Fertilizers Recommendation System for Disease Prediction

PROJECT REPORT

Submitted by

Team ID: PNT2022TMID01892

M. ASFAQ AHAMED (7376201EC503)

V. SUDARSHAN (7376201EC519)

DG. HENRY BRUNO (7376201EC509)

V. THIBESH (7376201EC522)

1. INTRODUCTION

● Agriculture is the most important sector in today’s life. Most plants are affected by a wide variety of bacterial and fungal diseases. Diseases on plants placed a major constraint on the production and a major threat to food security. Hence, early and accurate identification of plant diseases is essential to ensure high quantity and best quality. In recent years, the number of diseases on plants and the degree of harm caused has increased due to the variation in pathogen varieties, changes in cultivation methods, and inadequate plant protection techniques.

2. LITERATURE SURVEY

2.1 Existing problem Indumathi proposed a method for leaf disease detection and suggest fertilizers to cure leaf diseases. But the method involves less number of train and test sets which results in poor accuracy. Pandi selvi proposed a simple prediction method for soil-based fertilizer recommendation system for predicted crop diseases. This method gives less accuracy and prediction. Shiva reddy proposed an IoT based system for leaf disease detection and fertilizer recommendation which is based on Machine Learning techniques yields less 80 percentage accuracies.

2.2 Proposed solution In this project work, a deep learning based neural network is used to train the collected datasets and test the same. The deep learning based neural network is CNN which gives more than 90% classification accuracies. By increasing the more number of dense layers and by modifying hyperparameters such as number of epochs, batch size, the accuracy rate can be increased to 95% to 98%.

Existing Problem

- Adequate mineral nutrition is central to crop production. However, it can also exert considerable Influence on disease development. Fertilizer application can increase or decrease development of diseases caused by different pathogens, and the mechanisms responsible are complex, including effects of nutrients on plant growth, plant resistance mechanisms and direct effects on the pathogen. The effects of mineral nutrition on plant disease and the mechanisms responsible for those effects have been dealt with comprehensively elsewhere. In India, around 40% of land is kept and grown using reliable irrigation technologies, while the rest relies on the monsoon environment for water. Irrigation decreases reliance on the monsoon, increases food security, and boosts agricultural production.

- Most research articles use humidity, moisture, and temperature sensors near the plant's root, with an external device handling all of the data provided by the sensors and transmitting it directly to an Android application. It was created to measure the approximate values of temperature, humidity and moisture sensors that were programmed into a microcontroller to manage the amount of water.

3. THEORITICAL ANALYSIS

3.1 Block diagram

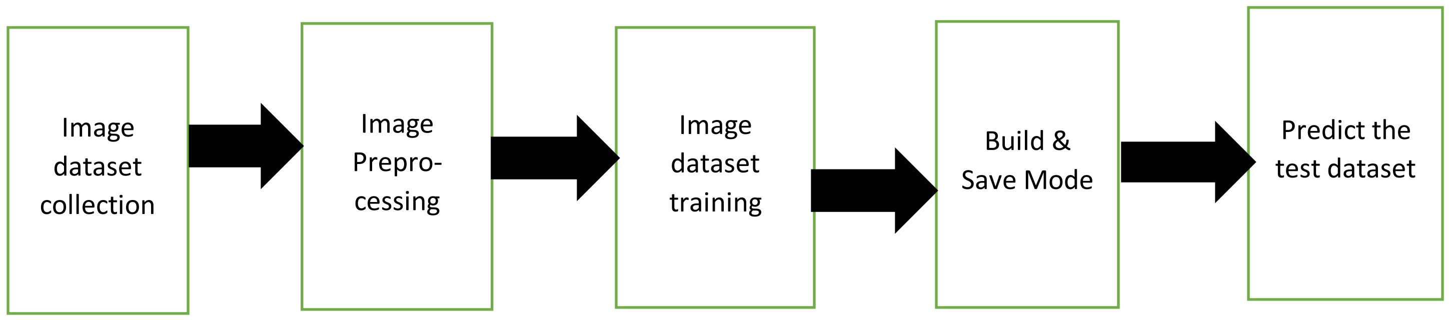


Figure.3.1. Block Diagram of the project

The block diagram of the entire project is shown in Fig.3.1. First step is the image dataset collection followed by image preprocessing. The third step is the training of image datasets with initializing different hyper parameters. Then build the model and save the model file with .h5 format. The final stage is the testing of existing or new datasets using the trained model.

3.2 Hardware/Software designing

The software used for training and testing the dataset is Python. The Jupyter notebook (Notebook of IBM cloud also) is used for python programming. The neural network used for training and testing the model is Convolutional Neural Network (CNN).

