#**Behavioral Cloning**

##Writeup Template

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

####1. Submission includes all required files and 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

* **“model.h5“** containing a trained convolution neural network

* **“BehavioralCloning.pdf”** summarizing the results

####2. Submission includes functional code

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

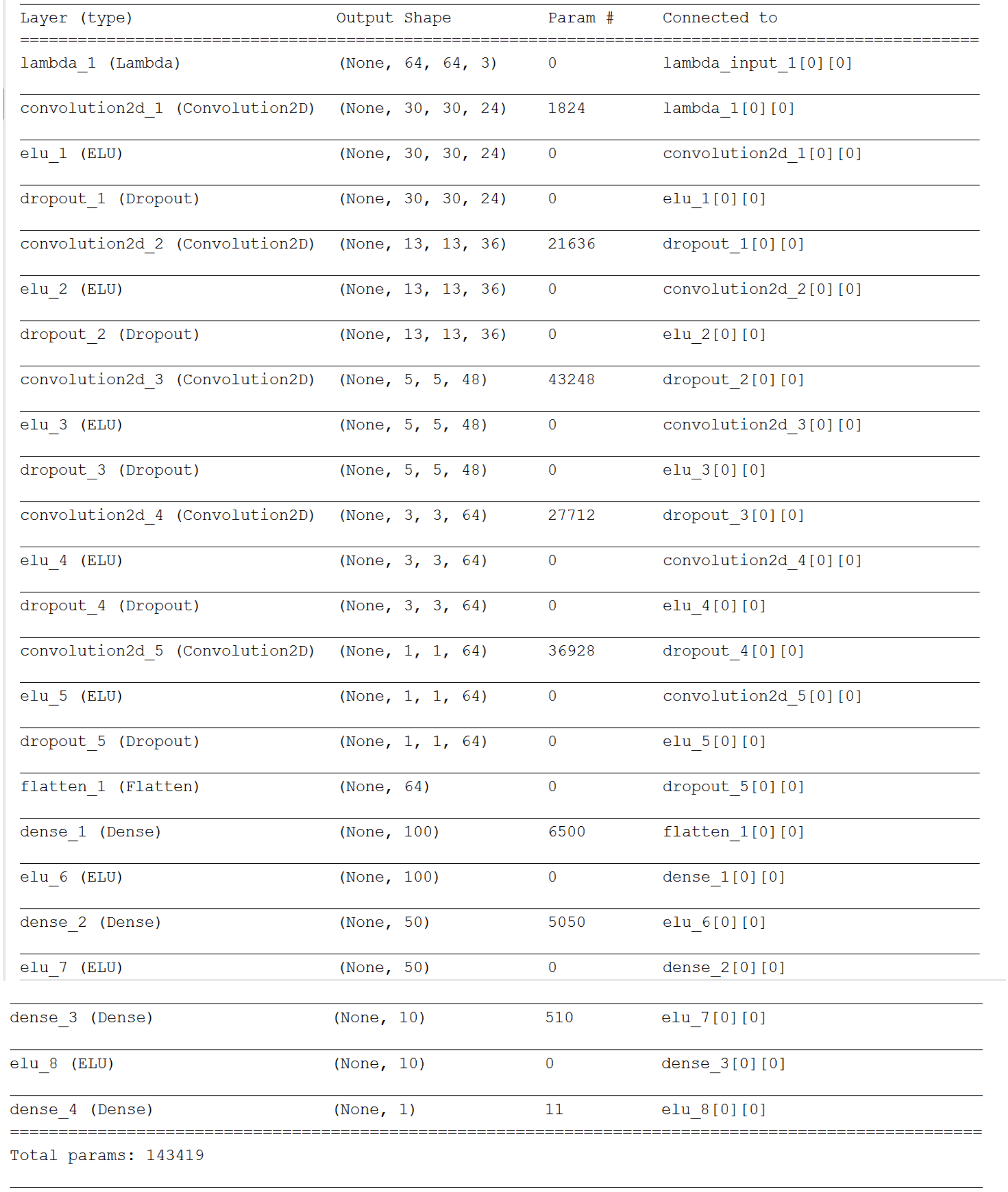
####3. Submission code is usable and readable

