Cats and Dogs Classification Using MobileNet And Transfer Learning

##### AUTHOR

*Muhammad Omer Bin Atique  
2020335 ES 304 B, BCS, Batch 30   
GIKI*

*Abstract*—I present to you a very simple binary classification neural network for a Cat and Dog data set. This is one of the more basic image classification problems. The model build can accurately classify between Cat and Dog image. Furthermore, the same model with a few changes can definitely be used for multi-classification problems.

Keywords—Model, Image Classification, Binary classification, Neural network, Images, Mobile Net, Transfer Learning, Accurate, Multi-Classification.

# Introduction

## Binary Classification

Binary obviously means 2. So, just from the termanology we can deduce what binary classification means. It is the classification between 2 classes. We can have binary classification on tabular data as well as images or videos.. It can range from 1or 0, dead or Alive, Animal or Human, or in our case Cat or Dog.

## Multi-Classification

Simailar to Binary classification, multi-classification is between more than 2 classes. There are many application which deal with multi-classification. One popular one is classification over Cifar-10 or Cifar-100 datasets.

##### METHODOLOGY

# i. Data Set

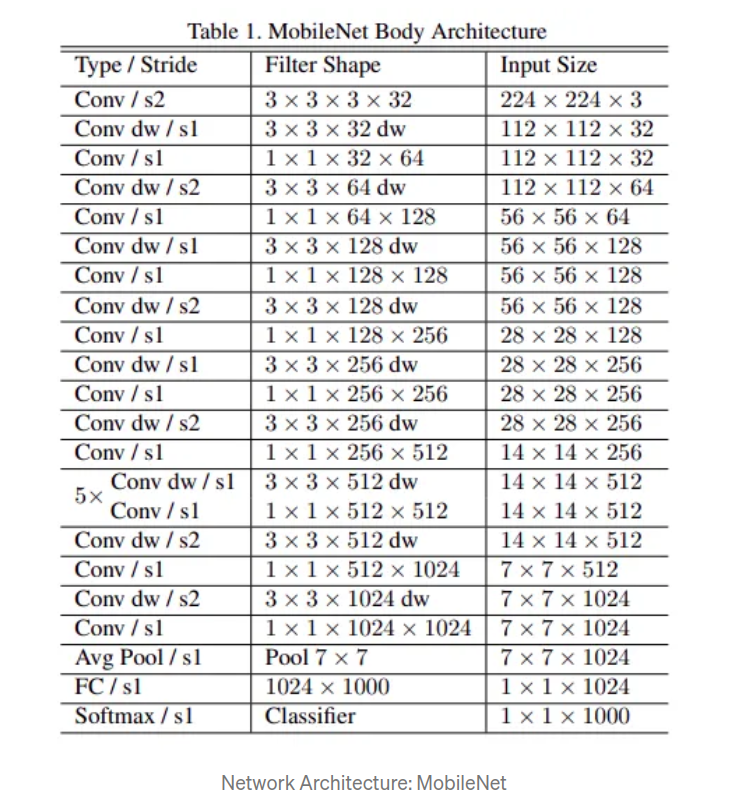
The data set used is from Kaggle. It is a data set of cats and dogs images with both seperate training sets and test sets. The training set has exactly 8006 images. Out of which 4005 are dog images and 4001 are cat images. The test set has 2024 images with each each category having 1012 images.

# Transfer Learning

Transfer learning is a machine learning technique where a model trained on one task is re-purposed on a second related task. Transfer learning is an optimization that allows rapid progress or improved performance when modeling the second task. Transfer learning is popular in deep learning given the enormous resources required to train deep learning models or the large and challenging datasets on which deep learning models are trained. We have used a pretrained model while training this model called MobileNet.

# MobileNet

#### MobileNet is a simple but efficient and not very computationally intensive convolutional neural networks for mobile vision applications. MobileNet is widely used in many real-world applications which includes object detection, fine-grained classifications, face attributes, and localization.

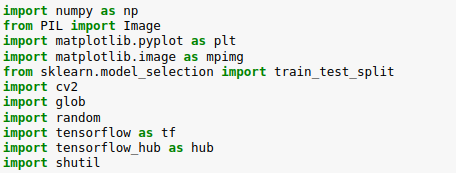


I have used MobileNet as the training architecture for my Convolution Neural Network as it is computationally very efficient.

# Python Code Implementation

## Included Libraries

First let’s mention the included libraries.



