BHARATI VIDYAPEETH (DEEMED TO BE UNIVERSITY)

COLLEGE OF ENGINEERING

DEPARTMENT OF ENGINEERING & TECHNOLOGY OFFCAMPUS, KHARGHAR, NAVI MUMBAI,410210

Mini Project Report

On

<u>Leaf Disease Detection</u>

Subject-: - <u>Mini Project</u>

Presented By

Roll No.	Name	PRN
3	Komal Giri	2043110165
22	Kanika Tilkan	2043110187
37	Amrit Chandra	2043110203

Signature of Internal Examiner

BHARATI VIDYAPEETH (DEEMED TO BE UNIVERSITY)

COLLEGE OF ENGINEERING

DEPARTMENT OF ENGINEERING & TECHNOLOGY OFFCAMPUS, KHARGHAR, NAVI MUMBAI,410210

This is to certify that the project entitled, "**Leaf Disease Detection**", which is being submitted here with for the award of B.Tech. CSBS Department, is the result of the work completed by Komal Giri under my supervision and guidance within the four walls of the institute and the same has not been submitted elsewhere for the award of any degree.

Guide
DEPARTMENT OF ENGINEERING &
TECHNOLOGY OFFCAMPUS,
KHARGHAR, NAVI MUMBAI

(Head of Department)
DEPARTMENT OF ENGINEERING &
TECHNOLOGY OFFCAMPUS
KHARGHAR, NAVI MUMBAI

Principal
DEPARTMENT OF ENGINEERING &
TECHNOLOGY OFFCAMPUS,
KHARGHAR, NAVI MUMBAI

BHARATI VIDYAPEETH (DEEMED TO BE UNIVERSITY)

COLLEGE OF ENGINEERING

DEPARTMENT OF ENGINEERING & TECHNOLOGY OFFCAMPUS, KHARGHAR, NAVI MUMBAI,410210

This is to certify that the project entitled, "**Leaf Disease Detection**", which is being submitted here with for the award of B.Tech. CSBS Department, is the result of the work completed by Kanika Tilkan under my supervision and guidance within the four walls of the institute and the same has not been submitted elsewhere for the award of any degree.

Guide
DEPARTMENT OF ENGINEERING &
TECHNOLOGY OFFCAMPUS,
KHARGHAR, NAVI MUMBAI

(Head of Department)
DEPARTMENT OF ENGINEERING &
TECHNOLOGY OFFCAMPUS
KHARGHAR, NAVI MUMBAI

Principal
DEPARTMENT OF ENGINEERING &
TECHNOLOGY OFFCAMPUS,
KHARGHAR, NAVI MUMBAI

BHARATI VIDYAPEETH (DEEMED TO BE UNIVERSITY)

COLLEGE OF ENGINEERING

DEPARTMENT OF ENGINEERING & TECHNOLOGY OFFCAMPUS, KHARGHAR, NAVI MUMBAI,410210

This is to certify that the project entitled, "**Leaf Disease Detection**", which is being submitted here with for the award of B.Tech. CSBS Department, is the result of the work completed by Amrit Chandra under my supervision and guidance within the four walls of the institute and the same has not been submitted elsewhere for the award of any degree.

Guide
DEPARTMENT OF ENGINEERING &
TECHNOLOGY OFFCAMPUS,
KHARGHAR, NAVI MUMBAI

(Head of Department)
DEPARTMENT OF ENGINEERING &
TECHNOLOGY OFFCAMPUS
KHARGHAR, NAVI MUMBAI

Principal
DEPARTMENT OF ENGINEERING &
TECHNOLOGY OFFCAMPUS,
KHARGHAR, NAVI MUMBAI

Abstract

One of the important and tedious tasks in agricultural practices is detection of disease on crops. It requires huge time as well as skilled labor. This paper proposes a smart and efficient technique for detection of crop disease which uses computer vision and machine learning techniques. The proposed system is able to detect 20 different diseases of 5 common plants with 93% accuracy. Identification of the plant diseases is the key to preventing the losses in the yield and quantity of the agricultural product. The studies of the plant diseases mean the studies of visually observable patterns seen on the plant. Health monitoring and disease detection on plant is very critical for sustainable agriculture. It is very difficult to monitor the plant diseases manually. It requires tremendous amount of work, expertise in the plant diseases, and also require the excessive processing time.

