sampling method, as proposed in [12], during the sparsification operation, we opted for a voxel-based approach. In this method, we randomly selected a point within each voxel's range as the representative point for that voxel. This modification serves to bring our point cloud data closer to real-world conditions, while simultaneously reducing the computational load for subsequent point cloud data processing.

1.1.2. Dynamic Graph CNN

When it comes to extracting geometric features from point clouds, this paper adopts the ball query method from the point cloud Dynamic Graph convolutional neural network (Dynamic Graph CNN) [13]. This method involves identifying all points within a specified radius of each point and combining the resulting vectors, which consist of the central point and its neighboring points. These combined vectors are utilized for graph convolution-based feature learning. Through a series of operations, the model learns the pertinent geometric relationships between the central point and the neighboring points within the sphere. The learned features, along with the three-dimensional coordinates of the corresponding point, are then utilized as feature dimensions for that point.

1.1.3. Differences from Pointpillars Model

Our AGPNet network model was developed in response to the PointPillars [12] network model's limitations in accurately detecting occluded or small objects. While AGPNet draws inspiration from the PointPillars network structure, it differs significantly in several key aspects. Firstly, our AGPNet model omits the Backbone (2D CNN) module found in the PointPillars model, opting instead for point cloud processing using the Dynamic Graph CNN module before voxelizing the features within the Pillar Feature Net. Secondly, we introduce a Data Augmentation module to our model to address the challenges posed by occlusion in point cloud data within real-world environments, as well as variations in point cloud data distances. Lastly, in the Pillar Feature Net module, the PointPillars model voxelizes the input point data, including three-dimensional coordinates $(\mathrm{x}, \mathrm{y}, \mathrm{z})$, reflectivity (r), the distance of three-dimensional coordinates from the column's center point (x_c, y_c, z_c), and the offset distance from the center of the column's xy-plane (x_p, y_p). These parameters make up the nine-dimensional feature space for each point. In contrast, our Pillar Feature Net module in AGPNet takes as input the three-dimensional coordinates of the point cloud $(\mathrm{x}, \mathrm{y}, \mathrm{z})$ and the features obtained after learning from the corresponding point cloud convolutional neural network, resulting in a feature space with C dimensions.

1.2. Contributions

The Key contributions of this paper are as follows:

- We introduce the AGPNet network framework, which offers a novel integration of Data Augmentation and Dynamic Graph CNN modules with PointPillars. This innovative approach not only enhances the diversity of point cloud data but also transforms it into pseudo-image data, resulting in a notable acceleration in the detection process;
- Our proposed model effectively addresses the issue of inadequate detection accuracy encountered when dealing with obstructed or small-sized objects, a common limitation in the

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 27 October 2023 doi:10.20944/preprints202310.1729.v1

current industry-standard PointPillars model. Importantly, our model strikes a balance between detection speed and accuracy;

- Experimental results demonstrate that our method achieves a detection accuracy approximately 6-7% higher than that of the PointPillars model when evaluated on the KITTI dataset. Remarkably, it accomplishes this without sacrificing detection speed, ensuring it meets the stringent requirements for both accuracy and speed in a real-world autonomous driving environment.

2. Proposed 3D Object Detection

The AGPNet network model, as illustrated in Figure 1, addresses the transformation of discrete point cloud data $P = \{p_1, \ldots, p_n\} \subseteq R^3$. Our primary objective is to convert this data into a more representative form, one that better captures the geometric characteristics of the surrounding point cloud data, ultimately enhancing the accuracy of three-dimensional object detection within point clouds. As explained in the initial section, our approach to converting point cloud data into pseudo images entails several key steps $P = \{p_1, \ldots, p_n\} \subseteq R^3$ to $F(H \times W \times C)$. Firstly, we apply Data Augmentation module to the point cloud data $P = \{p_1, \ldots, p_n\} \subseteq R^3$ to obtain enhanced three-dimensional point cloud data $\widetilde{P} = \{p_1, \ldots, p_N\} \subseteq R^3$ $(N \neq n)$. Subsequently, we employ Dynamic Graph CNN module to upgrade the point cloud data's dimensions $\widehat{P} = \{p_1, \ldots, p_N\} \subseteq R^c$ $(N \neq n)$, and then pass it through the Pillar Feature Net module, which transforms it into pseudo image $F(H \times W \times C)$. The pseudo image is then processed by the Detection Head (SSD) module to facilitate three-dimensional object detection, ultimately producing prediction results.

$$P \rightarrow \widetilde{P} \rightarrow \widehat{P} \rightarrow F \tag{1}$$

As depicted in Figure 1, the implementation process of equation (1) primarily comprises four modules:

- The Data Augmentation module augments point cloud data diversity, facilitating the acquisition of more varied point cloud features;
- The Dynamic Graph CNN module performs dimension expansion on each point, encompassing its three-dimensional coordinates and pertinent geometric relationships with nearby points;
- The Pillar Feature Net module voxels the dimension-expanded point cloud data and conducts related processing to convert it into pseudo-image information representing the point cloud data;
- The Detection Head (SSD) module processes the two-dimensional pseudo-image features and generates three-dimensional detection results.

The quality of the pseudo-image data produced by the last two modules is directly influenced by the data processing outcomes from the initial two modules, which subsequently impacts the object detection results from the fourth module.

2.1. Data Augmentation

In Figure 1, our shape enhancement scheme is illustrated within the Data Augmentation module as per the design. The object point cloud is divided into six regions based on the diagonal connections of the real bounding box, as it is often encountered in Lidar point cloud scans on object surfaces. This division into six areas enhances our point cloud data's ability to mimic real-world point cloud data.

