2019 IEEE 8th Data Driven Control and Learning Systems Conference
May 24-27, 2019, Dali, Yunnan Province, China

Comparison of 3D Object Detection Based on LiDAR Point Cloud

Haoran Li[1], Xiaolei Zhou[2], Yaran Chen[1], Qichao Zhang[1], Dongbin Zhao[1], Dianwei Qian[2]

1. The State Key Laboratory of Management and Control for Comlpex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing 100190 and The University of Chinese Academy of Sciences, Beijing 100049, P. R. China

2. School of Control and Computer Engineering, North China Electric Power University, Beijing 102206, P. R. China
E-mail: lihaoran2015@ia.ac.cn, zhxl0112@163.com, chenyaran2013@ia.ac.cn, zhangqichao2014@ia.ac.cn, dongbin.zhao@ia.ac.cn, dianwen.qian@ncepu.edu.cn

Abstract: 3D object detection and scene understanding are the key technologies for autonomous driving scenarios. Due to the differences in configuration and datasets used by each 3D object detection algorithm, it is difficult to evaluate the performance of each method. In this work, we provide a comparison of the advanced 3D object detection networks based on LiDAR point cloud in recent two years and analyze each network structure in detail. For the open-sourced networks, we reproduce them on KITTI dataset benchmark with following their original algorithms. Meanwhile, in order to provide more powerful results, we also utilize nuScenes dataset to retrain the networks as mentioned above. The experimental results show that the performance of the networks with point cloud and images as input is better than that of a single input network.

Key Words: LiDAR point cloud, 3D object detection, Autonomous driving, Deep learning

1 Introduction

Perception algorithm is a key point for the autonomous system. At present, the processing methods for 2D images have been relatively mature, such as object detection [1–5] and instance segmentation [6, 7]. RCNN [1], as the first deep learning model in the field of object detection, greatly improves the accuracy of the task. It establishes the status of CNN in 2D object detection. SPPNet [2] and Fast-RCNN [3] reduce the operating time of the prior network. But the expensive computation is really a bottleneck until the appearance of Faster-RCNN [4]. However, the imaging process of the camera is projecting 3D space into 2D view. This process will lose a lot of spatial information, and the 2D images can no longer satisfy people's needs [8].

LiDAR emits the laser beam to provide reliable depth information. The data obtained by LiDAR is point cloud which can be used to detect the object accurately. It contains the position and the reflection intensity of each point. What's more, it has unique advantages for shape representation and the complex scene of the object. Compared to camera images which are easily damaged by illumination, point cloud is robust for different lighting conditions. However, LiDAR point cloud is a disordered set of discrete points distributed in 3D space, which is the biggest difference between 3D point cloud data and 2D image data. In addition, 2D images have high-density pixels, but point cloud is very sparse points. For these reasons, 2D object detection methods cannot be directly applied to 3D point cloud. So how to effectively process point cloud has become an important issue.

Point cloud are commonly used for object detection, classification [9] and SLAM [10]. In the past two years, a lot of point cloud processing methods have emerged. They can be divided into two main categories. One is using point cloud only without other sensor information. PointNet [11] proposed by Qi et al. is the primary network to directly consume the raw point cloud and it is really a major breakthrough in point cloud processing. But it ignores the local features of the point cloud, leading to unsatisfied accuracy. PointNet++ [12] makes up for the deficiency of PointNet and pays attention to the extraction of local features. PointSIFT [13] is inspired by scale-invariant feature transform and combines set abstraction module and feature propagation module proposed in PointNet++ which achieves better performance. Shandong University proposes PointCNN [14]. This network achieves point cloud classification by learning a χ-transformation from the point cloud. This kind of methods usually carries out classification and semantic segmentation tasks for indoor environment and CAD data. Furthermore, some networks based only on point cloud can also be applied to 3D object detection, such as Complex-YOLO [15], VoxelNet [16], etc. Another kind of methods combines the point cloud data with 2D image data to complete the special task, such as AVOD [17], MV3D [18], F-PointNet [31]. This kind of models is always used to 3D object detection in autonomous driving scenarios. Moreover, we would like to focus on these networks in the next context.

In this paper, we pay attention to state-of-the-art methods of performing 3D object detection based on LiDAR point cloud. In order to access the performance of different networks, we conduct retrained experiments both on KITTI [20] and nuScenes [21] dataset for the publicly available methods. Considering the completeness of the comparison, we adopt the experimental results on the KITTI official website [16]. The experiments are used to make the conclusion more convincing since it is really necessary to use the same criteria to evaluate the networks.

