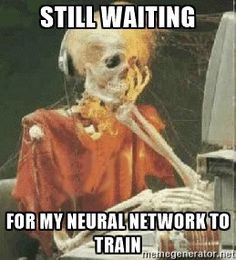
**LEAF DISEASE CLASSIFICATION USING DEEP LEARNING WITH API AND APPLICATION**

Written by [Jenish Patel](https://medium.com/u/ccaa2c073c25) and [Ujas](https://medium.com/u/b8704922144b) Patel

Photo by [Oliver Hihn](https://unsplash.com/@mr_kuchen?utm_source=medium&utm_medium=referral) on [Unsplash](https://unsplash.com?utm_source=medium&utm_medium=referral)Loading………

**Abstract:**

The early detection of diseases is important in agriculture for an efficient crop yield. The bacterial spot, late blight, septoria leaf spot and yellow curved leaf diseases affect the crop quality of tomatoes. Automatic methods for classification of plant diseases also help taking action after detecting the symptoms of leaf diseases.

Crop diseases are a major threat to food security, but their rapid identification remains difficult in many parts of the world due to the lack of the necessary infrastructure. The combination of increasing global smartphone penetration and recent advances in computer vision made possible by deep learning has paved the way for smartphone-assisted disease diagnosis. Using a public dataset of <https://www.kaggle.com/emmarex/plantdisease>: 20.6K images of diseased and healthy plant leaves collected under controlled conditions, we train a deep convolutional neural network to identify 14 crop species and 26 diseases (or absence thereof). The trained model achieves an accuracy of 99.35% on a held-out test set, demonstrating the feasibility of this approach. When testing the model on a set of images collected from trusted online sources — i.e. taken under conditions different from the images used for training — the model still achieves an accuracy of 31.4%. While this accuracy is much higher than the one based on random selection (2.6%), a more diverse set of training data is needed to improve the general accuracy. Overall, the approach of training deep learning models on increasingly large and publicly available image datasets presents a clear path towards smartphone-assisted crop disease diagnosis on a massive global scale.

**Introduction**

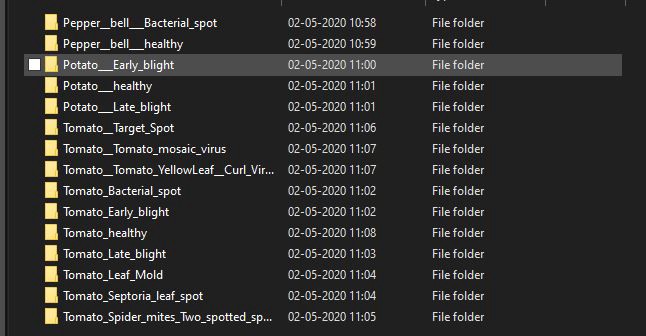
The problem of efficient plant disease protection is closely related to the problems of sustainable agriculture and climate change . Research results indicate that climate change can alter stages and rates of pathogen development; it can also modify host resistance, which leads to physiological changes of host-pathogen interactions . The situation is further complicated by the fact that, today, diseases are transferred globally more easily than ever before. New diseases can occur in places where they were previously unidentified and, inherently, where there is no local expertise to combat them .

Inexperienced pesticide usage can cause the development of long-term resistance of the pathogens, severely reducing the ability to fight back. Timely and accurate diagnosis of plant diseases is one of the pillars of precision agriculture. It is crucial to prevent unnecessary waste of financial and other resources, thus achieving healthier production, by addressing the long-term pathogen resistance development problem and mitigating the negative effects of climate change.

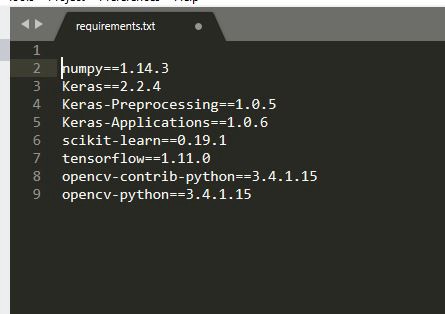
**Materials and Methods**

The entire procedure of developing the model for plant disease recognition using deep CNN is described further in detail. The complete process is divided into several necessary stages in subsections below, starting with gathering images for classification process using deep neural networks.

