Machine Learning Engineer Nanodegree

Capstone Project

Classifying Flowers Data Set

Reddy pavan kumar

December 28th, 2018

Proposal:

Classifying Flowers data Set

I. Definition

Project Overview:-

In modern times, people have sought ways to cultivate, buy, wear, or otherwise be around flowers and blooming plants, partly because of their agreeable appearance and smell. Around the world, people use flowers to mark important events in their lives.People therefore grow flowers around their homes, dedicate parts of their living space to flower gardens, pick wildflowers, or buy commercially-grown flowers from florists.

History:-

The Ancient Egyptians:-

The earliest known flower arranging dates back to ancient Egypt. Egyptians were decorating with flowers as early as 2,500 BCE. They regularly placed cut flowers in vases, and highly stylized arrangements were used during burials, for processions, and simply as table decorations. Illustrations of arranged flowers have been found on Egyptian carved stone reliefs and painted wall decorations.Flowers were selected according to symbolic meaning, with emphasis on religious significance. The lotus flower or water lily, for example, was considered sacred to Isis and was often included in arrangements. Many other flowers have been found in the tombs of the ancient Egyptians, and garlands of flowers were worn by loved ones and left at the tombs. These included blue scilla, poppy-flowered anemone, Iris sibirica, delphinium, narcissus, palm tree, papyrus and rose. Egyptian wall paintings depicting roses have been found in tombs dating from the fifth century B.C. to Cleopatra' s time.

The Ancient Greeks and Romans:-

The Greeks and the Romans also used flowers. The ancient Greeks used flowers and herbs for adornment and decorations included in artwork. They did not often use vases, focusing instead on garlands and wreaths. They would place plant material, such as olive branches, in terracotta. The leafy branches were probably used for weddings. They also tossed petals onto floors and beds. Like the Egyptians, the Greeks and Romans had preferences for the flowers and foliage they used.

Refer the link:-

https://en.wikipedia.org/wiki/History_of_flower_arrangement

Applications:-

A) can used by kids in schooling to Cleary differentiate between flowers based on medium. And it is easy to use.
B) Can be used for research purposes on different categories of flowers.

We can also expect good performances around 80% accuracy with this project as we are considering only a sub category of different types of flowers

FUTURE: We can even extend this algorithm to remaining categories of flowers.

.In this project, I have taken 5 different flowers to classify and their relevant description.

1.Daisy:- Bellis perennis is a common European species of daisy, of the Asteraceae family, often considered the archetypal species of that name. Many related plants also share the name "daisy", so to distinguish this species from other daisies it is sometimes qualified as common daisy, lawn daisy or English daisy.(Wikipedia)

2.Dandelion:- Taraxacum is a large genus of flowering plants in the family Asteraceae, which consists of species commonly known as dandelions. They are native to Eurasia and North America, but the two commonplace species worldwide, T. officinale and T. erythrospermum, were introduced from Europe and now propagate as wildflowers.(Wikipedia)

3.Rose:-A rose is a woody perennial flowering plant of the genus Rosa, in the family Rosaceae, or the flower it bears. There are over three hundred species and thousands of cultivars. They form a group of plants that can be erect shrubs, climbing or trailing with stems that are often armed with sharp prickles.(Wikipedia)

4.Sunflower:-Helianthus annuus, the common sunflower, is a large annual forb of the genus Helianthus grown as a crop for its edible oil and edible fruits. This sunflower species is also used as wild bird food, as livestock forage, in some industrial applications, and as an ornamental in domestic gardens.(Wikipedia)

5.Tulip:-Tulips form a genus of spring-blooming perennial herbaceous bulbiferous geophytes. The flowers are usually large, showy and brightly coloured, generally red, pink, yellow, or white. They often have a different coloured blotch at the base of the tepals, internally.(Wikipedia)

Problem Statement:-

To understand the difference between types of flowers. The aim of this project is to predict the type of flower it belongs to by visualizing image.

In the project, I am going to use various Convolutional Neural Networks to predict the types of images and compare their performance and finally declare my final model.

Metrics:-

I want to use accuracy as evaluation metric for flowers classification. Accuracy is a common metric for categorical classifiers

Accuracy = (images correctly classified) / (all images)

During development, a validation set was used to evaluate the model. I want to use a small set of training images as my validation images.