The key contributions of our work are as follows:

- We summarize point cloud processing methods, analyze and compare the submodule of state-of-the-art 3D object detection algorithms in recent two years in detail.
- We reproduce the algorithms on KITTI and nuScenes

This work is supported partly by Beijing Science and Technology Plan under Grants Z181100004618003, Beijing Natural Science Foundation under Grant No. L172054, and No. GJHZ1849 International Partnership Program of Chinese Academy of Sciences.

978-1-7281-1454-5/19/$31.00 ©2019 IEEE

dataset, and compare the performance.

- We analyze the results of the comparison and explain the pros and cons of each algorithm.

2 Related Work

LiDAR point cloud has a special data structure. Compared to regular data such as 2D image data and natural language data, point cloud is a set of disordered vectors which greatly increases the difficulty of deep neural network learning its features. There is a very strong relationship between each point, and each point coupled with its neighbors can be viewed as a subset, so each point cannot be treated separately. Point cloud has strong 3D features. Consequently, lots of problems should be considered when processing the point cloud.

The previous methods for point cloud processing are mainly categorized as four classes. 1) Applying 3D convolution neural networks on voxels which are from point cloud voxelization. The basic idea of this method is to extract features manually. Firstly, the 3D space should be discretized and regularized, then each voxel after voxelization should be treated as a processing unit. The features of each cell are encoded by point cloud within the cell. Hand-crafted features can yield satisfactory detection results when the rich 3D information is available. However, when the scene is complicated, the experimental precision is poor. Furthermore, volumetric representation has not been widely used due to data sparsity and computation cost of 3D convolution. 2) Converting 3D point cloud into several 2D images by multi-view projection and employing 2D CNNs to do shape classification [25]. Relying on the mature 2D convolution networks, this method has good performance in detection tasks. However, this method drops some spatial information of point cloud. It is hard to apply this method to point cloud segmentation task. 3) Using spectral CNNs, [26, 27] firstly utilizes spectral CNNs on meshes. However, this method is limited by a variety of grids, and it is not easy to extend them to non-isometric shapes. 4) Applying feature-based DNNs, [28, 29] first convert 3D data into a vector, then classify them using a fully connected network after extracting shape features. However, this method is restricted by the extracted features.

3 Network Structure

This section will introduce several common networks from data preparation, fusion methods of models, loss function, etc.

3.1 Data preparation

At present, there are two main ways to pre-process point cloud. One is the projection-based method, which projects point cloud data onto a plane. The other is the raw point cloud-based method. In other words, point cloud is processed directly without any other operations.

3.1.1 Projection-based Method

Projection-based method aims to project 3D LiDAR point cloud data onto one or more 2D planes, then the projected images can be treated as 2D images. For instance, AVOD, Complex-YOLO, SBNet [44], BirdNet [40], RT3D [39] and PIXOR [41] all adopt the bird's eye view (BEV) projection. In addition to the BEV map, MV3D also has a front view projection. LMNet [43] just uses a front view which may cause a relative lower detection accuracy but really less computational cost. Both front view and bird's eye view are efficient and compact. Compared with the front view, each object on the BEV map has a lower probability of occlusion and on this account, it is widely adopted. The BEV map is always encoded by height, intensity and density and each piece of point cloud data is cropped to a fixed dimension. The resolution of the BEV map is approximately 0.1 meter. VeloFCN [30] is the first one to project the point cloud data to an image plane coordinate system. However, the main problem of this method is information loss. The projection from 3D to 2D causes information compression on the spatial structure. It is inevitable to lose geometric information. But it is still a successful conversion method of irregular point cloud data processing.

3.1.2 Raw Point Cloud-based Method

Recently, a novel type of network architecture has been developed. It consumes raw point cloud without converting it to other formats mentioned above. Most of these methods are based on PointNet variants to extract point cloud features which really implement end-to-end learning. VoxelNet partitions the space of point cloud into many voxels, in which each voxel is encoded by features of points within it. F-PointNet is similar to F-PC-CNN [36]. These two networks all take an image as input. The first step generates 2D bounding boxes from a 2D image. Then according to the known camera projection matrix and 2D bounding boxes, a 3D search space can be obtained. The operation is just performed directly in the point cloud area of the 3D bounding boxes. PointFusion [33] directly uses PointNet for point cloud processing branch. Direct processing retains the original information of the point cloud to the utmost extent. Besides, the network structure is always simple. The main problem with this method is that PointNet structure has poor local feature perception for point cloud. This kind of network is mainly used for point cloud classification and semantic segmentation tasks of a simple scene such as CAD models–ModelNet40 [22] and indoor scenes, such as ScanNet [32]. But there are relatively few applications in the field of 3D object detection in actual road scene.

