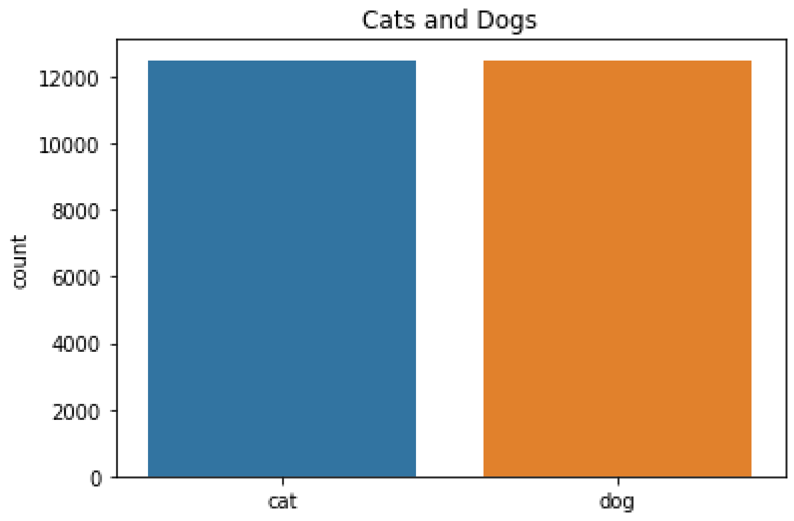
Image classification - Cat and Dog images

1. Overview:

The Image classification of Cats and Dogs contains 25000 images using which we have to train the model. The data is extracted from Kaggle which consist of a training dataset and a testing dataset. The model is trained using the training dataset and then used to predict the images of 12500 cats and dogs which are provided in the testing dataset. Here I have used different networks to train the model. Convolutional Neural Network, ResNet50, ResNet32, VGGN16 & VGGN19. We will be comparing the performance metrics of these models with the validation data and try to predict the result for the testing data. The code is run in Google Colab as gpu is required to train the model with the huge dataset.

2. Dataset Description :

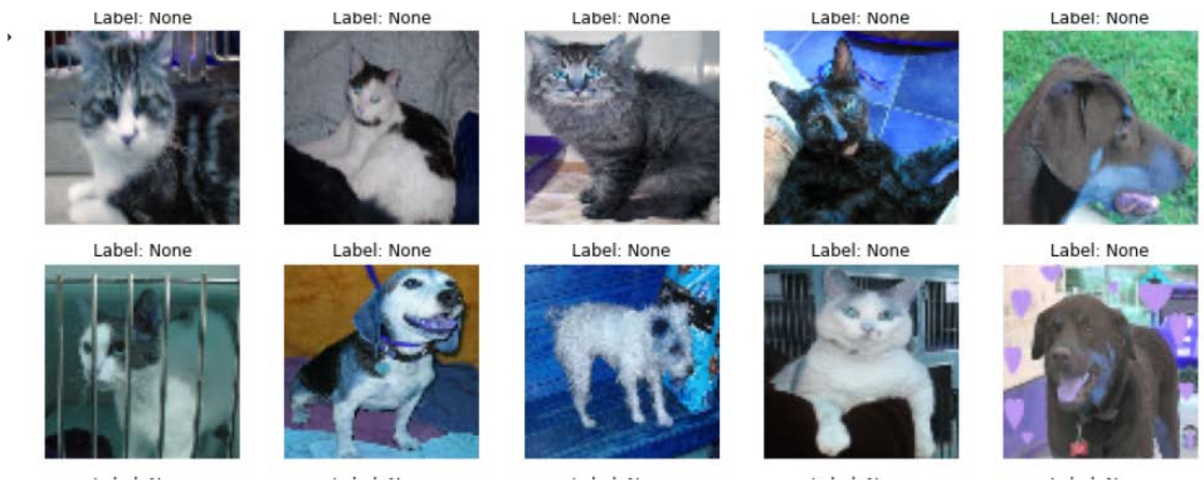
We have plotted the training dataset to see how many images of cats and dogs are available



The train and test data images of 20 dogs and cats are plotted here

Train data images

Test Data images



3. Data Preprocessing:

As the testing data provided here is used to predict the result, we had to use part of the training data for validation purpose. Here we have split the training data into train data set and test data set. Data has been labelled as cat and dog and I have used one hot code representation of [1,0] for cat and [0,1] for dog. As the data is huge it takes a lot of time to load the data. Hence the data has been processed and saved as train_data.npy & test_data.npy for image size 75 and train_data_224.npy & test_data_224.npy for image size 224 .The train data is split with a test_size of 0.5.

4. Libraries Used:

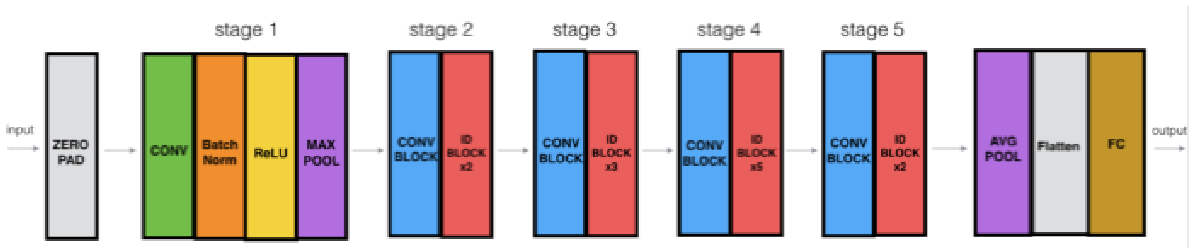
- NumPy- For working with arrays
- Pandas – For reading/writing data
- Matplotlib – to display images
- TensorFlow Keras models
- TensorFlow Keras layers
- OpenCV – Used to handle image operations

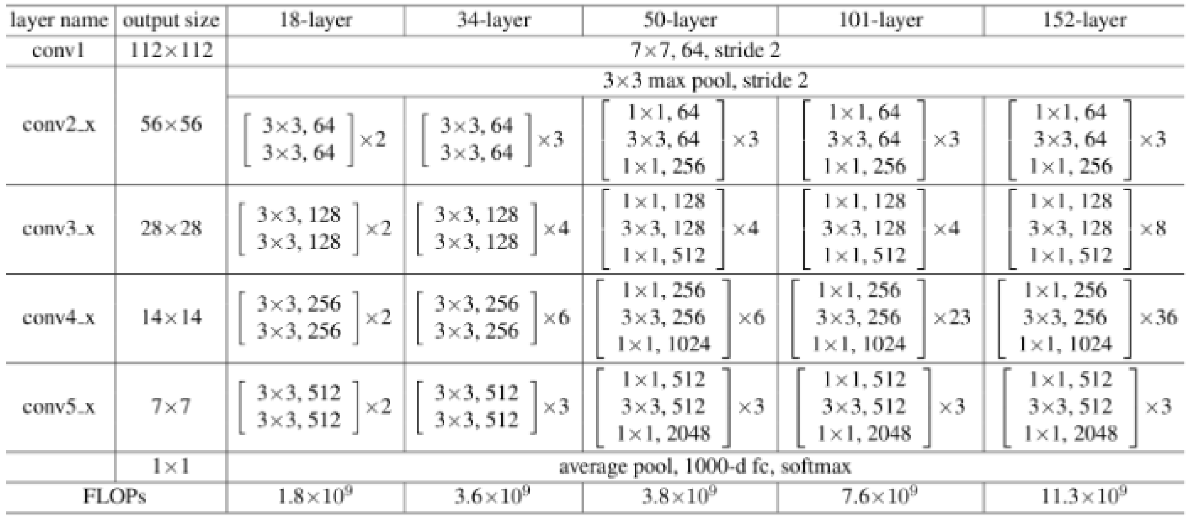
5. Model Architecture

I have tried using Convolutional Neural Network ,ResNet50, ResNet32,VGGN16 & VGGN19. Among these models ResNet50 gave the best accuracy when trained with the image size of 224. The model architecture of ResNet50 is detailed below.

The idea behind using ResNet50 is to make use of the transfer learning technique to overcome the huge computation resource used by deep convolutional neural network. Here, we will reuse the model weights from pre-trained models that were developed for standard computer vision benchmark datasets like ImageNet. The last layer with a categorical class of 1000 will be replaced with our fully connected layer that provides a two categorical class.

The ResNet-50 model consists of 5 stages each with a convolution and Identity block. Each convolution block has 3 convolution layers and each identity block also has 3 convolution layers. The ResNet-50 has over 23 million trainable parameters.

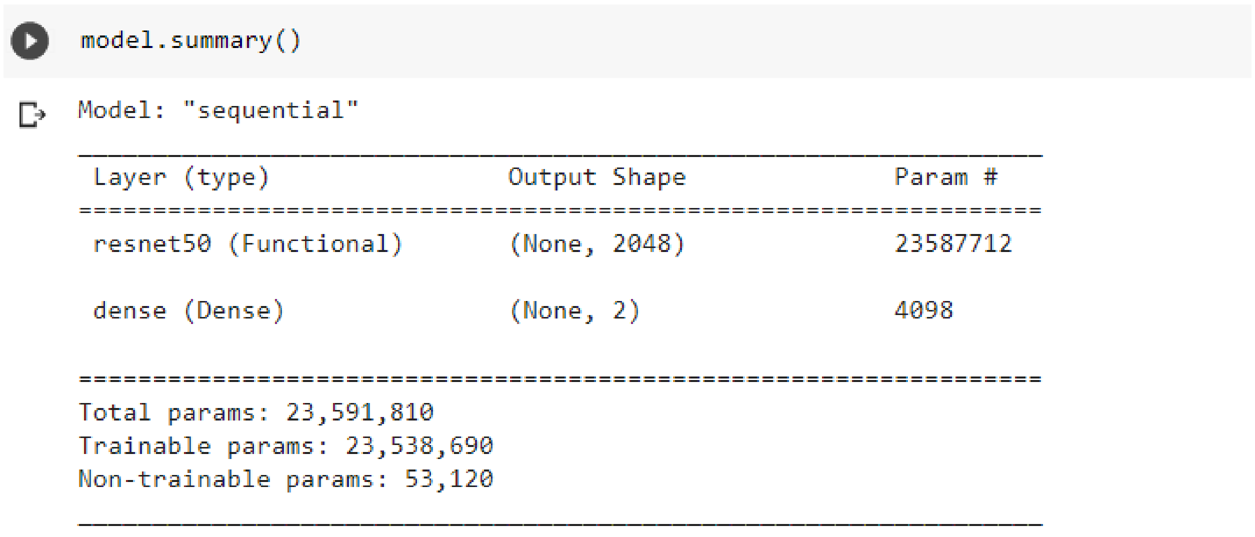




layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

6. Model Compilation and Training

During the model compilation, the optimizer algorithm, loss function and the list of metrics are parameters which are to be taken care of. SGD is used as the optimization algorithm, binary cross entropy is used as the loss function and accuracy is the only metric used. Sequential models fit is used to train the model. Model is trained for 10 epochs .The sigmoid activation is used at the output layer which provide two categories of classes.

