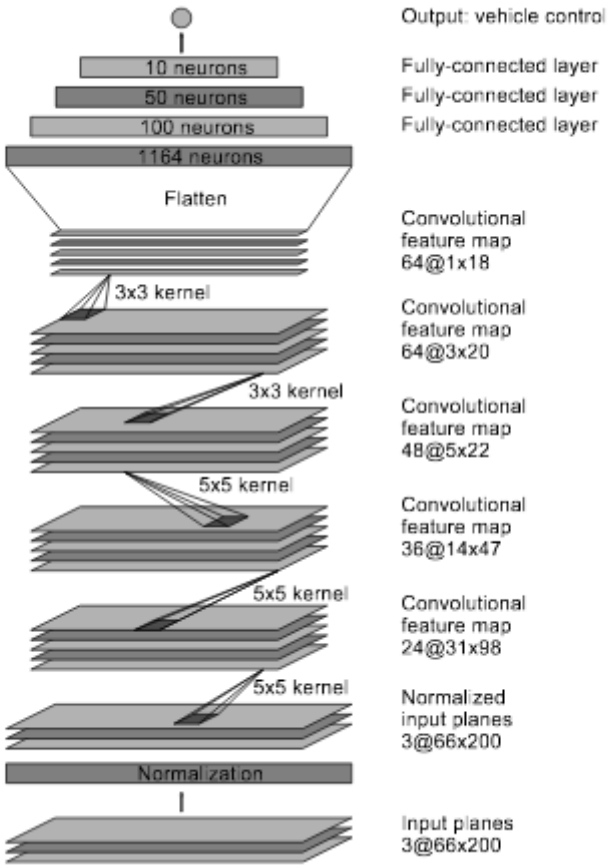
Model Architecture

I decided to go with the NVIDIA architecure as it is a model which is proven in practice. The architecure of the model is as follows.



Nvidia model - keras implementation.

```
model = Sequential()
    model.add(Lambda(lambda x: (x / 127.5) - 1., input_shape=(66, 200,
3)))
    model.add(Convolution2D(24, 5, 5, subsample=(2, 2), border_mode="vali
d", activation="elu"))
    model.add(Convolution2D(36, 5, 5, subsample=(2, 2), border_mode="vali
d", activation="elu"))
    model.add(Dropout(.5))
    model.add(Convolution2D(48, 5, 5, subsample=(2, 2), border_mode="vali
d", activation="elu"))
    model.add(Dropout(.5))
    model.add(Convolution2D(64, 3, 3, border_mode="valid", activation="el
u"))
    model.add(Dropout(.5))
    model.add(Convolution2D(64, 3, 3, border_mode="valid", activation="el
u"))
    model.add(Dropout(.5))
    model.add(Flatten())
    model.add(Dense(100,activation='elu'))
    model.add(Dropout(.25))
    model.add(Dense(50,activation='elu'))
    model.add(Dense(10,activation='elu'))
    model.add(Dense(1))
```