Hence, image processing is used for the detection of plant diseases. Disease detection involves the steps like image acquisition, image pre-processing, image segmentation, feature extraction and classification. This paper discussed the methods used for the detection of plant diseases using their leaves images. This paper also discussed some segmentation and feature extraction algorithm used in the plant disease detection. Diseases in plants cause major production and economic losses as well as reduction in both quality and quantity of agricultural products. Now a day's plant diseases detection has received increasing attention in monitoring large field of crops. Farmers experience great difficulties in switching from one disease control policy to another.

The naked eye observation of experts is the traditional approach adopted in practice for detection and identification of plant diseases. In this paper we review the need of simple plant leaves disease detection system that would facilitate advancements in agriculture. Early information on crop health and disease detection can facilitate the control of diseases through proper management strategies. This technique will improves productivity of crops. This paper also compares the benefits and limitations of these potential methods. It includes several steps viz. image acquisition, image pre-processing, features extraction and neural network based classification.

Index

Chapter 1

Introduction

In India about 70% of the populace relies on agriculture. Identification of the plant diseases is important in order to prevent the losses within the yield. It's terribly troublesome to observe the plant diseases manually. It needs tremendous quantity of labor, expertise within the plant diseases, and conjointly need the excessive time interval. Hence, image processing and machine learning models can be employed for the detection of plant diseases. In this project, we have described the technique for the detection of plant diseases with the help of their leaves pictures. Image processing is a branch of signal processing which can extract the image properties or useful information from the image. Machine learning is a sub part of artificial intelligence which works automatically or give instructions to do a particular task. The main aim of machine learning is to understand the training data and fit that training data into models that should be useful to the people. So it can assist in good decisions making and predicting the correct output using the large amount of training data. The color of leaves, amount of damage to leaves, area of the leaf, texture parameters are used for classification. In this project we have analyzed different image parameters or features to identifying different plant leaves diseases to achieve the best accuracy. Previously plant disease detection is done by visual inspection of the leaves or some chemical processes by experts. For doing so, a large team of experts as well as continuous observation of plant is needed, which costs high when we do with large farms. In such conditions, the recommended system proves to be helpful in monitoring large fields of crops. Automatic detection of the diseases by simply seeing the symptoms on the plant leaves makes it easier as well as cheaper. The proposed solution for plant disease detection is computationally less expensive and requires less time for
prediction than other deep learning-based approaches since it uses statistical machine learning and image processing algorithm.

Chapter 2

Literature Survey

In 2015, S. Khirade et Al. tackled the problem of plant disease detection using digitalimage processing techniques and back propagation neural network (BPNN) [1].Authors have elaborated different techniques for the detection of plant disease usingthe images of leaves. They have implemented Otsu’s thresholding followed byboundary detection and spot detection algorithm to segment the infected part in leaf.After that they have extracted the features such as color, texture, morphology, edges etc. for classification of plant disease. BPNN is used for classification i.e. to detect the plant disease.Shiroop Madiwalar and Medha Wyawahare analyzed different image processingapproaches for plant disease detection in their research [2]. Authors analyzed the color and texture features for the detection of plant disease. They have experimented their algorithms on the dataset of 110 RGB images. The features extracted for classification were mean and standard deviation of RGB and YCbCr channels, grey level cooccurrence matrix (GLCM) features, the mean and standard deviation of the image convolved with Gabor filter. Support vector machine classifier was used for classification.