3.2 Fusion Mode

Although point cloud data has really accurate geometric information, the data is very sparse. The image, as high-resolution data, has rich texture features that distinguish objects one by one. The experimental results show that the detection accuracy of the network using point cloud and image fusion is commonly higher than that using point cloud as input only. Therefore, the integration of the two data is really hot topics of recent researches. There are many ways to integrate 3D point cloud data with image data, which can be divided into two main categories.

3.2.1 Feature Level Fusion

This method is performing a specific combination of the depth features extracted from the 3D point cloud data and the corresponding image regions. MV3D generates 3D proposals from the point cloud BEV map, then projectes it onto the LiDAR front view and RGB images to get proposals from three views and performes feature fusion after performing ROI Pooling operations on all three proposals. While AVOD fuses the two regions of interest which are obtained from the given 3D anchor projecting to the BEV map and RGB image. PointFusion uses PointNet and ResNet [34] for point cloud processing and image feature extraction, respectively. The advantage of this method is that it can fully extract the optimal feature representation under the current spatial structure from the current sensor data. The key to this method is how to combine the two features to achieve the complementary effect of the two data advantages. The current method mostly uses concatenation and mean. This fusion method requires precise time synchronization between the LiDAR sensor and the camera sensor.

3.2.2 Object Level Fusion

Object level fusion always contains two stages. Firstly, a local result is obtained from the first step. Then the second step takes advantage of the result. For example, F-PointNet first adopts a mature 2D image detection method to detect 2D bounding boxes, then the 2D bounding boxes are lifted to frustums that are similar to the 3D candidate boxes. The network is also sensitive to the synchronism of the two sensors. RoarNet [35] uses an obvious two-stage detection. The first stage detects 2D bounding boxes on the images and predicts the corresponding 3D regional information. The second stage performs object detection before extracting the features of the 3D point cloud data of the corresponding region. F-PC-CNN is a network that can combine 3D point cloud with any 2D object detection network. It firstly uses PC-CNN [37] to detect the 2D bounding boxes, then projects point cloud data onto the plane of the image, and performs subsequent processing on the point cloud projection data corresponding to the 2D bounding boxes region to obtain the final detection result. The problem with this method is that the sensor's field of view is different due to the different installation height and occlusion. For example, there is a car in front that blocks the image field of view, making it difficult to see the vehicles behind it, and LiDAR sensor is often installed at a high altitude and can scan most vehicles. In this case, the candidate area based on the image often misses, and the miss is irreversible.

3.3 Region Proposal Network

Region proposal network (RPN) is proposed in the Faster-RCNN network which is widely used in 2D image object detection and plays an important role in detection tasks. The main reason is that it simplifies the object detection problem into a classification problem, reduces the difficulty of the problem, makes the network training less difficult, and makes the network more purposeful. It is for this reason that many 3D point cloud-based object detection algorithms also adopt this structure. The application of RPN is mainly in 2D projection images. When the features of the projected image are extracted, the RPN is used to find the region of interest. 3D RPN is proposed in [38] for 3D object detection of RGB-D images and in each network, 3D RPN is modified according to the specific network structure based on Faster-RCNN. MV3D is the first object detection algorithm using 3D RPN for autonomous driving scenarios. MV3D matches each pixel in the BEV map with the anchor, and then sends each anchor to the RPN to generate 3D proposals. While AVOD uses the RPN on the merged features, called Multimodal Fusion RPN. Complex-YOLO proposes a special Euler-Region-Proposal Network (E-RPN) to locate the object by adding a fictitious and a true score to the regression network. In VoxelNet, RT3D, and BirdNet, RPN is used after the convolution operation of the point cloud features. In the RoarNet network, both the RoarNet_2D and RoarNet_3D phases use the RPN. At present, RPN still holds a very important position in the field of object detection.

3.4 Loss Function

The loss function is the learning goal of the whole network, guiding the convergence direction of network weight, which is crucial for the performance of the final algorithm. There are several different methods for the design of the loss function of 3D object detection.

One of the most straightforward methods is to fit the coordinates of the eight corners of the 3D bounding boxes with a deep network. This kind of loss function is used in methods such as MV3D, VeloFCN, and PointFusion. The problem with this approach is that the 8 fitted corners may not conform to the constraints of the rectangle, causing the final 3D bounding boxes irregular.