In [5]:

```
# Model details
# _________________________________________________________________
_____________________
# Layer (type)                 Output Shape          Param #     Connected t
o
# =================================================================
=====================
# lambda_1 (Lambda)            (None, 66, 200, 3)    0           lambda_inpu
t_1[0][0]
# _________________________________________________________________
_____________________
# convolution2d_1 (Convolution2D)  (None, 31, 98, 24)    1824        lambda_1[0]
[0]
# _________________________________________________________________
_____________________
# convolution2d_2 (Convolution2D)  (None, 14, 47, 36)    21636       convolution
2d_1[0][0]
# _________________________________________________________________
_____________________
# dropout_1 (Dropout)          (None, 14, 47, 36)    0           convolution
2d_2[0][0]
# _________________________________________________________________
_____________________
# convolution2d_3 (Convolution2D)  (None, 5, 22, 48)     43248       dropout_1
[0][0]
# _________________________________________________________________
_____________________
```

```
# dropout_2 (Dropout)              (None, 5, 22, 48)     0           convolution
2d_3[0][0]
# ____________________________________________________________________________
____________________
# convolution2d_4 (Convolution2D)  (None, 3, 20, 64)     27712       dropout_2
[0][0]
# ____________________________________________________________________________
____________________
# dropout_3 (Dropout)              (None, 3, 20, 64)     0           convolution
2d_4[0][0]
# ____________________________________________________________________________
____________________
# convolution2d_5 (Convolution2D)  (None, 1, 18, 64)     36928       dropout_3
[0][0]
# ____________________________________________________________________________
____________________
# dropout_4 (Dropout)              (None, 1, 18, 64)     0           convolution
2d_5[0][0]
# ____________________________________________________________________________
____________________
# flatten_1 (Flatten)              (None, 1152)          0           dropout_4
[0][0]
# ____________________________________________________________________________
____________________
# dense_1 (Dense)                  (None, 100)           115300      flatten_1
[0][0]
# ____________________________________________________________________________
____________________
# dropout_5 (Dropout)              (None, 100)           0           dense_1[0]
[0]
# ____________________________________________________________________________
____________________
# dense_2 (Dense)                  (None, 50)            5050        dropout_5
[0][0]
# ____________________________________________________________________________
____________________
# dense_3 (Dense)                  (None, 10)            510         dense_2[0]
[0]
# ____________________________________________________________________________
____________________
# dense_4 (Dense)                  (None, 1)             11          dense_3[0]
[0]
# ============================================================================
======================
# Total params: 252,219
# Trainable params: 252,219
# Non-trainable params: 0
```

Attempts to reduce overfitting

Initially RELU activation were used for the architecture and using ELU provided better results on the track.Dropout is used to reduce overfitting of the model. Adam optimiser with a learning rate of .0001 was used.

Since the model is trained from scratch and the higher level features were not transferred using transfer learning, it became evident that desired model might not be the final model after the last epoch. The model was checkpointed after every epoch to examine the performace of the model on the track. It helps examine how the model is evolving over the epochs.

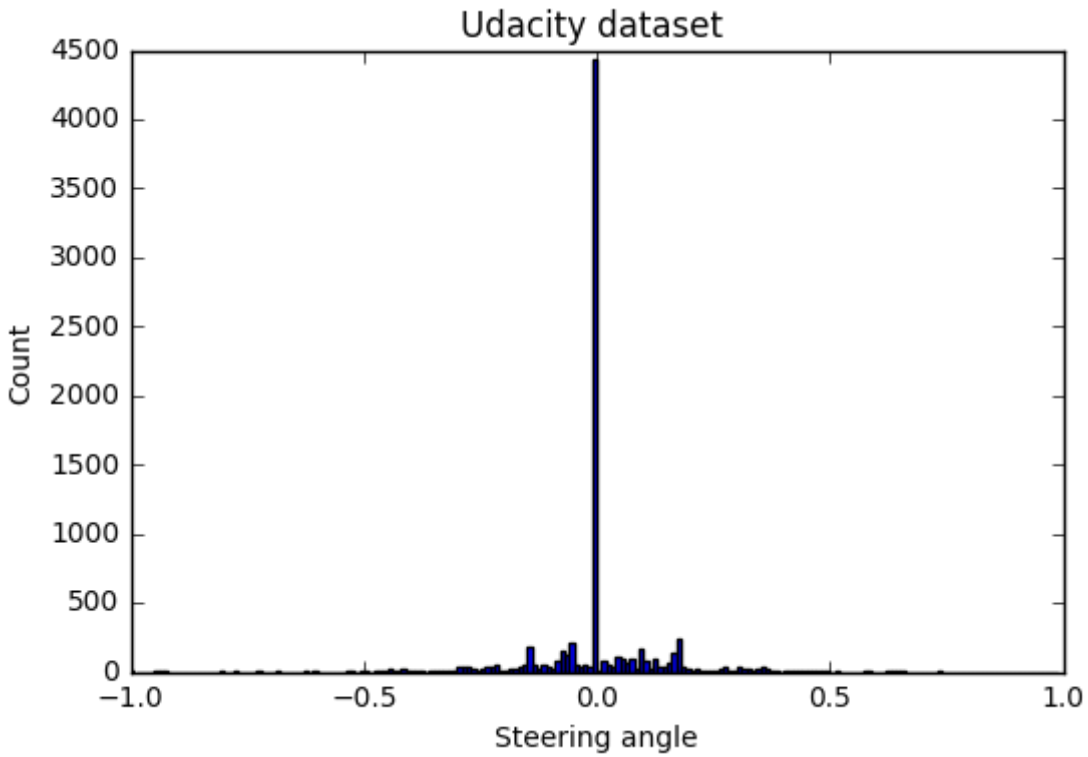
Model Parameters

- Samples per epoch - 81920
- Number of epoch - 10

Training Data Collection and Exploration

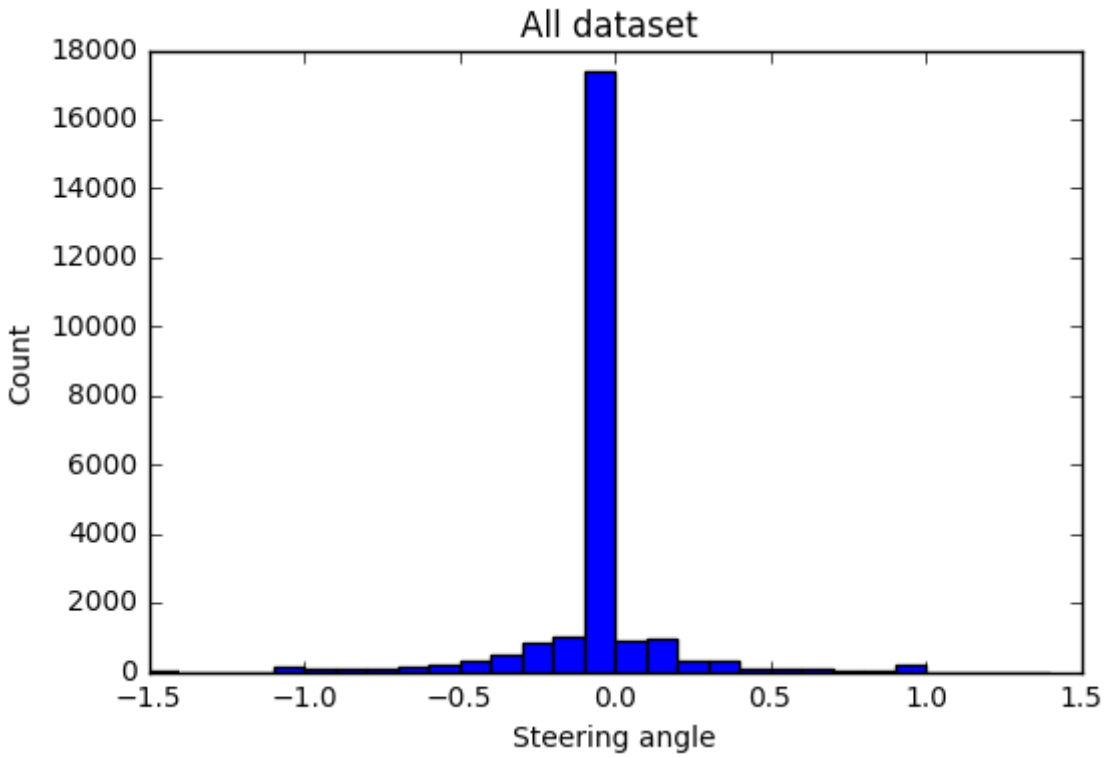
- The dataset provided by udacity is dominated by zero steering angles values.
- When the model was trained using the udacity dataset. The car was not able to make sharp turns or recover back to th e center of the road.
- To augment the udacity dataset more general driving,sharp turns, and recovery video was collected.
- The final dataset was an amalgamation of the above dataset.

