# Self-Driving Car Project #3

## Behavioral Cloning

Complete on April 15th, 2018

**Haowei Zhang**

# Part 1 Data Set Acquisition & Exploration

Task 1:
Provide a basic explanation of how to generate data set and show few examples.

Response:
A simulator is provided to generate data set with varying configurations. In this case, we only consider images generated from camera located in the center of the car. Following are the key elements to generate a sufficiently robust data set for training and validation purposes.

(1) Keep the car in the center lane
This provides positive training samples to the network you will be training with. In reality, you always want to keep the car in the center lane and avoid crashing into the roadside.

(2) Recovery driving from the sides
Although positive training samples indicate the car should always stay in the center lane, there would likely be cases where the car falls close to the roadside. And this is where recovery driving is important since it can help guide the car back to the center lane.

(3) Drive smoothly around the curves
When taking left/right turns, it is easily to overshoot the turning angle and crash into the roadside if you are driving too fast. Therefore, it is important to collect some training samples that especially deal with curves in a smooth way.

(4) Data augmentation
Track 1 is left-biased, meaning the car would always need to turn left to stay on track. However, in our cases the car should be able to adapt to new environment and make left/right turns. This requires data augmentation which flips the collected raw images left to the right. This resolves the left-biased problem and makes our training model more robust.

The full data set collected has 12,000 images, in which 8000 images are collected when the car is driving in the center lane. 3000 images are collected as recovery driving from the sides and the rest focuses on driving smoothly around the curves.
Task 2:
Describe image pre-processing techniques applied on the data set.

Response:
The raw images collected from the simulator are 3-channelled RGB images of shape (160, 320, 3) as in Figure 1. However, for the network to have best training result, certain pre-processing techniques are required as follows.

Figure 1: Image from center camera when the car is driving in the center lane

(1) Cropping
The background of captured training image, i.e sky and trees, shall be cropped so as not to affect the training behavior because they are not features to extract.

(2) Conversion to Grayscale
Convert to grayscale single-channel image for the network to train and extract features more easily.

(3) Normalization
Bring images to the same level for training purposes.

(4) Resizing
Resize the cropped image to a shape of (64, 64, 1) so as a square image it can easily fit into the convolutional neural network model.

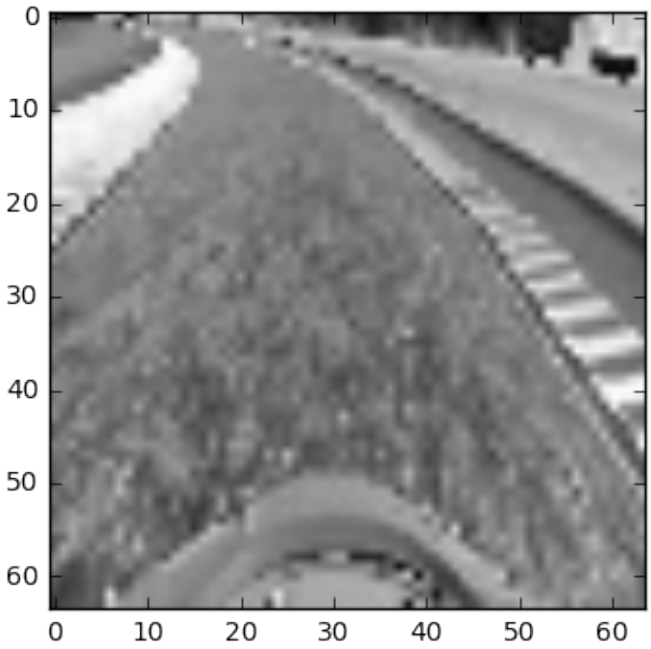


Figure 2: Image from Figure 1 after pre-processing

# Part 2 Model Architecture

Task 1:
Describe the model architecture.

Response:
Following is solution approach my model architecture. Please refer to line 79-94 in model.py.

(1) Layer: Input
Description: 64 × 64 × 1 grayscale images

(2) Layer: Convolutional Layer
Description: takes input of 32 × 32 × 1 grayscale images; uses 3 × 3 × 1 kernel with depth of 32; produces an output of shape 64 × 64 × 32 using RELU activation function

(3) Layer: Max Pooling Layer (including Dropout)
Description: takes input of shape 64 × 64 × 32; max pool with a kernel of shape 2 × 2 × 1; produces an output of shape 32 × 32 × 32. Drop out 25% of connections to avoid overfitting.

(4) Layer: Convolutional Layer
Description: takes input of shape 32 × 32 × 32; uses 3 × 3 × 32 kernel with depth of 32; produces an output of shape 32 × 32 × 32 using RELU activation function.

(5) Layer: Max Pooling Layer (including Dropout)
Description: takes input of shape 32 × 32 × 32; max pool with a kernel of shape 2 × 2 × 32; produces an output of shape 16 × 16 × 32. Drop out 25% of connections to avoid overfitting.