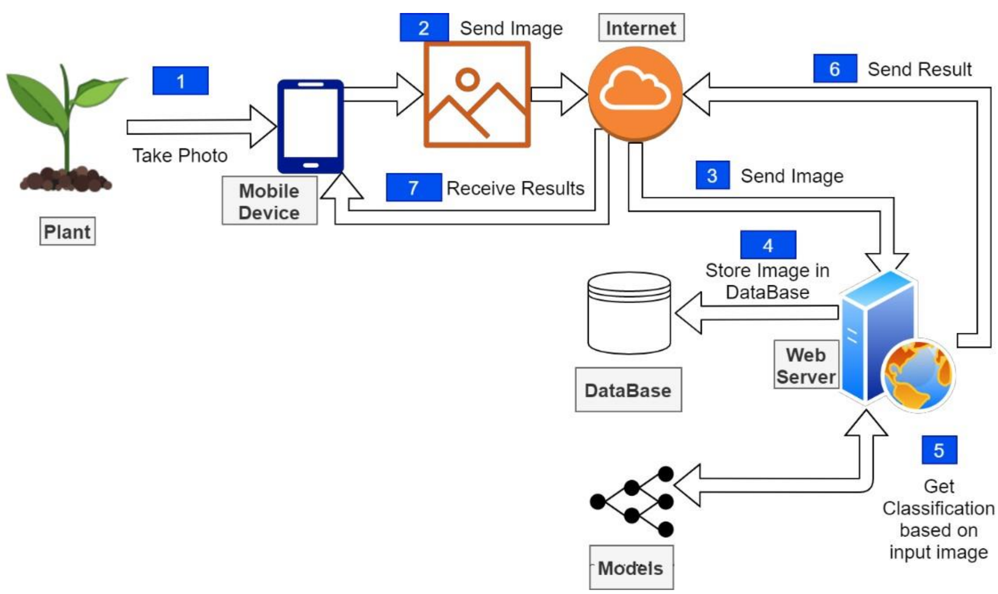
Authors concluded that GCLM features are effective to detect normal leaves. Whereas color features and Gabor filter features are considered as best for detecting anthracnose affected leaves and leaf spot respectively. They have achieved highest accuracy of 83.34% using all the extracted features.Peyman Moghadam et Al. demonstrated the application of hyperspectral imaging in plant disease detection task [3]. visible and near-infrared (VNIR) and short-wave infrared (SWIR) spectrums were used in this research. Authors have used k-means clustering algorithm in spectral domain for the segmentation of leaf. They have proposed a novel grid removal algorithm to remove the grid from hyperspectral images. Authors have achieved the accuracy of 83% with vegetation indices in VNIR spectral range and 93% accuracy with full spectrum.

Though the proposed method achieved higher accuracy, it requires the hyperspectral camera with 324 spectral bands so the solution becomes too costly.Sharath D. M. et Al. developed the Bacterial Blight detection system for Pomegranate plant by using features such as color, mean, homogeneity, SD, variance, correlation, entropy, edges etc. Authors have implemented grab cut segmentation for segmenting the region of interest in the image [4]. Canny edge detector was used to extract the edges from the images. Authors have successfully developed a system which can predict the infection level in the fruit.Garima Shrestha et Al. deployed the convolutional neural network to detect the plant disease [5]. Authors have successfully classified 12 plant diseases with 88.80% accuracy. The dataset of 3000 high resolution RGB images were used for experimentation. The network has 3 blocks of convolution and pooling layers. This makes the network computationally expensive. Also the F1 score of the model is 0.12 which is very low because of higher number of false negative predictions.

