

# **Revolutionizing Potato-Farming**

## **A MINI PROJECT REPORT**

**18CSC305J - ARTIFICIAL INTELLIGENCE**

*Submitted by*

**BALA AKHIL IMMADISSETTI (RA2111026010412)**

**CHAKRADHAR (RA2111026010370)**

**ABHITESH (RA2111026010369)**

*Under the guidance of*

**Dr. M. KARAPAGAM**

Assistant Professor, Department of Computer Science and Engineering

*in partial fulfillment for the award of the degree*

*of*

**BACHELOR OF TECHNOLOGY**

*in*

**COMPUTER SCIENCE & ENGINEERING**

*of*

**FACULTY OF ENGINEERING AND TECHNOLOGY**



**SRM**  
INSTITUTE OF SCIENCE & TECHNOLOGY  
Deemed to be University u/s 3 of UCC Act, 1956

S.R.M. Nagar, Kattankulathur, Chengalpattu District

**MAY 2024**

# **SRM INSTITUTE OF SCIENCE AND TECHNOLOGY**

(Under Section 3 of UGC Act, 1956)

## **BONAFIDE CERTIFICATE**

Certified that Mini project report titled **“REVOLUTIONIZING POTATO-FARMING”** is the bona fide work of **Balaakhil immadisetti (RA2111026010412), Abhitesh (RA2111026010369) and Chakradhar (RA2111026010370)** who carried out the minor project under my supervision. Certified further, that to the best of my knowledge, the work repeated herein does not form any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

### **SIGNATURE**

Dr. M. Karapagam

Assistant Professor

Department of Computer Science and Engineering

## ABSTRACT

Potato, a crucial global food crop, faces persistent threats from diseases like early blight and late blight, jeopardizing both yields and economic stability. In response, we present an innovative approach using CNN for early disease detection in potato crops. Our objective is to empower farmers with rapid and accurate disease identification, facilitating timely treatments to prevent economic losses and reduce agricultural waste. Our CNN model achieved a remarkable validation accuracy of 99.59%, outperforming conventional methods. With 183,747 trainable parameters, our efficient and accurate model is set to revolutionize agricultural practices. Furthermore, we are deploying our model to Google Cloud and developing a React Native mobile app for potato prediction. This integration aims to provide farmers with real-time disease insights, preventing economic losses and ensuring global food security, thereby enhancing economic stability in the potato farming sector.

# TABLE OF CONTENTS

<b>ABSTRACT</b>	<b>iii</b>
<b>TABLE OF CONTENTS</b>	<b>iv</b>
<b>LIST OF FIGURES</b>	<b>v</b>
<b>ABBREVIATIONS</b>	<b>vi</b>
<b>1 INTRODUCTION</b>	<b>6</b>
<b>2 LITERATURE SURVEY</b>	<b>7</b>
<b>3 SYSTEM ARCHITECTURE AND DESIGN</b>	<b>8</b>
3.1 Architecture diagram of proposed approach	8
3.2 UML Diagrams	9 - 10
<b>4 METHODOLOGY</b>	<b>10</b>
4.1 Data Set Description	11 - 12
<b>5 CODING AND TESTING</b>	<b>13 - 31</b>
<b>6 RESULTS</b>	<b>32-33</b>
6.1 Training and Testing accuracies	32
6.2 Training and Validation accuracy plot	32
6.3 Sample Test Images with predicted accuracy	33
<b>7 CONCLUSION AND FUTURE ENHANCEMENT</b>	<b>34</b>
7.1 Conclusion	
7.2 Future Enhancement	
<b>REFERENCES</b>	<b>35</b>

# CHAPTER 1

## INTRODUCTION

Potato cultivation faces significant threats from two common diseases, Early Blight and Late Blight. These formidable pathogens pose a substantial risk to global food security and the economic stability of potato farming communities.

