3D Object Detection using KIITI Dataset

Abhishek Jain, Nagarjun Vinukonda, Rishabh Chadha, Vrushabh Desai

***Abstract*—3D Object detection is an active research problem for Perception of Autonomous Vehicles. The goal of our project is to understand the Frustum PointNet architecture and experiment with possible design modifications and evaluate their influence on performance metrics. There are various components in the 3D Object Detection Pipeline and we have experimented with a few modifications in the architecture and loss functions which are discussed in the report.**

I. Introduction

Object detection in 2D is well known and there has been a lot of progress in that domain . However, the ideas from 2D object detection are not directly transferable to 3D object Detection on Point Clouds. This is because of the change in the structure of the data. A reliable method for 3D Object Detection is important for autonomous vehicles to obtain a better understanding of the environment around it and make corresponding decisions. Compared to 2D object detection where the model is trained to draw bounding boxes around the object of interest in the image plane, 3D object detection requires estimation of the object size and related details in the 3D world. For 3D object detection, we need to draw a 3D bounding box in 3D space (rectangular cuboid).

Fig. 1: Sample image from the KITTI Dataset

II. Dataset

The dataset [2] has been captured from a VW station wagon for use in mobile robotics and autonomous driving research. In total, the data of 6 hours of traffic scenarios has been recorded at 10-100 Hz using a variety of sensor modalities such as high-resolution color and grayscale stereo cameras, a Velodyne 3D laser scanner and a high-precision GPS/IMU inertial navigation system. The scenarios are diverse, capturing real-world traffic situations and range from freeways over rural areas to inner-city scenes with many static and dynamic objects. The data contains 7481 training point-cloud data and images and 7518 testing point cloud data and images. The data falls into 2 types of classifications: 1) Based on the difficulty which depends on the amount of occlusion. 2) Based on the object type - Car, Pedestrian and Cyclist.

III. Related Work

There have been different approaches to tackle the problem of applying Deep Learning on Point Clouds. Few of them involve representing point clouds as a collection of 2D images over which 2D convolution is applied. Other approaches have attempted to discretize the point cloud into a volumetric 3D grid and then apply 3D convolutions over that volume to classify if it contains an object of interest. PointNet[3] was a pioneering architecture which directly worked on point clouds.

Frustum PointNets[1] take advantage of existing 2D detection algorithms to help in localizing an object of interest in 3D point clouds. The technique takes a LiDAR point cloud and a 2D bounding box containing an object as input, and outputs an estimated 3D bounding box for the object. This is done by an efficient amalgam of both 2D object detection and 3D object detection along with a reduced computational burden leading to efficiency in terms of both speed and memory.

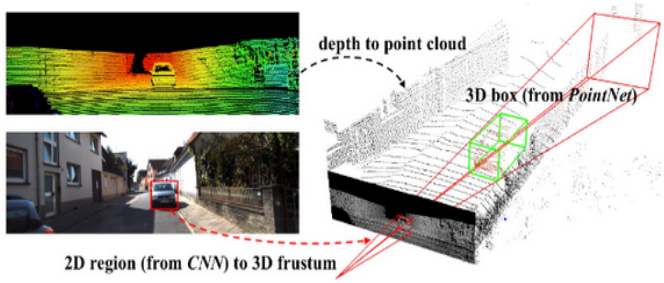

Fig. 2: 3D Frustum Region Proposal from 2D projection

This is achieved in the following manner: 2D object detection techniques are used to generate region proposals in 2D space using RGB images. 3D Frustums are then extruded in the point cloud space using these 2D space proposals thus generating 3D Frustum point cloud proposals as shown in Fig. 2. Then, 3D object detections techniques (Point-Net based) are used in a 2-way fashion to generate 3D bounding boxes in 3D space. The 3D object detection is performed in 2 steps:

1) Classification
2) Regression

Classification is used to segment image into the three identified classes: Cars, Pedestrians and Cyclists. In a multi-class detection case, the technique also leverages the semantics from a 2D detector for better instance segmentation. For example, if we know the object of interest is a pedestrian, then the segmentation network can use this prior to find geometries

that look like a person. Correspondingly, the detected object is classified into the classes as per merit and a 3D bounding box is generated which leads to the second part of the detection :- Regression.

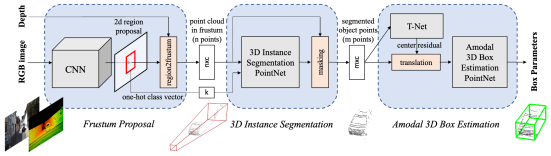

Fig. 3: Frustum Point-Net Architecture overview

Regression is used to get the best estimate of the placement of the 3D bounding box around the detected object. The technique uses a light-weight regression PointNet (T-Net) to estimate the true center of the complete object and then transform the coordinate such that the predicted center becomes the origin of the bound box. Hence, the problem is firmly governed by two major techniques: Classification and Regression. Correspondingly, the training loss involves both classification and regression losses as major contributors. There is potential to change the 2D detection method in the architecture, recent 2D detectors have been specifically designed for application on embedded systems.