The CNN has following layers:

- Convolutional layer (32x32 kernal (3x3))
- Max-pool layer (kernel(2x2))

● Flatten layer

● Dense layer (different layers with different size)

● Drop out layer (optional)

● Final output dense layer(size 6x1 for fruit dataset and 9x1 for Vegetable dataset)

In the preprocessing step, images are normalized to 1 and then resized to 128x128. The images are arranged in different batch sizes. Then train set and test set are formed from the collected datasets. In order to do the above steps in Python, the following Python libraries must be imported before starting the process:

● NumPy

● TensorFlow

● Keras

● Matplotlib (optional for data visualization)

The following activation functions used in the CNN training:

● RELU at the end of convolution layer and Max Pool layer

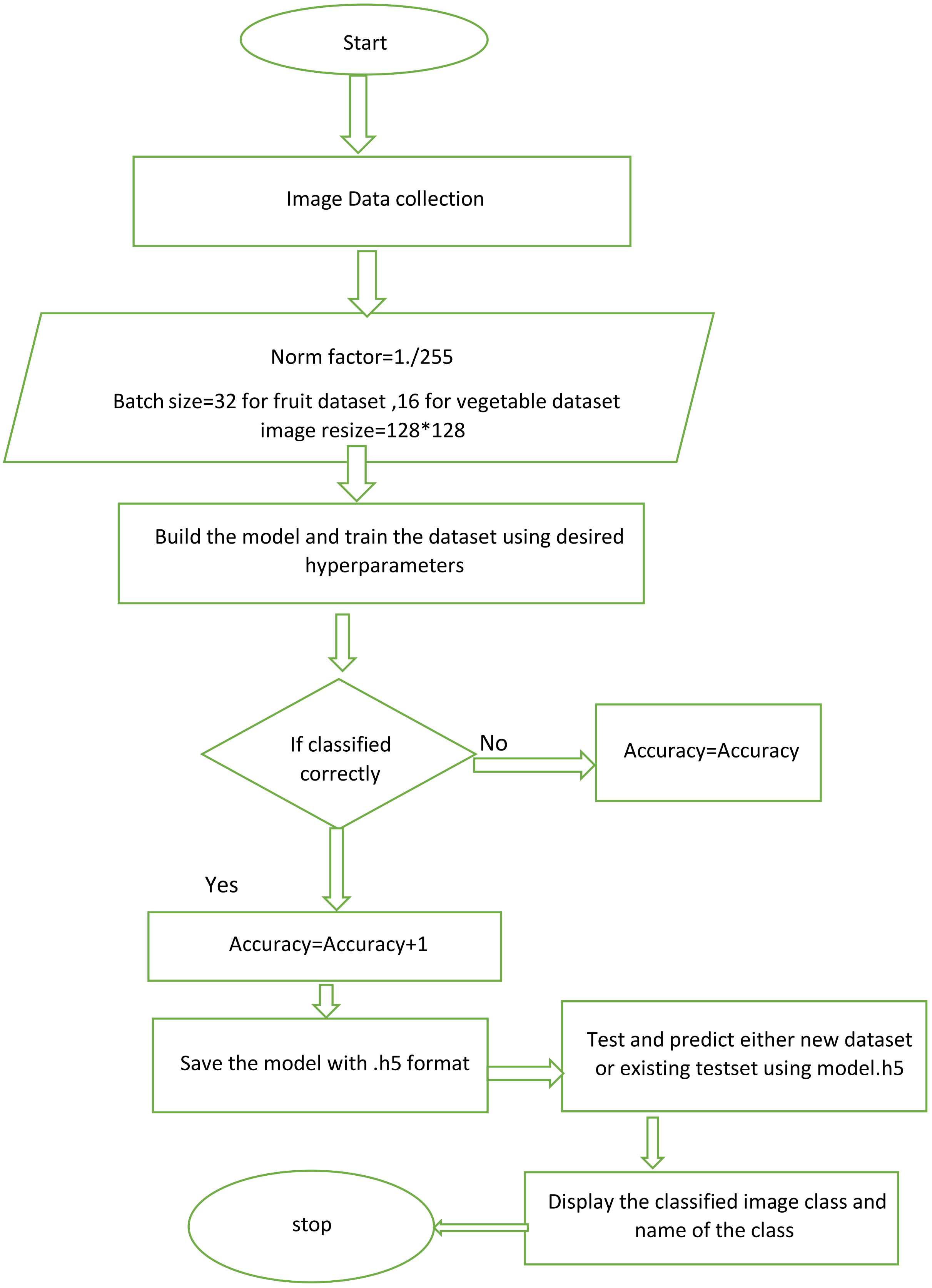
● SoftMax at the end of output dense layer

● For testing the dataset argmax is used, its an optional

4. EXPERIMENTAL INVESTIGATIONS

Analysis made while working on the solution The batch sizes are varied and tested. For different batch sizes, the CNN gives different accuracies. The batch size determines the number of iterations per epoch. Another important hyper parameter is the number of epochs. This determines accuracy and it has high influence on accuracy compared to other hyper parameters. The accuracy can be varied from 80% to 90% in vegetable dataset and 95% to 98% in the case of fruit dataset by increasing the number of epochs. The size of test dataset and train dataset also has very high influence on accuracies. The accuracy can be increased by using more number of images in train dataset. The computational time for model building is increased when the size of the train dataset increased and also number of epochs increased. The batch size of train dataset and test dataset also play a vital role in computational time. The Neural Network complexity is increased when more number of convolutional layers increased. If the number of layers increased, better accuracy result will obtain. At the same increasing the number of layers in CNN leads to more training time and also requires more time to build a model. The model .h5 size depends on the size of train datasets. But the memory requirement depends on the size of train dataset and CNN architecture complexity.