Another type of methods is to approach the center coordinates, shape information of the bounding boxes and the rotation angle. Both VoxelNet and RT3D uses center points, length, width, height, and z-axis yaw angle of the 3D bounding boxes as the network learning goal. F-PC-CNN uses the center point of the 3D bounding box, the coordinates of the three lower left corners and the width of the box as the fitting goals. Both VoxelNet and F-PC-CNN learned 7-dimensional vectors, which greatly reduces the fitting complexity and computation compared to the eight corner method. The author of PIXOR believes that the objects are on the ground and will not fly into the air which is a very reasonable explanation. Therefore the detected objects are in the form of 2D bounding boxes on the bird's eye view. The regression goal is the center coordinate of the 2D bounding box, the size of the box and its rotation angle. In other words, it needs to fit the 5-dimensional vector, which greatly reduces the cost computation compared to fitting a 24-dimensional vector of 8 corners.

There is also a method of combining the two methods mentioned above. For example, AVOD utilizes the coordinates of the bottom four points and height of the upper and lower bottom surfaces from the ground plane as the fitting target of the network and calculates the variance of the loss function.

There are also some special ways to calculate losses. For example, F-PointNet regresses the distance of the corresponding eight points between the predicted box and the

Table 1: Point cloud processing method properties

Method	Inputs	Proprocessing	RPN	Regression loss	Data Augmentation
AVOD	LiDAR+image	projection	$\surd$	4corner+2height	$\times$
MV3D	LiDAR+image	projection	$\surd$	8corner	$\times$
F-PointNet	LiDAR+image	original	$\times$	8corner distance	2d scale; 3d resample,flip,shift
PointFusion	LiDAR+image	original	$\times$	8corner	$\times$
Complex-YOLO	LiDAR	projection	$\surd$	YOLOv2+E-distance	$\times$
VoxelNet	LiDAR	original	$\surd$	location+size+rotation	3d pertubation,rotation,scale
VeloFCN	LiDAR	projection	$\times$	8corner	3d zoom,rotation
RT3D	LiDAR	projection	$\surd$	location+size+rotation	$\times$
PIXOR	LiDAR	projection	$\times$	location+size+rotation	3d zoom
LMNet	LiDAR	projection	$\times$	8corner	3d rotation
F-PC-CNN	LiDAR+image	projection	$\times$	location+1corner+rotation	$\times$
BirdNet	LiDAR	projection	$\surd$	size	$\times$
SBNet	LiDAR	projection	$\times$	not mentioned	$\times$
RoarNet	LiDAR+image	projection	$\surd$	location+rotation+size	$\times$

ground truth. Complex-YOLO uses its unique E-RPN coding method. The loss function adopts the combination of the loss in YOLOv2 [42] and Euler loss to perform network regression training.

3.5 Data Augmentation

In deep learning, when the size of the dataset is not large enough, the network cannot learn enough features. Data augmentation is often used to make up for the shortcomings. How to carry out effective data augmentation is an important part of improving algorithm performance. For networks which take point cloud and images as input, these two kinds of data can be extended simultaneously. The main methods of data augmentation are as follows.

The image data augmentation approaches based on external attributes include : 1) Randomly rotating the image at a certain angle to change the orientation of the image content; 2) Flipping the image horizontally or vertically; 3) Zooming in or zooming out the image by a certain ratio; 4) Panning the image in a certain way on the image plane; You can specify the pan range and pan step in a random or artificially defined way, panning horizontally or vertically or changing the position of the image content; 5) Enlarging or reducing the content of the image according to a specified scale factor or changing the degree of blurring of the image content.

The image data augmentation approaches based on internal attributes include : 1) Performing an exponential operation on the S and V components of each pixel to increase the illumination variation in the HSV color space of the image; 2) Randomly perturbating each RGB pixel of the image, the commonly used noise modes are salt-and-pepper noise and Gaussian noise; 3) Performing principal component analysis (PCA) in the RGB color space of the training dataset pixel to transform each pixel of each image.

For the augmentation of point cloud data, the methods based on external attributes including rotation, flipping, scaling, and panning are also applicable to point cloud. In addition, resampling operation of the point cloud data is used to extract different features of the same point cloud data.

Most 3D object detection algorithms use more than one way for data augmentation of which some methods are randomly selected for this task. For instance, in F-PointNet, resampling, flipping, and panning are randomly used to augment point cloud. VoxelNet uses the methods of adding perturbation, rotation and scaling. In addition to this, the special data augmentation processing in F-PointNet also lies in the generation of the 2D bounding boxes in 2D image detection. It uses the random translation and scaling to extend the 2D ground truth boxes to generate a variety of different size 2D boxes for the generation of the frustum. Both the VeloFCN and the PIXOR uses a zooming pattern to augment the point cloud. Besides, the rotation method is also used in VeloFCN. LMNet randomly rotates $[-15°–15°]$ of point cloud data around the z-axis. The basic structure of each network is summarized in Table 1.