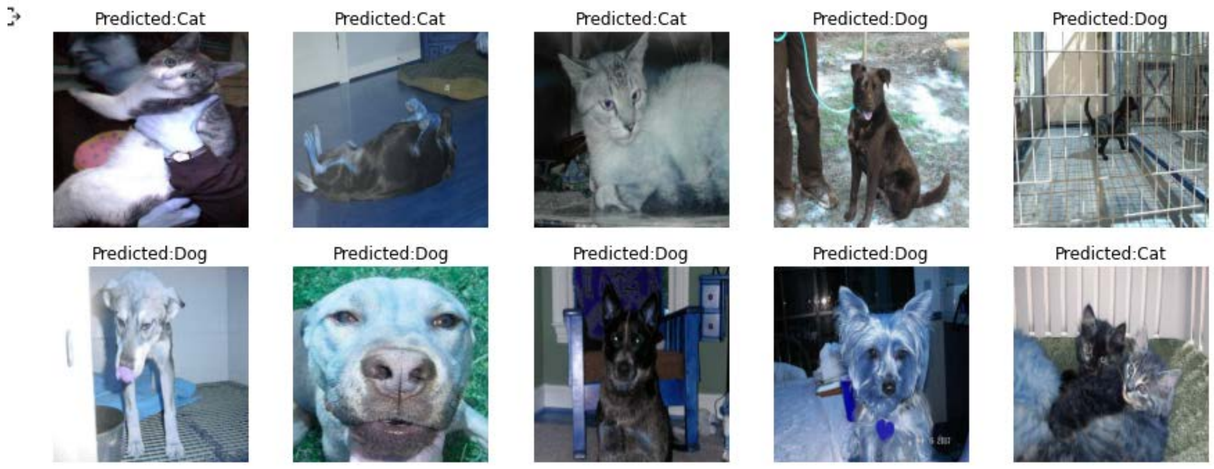


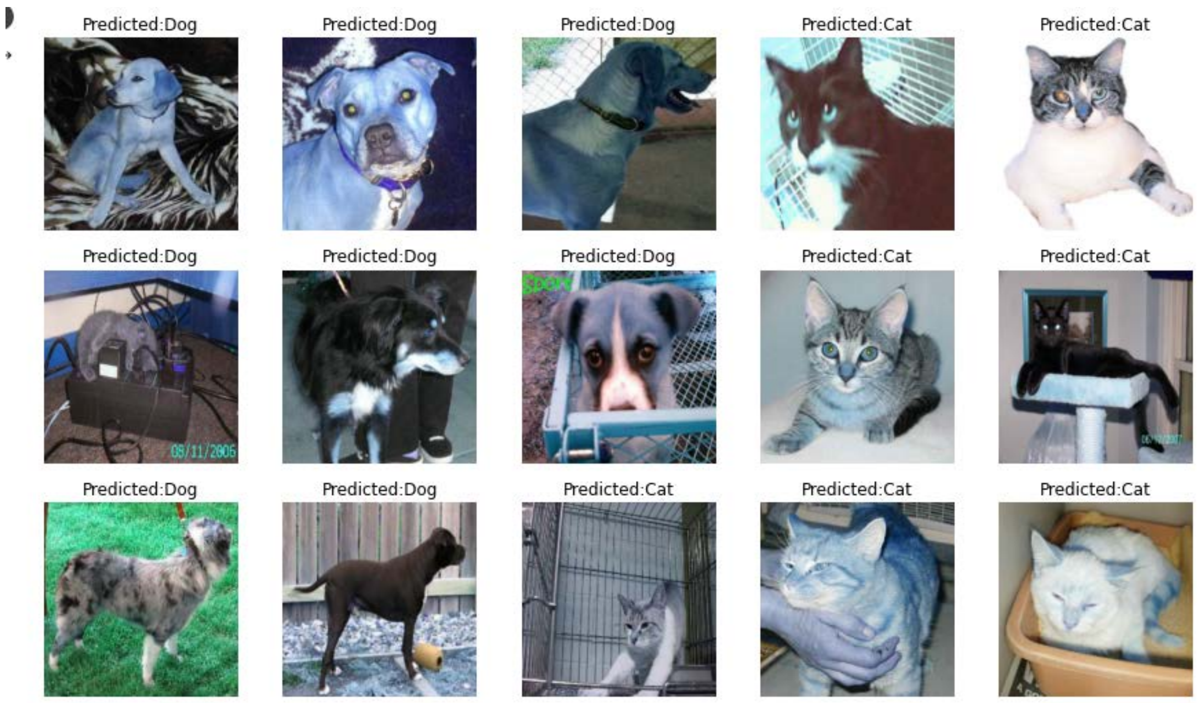
```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 resnet50 (Functional)       (None, 2048)              23587712

 dense (Dense)               (None, 2)                 4098

=================================================================
Total params: 23,591,810
Trainable params: 23,538,690
Non-trainable params: 53,120
_________________________________________________________________
```

