International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 08 Issue: 05 | May 2021 www.irjet.net p-ISSN: 2395-0072

A Brief Comparison of 3D Object Detection with VoxelNet and Frustum PointNet

Sreeprad D[1], Hariharan N[2], Dhanush S[3]

[1-3]*D410 S and P Living Spaces, Kamaraj Street*

Abstract - *Computer Vision serves as the basis for Artificial Intelligence to help the computer contemplate the visual world. With the emergence of Convolutional Neural Networks, this concept has been utilized for extracting the features of an image and understanding its significance in constructing the image. This led to the birth of Object Detection for computers to find and locate objects in the given image, serving various purposes in the automation world. For Autonomous Vehicles, this is one the most important concepts required to detect obstacles and to be trained to avoid them. In this paper, we explore the two distinct methods of Faster-RCNN to find and locate other vehicles that pose as an obstacle with the data obtained from various sensors placed atop our vehicle. We implement the two top-end models for 3D object detection (i.e. Voxelnet and Pointnet) and compare the strengths and weaknesses of the two models. We use TensorFlow and Keras to train our model to plot and identify the location of the obstacles in real time. We use the Mayavi library to build an environment to visualize the result and plot the bounding boxes of 3D point clouds.*

Key Words: **3D object detection, Pointnet, Voxelnet, Autonomous vehicle.**

1. INTRODUCTION

With the advent of object detection, point cloud 3D object detection has become forerunner and has gained momentum as a research topic in the 3D computer vision community. Understanding a wide range of applications of 3D recognition in robotics, Augmented Reality etc has become a fundamental task of Industrial Revolution 4.0. One such field pivots around using techniques closely related to the Convolutional neural network (CNN), being extensively researched.

However, use of LiDAR in detection of 3D objects remains a challenge. Moreover, deploying autonomous vehicles in urban environments poses technological hence administrative and legislative challenges; as also, it engulfs other tasks of detecting moving objects, generating, and representing sparse point clouds, distance measurement & accuracy etc. Regardless of its complications, LiDAR often outperforms 2D object detection due to the fundamental difference in its modalities. This being a) sparse representation over dense images (as noticed in ROI mapping) and b) using point clouds which are 3D while image-based object detection is 2D.

In this paper we dive into two architectures of 3D object detection - voxelnet and frustum pointnet - that augment LiDAR, learning directly from point clouds. Hence, by making use of comparative analysis and feature-extraction methodology, and taking into account challenges of quantization, localization, effective range of detection, sampling uniformity/non-uniformity etc, we evaluate their respective performance.

Some early works have tremendously experimented on using 3D object detection or by projection of point cloud onto images [], thus offering some perspectives and insights, being advantageous. However, using this generalized view neglects the applicability and efficiency of convolutional neural networks - ending up impractical in application. In order to address these issues, we have taken two well performing models and compare them in terms of accuracy, results, localization, ease of object classification etc.

1.1 Voxelnet

What Voxelnet typically does is equally divide and equally space out '3D voxels'. So it then morphs these collection of points within each voxel into a simplified object manifestation, i.e. shape representation, by using a convolutional middle layer such as ConvMD (cin, cout, k, s, p) which will be explained later on. An Advantage here is the convolutional middle layers will aggregate voxel-wise characteristics within an expanding receptive field, adding more context to the shape description. To simplify, it's a progressive, expanding methodological apparatus invoked for vividly defining shape boundaries. This is what gives its bounding boxes high accuracy. But as foreseen, the model requires N number of steps which could be memory consuming and time taking procedure.

For even better comparison, Voxelnet has reportedly been said to "outperform the State of the Art, LIDAR based 3D object detection methods by a large margin", according to Y Zhou et al (2017).

1.2 Pointnet

Pointnet is a deep learning framework which directly consumes the 3D point cloud into its learning framework. It then applies these point clouds into a unified task of 3 approaches involving semantic or part segmentation of parts and object classification. By taking in raw point cloud data, Pointnet is a pioneer neural network model in machine

learning in the sense of it being simple and effective for point cloud recognition. It does so with these sets of points without rendering or voxelization so that it can reduce unnecessary colossal tasks. Thereby, Pointnet too is highly effective and in performing against State of the Art 3D object detection methods.

