

CSC420 Project Presentation

Road detection and Object detection

(Sample Project 4)

Zilun Zhang
Jiahuang Lin
Yaolong Wang

Object Detection


Dataset :

- Divided KITTI (left) into training set and testing set with ratio 4 : 1
- With label



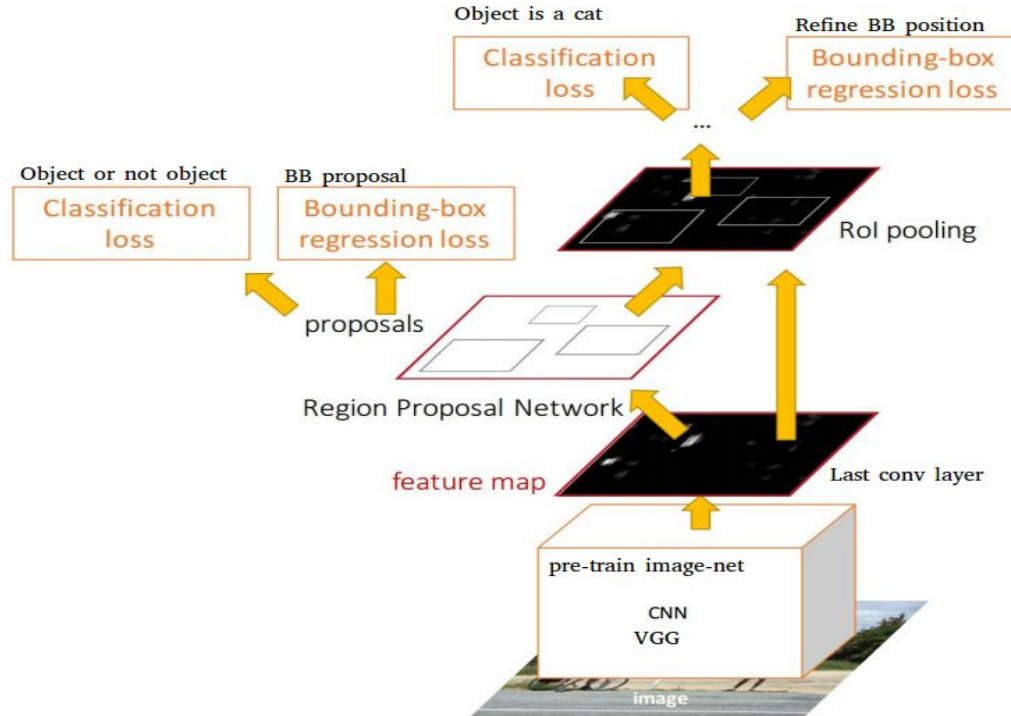
Object Detection

Main architecture introduction:

- Faster RCNN tensorflow implmented version with modified
 - Credit to [CharlesShang](#)
 - 4 classes (Car, Cyclist, Pedestrain and Don't Care)
 - Use a pre-trained VGG16 on ImageNet
 - 200000 iterations
 - 8 Hours training with 1080 TI GPU
- 

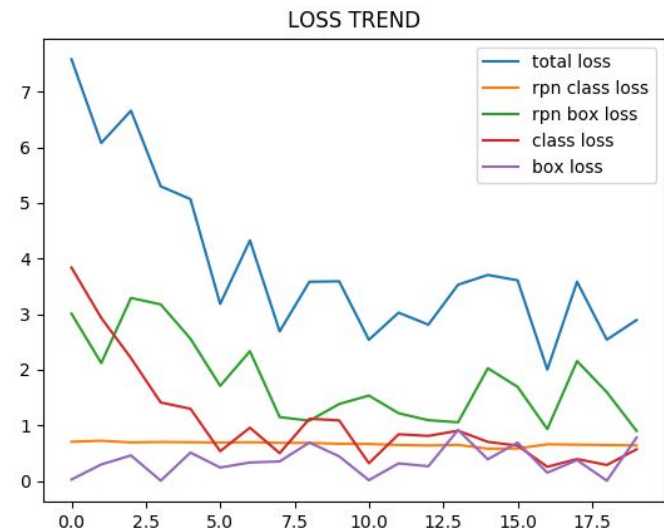
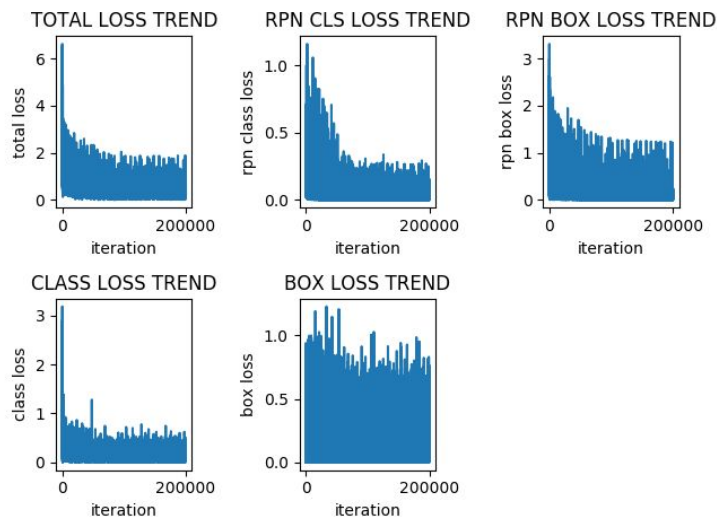
Object Detection