Chapter 3

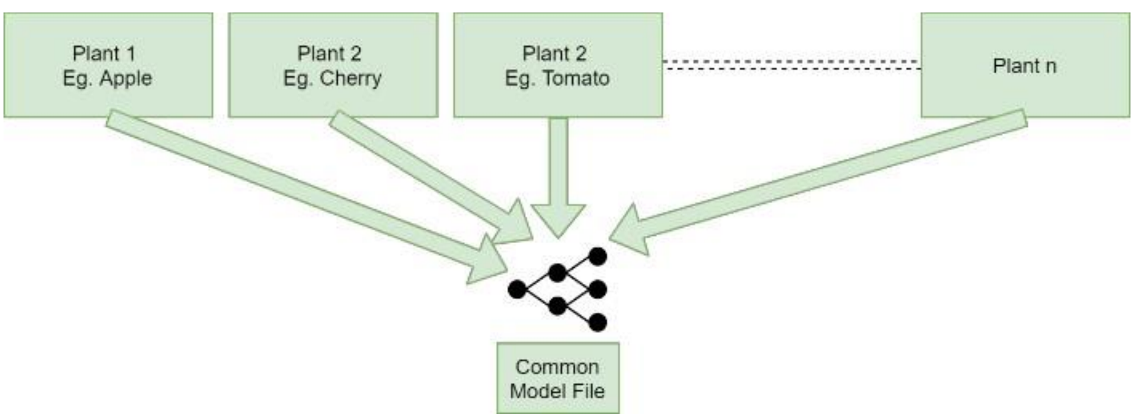
System Design

This research proposes a web-based application which will integrate with a mobile application to identify the various plant diseases based on the images of the plant leaves as shown in the proposed system will be called 'PotBot'. PotBotwill allow farmers to take pictures of plant leaves from their smart phones. The image will be submitted to the application server, through the internet, where the image will be classified on the basis of the information available. The result will bethen displayed on the mobile device. Farmers can then use this information to recognize if any diseases have plagued their fields and can take action accordingly. As classification process takes a high amount of computation power, we have decided to segregate that work to the web-server which will host the web application. Farmers can also utilize the web-portal to access the services of PotBot in a similar manner as the mobile application. Going to the technical aspect, we propose to utilize Flask/Django, which are Python based, to build the web-application which will also act as the server for the mobile platform. The mobile application will be built initially for Android based handheld systems. The android application itself will be written with a mixture of Java and Kotlin languages. For the processing of the models TensorFlow will be used on the back-end server. The interface of both theweb application and the mobile application will be user-friendly and simple as shown in This will make the system easy to use.

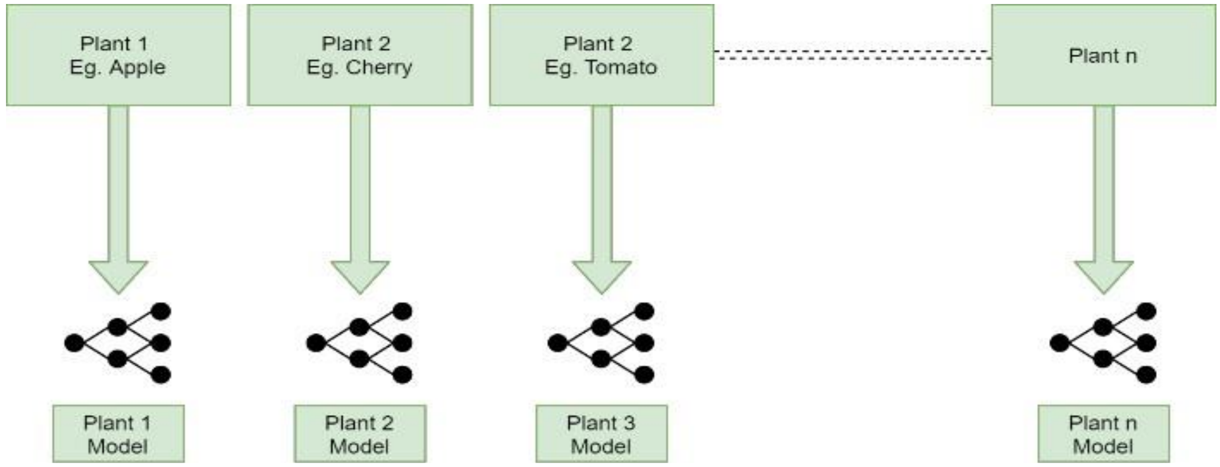


The proposed system has tremendous potential to aid the small-scale farmers. But measures need to be taken to ensure that the application is practically valid and relevant. In a third world country like India, literacy rate among farmers is low. This will pose a challenge to make this application widely usable. Proper training and guidance are necessary to make the application practically usable. PotBot can complement the ongoing agricultural initiatives from the government to enhance the crop yields. Region specific plants need to be focused and systematically studied.

The result of these studies must be integrated into the system. Young generation of farmers need to be targeted for the use of this application as they will have the flexibility to understand and learn to use the application. One major challenge that can be addressed in the future will be to make the application work completely offline.
With the advancement in Deep Learning and Mobile Technologies, this will be a major achievement. Farmers will no longer be dependent on internet connection to utilize the application. Information and Communication Technologies needs to be integrated with agriculture to improve the current status of the farmers. ICTs can help to gather and manage large amount of information which can be stored and retrieved at the user’s convenience. This will help farmers to access relevant and accurate information on demand.



Old Approach for training neural network.