What Pointnet uses is called max pooling, it is a single and symmetric function of dealing with raw input data that is expected to be an unordered set.

2. RELATED WORK

As we discussed earlier, tremendous research has been made on 3D understanding, its implementation using LiDAR. Some detailed and well defined, earlier approaches [11, 13, 14] show us exciting hand-crafted representations. Newer popular representations using 3D convolutions to compute 3D voxel grid [10, 12]. But, most such efforts have focused on hand-crafted feature representations such as bird's eye view projection until VoxelNet. The work of Yin Zhou; et al., [7] introduces a generic 3D detection network that unifies feature extraction and bounding box prediction into a single stage, thus providing a single end-to-end trainable neural network. VoxelNet takes the input of point cloud as equally spaced 3D voxels and transforms a group of points within each voxel into a unified feature representation. This is achieved through the voxel feature encoding (VFE) layer. This layer is then connected to a RPN layer to generate detections. Although this seems to be a single and clean solution to process LiDAR data, VoxelNet is slow and expensive when dealing with such a large amount of voxels.

The work of Qi; et al., [8] introduces another approach for 3D object detection using Frustum PointNet. This work uses RGBD data to perform 3D object detection instead of using 3D voxels. It directly operates on raw point clouds by popping up RGB-D scans. Instead of solely relying on 3D proposals, Frustum PointNet leverages both mature 2D object detectors and advanced 3D deep learning for object localization, achieving efficiency as well as high recall for even small objects.

In this paper, we compare the concept and workings of VoxelNet and Frustum PointNet in effectively detecting objects in the environment and determine the performance of each in different scenarios.

3. IMPLEMENTATION

3.1 Voxelnet:

VoxelNet is a 3D object detection model that solves the problems faced by traditional models by using revolutionary new methods (learns the features of the objects directly from the point cloud instead of detecting the manually crafted features used by traditional object detection models that bottleneck the usable information and increase the computational complexity and memory density). It uses LiDAR sensors to obtain the input data rather than using 2D image and depth data. Though this is an expensive method, the use of point cloud data helps make it a complete end-to-end model for 3D object detection. VoxelNet achieves this using 3 functional blocks, similar to FRCNN, (1) Feature learning network (2) Convolutional middle layers (3) Region proposal network.

Feature learning network:(Pre-processing)

Voxel partitioning and grouping of point cloud:

The 3D space information from the point cloud is divided into equal partitions called voxels based on the dimensions of the 3D space. The point clouds are then mapped into their respective voxels based on their position in space.

Random sampling:

Unlike ocular data, the data from point clouds is sparse and has a very highly varying density due to the factors such as non-uniform sampling, occlusion and the pose of the object relative to the sensor. This can cause issues such as increased memory and computational requirements and biased data due to the highly varying density of points. This is solved by fixing a random limiter for the number of points that can reside in a voxel. This solves the above mentioned problems and greatly decreases the input bias caused by the erratic point density.

Stacked voxel feature encoding:

The stacked voxel feature encoding layers accomplish the task of encoding the features of the points into the voxels they are located in. The VFE layers take in the xyz coordinates of the points in the voxel as input and passes it through linear layers with batch norm and ReLU to obtain the pointwise feature representation. It is then aggregated by concatenating the outputs to their corresponding voxel-wise feature.

Sparse tensor representation: It is obtained by processing only the non-empty voxels from which a list of voxel features are acquired. The voxel-wise features can be represented as a sparse tensor of 4 dimensions. As over 90% of the voxels are empty from the point cloud data, representing the non-empty voxels greatly decreases the memory usage.

Convolutional middle layers:

The model uses ConvMD(cin, cout, k, s, p) to represent an Mdimensional convolution operator where cin and cout are the number of input and output channels, k, s, and p are the M-dimensional vectors corresponding to kernel size, stride size and padding size respectively. When the size across the M-dimensions are the same, a scalar is used to represent the size e.g. k for k = (k, k, k). Each convolutional middle layer applies 3D convolution, BN layer, and ReLU layer

sequentially. The convolutional middle layers aggregate voxel-wise features within a progressively expanding receptive field, adding more context to the shape description.

