Behavioral Cloning Project

The goals / steps of this project are the following:

- Use the simulator to collect data of good driving behavior
- Build, a Nividia convolution neural network in Keras that predicts steering angles from 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
- Summarize the results with a written report

Rubric Points

Here I will consider the rubric points (https://review.udacity.com/#!/rubrics/432/view) individually and describe how I addressed each point in my implementation.

Files Submitted & Code Quality

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.md or writeup_report.pdf summarizing the results

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

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

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

1.The model consists of a Nividia convolution neural network with 5 convolutional layers and four fully connected layers.

2.The model includes ELU layers to introduce nonlinearity. The data is normalized and cropped into 60x320 in the model using a Keras lambda layer and cropping layer.

3.The model in convolutional layer contains 2x2 subsampling layers in order to reduce overfitting. And it was trained and validated on different data sets to ensure that the model was not overfitting.

4.The model was tested by running it through the simulator and ensuring that the vehicle could stay on the track.

5.The model used an adam optimizer, with the learning rate 0.0001.

6.Training data was chosen to keep the vehicle driving on the road. I just used a combination of center lane driving and cropping each one into a 60x320 size.

Model Architecture and Training Strategy

1.Solution Design Approach

The overall strategy for deriving a model architecture was to try from the simple to the complex.

My first step was to use a convolution neural network model similar to the one shown in the lecture but I thought the model might not be appropriate because the result of the model has already been shown on the lecture but I just want a try and get some experience.

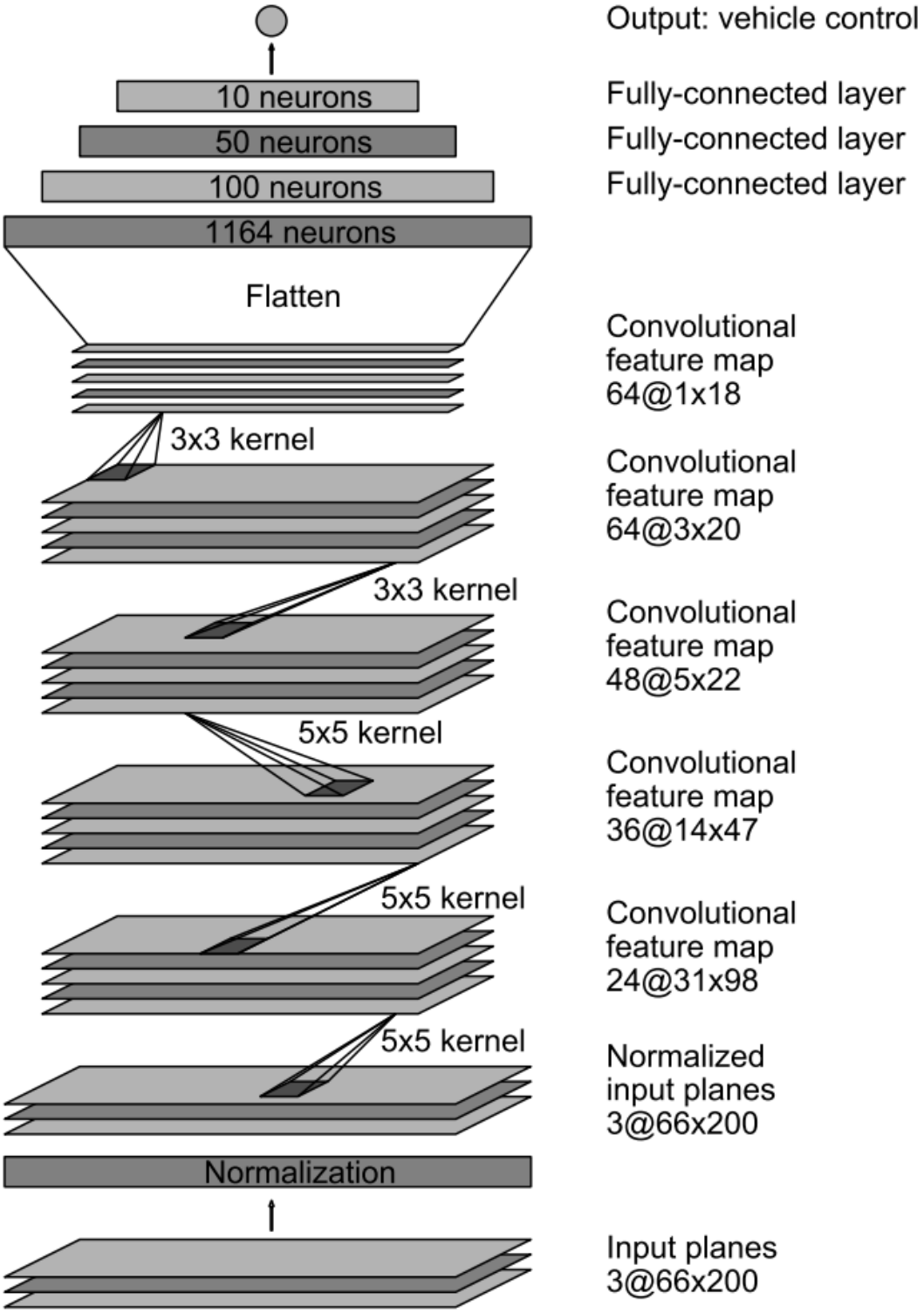
In order to gauge how well the model was working, I split my image and steering angle data into a training and validation set. I found that my first model had a low mean squared error on the training set and a low accuracy and it works no well(the car will go off the road). So I come to try the Nividia model and trying different combinations of dropout layer and subsampling on the convolutional layer and finally get the appropriate model.