Here is what my dataset PlantVillage/ file structure looks like.



If you won’t be using Kaggle Kernel you will have to install this packages



then you must satisfy the versions of different packages that we are going to use initially i have tried doing on kaggle then i moved to jupyter notebook and it had taken a lot time for me to train the data and created the desried model .

1. [Numpy](http://www.numpy.org/) : a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. (Source : [Wikipedia](https://en.wikipedia.org/wiki/Scikit-learn) )
2. [Sklearn](https://scikit-learn.org/stable/) : a [free software](https://en.wikipedia.org/wiki/Free_software) [machine learning](https://en.wikipedia.org/wiki/Machine_learning) [library](https://en.wikipedia.org/wiki/Library_%28computing%29) for the [Python](https://en.wikipedia.org/wiki/Python_%28programming_language%29) programming language. It features various [classification](https://en.wikipedia.org/wiki/Statistical_classification), [regression](https://en.wikipedia.org/wiki/Regression_analysis) and [clustering](https://en.wikipedia.org/wiki/Cluster_analysis) algorithms including [support vector machines](https://en.wikipedia.org/wiki/Support_vector_machine), [random forests](https://en.wikipedia.org/wiki/Random_forests), [gradient boosting](https://en.wikipedia.org/wiki/Gradient_boosting), [*k*-means](https://en.wikipedia.org/wiki/K-means_clustering) and [DBSCAN](https://en.wikipedia.org/wiki/DBSCAN), and is designed to interoperate with the Python numerical and scientific libraries [NumPy](https://en.wikipedia.org/wiki/NumPy) and [SciPy](https://en.wikipedia.org/wiki/SciPy). (Source : [Wikipedia](https://en.wikipedia.org/wiki/Scikit-learn) )
3. [Keras](https://keras.io/) : Keras is an open source neural network library written in Python. It is capable of running on top of TensorFlow, Microsoft Cognitive Toolkit, or Theano. Designed to enable fast experimentation with deep neural networks, it focuses on being user-friendly, modular, and extensible. (Source : [Wikipedia](https://en.wikipedia.org/wiki/Scikit-learn) )
4. [Matplotlib](https://matplotlib.org/) : a plotting library for the Python programming language and its numerical mathematics extension.

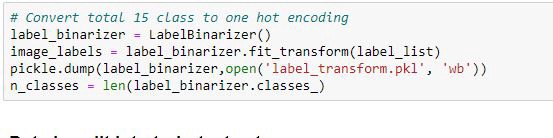
Next thing I did was to load the dataset — images of diseased plants



From the image above  
A — I picked just 700 images from each folder but you can choose to add more  
B — I converted each image to an array using the function below (left). After converting each image to an array using this same function, you should have something similar to what i have in the image below (right) for each image.



Using [Scikit-learn’s Label Binarizer](http://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.LabelBinarizer.html) , I converted each image label to binary levels. Then I saved the label binarizer instance using [pickle](https://docs.python.org/3/library/pickle.html) after which I printed the classes from the label binarizer.

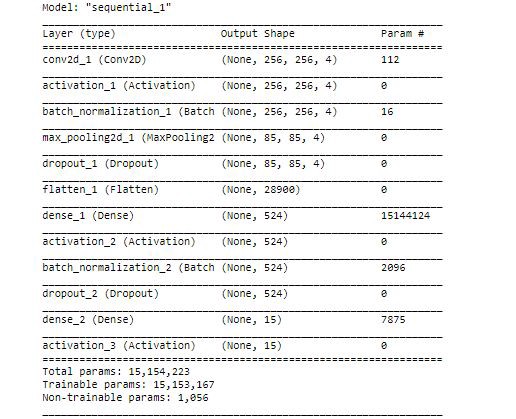


In (in the image below) I further pre-process the input data by scaling the data points from [0, 255] (the minimum and maximum RGB values of the image) to the range [0, 1]. I then performed a training/testing split on the data using 80% of the images for training and 20% for testing. Icreated an image generator object which performs random rotations, shifts, flips, crops, and sheers on our image dataset. This allows us to use a smaller dataset and still achieve high results.

