SDC - Project 3

Behavioral Cloning

```
The goals / steps of this project are the following:
* Use the simulator to collect data of good driving behavior
* Build, a convolution neural network in Keras that predicts steering angles from i
mages
* Train and validate the model with a training and validation set
* Test that the model successfully drives around track one without leaving the road
* Summarize the results with a written report
```

Files Submitted & Code Quality

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

My project includes the following files:

- clean_videos.py containing code a used to scrub training runs to ensure I only captured appropriate training data
- 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
- model.ipynb jupyter notebook showing the work done to create the model.py file and various visuzalization, cross checks
- writeup.pdf (This file)

2. Submission includes functional code

Using the Udacity provided simulator and my drive.py file, the car can be driven autonomously around the track by executing

```
1 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 I used for training and validating the model, and it contains comments to explain how the code works.

Model Architecture and Training Strategy

Model Architecture

My model consists of the following layers:

1. reprocess lambda -> image nomrmalization
2. conv1 -> 32 filters 5x5 with 2x2 pulling and relu activation
3. conv2 -> 64 filters 5x5 with 2x2 pooling and relu activation
4. conv3 -> 128 filters 3x3 with 2x2 pooling and relu activation
5. conv4 -> 128 filters 2x2 with 2x2 pooling and relu activation
6. Flatten layer
7. fc1 -> 1024 neurons with .5 dropout and relu activation
8. fc2 -> 128 neuron with .5 dropout and relu activation
9. fc3 -> 64 neuron with .5 dropout and relu activation
10. output layer with 1 output

compiled with Mean square error loss and ADAM optimization

Appropriate training data

Training data was chosen to keep the vehicle driving on the road. I used a combination of center lane driving, and recovering from the left and right sides of the road. Three cameras were used to capture video. I augmented the center camera data with side cameras and added a bias of +- .12 to the steering angle to compensate for the camera being on the side of the car. For each image, I also flipped it horizontally and inverted the steering angle. So for any given frame, I generated 6 images with 6 steering unique steering angles. I created a training set of approximately 120,000 images/angle combinations.

3. Creation of the Training Set & Training Process

To capture good driving behavior, I two good center line laps in both directions on both tracks one 1-2 laps in either direction or recover (edge of the road back to center). I then used the left and right, modified the steering angle and flipped each image and steering angle to generate 6 images per frame. I generated roughly 120,000 frames. To make sure I was feeding the network good data, I scrubbed the videos 5 frames at a time to find "bad driving" and remove it. See clean_videos.py for the preprocessing that was done.