Early blight, caused by *Alternaria solani*, is a common and destructive fungal disease affecting potato plants. The pathogen primarily targets leaves, stems, and tubers, leading to characteristic dark lesions with concentric rings. As the infection progresses, affected foliage may prematurely defoliate, reducing the plant's ability to photosynthesize and ultimately impacting tuber development.

The *Phytophthora infestans* causes late blight, a serious and highly infectious hazard to potato harvests. This devastating disease affects leaves, stems, and tubers, causing water-soaked lesions that can quickly lead to plant defoliation. Late blight is particularly notorious for its ability to spread rapidly during cool and damp weather conditions, posing a substantial risk to potato farming communities. The historical impact of late blight, famously linked to the Irish Potato Famine in the 19th century, underscores the urgency of effective preventive measures.

The consequences of early blight and late blight extend beyond the visible symptoms on potato plants. These diseases not only compromise the quality and quantity of the potato yield but also contribute to economic losses for farmers. Decreased yields, coupled with the cost of fungicides and other control measures, amplify the financial burden on agricultural communities. Additionally, the potential for post-harvest losses due to tuber rot further exacerbates the economic impact. Addressing these challenges is imperative for sustaining potato farming and ensuring global food security.

In response to the critical challenge posed by these two common diseases, we introduce an innovative solution. Our approach involves the development of a mobile app powered by Convolutional Neural Networks (CNNs) for early disease detection in potato crops. By harnessing advanced image analysis techniques, our mobile app aims to empower farmers with a rapid and accurate means of identifying these diseases. By detecting the type of disease, farmers can take informed actions, such as implementing timely and precise treatments, thereby minimizing economic losses. This user-friendly platform marks a transformative step towards revolutionizing agricultural practices, securing global food stability, and reinforcing the economic resilience of potato farming communities.

## CHAPTER 2

### LITERATURE SURVEY

Fizzah Arshad presents PLDPNet, a hybrid model for automatic prediction of potato diseases. By combining VGG model, Inception-V3 model and leveraging vision transformers, PLDPNet achieves an impressive 98.66% overall accuracy and 96.33% F1-score on a public potato leaf dataset. This novel framework demonstrates effective and accurate detection of potato crop diseases, making it a promising solution for practical applications.[2] Lakshmanarao employs Convolutional Neural Networks ("Convnets") for plant disease detection and classification, using the PlantVillage dataset with images of 15 classes from potato, pepper, and tomato plants. The study achieves notable accuracies of 95%, 98.3%, and 98.5% for disease detection in tomato, potato, and pepper plants, respectively. The results underscore the efficacy of Convnets in accurate plant leaf disease identification within the agricultural sector.[3]

Ahmed highlights the severe threat of Potato Late Blight (PLB) caused by *Phytophthora infestans* in global potato production, particularly in Pakistan where it can lead to 100% yield losses. Developing a predictive model based on two years of data, the study identifies significant factors such as temperature, humidity, rainfall, and wind speed influencing PLB development. This model offers insights into early disease prediction and management strategies amid the environmental challenges associated with PLB.[5] W.R. Henshall compares disease prediction models for potato late blight, including the sophisticated Shtienberg model. Analysing data from high and low disease risk areas, the study finds the Shtienberg model yields results similar to established models, showcasing its potential for optimizing fungicide application and reducing chemical use in potato farming.[6]

Kumar Sanjeev's study delves into the significance of potatoes in India, emphasizing their role as a low-cost energy source in human diets and industrial applications. Focusing on the major threats of Early Blight and Late Blight diseases to potato yields, the research utilizes a dataset of 2152 potato leaf images from Plant Village. Employing a Feed Forward Neural Network (FFNN) model, the classifier achieves an impressive 96.5% accuracy, showcasing its effectiveness in the early and accurate prediction of potato leaf diseases.[7]

Rabbia Mahum presents a novel deep learning algorithm for efficient detection and classification of four potato leaf diseases. Utilizing an improved Efficient DenseNet model and addressing imbalanced data, the algorithm achieves a high accuracy of 97.2%, surpassing existing models and marking a significant advancement in potato disease management.[8]

