Report Behavior Cloning

I will list the rubric points and how I have fulfilled these points individually(below).

These files referenced are in fulfillment of the Project 4, Behavior Cloning in Udacity Self Driving Engineer.

Specifications:

- Are all required files submitted? The submission includes a model.py file, drive.py, model.h5 a writeup report, readme.pdf and video.mp4.

The following files have been submitted model.py, drive .py model.h5 this report(writeup, "Report Behavior Cloning.pdf"), and "run1.mp4" (video.mp4), Readme.pdf

- Is the code functional? The model provided can be used to successfully operate the simulation.

Ran the car autonomously around the track in a terminal window in AWS system with the following command (run1 is the directory that the images are located from the autonomous mode):

python drive.py model.h5 run1

Then executing the simulator and selection Autonomous mode.

Then take images folder (run1) and made a video with the following command:

Python video.py run1

- Is the code usable and readable? The code in `model.py` uses a Python generator, if needed, to generate data for training rather than storing the training data in memory. The `model.py` code is clearly organized, and comments are included where needed.

The program is commented and describes the functionality of the code

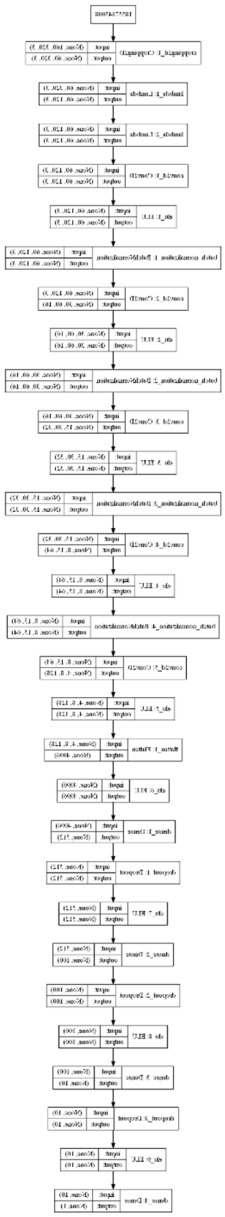
The model.py python file contains code for training and the neural network, the neural network model saved as 'model.h5'. The python file is well commented on how the code works.

- Has an appropriate model architecture been employed for the task? The neural network uses convolution layers with appropriate filter sizes. Layers exist to introduce nonlinearity into the model. The data is normalized in the model.

table 1a Model Architecture model architecture (model2F.py)

Layer (type)	Output Shape	Param #
cropping2d_1 (Cropping2D)	(None, 60, 320, 3)	0
lambda_1 (Lambda)	(None, 60, 120, 3)	0
lambda_2 (Lambda)	(None, 60, 120, 3)	0
conv2d_1 (Conv2D)	(None, 60, 120, 3)	12
elu_1 (ELU)	(None, 60, 120, 3)	0
batch_normalization_1 (Batch	(None, 60, 120, 3)	12
conv2d_2 (Conv2D)	(None, 30, 60, 16)	1216
elu_2 (ELU)	(None, 30, 60, 16)	0
batch_normalization_2 (Batch	(None, 30, 60, 16)	64
conv2d_3 (Conv2D)	(None, 15, 30, 32)	12832
elu_3 (ELU)	(None, 15, 30, 32)	0
batch_normalization_3 (Batch	(None, 15, 30, 32)	128
conv2d_4 (Conv2D)	(None, 8, 15, 64)	18496
elu_4 (ELU)	(None, 8, 15, 64)	0
batch_normalization_4 (Batch	(None, 8, 15, 64)	256
conv2d_5 (Conv2D)	(None, 4, 8, 128)	73856
elu_5 (ELU)	(None, 4, 8, 128)	0
flatten_1 (Flatten)	(None, 4096)	0
elu_6 (ELU)	(None, 4096)	0
dense_1 (Dense)	(None, 512)	2097664
dropout_1 (Dropout)	(None, 512)	0
elu_7 (ELU)	(None, 512)	0
dense_2 (Dense)	(None, 100)	51300
dropout_2 (Dropout)	(None, 100)	0
elu_8 (ELU)	(None, 100)	0
dense_3 (Dense)	(None, 10)	1010
dropout_3 (Dropout)	(None, 10)	0
elu_9 (ELU)	(None, 10)	0
dense_4 (Dense)	(None, 1)	11

Total params: 2,256,857
Trainable params: 2,256,627
Non-trainable params: 230



This model is the NVidia model.