For validation I want to use "categorical_crossentropy" as loss metric for CNN, optimizer as "momentum gradient descent" and also metrics as "accuracy"

II. Analysis

Data Exploration:-

The Classifying flower dataset has 4242 images;
Total 5 categories of images are present in flower dataset, namely

Daisy dandelion rose sunflower tulip

The dataset that I am working is downloaded from

https://www.kaggle.com/alxmamaev/flowers-recognition. This dataset contains 4242 images of flowers.

content :-

The pictures are divided into five classes: Daisy, tulip, rose, sunflower, dandelion. For each class there are about 800 photos. Photos are not high resolution, about 320x240 pixels. Photos are not reduced to a single size, they have different proportions

Citation:-

The data collection is based on the data flicr, google images, yandex images.

Dataset for classifying different flowers. Main categories have been taken:-

https://yandex.com/images

https://www.flickr.com/photos/tags/data

The data is open – sourced and can be download for education purpose with no citation.

Exploratory Visualization:-

There are 5 total image categories.

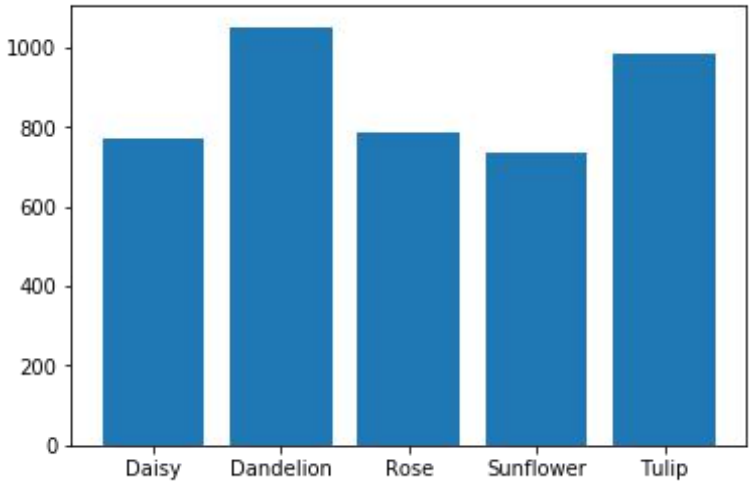
There are 4323 total flowers images.

There are 3501 training flowers images.

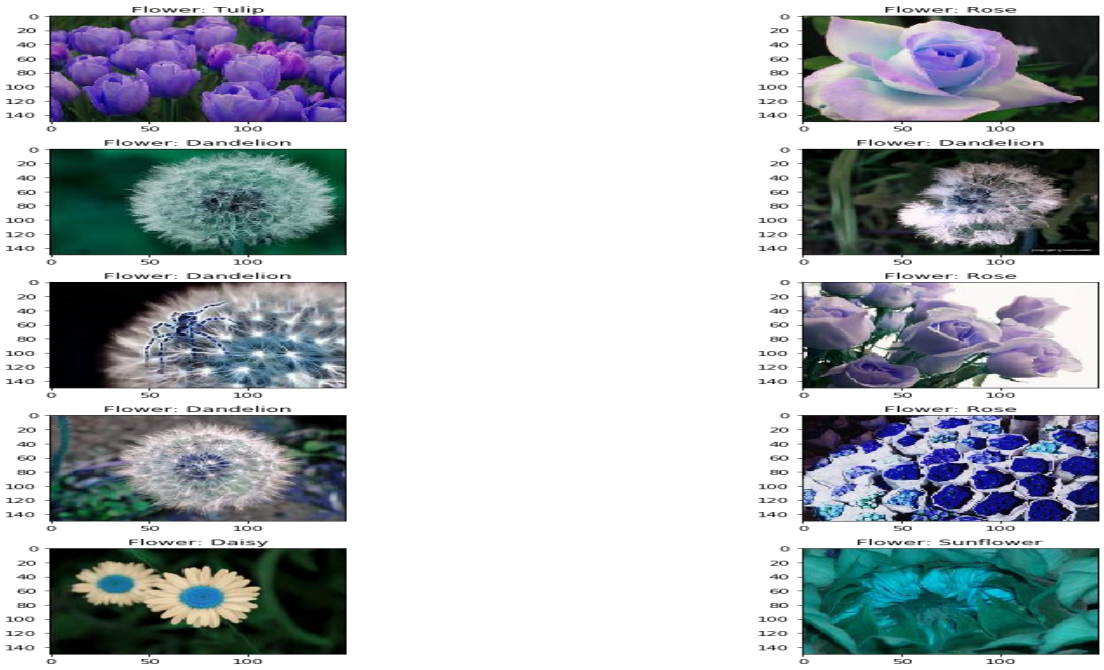
There are 389 validation flowers images.

