CS577-Assignment 4

Binary Class Classifier

Name : Sourav Yadav **Spring 2020**

AID: A20450418

1.Problem Statement :

Classification of Cats and Dogs. This is three class classification problem which we are going to solve by implementing Conv neural networks.

2.Proposed Solution :

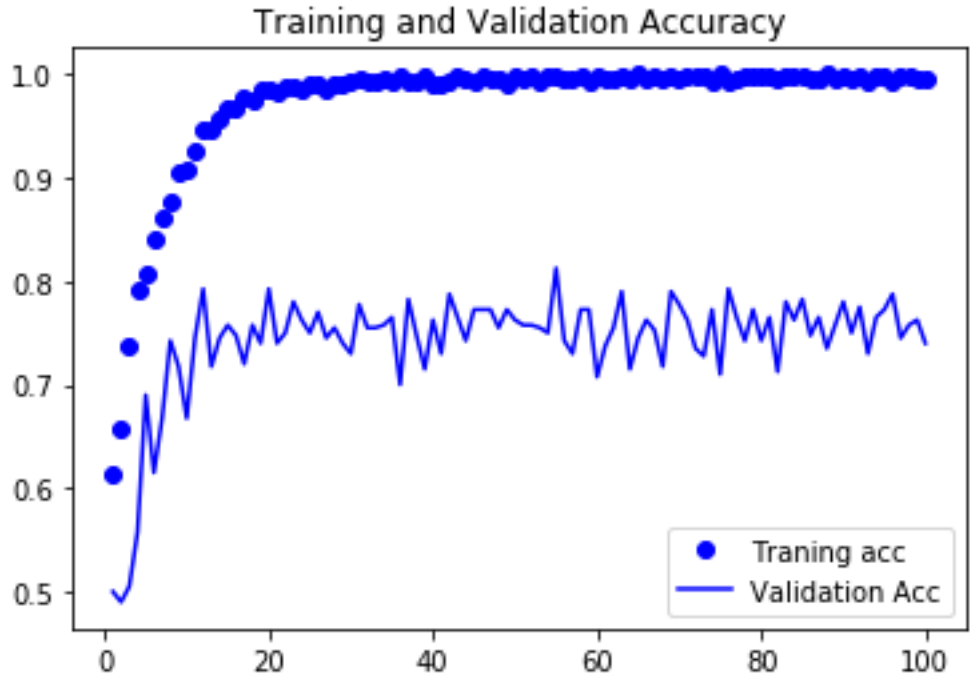
We will design a Conv neural network in Python with Keras and then train the network on train and validation data. Finally we will evaluate performance of the network on test data.

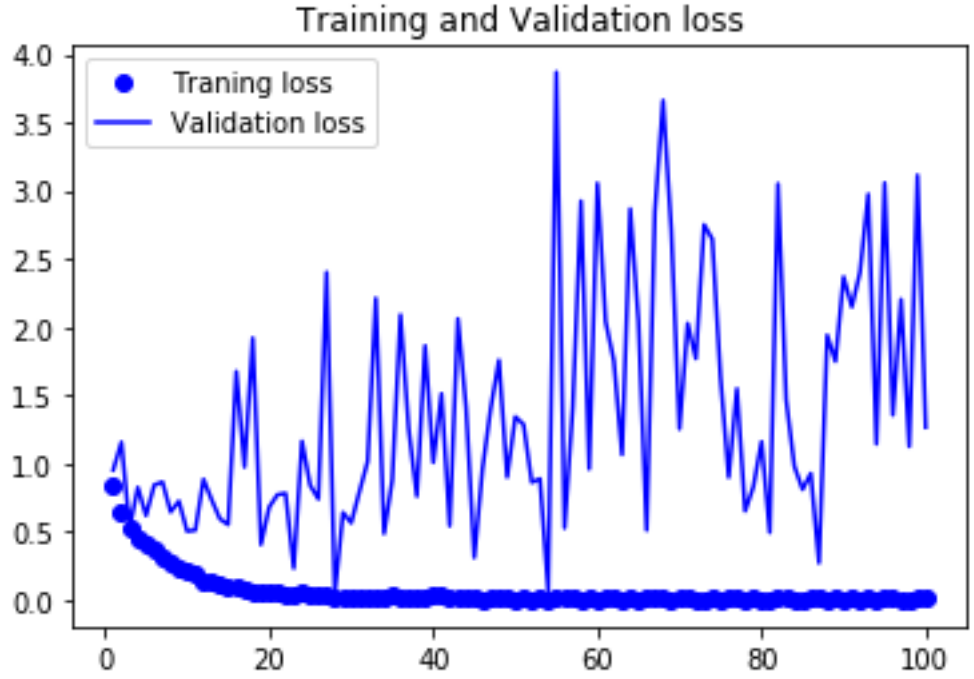
3.Implementation details :

- We first load the data Cats and Dogs Data from the below Link.
- https://www.microsoft.com/en-us/download/details.aspx?id=54765

- Then we are going to One Hot Encode the labels of the dataset.
- We split the Iris data in Train , Test and validation Set using train_test_split() method in 8:2 ratio.
- Then we normalized the features.
- We designed neural network with 4 Convolution Network layers.
- Activation used in the hidden layers is Relu and Activation used in the output node is Sigmoid.
- We used various Loss Function and Optimizer to evaluate the network that are discussed in the results section.
- Now we train our network with train data and observed its performance . Results are discussed in below section.