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

####1. An appropriate model architecture has been employed

Similar with NVIDA model, My model consists of a convolution neural network with **three times 5x5 filter and 2 times 3x3 filter** (model.py **lines 299-333**) **. See network structure summary in below:**



```
Layer (type)                     Output Shape          Param #     Connected to
====================================================================================================
lambda_1 (Lambda)                (None, 64, 64, 3)     0           lambda_input_1[0][0]
____________________________________________________________________________________________________
convolution2d_1 (Convolution2D)  (None, 30, 30, 24)    1824        lambda_1[0][0]
____________________________________________________________________________________________________
elu_1 (ELU)                      (None, 30, 30, 24)    0           convolution2d_1[0][0]
____________________________________________________________________________________________________
dropout_1 (Dropout)              (None, 30, 30, 24)    0           elu_1[0][0]
____________________________________________________________________________________________________
convolution2d_2 (Convolution2D)  (None, 13, 13, 36)    21636       dropout_1[0][0]
____________________________________________________________________________________________________
elu_2 (ELU)                      (None, 13, 13, 36)    0           convolution2d_2[0][0]
____________________________________________________________________________________________________
dropout_2 (Dropout)              (None, 13, 13, 36)    0           elu_2[0][0]
____________________________________________________________________________________________________
convolution2d_3 (Convolution2D)  (None, 5, 5, 48)      43248       dropout_2[0][0]
____________________________________________________________________________________________________
elu_3 (ELU)                      (None, 5, 5, 48)      0           convolution2d_3[0][0]
____________________________________________________________________________________________________
dropout_3 (Dropout)              (None, 5, 5, 48)      0           elu_3[0][0]
____________________________________________________________________________________________________
convolution2d_4 (Convolution2D)  (None, 3, 3, 64)      27712       dropout_3[0][0]
____________________________________________________________________________________________________
elu_4 (ELU)                      (None, 3, 3, 64)      0           convolution2d_4[0][0]
____________________________________________________________________________________________________
dropout_4 (Dropout)              (None, 3, 3, 64)      0           elu_4[0][0]
____________________________________________________________________________________________________
convolution2d_5 (Convolution2D)  (None, 1, 1, 64)      36928       dropout_4[0][0]
____________________________________________________________________________________________________
elu_5 (ELU)                      (None, 1, 1, 64)      0           convolution2d_5[0][0]
____________________________________________________________________________________________________
dropout_5 (Dropout)              (None, 1, 1, 64)      0           elu_5[0][0]
____________________________________________________________________________________________________
flatten_1 (Flatten)              (None, 64)            0           dropout_5[0][0]
____________________________________________________________________________________________________
dense_1 (Dense)                  (None, 100)           6500        flatten_1[0][0]
____________________________________________________________________________________________________
elu_6 (ELU)                      (None, 100)           0           dense_1[0][0]
____________________________________________________________________________________________________
dense_2 (Dense)                  (None, 50)            5050        elu_6[0][0]
____________________________________________________________________________________________________
elu_7 (ELU)                      (None, 50)            0           dense_2[0][0]
____________________________________________________________________________________________________
dense_3 (Dense)                  (None, 10)            510         elu_7[0][0]
____________________________________________________________________________________________________
elu_8 (ELU)                      (None, 10)            0           dense_3[0][0]
____________________________________________________________________________________________________
dense_4 (Dense)                  (None, 1)             11          elu_8[0][0]
====================================================================================================
Total params: 143419
____________________________________________________________________________________________________
```

The model includes ELU layers to introduce nonlinearity , and the data is normalized in the model using a Keras lambda layer (code line **300**).

####2. Attempts to reduce overfitting in the model

The model contains dropout layers in order to reduce **overfitting (model.py lines between 305 and 318).**

The model was trained and validated on **different data sets** to ensure that the model was not overfitting (code line **between 27 and 91**).

####3. Model parameter tuning

The model used an adam optimizer, so **the learning rate was manually tuned to 0.0001.** (model.py **line 327**).

####4. Appropriate training data

Training data was chosen to keep the vehicle driving on the road. I used a combination of center lane driving, recovering from the left and right sides of the road.

0.25 is added to the steering angle for the images from left camera.

0.25 is reduced to the steering angle for the images from right camera. See model.py line 87 to 91.

Besides, **three different data sets are used for training including udacity data, normal driving data and data with opposite driving direction**.