Our data augmentation process is specifically executed in three steps:

- Regional Ensemble Point Cloud Pruning: A certain probability is applied to randomly remove point cloud data from one of the six regions, as depicted in the blue section in the figure. This action simulates point cloud occlusion experienced in real-world environments.
- Inter-Region Point Cloud Exchange: With a specified probability, an area is randomly selected from the six regional point cloud sets in the current frame. Corresponding point cloud data is then exchanged with another region on the same side of objects belonging to the same category.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 27 October 2023 doi:10.20944/preprints202310.1729.v1

This exchange is illustrated by the yellow and green portions in the figure. It serves to enrich the point cloud data's ability to capture the diversity among objects of the same category.

- Subsequent Sparsification: For the point cloud data involved in the second step, a voxel-based method is used to randomly select a point within non-empty voxels. This point serves as a representative feature of the voxel area, and downsampling is applied, as seen in the yellow part of the point cloud in the figure. This operation simulates the variation in point cloud density in real environments.

Through these operations, we transform the initial point cloud data $P = \{p_1, \ldots, p_n\} \subseteq R^3$ into enhanced point cloud data $\widetilde{P} = \{p_1, \ldots, p_N\} \subseteq R^3\ (N \neq n)$. Our data augmentation procedure significantly reduces the computational cost of subsequent point cloud data processing while effectively mimicking the characteristics of point cloud data in real-world scenarios.

2.2. Dynamic Graph CNN

In view of the close-to-distant characteristics of 3D point cloud data in the real environment, we use the Ball Query method to perform graph convolution operations on point clouds. The implementation steps are shown in the Dynamic Graph CNN module in Figure 1. Definition $\widetilde{P} = \{p_1, \ldots, p_N\} \subseteq R^3\ (N \neq n)$, $\widetilde{P}$ is a set of N points in the point cloud, where $p_{i\in\{1\ldots N\}} = (x_i, y_i, z_i)$ means that each point includes its three-dimensional coordinate value. The discrete graph structure composed of points and adjacent points can be expressed as $G = (\widetilde{P}, E)$, where:

$$E = \{(p_i, p_j) | \parallel p_i - p_j \parallel < r, i = \{1, \ldots, N\}, j = \{1, \ldots, m\}\} \quad (2)$$

In other words, this involves the set of m neighboring points located within a radius r of the given point and within the sphere. Next, we process the matrix consisting of point coordinates and coordinates of neighboring points through a shared multi-layer perceptron (MLP) to perform a dimensionality transformation, yielding the resulting feature e_{ij}:

$$e_{ij} = \mathrm{ReLU}(\theta_m \cdot (p_i - p_j) + \emptyset_m \cdot p_i) \quad (3)$$

which $\Theta = (\theta_1, \ldots \theta_m, \emptyset_1, \ldots \emptyset_m)$ represents the learnable parameter of the shared multi-layer perceptron (MLP).

Finally, the corresponding features are aggregated through the max pooling operation:

$$x_{im} = \max_{(i,j)\in E}(e_{ij}) \quad (4)$$

By employing the point cloud graph convolutional neural network operation, we acquire high-dimensional point cloud data $\widehat{P} = \{p_1, \ldots, p_N\} \subseteq R^c\ (N \neq n)$ that includes the three-dimensional coordinates of each point and the characteristics of its neighboring points.

2.3. Pillar Feature Net

Following the preceding operation of the point cloud convolutional neural network, we've obtained point cloud data $\widehat{P} = \{p_1, \ldots, p_N\} \subseteq R^c\ (N \neq n)$ with dimension C. The first three dimensions of each point represent the x, y, and z-axis coordinates of the point cloud data. Next, we will execute Pointpillar operations on these points to transform them into pseudo-image data. The operational details are depicted in the Pillar Feature Net module in Figure 1.

- Firstly, we discretize the point cloud by uniformly dividing it into a grid($0.16^2 m^2$) on the x-y plane, generating a collection of columns, denoted as P, which function as voxels with infinite spatial extent along the z-coordinate.
- Secondly, we construct a dense tensor with dimensions (C, P, N) by imposing specific constraints on the number of non-empty bins per sample, P, and the number of points per bin, N. In cases where a sample or pillar contains an excessive number of points, we downsample the data by randomly selecting P points or, if needed, apply zero padding to ensure completeness.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 27 October 2023 doi:10.20944/preprints202310.1729.v1

- Thirdly, we employ a simplified version of PointNet to produce a tensor with dimensions (C, P, N). Subsequently, a maximum operation is applied to create an output tensor with dimensions (C, P).
- Fourthly, we disperse these features back to their original pillar positions, effectively generating pseudo-image data denoted as $F(H \times W \times C)$, with H and W representing the height and width of the canvas.

2.4. Detection Head (SSD)

In the Detection Head (SSD) module, we employ a two-dimensional object detection network based on SSD to conduct object frame matching detection on the pseudo-picture features after feature compression and conversion. In the final layer of the network, height regression prediction parameters for the object frame are output through the fully connected layer, restoring the z-axis height information of the object frame.

2.5. Loss Function

In the context of the three-dimensional object detection model, the model's output must provide information about the size, orientation, and precise object classification results. Consequently, our loss function primarily consists of a weighted sum of the object frame regression loss function and the classification loss function.

2.5.1. Object frame regression loss function

The object frame regression part needs to predict the center position coordinates of the object's three-dimensional object frame, the length, width and height of the frame (l, w, h) and the direction angle of the frame θ. In order to facilitate effective training of the model and enable rapid convergence, the predicted object frame parameters $(x, y, z, l, w, h, \theta)$ are normalized. For convenience of description, the following definitions are given:

(1) $(x^{gt}, y^{gt}, z^{gt}, l^{gt}, w^{gt}, h^{gt}, \theta^{gt})$ represents the real object frame parameter;

(2) $(x^{a}, y^{a}, z^{a}, l^{a}, w^{a}, h^{a}, \theta^{a})$ denotes the object frame parameter of the positive sample in the prediction;

(3) $\{a_i^{pos}\}_{i=1\ldots Npos}$ stands for the set of positive sample object frames in the prediction, with a total of N_{pos};

(4) $\{a_i^{neg}\}_{i=1\ldots Nneg}$ signifies the set of negative sample object frames in the prediction, with a total of N_{neg}.

