Image Classification Using Convolutional Neural Networks
&
Transfer Learning

This assignment is mainly focused on the Image Classification using Convolutional Neural Networks also known as CNN. Out of many Neural Networks, Convolutional Neural Networks is considered to be better for image classification.Let us see how CNN is better than Multilayer Feedforward Neural Network (MLFFNN) and Recurrent Neural Network(RNN)

1. CNN vs MLFFNN:

CNN is better than the traditional (MLFFNN) because mainly of the number of parameters that are to be learnt. For computer vision we need images with very high pixels in order to recognize the images. This leads to a very large number of input nodes in the case of MLFFNN there by many number of hidden nodes. Hence the size of weight matrix is very high. In CNN the number of weights to be learnt is drastically reduced because of Convolution. In convolutions filters are used of small sizes as 3x3 or 5x5 depending on the requirement. Based on the number of filters being used the number of weight parameters that have to be learnt can be calculated. In MLFFNN if it is in millions then it can be reduced to hundreds of parameters using the concepts of Convolution and Pooling.

2. CNN vs RNN:

The main aim of RNN is used to learn the sequence. It remembers the previous events. But in Image Classification we do not need any kind of memory. RNN can rather be used for image captioning. On the other hand CNN will learn the various components of an image (vertical edges, horizontal edges, lines, curves) and further combines these smaller components to recognize larger structures.

Hence choosing CNN over MLFFNN and RNN is justifiable. Below are the 3 major layers present in CNN and the final CNN is formed by arranging many of these layers together to form a deep neural network.

1) Convolution
2) Pooling
3) Fully Connected Neural Network

In this assignment we will look into a basic CNN at first, and then increase the number of layers and then look into various pre trained models present in CNN.

Basic CNN:

This is a binary classifier that is trained on ow to distinguish between CAT and DOG. Hence the output layer consists of only one node which indicates 1 for Dog and 0 for Cat.

Creation of Classifier:

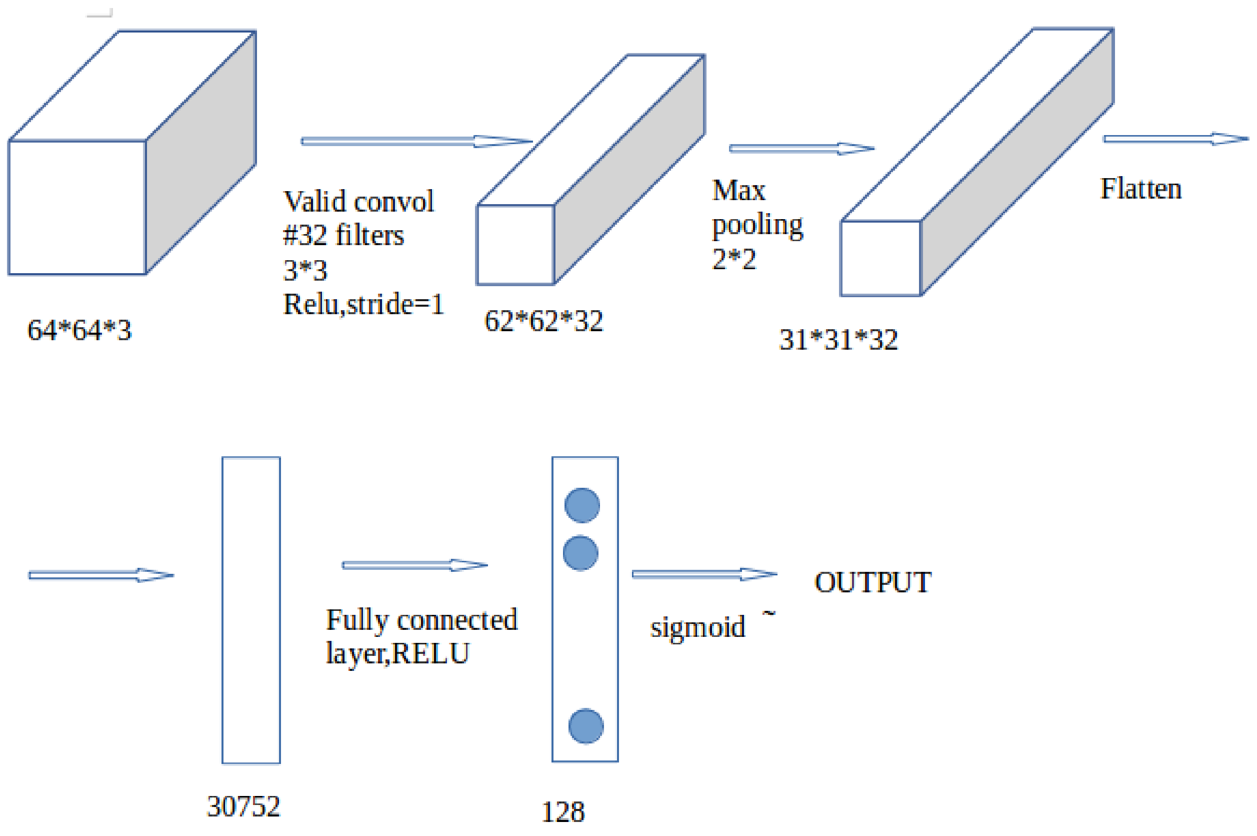
1. **Convolution Operation**
    - The number of filtered used in this operation is 32.
    - (3,3) is the shape of filter.
    - Input image size is 64x64x3 , 3 here stands for RGB
    - Activation function used is “**RELU**”
    - Valid Convolution is performed and hence the image size shrinks.
2. **Max Pooling Operation** The size of the pool is 2x2. Very small size is considered in order to avoid pixel loss, so that we get precise location of where the feature is located.
3. **Flatten Operation** This will convert 2D into 1D array so that it could be used in fully connected neural network. i.e., the next layer
4. **Fully Connected Layer** In this the flattened output of the previous layer is given as the input. This layer can be considered as hidden layer as it lies in between the flattened output and the final output layer.
    - The first parameter units indicates the number of hidden nodes that are present in this layer.
    - The second parameter indicates the activation function used. In this case it is **"RELU"**
5. **Output Layer** This layer contains only one node, as it is a binary classifier we get to know if the image contains CAT or DOG. Sigmoid activation function is used here as it can easily reduce the output to 0 or 1.