Data was normalized with the following command:

```
# Normalize the data

model.add(Lambda(lambda x: (x/127.5) - 0.5))
```

Nonlinearity were introduced with ELU and dropouts in the model

- Has an attempt been made to reduce overfitting of the model? Train/validation/test splits have been used, and the model uses dropout layers or other methods to reduce overfitting.

Dropouts add after the flatten layer to reduce overfitting. Train/Validation test splits have been used(0.80/0.20) have been used. Also by reducing the number of epocs to 11 the optimum for me .

- **Have the model parameters been tuned appropriately?** Learning rate parameters are chosen with explanation, or an Adam optimizer is used.

An Adam optimizer was used.

- **Is the training data chosen appropriately?** Training data has been chosen to induce the desired behavior in the simulation (i.e. keeping the car on the track).

Example images below:

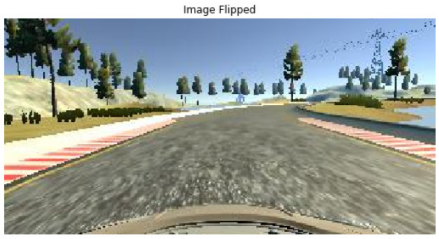
- Is the solution design documented? The README thoroughly discusses the approach taken for deriving and designing a model architecture fit for solving the given problem.

The images collected 54,242, 41760 were used for training, 10448 were used for validation these images were flipped using the following command and so was the steering measurement:

```
augment_images.append(cv2.flip(image,1))

augmented_measurements.append(measurement*-1.0)
```

Below is an example of a flipped image:



The brightness was altered so all were not the same brightness, I used a uniform distribution that adjust the brightness of the images from 0.25(dark) to 1.0(bright).

Also there are three camera's left, center, and right. The left camera is positioned slightly to the left of the vehicle center line, while the right camera is positioned slightly to the right of the vehicle center line. All of these camera's were used in the training set as follows:

first add measurement data with an offset correction for left and right, no correction for center camera.

```
        correction = 0.1
    measurement = float(line[3])
    # Center CAM
    if i == 0 :
      measurements.append(measurement)
    # Left CAM
    elif i == 1 :
      measurements.append(measurement+correction)
    #  Right CAM
    else:
      measurements.append(measurement-correction)
```

- Is the model architecture Is the creation of the training dataset and training process documented? The README provides sufficient details of the characteristics and qualities of the architecture, such as the type of model used, the number of layers, the size of each layer. Visualizations emphasizing particular qualities of the architecture are encouraged.

Earlier in this document table 1 was presented with the model architecture.

- Is the creation of the training dataset and training process documented? The README provides sufficient details of the characteristics and qualities of the architecture, such as the type of model used, the number of layers, the size of each layer. Visualizations emphasizing particular qualities of the architecture are encouraged. Here is one such tool for visualization.The README describes how the model was trained and what the characteristics of the dataset are. Information such as how the dataset was generated and examples of images from the dataset must be included. The README describes how the model was trained and what the characteristics of the dataset are.

Information such as how the dataset was generated and examples of images from the dataset must be included.

The training set discussion was earlier

- **Is the car able to navigate correctly on test data?** No tire may leave the drivable portion of the track surface. The car may not pop up onto ledges or roll over any surfaces that would otherwise be considered unsafe(if humans were in the vehicle).

The vehicle made it around the track successfully and did not leave the road surface.

-