The regression residual between the actual object frame and the predicted object frame is calculated as follows:

$$\Delta x = \frac{x^{gt} - x^{a}}{d^{a}}, \Delta y = \frac{y^{gt} - y^{a}}{d^{a}}, \Delta z = \frac{z^{gt} - z^{a}}{h^{a}} \tag{5}$$

$$\Delta \omega = \log \frac{\omega^{gt}}{\omega^{a}}, \Delta l = \log \frac{l^{gt}}{l^{a}}, \Delta h = \log \frac{h^{gt}}{h^{a}} \tag{6}$$

$$\Delta \theta = \sin(\theta^{gt} - \theta^{a}) \tag{7}$$

Among them, $d^{a} = \sqrt{(\omega^{a})^2 + (l^{a})^2}$ is the diagonal length of the matrix where the length and width of the positive sample object frame are predicted.

Object frame regression loss function We use the Smooth L1 [15] function to calculate the object frame regression loss, including the regression loss value of $\Delta x, \Delta y, \Delta z, \Delta w, \Delta l, \Delta h, \Delta \theta$. Define $\mathcal{L}_{loc}$ as the position regression loss function (the subscript loc represents location, that is, the position of the object frame), and the formula is as follows:

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 27 October 2023 doi:10.20944/preprints202310.1729.v1

$$\mathcal{L}_{loc} = \sum_{b\in(x,y,z,\omega,l,h,\theta)} \text{smooth}_{L_1}(\Delta b) \tag{8}$$

$$\text{smooth}_{L_1}(x) = \begin{cases} 0.5x^2 & \text{if } |x| < 1 \\ |x| - 0.5 & \text{if } |x| \geq 1 \end{cases} \tag{9}$$

While the $\mathcal{L}_{loc}$ loss function calculates the object frame angle loss $\Delta\theta$, it does not address the specific orientation problem of the object frame. We approach the task of determining the specific orientation of the object frame as a two-class problem with mutually exclusive outcomes. Once the rotation angle $\Delta\theta$ of the object frame is established, there are only two possible orientations: the current orientation and the orientation after a 180° rotation. To determine the specific orientation of the object frame, we introduce a direction regression loss function $\mathcal{L}_{dir}$ (where the subscript 'dir' denotes direction, representing the orientation of the object frame). In this context, we utilize the softmax classification loss function [16], denoted as $\mathcal{L}_{dir}$ loss function in Formula (10). We assign a direction category, denoted as category 'i,' with i=1 corresponding to the current orientation and i=2 to the orientation after a 180° rotation. The value predicted by the model for category 'i' is represented as x_i.

$$\mathcal{L}_{dir}(x_i) = \frac{1}{1 + e^{x_j - x_i}} \tag{10}$$

Among them, $i = 1, 2;\ j = 1, 2;\ j \neq i$.

2.5.2. Classification loss function

For the object classification task, we are confronted with the challenge of categorizing three distinct types of objects in the dataset. Additionally, the object categories exhibit an issue of imbalanced sample distribution. To address this issue, we employ the focal loss [3] classification function to tackle the multi-classification problem within the context of imbalanced samples.

The classification loss function $\mathcal{L}_{cls}$ (the subscript cls represents class, that is, the object frame category) is defined as follows:

$$\mathcal{L}_{cls} = -\alpha_a(1 - p^a)^\gamma \log p^a \tag{11}$$

Among them, $a \in \{0,1,2\}$ represents three types of objects respectively, 0 represents the car category, 1 represents the pedestrian category, and 2 represents the bicycle category. p^a is the object frame category confidence. In the article, α is set to 0.25 and γ is set to 2.

2.5.3. Overall loss function

The overall loss function of the model is expressed as follows:

$$\mathcal{L} = \frac{1}{N_{pos}} (\beta_{loc1}\mathcal{L}_{loc1} + \beta_{loc2}\mathcal{L}_{loc2} + \beta_{cls}\mathcal{L}_{cls} + \beta_{dir}\mathcal{L}_{dir}) \tag{12}$$

Considering the challenge of imbalanced object categories, we introduce different scale parameters for large and small objects: β_{loc1} is the position regression loss function for the 'car' object category, set to 1; β_{loc2} is the position regression loss function for the 'bicycle' and 'pedestrian' object categories, set to 2; β_{cls} is the object category loss function, set to 1; and β_{dir} is the object direction loss function, set to 0.2. This strategic parameter adjustment strengthens the weight attributed to small objects. As a result, the model prioritizes the accuracy of small object detection during training.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 27 October 2023 doi:10.20944/preprints202310.1729.v1

3. Evaluations

3.1. Implementation Details

3.1.1. Experiment platform

All experiments in this paper were conducted within the following hardware environment: Intel Xeon (R) CPU E5-2603, GTX 1050 Ti GPU graphics card; and software environment: 64-bit Ubuntu 18.04 LTS, Python 3.7, PyTorch 1.3.0, OpenPCDet 0.5.2, CUDA 10.0, cuDNN 7.5.0, PCL 1.11, MATLAB R2019b, numpy, tensorboardX, and numba. OpenPCDet is a 3D object detection code library for lidar point cloud analysis developed by the Open-MMLab team. It is commonly employed for 3D object detection based on lidar data. The Point Cloud Library (PCL) is a comprehensive library that provides various commonly used point cloud processing functions.