Faster-RCNN:



Object Detection

Training summary (loss):



Object Detection

Sample result present:

Cyclist detections with $p(\text{Cyclist} \mid \text{box}) \geq 0.4$



Object Detection

Sample result present:

Car detections with $p(\text{Car} \mid \text{box}) \geq 0.4$



Object Detection

Sample result present:

Cyclist detections with $p(\text{Cyclist} \mid \text{box}) \geq 0.4$



Object Detection

Main problems for now:

- For single picture with lots of Pedestrian and Cyclist, easy to be confused.
- For picture with overlapped Pedestrian and Cyclist, easy to be confused.

Potential improvement for further investigation:

- increase weight for penalizing wrong classification



Object Detection

Sample result present:

Cyclist detections with $p(\text{Cyclist} \mid \text{box}) \geq 0.4$



Car Orientation Detection

Dataset:

- Based on division of previous object detection
- Create 12 folders for 12 classes ($12 * 30$ degrees)
- Crop car batch from each image based on its label and put them into their corresponding category.
- 23,097 training car batches and 5647 testing car batches.



Car Orientation Detection


Data pre-processing:

- Resize images to their median shape ($49 * 83$)
- Calculate HOG descriptor for each image (2592 features per image) as feature vector
- Result is in a list of 12 matrices, each corresponds to a category
- Matrix[i, j] means i^{th} image in this class, j^{th} feature



Car Orientation Detection

Algorithm and result summary:

- For each orientation class, train a SVM classifier (12 SVMs in total)
 - True labelled data is just the corresponding matrix and false labelled data is randomly picked from the rest 11 matrices.
 - Ratio between the above two labelled data number is 1 : 1.1
 - Use all true label's data for each SVM
 - Test using one-hot vector as ground truth (1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
 - Test accuracy is around 84%
- 

Car Orientation Detection

Result summary (Confusion matrix, 12 * 5647 in total):

	0	1
0	52678	9428
1	1855	3791



Car Orientation Detection

Result demo (true label is 88 degrees):



Road Detection

Dataset introduction:

- KITTI road detection dataset
- Containing left and right image file, 580 images for each of them. The data has already been divided into training and testing dataset.
- Training dataset has corresponding images labeled with ground truth for each image.
- Training dataset has already been equally divided into 3 sub-datasets

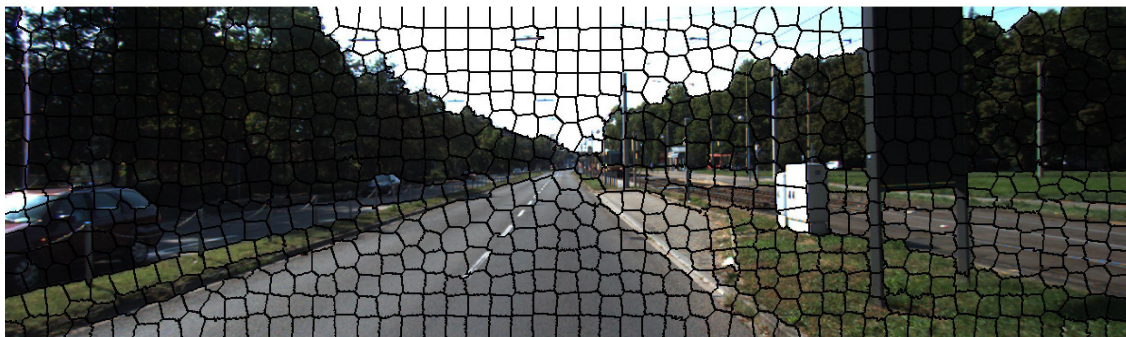


Road Detection

Data pre-processing:

- Apply SLIC algorithm for each image
- Resize each superpixel to 128 x 128
- Extract features (1 x 64) from each superpixel as:

Height | Width | Area | Perimeter | RGB channel histogram



Road Detection

Neural Network main architecture:

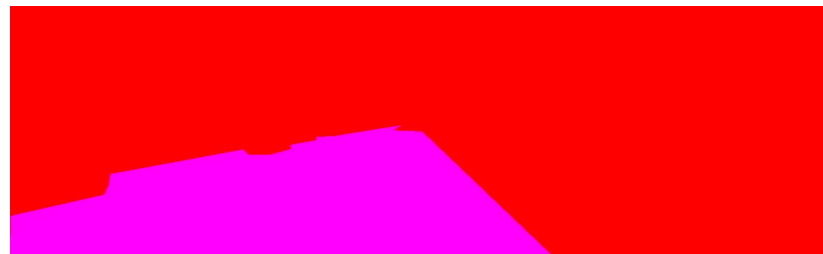
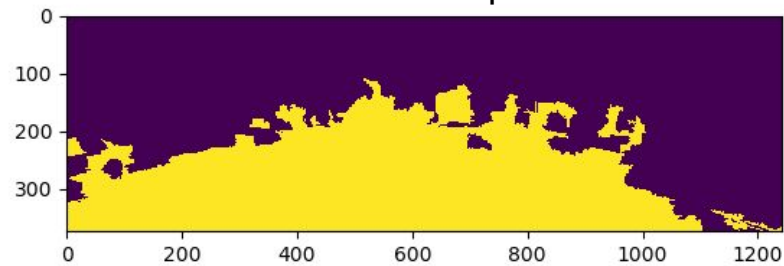
- A two-layer Feedforward-NN, with 40 and 20 neurons in the two hidden layers.
- Use the predivided sub-training dataset (96 images) to train the model
- 200 epochs in total



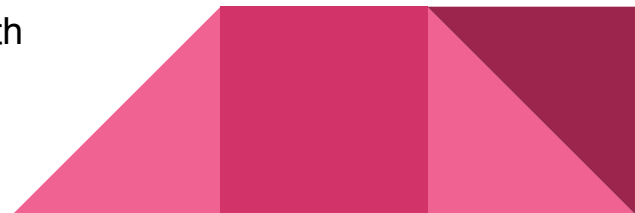
Raw Image



Test Output



Ground Truth



Add 3D feature (height)

We first tried to use height in 3D camera coordinate as an input feature of neural network, but the result turns out very bad (goes down to about 80 percent).

Then we used another method. We use add a threshold T to the pixels that are classified as road, that if its height in 3D coordinate is greater than T , change its label to 'not road'.

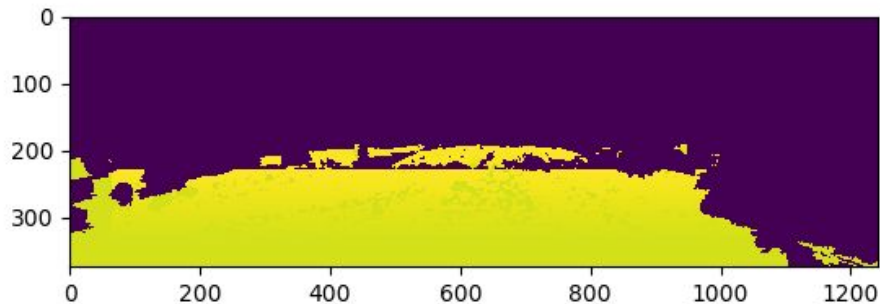
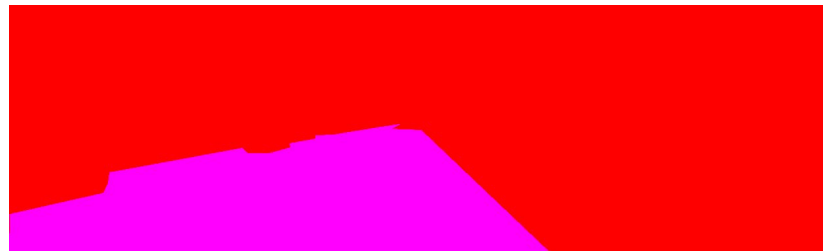


Add 3D feature (height)

