cs577 Assignment 4: Program Report

Pushpendra Singh Chauhan

Department of Computer Science

Illinois Institute of Technology

April 27, 2021

Abstract

This is a report for programming question in cs577 Assignment 4. It will contain the description of the problem I am trying to solve, the algorithms I employed to solve the problem, implementation details like program design issues and the results obtained. The results obtained will be analyzed for correctness. The performance of the algorithm will be evaluated and discussed.

1 Problem Statement:

- BINARY CLASSIFICATION:- (a). Download the Kaggle's Cats and Dogs dataset and select a subset of 2000 dogs and 2000 cat images to be used for training, validation and testing. The use of subsets is intended to simulate the condition of limited data. (b). Define a training, validation and testing data generators. (c). Build a convolutional neural network using several convolution, pooling and normalization layers, followed by one or more dense layers. (d). Evaluate performance and tune hyperparameters as needed. (e). Visualize the activation of some of the convolution layers and draw a conclusion. (f). Visualize the filters learned in training by finding the input that will maximize their response and draw a conclusion. (g). Replace your convolution layers with the pre-trained convolution base of VGG16.

Train with the convolution base frozen and evaluate the results. (h). Once the previous training step is complete fine tune the convolution base: unfreeze the convolution base and continue to train. Evaluate the results you get and draw a conclusion. (i). Modify the data generator to perform data augmentation. Retrain with frozen convolution base and evaluate the results.

- MULTICLASS CLASSIFICATION :- (a). Download the CIFAR-10 and load the pickled data into your program. (b). Build a basic convolutional neural network with several convolution, pooling and normalization layers. Flatten the output of the convolution layers and pass it to a single dense layer that will produce the output using softmax activation. (c). Test the performance of the model you built and tune hyper parameters as needed. (d). Add one or two inception blocks and test performance. (e). Remove the inception blocks and add one or two residual blocks instead. Test performance and compare to previous results.

2 Proposed Solution:

- Used 'to_categorical()' function of keras library for encoding the known class labels in multi-class classification problem of assignment.
- Used tensorflow and keras library for designing a fully connected neural network.
- Used 'matplotlib' library for plotting training and validation loss as a function of epochs, training and validation accuracy as a function of epochs, training and validation error as a function of epochs.
- Used 'cifar10' dataset from keras library which is also available in UCI repository for multi-class classification problem.
- Used Kaggle's 'Cats and Dogs' dataset for binary classification problem. Download the subset of 2000 dogs and 2000 cat images to be used for training, validation and testing from this link: https://drive.google.com/file/d/1G2i5WbATNHNwb4tN38mnzPlJ9L0OQ3cP/view?usp=sharing . After downloading put the 'cats and dogs classification data.rar' file in 'data' folder and extract it here.

- Use pre-trained weights of VGG16 trained on Imagenet dataset for the implementation of transfer learning.
- Used functional API of keras library for implementing the inception blocks and residual blocks.

3 Implementation Details:

- BINARY CLASSIFICATION :- (a). Defined training, validation and testing data generators. (b). Built a convolutional neural network with several convolution, pooling and normalization layers. Also added dropout layer for controlling overfitting. (c). Compile the network and trained the model. (d). Plot the training and validation loss as a function of epochs. Plot the training and validation accuracy as a function of epochs. (e). Evaluating the performance on the test data. (f). Visualize the activations of some of the convolution layers. (g). Visualize the filters learned in training by finding the input that will maximize their response. (h). Replace our convolution layers with the pre-trained convolution base of VGG16. Trained with the convolution base frozen and evaluated the results on test data. (i). After completing the previous step fine-tuned the convolution base: unfreeze the convolution base and continue to train. Evaluated the performance on test data. (j). Modified the data generator to perform the data augmentation. Retrained with frozen convolution base and evaluated the performance on test data.
- MULTI-CLASS CLASSIFICATION :- (a). Load the CIFAR10 data from keras datasets. (b). Split the training data into training and validation subset. (c). Convert the class labels to categorical. (d). Normalize the image data. (e). Designed a convolutional neural network with several convolution, pooling and normalization layers for multi-class classification. (f). Used 'softmax' activation function and 10 neurons in output layer as it is multi-class classification problem. (g). Compiled and trained the network. (h). Plot the

training and validation loss as a function of epochs. (i). Plot the training and validation accuracy as a function of epochs. (j). Evaluated the model on test data. (k). Added one inception block. Again trained the network and evaluated the performance on test data. (l). Removed inception block and added one residual block. Again trained the network and evaluated the performance on test data.

- Folder structure: AS1 (main folder) →

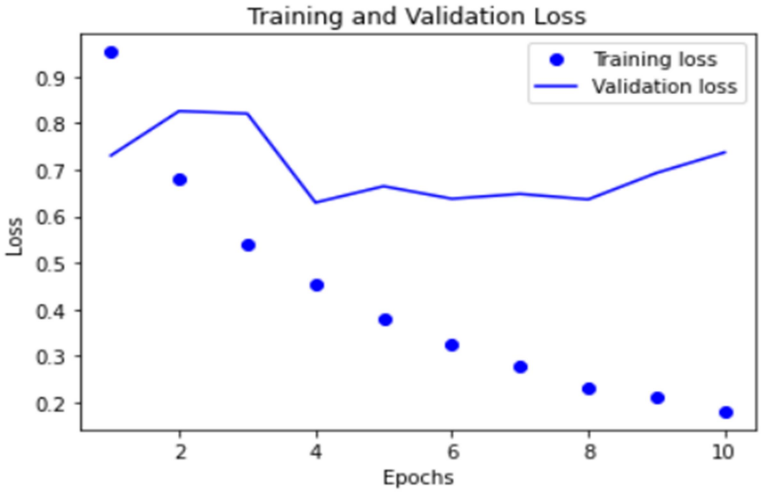
  (1). src (sub-folder) – It contains two source code files named: 'Binary Classification.ipynb' and 'Multiclass Classification.ipynb').

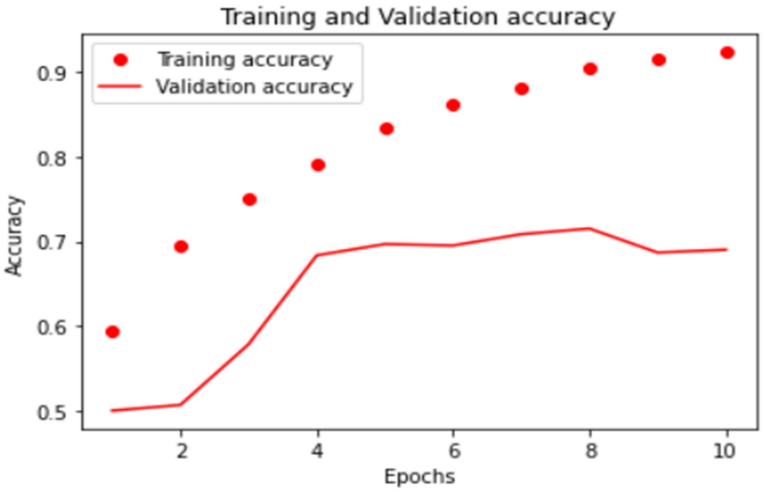
  (2). data (sub-folder) – Download the 'cats and dogs classification data.rar' file from the link given in 'data.txt' file. Put 'cats and dogs classification data.rar' file in this folder and extract this file here. Links to the dataset will be given in 'data.txt' file. 'data.txt' file will reside in this folder.

  (3). doc (sub-folder) – It contains two pdf files ('review questions answers.pdf' and 'program report.pdf').