There are 433 testing flowers images.

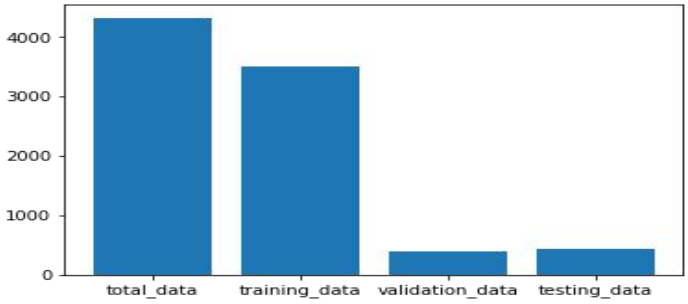
```
Total dataset image categories.
```



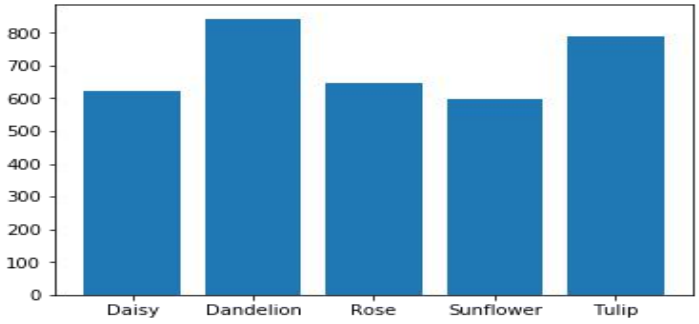
Sample images form the total dataset is



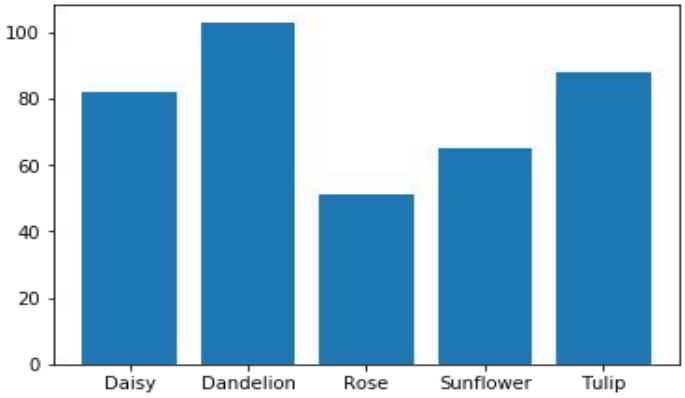
Total data is divided into train valid test data.



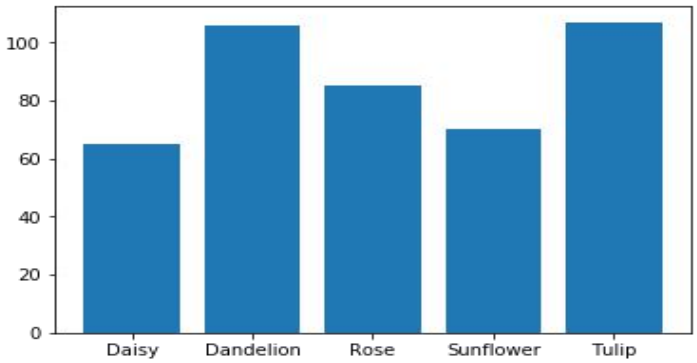
Training `categories:-`



Validation `categories:-`



Testing categories:-



So far, data is in good way and retrieving successfully.

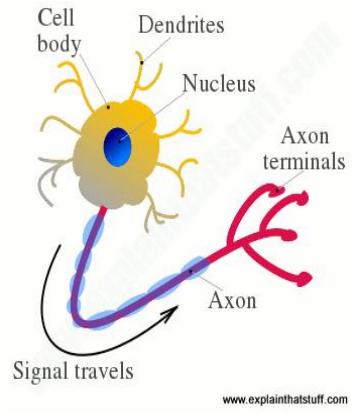
Algorithms and Techniques:-

Theory behind Scenes:-

Artificial neural networks (ANNs): ANNs are computing systems vaguely inspired by the biological neural networks that constitute animal brains and humans. these systems "learn" to perform tasks by considering examples, generally without being programmed with any task-specific rules.