5. FLOWCHARTS

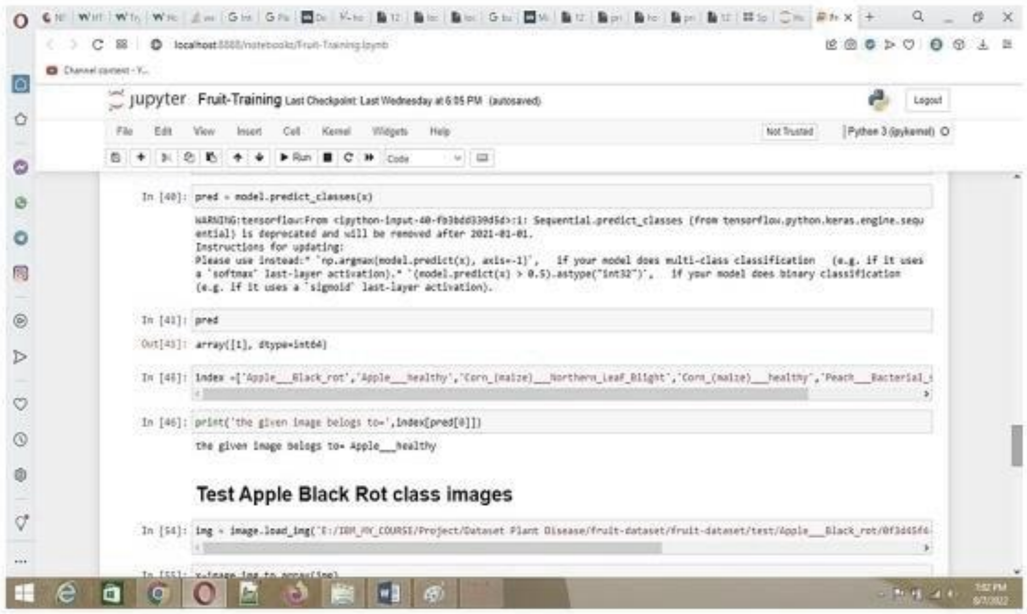


Start
Image Data collection
Norm factor=1./255
Batch size=32 for fruit dataset ,16 for vegetable dataset
image resize=128*128
Build the model and train the dataset using desired hyperparameters
If classified correctly
No
Accuracy=Accuracy
Yes
Accuracy=Accuracy+1
Save the model with .h5 format
Test and predict either new dataset or existing testset using model.h5
Display the classified image class and name of the class
stop

6. RESULTS

Final output of the project given below in the form of screenshot:

Training and



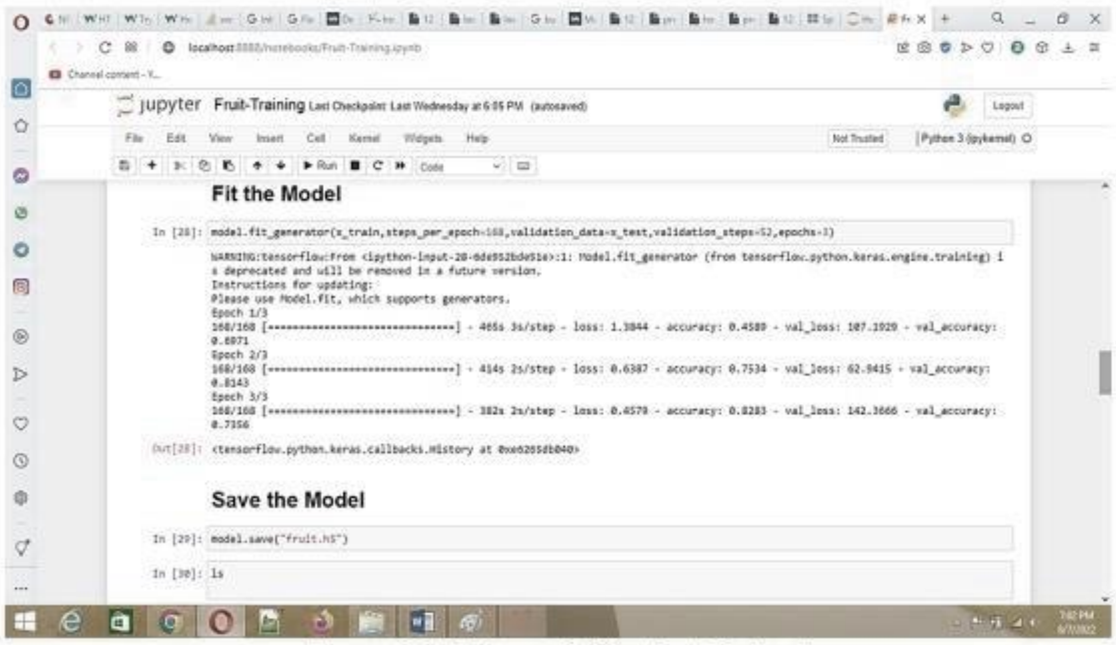
jupyter Fruit-Training Last Checkpoint: Last Wednesday at 6:05 PM (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code

```
In [40]: pred = model.predict_classes(x)

         WARNING:tensorflow:From <ipython-input-40-fb3bdd339d5d>:1: Sequential.predict_classes (from tensorflow.python.keras.engine.sequ
         ential) is deprecated and will be removed after 2021-01-01.
         Instructions for updating:
         Please use instead:* `np.argmax(model.predict(x), axis=-1)`,   if your model does multi-class classification   (e.g. if it uses
         a `softmax` last-layer activation).* `(model.predict(x) > 0.5).astype("int32")`,   if your model does binary classification
         (e.g. if it uses a `sigmoid` last-layer activation).

In [41]: pred

Out[41]: array([1], dtype=int64)

In [46]: index =['Apple___Black_rot','Apple___healthy','Corn_(maize)___Northern_Leaf_Blight','Corn_(maize)___healthy','Peach___Bacterial_

In [46]: print('the given image belogs to=',index[pred[0]])

         the given image belogs to= Apple___healthy
```

Test Apple Black Rot class images

```
In [54]: img = image.load_img('E:/IBM_MY_COURSE/Project/Dataset Plant Disease/fruit-dataset/fruit-dataset/test/Apple___Black_rot/0f3d45f6
```

Figure.6.2 Test the Fruit dataset



jupyter Fruit-Training Last Checkpoint: Last Wednesday at 6:05 PM (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code

Fit the Model

```
In [28]: model.fit_generator(x_train,steps_per_epoch=168,validation_data=x_test,validation_steps=52,epochs=3)

         WARNING:tensorflow:From <ipython-input-28-6de952bde51e>:1: Model.fit_generator (from tensorflow.python.keras.engine.training) i
         s deprecated and will be removed in a future version.
         Instructions for updating:
         Please use Model.fit, which supports generators.
         Epoch 1/3
         168/168 [==============================] - 465s 3s/step - loss: 1.3844 - accuracy: 0.4589 - val_loss: 107.1929 - val_accuracy:
         0.6071
         Epoch 2/3
         168/168 [==============================] - 414s 2s/step - loss: 0.6387 - accuracy: 0.7534 - val_loss: 62.9415 - val_accuracy:
         0.8143
         Epoch 3/3
         168/168 [==============================] - 382s 2s/step - loss: 0.4579 - accuracy: 0.8283 - val_loss: 142.3666 - val_accuracy:
         0.7356

Out[28]: <tensorflow.python.keras.callbacks.History at 0xe6285db040>
```

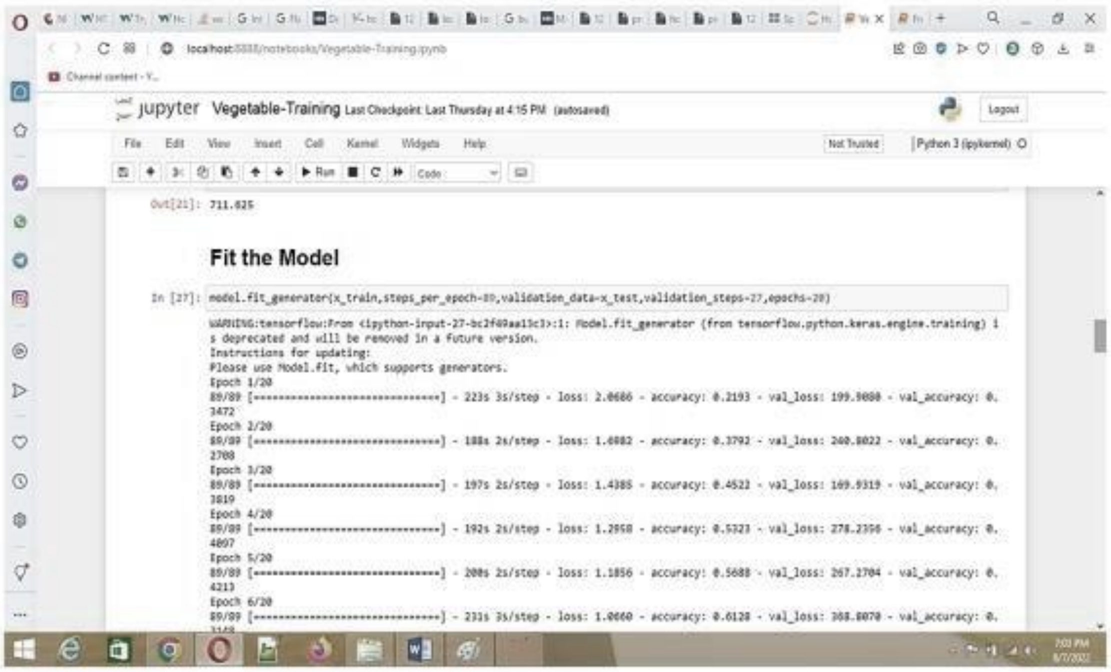
Save the Model

```
In [29]: model.save("fruit.h5")

In [30]: ls
```

