README_Naman.md

Behavioral-Cloning

Udacity Self Driving Car Engineer Nanodegree - Project 3

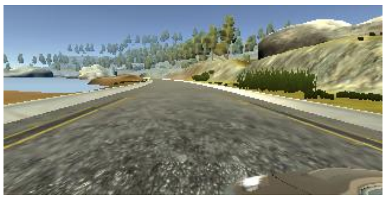
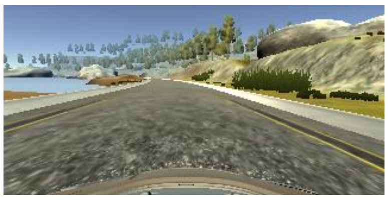
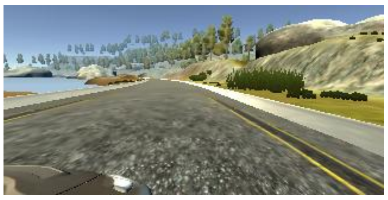
The goal of this project is to train a vehicle to drive autonomously in a simulator by taking camera images as inputs to a deep neural network and outputting a predicted steering angle based on the image.

The project is broken up in to 4 distinct stages:

- Stage 1: Data Aquisition
- Stage 2: CNN Model Architecture
- Stage 3: Data Cleaning, Preprocessing and Augmentation
- Stage 4: Final Architecture and Model Training
- Stage 5: Testing the Model

Data Aquisition:

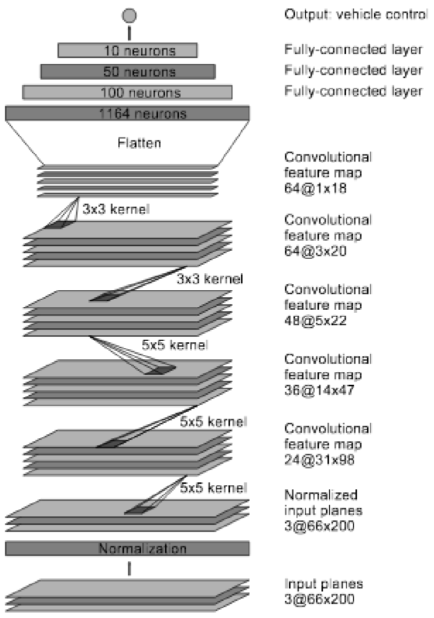
The veicle was drive in the simulator in the training mode and camera data (left , center , right) and vehicle control intputs data(steering, brake and throttle) was recored and stored to be used for training the model

Left	Center	Right
		

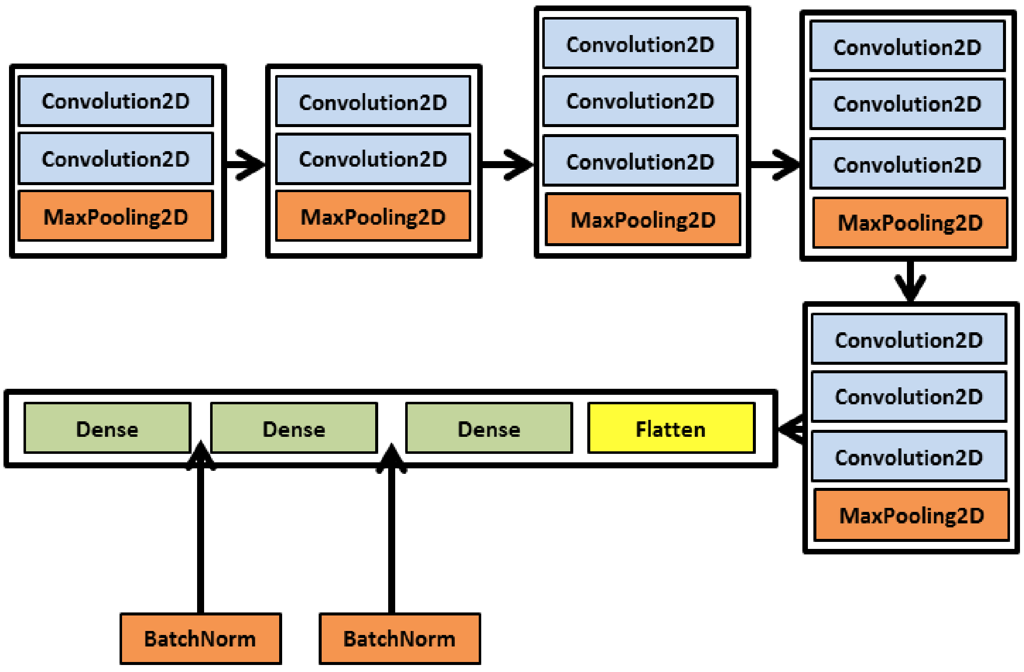
Model Architecture

Following three approaches were tried to complete this project

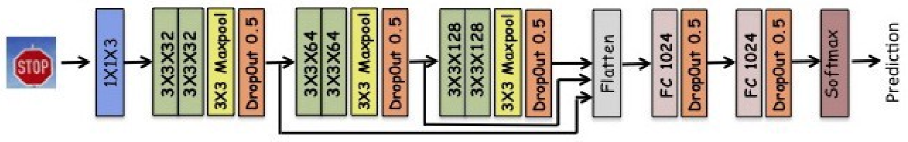
1. Using the Nvidia's End to End Learning for Self-Driving Cars model architecture.The figure below shows the various layers of the model



1. Transfer Learing by retuning the Imagenet Model with Batch Normalization. This model and its pre-trained weights are taken from Fastai website on Deep Learning. The model is similar to VGG16 with two additional Batch Normalization layers added with the Dense layers. Figure below shows the new model architecture



1. Self -defined model based on the feedback that i recieved on my second Udacity project (Traffic Sign



Classification)

Model architecture

Data Cleaning, preprocessing and augmentation

The following techniques were adopted for preprocessing augmentation:

- Cropping the images to remove the bacgground noise and reduce the processing speed. The images are cropped to 66x200

- Reducing the straight line driving data
- Jittering and brightness change is randomly applied to the data
- Steering adjustement of 0.25 angle (+ for left and - for right images)
- To compensate for the left turning, 50% of images were flipped and their steering angle is reversed

Final Architecture and Model Training

After trying all the approaches I chose to use the NVIDIA model as it gives good results and is relatively faster than the self defined model Adam optimizer with learning rate = 0.001 and Mean Squared Error loss method was used to train the model. The model was trained using early stopping callback which discontinues training when validation loss fails to improve for consecutive epochs. When training is complete the model and weights are saved to be used for autonomous driving in the simulator.

Model Testing

The model was tested using the autonomous driving mode in the simulator. The saved model and weights are used to predict the based on the stream of images fed to the model via drive.py script. Throttle is kept constant at .35 The predicted steering angle is passed to the car as a control and the car steers itself accordingly.

The car drives safely around the course for multiple laps without hitting the curbs or going off of track.