A typical brain contains something like 100 billion miniscule cells called **neurons** (no-one knows exactly how many there are and estimates go from about 50 billion to as many as 500 billion).

Each neuron is made up of a **cell body** (the central mass of the cell) with a number of connections coming off it: numerous **dendrites** (the cell's inputs—carrying information toward the cell body) and a single **axon** (the cell's output—carrying information away). Neurons are so tiny that you could pack about 100 of their cell bodies into a single millimeter. Inside a computer, the equivalent to a brain cell is a tiny switching device called a transistor. The latest, cutting-edge microprocessors (single-chip computers) contain over 2 billion transistors; even a basic microprocessor has about 50 million transistors, all packed onto an integrated circuit just 25mm square (smaller than a postage stamp)!

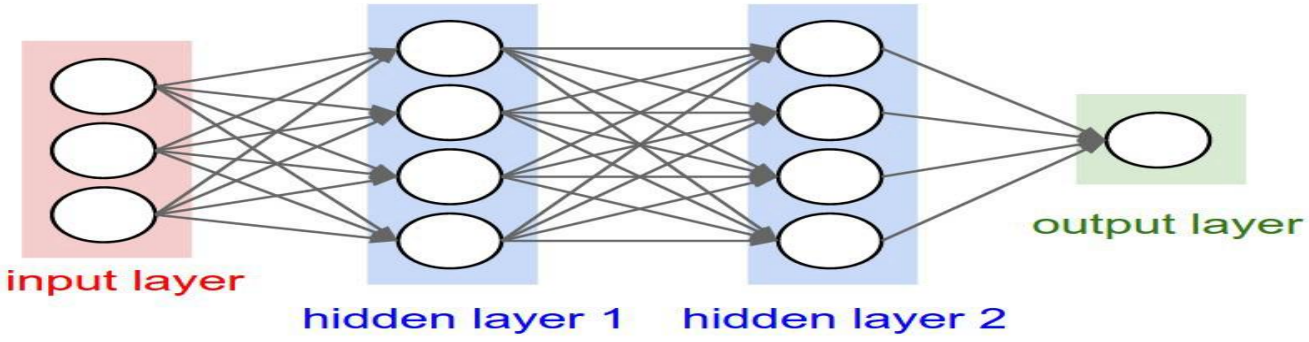


ANN is a set of connected neurons organized in layers:

- **input layer:** brings the initial data into the system for further processing by subsequent layers of artificial neurons. Like dendrites in human neuron
- **hidden layer:** a layer in between input layers and output layers, where artificial neurons take in a set of weighted inputs and produce an output through an activation function. Like nucleus in human neuron.
- **output layer:** the last layer of neurons that produces given outputs for the program. Like axon in human neuron

NOTE: no one how they process information inside.

A typical ANN will look like:



WORKING PROCEDURE:-

Step 1- Model initialization: -

A random initialization of the model is a common practice. The rationale behind is that from wherever we start, if we are perseverant enough and through an iterative learning process, we can reach the pseudo-ideal model.

Step 2- Forward propagate:-

The natural step to do after initializing the model at random, is to check its performance.
We start from the input we have, we pass them through the network layer and calculate the actual output of the model straightforwardly.

Step 3- Loss function:-

At this stage, in one hand, we have the actual output of the randomly initialized neural network.
On the other hand, we have the desired output we would like the network to learn. Then we define what we call: loss function (for intuition just think loss function like absolute difference or squared difference, but it changes with model to model). Basically, it is a performance metric on how well the NN manages to reach its goal of generating outputs as close as possible to the desired values.

Step 4- Differentiation:-

we can use any optimization technique that modifies the internal weights of neural networks in order to minimize the total loss function that we previously defined.

Step 5- Back-propagation:-

Then error in loss function is propagated from output layer to input layer by using differentiation we solved in step-4

Step 6- Weight update:-

Then corresponding weights in the network are changed in order with learning rate.

*New weight = old weight—Derivative Rate * learning rate*

Step 7- Iterate until convergence:-

Just iterate the procedure from step2 to step6 until convergence

NOTE: How many iterations are needed to converge?

- This depends on how strong the learning rate we are applying. High learning rate means faster learning, but with higher chance of instability.
- It depends on the optimization method
- It depends as well on the meta-parameters of the network (how many layers, how complex the non-linear functions are

Classifier I used for this data set is VGG16 and RENET50 neural networks.since I have small amount of data I use the data augmentation.

