Project 3:Behavioral Cloning

1. Files Submitted and Code Quality:

The submission includes all required files which 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 (not changed kept in its originality as provided in the Udacity git hub).
- **model.h5** containing a trained convolution neural network.
- **writeup_report.pdf** summarizing the results.
- **track1.mp4** and **track2.mp4** are the recoded videos which was run in autonomous mode for track1 and track2 provided in simulator respectively.

2. Submission includes functional code:

Using the Udacity provided simulator and Udacity provided **drive.py** file (nothing was changed as the file downloaded had constant speed), the car can be driven autonomously around the track by executing the command:

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

3. Submission code is usable and readable:

The model.py file contains the code for training and saving the convolution neural network. The file shows the pipeline used for training and validating the model and it contains comments to explain how the code works.

Model Architecture and Training Strategy

1. An appropriate model architecture has been employed

Deep Neural Network involving convolution layers with rectified linear activations, fully connected layers using Keras. The output size of each layer varies from 36 to 96 for convolution networks and 100 down to 1 for fully connected layers. The filter size varies from 3x3 to 2x2

2. Attempts to reduce over-fitting in the model

Several dropout layers with the percentage of dropout varying from 10 to 40 has been introduced in convolution network

The model was trained and validated on different data sets to ensure that the model was not overfitting. The model was tested by running it through the simulator and ensuring that the vehicle could stay on the track. 20% of data was used for validation.

3. Model parameter tuning

The optimization used is Adaptive Motion Estimation, which is ,with Mean Squared Error loss function.

4. Appropriate training data

The data collected from center lane driving in track 1 and track2, recovery driving in track 1 and smooth driving in sharp curves were captured. Even the side cameras were used as input and a correction factor as mentioned in the example was used.

Model Architecture and Training Strategy

1. Solution Design Approach

The solution has been derived from the nvidia [1] autonomous architecture. Initially the same model as depicted in [1] was used. It could drive only on first track and could drive for a short while on second track.

During repetitive attempts to make the model work for both the tracks an additional convolution layer was added and the outputs of each layer was increased each time so the initial layer had output layers increasing from 12 to 36. Finally dropout layers were introduced to avoid over-fitting of data

2. Final Model Architecture

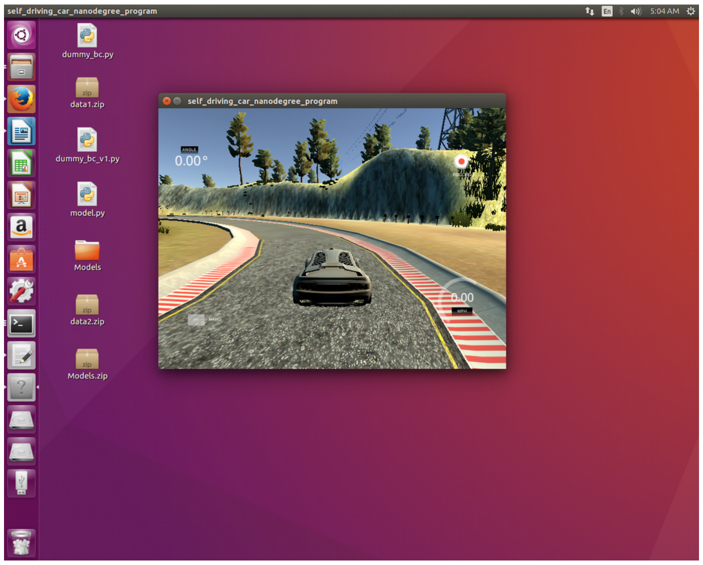
The model architecture consists of following layers in the same order:

- Lambda normalization layer
- Cropping the image
- Convolution layer with kernel size 3x3 and output size 36 and subsampling by 2x2
- Dropout layer with 10% drop of output
- Convolution layer with kernel size 3x3 and output size 48 and sub sampling by 2x2
- Dropout layer with 20% drop of output
- Convolution layer with kernel size 3x3 and output size 60 and sub sampling by 2x2
- Dropout layer with 30% drop of output
- Convolution layer with kernel size 3x3 and output size 72 and sub sampling by 2x2
- Dropout layer with 40% drop of output
- Convolution layer with kernel size 2x2 and output size 84
- Convolution layer with kernel size 2x2 and output size 84

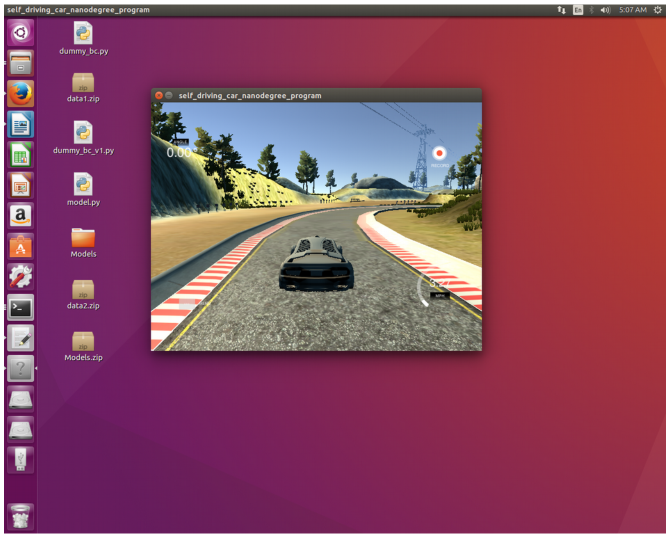
- flat connected layer reducing the output from 100 to 1

3.Creation of the Training Set & Training Process

- Center oriented -left oriented drive on track1- with 4 laps. Example sample is attached below:



- Center oriented-right oriented drive on track2- with 4 laps (car reversed and driven)



- Recovery lap on track1 with both left oriented and left oriented driving 2 laps and smooth driving around the sharp curves-1 lap
- Track2- 4laps (2 laps with the car driving in the way it was placed by the simulator and 2 laps with car driving in the opposite direction placed by the simulator .

[1] “End to End Learning of Self Driving Cars”, Mariusz Bojarski, et.al, 26 April 2016