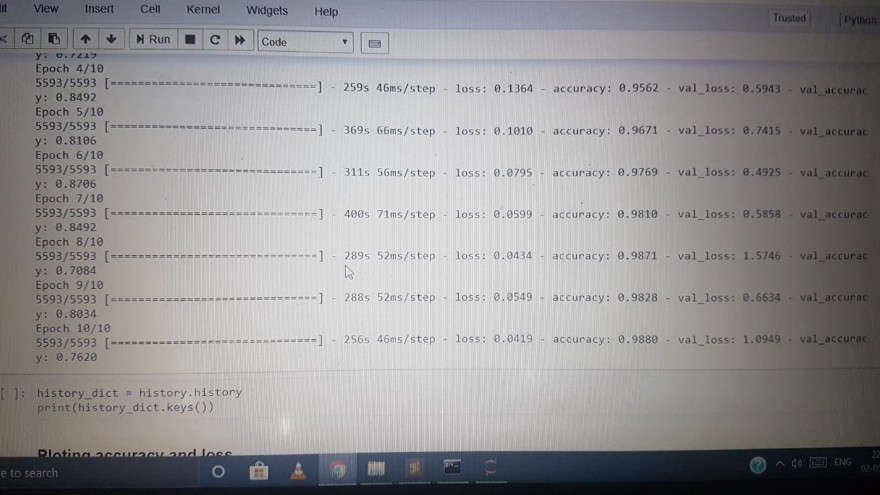


Next I created my model. Then I created the first **CONV** => **RELU** => **POOL**. Our **CONV** layer has 32 filters with a 3 x 3 kernel and **RELU** activation (Rectified Linear Unit). We apply batch normalization, max pooling, and 25% (0.25) dropout.

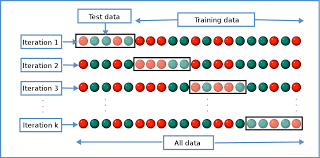
***Dropout*** *is a regularization technique for reducing overfitting in neural networks by preventing complex co-adaptations on training data. It is a very efficient way of performing model averaging with neural networks. (Source :* [*Wikipedia*](https://en.wikipedia.org/wiki/Dropout_%28neural_networks%29) *)*

This is how our model looks before we are going to train

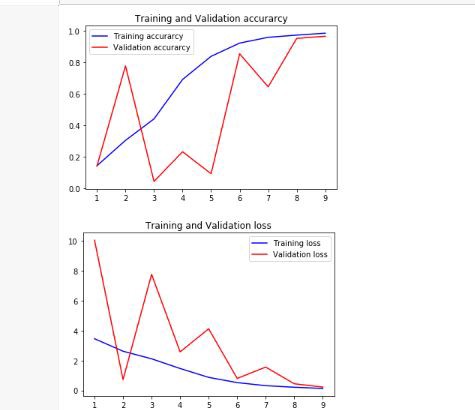
I used the [Keras Adam Optimizer](https://keras.io/optimizers/#adam) for my model. Training our network is initiated in where we call **model.fit\_generator** , supplying our data augmentation object, training/testing data, and the number of epochs we wish to train for. I used an **epochs value of 10** for this project.

It had took me long to train this model….

Evalution is main part of Neural Network this is the place where we get to know that whether our model is underfitting ,overfitting,and from here we are going to have a check on numbers of epochs,finding flaws and tuning hyperparameters well like you can do resampling of samples of splitting dataset using Cross-validation where you can get train,test and validate part that mostly corrects overfitting problems.Look at the image (below)



**~Training and Testing Accuracy and Error**



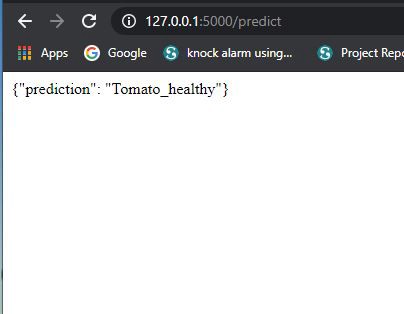
Save our model and as you know we are going to use our model again in api and I have also deployed this model using Flask Api.