## CHAPTER 3

### SYSTEM ARCHITECTURE AND DESIGN

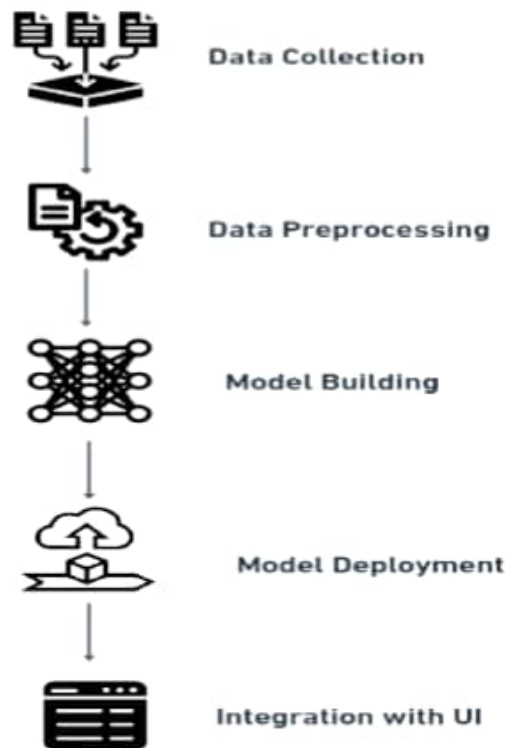


Fig.3.2 Architecture diagram of our approach

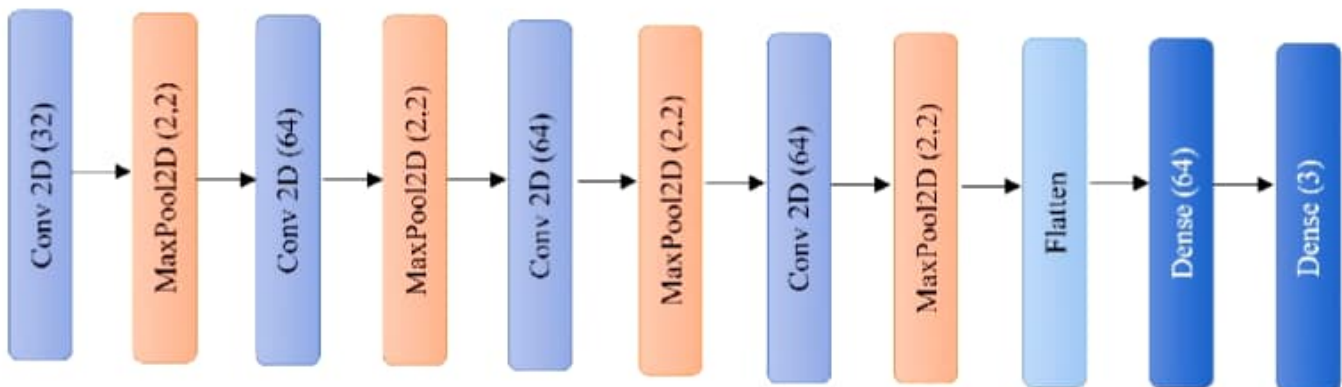
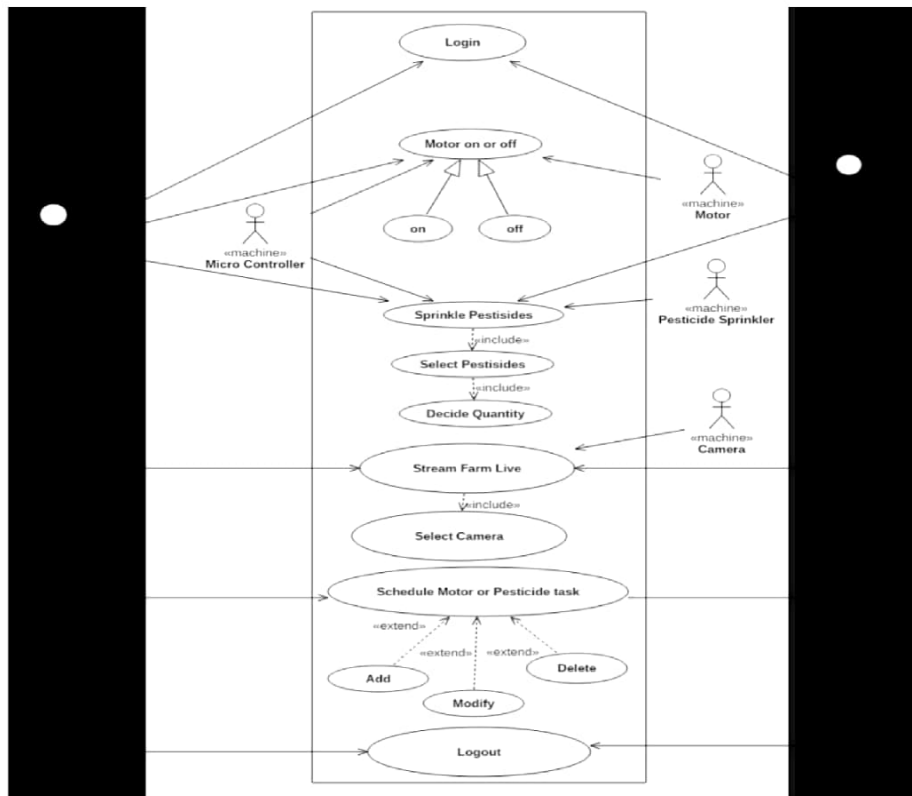


