PROJECT REPORT

UDACITY

SELF-DRIVING CAR NANODEGREE

Project 4: Behavioral Cloning

AUTHOR:
Thomas Teh

Date: March 12, 2019

1 Introduction

This project is part of the Self-Driving Car Nanodegree by Udacity and its objective is to train a neural network model which predict steering angles from images. While it is encouraged to collect dataset using the simulator, I have only used the dataset provided by Udacity.

2 Data Splitting

The original data is shuffled and split into the training set and validation set. The training set size is 75% of the data and the remaining is the validation set.

3 Preprocessing and Data Augmentation

I augment the original data set by the following:

- **Flipping the images and angles:** As the first track in the simulator has more left turns than right turns, it is important to augment the data such that the neural network knows how to predict the steering angles given a right turn. This can be done by flipping the image horizontally and the negate the sign of the steering angles.
- **Translation:** The model failed to learn nor complete the track with only augmentation of the flipped images. I had to resolve to augment the data further with random translation along the x and y axis. The image is subjected to random translation of 50 pixels along x-axis and 20 pixels along the y-axis.
- **Cropping:** The cropping is done within the neural network model. The images are cropped at 70 pixels from the top and 25 pixels from the bottom. This is to remove any unneeded parts of an image like the body of the car, the sky and etc.
- **Random brightness:** An additional augmentation method by introducing random brightness was implemented. However, via my experiments, I did not find it useful in getting the neural network to work. Hence it is not used.

4 Architecture and Hyperparameters

The architecture that I used is shown below:

Input
Convolution Layer Kernel size: 3×3, N Kernels: 32, Activation: ELU, Stride: 2
Convolution Layer Kernel size: 3×3, N Kernels: 48, Activation: ELU, Stride: 2
Convolution Layer Kernel size: 3×3, N Kernels: 64, Activation: ELU, Stride: 2
Convolution Layer Kernel size: 3×3, N Kernels: 128, Activation: ELU, Stride: 1
Convolution Layer Kernel size: 3×3, N Kernels: 128, Activation: ELU, Stride: 1
Max Pooling Layer
Dense Layer 256 units, Activation: ELU
Drop Out Layer Drop out probability = 0.50
Dense Layer 128 units, Activation: ELU
Drop Out Layer Drop out probability = 0.50
Dense Layer 10 units, Activation: ELU
Dense Layer: 1 unit

The model is trained with ADAM optimization algorithm with the following hyperparameters:

Hyperparameters	Values
Epoch	10
Batch Size	96 (Post Augmentation)

I used the default values for the other hyperparameters for the ADAM optimizer.
Other comments:

- ELU activation works well and since it has regularization effect, I did not implement the batch normalization.
- Drop out layers are used to prevent overfitting since we have a large number of neurons in the dense layers.

5 Ideas for Improvements

The following are some ideas for improving the project:

- Collect the data for track 2 of the simulator and train the network to predict the steering angles.
- Implement the lane detection algorithm on the raw images to help the neural network to detect the lane so that the model will steer the car to stay in lane.
- Implement deep reinforcement learning on the problem.