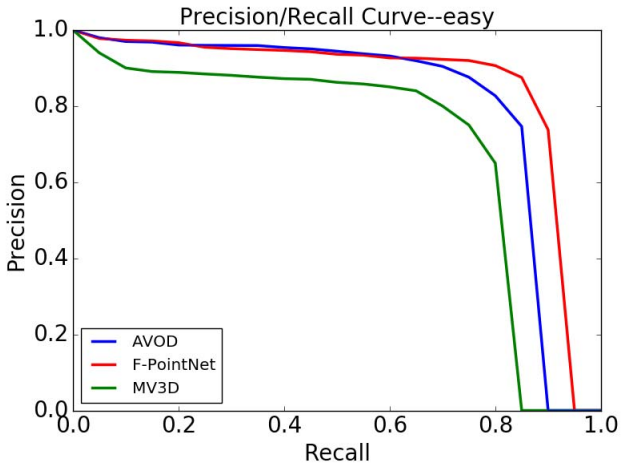

Fig. 1: Precision vs Recall: class car of easy level on KITTI

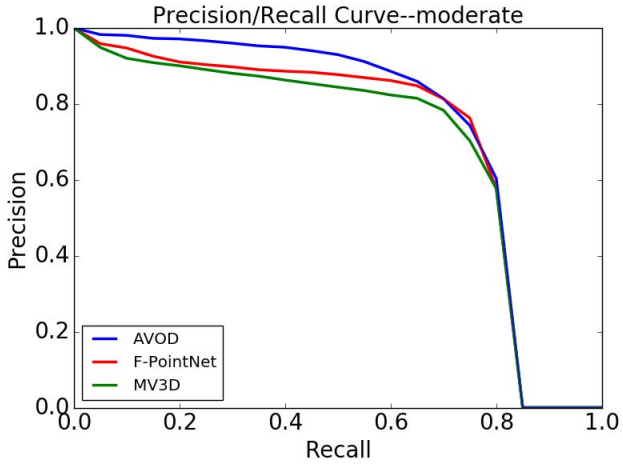

Fig. 2: Precision vs Recall: class car of moderate level on KITTI

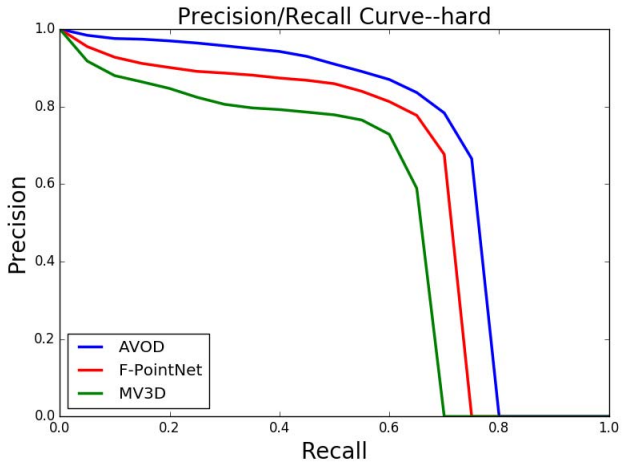

Fig. 3: Precision vs Recall: class car of hard level on KITTI

4 Experimental Results and Discussion

We just focus on the car detection task. We follow the official KITTI evaluation protocol, where the Intersection of Union (IoU) threshold is set to 0.7 for the class car. We compare AVOD, MV3D, F-PointNet using the average precision (AP) metric. In Fig. 1, Fig. 2, Fig. 3, Fig. 4, the area under the precision vs recall curve is AP. Therefore, the larger the area enclosed by the curve and the two axes is, the performance is better.

In this paper, we refer some results on KITTI official website. Table 2 shows the AP of advanced 3D object detection algorithms in recent two years in 3D space for cars, pedestrians, and cyclists, and KITTI follows the easy, medium, hard difficulty classification. In order to make the experimental results more convincing and obtain quality experimental results, we retrain the open-sourced networks – AVOD, MV3D, F-PointNet using KITTI and nuScenes datasets both on Titan Xp server. The experimental results for class car show in Table 3.

