A Project Report on AI based Fertilizer Recommendation System based on Diseases of Fruits and Vegetables leaves

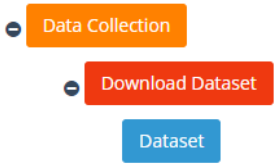
R. Mohan
Vellore Institute of Technology,
Chennai Campus

Objectives of the Project

- to preprocess the images.
- applying the CNN algorithm to the dataset.
- to find how deep neural networks detect the disease.
- to find the accuracy of the model.
- to build web applications using the Flask framework.

Steps involved in the Projects

1. Download the dataset from the project page and upload in in Jupiter notebook



2. Preprocessing of Images using Tensor Flow models and keras preposcessing libraries

Image Augmentation

```
In [2]: from tensorflow.keras.preprocessing.image import ImageDataGenerator

In [3]: train_datagen=ImageDataGenerator(rescale=1./255,zoom_range=0.2,horizontal_flip=True,vertical_flip=Fals

In [4]: test_datagen=ImageDataGenerator(rescale=1./255)
```

3.upload the dataset in jupiter notebook

```
In [7]: x_train=train_datagen.flow_from_directory(r'D:\Courses\General Work\Faculty Development Program Attende
                                                 class_mode='categorical',batch_size=24)
Found 5384 images belonging to 6 classes.

In [8]: x_test=test_datagen.flow_from_directory(r'D:\Courses\General Work\Faculty Development Program Attended\
                                                class_mode='categorical',batch_size=24)
Found 1686 images belonging to 6 classes.

In [9]: x_train.class_indices
Out[9]: {'Apple___Black_rot': 0,
         'Apple___healthy': 1,
         'Corn_(maize)___Northern_Leaf_Blight': 2,
         'Corn_(maize)___healthy': 3,
         'Peach___Bacterial_spot': 4,
         'Peach___healthy': 5}
```

4.Build the CNN Model and apply relu and softmax as activation function. Use Hiiden layer to capture the intricate features

```
In [10]: from tensorflow.keras.models import Sequential
         from tensorflow.keras.layers import Dense,Convolution2D,MaxPool2D,Flatten

In [11]: model=Sequential()

In [12]: model.add(Convolution2D(32,(3,3),input_shape=(64,64,3),activation='relu'))

In [13]: model.add(MaxPool2D(pool_size=(2,2)))

In [14]: model.add(Flatten())
```

Hidden Layers

```
In [15]: model.add(Dense(300,activation='relu'))
         model.add(Dense(150,activation='relu'))
```

5. Complete with output layer with 6 nodes. 6 nodes are choosen base on available cateriles under fruits. For Vegetables, there are 9 output layers based on 9 categories.

```
In [16]: x_train.class_indices
Out[16]: {'Apple___Black_rot': 0,
          'Apple___healthy': 1,
          'Corn_(maize)___Northern_Leaf_Blight': 2,
          'Corn_(maize)___healthy': 3,
          'Peach___Bacterial_spot': 4,
          'Peach___healthy': 5}

In [17]: model.add(Dense(6,activation='softmax'))

In [18]: model.compile(loss='categorical_crossentropy',optimizer='adam',metrics=['accuracy'])

In [19]: len(x train)
```

6. Comple, Fit and run the model

```
In [21]: model.fit_generator(x_train,steps_per_epoch=len(x_train),validation_data=x_test,validation_steps=len(x_

C:\Users\Admin\AppData\Local\Temp/ipykernel_9744/1582812018.py:1: UserWarning: `Model.fit_generator`
is deprecated and will be removed in a future version. Please use `Model.fit`, which supports generat
ors.
  model.fit_generator(x_train,steps_per_epoch=len(x_train),validation_data=x_test,validation_steps=le
n(x_test),epochs=10)

Epoch 1/10
225/225 [==============================] - 85s 360ms/step - loss: 0.6230 - accuracy: 0.7903 - val_los
s: 0.3480 - val_accuracy: 0.8814
Epoch 2/10
225/225 [==============================] - 37s 165ms/step - loss: 0.2986 - accuracy: 0.9023 - val_los
s: 0.3396 - val_accuracy: 0.8760
Epoch 3/10
```