The following parameters can be tuned to optimize the classifier:

❖ Training parameters:

➢ Training length (number of epochs)

➢ Batch size (how many images to look at once during a single training step)

❖ Neural network architecture:

➢ Number of layers

➢ Layer types (convolutional , fully-connected , or pooling)

Benchmark:-

Bench mark model: Any CNN model that gives accuracy around 20% is my benchmark model.
My created model
Conv2D layer with 16 filters,kernel_size is equal to 2 ,padding is 'same' and input shape is (150, 150 ,3)
Max Pooling layer with pooling size is equal to 2.
flatten layer and Dense layers with 5 units and activation of "SoftMax"
At compilation phase loss='categorical_crossentropy', optimizer='SGD', metrics=['accuracy'] are used.
check pointer is used with 'weights. best. from bench. hdf5', as file path and save_best_only is tuned to True
bench model is trained on fit_generator method in order to achieve the data augmentation with 32 as batch size and 110 steps per epoch and number epochs are 1

Architecture:-

_____

Layer (type) Output Shape Param #

==========

conv2d_1 (Conv2D) (None, 150, 150, 16) 208

_____

max_pooling2d_1 (MaxPooling2 (None, 75, 75, 16) 0

_____

flatten_1 (Flatten) (None, 90000) 0

_____

dense_1 (Dense) (None, 5) 450005

===

Total params: 450,213

Trainable params: 450,213

Non-trainable params: 0

I declared bench mark model with single convolutional layer and dense layer.

It gives me the training accuracy: 23.19%

It gives me the testing accuracy: 24.7113%

III. Methodology

Data Preprocessing:-

Data Preprocessing The preprocessing done in the "Prepare data" notebook consists of the following steps:

There are also some preprocessing steps which are done as the images get loaded into memory before

1. The list of images is randomized: to get good redistribution of data when applying split
2. The images are divided into a training set and a validation set (10% of original train set): To avoid overfitting
3. The images are resized into 3D tensor with shape of width = 150, height = 150, channels = 3 shaped arrays.: so that every image will have same dimensions.
5. all images are then normalized by dividing with 255.
6. all labels are converted to 5 categories by using keras.utils.to_categorical: in order use for categorical classification.

Implementation:-

As in implementation model I have created two models.one is using VGG16 model with pretrained weights and another is using ResNet50 model with pretrained weights.

Model1: RESNET50 Model:-

Training phase:-

I choose the RESNET50 model and add some more layer to the model which will looks like.

Architecture:-

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
resnet50 (Model)             (None, 5, 5, 2048)        23587712
_________________________________________________________________
flatten_2 (Flatten)          (None, 51200)             0
_________________________________________________________________
dense_2 (Dense)              (None, 512)               26214912
_________________________________________________________________
dense_3 (Dense)              (None, 5)                 2565
=================================================================
Total params: 49,805,189
Trainable params: 49,752,069
Non-trainable params: 53,120
```

Pretrained weights for Resnet50 are downloaded form.
https://www.kaggle.com/ishootlaser/resnet50-pretrained-weights

Compilation phase:-

Trained model has compiled using "SGD" as optimizer and 'categorical_crossentropy' as loss function, used metrics is "accuracy".
check pointer is used with 'weights.best.from_resnet50.hdf5', as file path and save_best_only is tuned to True
Model is trained by using fit method with parameters 20 as epochs 32 as batch size
While 90% of training data is using for training purpose remaining 10% of data is used for validation purposes.
And used data augmentation.

Model1: VGG16 Model:-

Training phase:-

I choose the RESNET50 model and add some more layer to the model which will looks like.

Architecture:-

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
```

vgg16 (Model) (None, 4, 4, 512) 14714688

_____

flatten_3 (Flatten) (None, 8192) 0

_____

dense_4 (Dense) (None, 512) 4194816

_____

dense_5 (Dense) (None, 5) 2565

==========

Total params: 18,912,069

Trainable params: 18,912,069

Non-trainable params: 0

Pretrained weights for VGG16 are downloaded form

https://www.kaggle.com/keras/vgg16

Compilation phase:-

Trained model has compiled using "SGD" as optimizer and 'categorical_crossentropy' as loss function, used metrics is "accuracy".

check pointer is used with 'weights.best.from_VGG1650.hdf5', as file path and save_best_only is tuned to True

Model is trained by using fit method with parameters 20 as epochs 32 as batch size

While 90% of training data is using for training purpose remaining 10% of data is used for validation purposes. And used data augmentation.