Proposed New way of training neural network

Technology and Language used for project: -

Languages:

1. Python: Version 3.10.2

Tools with their versions:

1. Operating System: Windows
2. Editors and IDE:
    a. Visual Studio Code with Jupyter notebook: Version 1.73
3. Dataset: APPLE_DISEASE_DATASET
4. Libraries and Framework:
    a. NumPy: Version 1.23.1
    b. Pandas: Version 1.4.3
    c. Seaborn: Version 0.11.2
    d. Matplotlib: Version 3.5.2
    e. Scikit Learn: Version 1.1.2
    f. Keras: Version 4
    g. Tensorflow: Version Nightly

Chapter 4

Implementation

For this project we have used public dataset for plant leaf disease detection called Plant Village curated by Sharada P. Mohanty et Al. The dataset consists of 87000 RGB images of healthy and unhealthy plant leaves having 38 classes out of which We have selected only 25 classes for experimentation of our algorithm.

```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import tensorflow as tf
from tensorflow.keras import models, layers
```
[9] Python

```python
BATCH_SIZE = 32
IMAGE_SIZE = 256
CHANNELS = 3
EPOCHS = 500
```
[10] Python

```python
dataset = tf.keras.preprocessing.image_dataset_from_directory(
    "D:\kasmirrrrrr\APPLE_DISEASE_DATASET",
    shuffle = True,
    image_size = (IMAGE_SIZE,IMAGE_SIZE),
    batch_size = BATCH_SIZE
)
```
[11] Python

```
Found 419 files belonging to 4 classes.
```


```python
dataset = tf.keras.preprocessing.image_dataset_from_directory(
    "D:\kasmirrrrrr\APPLE_DISEASE_DATASET",
    shuffle = True,
    image_size = (IMAGE_SIZE,IMAGE_SIZE),
    batch_size = BATCH_SIZE
)
```
[11] Python

```
Found 419 files belonging to 4 classes.
```


```python
class_names = dataset.class_names
class_names
```
[12] Python

```
['APPLE ROT LEAVES', 'HEALTHY LEAVES', 'LEAF BLOTCH', 'SCAB LEAVES']
```

```python
plt.figure(figsize=(10,10))
for image_batch, label_batch in dataset.take(1):
    print(image_batch.shape)
    print(label_batch.numpy())
    for i in range(0,9):
        ax = plt.subplot(3,3,i+1)
        plt.imshow(image_batch[i].numpy().astype("uint8"))
        plt.title(class_names[label_batch[i]])
        plt.axis("off")
```

[13]

Python

```
(32, 256, 256, 3)
[3 1 0 2 3 3 2 0 0 3 0 3 0 0 1 1 3 0 0 2 0 3 3 2 2 3 1 3 2 3 0 3]
```

```python
len(dataset)
```
[14] Python

```
14
```

```python
def get_dataset_partitions_tf(ds, train_split=0.9, test_split=0.1, shuffle=True, shuffle_size=10000):
    assert (train_split + test_split) == 1

    ds_size = len(ds)

    if shuffle:
        ds = ds.shuffle(shuffle_size, seed=12)

    train_size = int(train_split * ds_size)
    test_size = int(test_split * ds_size)

    train_ds = ds.take(train_size)
    test_ds = ds.skip(test_size)

    return train_ds, test_ds
```
[15] Python

```python
train_ds = train_ds.cache().shuffle(1000).prefetch(buffer_size=tf.data.AUTOTUNE)
test_ds = test_ds.cache().shuffle(1000).prefetch(buffer_size=tf.data.AUTOTUNE)
```
[17] Python

```python
resize_and_rescale = tf.keras.Sequential([
  layers.experimental.preprocessing.Resizing(IMAGE_SIZE, IMAGE_SIZE),
  layers.experimental.preprocessing.Rescaling(1./255),
])
```
[18] Python

```python
data_augmentation = tf.keras.Sequential([
  layers.experimental.preprocessing.RandomFlip("horizontal_and_vertical"),
  layers.experimental.preprocessing.RandomRotation(0.2),
  layers.experimental.preprocessing.RandomContrast(0.2),
  layers.experimental.preprocessing.RandomZoom(0.2)
])
```
[19] Python

```python
input_shape = (BATCH_SIZE, IMAGE_SIZE, IMAGE_SIZE, CHANNELS)
n_classes = 4

model = models.Sequential([
    resize_and_rescale,
    data_augmentation,
    layers.Conv2D(32, kernel_size = (3,3), activation='relu', input_shape=input_shape, padding='same'),
    layers.MaxPooling2D((2, 2)),
    layers.Conv2D(64,  kernel_size = (3,3), activation='relu', padding='same'),
    layers.MaxPooling2D((2, 2)),
    layers.Conv2D(128,  kernel_size = (3,3), activation='relu', padding='same'),
    layers.MaxPooling2D((2, 2)),
    layers.Conv2D(128, (3, 3), activation='relu', padding='same'),
    layers.MaxPooling2D((2, 2)),
    layers.Conv2D(128, (3, 3), activation='relu', padding='same'),
    layers.MaxPooling2D((2, 2)),
    layers.Conv2D(256, (3, 3), activation='relu', padding='same'),
    layers.MaxPooling2D((2, 2)),
    layers.Conv2D(256,  (3, 3), activation='relu', padding='same'),
    layers.MaxPooling2D((2, 2)),
    layers.Conv2D(256, (3, 3), activation='relu', padding='same'),
    layers.MaxPooling2D((2, 2)),
    layers.Flatten(),
    layers.Dense(256, activation='relu'),
    layers.Dropout(0.1),
    layers.Dense(500, activation='relu'),
    layers.Dropout(0.1),
    layers.Dense(n_classes, activation='softmax')
])

model.build(input_shape=input_shape)
```