7. Find the accuracy of the model

```
Epoch 8/10
225/225 [==============================] - 43s 191ms/step - loss: 0.0843 - accuracy: 0.9708 - val_los
s: 0.2718 - val_accuracy: 0.9170
Epoch 9/10
225/225 [==============================] - 41s 182ms/step - loss: 0.0832 - accuracy: 0.9718 - val_los
s: 0.1267 - val_accuracy: 0.9579
Epoch 10/10
225/225 [==============================] - 40s 179ms/step - loss: 0.0584 - accuracy: 0.9807 - val_los
s: 0.1023 - val_accuracy: 0.9644
```

]: <keras.callbacks.History at 0x1f1b063b400>

Simillarly for vegitable models and found more than 90 percent accuracy

```
In [70]: x_train.class_indices

Out[70]: {'Pepper,_bell___Bacterial_spot': 0,
          'Pepper,_bell___healthy': 1,
          'Potato___Early_blight': 2,
          'Potato___Late_blight': 3,
          'Potato___healthy': 4,
          'Tomato___Bacterial_spot': 5,
          'Tomato___Late_blight': 6,
          'Tomato___Leaf_Mold': 7,
          'Tomato___Septoria_leaf_spot': 8}
```

```
In [71]: model.add(Dense(9,activation='softmax'))
```

```
In [75]: model.fit_generator(x_train,steps_per_epoch=len(x_train),validation_data=x_test,validation_steps=len(x_
```

```
Epoch 1/10
C:\Users\Admin\AppData\Local\Temp/ipykernel_18036/1582812018.py:1: UserWarning: `Model.fit_generator` is deprecated and will be removed in a future version. Please use `Model.fit`, which supports generators.
  model.fit_generator(x_train,steps_per_epoch=len(x_train),validation_data=x_test,validation_steps=len(x_test),epochs=10)
238/238 [==============================] - 71s 296ms/step - loss: 1.2822 - accuracy: 0.5718 - val_los
s: 0.8739 - val_accuracy: 0.6906
Epoch 2/10
238/238 [==============================] - 73s 308ms/step - loss: 0.6694 - accuracy: 0.7609 - val_los
s: 0.5395 - val_accuracy: 0.8018
Epoch 3/10
238/238 [==============================] - 116s 487ms/step - loss: 0.4785 - accuracy: 0.8332 - val_lo
Epoch 9/10
238/238 [==============================] - 87s 367ms/step - loss: 0.2100 - accuracy: 0.9256 - 
s: 0.1917 - val_accuracy: 0.9315
Epoch 10/10
238/238 [==============================] - 103s 434ms/step - loss: 0.1837 - accuracy: 0.9347 -
ss: 0.2167 - val_accuracy: 0.9274
```

Out[75]: <keras.callbacks.History at 0x1fd8486f610>

8. save the model as h5 file

```
In [23]: model.save('Fertilizer_Fruits.h5')

In [24]: ls

 Volume in drive C is Windows
 Volume Serial Number is 4CE6-85EB
```

```
In [77]: model.save('Fertilizer_Vegetables.h5')

In [78]: ls

 Volume in drive C is Windows
 Volume Serial Number is 4CE6-85EB

 Directory of C:\Users\Admin
```

9. Testing the model

```
In [28]: img=image.load_img(r"C:\Users\Admin\Desktop\fig1.jpg")

In [29]: img
```

Out[29]:

```
In [30]: img=image.load_img(r"C:\Users\Admin\Desktop\fig1.jpg",target_size=(64,64))

In [31]: img
```

Out[31]:

```
In [32]: x=image.img_to_array(img)

In [33]: x

Out[33]: array([[[104.,  88.,  89.],
                 [119., 103., 104.],

Out[39]: array([1], dtype=int64)

In [40]: x_train.class_indices

Out[40]: {'Apple___Black_rot': 0,
          'Apple___healthy': 1,
          'Corn_(maize)___Northern_Leaf_Blight': 2,
          'Corn_(maize)___healthy': 3,
          'Peach___Bacterial_spot': 4,
          'Peach___healthy': 5}

In [41]: index=['Apple___Black_rot','Apple___healthy','Corn_(maize)___Northern_Leaf_Blight','Corn_(maize)___heal

In [42]: index[y[0]]

Out[42]: 'Apple___healthy'
```

simillary for Vegetables

```
In [82]: img=image.load_img(r"C:\Users\Admin\Desktop\vfig1.jpg")

In [83]: img
```