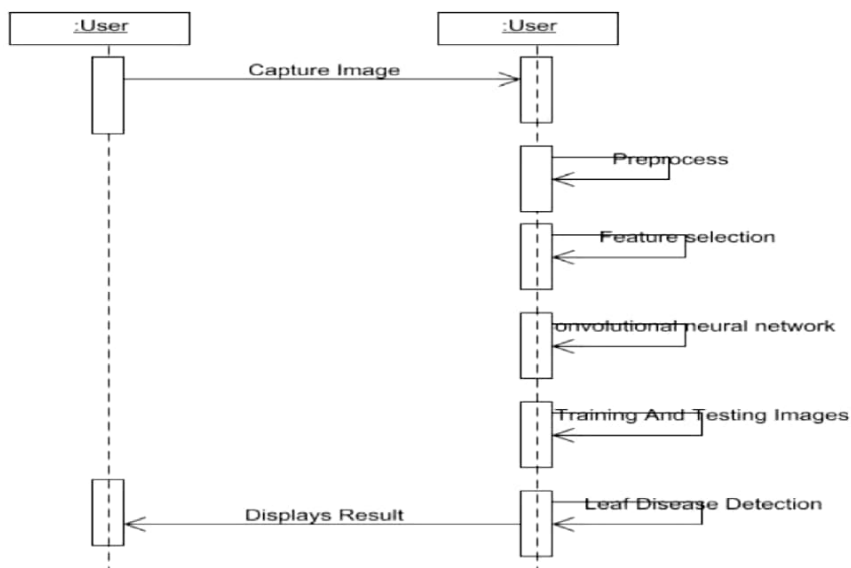
Fig.3.2 Architecture diagram of custom CNN model