Figure.6.1. Fit a model for Fruit dataset

Testing of Fruit dataset

Train and Test Vegetable dataset



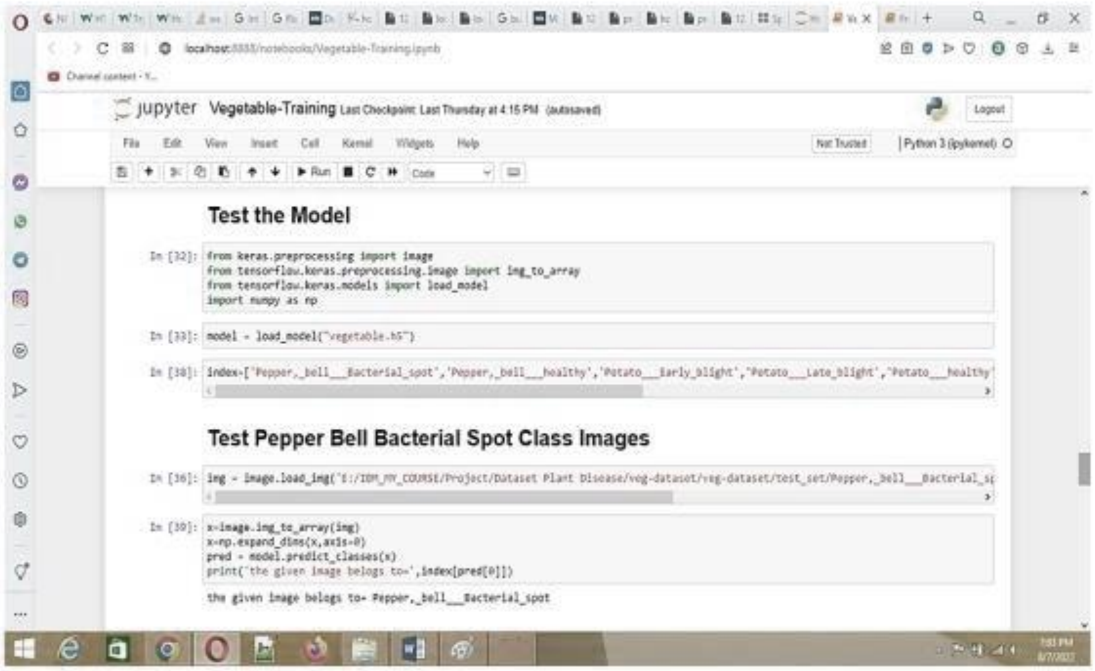
jupyter Vegetable-Training Last Checkpoint: Last Thursday at 4:15 PM (autosaved)

```
Out[21]: 711.625
```

Fit the Model

```
In [27]: model.fit_generator(x_train,steps_per_epoch=89,validation_data=x_test,validation_steps=27,epochs=20)

WARNING:tensorflow:From <ipython-input-27-bc2f49aa13c3>:1: Model.fit_generator (from tensorflow.python.keras.engine.training) i
s deprecated and will be removed in a future version.
Instructions for updating:
Please use Model.fit, which supports generators.
Epoch 1/20
89/89 [==============================] - 223s 3s/step - loss: 2.0686 - accuracy: 0.2193 - val_loss: 199.9080 - val_accuracy: 0.
3472
Epoch 2/20
89/89 [==============================] - 188s 2s/step - loss: 1.6882 - accuracy: 0.3792 - val_loss: 240.8022 - val_accuracy: 0.
2708
Epoch 3/20
89/89 [==============================] - 197s 2s/step - loss: 1.4385 - accuracy: 0.4522 - val_loss: 169.9319 - val_accuracy: 0.
3819
Epoch 4/20
89/89 [==============================] - 192s 2s/step - loss: 1.2958 - accuracy: 0.5323 - val_loss: 278.2396 - val_accuracy: 0.
4097
Epoch 5/20
89/89 [==============================] - 200s 2s/step - loss: 1.1856 - accuracy: 0.5688 - val_loss: 267.2704 - val_accuracy: 0.
4213
Epoch 6/20
89/89 [==============================] - 231s 3s/step - loss: 1.0660 - accuracy: 0.6128 - val_loss: 388.8070 - val_accuracy: 0.
```

Figure.6.3. Train the Vegetable dataset



jupyter Vegetable-Training Last Checkpoint: Last Thursday at 4:15 PM (autosaved)

Test the Model

```
In [32]: from keras.preprocessing import image
         from tensorflow.keras.preprocessing.image import img_to_array
         from tensorflow.keras.models import load_model
         import numpy as np

In [33]: model = load_model("vegetable.h5")

In [38]: index=['Pepper,_bell___Bacterial_spot','Pepper,_bell___healthy','Potato___Early_blight','Potato___Late_blight','Potato___healthy'
```