The final step was to run the simulator to see how well the car was driving around track one. There were a few spots where the vehicle fell off the track like the first left turning, the bridge and the intersection after the bridge. In order to improve the driving behavior in these cases, I increase the training epoch from 3 to 6.

At the end of the process, the vehicle is able to drive autonomously around the track without leaving the road.

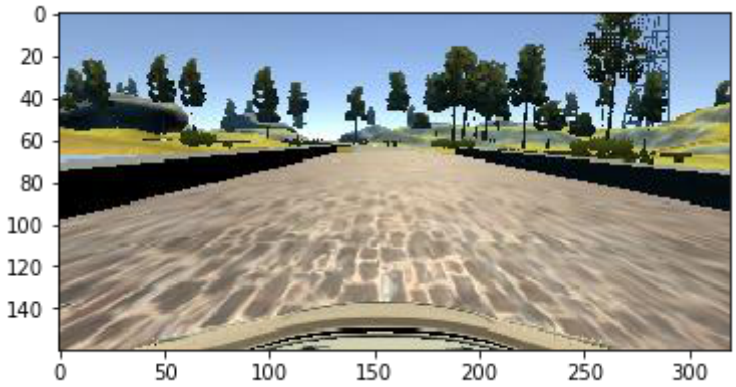
2.Final Model Architecture

The final model architecture consisted of a convolution neural network with the 5 convolutional layers size of 5x5, 5x5, 5x5, 3x3, 3x3 and depths 24, 48, 64, 64, 64. And the four fully connected layers with size 100,50,10,1. Here is a visualization of the architecture:

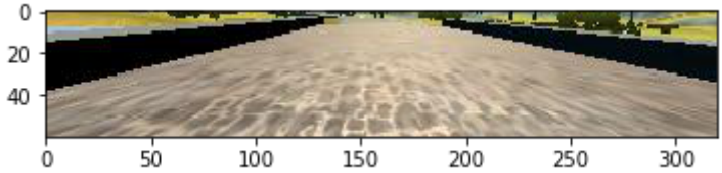


3.Creation of the Training Set & Training Process

To capture good driving behavior, I cropped the images from 160x320 to 60x320. Here is an example of the original image of center lane driving:



I then cropped the center images to the images with size 60x320 shown below:



I used this training data for training the model. The validation set helped determine if the model was over or under fitting. The ideal number of epochs was 6 as evidenced by the final performance of the car on the road.

In [ ]: