# Motivation

The motivation for doing this project is to enable a vehicle (JetBot) to navigate autonomously with images captured from a single-lens camera. Enabling the vehicle to be autonomous would help improve road safety and allow disabled personnel to reach their destinations in a safe and timely manner.

# Dataset

Two tracks were used in the collection of data. All data collected consists of 224 x 224 images (input data), left and right motor values (labels). The collection of data is done by driving the bot with a game controller around each track.

Track One:

* White background with black and thin lane lines (3 mm).
* Lane line is one to two pixels wide on the captured image.

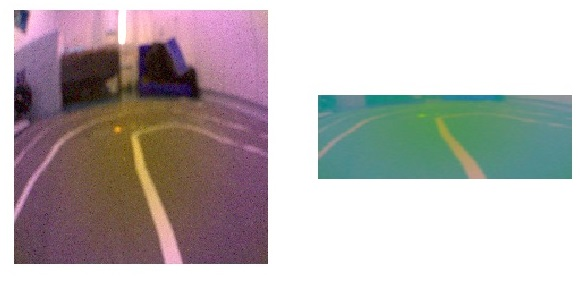
Track Two:

* Black background with thick and white lane lines.

## Preprocessing

The Input data is converted from RGB to YUV. The top part of the image is cropped to make the model focus on the lane. An example is shown below:

Left: Original image, Right: Converted image



The labels are in the format of left and right motor values which are then converted to one-hot encoding (Left, Center, and Right).

Train, test, and validation split are 70%, 15%, and 15% respectively.

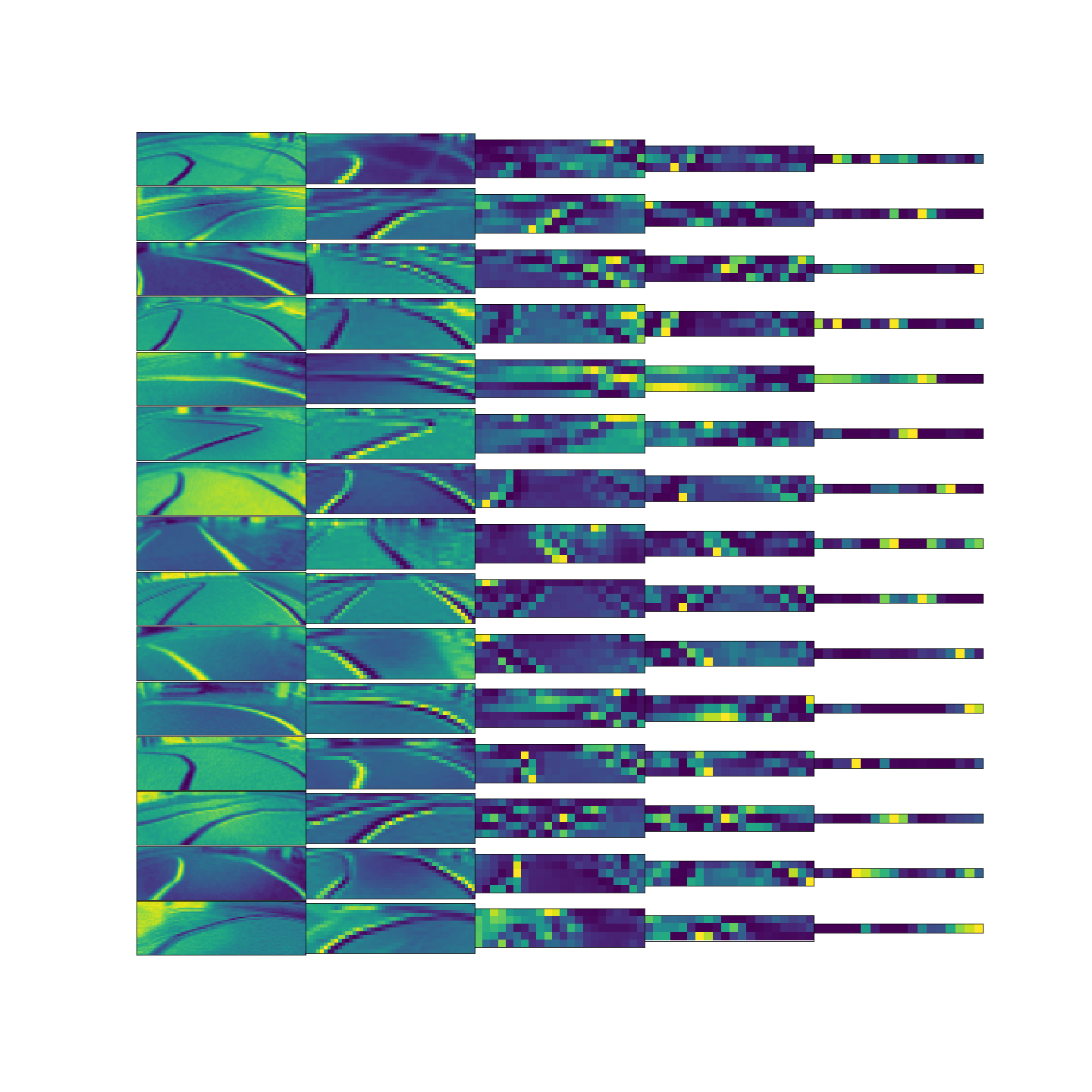
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Train** | **Test** | **Validation** | **Total** |
| **Black Track** | 6500 | 813 | 813 | 8126 |
| **White Track** | 7822 | 1676 | 1675 | 11173 |