Flask which is a Micro web framework for all those python geeks who want to learn some web development and python .

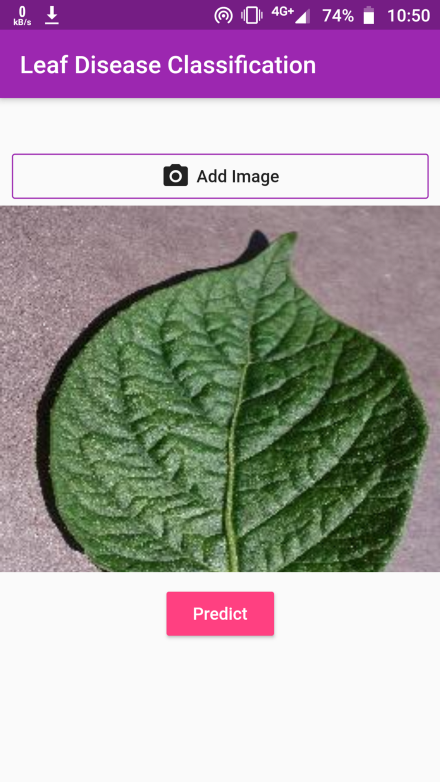
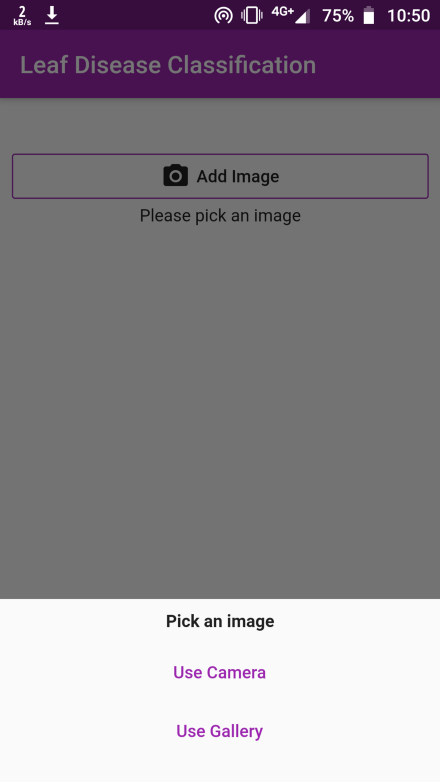
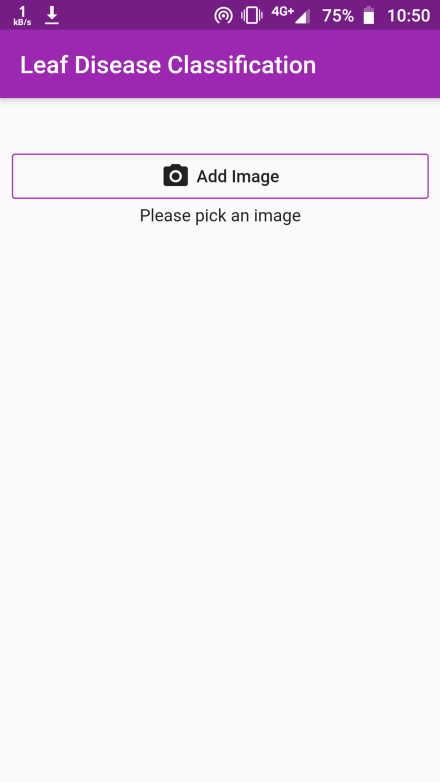
But what good is a Model that can’t be used? So in this Post I will talk about how I deployed this model for use with an API in web browser and later we also have created app using Flutter.