# UML DIAGRAMS :-

## USE CASE DIAGRAM:

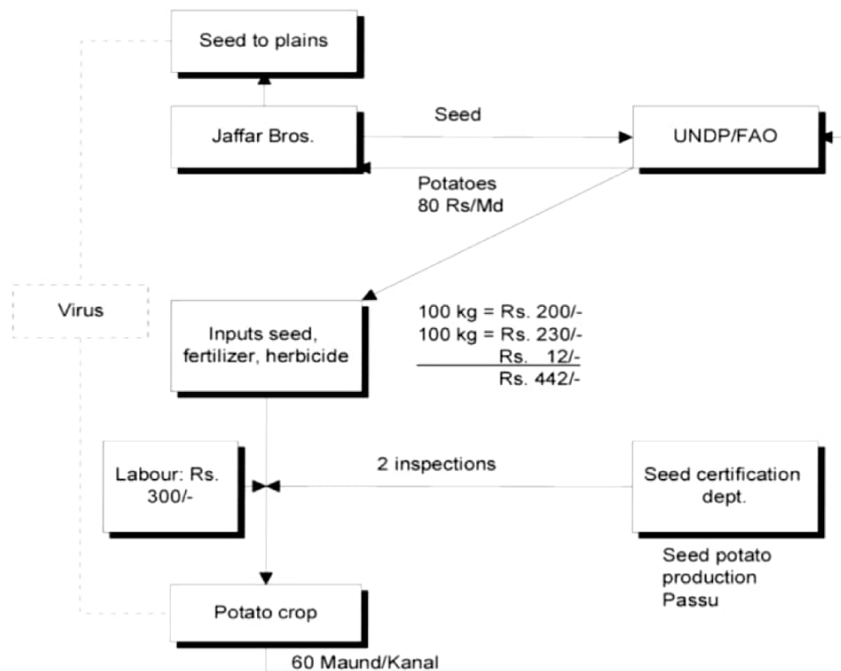


## SEQUENTIAL DIAGRAM:

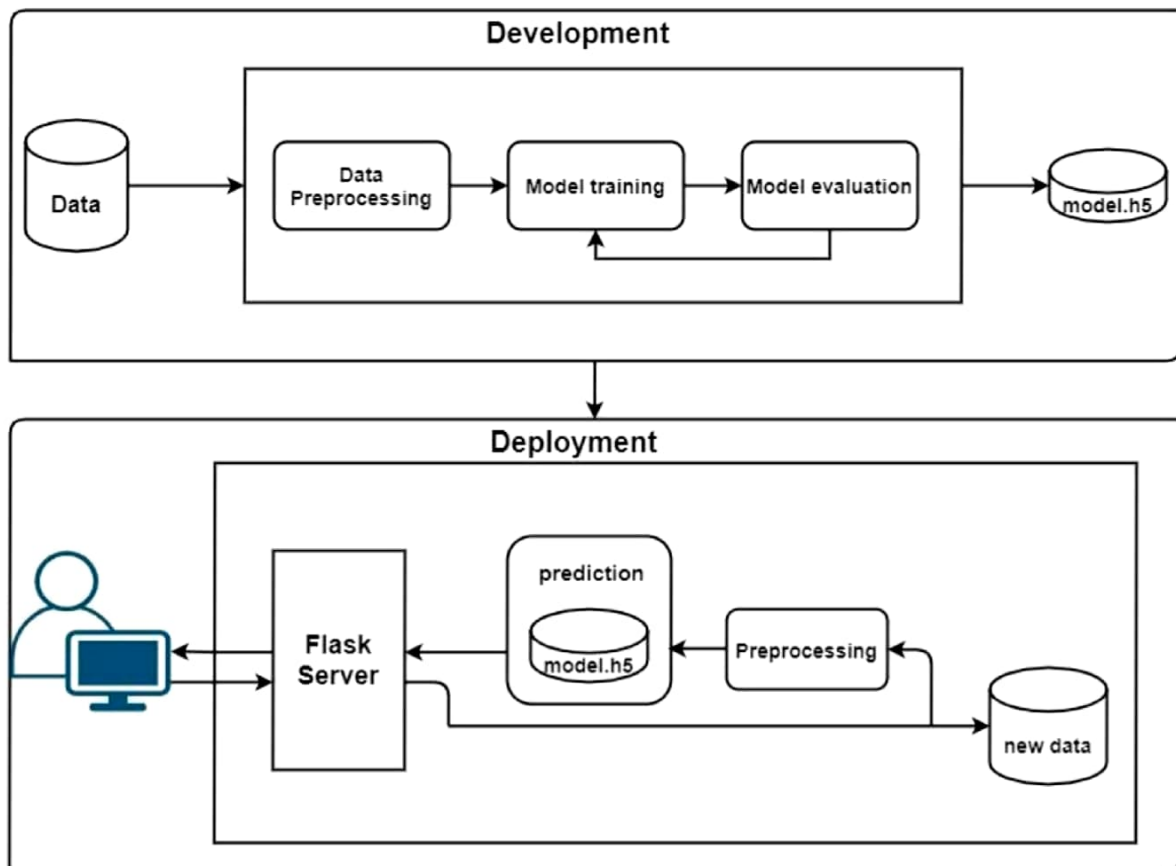




## FLOW DIAGRAM :-



## DEPLOYMENT DIAGRAM:-



## CHAPTER 4

### METHODOLOGY

Our methodology is designed to address the intricate task of potato leaf disease classification, a critical aspect for ensuring crop health and productivity. The approach encompasses a systematic pipeline, starting with the collection of a diverse dataset from "PlantVillage" on Kaggle, followed by meticulous preprocessing, augmentation, and utilization of Convolutional Neural Networks (CNNs) for model building. The final phase involves thorough evaluation, employing standard metrics to evaluate the model's performance. Each step is meticulously developed to enhance the model's accuracy and robustness in classifying potato leaf diseases.

#### A. DATA COLLECTION

The first phase of our methodology, we gathered a diverse dataset from "PlantVillage" on Kaggle. This dataset comprises high-resolution images of potato plants, each standardized to a size of 256x256 pixels. Within our dataset, we meticulously categorized the images into three classes: Healthy, Early Blight, and Late Blight.

#### B. DATA PREPROCESSING

The images were resized to a standardized dimension of 256x256 pixels. Subsequently, normalization was applied to bring pixel values to a common scale.

#### C. MODEL BUILDING

We proceed to construct a CNN model tailored for the classification of most commonly occur potato diseases. The architecture begins with Conv2D layers, which convolve over the input images, extracting essential features. These are followed by MaxPooling layers, which perform spatial reduction and retain critical information. The activation functions utilized in our model include the rectified linear unit to introduce non-linearity and softmax activation at the end for the final classification.