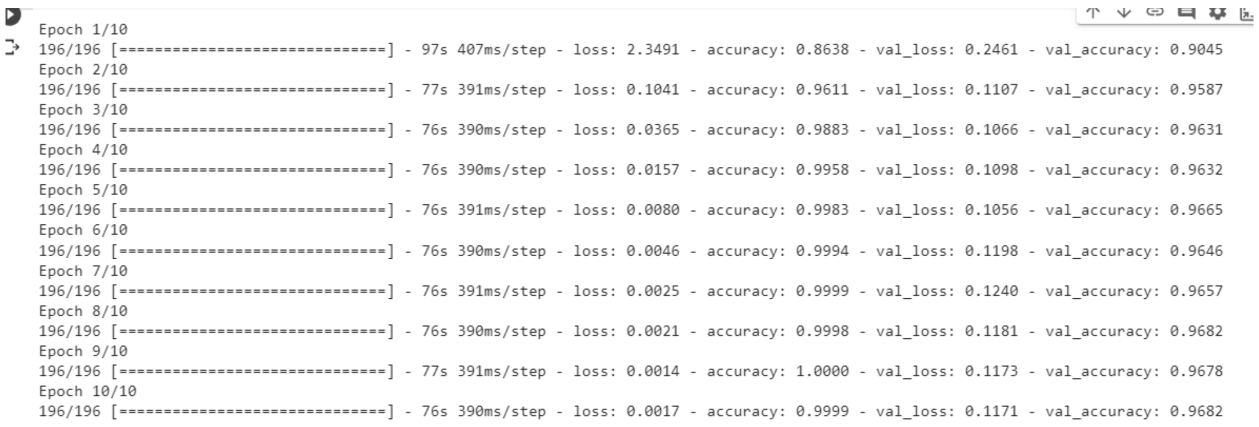
7. Model Evaluation

The model is trained for 10 epochs and the evaluation is done based on the training accuracy, training loss ,validation accuracy & validation loss. The model is able to give a training accuracy of .99 and validation accuracy of .9682.
20 images from the test set are taken and the predicted results gave good accuracy with only two images misclassified.