(6) Layer: Fully Connected Layer
Description: takes input of shape 16 × 16 × 32; flatten the input to a 1D array; produces an output of a 1D array of size 100. Drop out 50% of connections to avoid overfitting.

(7) Layer: Fully Connected Layer
Description: takes input of a 1D array of size 100; produces an output of a 1D array of size 1.

Since I only have steering angle as the output, I use the loss function of mean squared error and Adam Optimizer to reduce the error.

Task 2:

Describe how you trained your model. The discussion can include the type of optimizer, the batch size, number of epochs and any hyperparameters such as learning rate.

Response:

To train the model, I use the Adam Optimizer so the learning rate was not tuned manually. The total training is finished within 10 epoches. And for a full data set of 12,000 images, I split it into 10,000 images for training set and 2,000 for validation set. The squared error for training and validation set across epoches is as follows.

Task 3:

Show a clip of your self-driving car on the track.

Response:

Please refer to video.mp4 in my submission files.

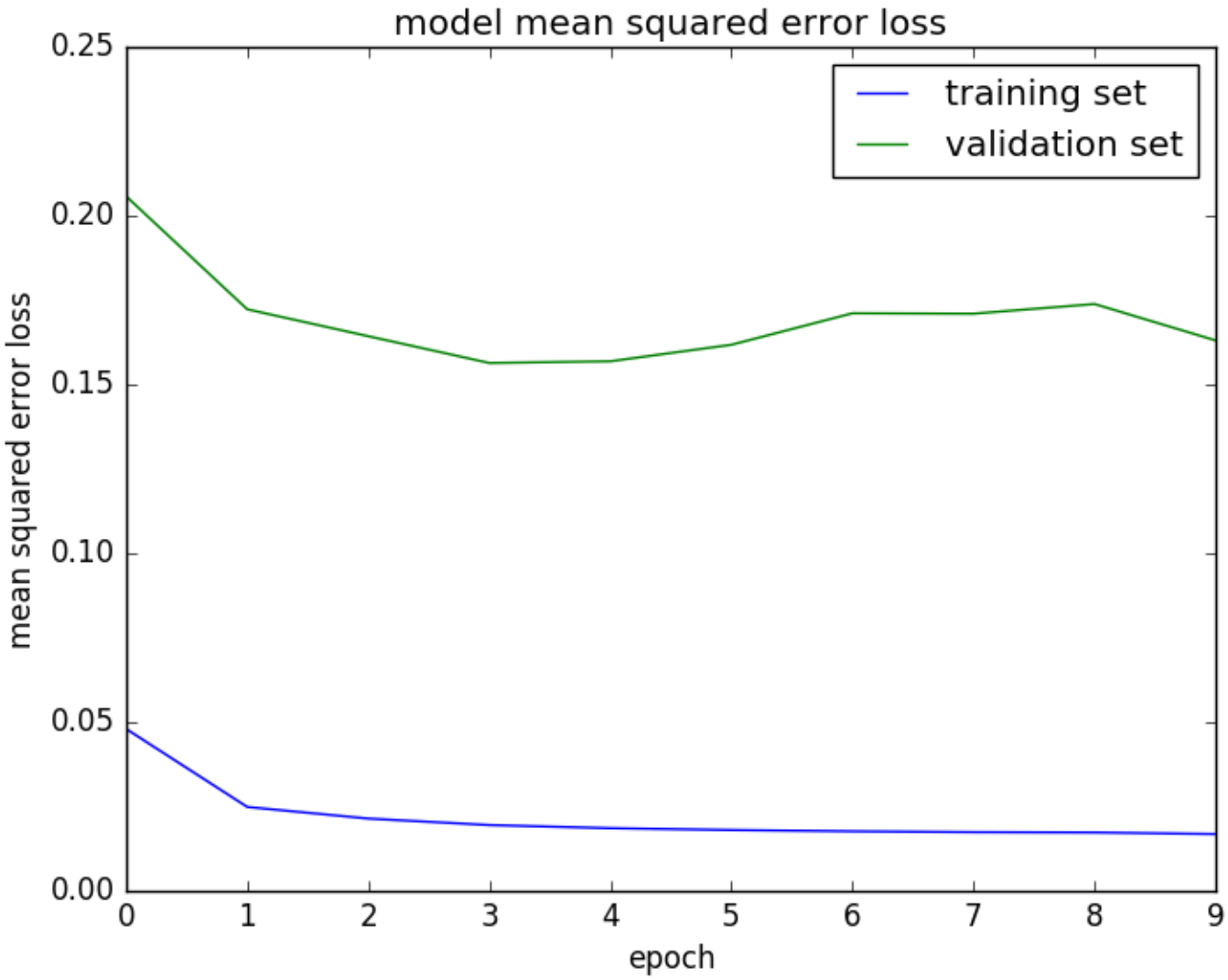


Figure 3: MSE for training and validation set across epoches