4. Results and discussion:



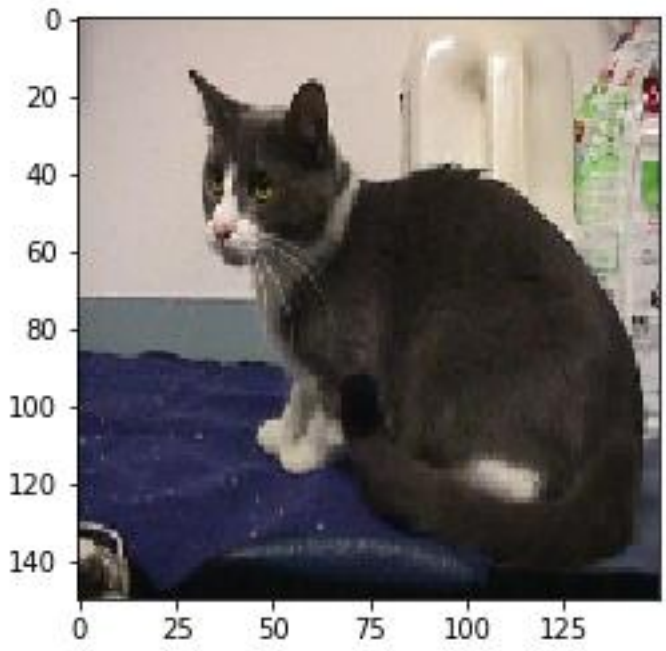


After Epochs =20 the validation results converge . hence we choose 20 as our epoch.

Evaluation on Test data:

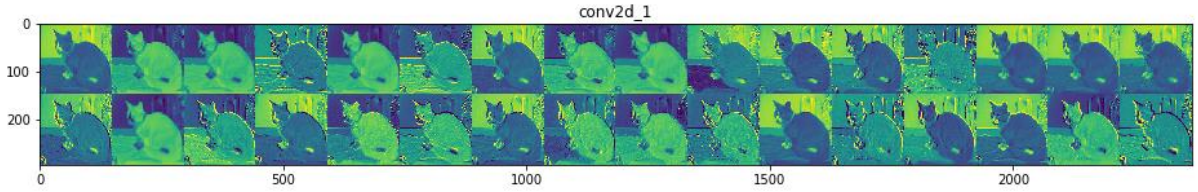
```
Loss: 1.830385804176330 6
Accuracy: 0.7375000119209 29
```

Visualising Convolution layer 1 and layer 2 :

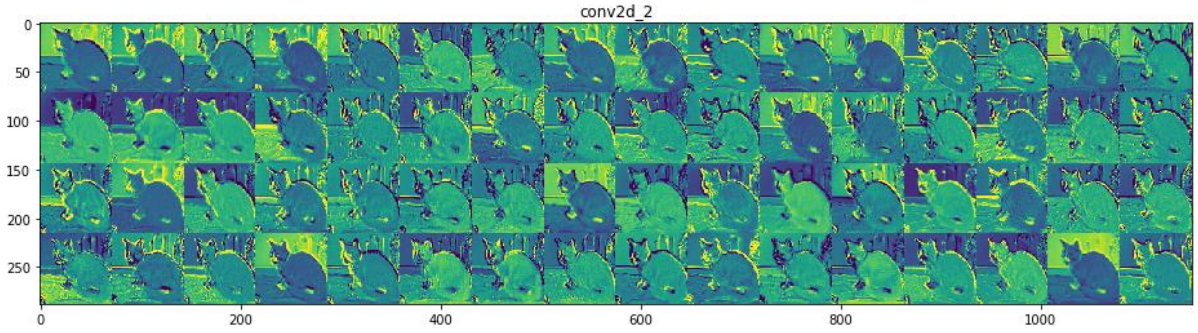


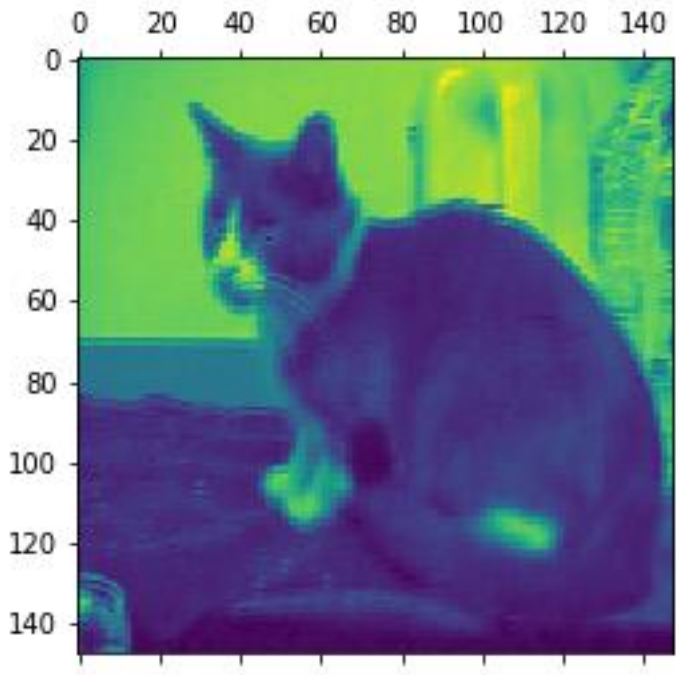
Original Image

Convolution Layer 1:

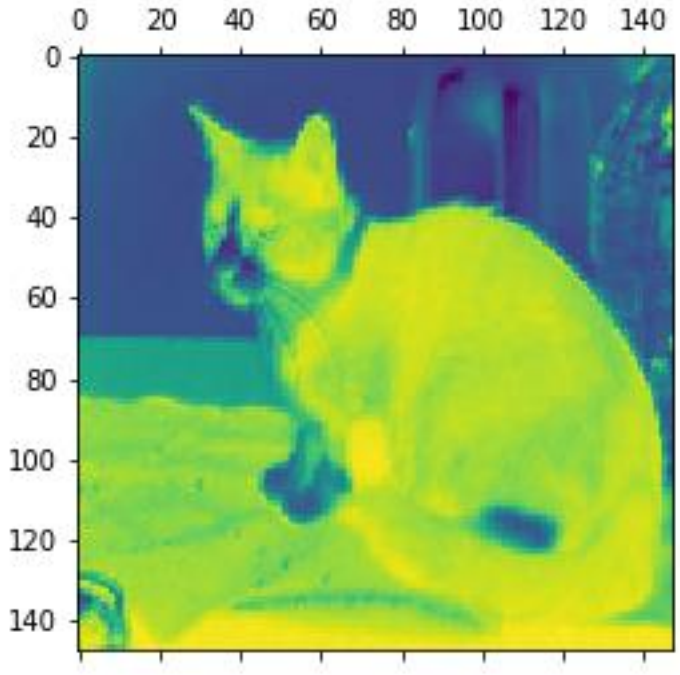


Convolution Layer 1:

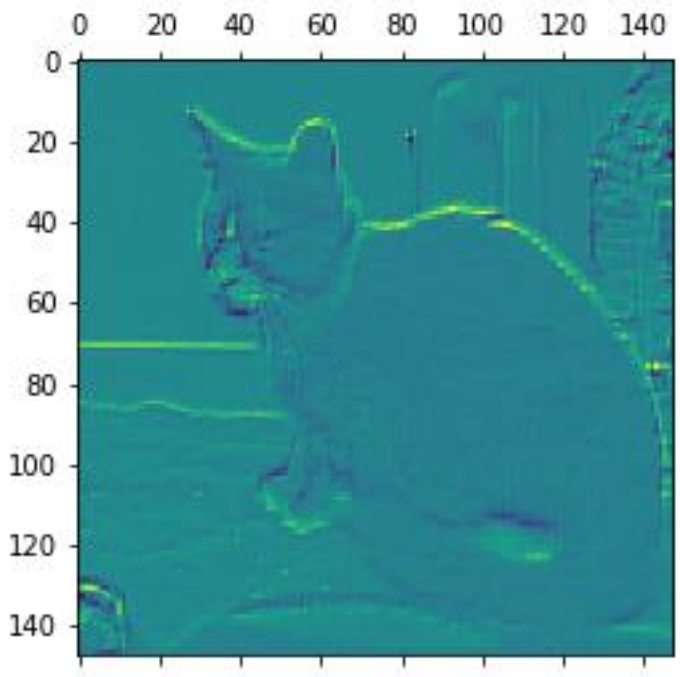




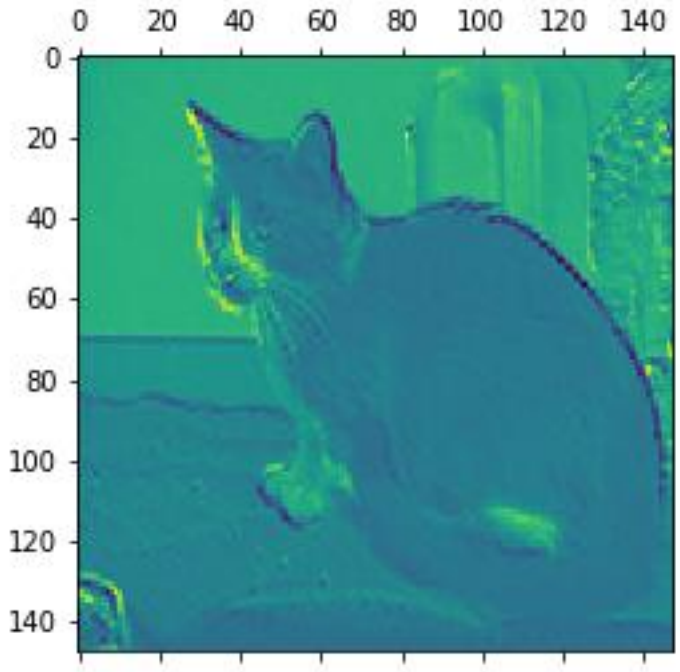
Channel 5 : Detects brighter parts of the image