After adding all the required layers we compile them in the end.

- Optimizer parameter "adam" indicates **Stochastic Gradient Descent Algorithm**
- Loss function chosen is **Binary Cross Entropy**
- "Accuracy" is the performance metrics chosen

After the classifier is designed, it is trained with the image dataset. Image data set consists of 1600 Training samples and 400 Validation samples and around 2000 testing samples. The time taken for such a small dataset was nearly 1 hour. The training algorithm followed was regular **Stochastic Gradient Descent Algorithm.**

Model Architecture:



After completion of training the metrics are as follows:

```
loss: 0.0261 - acc: 0.9916 - val_loss: 1.7523 - val_acc: 0.7186
```

We observe that the accuracy is nearly **99.16%** which is quite high.

Once the training is completed the model is tested with the Test Data set. Below are the results obtained.

```
Total Dogs:  1012
Predicted as Dogs :  856
Predicted as Cats :  156


Total Cats:  1011
Predicted as Dogs :  559
Predicted ass Cats :  452
```

It is observed that out of 1012 dogs it has predicted 856 as dogs and the rest as cats that is almost **84.56%** is the accuracy for Dogs. Similarly out of 1011 cats the model has predicted 452 as cats and the rest of them as dogs which counts to **44.7%** of accuracy.

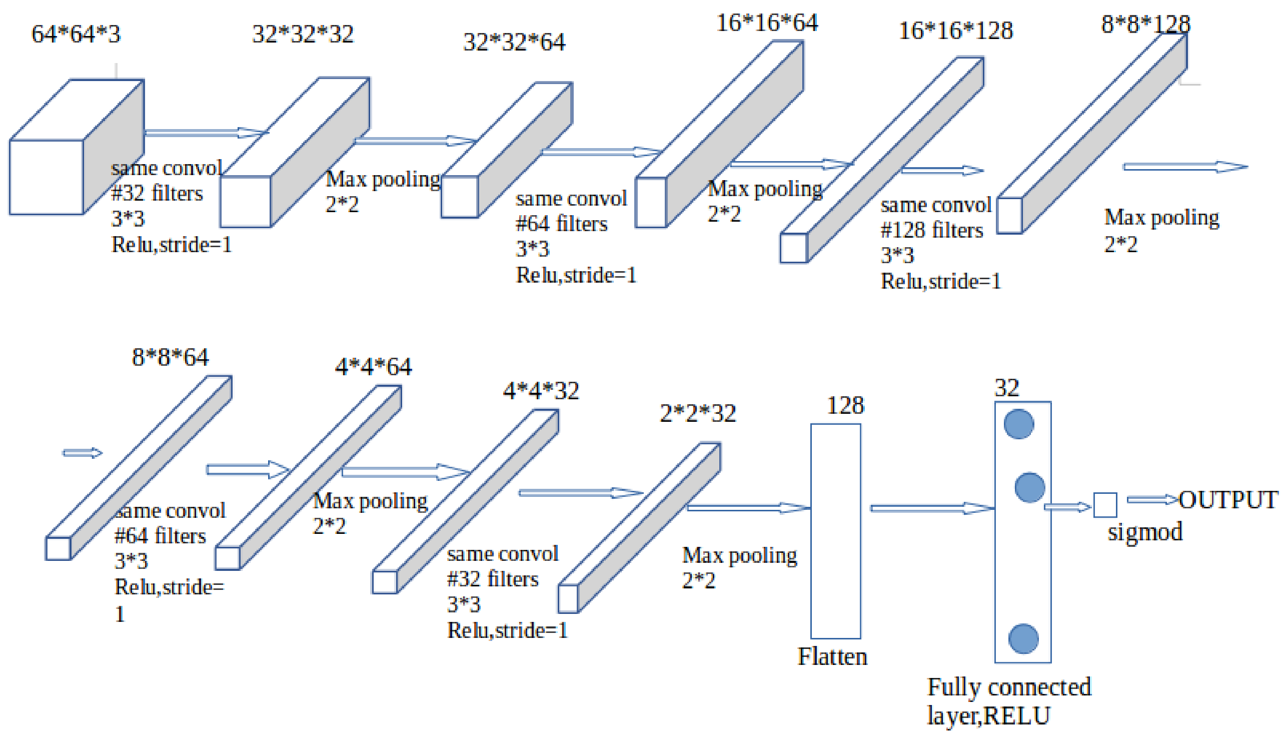
Improvised CNN:

In this the number of layers are increased when compared to the previous CNN. Below is the architecture. The stride value is considered as 1.

- Convolution : 3x3 with 32 filters, Same Convolution (Image does not shrink, image is padded).
