Using Tensorflow for crop disease classification by image classification

Introduction

Over the past decade, Computer Vision witnessed tremendous advancement thanks to Machine Learning and Artificial Intelligence. Technological advancements enabled us to feed larger datasets to neural networks and get better results with high accuracy, thus overcoming the "data plateau". Google Lens is a great example of how we have trained machines to not only detect multiple objects but also classify them. In this article, we see how the agricultural industry can benefit from such emerging technologies.

Theory

One of the most challenging aspects of training any image classifier by neural networks is the amount of data it requires. Conventionally, thousand of positive training set images are to be present if you want your model to be accurate and with their individual bounding boxes files(.XML). This is where Transfer Learning comes into the picture. In essence, transfer learning can be signified as how pre-trained models can be used to solve new problems. In this case the crop disease classification problem can be solved by using Tensorflow's pre-trained models available at https://www.tensorflow.org/resources/models-datasets.
These models are nothing but convolutional neural networks trained on millions of images varying in multiple classes. Convolutional neural networks aka ConvNets can be visualised as having
1.)Input Layer
2.)Hidden Layer
3.)Output Layer
Input layer is where we input our dataset(with labels) and for each training example we keep modifying the parameters of hidden layer unless we get the correct corresponding output(or label) at the output layer. The modification of hidden layer is basically changing "weights" of the activation function present in each node.

Methodology

In this article, we use the Inception V3 model, 1st runner up for image classification in ILSVRC(ImageNet Large Scale Visual Recognition Competition).

This model was trained on 1.2 million training images, 50,000 validation images and 100,000 testing images. Another challenging aspect of deep learning is the hardware it requires to train models. Enter Google Colab (https://colab.research.google.com). Gone are the days where you would wait for days for your model to "bake". Google Colab is a FREE Juptyer notebook environment provided by Google with GPUs and TPUs.

Let's take the example of brinjal crop. We'll see the three major diseases in this article:

1. Sunburnt Brinjal:- 155 images
2. Phomopsis blight:- 487 images
3. Cercospora leaf spot:- 374 images

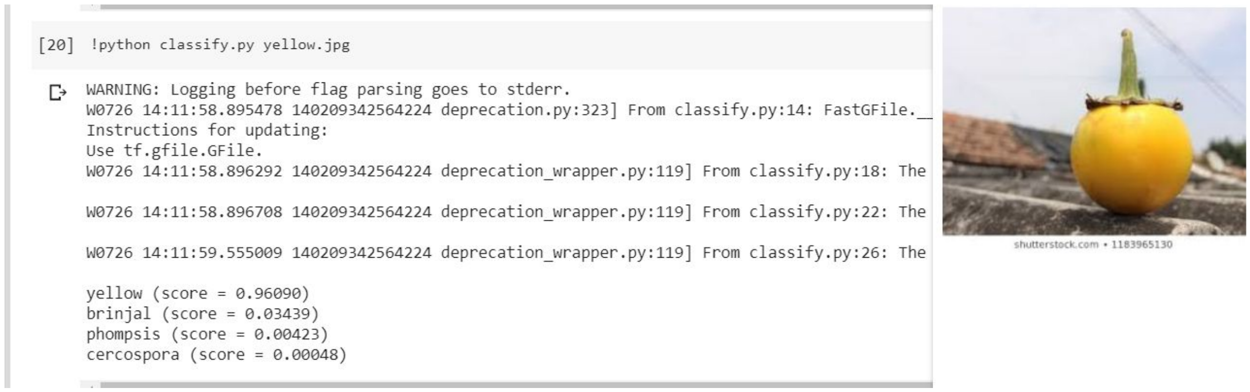
For the healthy brinjal crop dataset is collected by employing Web Scraping and 60-70 images are collected. Through Image Augmentation i.e, angular rotation, noise addition, horizontal and vertical flip, the dataset is increased for the model's requirements. This was necessary as dataset for the diseased brinjal types were scarce and not recorded anywhere. The model is trained for 2000 training steps . On Google Colab's GPU (Tesla K80) it takes around 5-10 minutes for the training to complete. Accuracy measure for each class is as follows:-