Channel 4 : Detecting Dark Colours



Channel 3 : Detecting Edges

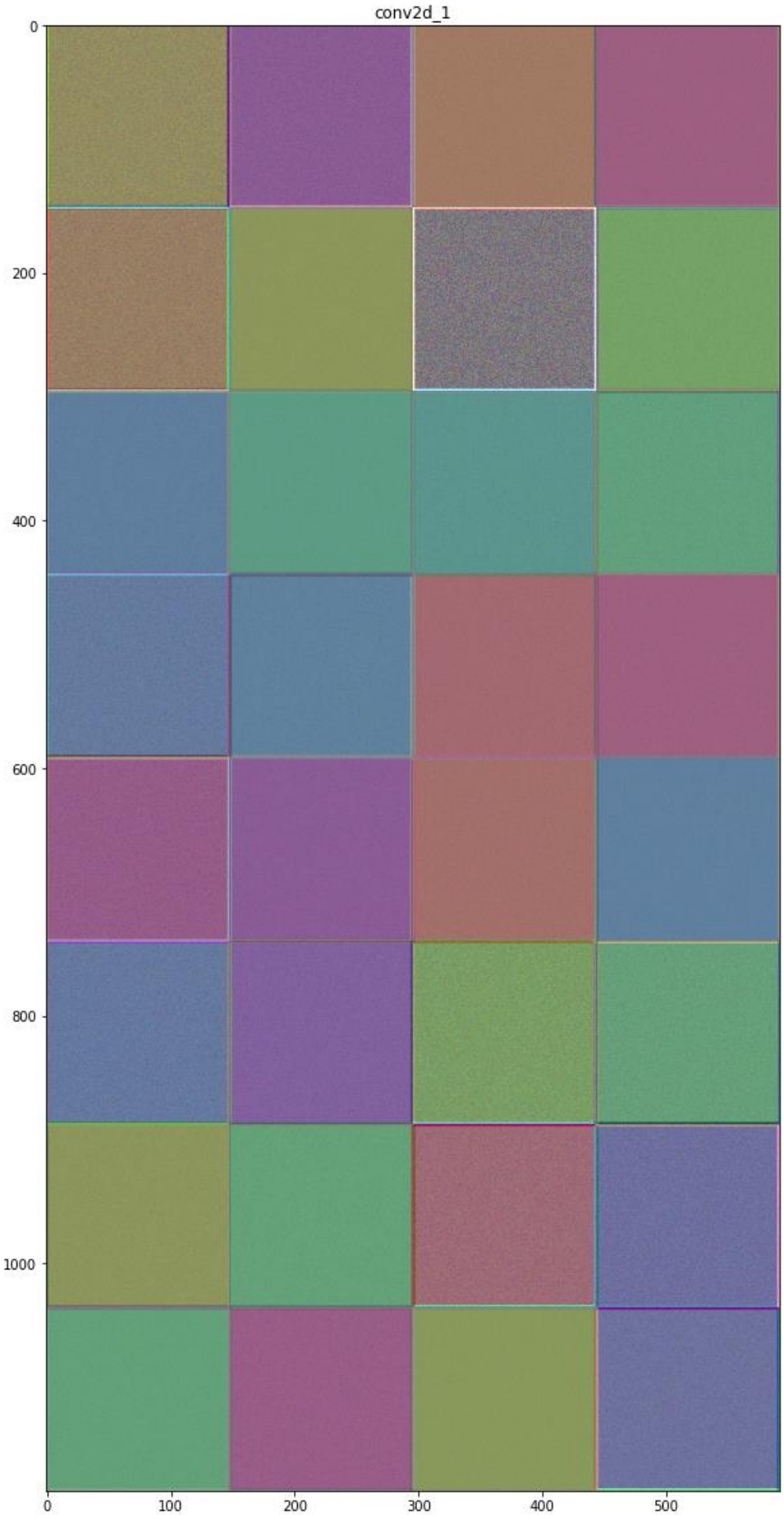


Channel 8 : Detects vertical edges of the image

From the above visualization of 2 layers we can see that different filters detects different features in the image.