**Numpy:** is used for mathematical calculations and array operations.

**PIL:** .

**Matplotlib.pyplot:** Helps visualize images in python.

**Cv2:** helps working with images.

**Sklearn:** linalg stands for linear algebra. Helps with calculations involving linear algebra.

**Random:** Helps in generating random numbers.

**Glob:**The glob module finds all the pathnames matching a specified pattern according to the rules used by the Unix shell.

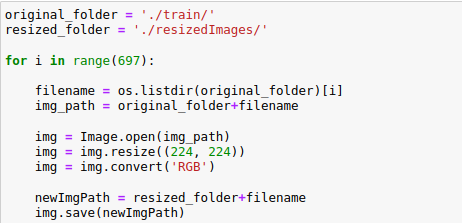
**Tensorflow:** For large numerical computations keeping deep learning in mind.

**Tensorflow\_hub:** TensorFlow Hub is a library to foster the publication, discovery, and consumption of reusable parts of machine learning models.

**Shutil:** The shutil module offers a number of high-level operations on files and collections of files.

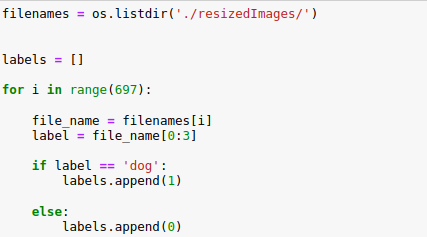
## Preprocessing

* *Resize Images:*

Images from the training set were extracted and then resized for the ease of training from their arbritrary size to (224 x 224). Then they were set to a new folder called resizedImages. Only a total of 695 images were chosen to be trained and resized for ease of computation.

*Classification of Labels:*

As we know that there are a total of 2 labels 0 and 1. 0 represents cats and 1 represents dogs. We make a list of the size of training set (695) and for each image with the name cat in its file name we write 0 to the corresponding index in the list and 1 for dogs.



## Train-Test Split:

Now we use Sklearn.model\_selection.train\_test\_split() function to split the training set and the y labels into total 4 sets. X and y sets for training the model and X and y sets for the test set to evaluate the model on unseen data.



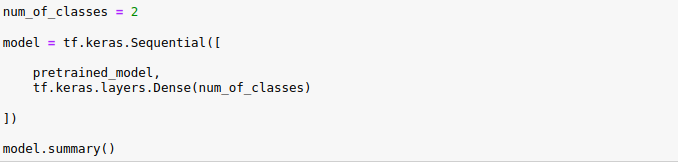
 Then we normalize the X sets to map the values between 0 and 1 for ease of computation and better training.

## Training:

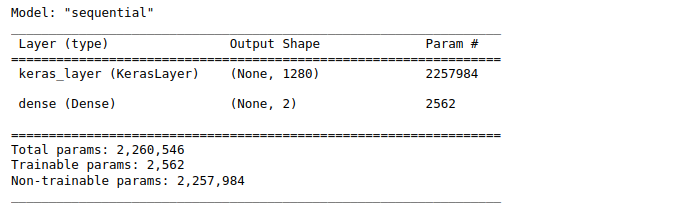
## Now we import the Mobile Net model and make it into a neural network layer. The model is taken from its online repository and tensorflow hub is used for making a layer.



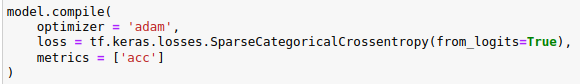
Then we make the model for training:



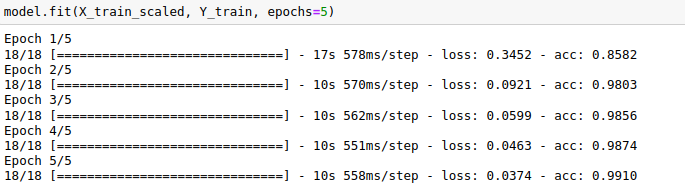
The summary is as follows:



Then we compile the model:

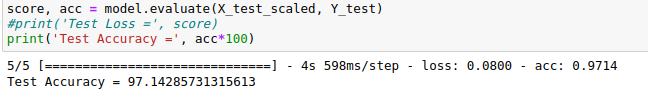


Then we train the model using model.fit() for 5 epochs:



## Evaluation

Then we evaluate the model using the test sets. We get an accuracy of 97.14% :



##### RESULTS

We then try to predict some custom pictures of our own. Using a custom build classifyImage function. Some examples are given below.