Region proposal network:

A region proposal network following the architecture derived from the faster r-cnn is heavily modified and incorporated into this model which takes the input feature map from the last convolutional middle layer. The RPN has 3 blocks of fully convolutional layers. The first layer of each block down-samples the feature map by half via a convolution with a stride size of 2, followed by a sequence of convolutions of stride 1 *q. After each convolution layer, BN and ReLU operations are applied. The output of every block is then upsampled to a fixed size and concatenated to construct the high-resolution feature map. The feature map is mapped to the probability score map and regression map.

3.2 Pointnet:

Pointnet also uses a deep learning framework that takes point cloud data as input. But, unlike voxelnet pointnet takes a different approach in rather than grouping the point cloud data into voxels the model learns to select and take the most valuable data which in turn later is used to determine the shape of the overall structure. Pointnet achieves this by using depth data along with 2D images from the Kinect sensor. This is a cheaper alternative compared to a LiDAR sensor.

Pointnet has two functional modules, the classification network and the segmentation network. And the model comprises three modules, a symmetry function to process the unordered input. Moreover architecturally the model is classified into classification network and segmentation network.

Classification network:

The classification network attempts to find the global maxima to fit the classes for the data. Classification network uses a max-pooling layer to aggregate the global features and scores the fit of each of the classes.

Segmentation network:

The segmentation network takes in the features and scores from the classification network and tries to find the local maxima for the network. It tries to fit the classes to specific regions.

4. RESULT:

In this paper, we have reviewed two methods to localize and detect objects in the environment and provide 3D bounding box outputs. We evaluate and compare the performances of both VoxelNet from the work of Yin Zhou; et al., [7] and Frustum Pointnet from the work of Qi; et al., [8] on the KITTI 3D object detection benchmark which contains 7,481 training images and point clouds and 7,518 test images and point clouds, covering three different categories: Car, Pedestrian, and Cyclist. For each of these classes, the detection outcomes are evaluated on the basis of three difficulty levels: easy, moderate, and hard, determined according to the object size, occlusion state, and truncation level in the image captured by the camera sensor. Table 1 shows the test results from the KITTI server [7] for average precision in bird's eye view detection; Table 2 shows the test results from the KITTI server for average precision in 3D detection [9]. For analysis, the two methods are compared with several other algorithms such as VeloFCN [2], 3D-FCN [3], MV [4], Mono3D [5], and 3DOP [6].

Table -1:

Preparation of Manuscript										
		Car			Pedestrian			Cyclist		
Method	Modality	Easy	Moderate	Hard	Easy	Moderate	Hard	Easy	Moderate	Hard
Mono3D	Mono	5.22	5.19	4.13	N/A	N/A	N/A	N/A	N/A	N/A
3DOP	Stereo	12.63	9.49	7.59	N/A	N/A	N/A	N/A	N/A	N/A
VeloFCN	LiDAR	40.14	32.08	30.47	N/A	N/A	N/A	N/A	N/A	N/A
MV(BV+FV)	LiDAR	86.18	77.32	76.33	N/A	N/A	N/A	N/A	N/A	N/A
MV(BV+FV+RGB)	LiDAR+Mono	86.55	78.1	76.67	N/A	N/A	N/A	N/A	N/A	N/A

VoxelNet	LiDAR	89.6	84.81	78.57	65.95	61.05	56.98	74.41	52.18	50.49
PointNet	LiDAR+ Mono	88.7	84	75.33	58.09	50.22	47.2	75.38	61.96	54.68

Table -2:

Preparation of Manuscript										
Method	Modality	Car			Pedestrian			Cyclist		
		Easy	Moderate	Hard	Easy	Moderate	Hard	Easy	Moderate	Hard
Mono3D	Mono	2.53	2.31	2.31	N/A	N/A	N/A	N/A	N/A	N/A
3DOP	Stereo	6.55	5.07	4.1	N/A	N/A	N/A	N/A	N/A	N/A
VeloFCN	LiDAR	15.2	13.66	15.98	N/A	N/A	N/A	N/A	N/A	N/A
MV(BV+FV)	LiDAR	71.19	56.6	55.3	N/A	N/A	N/A	N/A	N/A	N/A
MV(BV+FV+RGB)	LiDAR+ Mono	71.29	62.68	56.56	N/A	N/A	N/A	N/A	N/A	N/A
VoxelNet	LiDAR	81.97	65.46	62.85	57.86	53.42	48.87	67.17	47.65	45.11
PointNet	LiDAR+ Mono	81.2	70.39	62.19	51.21	44.89	40.23	71.96	56.77	50.39

5. CONCLUSIONS

This paper reviewed the state-of-the-art of 3D object detection within the context of autonomous vehicles by analyzing the working of VoxelNet and Frustum Pointnet. We analyzed sensors technologies and discussed standard datasets for 3D object Detection for vehicles. VoxelNet and PointNet were reviewed alongside similar works that were categorized based on sensor modality: monocular images, point clouds (obtained through lidars or depth cameras) and fusion of both.