Test Pepper Bell Bacterial Spot Class Images

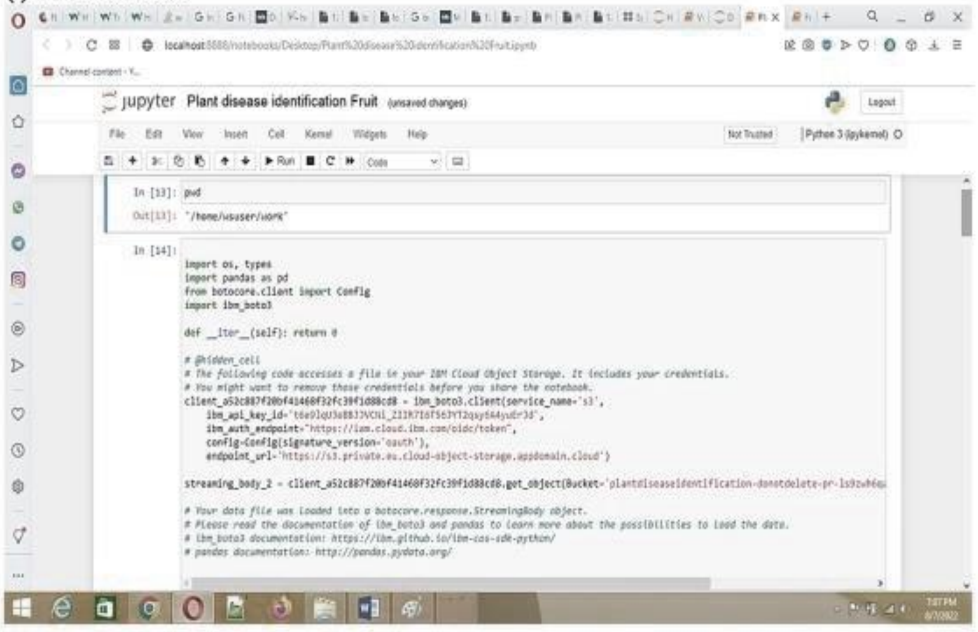
```
In [36]: img = image.load_img('E:/IBM_MY_COURSE/Project/Dataset Plant Disease/veg-dataset/veg-dataset/test_set/Pepper,_bell___Bacterial_sp

In [39]: x=image.img_to_array(img)
         x=np.expand_dims(x,axis=0)
         pred = model.predict_classes(x)
         print('the given image belogs to=',index[pred[0]])

the given image belogs to= Pepper,_bell___Bacterial_spot
```

Figure.6.4. Test the Vegetable dataset

Train and Test Vegetable dataset IBM Cloud

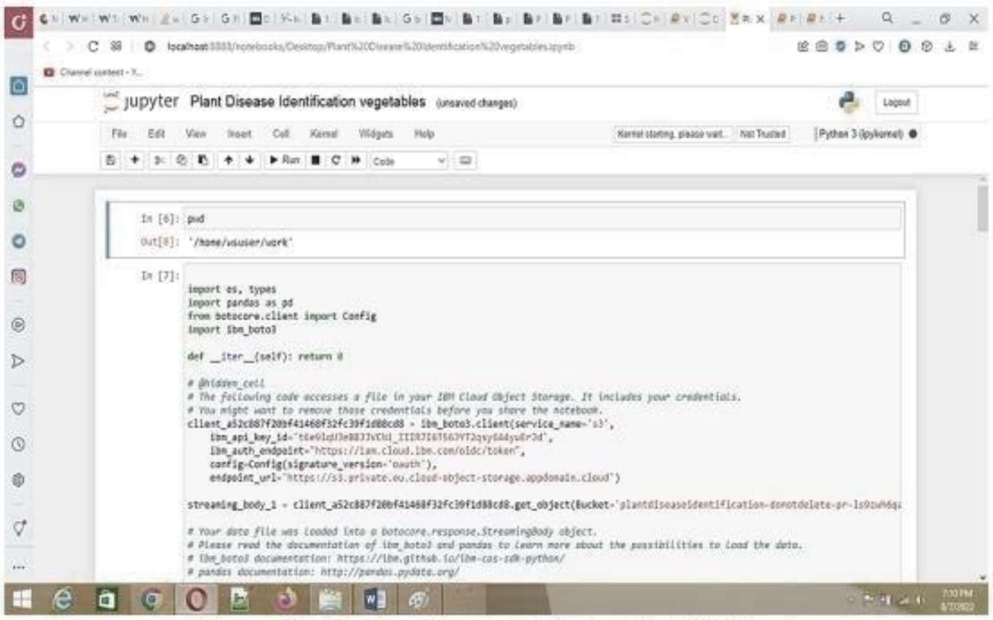
Due to CUH limit exceeds, I have downloaded the notebooks and opened in Jupyter notebook

(i). Fruit dataset


jupyter Plant disease identification Fruit (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code

```
In [13]: pwd
Out[13]: '/home/wsuser/work'

In [14]:
import os, types
import pandas as pd
from botocore.client import Config
import ibm_boto3

def __iter__(self): return 0

# @hidden_cell
# The following code accesses a file in your IBM Cloud Object Storage. It includes your credentials.
# You might want to remove those credentials before you share the notebook.
client_a52c887f20bf41460f32fc39f1d88cd8 = ibm_boto3.client(service_name='s3',
    ibm_api_key_id='t6e9lqU3e8BJJVCNl_2IIR7I6T563YT2qsy644yuEr3d',
    ibm_auth_endpoint="https://iam.cloud.ibm.com/oidc/token",
    config=Config(signature_version='oauth'),
    endpoint_url='https://s3.private.eu.cloud-object-storage.appdomain.cloud')

streaming_body_2 = client_a52c887f20bf41460f32fc39f1d88cd8.get_object(Bucket='plantdiseaseidentification-donotdelete-pr-1s9zwh6q

# Your data file was loaded into a botocore.response.StreamingBody object.
# Please read the documentation of ibm_boto3 and pandas to learn more about the possibilities to load the data.
# ibm_boto3 documentation: https://ibm.github.io/ibm-cos-sdk-python/
# pandas documentation: http://pandas.pydata.org/
```