- Max Pooling : 2x2
- Convolution : 3x3 with 64 filters, Same Convolution
- Max Pooling : 2x2
- Convolution : 3x3 with 128 filters, Same Convolution
- Max Pooling : 2x2
- Convolution : 3x3 with 64 filters, Same Convolution
- Max Pooling : 2x2
- Convolution : 3x3 with 32 filters, Same Convolution
- Max Pooling : 2x2
- Flatten layer to convert 2D to 1D
- Fully Connected : 32 nodes
- Fully Connected : 1 node

In the last layer Sigmoid activation function is used as it is a binary classifier and the output has to be converted to 0 or 1.

Model Architecture:



After training the classifier (1.5 hours) with the same dataset used in the earlier classifier training, below are the metrics obtained.

```
loss: 0.0211 - acc: 0.9927 - val_loss: 1.2015 - val_acc: 0.7884
```

We observe that the accuracy is little higher than that of the previous model i.e., **99.27%.**

Once the training is completed the model is tested with the Test Data set. Below are the results obtained.

```
Total Dogs:  1012
Predicted as Dogs :  863
Predicted as Cats :  149

Total Cats:  1011
Predicted as Dogs :  360
Predicted as Cats :  651
```

It is observed that out of 1012 dogs it has predicted 863 as dogs and the rest as cats that is almost **85.27%** is the accuracy for Dogs. Similarly out of 1011 cats the model has predicted 452 as cats and the rest of them as dogs which counts to **64.39%** of accuracy.

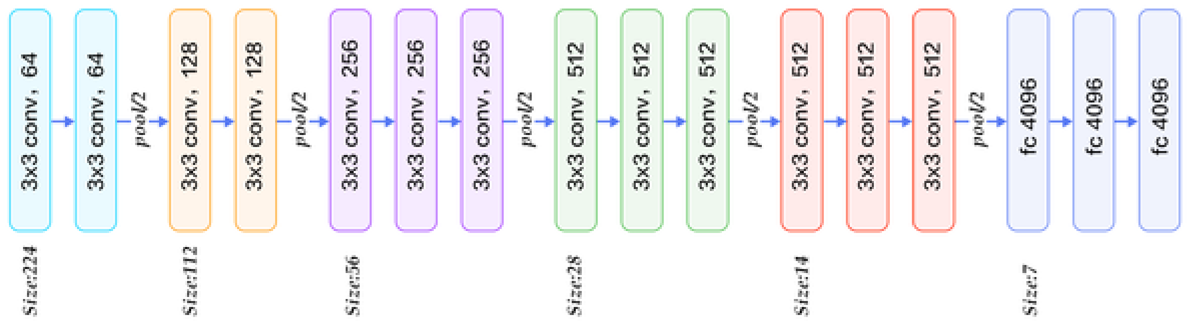
Therefore from the above two classifiers it is noted that the deeper the network better the results.