IV. Proposed Method

SqueezeDet [4] is an architecture that was designed particularly for carrying out 2D object detection which can be deployed on embedded systems in Autonomous vehicles. Our motivation to use SqueezeDet was due to its significant low memory model size and high speed.

Method	Car mAP	Cyclist mAP	Pedestrian mAP	All mAP	Model size (MB)	Speed (FPS)
FRCN + VGG16[2]	86.0	-	-	-	485	1.7
FRCN + AlexNet[2]	82.6	-	-	-	240	2.9
SqueezeDet (ours)	82.9	76.8	70.4	76.7	7.9	57.2
SqueezeDet+ (ours)	85.5	**82.0**	**73.7**	**80.4**	26.8	32.1
VGG16-Det (ours)	**86.9**	79.6	70.7	79.1	57.4	16.6
ResNet50-Det (ours)	86.7	80.0	61.5	76.1	35.1	22.5

Fig. 4: Summary of detection accuracy, model size and inference speed

Also, most architectures which deal with 3D point clouds use Smooth L1 loss for regression, we investigated the use of L2 loss over this as well.

V. Experiments

For the Frustum PointNet architecture, 2D region proposals are extruded in the form of a frustum in the point cloud which helps to localize the objects of interest. We replaced the original Fast-RCNN 2D proposals by proposals from SqueezeDet. This would lead to different 3D frustum point cloud regions of interest and hence would change the overall results of the model as well

Fig. 5: Comparison of the size Model size and speed of SqueezeDet and Fast RCN.

A. Results

We analyzed the performance of the network using 2 kinds of losses:

1) Smooth L1 loss
2) MSE Loss

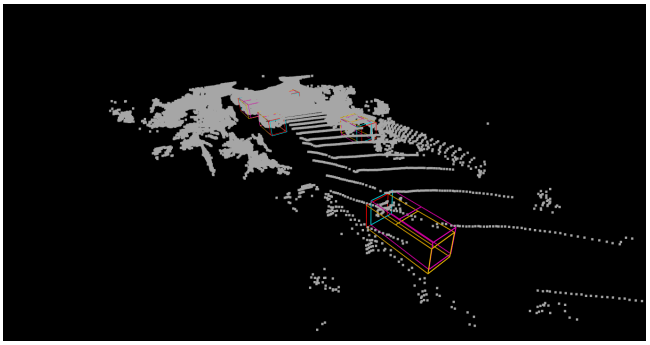

Fig. 6: Object detection using Smooth L1 Loss

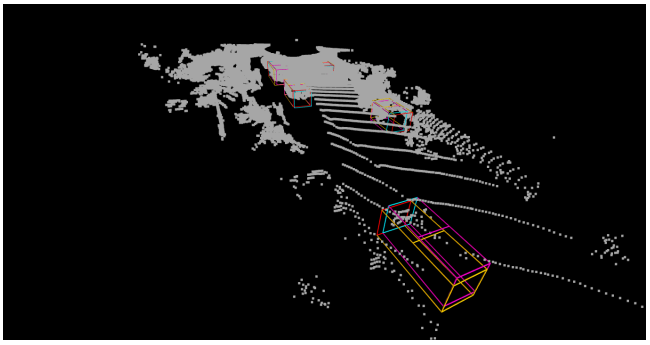

Fig. 7: Object detection using MSE Loss

The above figures show the 3D object detections with the two kinds of losses discussed in our approach. The pink color denotes the actual ground truth bounding box with blue color denoting the actual heading directions whereas the yellow and red color denote the predicted bounding box and heading direction respectively.

We used Adam Optimiser with a learning rate of 10-3 with a batch size of 32 to get the following graphs.

The MSE loss function initially gave a huge loss value before converging down to an optimum value both in the training and validation losses.

VI. Discussion

During training, if the MSE loss is used the initial values are very high as evidenced by the graph the model takes longer to train. However, as the training continues the loss starts to decrease consistently. For the metrics in 2D object detection, the values were close to the values for the original