Figure.6.5. Training Fruit Dataset in IBM Cloud


jupyter Plant Disease Identification vegetables (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Kernel starting, please wait...
Not Trusted
Python 3 (ipykernel)
Run
Code

```
In [6]: pwd
Out[6]: '/home/wsuser/work'

In [7]:
import os, types
import pandas as pd
from botocore.client import Config
import ibm_boto3

def __iter__(self): return 0

# @hidden_cell
# The following code accesses a file in your IBM Cloud Object Storage. It includes your credentials.
# You might want to remove those credentials before you share the notebook.
client_a52c887f20bf41460f32fc39f1d88cd8 = ibm_boto3.client(service_name='s3',
    ibm_api_key_id='t6e9lqU3e8BJJVCNl_IIIRJI6T563YT2qsy644yuEr3d',
    ibm_auth_endpoint="https://iam.cloud.ibm.com/oidc/token",
    config=Config(signature_version='oauth'),
    endpoint_url='https://s3.private.eu.cloud-object-storage.appdomain.cloud')

streaming_body_1 = client_a52c887f20bf41460f32fc39f1d88cd8.get_object(Bucket='plantdiseaseidentification-donotdelete-pr-1s9zwh6q

# Your data file was loaded into a botocore.response.StreamingBody object.
# Please read the documentation of ibm_boto3 and pandas to learn more about the possibilities to load the data.
# ibm_boto3 documentation: https://ibm.github.io/ibm-cos-sdk-python/
# pandas documentation: http://pandas.pydata.org/
```

Figure.6.6. Training Vegetable Dataset in IBM Cloud

7. ADVANTAGES & DISADVANTAGES

List of advantages

➤ The proposed model could predict the disease just from the image of a part icular plant.

➤ Easy to use UI.

➤ Model has some good accuracy in detecting the plant just by taking the input(leaf).

➤ These kind of web applications can be used in the agricultural sector as well as for small house hold plants as well.

Disadvantages:

➤ Prediction is limited to few plants as we havent trained all the plants.

8. APPLICATIONS

1. The trained network model used to classify the image patterns with high accuracy.
2. The proposed model not only used for plant disease classification but also for other image pattern classification such as animal classification.
3. This project work application involves not only image classification but also for pattern recognition.

9. CONCLUSIONS

The model proposed here involves image classification of fruit datasets and vegetable datasets. The following points are observed during model testing and training:

● The accuracy of classification increased by increasing the number of epochs.

● For different batch sizes, different classification accuracies are obtained.

● The accuracies are increased by increasing more convolution layers.

● The accuracy of classification also increased by varying dense layers.

● Different accuracies are obtained by varying the size of kernel used in the convolution layer output.

● Accuracies are different while varying the size of the train and test datasets.

10. FUTURE SCOPE

• As of now we have just built the web application which apparently takes the input as an image and then predict the out in the near future we can develop an application which computer vision and AI techniques to predict the infection once you keep the camera near the plant or leaf this could make our project even more usable

• This further research is implementing the proposed algorithm with the existing public datasets. Also, various segmentation algorithms can be implemented to improve accuracy. The proposed algorithm can be modified further to identify the disease that affects the various plant organs such as vegetables and fruits.

11. BIBILOGRAPHY

[1]. R Indumathi Leaf Disease Detection and Fertilizer Suggestion", IEEE International Conference on System, Computation, Automation and Networking (ICSCAN), 29-30 March 2019, DOI: 10.1109/ICSCAN.2019.8878781.

[2]. P. Pandi Selvi, P. Poornima, "Soil Based Fertilizer Recommendation System for Crop Disease Prediction System", International Journal of Engineering Trends and Applications (IJETA) – Volume 8 Issue 2, Mar-Apr 2021.

[3]. H Shiva reddy, Ganesh hedge, Prof. DR Chinnaya3, "IoT based Leaf Disease Detection and Fertilizer Recommendation", International Research Journal of Engineering and Technology (IRJET), Volume: 06 Issue: 11, Nov 2019, e-ISSN: 2395- 0056.

APPENDIX

A. Source Code (Jupyter notebook python code) fruit.ipynb (due to limited page size the code vegetable.ipynb uploaded in github)

#!/usr/bin/env python

# coding: utf-8

# In[1]: pwd

# In[2]: cd E:/IBM_MY_COURSE/Project/Dataset Plant Disease/fruitdataset/fruit-dataset

Apply ImageDataGenerator functionality to Train and Test set

Preprocessing # In[3]: from keras.preprocessing.image import ImageDataGenerator train_datagen = ImageDataGenerator(rescale=1./255,shear_range=0.2,zoom_range=0.2,horizontal_fli p=True) test_datagen = ImageDataGenerator(rescale=1) # In[4]: pwd

# In[5]: x_train = train_datagen.flow_from_directory('E:/IBM_MY_COURSE/Project/Dataset Plant Disease/fruit-dataset/fruitdataset/train',target_size=(128,128),batch_size=32,class_mode='categorical')