Pre -Trained Keras Models:

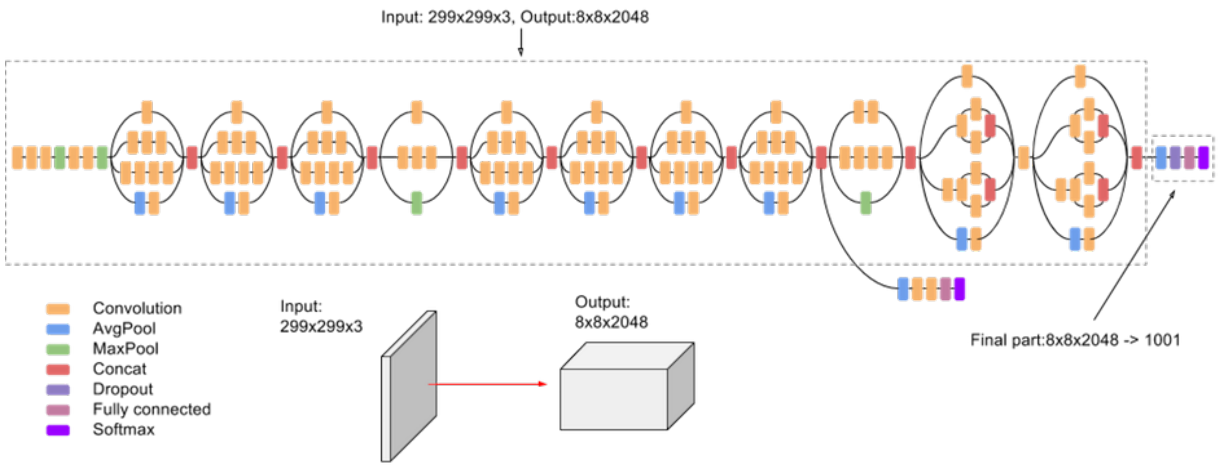
Below are few of the pre-trained models that are trained on ImageNet Dataset are demonstrated in this assignment.

1. **VGG16 Model**: VGG 16(Visual Geometric Group 16) is a deep convolutional neural network used to classify images by processing it in multiple layers.VGG16 has 16 layers in it and uses 3*3 convolution and 2*2 max pooling strategies in the whole network.

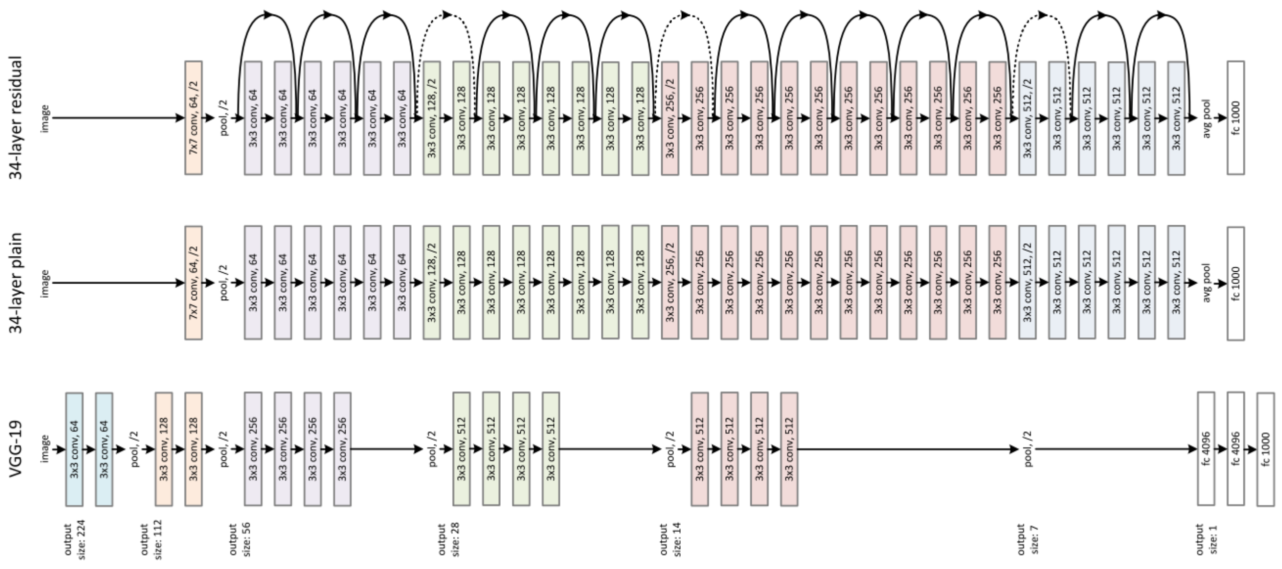
Description of layers:



2. **Inception** : Inception v3 model is a deep convolutional network used for image recognition. This model consists of two parts , feature extraction part using convolutional neural network and a classification part using fully connected layers and soft max layers.The architecture of this model is complex and huge.it consists of several modules and each module is composed of set of convolutional networks to extract features .

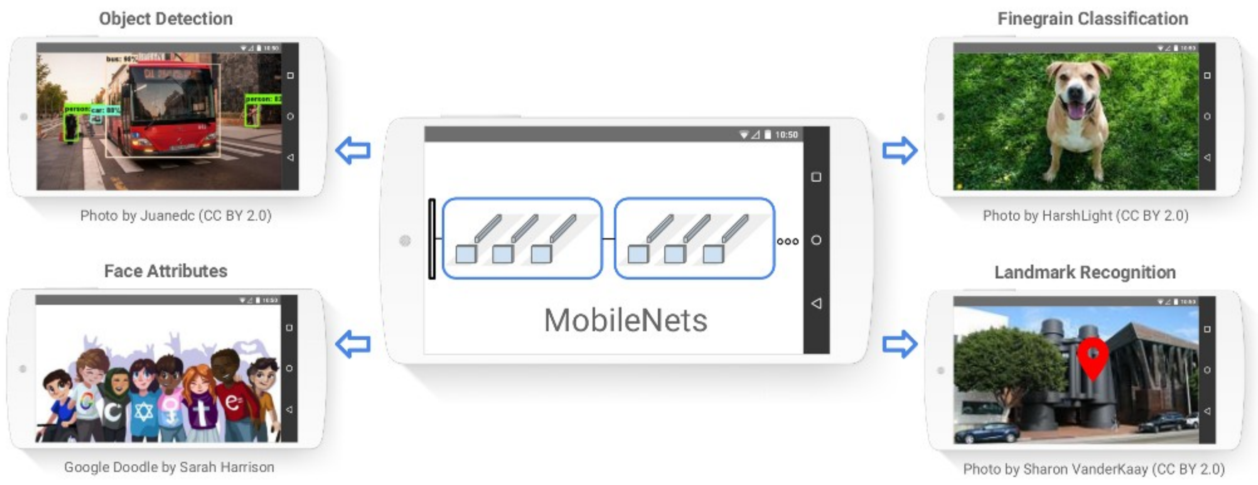


3. **ResNet50** : ResNet50 is a 50 layer residual network,this model instead of learning all the features learns only some residuals i.e., this model skips some of the layers . The main idea in the Resnet concept is “identity shortcut connection” which skips one or more layers. This is mainly done inorder to avoid the Vanishing Gradient problem.



4. **MobileNet** : This network is light weight deep convolutional neural network which is vastly small in size and faster in performance.The size of it is about 17MB and 4.7M parameters,which is very small when compared to other networks.the disadvantage of this network is there is very small reduction in accuracy than other networks. Its architecture is more suitable for mobile and embedded based vision applications. Its architecture uses **depthwise seperable convolutions** using which the number of parameters are drastically reduced. This results in light weight deep neural networks. The normal convolution is replaced by depth wise convolution followed by pointwise convolution which is call as depthwise seperable convolution. Doing this will reduce the size of the weight parametrs that have to be claculated. It is favourable in mobile and embedded vision applications where the compute power is less. By using the depthwise seperable convolution there is some sacrifice in the accuracy.

Applications of MobileNet:


Object Detection
Photo by Juanedc (CC BY 2.0)
Face Attributes
Google Doodle by Sarah Harrison
MobileNets
Finegrain Classification
Photo by HarshLight (CC BY 2.0)
Landmark Recognition
Photo by Sharon VanderKaay (CC BY 2.0)

Fine-grain Classification: It is a sub-field of object recognition, aims to distinguish subordinate categories within entry level categories. It can also be thought as classification with categories that are very similar.