Quantitative results, obtained from the KITTI benchmark, showed that monocular methods are not reliable for 3D object detection, due to lack of depth information, which prevents accurate 3D positioning [1]. Performance from LiDAR methods, such as VeloFC and MV, were surpassed by that of VoxelNet in 3D object detection and accurate localization in both bird's eye view and 3D detection. While the performance of PointNet is also better when compared to VeloFC and MV, VoxelNet delivers a better performance when it comes to 3D object detection. Though PointNet is faster in localizing the objects due to its FRCNN algorithm on 2D image, VoxelNet's higher score shows it's better in overall performance.

REFERENCES

[1] Arnold, E., Al-Jarrah, O.Y., Dianati, M., Fallah, S., Oxtoby, D. and Mouzakitis, A., 2019. A survey on 3d object detection methods for autonomous driving applications. IEEE Transactions on Intelligent Transportation Systems, 20(10), pp.3782-3795.

[2] Li, B., Zhang, T. and Xia, T., 2016. Vehicle detection from 3d lidar using fully convolutional network. arXiv preprint arXiv:1608.07916.

[3] Li, B., 2017, September. 3d fully convolutional network for vehicle detection in point cloud. In 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (pp. 1513-1518). IEEE.

[4] Chen, X., Ma, H., Wan, J., Li, B. and Xia, T., 2017. Multi-view 3d object detection network for autonomous driving. In Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (pp. 1907-1915).

[5] Chen, X., Kundu, K., Zhang, Z., Ma, H., Fidler, S. and Urtasun, R., 2016. Monocular 3d object detection for autonomous driving. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 2147-2156).

[6] Chen, X., Kundu, K., Zhu, Y., Berneshawi, A.G., Ma, H., Fidler, S. and Urtasun, R., 2015. 3d object proposals for accurate object class detection. In Advances in Neural Information Processing Systems (pp. 424-432).

[7] Zhou, Y. and Tuzel, O., 2018. Voxelnet: End-to-end learning for point cloud based 3d object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 4490-4499).

[8] Qi, C.R., Liu, W., Wu, C., Su, H. and Guibas, L.J., 2018. Frustum pointnets for 3d object detection from rgb-d data. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 918-927).

[9] Gujjar, H.S., 2018. A Comparative Study of VoxelNet and PointNet for 3D Object Detection in Car by Using KITTI Benchmark. International Journal of Information Communication Technologies and Human Development (IJICTHD), 10(3), pp.28-38.

[10] Engelcke, M., Rao, D., Wang, D.Z., Tong, C.H. and Posner, I., 2017, May. Vote3deep: Fast object detection in 3d point clouds using efficient convolutional neural networks. In 2017 IEEE International Conference on Robotics and Automation (ICRA) (pp. 1355-1361). IEEE.

[11] Maturana, D. and Scherer, S., 2015, May. 3d convolutional neural networks for landing zone detection from lidar. In 2015 IEEE international conference on robotics and automation (ICRA) (pp. 3471-3478). IEEE.

[12] Wang, D.Z. and Posner, I., 2015, July. Voting for voting in online point cloud object detection. In Robotics: Science and Systems (Vol. 1, No. 3, pp. 10-15607).

[13] Tuzel, O., Liu, M.Y., Taguchi, Y. and Raghunathan, A., 2014, September. Learning to rank 3d features. In European Conference on Computer Vision (pp. 520-535). Springer, Cham.

[14] Maturana, D. and Scherer, S., 2015, September. Voxnet: A 3d convolutional neural network for real-time object recognition. In 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (pp. 922-928). IEEE.R. Nicole, "Title of paper with only first word capitalized," J. Name Stand. Abbrev., in press.

[15] K. Elissa, "Title of paper if known," unpublished.