Out[83]:

```
In [84]: img=image.load_img(r"C:\Users\Admin\Desktop\vfig1.jpg",target_size=(64,64))

In [85]: img
```

Out[85]:

```
In [86]: x=image.img_to_array(img)
```

```
        'Tomato___Leaf_Mold': 7,
        'Tomato___Septoria_leaf_spot': 8}
```

```
In [95]: index=['Pepper,_bell___Bacterial_spot',
          'Pepper,_bell___healthy',
          'Potato___Early_blight',
          'Potato___Late_blight',
          'Potato___healthy',
          'Tomato___Bacterial_spot',
          'Tomato___Late_blight',
          'Tomato___Leaf_Mold',
          'Tomato___Septoria_leaf_spot']

In [96]: index[y[0]]

Out[96]: 'Tomato___Septoria_leaf_spot'
```

10. building Web Applicaation interface using FLASK Module

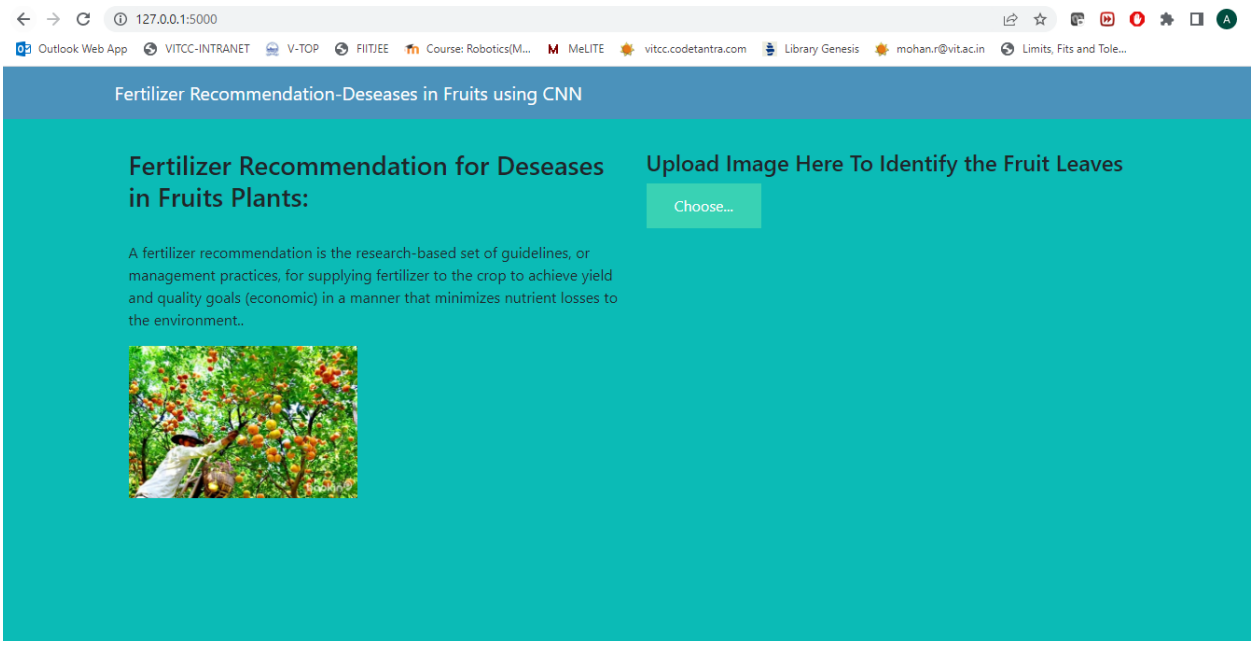
```
import requests
from tensorflow.keras.preprocessing import image
from tensorflow.keras.models import load_model

import numpy as np
import pandas as pd
import os
import tensorflow as tf

from flask import Flask, request, render_template, redirect, url_for
from werkzeug.utils import secure_filename
from tensorflow.python.keras.backend import set_session

app = Flask(__name__)
```