3.1.2. Experimental data

The KITTI dataset was established jointly by the Karlsruhe Institute of Technology (KIT) in Germany and the Toyota Technical Institute of Chicago (TTIC) in 2012, specifically for autonomous driving tasks. This dataset comprises data collected from various sensors, including optical cameras, lidar, and others, mounted on a Volkswagen Passat, as depicted in Figure 3. It includes a total of 14,999 images along with their corresponding point cloud data. Out of these, 7,481 groups were allocated for training, while 7,518 groups were designated for testing. The KITTI dataset offers annotations for a grand total of 80,256 objects within the three primary categories: cars, pedestrians, and bicycles, making it valuable for tasks related to environmental perception in autonomous driving, including 2D and 3D detection. It's essential to note that the KITTI 3D object detection test set does not provide object labels. Consequently, to acquire evaluation data, detection results must be submitted to the official website. In our research, we divided the official KITTI 3D object detection training dataset into a training set and a validation set, with 3,712 samples assigned to training and 3,769 samples allocated for validation, following the guidelines in [3]. Additionally, the KITTI dataset has been categorized into three difficulty levels based on different occlusion conditions, as presented in Table 1: 'simple,' 'medium,' and 'difficult.' This classification helps in assessing the object detection performance under varying complexities.

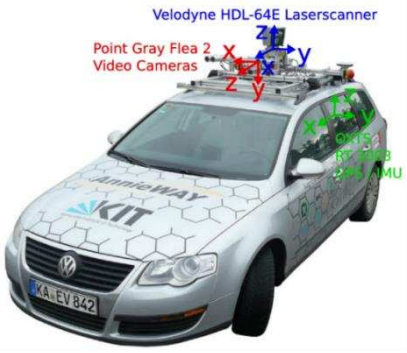

Figure 2. KITTI data set collection equipment, the sensors used by vehicles include: inertial navigation system (GPS/IMU), lidar (Velodyne HDL-64E), grayscale and color cameras.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 27 October 2023 doi:10.20944/preprints202310.1729.v1

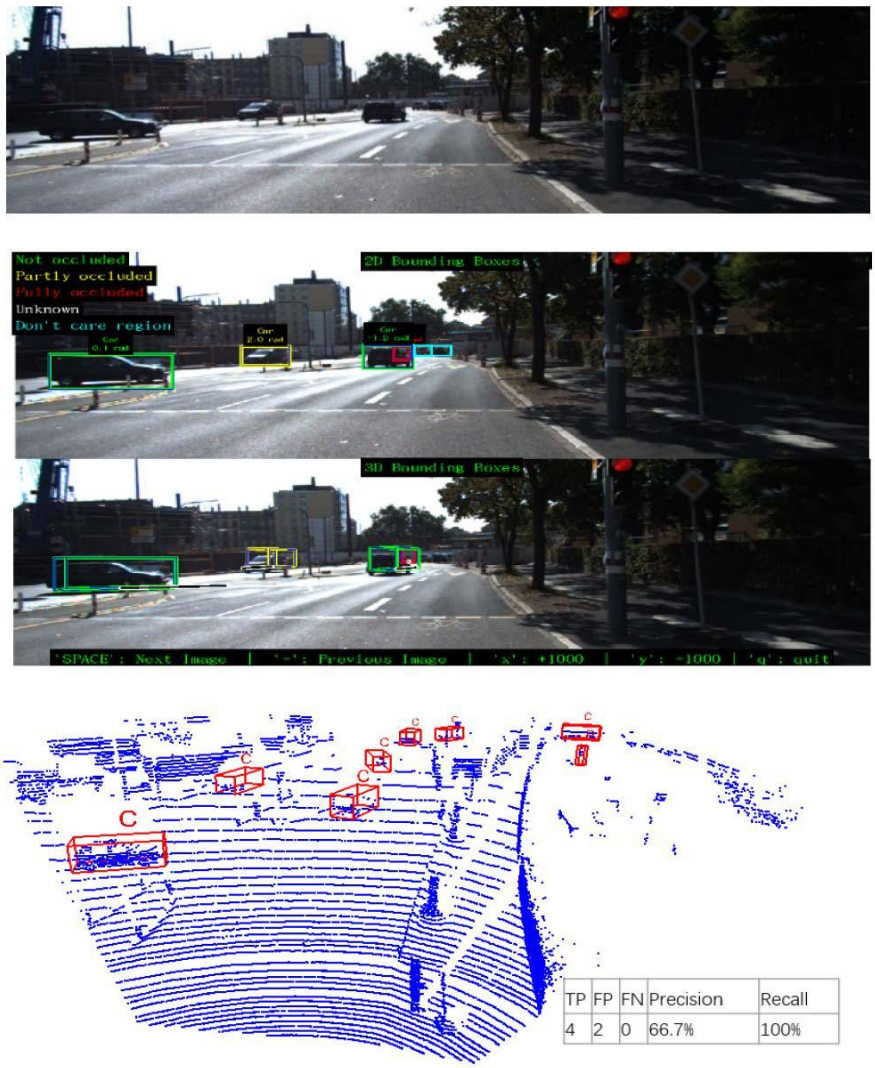

Figure 3. Visualization of the verification results of the AGPNet model under the verification set 0006 data set, (a) simulated verification set 0006 camera image. (b) Simulation verification set of two-digit image detection situation and sum. (c) Simulation verification set 3D point cloud detection situation

Table 1. Description of difficulty of data set detection

Difficulty	Bounding box Minimum Height	Object occlusion level	The maximum degree of truncation of an object
Simple	40px	0, fully visible	15%
Medium	25px	1, partial occlusion	30%
Difficult	25px	2, hard to see	50%