Dog and Cat detection accuracies of pre-trained Keras Models:

Below are the results obtained after testing our dataset on the pre-trained keras models.

Pre Trained Model Name	Accuracy for Dog detection	Accuracy for Cat detection
VGG16	0.9298	0.6904
Inception	0.9357	0.8110
ResNet50	0.9328	0.6824
MobileNet	0.9249	0.7388

It is observed that among all the four pre-trained models performance of **Inception** model is the best with **93.57%** in Dog detection and **81.1%** in Cat detection.

Transfer Learning:

In our daily life we apply this concept of transfer learning in enormous situations. Like we use the concept learnt during riding bicycle to ride a bike, we do not start it from the very beginning. Similarly learning Tennis from badminton, learning Violin from Guitar and many more. Therefore Transfer Learning can be defined as the process in which knowledge gained by a model while performing one task can be used to perform some other task. For example, if a model is trained on image classification of dogs and cats then it can be used for reading X-Ray scans. Suppose that a model is trained on image recognition. If we want to use this model to perform some other tasks like radiology diagnosis, the last layer of the neural network and add a new layer with randomly initialised weights and that output can be used for radiology diagnosis. A new dataset is taken (in this scenario the radiology dataset) and then retrain the neural network with this new dataset. There are various ways in which we can train the dataset. Below are the various scenarios:

Case 1 : New dataset is small and similar to original dataset

In this case as the dataset is small it is sufficient if we train only the last layer. In our assignment we use the pre-trained VGG16 model of Keras trained on ImageNet Dataset for the purpose of Cat and Dog binary Classification.

Case 2 : New dataset is large and similar to the original dataset

Here the model can be trained in two ways.

Train the entire model as we have large data, we can be confident that the model is not Over Fitted if trained throughout all layers of the model.

Freeze a few layers of the beginning and then train the remaining layers of the network. This process is called as the “Fine Tuning”

In Fine Tuning the first few layers are freezed because in the very initial stages of learning the model is trained on what type of dataset is used, recognising edges (vertical and horizontal), few features which are present in all the images. This part of learning can be skipped and the further task of learning the actual dataset can be carried out.

Case 3: New dataset is small but very different from the original dataset

Since the dataset is very small, We may want to extract the features from the earlier layer and train a classifier on top of that.

Add few FC layers and output layer.

Set the weights for earlier layers and freeze them.

Train the network.

Case 4 : New dataset is large and very different from the original dataset.

This is straightforward. since you have large dataset, you can design your own network or use the existing ones.

Transfer Learning using VGG16:

In our assignment we are using the pre-trained VGG16 model of Keras trained on ImageNet dataset for binary classification of Dogs and Cats. In this model the entire Fully Connected layers of the VGG16 pre-trained model are replaced with a new network which consists of the following layers:

- Flatten Layer : Converts 2D to 1D array
- Dense Layer : It is fully connected layer with 1024 nodes, RELU activation function
- Dropout Layer : Probability of considering nodes is 0.8
- Dense Layer : It is fully connected layer with 1024 nodes, RELU activation function
- Dense Layer : Output layer 1 node, Sigmoid activation function

The time taken to train this model is almost 2.5 hours.

Dropout Layer:

It refers to dropping out few nodes in the layer (hidden or visible) in a neural network. It refers to ignoring a few units while training the network. The nodes are chosen randomly, and these nodes are ignored in a particular forward or backward propagation. Here we are mentioning 0.8 which refers to the probability of number of nodes that have to be considered while training.

The main intention of using Dropout layer is to prevent overfitting. This is mainly done in the fully connected layer as the nodes develop co dependency between each other which reduces the individual power of each neuron which leads to overfitting of training data.

Binary Cross Entropy Loss Function:

While compiling we use Binary Cross Entropy Loss function rather than Categorical Cross Entropy loss function because we have only 0 or 1 as our output.

Results Obtained after training:

```
Total Dogs:  1012
Dogs :  995
Cats :  17
Accuracy : 0.983201581027668

Total Cats:  1011
Dogs :  88
Cats :  923
Accuracy : 0.9129574678536103
```

We can observe that the Accuracy for Dog prediction has drastically increased to **98.3%** and that of Cat has increased to **91.29%** which has surpassed the prediction done by the pre-trained models.