Udacity provided data



Collected data + udacity data

Size of the dataset - 23771

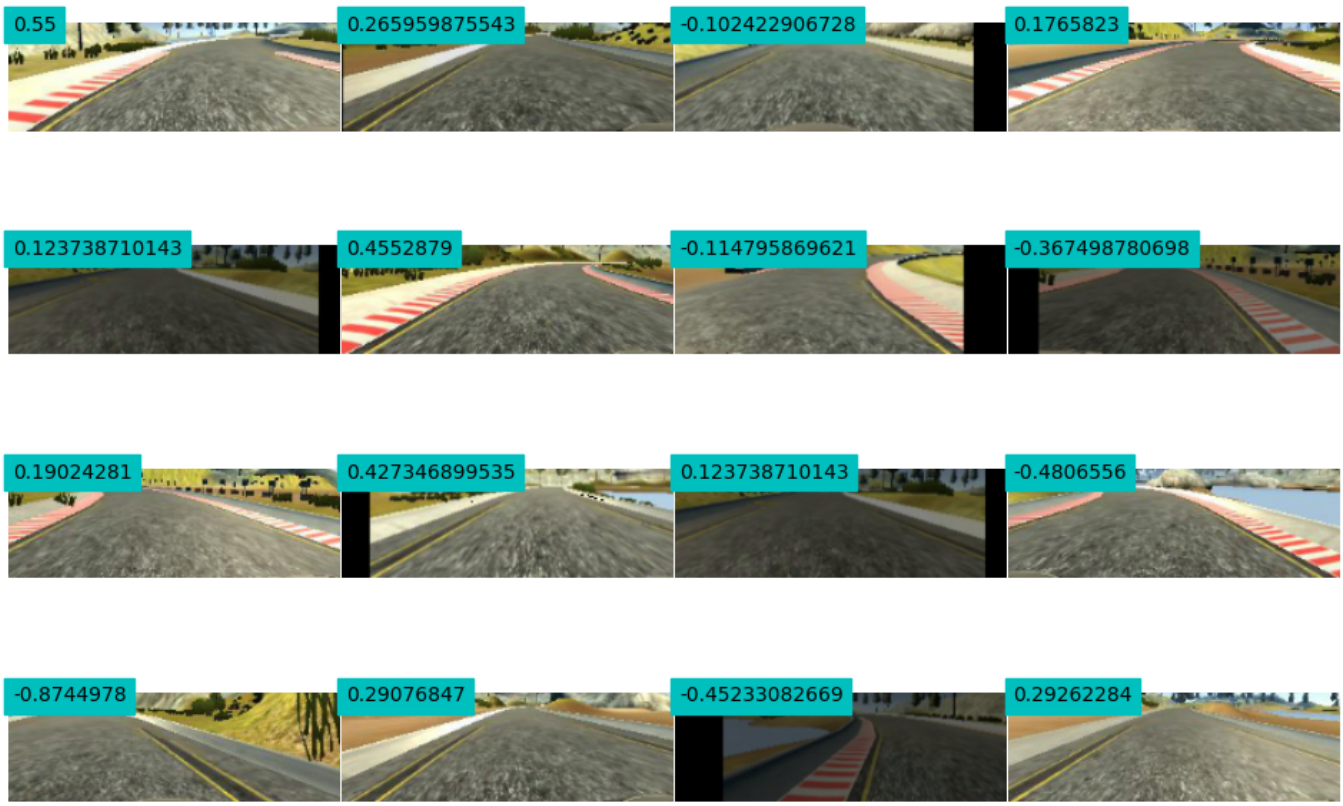


Data Augmentation and preprocessing

- The image is normalised to have a range of [-1 1] using a keras lambda layer.
- YUV format is fed into the model.
- Random brigntness augmentation
- Cropping the image
  - The images were cropped so the bonnet of the car, the sky and the tree are removed.
  - The images fed to the network contains only the road with a vantage point.
- A gaussian blur with a 3X3 kernel is applied to the image.
- one of the three cameras is chosen at random for the training set. when the left and right cameras are used a correction of 0.25 is used to correct th esteering angle.
- while augmenting the steering angle the values are ensured to between [-1 1]
- Half the images in a batch are flipped horizontally and the angle multiplied by -1.0
- Images are shifted at random along the x axis.
- Data with throttle valves below .25 is dropped from the dataset as it does not represent driving behaviour.

*****Only the training data is augmented. only unaugmented center camera is used for validation set.***

Augmented Images



Un augmented Images