3.1.3. Evaluation index

In this article, we calculate the target detection evaluation metrics using the prediction of positive samples (TP - True Positives), the prediction of negative samples (FP - False Positives), and the absence of true positive samples (FN - False Negatives) to derive accuracy (precision) and recall. Furthermore, we compute the average accuracy (AP - Average Precision) and mAP (Mean Average Precision) based on accuracy and recall values.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 27 October 2023 doi:10.20944/preprints202310.1729.v1

First, we predict the IOU (Intersection over Union) of the area or volume of the object frame and the real object frame. During the experiment, we set the IOU of the car category object to 0.7, and the pedestrian and bicycle category object IOU to 0.5;

$$\mathrm{IOU} = \frac{\mathrm{area}(B_p \cap B_{gt})}{\mathrm{area}(B_p \cup B_{gt})} \tag{13}$$

where B_p is the predicted object box (predict bounding box) and B_{gt} is the ground truth bounding box.

If IOU is greater than the threshold, the object frame result is determined to be TP. If it is less than the threshold, it is determined to be FP. For the true value of the object frame that does not intersect with the object frame prediction result, it is determined to be FN. It can be performed through TP, FP, and FN. Calculation of precision and recall;

$$\mathrm{precision} = \frac{TP}{TP + FP} \tag{14}$$

$$\mathrm{recall} = \frac{TP}{TP + FN} \tag{15}$$

Finally, mAP is calculated using the precision corresponding to the 40recall value:

$$\mathrm{mAP}|_R = \frac{1}{|R|} \sum_{r \in R} \rho_{\,\mathrm{interp}}(r) \tag{16}$$

Which takes $R = \{\frac{1}{40}, \frac{2}{40}, \frac{3}{40}, \ldots, 1\}$.

In the multi-scene continuous frame detection process, first, a frame of point cloud data is extracted from the scene continuous frame data set in chronological order, and put into the trained model to obtain the object frame result predicted by the model; the prediction result and KITTI tracking result are compared with the object frame true Compare the value results, calculate the two-dimensional IOU value from the BEV perspective, and obtain the TP, FP, and FN values of the point cloud data of the frame. Traverse all frame point cloud data in chronological order; count the number of TP, FP, and FN of all object frames; finally, calculate the evaluation indicators precision, recall, and average detection time based on the number of TP, FP, and FN.

$$\mathrm{precision} = \frac{\sum_{i=1}^{n} TP_i}{\sum_{i=1}^{n} TP_i + \sum_{i=1}^{n} FP_i} \tag{17}$$

$$\mathrm{recall} = \frac{\sum_{i=1}^{n} TP_i}{\sum_{i=1}^{n} TP_i + \sum_{i=1}^{n} FN_i} \tag{18}$$

3.2. Comparison with state-of-the-art methods

To validate the efficacy of our voxel-based 3D object detection model, we conducted a comparative analysis with two top-performing 3D object detection models, PointPillars and SECOND, which currently hold the highest rankings for voxel detection accuracy on the KITTI official website. Additionally, we included PointRCNN, known for achieving the highest detection accuracy based on point cloud data, for a comprehensive evaluation. To ensure a fair comparison, we partitioned the training and testing datasets for the models according to the data set classification described in Section 3.1.2. Throughout the training and testing phases, we utilized the two-dimensional Intersection over Union (2D IOU) and three-dimensional Intersection over Union (3D IOU) thresholds as specified in Section 2. The visual representation of AGPNet's verification results is presented in Figure 3. Following the convergence of all three models after their respective training processes, we present the experimental results for the test set in Tables 2 and 3. To provide a comprehensive overview, we computed the average detection results for both two-dimensional and three-dimensional IOU for each object category with varying levels of difficulty. These results are visualized in the scatter plot for the three object types, as depicted in Figure 4.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 27 October 2023 doi:10.20944/preprints202310.1729.v1

From the two-dimensional Intersection over Union (2D IOU) detection results (a, b, c) shown in Figure 4, it is evident that the AGPNet model proposed in this article exhibits a negligible time difference, being only 0.03 seconds slower than the PointPillars model, which boasts the shortest detection time. Moreover, the mean Average Precision (mAP) of AGPNet surpasses that of all other voxel-based three-dimensional object detection models, with the exception of the 'Bicycle' category. Specifically, for relatively small objects like bicycles and pedestrians, AGPNet achieves a 14% and 15% higher mAP than the PointPillars model. When it comes to detecting larger objects such as cars, AGPNet outperforms the PointRCNN model, currently the highest-accuracy point-based method on KITTI, by 4%. On average, AGPNet enhances the detection accuracy of the three object categories by 11% compared to the PointPillars network. While our AGPNet model's detection accuracy on the pedestrian and bicycle datasets remains slightly lower than the PointRCNN network model, which directly operates on point cloud data, it achieves a 15% and 5% lower accuracy on bicycles and pedestrians, respectively, while offering a 47% improvement in detection speed.

Table 2. Two-dimensional IOU results from BEV perspective.

Method	Model	Time/s	Car			Bicycle			Pedestrian		
			Simple	Medium	Difficult	Simple	Medium	Difficult	Simple	Medium	Difficult
Voxel	Pointpillars	0.21	89.45	82.26	80.86	66.99	53.61	49.97	51.90	45.26	41.55
	SECOND	0.31	88.77	82.40	78.91	76.36	63.56	60.07	49.78	45.83	42.71
	AGPNet	0.24	90.93	86.75	84.78	77.99	59.86	55.97	59.14	52.51	47.40
Point	PointRCNN	0.45	89.91	86.49	80.11	88.20	72.80	66.97	64.22	55.47	48.35

Table 3. Three-dimensional IOU results from BEV perspective.