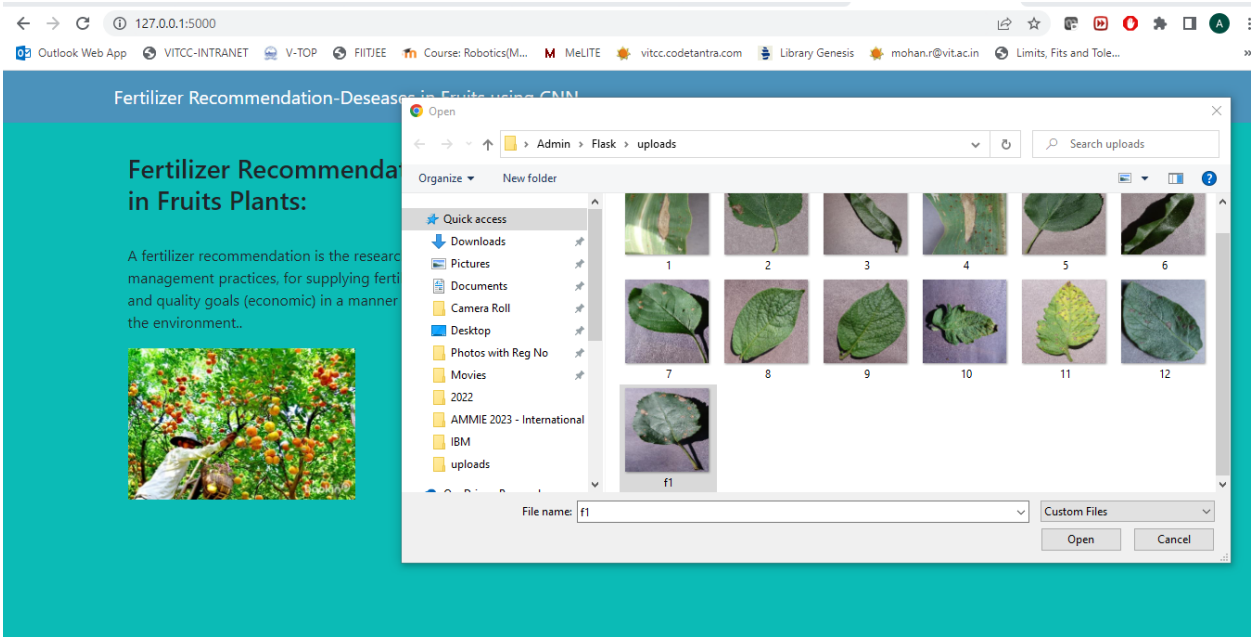
```
model = load_model("Fertilizer_Fruits.h5")


#render home page
@app.route('/')
def index():
    return render_template('index.html')


#render predict page
@app.route('/predict',methods=['GET','POST'])
def upload():
    if request.method == 'POST':
        f=request.files['image']
        basepath = os.path.dirname(__file__)
        filepath = os.path.join(basepath,'uploads',f.filename)
        f.save

        img = image.load_img(filepath,target_size = (64,64))
        x=image.img_to_array(img)
        x=np.expand_dims(x,axis=0)
        pred=np.argmax(model.predict(x),axis=1)
        index=['Apple___Black_rot',
               'Apple___healthy',
               'Corn_(maize)___Northern_Leaf_Blight',
               'Corn_(maize)___healthy',
               'Peach___Bacterial_spot',
               'Peach___healthy']

        text = 'The classied fruit is:'+str(index[pred[0]])
    return text


if __name__ == "__main__":
    app.run(debug=False)
```