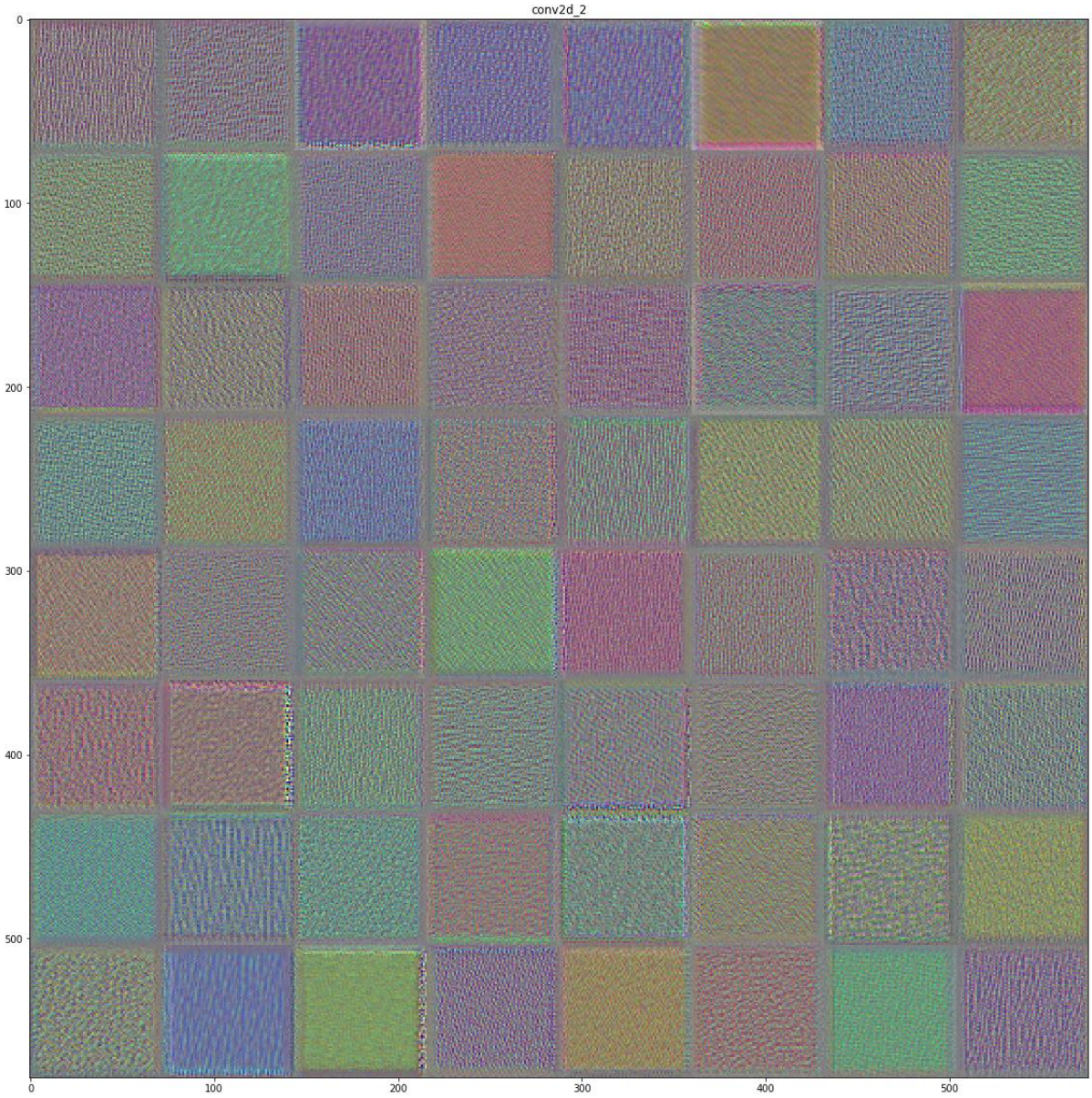
Visualization of Learned Filters :