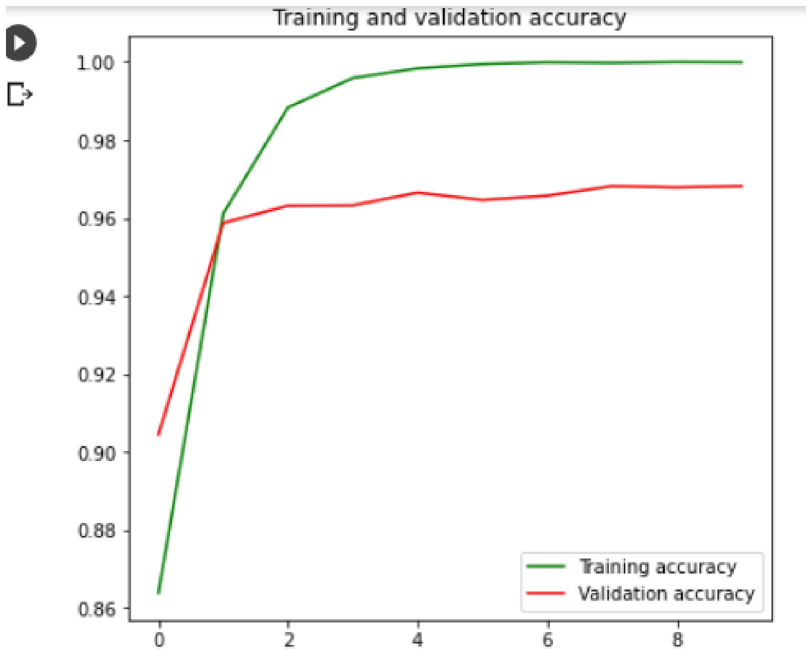




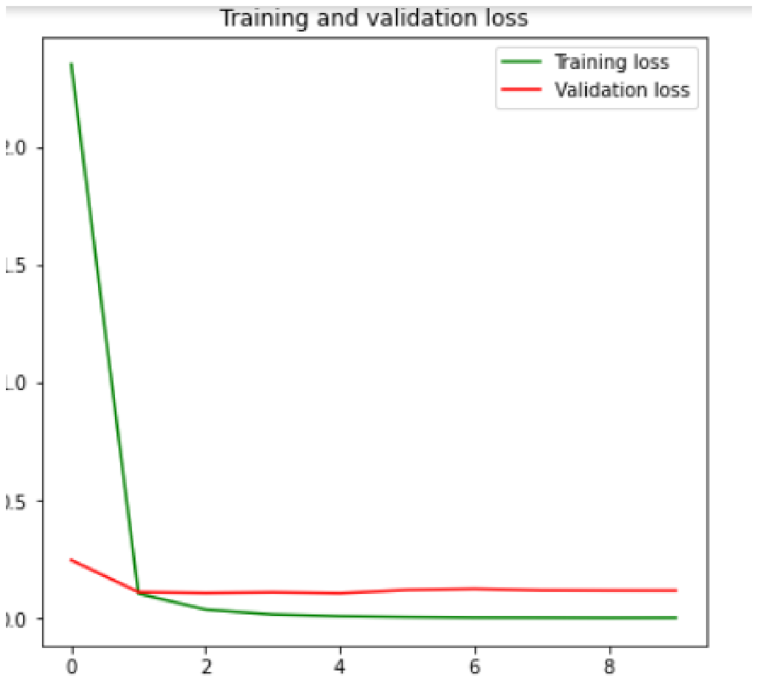
Training result per epoch


```
Epoch 1/10
196/196 [==============================] - 97s 407ms/step - loss: 2.3491 - accuracy: 0.8638 - val_loss: 0.2461 - val_accuracy: 0.9045
Epoch 2/10
196/196 [==============================] - 77s 391ms/step - loss: 0.1041 - accuracy: 0.9611 - val_loss: 0.1107 - val_accuracy: 0.9587
Epoch 3/10
196/196 [==============================] - 76s 390ms/step - loss: 0.0365 - accuracy: 0.9883 - val_loss: 0.1066 - val_accuracy: 0.9631
Epoch 4/10
196/196 [==============================] - 76s 390ms/step - loss: 0.0157 - accuracy: 0.9958 - val_loss: 0.1098 - val_accuracy: 0.9632
Epoch 5/10
196/196 [==============================] - 76s 391ms/step - loss: 0.0080 - accuracy: 0.9983 - val_loss: 0.1056 - val_accuracy: 0.9665
Epoch 6/10
196/196 [==============================] - 76s 390ms/step - loss: 0.0046 - accuracy: 0.9994 - val_loss: 0.1198 - val_accuracy: 0.9646
Epoch 7/10
196/196 [==============================] - 76s 391ms/step - loss: 0.0025 - accuracy: 0.9999 - val_loss: 0.1240 - val_accuracy: 0.9657
Epoch 8/10
196/196 [==============================] - 76s 390ms/step - loss: 0.0021 - accuracy: 0.9998 - val_loss: 0.1181 - val_accuracy: 0.9682
Epoch 9/10
196/196 [==============================] - 77s 391ms/step - loss: 0.0014 - accuracy: 1.0000 - val_loss: 0.1173 - val_accuracy: 0.9678
Epoch 10/10
196/196 [==============================] - 76s 390ms/step - loss: 0.0017 - accuracy: 0.9999 - val_loss: 0.1171 - val_accuracy: 0.9682
```