1) Healthy Brinjal:- 95.95%
2) Phomopsis blight :- 95.57%
3) Cercospora leaf spot :- 99.75%
4) Sunburnt Brinjal :- 96.09%

```
[22] !python classify.py brinjal.jpg

WARNING: Logging before flag parsing goes to stderr.
W0726 14:13:38.035212 140678739363712 deprecation.py:323] From classify.py:14: FastGFile.__
Instructions for updating:
Use tf.gfile.GFile.
W0726 14:13:38.036167 140678739363712 deprecation_wrapper.py:119] From classify.py:18: The

W0726 14:13:38.036561 140678739363712 deprecation_wrapper.py:119] From classify.py:22: The

W0726 14:13:38.575740 140678739363712 deprecation_wrapper.py:119] From classify.py:26: The

brinjal (score = 0.95958)
yellow (score = 0.03773)
phompsis (score = 0.00149)
cercospora (score = 0.00119)
```

```
[20] W0726 14:11:59.555009 140209342564224 deprecation_wrapper.py:119] From classify.py:26: The

yellow (score = 0.96090)
brinjal (score = 0.03439)
phompsis (score = 0.00423)
cercospora (score = 0.00048)

[21] !python classify.py cercospora.jpg

WARNING: Logging before flag parsing goes to stderr.
W0726 14:12:48.083767 140315051100032 deprecation.py:323] From classify.py:14: FastGFile.__
Instructions for updating:
Use tf.gfile.GFile.
W0726 14:12:48.084577 140315051100032 deprecation_wrapper.py:119] From classify.py:18: The

W0726 14:12:48.084960 140315051100032 deprecation_wrapper.py:119] From classify.py:22: The

W0726 14:12:48.619469 140315051100032 deprecation_wrapper.py:119] From classify.py:26: The

cercospora (score = 0.99757)
yellow (score = 0.00178)
brinjal (score = 0.00059)
phompsis (score = 0.00007)
```

cercospora.jpg

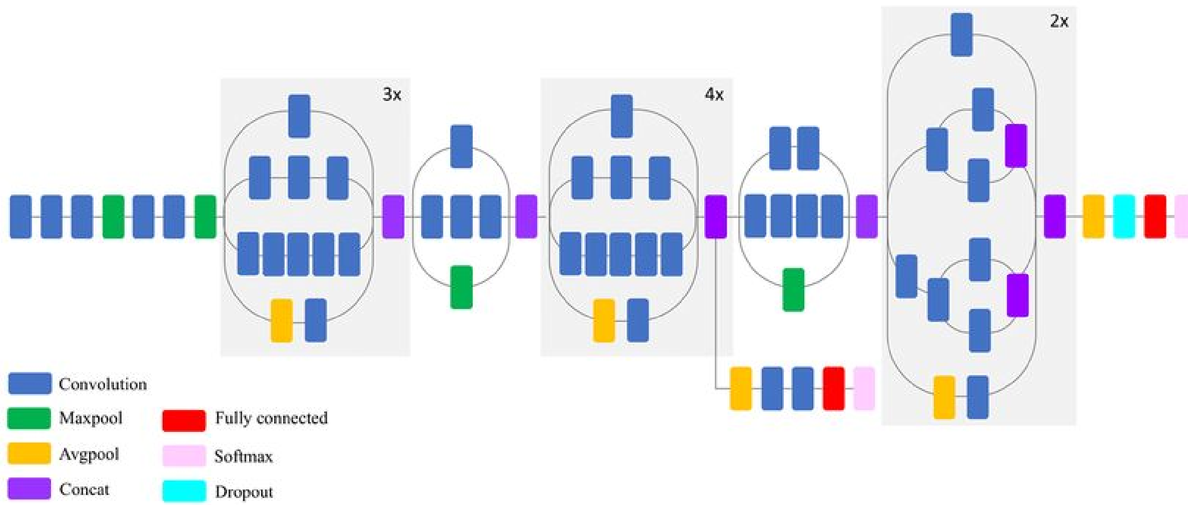


Tensorflow also provides libraries to deploy your model on various other devices like Android, iOS, Raspberry Pi etc. which is Tensorflow Lite. A mobile app can be devised to enable farmers to better understand the types of crop diseases; how to control and treat them.

Conclusion

Image classifiers while previously required tremendous hardware support and data can now easily be concocted and deployed by

1) ConvNets for multi-class image classification.



2) Tensorflow and Tensorflow Lite for a robust framework to deploy models to mobile devices.

3) Google Colab for training ML/AI models terminating the need of GPU and CPU intensive hardware and seamless cloud deployment.

4) Image Augmentation for dataset manipulation and multiplication.

Emerging technologies mentioned in this article enables developers to create more use-cases and implement them in real-time to solve problems and immensely benefit society without computational impediments.