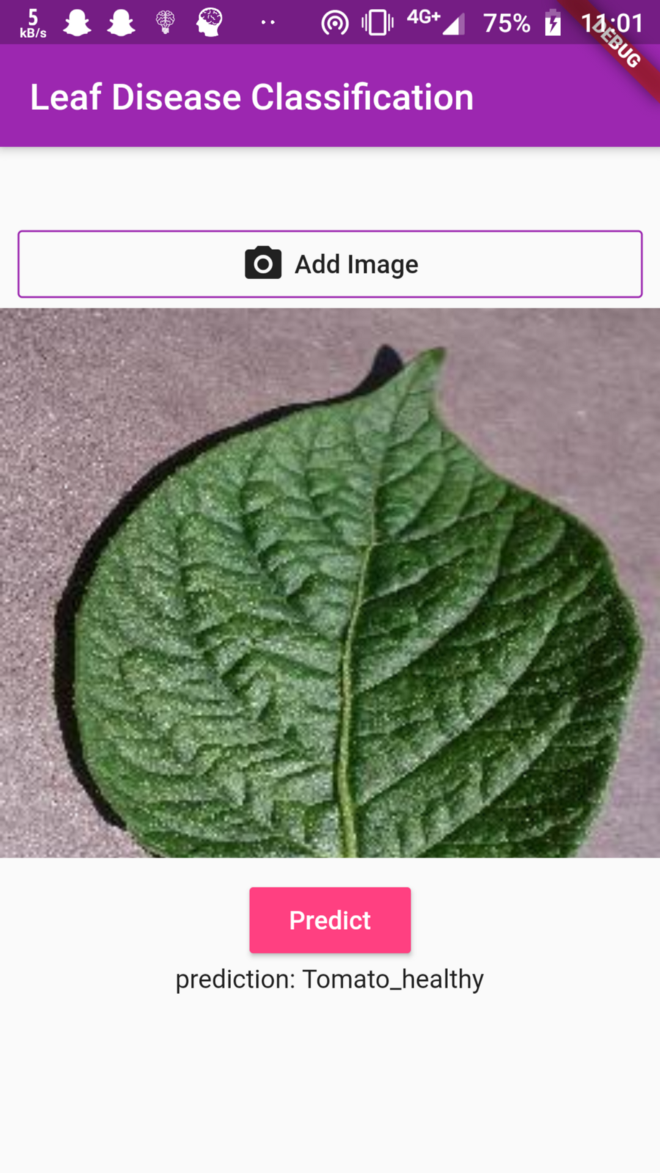
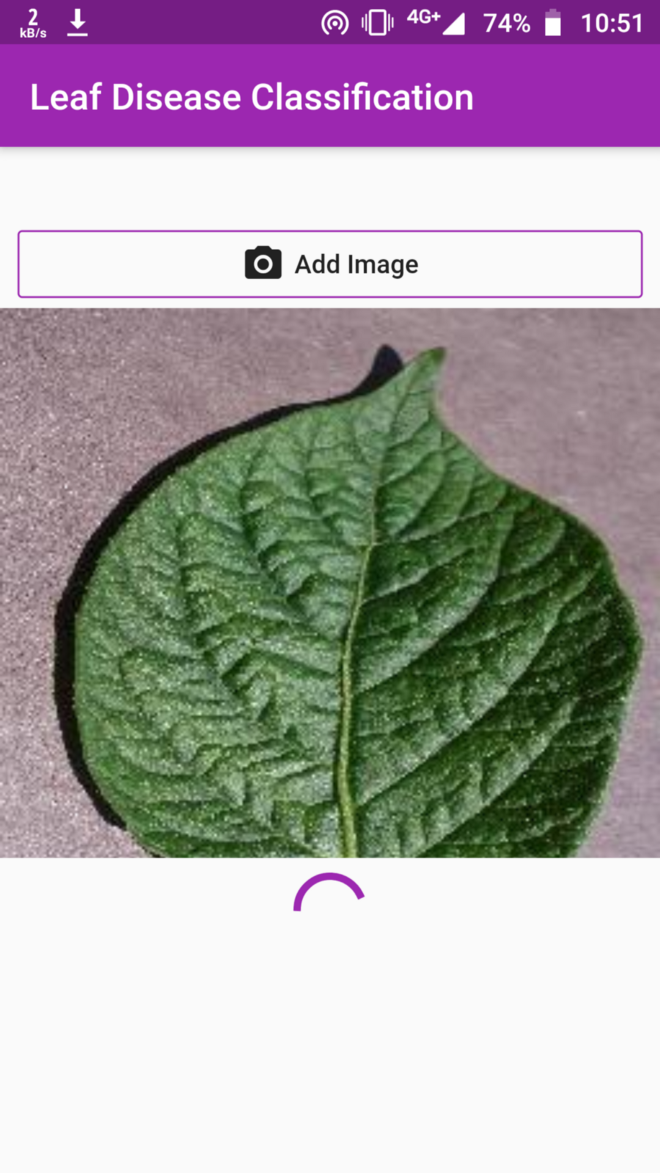
To deploy our trained model for use via an API, we would do something similar to the following :