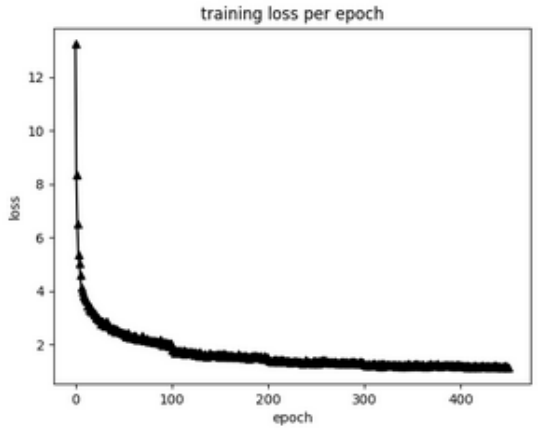

Fig. 8: Training Loss for Smooth L1

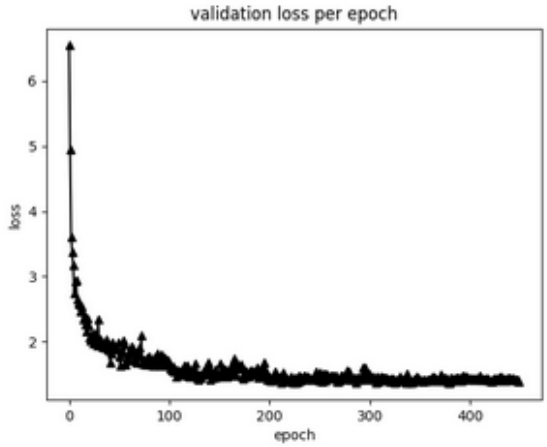

Fig. 9: Validation Loss for Smooth L1

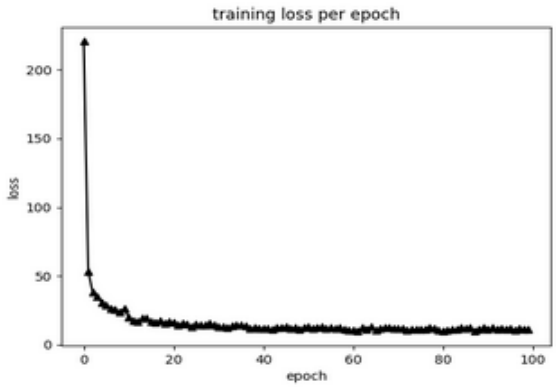

Fig. 10: Training Loss for MSE

Frustum PointNet architecture. However, the values for AP in 3D object detection were significantly lesser than the results of the original Frustum PointNet model.

One probable reason for this difference in the 3D proposals generated by the 2D detector which was changed. The authors in the original paper performed significant fine tuning and used

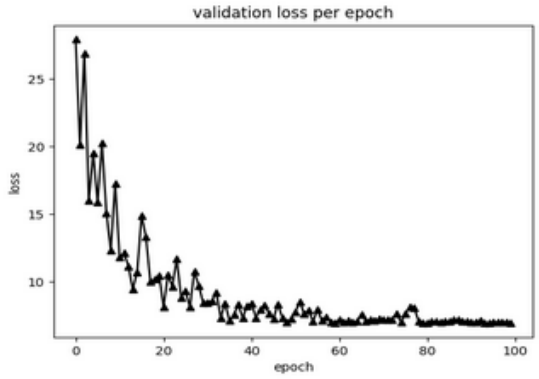

Fig. 11: Validation Loss for MSE

Car detection 2D	Easy	Moderate	Hard
Frustum PointNet Original	0.87	0.819	0.73
Modified Frustum PointeNet	0.88	0.814	0.83

Fig. 12: 2D Detection Results (mAP)

Car detection 3d	Easy	Moderate	Hard
Frustum Point Original	0.82	0.68	0.61
Modified Frustum PointNet	0.58	0.55	0.52

Fig. 13: 3D Detection Results (mAP)

an initial model trained on ImageNet and fine tuned it on the COCO dataset. The accuracy of 3D detections depends heavily on the initial regions proposed by the 2D detector. One major roadblock for making progress was the training time it took for the model in case any changes were made. It was in the order of 2 days. The SqueezeDet 2D detection model can be further fine tuned on other data sets before it is used on the KITTI Dataset (ex. COCO dataset).

VII. Conclusion and Future Directions

For this project we attempted to understand the overall pipeline for 3D object detection in Autonomous Vehicles and carry out some modifications to understand their influence on the overall performance. The performance of the model could be improved by fine tuning of the SqueezeDet detector as discussed above. Current State Of The Art methods for 2D object detection and alternatives to the PointNet architecture can also be incorporated into this pipeline to experiment with and potentially improve the performance of the model. The solutions to real world problems in 3D detection can get very complicated and understanding each sub-component of the model and their influence on the overall performance can take a significant amount of time to comprehend.

REFERENCES

[1] C. R. Qi, W. Liu, C. Wu, H. Su and L. J. Guibas, "Frustum PointNets for 3D Object Detection from RGB-D Data," 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, 2018, pp. 918-927, doi: 10.1109/CVPR.2018.00102.

[2] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for au-tonomous driving? the kitti vision benchmark suite.InCVPR, 2012.

[3] C. R. Qi, L. Yi, H. Su, and L. J. Guibas. Pointnet++: Deephierarchical feature learning on point sets in a metric space.arXiv preprint arXiv:1706.02413, 2017.

[4] B. Wu, A. Wan, F. Iandola, P. H. Jin and K. Keutzer, "SqueezeDet: Unified, Small, Low Power Fully Convolutional Neural Networks for Real-Time Object Detection for Autonomous Driving," 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Honolulu, HI, 2017, pp. 446-454, doi: 10.1109/CVPRW.2017.60.

[5] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meetsrobotics: The kitti dataset. The International Journal ofRobotics Research, 32(11):1231–1237, 2013. 5