[20] Python

```python
model.summary()
```

[21] Python

```
Output exceeds the size limit. Open the full output data in a text editor
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 sequential (Sequential)     (32, 256, 256, 3)         0

 sequential_1 (Sequential)   (None, 256, 256, 3)       0

 conv2d (Conv2D)             (None, 256, 256, 32)      896

 max_pooling2d (MaxPooling2D  (None, 128, 128, 32)     0
 )

 conv2d_1 (Conv2D)           (None, 128, 128, 64)      18496

 max_pooling2d_1 (MaxPooling  (None, 64, 64, 64)       0
 2D)

 conv2d_2 (Conv2D)           (None, 64, 64, 128)       73856

 max_pooling2d_2 (MaxPooling  (None, 32, 32, 128)      0
 2D)

 conv2d_3 (Conv2D)           (None, 32, 32, 128)       147584

...
Total params: 2,060,040
Trainable params: 2,060,040
Non-trainable params: 0
_________________________________________________________________
```

```python
model.compile(
    optimizer=tf.keras.optimizers.Adam(learning_rate=0.0001),
    loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=False),
    metrics=['accuracy']
)
```

[22] Python

```python
history = model.fit(
    train_ds,
    batch_size=BATCH_SIZE,
    verbose=1,
    epochs=EPOCHS
)
```

[23] Python

```
Output exceeds the size limit. Open the full output data in a text editor
Epoch 1/500
12/12 [==============================] - 37s 2s/step - loss: 1.3675 - accuracy: 0.3634
Epoch 2/500
12/12 [==============================] - 30s 2s/step - loss: 1.3214 - accuracy: 0.3859
Epoch 3/500
12/12 [==============================] - 31s 3s/step - loss: 1.3181 - accuracy: 0.4000
Epoch 4/500
12/12 [==============================] - 30s 2s/step - loss: 1.3208 - accuracy: 0.3324
Epoch 5/500
12/12 [==============================] - 29s 2s/step - loss: 1.3032 - accuracy: 0.3690
Epoch 6/500
12/12 [==============================] - 30s 2s/step - loss: 1.2838 - accuracy: 0.3887
Epoch 7/500
12/12 [==============================] - 29s 2s/step - loss: 1.2780 - accuracy: 0.3662
Epoch 8/500
12/12 [==============================] - 30s 2s/step - loss: 1.2538 - accuracy: 0.2873
Epoch 9/500
12/12 [==============================] - 31s 3s/step - loss: 1.2119 - accuracy: 0.3577
Epoch 10/500
12/12 [==============================] - 31s 3s/step - loss: 1.2090 - accuracy: 0.3408
Epoch 11/500
12/12 [==============================] - 26s 2s/step - loss: 1.1764 - accuracy: 0.4028
```

```
Epoch 499/500
12/12 [==============================] - 34s 3s/step - loss: 0.1491 - accuracy: 0.9465
Epoch 500/500
12/12 [==============================] - 30s 2s/step - loss: 0.1285 - accuracy: 0.9465
```

```python
scores = model.evaluate(test_ds)
```
[24] Python

```
13/13 [==============================] - 9s 573ms/step - loss: 3.1607 - accuracy: 0.4729
```

```python
scores
```
[25] Python

```
[3.160741090774536, 0.4728682041168213]
```

```python
history.params
```
[26] Python

```
{'verbose': 1, 'epochs': 500, 'steps': 12}
```

```python
history.history.keys()
```
[27] Python