Method	Model	Time/s	Car			Bicycle			Pedestrian		
			Simple	Medium	Difficult	Simple	Medium	Difficult	Simple	Medium	Difficult
Voxel	Pointpillars	0.21	80.62	67.88	65.21	61.95	48.70	45.55	47.49	40.77	36.73
	SECOND	0.31	84.68	71.89	68.31	71.60	59.41	57.62	41.27	36.28	32.92
	AGPNet	0.24	84.94	75.51	72.76	73.60	55.57	51.54	54.64	47.72	42.72
Point	PointRCNN	0.45	89.56	78.24	75.79	86.80	69.99	64.99	61.47	53.61	45.94

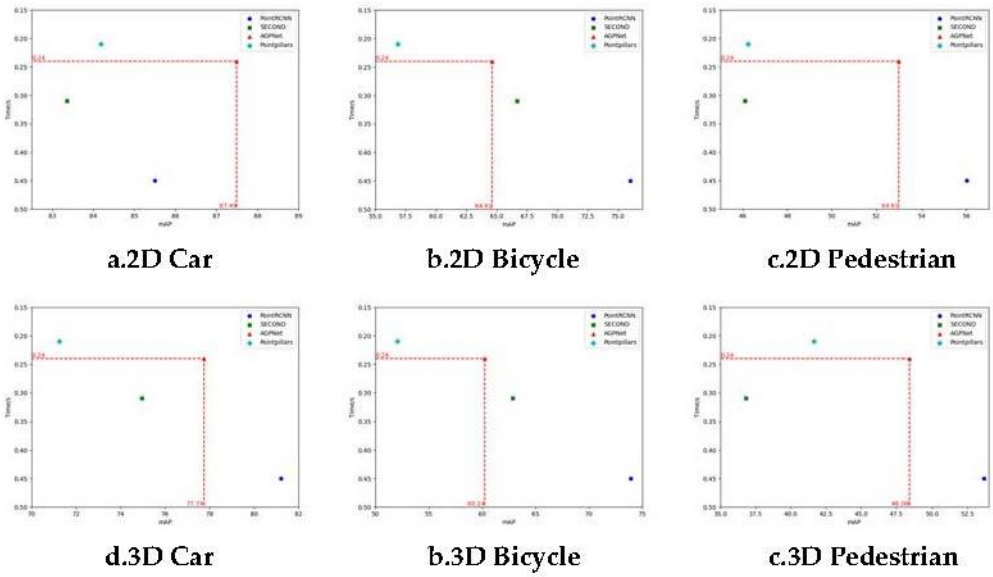

Figure 4. Detection results of three objects in two-dimensional and three-dimensional states of AGPNet's KITTI data set.

Analyzing the three-dimensional Intersection over Union (3D IOU) results presented in Figure 4d–f, we observe that the voxel-based AGPNet model, introduced in this article, exhibits lower

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 27 October 2023 doi:10.20944/preprints202310.1729.v1

detection accuracy when compared to the SECOND network, especially in the context of bicycle object detection. It also demonstrates reduced accuracy in detecting cars and pedestrians. Nonetheless, AGPNet's detection accuracy surpasses the current leading three-dimensional object detection method based on voxel analysis. Specifically, the mean Average Precision (mAP) of the AGPNet network, in the case of relatively small objects such as bicycles and pedestrians, is 16% higher than that of the PointPillars model. When it comes to detecting larger objects, including cars, AGPNet outperforms the PointRCNN model, which is currently the highest-accuracy point-based method on KITTI, by 9% compared to PointPillars. On average, the detection accuracy for all three object types improves by 14% compared to the PointPillars network. It's important to note that although the detection accuracy of our AGPNet model on pedestrian and bicycle datasets remains slightly lower than the PointRCNN network model, which operates directly on point cloud data, it exhibits a 22% and 11% lower accuracy on bicycles and pedestrians, respectively. However, it offers a significant 47% improvement in detection speed.

Figure 4 clearly demonstrates that the AGPNet, a voxel-based 3D object detection model proposed in this article, excels in achieving a balance between detection speed and 3D object detection accuracy. It emerges as the top-performing voxel-based 3D object detection model.

3.3. Multi-scene continuous frame object detection experiment

To enhance the model's ability to detect three-dimensional objects under conditions resembling real-world driving scenarios, this section presents a multi-scene continuous frame object detection experiment. The various road scenes encompass urban roads, apartment, highway, and campus roads, as illustrated in Figure 5. We employed the original KITTI dataset, as explained in Chapter 3, Section 1.2.3, as our source of data. Table 4 provides pertinent details about each continuous frame's point cloud data, with KITTI object tracking annotations available for urban areas, residential zones, and highways. However, it's worth noting that object annotation information is not provided for the campus road scene data. To enable our model to assess its prediction results across all four road conditions, we manually annotated the campus road scene object frames using CloudCompare software. These annotations encompass the dimensions and positions of all object frames requiring detection.

Figure 5. Multiple scene road conditions.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 27 October 2023 doi:10.20944/preprints202310.1729.v1

Table 4. Summary of multi-scenario data information.

Scene	Data	Time/s	Frames	Object Category		
				Car	Bicycle	Pedestrian
City	2011_09_26_drive_0014	32	320	32	4	5
	2011_09_26_drive_0056	30	300	30	1	2
Apartment	2011_09_26_drive_0035	13	137	13	1	2
	2011_09_26_drive_0039	40	401	40	1	2
Highway	2011_09_26_drive_0032	39	396	39	0	0
	2011_09_26_drive_0070	42	426	42	2	2
Campus	2011_09_26_drive_0038	11	116	11	_	_
	2011_09_26_drive_0043	15	151	15	_	_

The outcomes of continuous frame object detection across various scenes, as produced by the AGPNet model, are presented in Figure 6. We meticulously assessed the object detection results of our AGPNet model across these diverse continuous frames, subsequently comparing them with the detection results achieved by the widely adopted PointPillars model in the industry. A comprehensive summary of these results can be found in Table 5.