# Methods

PilotNet (Nvidia) is used to perform the classification and regression task. Pilotnet is part of Nvidia’s complete software stack for autonomous driving.

The network consists of a series of convolutional layers followed by dense layers which output steering angles depending on the images presented. We have decided to use the PilotNet as a starting point for training.

Below is the feature extracted from PilotNet model. The first two layers are detecting the lane very well.



# 

# Experiments

## Black Track

|  |  |
| --- | --- |
| **Regression Problem** | **Classification Prediction** |
| fig 1a. **Accuracy and Loss for MSE Output** | fig 2a. **Accuracy and Loss for Categorical Output** |
| fig 1b. **R Square Error** | fig 2b. **Classification Report** |
| fig 1c. **Regression Prediction Output** | fig 2c. **Classification Prediction Output** |

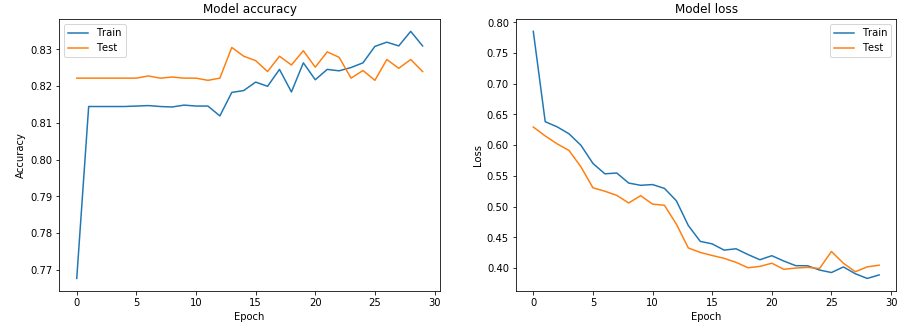
We started by tackling the problem as a regression task with two dense units without activation function as the output layer. The model has stopped training earlier as EarlyStopping was set to monitor validation loss. However, after analyzing the label data and R square error, we realized the data capture is more like a classification problem as the collected motor value only indicates left, right and straight direction.

To treat the problem as a classification task, we re-processed the data labels with one-hot encoding and changed the output layer to three dense units using softmax activation function. The outputs give high accuracy for train and test sets. However, the loss of the test set increases after epoch 20. This phenomenon suggests that the model has overfitted for the training data. We later then use the model to predict against test data set and the F1 score looks promising.

The final classification model is exported as hdf5 (.h5) and manually deployed to JetBot for testing. The real driving result wasn’t as good as the accuracy and other figures displayed above.

## White Track

Results for Training (Classification, White Track)



The model was trained for 30 epochs with the accuracy score plateauing early at one epoch and overfitting at 23 epochs. Further training is unlikely to improve the test accuracy. This could be due to disproportionately more straights than turns in the dataset resulting in high accuracy but unable to correctly predict turns due to the lines being too thin.

# Next Steps

Although the model shows promising figures F1 score on the test data set, the first deployment doesn’t run well as the JetBot is not able to follow the lane accurately.

The team is considering below methods in the next phase of the project:

1. Rethink the problem as steering angle (-180 to +180 degree) and speed to closer to real driving. This would require a new data collection method and also collect new data.
2. Collect more data, apart from train and cross-validation set, will have to segregate additional test set.
3. Consider additional data augmentation to increase the number of collected images.
4. Explore implementing with other architectures:
   1. VGG16
   2. Combine pre-trained segmentation model and combine with PilotNet
   3. CNN-LSTM
   4. Deep Q Learning
5. Deployment of the model into JetBot (Tensorflow RT, TFLite, HDF5).
6. Creation of automated workflow to deploy the latest model into the JetBot.
7. Integration with object detection model to detect traffic light and road sign.