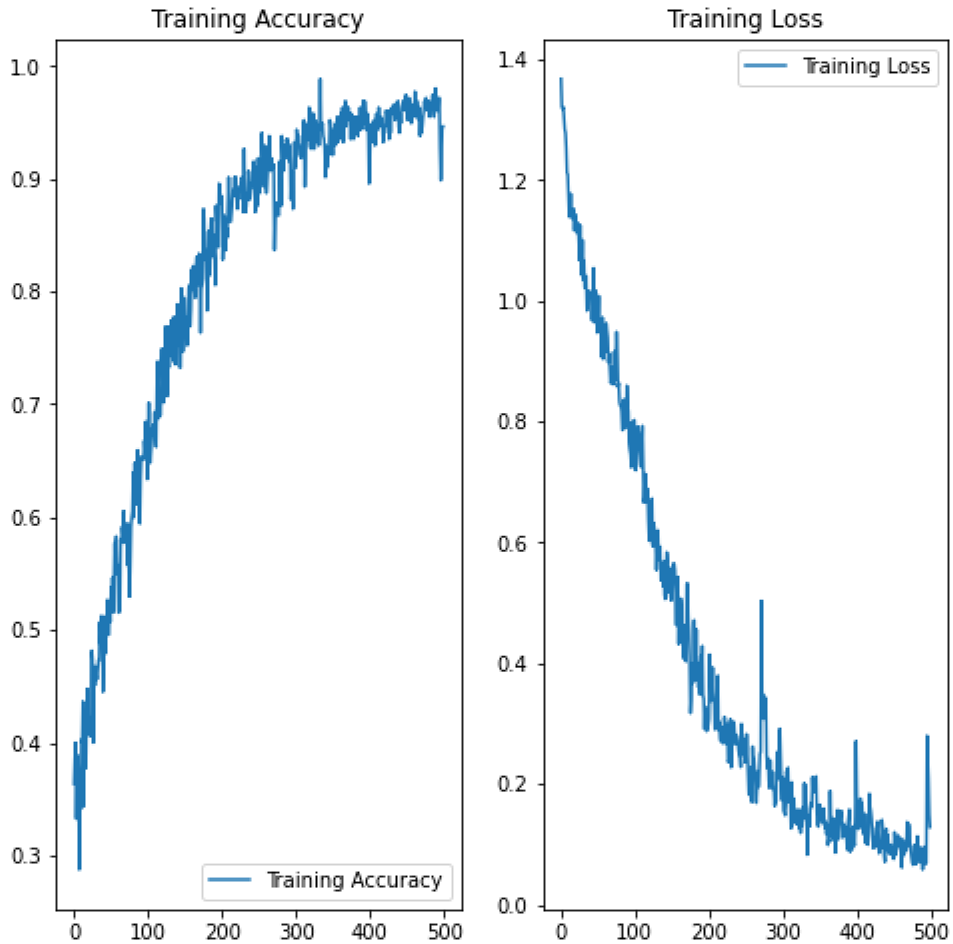
```
dict_keys(['loss', 'accuracy'])
```

```python
acc = history.history['accuracy']

loss = history.history['loss']
```
[28] Python

```python
plt.figure(figsize=(8, 8))
plt.subplot(1, 2, 1)
plt.plot(range(EPOCHS), acc, label='Training Accuracy')
plt.legend(loc='lower right')
plt.title('Training Accuracy')

plt.subplot(1, 2, 2)
plt.plot(range(EPOCHS), loss, label='Training Loss')
plt.legend(loc='upper right')
plt.title('Training Loss')
plt.show()
```
29] Python

Chapter 5

Results



Chapter 6

Conclusion

In summary, mental health problems are prevalent in college students, with substance use, anxiety, and mood disorders being the most common. Traditional college students are in a transitional age, young adulthood, which is associated with numerous stressors and during which many mental health problems often first occur.

In our project, 18-year-old are more prone to depression, anxiety and panic attacks and 21-year-old are the least prone.1st year students are more prone to depression, anxiety and panic attacks, whereas 4th year students are less prone. Depression is more among female students, and anxiety is more among male students. Students with higher CGPA experience depression, anxiety, and panic attacks more often as compared to those with lesser CGPA.

References

[1] S. D. Khirade and A. B. Patil, "Plant Disease Detection Using Image Processing," 2015 International Conference on Computing Communication Control and Automation, 2015, pp. 768-771, doi: 10.1109/ICCUBEA.2015.153.