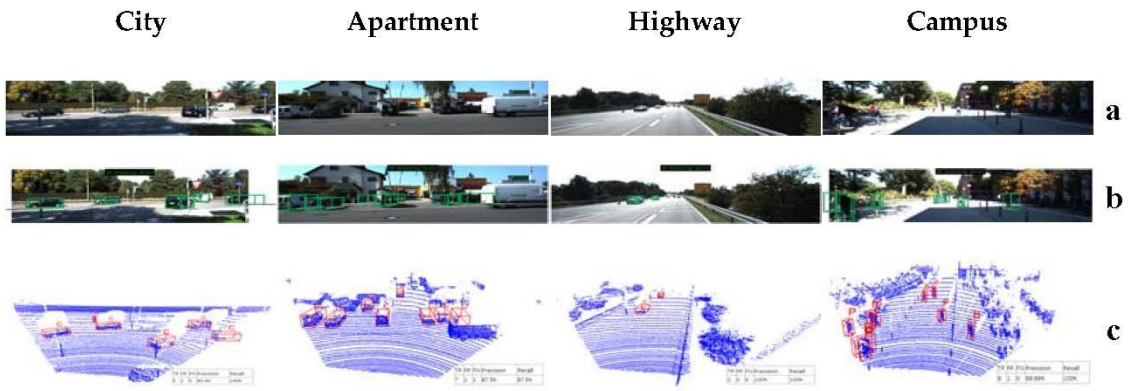

Figure 6. Schematic diagram of the true value and prediction results of point cloud object frames in four scenarios. a Camera image; b Reference ground truth object frame projection result; c AGPNet model prediction result.

Table 5 reveals that when compared to the PointPillars model, our AGPNet model demonstrated a noteworthy 21% increase in average detection accuracy across all four scenarios, accompanied by a 13% improvement in recall rate. It's also important to note that the detection time remains within the same order of magnitude.

Table 5. Comparison of detection results between AGPNet model and Pointpillars model in multiple scenarios.

Scenes	Frames	Model	Time/s	Precision	Recal
City	620	Pointpillars	**0.18s**	62.4%	79.1%
		AGPNet	0.21s	**76.2%**	**87.2%**
Apartment	538	Pointpillars	**0.20s**	67.5%	71.8%
		AGPNet	0.22s	**88.3%**	**89.3%**
Highway	822	Pointpillars	**0.19s**	76.2%	89.6%
		AGPNet	0.21s	**87.2%**	**92.9%**
Campus	267	Pointpillars	**0.18s**	75.1%	82.4%

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 27 October 2023 doi:10.20944/preprints202310.1729.v1

AGPNet	0.21s	**88.1%**	**92.7%**	

3.4. Ablation experiment

To verify the difference between the original PointPillars, which employs the Backbone network structure for feature extraction, and our network structure utilizing the three-dimensional point cloud Dynamic Graph CNN structure for more effective feature extraction, we conducted ablation experiments. These experiments included the original PointPillars model, the GNN-PointPillars model, where we removed the Backbone module and integrated the point cloud dynamic graph convolution network into the original PointPillars model, and the AGPNet model, which incorporates a Data Augmentation module ahead of the GNN-PointPillars model. All three models were trained using the training dataset summarized in section 3.1.2. Once they converged, they underwent testing on the test dataset. We evaluated the 2D mean Average Precision (mAP) and 3D mAP results from the Bird's-Eye View (BEV) perspective, employing the Intersection over Union (IOU) settings as described in previous sections. The results are presented in Tables 6 and 7.

Table 6. Two-dimensional IOU results from BEV perspective.

Model	Time/s	Car			Bicycle			Pedestrian		
		Simple	Medium	Difficult	Simple	Medium	Difficult	Simple	Medium	Difficult
Pointpillars	0.21	89.45	82.26	80.86	66.99	53.61	49.97	51.90	45.26	41.55
GNN-Pointpillars	0.24	91.45	85.01	82.30	76.36	63.56	60.07	49.78	45.83	42.71
AGPNet	0.24	90.93	86.75	84.78	88.20	72.80	66.97	64.22	55.47	48.35

Table 7. Three-dimensional IOU results from BEV perspective.

Model	Time/s	Car			Bicycle			Pedestrian		
		Simple	Medium	Difficult	Simple	Medium	Difficult	Simple	Medium	Difficult
Pointpillars	0.21	80.62	67.88	65.21	61.95	48.70	45.55	47.49	40.77	36.73
GNN-Pointpillars	0.24	83.74	71.91	67.37	73.60	59.41	57.62	41.27	36.28	32.92
AGPNet	0.24	84.94	75.51	72.76	86.80	69.99	64.99	61.47	53.61	45.94

We computed the average performance across the three difficulty levels for each object category and plotted a line graph illustrating the two-dimensional and three-dimensional Intersection over Union (IOU) detection results from the Bird's-Eye View (BEV) perspective. This graph captures the performance of the three models across all three categories, and it is presented in Figure 7.