Convolution Layer 1


conv2d_1
0
200
400
600
800
1000
0
100
200
300
400
500

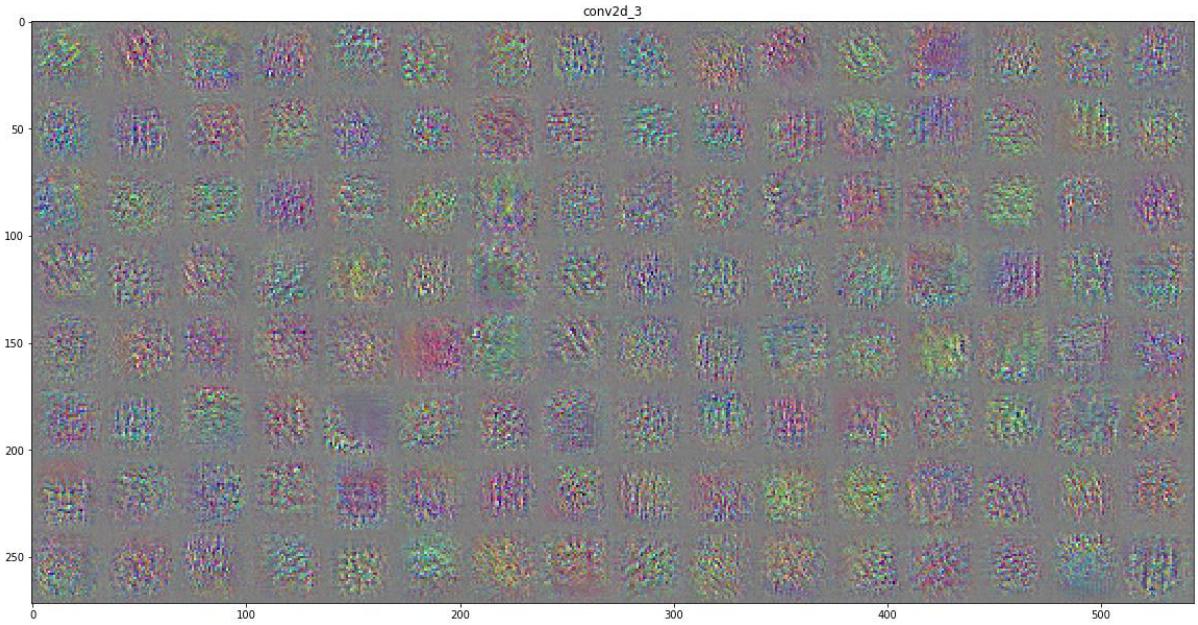
Detecting Colors

Convolution Layer 2



Detecting Patterns in the image

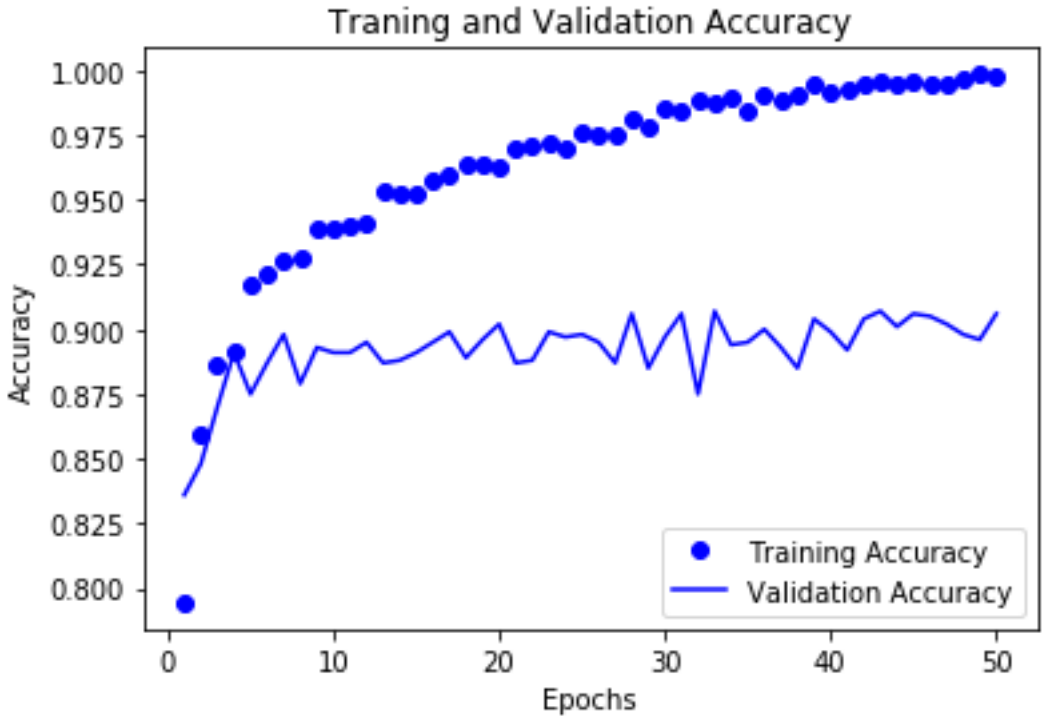
Convolution Layer 3

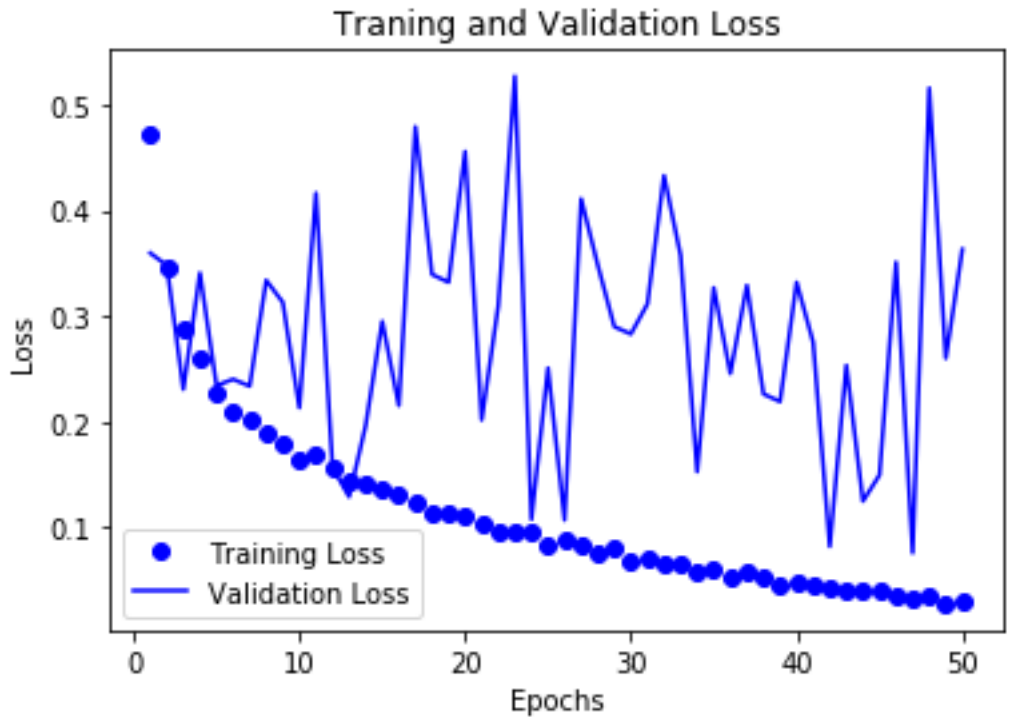


Detecting Higher level patterns

From the above patterns we see various filters detect various patterns as template matching criteria
.