[2] S. C. Madiwalar and M. V. Wyawahare, "Plant disease identification: A comparative study," 2017 International Conference on Data Management, Analytics and Innovation (ICDMAI), 2017, pp. 13-18, doi: 10.1109/ICDMAI.2017.8073478.

[3] P. Moghadam, D. Ward, E. Goan, S. Jayawardena, P. Sikka and E. Hernandez, "Plant Disease Detection Using Hyperspectral Imaging," 2017 International Conference on Digital Image Computing: Techniques and Applications (DICTA), 2017, pp. 1-8, doi: 10.1109/DICTA.2017.8227476.

[4] S. D.M., Akhilesh, S. A. Kumar, R. M.G. and P. C., "Image based Plant Disease Detection in Pomegranate Plant for Bacterial Blight," 2019 International Conference on Communication and Signal Processing (ICCSP), 2019, pp. 0645-0649, doi: 10.1109/ICCSP.2019.8698007.

[5] G. Shrestha, Deepsikha, M. Das and N. Dey, "Plant Disease Detection Using CNN," 2020 IEEE Applied Signal Processing Conference (ASPCON), 2020, pp. 109-113, doi: 10.1109/ASPCON49795.2020.9276722.

[6] Mohanty SP, Hughes DP and Salathé M (2016) Using Deep Learning for Image-Based Plant Disease Detection. Front. Plant Sci. 7:1419. doi: 10.3389/fpls.2016.01419

[7] R. M. Haralick, K. Shanmugam and I. Dinstein, "Textural Features for Image Classification," in IEEE Transactions on Systems, Man, and Cybernetics, vol. SMC-3, no. 6, pp. 610-621, Nov. 1973, doi: 10.1109/TSM.

[8] Anju Duhan and Satbir Singh. 2017. Sources of Agricultural Information Accessed by Farmers in Haryana, India.Int.J.Curr.Microbiol.App.Sci. 6(12): 1559-1565. doi: https://doi.org/10.20546/ijcmas.2017.612.175

[9] Kassam, A. (2003). Farming Systems and Poverty 2001: Improving Farmers' Livelihoods in a Changing World. By J. Dixon, A. Gulliver and D. Gibbon. Rome and Washington DC: FAO and the World Bank (2002), pp. 412, £27.00, ISBN 92-5.

[10] L. Ale, N. Zhang, H. Wu, D. Chen, and T. Han, “Online proactive caching in mobile edge computing using bidirectional deep recurrent neural network,” IEEE Internet of Things Journal,

vol. 6, no. 3, pp. 5520–5530, 2019.

[11] L. Ale, X. Fang, D. Chen, Y.Wang, and N. Zhang, “Lightweight deep learning model for facial expression recognition,” Proceedings of IEEE BigdataSE, 2019.

[12] L. Ale, A. Sheta, L. Li, Y. Wang and N. Zhang, "Deep Learning Based Plant Disease Detection for Smart Agriculture," 2019 IEEE Globecom Workshops (GC Wkshps), Waikoloa, HI, USA, 2019, pp. 1-6, doi: 10.1109/ GCWk shps45667.2019.9024439.

[13] Hughes, David P. and Marcel Salathé. “An open access repository of images on plant health to enable the development of mobile disease diagnostics through machine learning and crowdsourcing.” ArXiv abs/1511.08060 (2015): n. pag.

[14] Sharada P Mohanty, David P Hughes, and Marcel Salathé, "Using Deep Learning for Image-Based Plant Disease Detection," Frontiers in Plant Science, vol. 7, September 2016.

[15] A. K. Rangarajan, R. Purushothaman, and A. Ramesh, “Tomato crop disease classification using pre-trained deep learning algorithm,” Procedia Computer Science, vol. 133, pp. 1040–1047, Jan. 2018, doi: 10.1016/j.procs.2018.07.070.

[16] Huang, Gao, et al. “Densely Connected Convolutional Networks.” ArXiv:1608.06993 [Cs], Jan. 2018. arXiv.org, http://arxiv.org/abs/1608.06993.

[17] Martin Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving,Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek G. Murray, Benoit Steiner, Paul Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yum Xiaoqiang Zheng, "TensorFlow: A system for large-scale machine learning", 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16), USENIX Association (2016), pp. 265-283