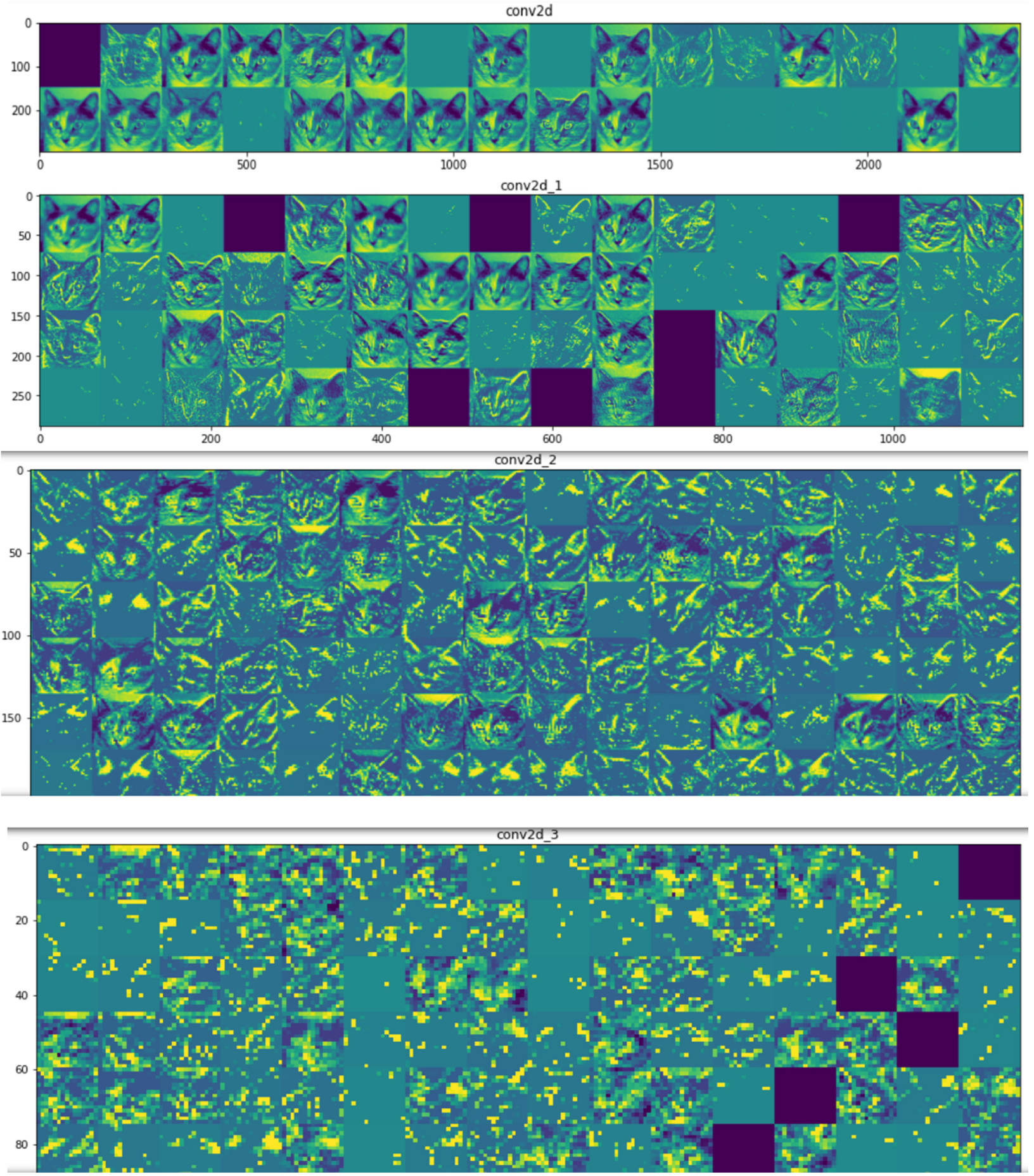
4 Results and discussion:

- Binary Classification:- (a). Plots for our own designed convolutional neural network:





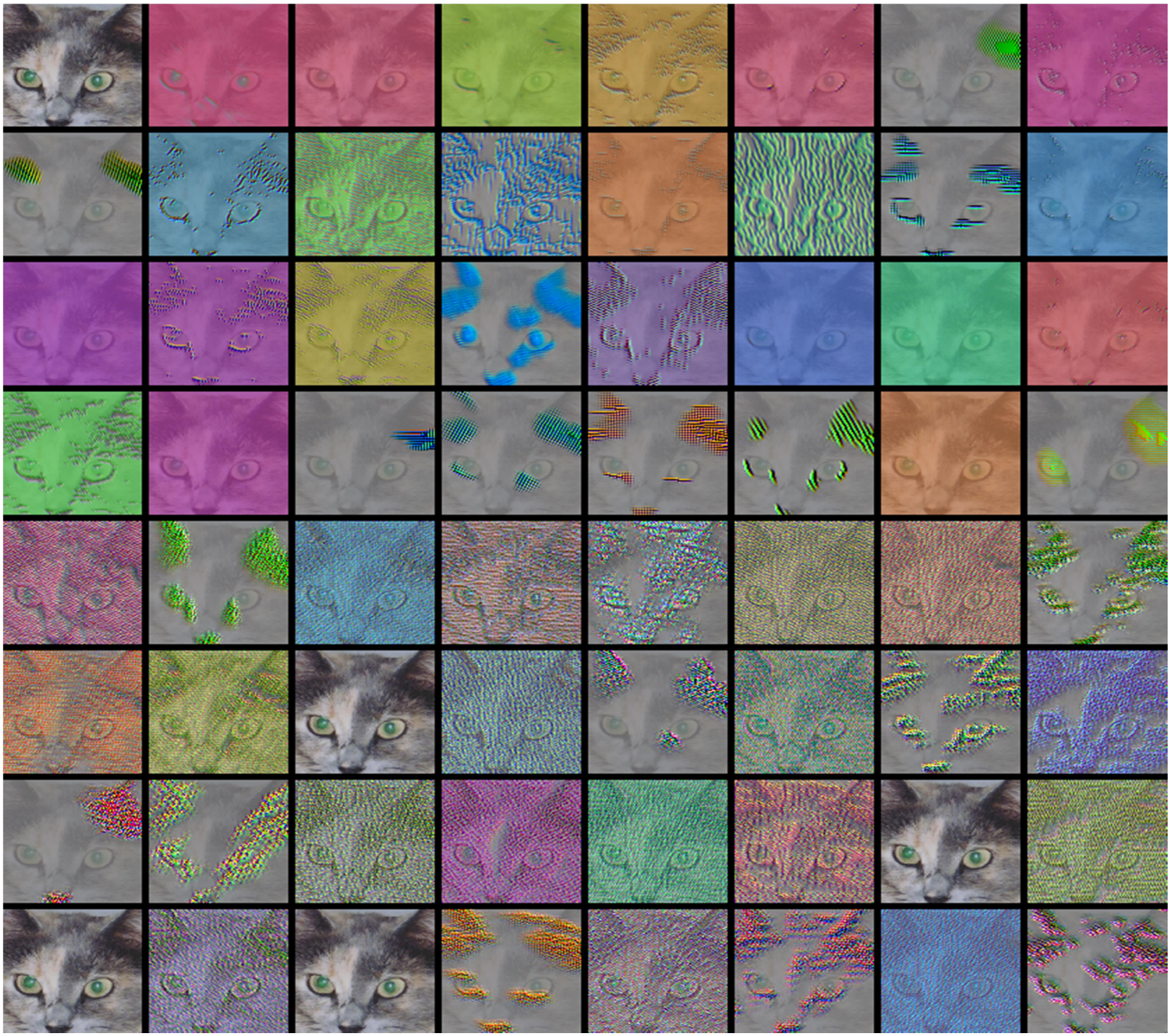
(b). Visualize the activations of some of the convolution layers:



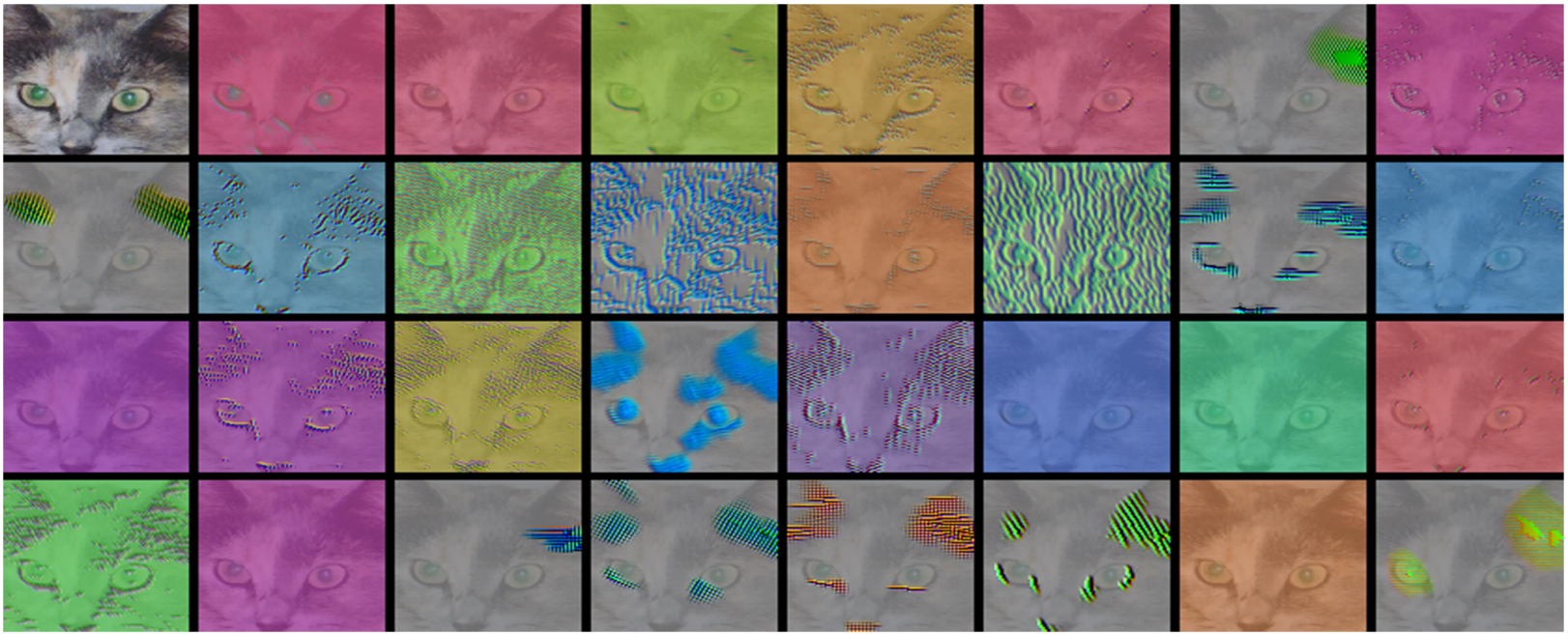
Conclusion:- We can see full shape of cat in first and second convolution layer. There are several filters that are not activated and are left blank in 1st and 2nd convolution layer. In these layers, the activations retain almost all of the information present in the initial picture. But, as we go deeper in the layers, the activations become increasingly abstract and less visually interpretable. They begin to encode higher-level concepts such as lines, borders, corners etc. Higher presentations carry increasingly less information about the visual contents of the image, and increasingly more information related to the class of the image..

(c). Visualize the filters learned in training:

Filters of 2nd conv2d layer (No. of filters = 64)

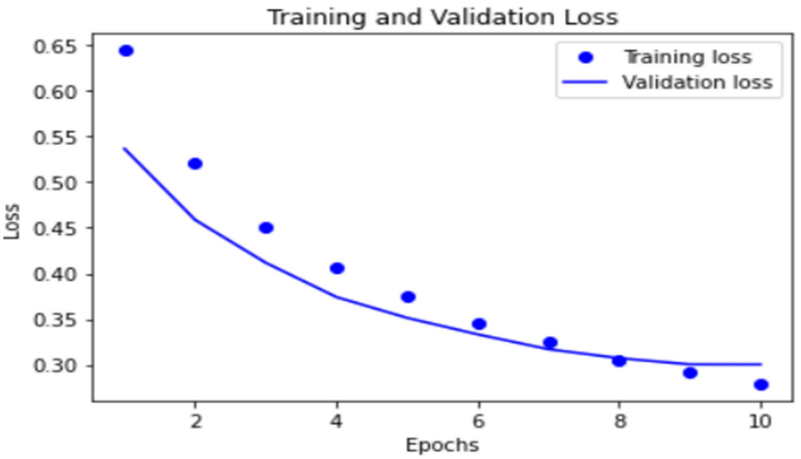


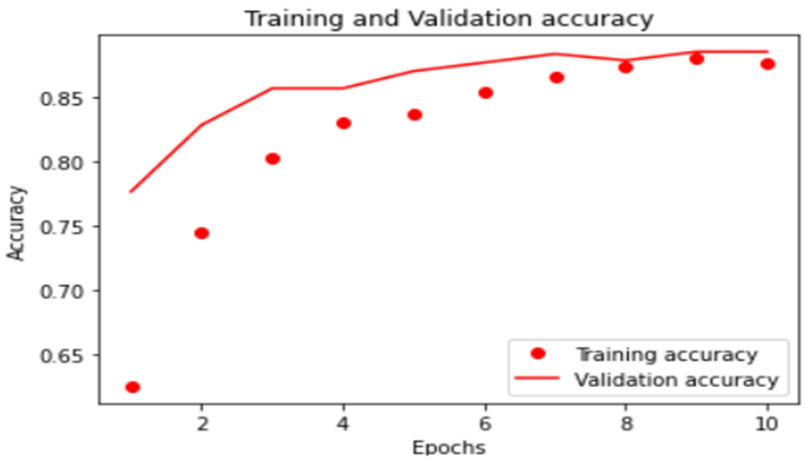
Filters of 1st conv2d layer (No. of filters = 32)



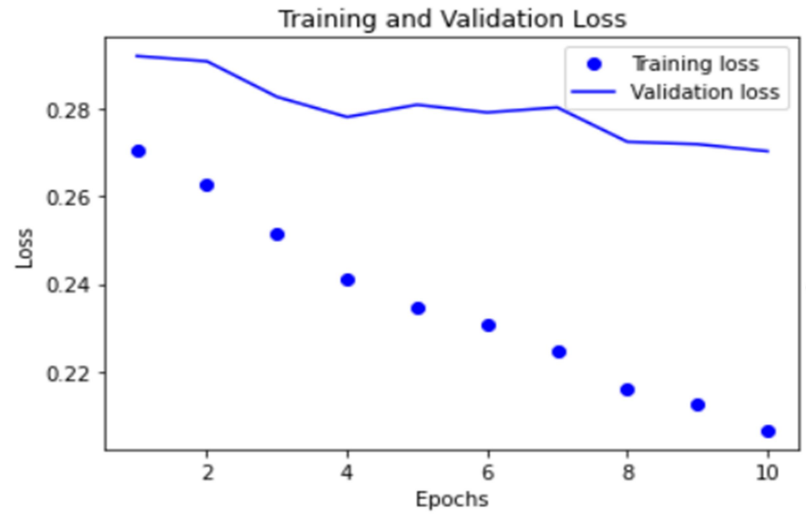
Conclusion:- Here we can clearly see that different filters has different colors, shapes and textures much similar to the input image.

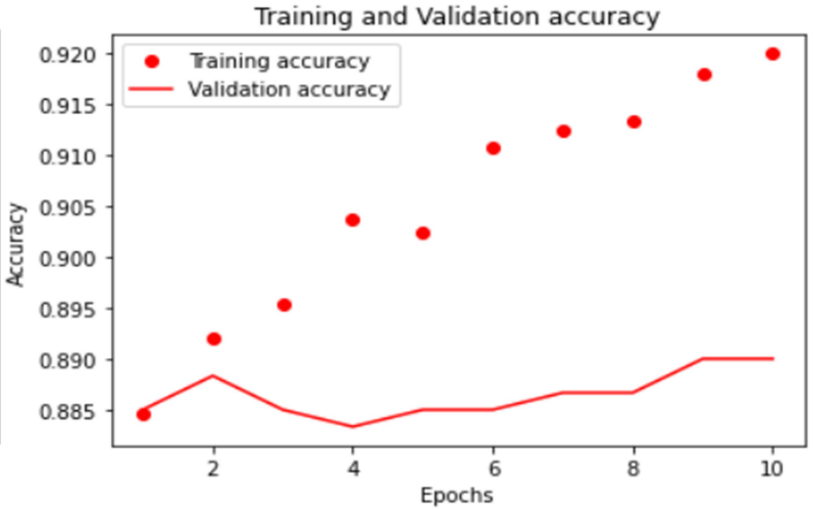
(d). Plots for pre-trained convolution base of VGG16:





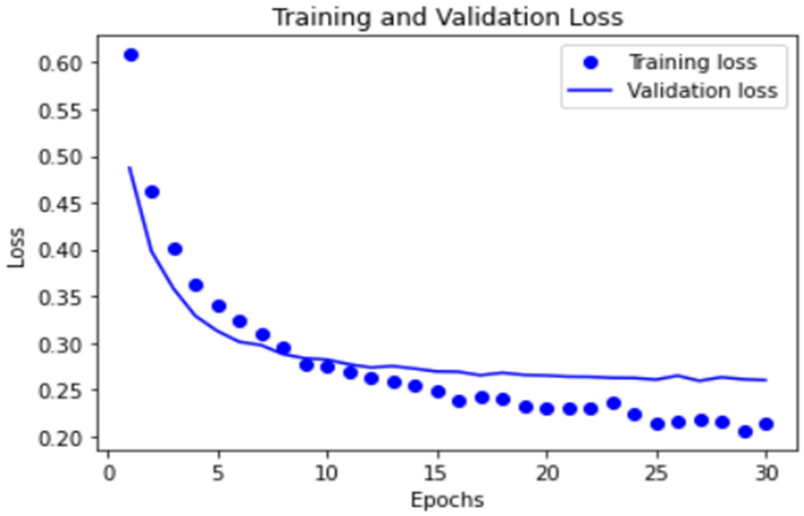
(e). Plots for fine-tuning the convolution base of VGG16:

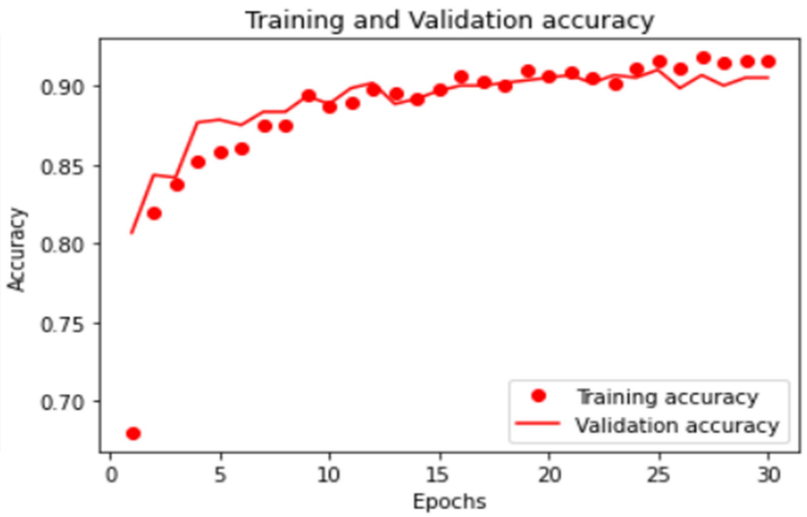




Conclusion:- By unfreezing the 'block5_conv1' layer of convolution base and again training custom network and unfrozen layer, we have increased the test accuracy by approximately 1.9%.