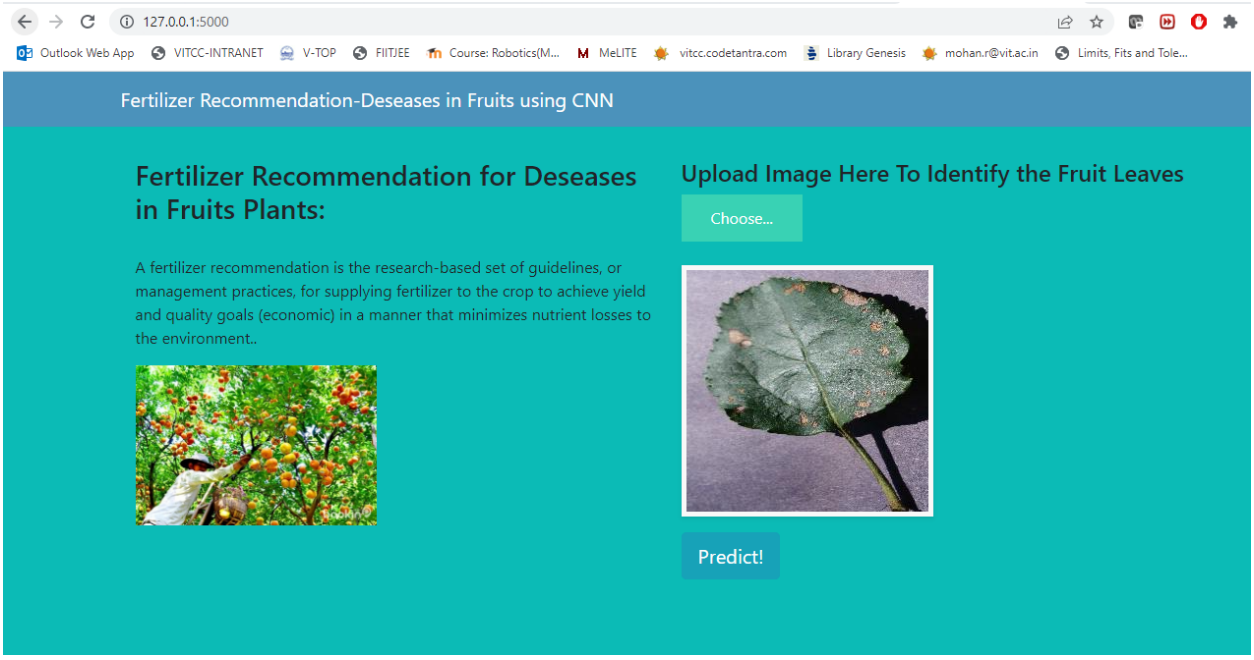
For Vegetables

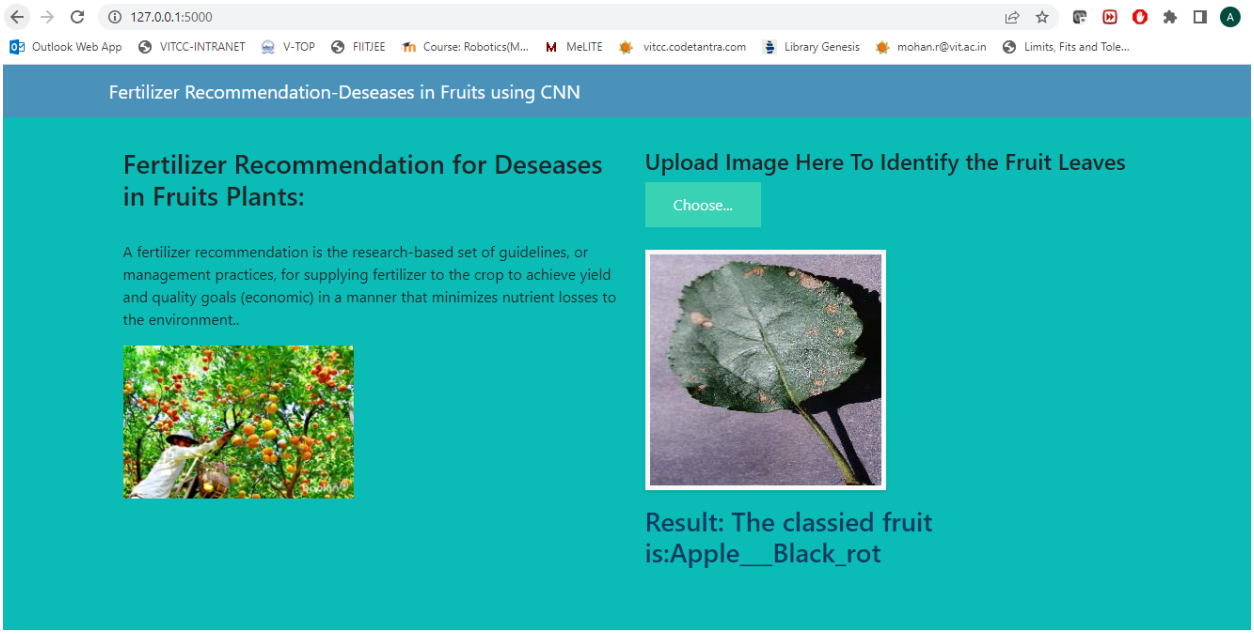
```
import requests
from tensorflow.keras.preprocessing import image
from tensorflow.keras.models import load_model

import numpy as np
import pandas as pd
import os
import tensorflow as tf

from flask import Flask, request, render_template, redirect, url_for
from werkzeug.utils import secure_filename
from tensorflow.python.keras.backend import set_session

app = Flask(__name__)

model = load_model("Fertilizer_Vegetables.h5")


#render home page
@app.route('/')
def index():
    return render_template('index.html')


#render predict page
@app.route('/predict',methods=['GET','POST'])
def upload():
    if request.method == 'POST':
        f=request.files['image']
        basepath = os.path.dirname(__file__)
        filepath = os.path.join(basepath,'uploads',f.filename)
        f.save

        img = image.load_img(filepath,target_size = (64,64))
        x=image.img_to_array(img)
        x=np.expand_dims(x,axis=0)
        pred=np.argmax(model.predict(x),axis=1)
```

```
    index=['Pepper,_bell___Bacterial_spot',
           'Pepper,_bell___healthy',
           'Potato___Early_blight',
           'Potato___Late_blight',
           'Potato___healthy',
           'Tomato___Bacterial_spot',
           'Tomato___Late_blight',
           'Tomato___Leaf_Mold',
           'Tomato___Septoria_leaf_spot']

    text = 'The classied Vegetable is:'+str(index[pred[0]])