IV. Results

Model1:

Model Evaluation and Validation:-

	precision	recall	f1-score	support
Daisy	0.90	0.88	0.89	65
Dandelion	0.92	0.95	0.94	106
Rose	0.89	0.85	0.87	85
Sunflower	0.92	0.96	0.94	70
Tulip	0.90	0.89	0.89	107
micro avg	0.91	0.91	0.91	433
macro avg	0.91	0.90	0.90	433
weighted avg	0.90	0.91	0.90	433

classification report for RESNET50 model to evaluate results:-

As we can see that all metrics (precision, recall, f1-score) for Dasiy,Dandelion,sunflower and Tulip are above 90%.for remaining one categories also gets above 84% f1-score. From this we can conclude that RESNET50 is robust to unseen data. And can be able to generalize the data pretty well.
We can absolutely trust this model for future improvisation. Not only good but it also exceeded my expected outcome of 80%.

Justification:-

When compared with my benchmark model, my model gives training accuracy up to 97%
Whereas my testing accuracy is around 90.3%
The results obtained from my model above satisfactory as both training and testing have accuracy scores are above 90% which is pretty good.
Now I can confidently say that my model1 solution significant to solve the problem.

Model2:

Model Evaluation and Validation:-

	precision	recall	f1-score	support
Daisy	0.92	0.86	0.89	65
Dandelion	0.95	0.93	0.94	106
Rose	0.83	0.91	0.87	85
Sunflower	0.93	0.99	0.94	70
Tulip	0.89	0.84	0.89	107
micro avg	0.90	0.90	0.90	433
macro avg	0.90	0.91	0.90	433
weighted avg	0.90	0.91	0.90	433

classification report for VGG16 model to evaluate results:-

As we can see that all metrics (precision, recall, f1-score) for Dasiy,Dandelion,sunflower and Tulip are above 90%.for remaining one categories also gets above 84% f1-score. From this we can conclude that RESNET50 is robust to unseen data. And can be able to generalize the data pretty well.
We can absolutely trust this model for future improvisation. Not only good but it also exceeded my expected outcome of 80%.

Justification:-

When compared with my benchmark model, my model gives training accuracy up to 92%
Whereas my testing accuracy is around 90.3%

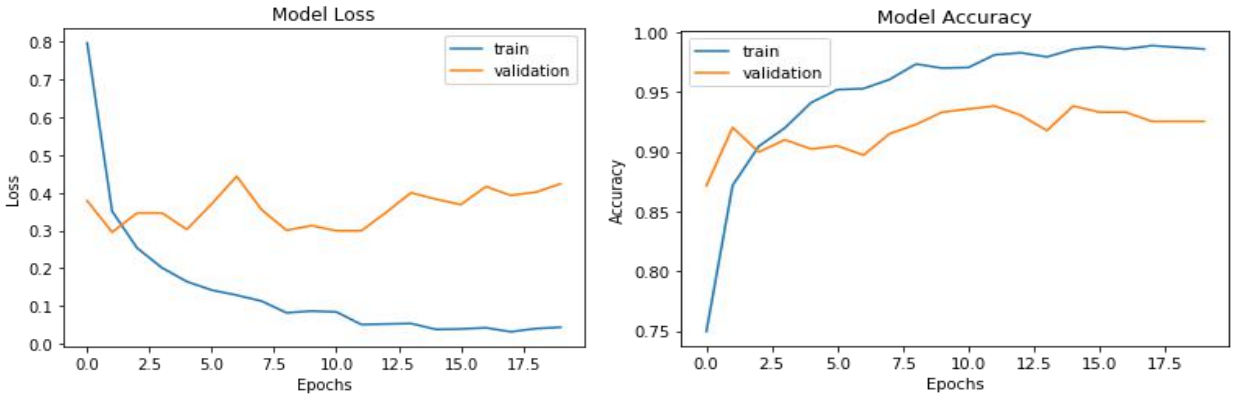
The results obtained from my model above satisfactory as both training and testing have accuracy scores are above 90% which is pretty good.
Now I can confidently say that my model1 solution significant to solve the problem.