4.1 Evaluation on KITTI Validation Set

We follow KITTI official website and evaluate on three difficulty regimes: easy, moderate, hard. Easy one means the maximum truncation for an object is 15%, and 30%, 50% for moderate and hard, respectively. Combined with Table 2, we can see, no matter what kind of methods, the less occlusion the target is, the better the effect is. With less occlusion, more features can be extracted. So the network has better performance. What's more, the effect of class car is obviously better than that of pedestrians and cyclists. The reason is that cars occupy a large proportion of the pixels in the image and there are more points in point cloud data. As we can see in Fig. 1, for the class car, when less occlusion, F-PointNet performs better than AVOD and MV3D. But in the case of more occlusion, as Fig. 2 and Fig. 3, the performance of F-PointNet is inferior to that of AVOD. F-PointNet uses an obvious two-stage detection network. In the first stage, it uses 2D object detection to obtain the 2D bounding boxes. 2D object detection method has been relatively mature and has a better effect on targets with less occlusion, but for targets with more truncation, it often misses, which is irreversible. This is also the reason why the accuracy of F-PointNet is slightly lower in moderate and hard.

There are some similarities in MV3D and AVOD. Both of them are the fusion of point cloud data with image data. For the class car, AVOD performs better than MV3D in three cases with different degrees of difficulty, and the processing speed of each frame is faster. Both AVOD and MV3D use RPN. In the MV3D network, RPN is used in the bird's eye view of the point cloud while AVOD uses multimodal fusion RPN. The multimodal fusion RPN merges RGB image and bird's eye view image of point cloud by 3D anchor and gathered more precise location information and context features which make the method more accurate. 3D proposals are projected into LiDAR front view images and the RGB images. Deep fusion features are obtained from the region of interests (ROIs). This step requires to calculate a large number of parameters. AVOD uses feature pyramid fusion. The network parameters are 16% [17] of MV3D. There are fewer parameters to calculate leading to that the network runs faster.

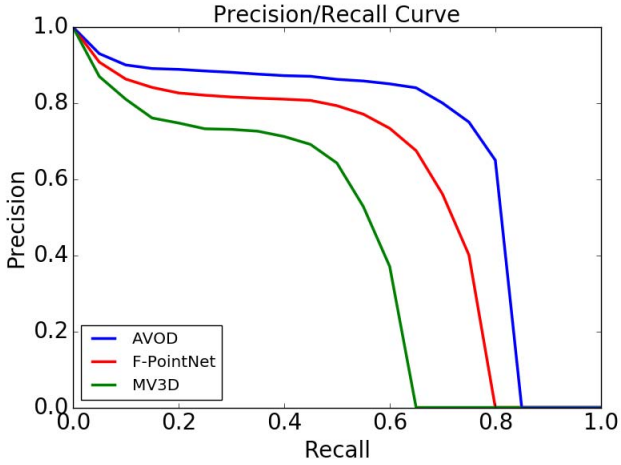

Fig. 4: Precision vs Recall: class car on nuScenes

4.2 Evaluation on Nuscenes Validation Set

The nuScenes dataset is released by NuTonomy [45] and Scale [46], which is a special dataset for autonomous driv-

ing. The total amount of data in nuScenes is larger than that in KITTI. It includes 17184 frames. We split the 17184 frames into a training and a validation set at 1:1. Besides, we just modify the nuScenes dataset according to the format of KITTI, and we do not classify the truncation level. In F-PointNet, the author doesn't mention the 2D object detection method, so we employ Faster-RCNN network to do the 2D detection task. For AVOD and MV3D, we strictly follow the original network settings. As we can see in Fig. 4, AVOD has the best performance, following by F-PointNet. In terms of running speed, AVOD runs 5.7 frames per second. Although slower than running on KITTI, but compared with the other two networks, it is still the fastest running network. This result also explains the structure of the network. But perhaps because of the differences in datasets, the overall performance on nuScenes is not as good as that on KITTI. The specific reasons need us to go further.

From Table 2, we can clearly see that the multi-view sensory-fusion model for 3D object detection is better than a single-sensor model in terms of detection accuracy. But regrettably, the codes of those models aren't released. Hence we can't verify it personally. But through our validation of the three fusion networks of AVOD, MV3D, F-PointNet, we can generally get the following results: AVOD works best, MV3D works poorly.

5 Conclusions

In this work, we analyze the advanced LiDAR-based 3D object detection algorithms on autonomous driving scenarios in recent two years. Moreover, the usage of network submodules is compared in detail, including network input, preprocessing, RPN, regression loss and data augmentation. We retain open-sourced algorithms both on KITTI and nuScenes dataset. By combining the experimental results of the KITTI official website, the results show that the performance of multi-view sensory-fusion model is better. In the task of 3D object detection, multi-sensor fusion may become the main research direction in the future.