Training Accuracy VS Validation Accuracy



Training loss VS Validation loss



8. Comparison with other models

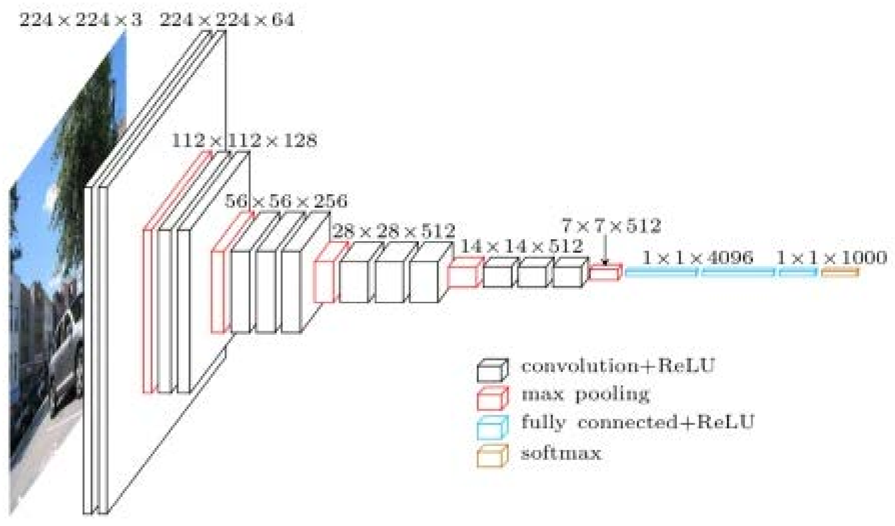
Following models are trained with the dog cat images dataset.

1) Convolution Neural Network with image size of 75 & 224 – This consist of a input layer , conv_2d, max_pool_2d, a dropout layer and a fully connected layer. The activation used in the convolutional layer is Relu. The fully connected layer has softmax which classifies into two classes.The optimizer used is adams,loss function binary cross entropy.The model is trained by using image size of 75 & 224. The accuracy is improved with image size 224.

2) ResNet50 with Image size 75 & 224 **-** ResNet-50 is a convolutional neural network that is 50 layers deep. Convolutional Neural Networks have a major disadvantage **-** Vanishing Gradient Problem. During backpropagation, the value of gradient decreases significantly, thus hardly any change comes to weights. To overcome this, ResNet is used. It make use of skip connection**.** The model is trained by using image size of 75 & 224. The validation accuracy is improved with image size 224.

3) ResNet32 **-** ResNet-32 is a convolution neural network backbone that is based off alternative ResNet networks such as ResNet-34, ResNet-50, and ResNet-101. As its name implies, ResNet-32 has 32 layers. It addresses the problem of vanishing gradient with the identity shortcut connection that skips one or more layers.

VGG Network



$224 \times 224 \times 3$
$224 \times 224 \times 64$
$112 \times 112 \times 128$
$56 \times 56 \times 256$
$28 \times 28 \times 512$
$14 \times 14 \times 512$
$7 \times 7 \times 512$
$1 \times 1 \times 4096$
$1 \times 1 \times 1000$
convolution+ReLU
max pooling
fully connected+ReLU
softmax

VGG Neural Network Architecture – Source

4) VGG16 - To train the model I have used VGG neural network that consist of 16 layers.VGG16 has 13 convolutional layers and 3 fully connected layers. The pretrained weigths of the convolutional layers are used . The input shape provided in (224,224,3). Removed the fully connected layer of the network in order to have a fully connected layer with output layer having softmax activation and 2 nodes. Basically removed the default 1000 classification of the VGG network with two classifications.

5) VGG19 - VGG19 is a variant of VGG model which in short consists of 19 layers. The input shape of (224,224,3) , 16 convolution layers, 3 Fully connected layer, 5 MaxPool layers and 1 SoftMax layer, and a classification layer with two classes.

Given below a comparison table with respect to training accuracy and validation accuracy

Model	Training Accuracy	Validation Accuracy
Convolution Neural Network with image size of 75	0.92	0.78
Convolution Neural Network with image size of 224	0.95	0.81
ResNet50 with Image size 75	0.999	0.901
ResNet50 with Image size 224	0.999	0.9682
ResNet32	0.851	0.667
VGGN19	0.996	0.973
VGGN16	0.998	0.974

9. Conclusion

Tried various architectures to train the model. Simple convolutional neural network and already trained networks ResNet50,ResNet32,VGGN16,VGGN19 were used. In order to reduce the run time, I tried to train model with different image size 75 & 224. Comparing the training accuracy and validation accuracy ResNet50 model with image size 224 gave the best result. Predicted the. labels for the testing dataset and captured the results in a csv file.