Which model to choose:-

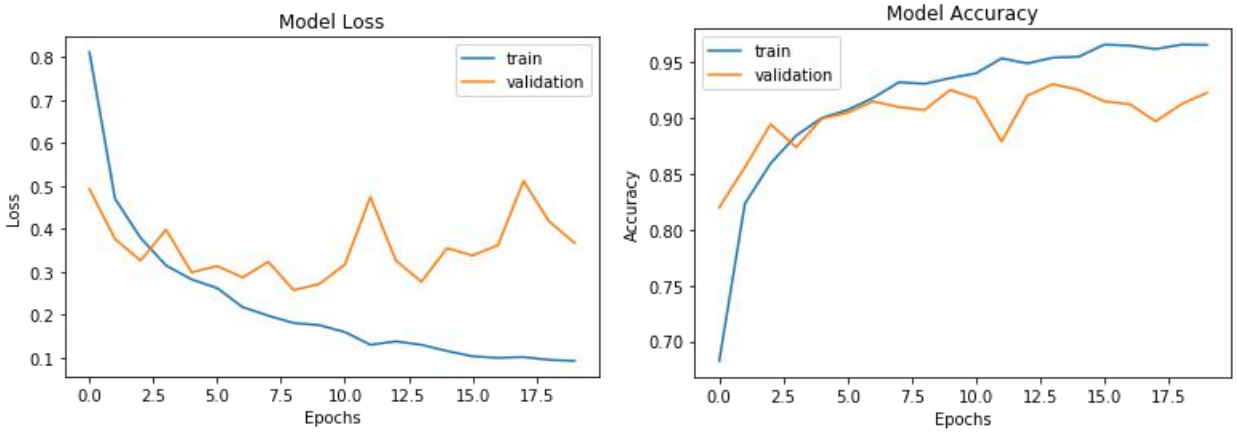
I think there is tie between VGG16 and RESNET50 it head to say which model is best.according to classification report f-score for both models 0.90 which is pretty good.if u have a less computation power like GPU and CPU I would like prefer the VGG16 model seen it as less number of convolution layer and pooling layer it take 8 epochs to find the best parameters and RESNET50 has taken 12 epochs to find best parameters and also it has more number of residual layers are more and more parameters need to be learned.

V. Conclusion

History of loss and accuracy for the RESNET50 model.



History of loss and accuracy for the VGG16 model.



Reflection:-

In this capstone project I have taken image classification as my thought of interest inspired from dog breed classifier. In this process I have learned many things.

1) First thing I have learned about data retrieval processes. When I am doing research about retrieval process, I have come across a lot of surprising methods try. Out of all I decided to use load files method in cv2
2) I have also learn how to use kaggle kernals and how to commit it and reproduce my work.
3) Then I used my skills on representation of overall number of images, and also category wise in both training and testing images by using Matplotlib library. I have realized that its most informative in beginning in understanding size of your project.
4) When I stated loading images, I have found many useful methods like image. load_img in keras. preprocessing, cv2.load_img, plt.imread etc., there are lots of them.
5) I am also learn how to handle the corrupted images in dataset
6) Then I have learned about using tqdm for reading a list of images.
7) I learn that Image resizing is more important in image classification because we can't expect every image of same size.
8) Then image normalization is used to standardize the all image's (division by 255). I came to know about how numbers in image array changes with brightness. How dark areas will replicate zeros, bright areas replicate 225 etc.,
9) Then I learned that How we divide the data into training and validation using traint_test_split method.
10) Here comes the heart of project, creating a CNN model, fitting it to training data and testing on test data evaluating validation curves, learn from confusion matrix doing modifications on models and there's more going on.
11) Last but not least without visualizing results we can't trust the robustness of a model.

Improvement:-

1)One thing, that can be improved from my models is we can use Kfold method for splitting data while fitting model, when we have enough time.
2)we can include many flower categories to it
3). Rather than differentiating between different flower categories, we can also train model to differentiate between what is flower and what is not.

FINAL NOTE:-

But, when I trying to improve my two models, I have done two things.
Instead of train_test_split method, I have used Kfold split method, soon I came to observe that, both models are taking infinite amount of time to fit training data, so to speak I kind of drop the idea of using it.

When I started this project I used RESNET50 and VGG16 model with out pre-trained weights I build the models manually in keras tried this takes days to train the model. So I quit the idea.

Though Both models are giving same performances on training and testing data,

I would preferred VGG16 model, because, its training time is far less than RESNET50 model,

Resnet50 model training time exponential to VGG16 training time

So, I consider VGG16 model as my final model.