(f). Plots for the model which is being trained using data augmentation:

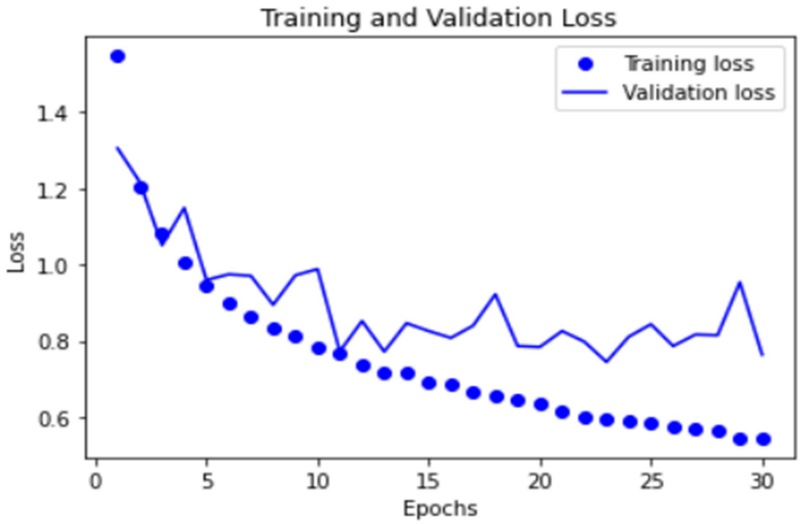


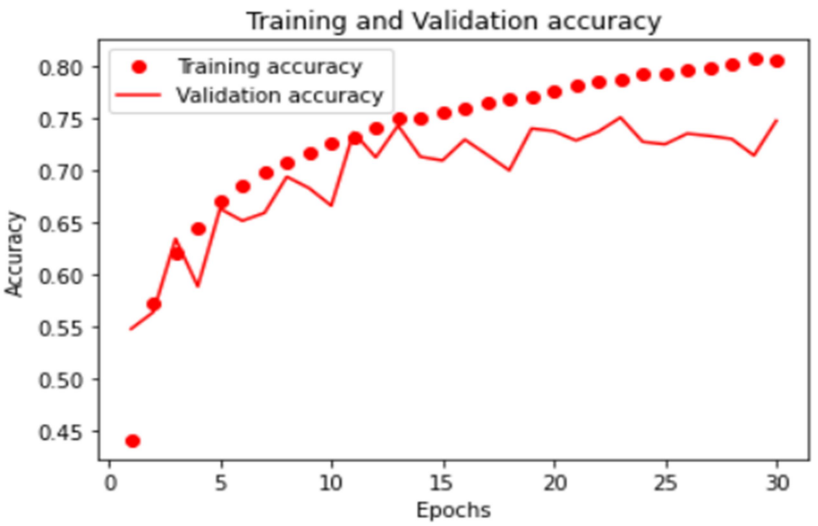


(g). Table for performance (test accuracy/test loss) for all the above models:

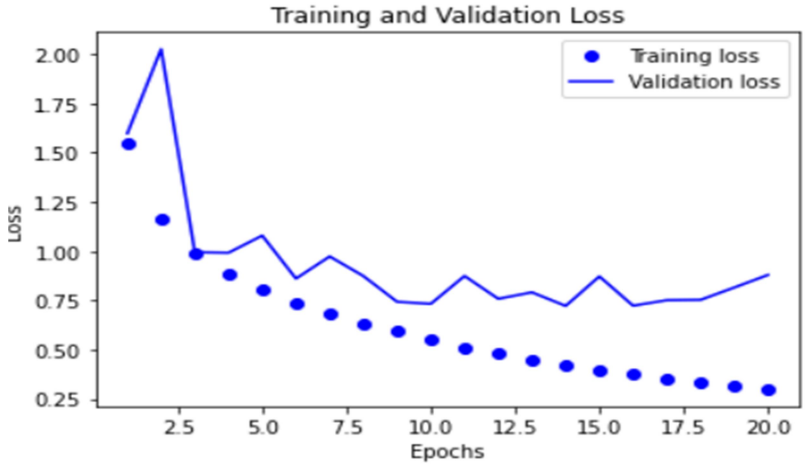
Model Characteristics	Test Loss	Test Accuracy (%)
Self-designed Convolutional Neural Network	0.7589	70.49
Using pre-trained convolution base of VGG16	0.2997	86.10
Fine-tuning the convolution base of VGG16	0.2625	88.09
Modifying data generator to perform data augmentation and using pre-trained VGG16	0.2454	89.30