  return text


if __name__ == "__main__":
  app.run(debug=False)
```




127.0.0.1:5000
Outlook Web App
VITCC-INTRANET
V-TOP
FIITJEE
Course: Robotics(M...
MeLITE
vitcc.codetantra.com
Library Genesis
mohan.r@vit.ac.in
Limits, Fits and Tole...
Fertilizer Recommendation-Deseases in Fruits using CNN
Fertilizer Recommenda
in Fruits Plants:
A fertilizer recommendation is the researc
management practices, for supplying ferti
and quality goals (economic) in a manner
the environment..
Open
Admin
Flask
uploads
Search uploads
Organize
New folder
Quick access
Downloads
Pictures
Documents
Camera Roll
Desktop
Photos with Reg No
Movies
2022
AMMIE 2023 - International
IBM
uploads
1
2
3
4
5
6
7
8
9
10
11
12
f1
File name:
f1
Custom Files
Open
Cancel


127.0.0.1:5000
Outlook Web App
VITCC-INTRANET
V-TOP
FIITJEE
Course: Robotics(M...
MeLITE
vitcc.codetantra.com
Library Genesis
mohan.r@vit.ac.in
Limits, Fits and Tole...
Fertilizer Recommendation-Deseases in Fruits using CNN
Fertilizer Recommendation for Deseases
in Fruits Plants:
A fertilizer recommendation is the research-based set of guidelines, or
management practices, for supplying fertilizer to the crop to achieve yield
and quality goals (economic) in a manner that minimizes nutrient losses to
the environment..
Upload Image Here To Identify the Fruit Leaves
Choose...
Predict!



Conclusion

Thus the prediction diseases of the fruits and vegetable plants are completed succeffly as stated in the project objectives.