# In[6]: x_test=test_datagen.flow_from_directory('E:/IBM_MY_COURSE/Project/Dataset Plant Disease/fruit-dataset/fruit-dataset/test',target_size=(128,128), batch_size=32,class_mode='categorical') # # Import the models

```
# In[7]: from tensorflow.keras.models import Sequential from
tensorflow.keras.layers import Dense,Convolution2D,MaxPool2D,Flatten

# # Initializing the models 10

# In[8]: model=Sequential()

# # Add CNN Layers

# In[9]:
model.add(Convolution2D(32,(3,3),input_shape=(128,128,3),activation='relu'))

# In[10]: x_train.class_indices

# # Add Pooling layer

# In[11]: model.add(MaxPool2D(pool_size=(2,2)))

# # Add Flatten layer # In[12]: model.add(Flatten())

# # Add Dense Layer

# In[21]: model.add(Dense(40, kernel_initializer='uniform',activation='relu'))
model.add(Dense(20, kernel_initializer='random_uniform',activation='relu'))

# # Add Output Layer # In[24]: model.add(Dense(6,activation='softmax',
kernel_initializer='random_uniform'))

# # Compile the model # In[25]:
model.compile(loss='categorical_crossentropy',optimizer='adam',metrics=['accur
acy' ]) # In[26]: len(x_train)

# In[27]: 5384/32
```

Fit the Model

# In[28]:

model.fit_generator(x_train,steps_per_epoch=168,validation_data=x_test,validat ion_st eps=52,epochs=3)

Save the Model

# In[29]: model.save("fruit.h5")

# In[30]: ls

Test the Model

# In[32]: from keras.preprocessing import image from tensorflow.keras.preprocessing.image import img_to_array from tensorflow.keras.models import load_model import numpy as np

# In[33]: model = load_model("fruit.h5")

Test Apple_Healthy Class images

# In[37]: img = image.load_img('E:/IBM_MY_COURSE/Project/Dataset Plant Disease/fruitdataset/fruit-dataset/test/Apple___healthy/00fca0da-2db3-481bb98a9b67bb7b105c___RS_HL 7708.JPG',target_size=(128,128)) 11

# In[39]: x=image.img_to_array(img) x=np.expand_dims(x,axis=0)

# In[40]: pred = model.predict_classes(x)

# In[41]: pred

# In[45]: index

=['Apple___Black_rot','Apple___healthy','Corn_(maize)___Northern_Leaf_Blight','Corn_(maize)___healthy','Peach___Bacterial_spot','Peach___healthy']

# In[46]: print('the given image belogs to=',index[pred[0]])

Test Apple Black Rot class images # In[54]: img = image.load_img('E:/IBM_MY_COURSE/Project/Dataset Plant Disease/fruitdataset/fruit-dataset/test/Apple___Black_rot/0f3d45f4-e121-42cda5b6- be2f866a0574___JR_FrgE.S 2870.JPG',target_size=(128,128))

# In[55]: x=image.img_to_array(img) x=np.expand_dims(x,axis=0) pred = model.predict_classes(x) print('the given image belogs to=',index[pred[0]])

Test Corn Northern leaf Blight class images

# In[56]: img = image.load_img('E:/IBM_MY_COURSE/Project/Dataset Plant Disease/fruitdataset/fruitdataset/test/Corn_(maize)___Northern_Leaf_Blight/00a14441-7a62- 4034-bc40b196aeab2785___RS_NLB 3932.JPG',target_size=(128,128))

# In[57]: x=image.img_to_array(img) x=np.expand_dims(x,axis=0) pred = model.predict_classes(x) print('the given image belogs to=',index[pred[0]])

Test Corn Healthy class images # In[58]: img = image.load_img('E:/IBM_MY_COURSE/Project/Dataset Plant Disease/fruitdataset/fruit-dataset/test/Corn_(maize)___healthy/0a68ef5a-027c41ae-b227- 159dae77d3dd___R.S_HL 7969 copy.jpg',target_size=(128,128))

# In[59]: x=image.img_to_array(img) x=np.expand_dims(x,axis=0) pred = model.predict_classes(x) print('the given image belogs to=',index[pred[0]]) # #

Test Peach Bacterial spot class images # In[60]: img = image.load_img('E:/IBM_MY_COURSE/Project/Dataset Plant Disease/fruitdataset/fruit-dataset/test/Peach___Bacterial_spot/00ddc106-692e4c67-b2e8- 569c924caf49___Rutg._Bact.S 1228.JPG',target_size=(128,128)) 12 # In[61]: x=image.img_to_array(img) x=np.expand_dims(x,axis=0) pred = model.predict_classes(x) print('the given image belogs to=',index[pred[0]])

Test Peach Healthy class images

# In[62]: img = image.load_img('E:/IBM_MY_COURSE/Project/Dataset Plant Disease/fruitdataset/fruit-dataset/test/Peach___healthy/1a07ce54-f4fd-41cfb088-144f6bf71859___Rutg._HL 3543.JPG',target_size=(128,128))

# In[63]: x=image.img_to_array(img) x=np.expand_dims(x,axis=0) pred = model.predict_classes(x) print('the given image belogs to=',index[pred[0]])