Part G: Replace VGG16 with frozen base (Before Tuning)

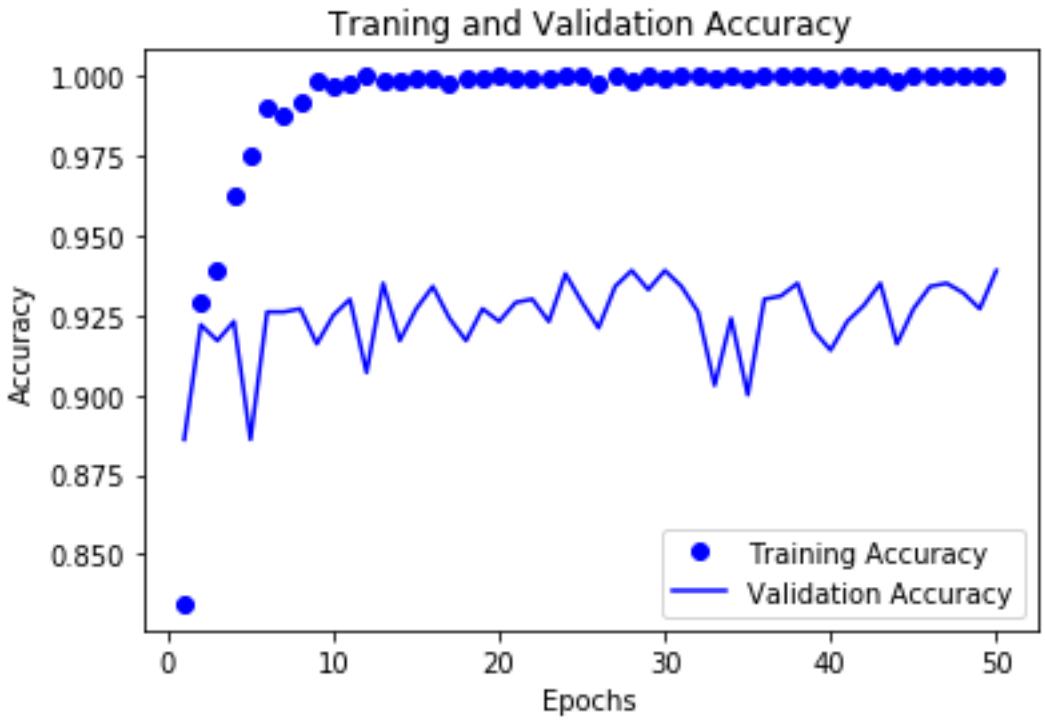


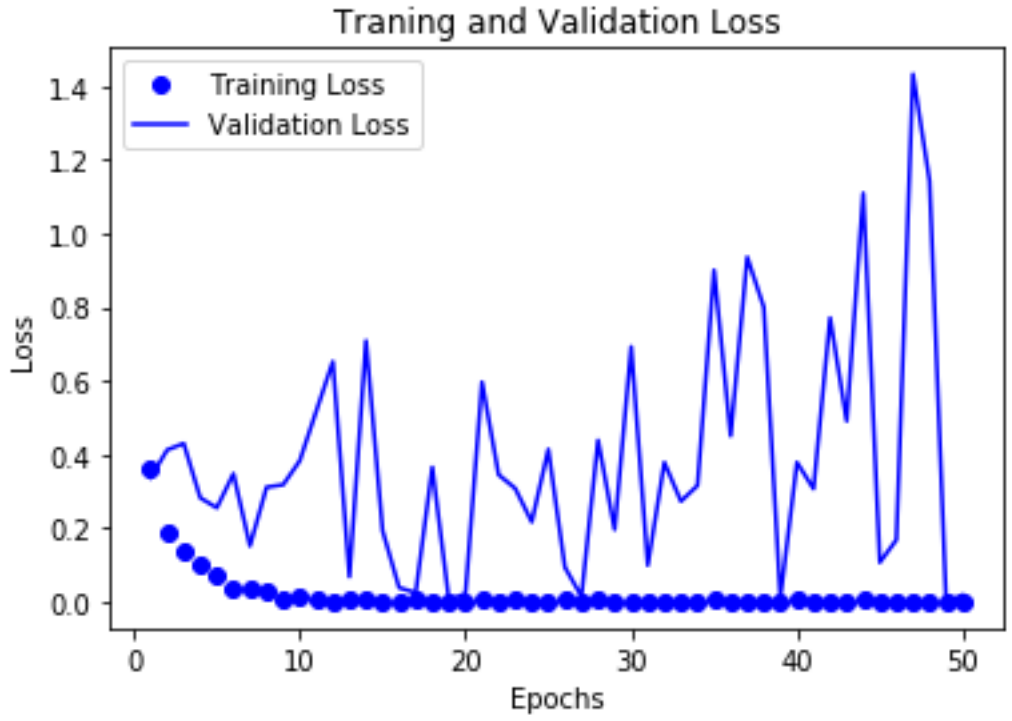


Epochs =10

```
Test Results
Loss : 0.5389158725738525
Accuracy : 0.887499988079071
```

Part F : Replace VGG16 with unfrozen base (After Tuning)