References

[1] Girshick, Ross, *et al.* "Rich feature hierarchies for accurate object detection and semantic segmentation", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2014.

[2] He, Kaiming, *et al.* "Spatial pyramid pooling in deep convolutional networks for visual recognition." *European Conference on Computer Vision (ECCV)*. 2014,37(9):1904-16.

[3] Girshick, Ross. "Fast r-cnn." *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 2015.

[4] Ren, Shaoqing, *et al.* "Faster r-cnn: Towards real-time object detection with region proposal networks." *Proceedings of the IEEE Conference on Neural Information Processing Systems (NIPS)*. 2017,39(6):1137-1149.

[5] Chen Y, Zhao D, Lv L, *et al.* "Multi-task learning for dangerous object detection in autonomous driving." *Information Sciences.* 2018,432:559-571.

[6] He, Kaiming, *et al.* "Mask r-cnn." *Proceeding of IEEE International Conference on Computer Vision (ICCV)*. 2017,PP(99):1-1.

[7] Long, Jonathan, Evan Shelhamer, and Trevor Darrell. "Fully convolutional networks for semantic segmentation." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2014,39(4):640-651.

[8] Gumhold, Stefan, Xinlong Wang, and Rob S. MacLeod. "Feature extraction from point clouds." *IMR.* 2001.

[9] Zhao D, Chen Y, Lv L., *et al.* "Deep reinforcement learning with visual attention for vehicle classification", *Proceedings of the IEEE Transactions on Cognitive and Developmental Systems (TCDS)*. 2017,9(4):356-367.

[10] H. Li, Q. Zhang, D. Zhao., "Comparison of methods to efficient graph SLAM under general optimization framework", *2017 32nd Youth Academic Annual Conference of Chinese Association of Automation (YAC)*. 2017,321-326.

[11] Qi, Charles R., *et al.* "Pointnet: Deep learning on point sets for 3d classification and segmentation." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017.

[12] Qi, Charles Ruizhongtai, *et al.* "Pointnet++: Deep hierarchical feature learning on point sets in a metric space." *Proceedings of the IEEE Conference on Neural Information Processing Systems (NIPS).* 2017.

[13] Jiang, Mingyang, Yiran Wu, and Cewu Lu. "PointSIFT: A SIFT-like network module for 3D point cloud semantic segmentation." *arXiv preprint arXiv:1807.00652.* 2018.

[14] Li, Yangyan, *et al.* "PointCNN: convolution on X-transformed points." *Proceedings of the IEEE Conference on Neural Information Processing Systems (NIPS).* 2018.

[15] Simon, Martin, *et al.* "Complex-YOLO: Real-time 3D object detection on point clouds." *arXiv preprint arXiv:1803.06199.* 2018.

[16] Zhou, Yin, and Oncel Tuzel. "Voxelnet: End-to-end learning for point cloud based 3d object detection." *arXiv preprint arXiv:1711.06396.* 2017.

[17] Ku, Jason, *et al.* "Joint 3d proposal generation and object detection from view aggregation." *arXiv preprint arXiv:1712.02294.* 2017.

[18] Chen, Xiaozhi, *et al.* "Multi-view 3d object detection network for autonomous driving." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017.

[19] "Kitti 3d object detection benchmark," *http://www.cvlibs.net/datasets/kitti/eval_object.php?obj_benchmark=3d*, accessed: 2018-12-24.

[20] Geiger, Andreas, Philip Lenz, and Raquel Urtasun. "Are we ready for autonomous driving? the kitti vision benchmark suite." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2012.

[21] "nuScenes 3d object detection benchmark", *https://www.nuscenes.org/data-collection*, accessed: 2018-11-12.

[22] Wu, Zhirong, *et al.* "3d shapenets: A deep representation for volumetric shapes." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015.

[23] Maturana, Daniel, and Sebastian Scherer. "Voxnet: A 3d convolutional neural network for real-time object recognition." *Proceedings of the IEEE conference on Intelligent Robots and Systems (IROS)*. 2015.

[24] Qi, Charles R., *et al.* "Volumetric and multi-view cnns for object classification on 3d data." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).* 2016.

[25] Su, Hang, *et al.* "Multi-view convolutional neural networks for 3d shape recognition." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015.

[26] Bruna, Joan, *et al.* "Spectral networks and locally connected networks on graphs." *arXiv preprint arXiv:1312.6203.* 2013.

[27] Masci, Jonathan, *et al.* "Geodesic convolutional neural networks on riemannian manifolds." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015.