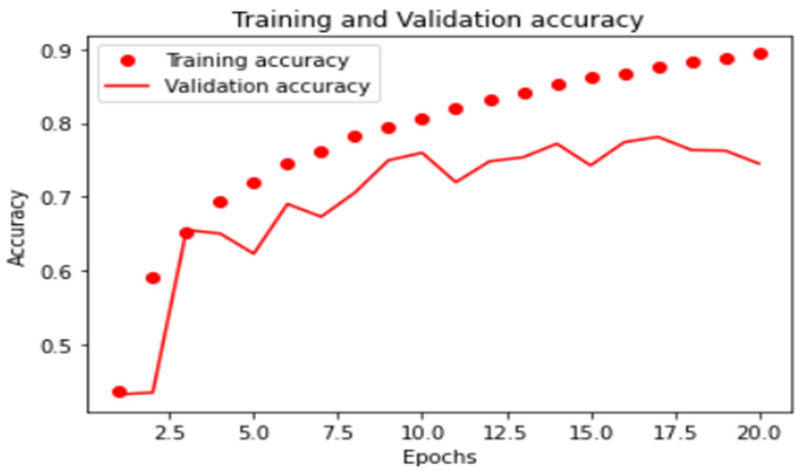
- Multi-class Classification:- (a). Plots for our own designed convolutional neural network:



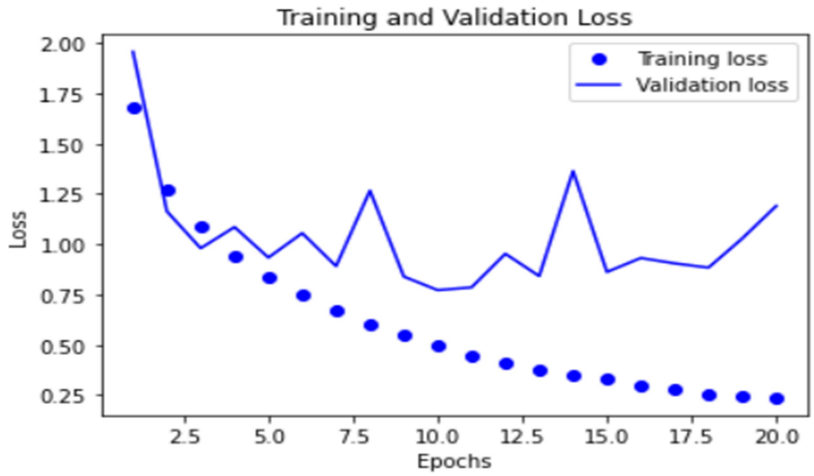


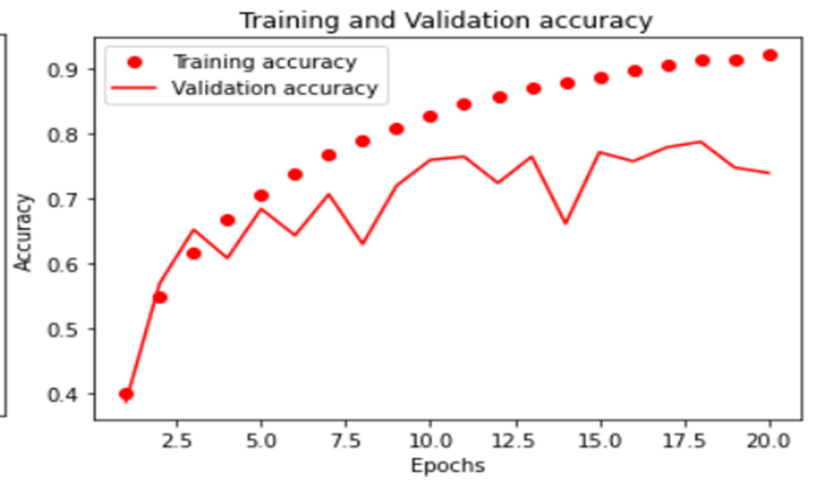
(b). Plots for convolutional neural network with one inception block:





(c). Plots for convolutional neural network with one residual block:





(d). Table for performance (test accuracy/test loss) for all the above models:

Model Characteristics	Test Loss	Test Accuracy (%)
Self-designed Convolutional Neural Network	0.7482	74.419
Convolutional Neural Network with one Inception block	0.7530	77.289
Convolutional Neural Network with one Residual block	0.9311	77.569

5 References:

- https://blog.keras.io/how-convolutional-neural-networks-see-the-world.html
- https://towardsdatascience.com/visualizing-intermediate-activation-in-convolutional-neural-networks-with-keras-260b36d60d0
- https://keras.io/guides/functional_api/