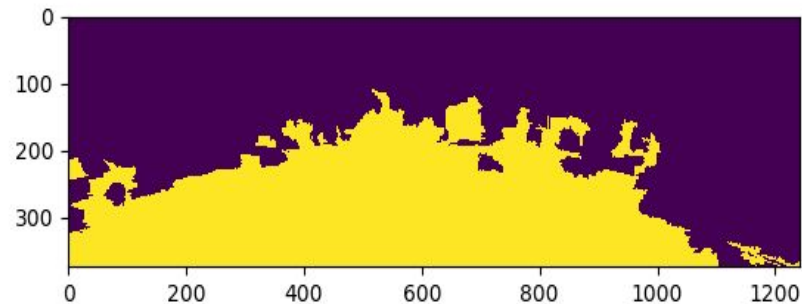
Raw Image



Ground truth

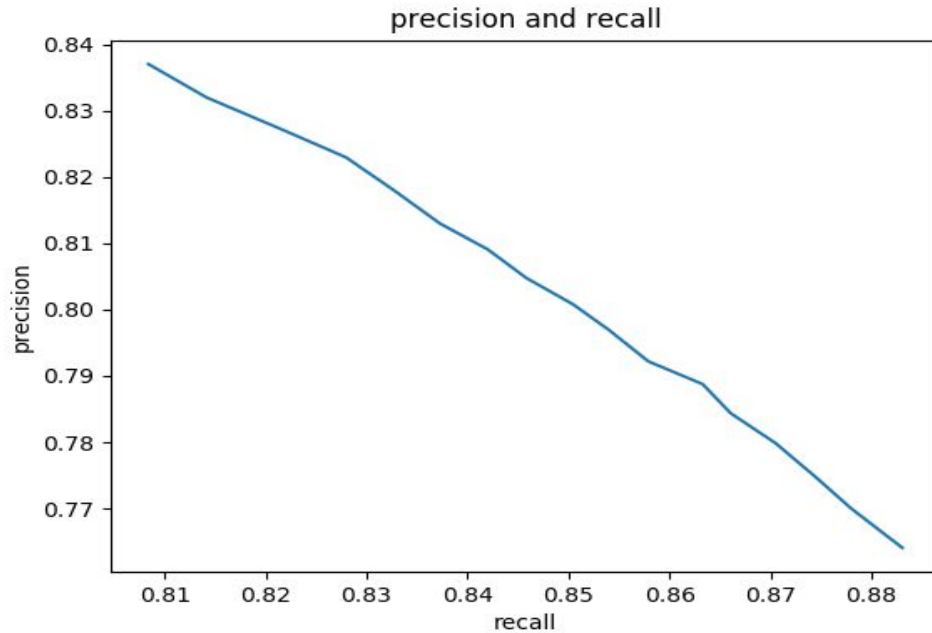


With height



Without height

Precision vs Recall



We applied a threshold T on the output value from the neural network.

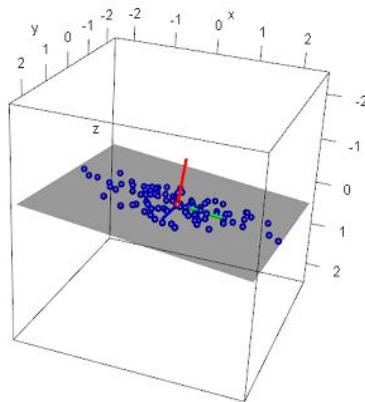
(If the output is greater than T , it's classified as road, otherwise it's not)

We tried on different T .

3D point cloud and plane fitting

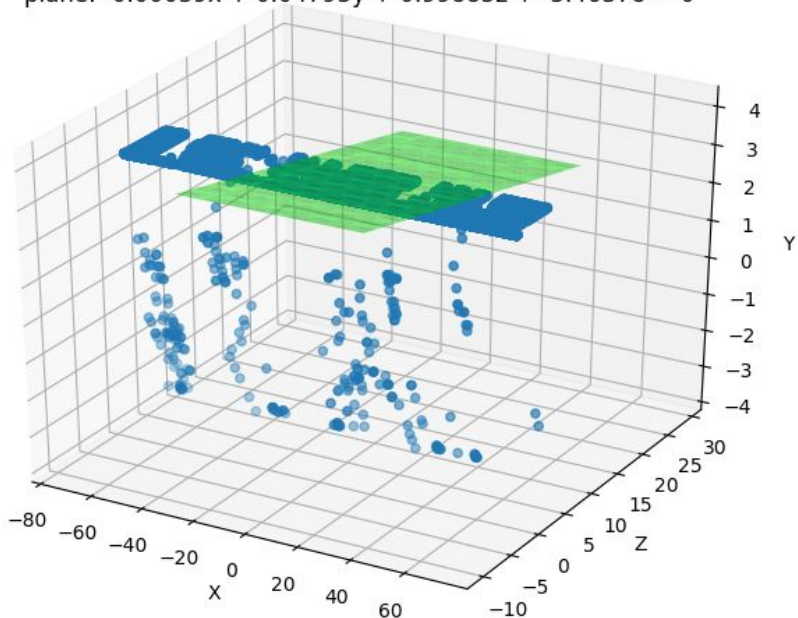
Plane fitting:

- Use PCA plane fitting, 2 largest eigenvectors naturally defines a plane.
- Equivalent to finding a plane that minimizes the sum of the squared distances of points in the point cloud from the plane.
- Really fast



3D point cloud and plane fitting

plane: $-0.00039x + 0.04793y + 0.99885z + -3.46378 = 0$



Problems for now: *noise in 3D point cloud*

Potential improvement:

- weighted PCA
- data pre-processing to clear out noise

Thanks!