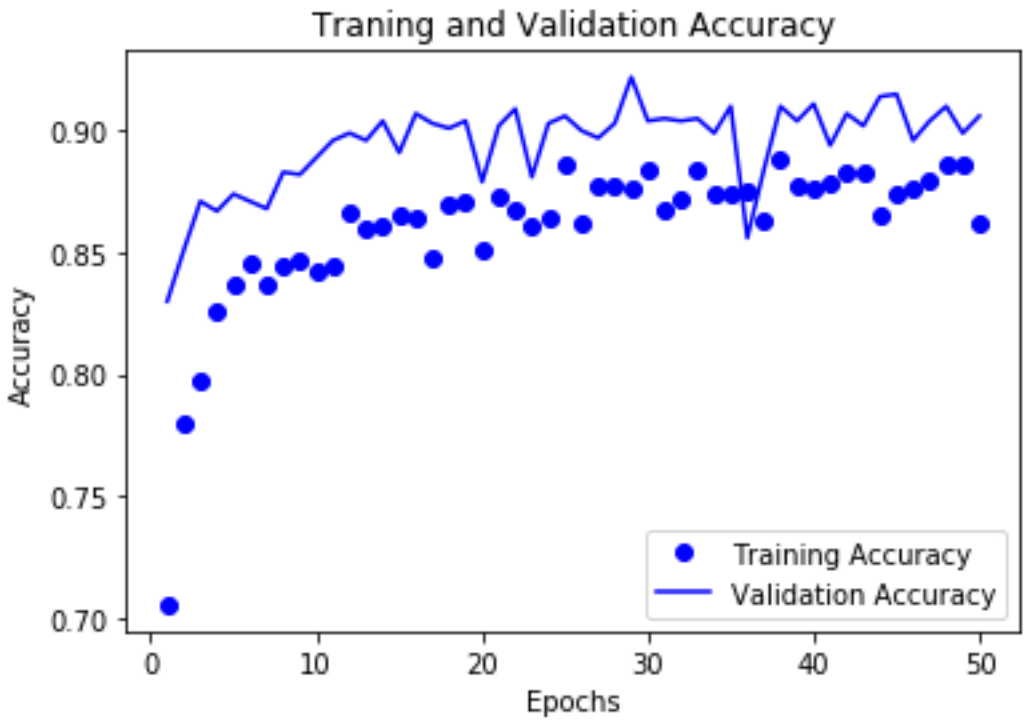
Epoch =10

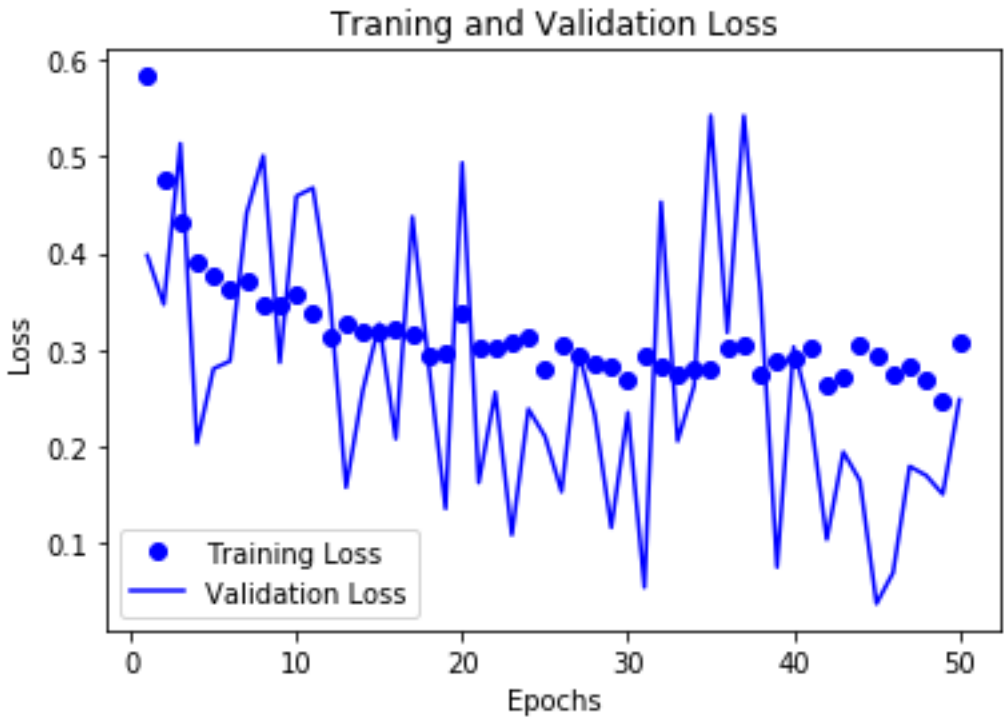
Evaluation on Test Data:

```
Loss : 0.3597032129764557
Accuracy : 0.918749988079071
```

We can see that after unfreezing the base , we get higher accuracy which results due to fine tuning of the base of the pre-trained network with dense layers.

Data Augmentation : After Training with Frozen Convolution Base





From the above graph we can see that Validation results improved after data augmentation.

Evaluation ON Test Data :

```
Test Results
Loss : 0.3437619805335998 5
Accuracy : 0.893750011920929
```