CarND- Behavioral Cloning Project

The goals / steps of this project are the following:

- Use the simulator to collect data of good driving behaviour
- Build, a convolution neural network in Keras that predicts steering angles from the raw images
- Train and validate the model with a training and validation set
- Test that the model successfully drives around track one without leaving the road
- Summarise the results with a written report

1. Project includes all required files and can be used to run the simulator in autonomous mode

My project includes the following files:

- model.py containing the script to create and train the model
- drive.py for driving the car in autonomous mode
- model.h5 containing a trained convolution neural network
- writeup_report.pdf summarizing the results

2. Project includes functional code

To run the code , go to code directory, open simulator in one terminal and run following code in another terminal

```
python drive.py model.h5
```

3. Project code is usable and readable

The model.py file contains the code for training and saving the model as "model.h5"

Model Architecture and Training Strategy

1. Solution Design Approach

Model Summary

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
cropping2d_1 (Cropping2D)    (None, 65, 320, 3)        0
_________________________________________________________________
lambda_1 (Lambda)            (None, 65, 320, 3)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 31, 158, 24)       1824
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 14, 77, 36)        21636
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 5, 37, 48)         43248
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 3, 35, 64)         27712
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 1, 33, 64)         36928
_________________________________________________________________
flatten_1 (Flatten)          (None, 2112)              0
_________________________________________________________________
dense_1 (Dense)              (None, 100)               211300
_________________________________________________________________
dropout_1 (Dropout)          (None, 100)               0
_________________________________________________________________
dense_2 (Dense)              (None, 50)                5050
_________________________________________________________________
dense_3 (Dense)              (None, 10)                510
_________________________________________________________________
dense_4 (Dense)              (None, 1)                 11
=================================================================
Total params: 348,219
Trainable params: 348,219
Non-trainable params: 0
```

The overall strategy for deriving a model architecture was to ensure that the model completed a successful lap around the simulator.

My first step was to use a convolution neural network model similar to the one developed by the autonomous driving team at NVIDIA.

In order to gauge how well the model was working, I splited the dataset into a training and validation set into (80-20)%.

The model includes ELU as activation as suggested by the reviewer to introduce non linearity.

For regularization, I introduced random dropout (dropout probability: 25%) .

Then I recorded multiple laps in both directions to prevent bias.

The final step was to run the simulator to see how well the car was driving around track one.

2. Attempts to reduce overfitting in the model

To avoid overfitting, dropout layers were introduced in the model and the model was trained and validated on different data sets to ensure that the model was not overfitting. Recorded laps in both direction to generalise the left and right turn. The model was tested on the simulator to ensure that the vehicle is staying on the track.

3. Model parameter tuning

To train the model, we used the Adam optimizer minimizing mean squared error.

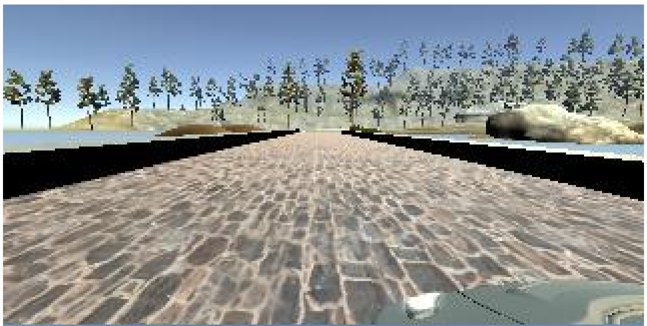
4. Appropriate training data

Training data was chosen to keep the vehicle driving on the road. I used a combination of center lane driving, recovering from the left and right sides of the road.

5. Creation of the Training Set & Training Process

Here is an example image of center lane driving:

I also made use of the left and right side camera images by adding and subtracting an angle of 0.2 from each of them respectively .



We don't need hills, tree and other non-road features, so I cropped the images.

After the collection process, I had more than 67528 number of images. Then I normalised the images.

I finally randomly shuffled the data set and used 80-20 ratio to split the data set.

There were a few spots where the vehicle fell off the track.

To improve the driving behaviour in these cases, I recorded a few more laps focused specifically on recovering vehicle once it has veered towards the side.

I also improved my augmenting process by flipping each image and inverting the angle.

Data set were collected based on the following strategy:

1. Center driving on 1st track - 2 laps
2. Center driving on 1st track (opposite direction) - 2 laps
3. Recovery driving from both sides - 1 lap

Images were recorded and stored in IMG folder. Below are the sample images from the dataset that are used to train the model.

Results

The car is able to drive autonomously around the track without leaving the road. Output video file is attached with name "output.mp4"