# ROB 535 F21 Perception Final Report


1st Bo-Ray Yeh
*University of Michigan*
Ann Arbor
boray@umich.edu

2nd Cheng-Lung Chiang
*University of Michigan*
Ann Arbor
clchiang@umich.edu

3rd Chien-Lung Chou
*University of Michigan*
Ann Arbor
lungchou@umich.edu

4th Sicong Guo
*University of Michigan*
Ann Arbor
stevengu@umich.edu

5th Hongrui Liu
*University of Michigan*
Ann Arbor
hongruil@umich.edu

6th Guangyan Shen
*University of Michigan*
Ann Arbor
gshen@umich.edu


## I. Introduction

In this project, we are aiming to classify image extract from the GTA5 game engine into classes. The main goal is identify if the designated vehicles existed in the image and classify. To resolve the problem, we train a neural network consisting with numbers of 2D convolution and fully-connected layers in a supervised way and the network. The code is hosted at https://github.com/chien-lung/GTA_classification

## II. Method

Although there are a lot of on-the-shelf networks can be found on Internet, we aim to build our own network from scratch according to our understanding from the lecture. Due to our computational limitation, we decided to built a compact neural network. We referred to the structure of VGG-16 net and constructed the net with three 2D convolutional layer, max pooling layers, and fully-connected layers illustrated in Fig. 1. After that, we inserted 3 fully-connected layers and a dropout layer to make this network more generalized. Finally, since we have three types of vehicle, we set the last fully-connected layer to 3 outputs combined with softmax activation to extract possibilities of the classes.

To generate loss for the multi-category classification task, we adopt the cross-entropy loss to compare the prediction with the one-hot encoded ground truth to back propagate the network properly, as follows,

$$L(x, y) = -\frac{1}{N}\sum_{i=1}^{N} y_i \times \log(NN(x_i)) \tag{1}$$

where $N$ is the size of the training dataset, $x$ is the input image, $y$ is the ground truth category, and $NN$ represents the function of our network.

## III. Experiments

We used Adam optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.999$ to optimize our network. Since we did't have computers with GPU, we trained our network on Google Colab. Finally, we achieved the accuracy of $65.7\%$ with the test dataset.

| Layer (type) | Output Shape | Param # |
|---|---|---|
| conv2d (Conv2D) | (None, 1050, 1912, 64) | 1792 |
| max_pooling2d (MaxPooling2D) | (None, 525, 956, 64) | 0 |
| conv2d_1 (Conv2D) | (None, 523, 954, 32) | 18464 |
| max_pooling2d_1 (MaxPooling2D) | (None, 261, 477, 32) | 0 |
| conv2d_2 (Conv2D) | (None, 259, 475, 32) | 9248 |
| max_pooling2d_2 (MaxPooling2D) | (None, 129, 237, 32) | 0 |
| conv2d_3 (Conv2D) | (None, 127, 235, 16) | 4624 |
| max_pooling2d_3 (MaxPooling2D) | (None, 63, 117, 16) | 0 |
| flatten (Flatten) | (None, 117936) | 0 |
| dense (Dense) | (None, 64) | 7547968 |
| dense_1 (Dense) | (None, 64) | 4160 |
| dense_2 (Dense) | (None, 64) | 4160 |
| dropout (Dropout) | (None, 64) | 0 |
| dense_3 (Dense) | (None, 3) | 195 |

Total params: 7,590,611
Trainable params: 7,590,611
Non-trainable params: 0

Fig. 1. Network architecture.

## IV. Contribution

We splits our team into two group working on the control and perception independently. In this perception final project, Bo-Ray Yeh was responsible for building the model training framework and wrote the report, Cheng-Lung tested the model and wrote the report. Chien-Lung trained our model and wrote the report. Sicong, Hongrui, and Guangyan own the control final project.