Figure 7 illustrates that after replacing the Backbone module of the PointPillars model with a Dynamic Graph CNN module, transforming it into the GNN-PointPillars model, we observed a slight decrease in bicycle detection accuracy. However, there were notable improvements in the detection accuracy for cars and pedestrians, with pedestrian detection accuracy showing a particularly significant increase. Further enhancing the model by adding the Data Augmentation module to the GNN-PointPillars configuration, creating the AGPNet model, resulted in improved detection accuracy for all three object categories. Notably, the accuracy of bicycle and pedestrian detection saw significant enhancements. The two-dimensional Intersection over Union (IOU) and three-dimensional IOU results revealed an increase of approximately 20% and 10%, respectively. Therefore, our data augmentation module effectively enhances the quality of model detection. Tables 6 and 7 also indicate that the AGPNet model improves the detection accuracy for smaller three-dimensional objects when maintaining a detection time within the same order of magnitude as the PointPillars model.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 27 October 2023 doi:10.20944/preprints202310.1729.v1

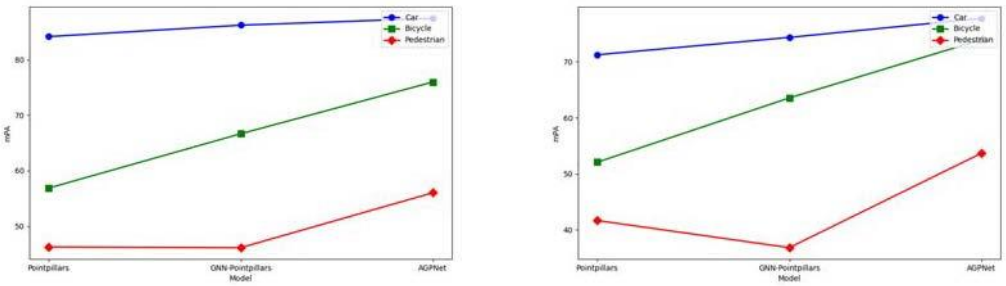

Figure 7. Line chart of detection results of Pointpillars, GNN-Pointpillars and AGPNet models from BEV perspective, (a) two-dimensional IOU detection results, (b) three-dimensional IOU detection results.

Through ablation experiments, we observed that the addition of a data enhancement module and a GNN (Graph Neural Network) graph convolution module to the original PointPillars model significantly enhances the quality of model detection while maintaining a detection time within the same order of magnitude as the PointPillars model.

4. Conclusions

In this paper, we introduce AGPNet, a three-dimensional point cloud data target detection network structure that amalgamates Data Augmentation, Dynamic Graph CNN, Pillar Feature Net, and Detection Head(SSD). Recognizing the challenge of achieving accurate detection in the presence of occluded or small objects within the PointPillars network model, we attribute this to limitations in feature extraction following the conversion of point cloud data into pseudo images. Therefore, we present a creative solution by integrating Data Augmentation and Dynamic Graph CNN modules into the PointPillars model. Experimental results demonstrate that our proposed AGPNet network model achieves a detection accuracy 6-7 percentage points higher than that of the PointPillars network model, while maintaining detection time within the same order of magnitude.

While our 3D point cloud voxel-based object detection method may not achieve the same detection accuracy as the direct object detection from point cloud data, it excels in detection speed. This attribute makes it highly suitable for real-time object detection needs in actual driving environments. In the future, we intend to further enhance detection accuracy and speed by integrating additional data augmentation and point cloud object feature extraction modules as outlined in this article.

References

1. Chen X, Ma H, Wan J, et al. Multi-view 3d object detection network for autonomous driving[C]. Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. IEEE, 2017: 1907-1915.
2. Ku J, Mozifian M, Lee J, et al. Joint 3d proposal generation and object detection from view aggregation[C]. 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2018: 1-8
3. Zhou Y, Sun P, Zhang Y, et al. End-to-end multi-view fusion for 3d object detection in lidar point clouds[C]. Conference on Robot Learning. PMLR, 2020: 923-932.
4. Shi S, Wang X, Li H. PointRCNN: 3d object proposal generation and detection from point cloud[C]. Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. IEEE, 2019: 770-779.
5. Qi C R, Liu W, Wu C, et al. Frustum PointNets for 3d object detection from rgb-d data[C]. Proceedings of the IEEE conference on computer vision and pattern recognition. IEEE, 2018: 918-927.
6. Shi W, Rajkumar R. Point-GNN: Graph neural network for 3d object detection in a point cloud[C]. Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. IEEE, 2020: 1711-1719.
7. Qi C R, Su H, Mo K, et al. PointNet: Deep learning on point sets for 3d classification and segmentation[C]. Proceedings of the IEEE conference on computer vision and pattern recognition. IEEE, 2017: 652-660.

8. Qi C R, Yi L, Su H, et al. PointNet++: Deep hierarchical feature learning on point sets in a metric space[J]. Advances in neural information processing systems, 2017:1-14.
9. Liu W, Anguelov D, Erhan D, et al. SSD: Single shot multibox detector[C]. Computer Vision–ECCV 2016: 14th European Conference, Amsterdam. Springer International Publishing, 2016: 21-37.
10. Zhou Y, Tuzel O. VoxelNet: End-to-end learning for point cloud based 3d object detection[C]. Proceedings of the IEEE conference on computer vision and pattern recognition. IEEE, 2018: 4490-4499.
11. Yan Y, Mao Y, Li B. SECOND: Sparsely embedded convolutional detection[J]. Sensors, 2018, 18(10): 1-17.
12. Lang A H, Vora S, Caesar H, et al. Pointpillars: Fast encoders for object detection from point clouds[C]. Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. IEEE, 2019: 12697-12705.
13. Wang Y, Sun Y, Liu Z, et al. Dynamic graph cnn for learning on point clouds[J]. ACM Transactions on Graphics (tog), 2019, 38(5): 1-12.
14. Zheng W, Tang W, Jiang L, et al. SE-SSD: Self-ensembling single-stage object detector from point cloud[C]. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. IEEE, 2021: 14494-14503.
15. Girshick R. Fast R-CNN[C]. Proceedings of the IEEE international conference on computer vision. IEEE, 2015: 1440-1448.
16. Jang E, Gu S, Poole B. Categorical reparameterization with gumbel-softmax[J]. arXiv preprint arXiv:1611.01144, 2016:1-13.