[28] Fang, Yi, *et al.* "3d deep shape descriptor." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015.
[29] Guo, Kan, Dongqing Zou, and Xiaowu Chen. "3d mesh labeling via deep convolutional neural networks." *ACM Transactions on Graphics (TOG)*. 2015,35(1):1-12.
[30] Li, Bo, Tianlei Zhang, and Tian Xia. "Vehicle detection from 3d lidar using fully convolutional network." *arXiv preprint arXiv:1608.07916*. 2016.
[31] Qi, Charles R., *et al.* "Frustum pointnets for 3d object detection from rgb-d data." *arXiv preprint arXiv:1711.08488.* 2017.
[32] Dai, Angela, *et al.* "ScanNet: Richly-Annotated 3D reconstructions of indoor scenes." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017.
[33] Xu, Danfei, Dragomir Anguelov, and Ashesh Jain. "Pointfusion: Deep sensor fusion for 3d bounding box estimation." *arXiv preprint arXiv:1711.10871.* 2017.
[34] He, Kaiming, *et al.* "Deep residual learning for image recognition." *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR).* 2016.
[35] Shin, Kiwoo, Youngwook Paul Kwon, and Masayoshi Tomizuka. "RoarNet: A robust 3D object detection based on regiOn approximation refinement." *arXiv preprint arXiv:1811.03818*. 2018.
[36] Du, Xinxin, *et al.* "A general pipeline for 3d detection of vehicles." *arXiv preprint arXiv:1803.00387*. 2018.
[37] Du, Xinxin, Marcelo H. Ang, and Daniela Rus. "Car detection for autonomous vehicle: LIDAR and vision fusion approach through deep learning framework." *Proceedings of the IEEE conference on Intelligent Robots and Systems (IROS)*. 2017.
[38] Song, Shuran, and Jianxiong Xiao. "Deep sliding shapes for amodal 3d object detection in rgb-d images." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).* 2016.
[39] Zeng, Yiming, *et al.* "Rt3d: Real-time 3-d vehicle detection in lidar point cloud for autonomous driving." *IEEE Robotics and Automation Letters*. 2018,1-1.
[40] Beltran, Jorge, *et al.* "BirdNet: a 3D Object Detection Framework from LiDAR information." *arXiv preprint arXiv:1805.01195.* 2018.
[41] Yang, Bin, Wenjie Luo, and Raquel Urtasun. "PIXOR: Real-Time 3D object detection from point clouds." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).* 2018.
[42] Redmon, Joseph, and Ali Farhadi. "YOLO9000: better, faster, stronger." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).* 2017,6517-6525.
[43] Minemura, Kazuki, *et al.* "LMNet: Real-time multiclass object detection on CPU using 3D LiDARs." *arXiv preprint arXiv:1805.04902.* 2018.
[44] Ren, Mengye, *et al.* "SBNet: Sparse blocks network for fast inference." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).* 2018.
[45] "nuTonomy Driverless Taxi", *https://www.nutonomy.com/*, accessed: 2018-12-20.
[46] "Scale Data Annotations", *Data Annotations*, accessed: 2018-12-24.

Table 2: 3D detection performance: Average precision (AP_{3D}) (in $\%$) of 3D detection on KITTI official website.

	Car			Pedestrian			Cyclist		
Method	Easy	Moderate	Hard	Easy	Moderate	Hard	Easy	Moderate	Hard
AVOD(FPN)	81.94	71.88	66.38	50.80	42.81	40.88	64.00	52.18	46.61
MV3D	71.09	62.35	55.12	-	-	-	-	-	-
F-PointNet	81.20	70.39	62.19	51.21	44.89	40.23	71.96	56.77	50.39
PointFusion	77.92	63.00	53.27	33.36	28.04	23.38	49.34	29.42	26.98
Complex-YOLO	55.63	49.44	44.13	19.45	15.32	14.80	28.36	23.48	22.85
VoxelNet	67.27	52.87	46.62	-	-	-	-	-	-
RT3D	23.49	21.27	19.81	-	-	-	-	-	-
BirdNet	14.75	13.44	12.04	14.31	11.80	10.55	18.35	12.43	11.88
RoarNet	83.71	73.04	59.16	-	-	-	-	-	-

Table 3: Retrained results on KITTI and nuScenes

	KITTI				nuScenes	
Method	FPS	Easy	Moderate	Hard	FPS	General
AVOD(FPN)	9.3	79.76	73.43	69.52	5.7	69.55
MV3D	2.8	67.97	70.23	56.82	1.2	44.40
F-PointNet	4.7	73.53	62.18	53.70	3.3	59.24