* Load our trained model
* Accept incoming data and preprocess it
* Predict using our loaded model
* Handling the prediction output.



For Application we have created it using Flutter…..





**Conclusion**

There are many methods in automated or computer vision plant disease detection and classification process, but still, this research field is lacking. In addition, there are still no commercial solutions on the market, except those dealing with plant species recognition based on the leaves images.

Many new approach of using deep learning method was explored in order to automatically classify and detect plant diseases from leaf images. The developed model was able to detect leaf presence and distinguish between healthy leaves and 13 different diseases, which can be visually diagnosed. The complete procedure was described, respectively, from collecting the images used for training and validation to image preprocessing and augmentation and finally the procedure of training the deep CNN and fine-tuning. Different tests were performed in order to check the performance of newly created model.

New plant disease image database was created, containing more than 3,000 original images taken from the available Internet sources and extended to more than 30,000 using appropriate transformations. The experimental results achieved precision between 91% and 98%, for separate class tests. The final overall accuracy of the trained model was 96.3%. Fine-tuning has not shown significant changes in the overall accuracy, but augmentation process had greater influence to achieve respectable results.

As the presented method has not been exploited, as far as we know, in the field of plant disease recognition, there was no comparison with related results, using the exact technique. In comparison with other techniques used and presented comparable or even better results were achieved, especially when taking into account the wider number of classes in the presented study.

An extension of this study will be on gathering images for enriching the database and improving accuracy of the model using different techniques of fine-tuning and augmentation.

The main goal for the future work will be developing a complete system consisting of server side components containing a trained model and an application for smart mobile devices with features such as displaying recognized diseases in fruits, vegetables, and other plants, based on leaf images captured by the mobile phone camera. This application will serve as an aid to farmers (regardless of the level of experience), enabling fast and efficient recognition of plant diseases and facilitating the decision-making process when it comes to the use of chemical pesticides.

Furthermore, future work will involve spreading the usage of the model by training it for plant disease recognition on wider land areas, combining aerial photos of orchards and vineyards captured by drones and convolution neural networks for object detection. By extending this research, the authors hope to achieve a valuable impact on sustainable development, affecting crop quality for future generations.

Alexnet and Googlenet are best model architectures we can more dive if we want to explore in CNN model architectures.

What we have learnt from this project is how deep learning plays a great role in classification of real life images and how variety of disease is detected by model as it had learned from dataset data which needs to be well and real without errors.

Because it depends a lot on origin of dataset and if thats good then data pre-processing part and splitting part and accuracy is going to depend on many factors too.

Deep learning is therefore booming in medical science in discovery of new drugs ,and its helping many more technolgies too ,The Future of Deep Learning is going to be very rich as it going to step in I/O embedded System with deep learning where real data is going to sent from different devices and decisions they need to take is going to decided by parallel running of incoming signal data from devices ingesting in cloud and where high parallel processing takes place and data is going to have Stream batch Processing as we need to work on Real Time Data.

Written by [Jenish Patel](https://medium.com/u/ccaa2c073c25) and [Ujas](https://medium.com/u/b8704922144b) Patel

Linkedin- [Jenish Patel](http://www.linkedin.com/in/jenish-patel-46479116a) and [Ujas Patel](https://www.linkedin.com/in/ujas3279/)

Github-<https://github.com/ujas3279/Leaf-Dieseas-Classification>

Email- jenishp619@gmail.com and ujas6520@gmail.com