#### D. MODEL EVALUATION

The dataset, meticulously split into training, validation, and test sets, is leveraged to assess the model's proficiency. For optimization, we employed the Adam optimizer with a Sparse Categorical Entropy loss function, ensuring efficient training for our classification task. A batch size of 32 and 25 epochs were utilized to strike a balance between computational efficiency and comprehensive learning.

#### E. MODEL DEPLOYMENT

# DATA SET DESCRIPTION :-

Dataset description for revolutionizing potato farming, we would ideally need specific details about the type of data you're looking for. Here are some potential aspects of potato farming data that could be included in a dataset:

**Date:** Date of data collection or observation.

**Location:** Geographic coordinates or name of the region where the data was collected.

**Weather Conditions:** Temperature, humidity, precipitation, and other relevant weather parameters.

**Soil Type:** Description of soil characteristics such as composition, pH level, nutrient content, etc.

**Crop Growth Stage:** Stage of potato growth at the time of observation (e.g., planting, emergence, flowering, harvesting).

**Crop Health:** Health status of potato plants, including presence of pests, diseases, or nutrient deficiencies.

**Crop Yield:** Quantity of potatoes harvested per unit area (e.g., tons per hectare).

**Fertilizer Application:** Type and amount of fertilizer applied during the growing season.

**Pesticide Usage:** Type and frequency of pesticide applications, if any.

**Irrigation Practices:** Method and frequency of irrigation, including any water management strategies.

**Crop Variety:** Name or code of the potato variety being grown.

**Field Management Practices:** Any specific practices implemented to manage weeds, improve soil health, etc.

**Equipment Usage:** Types of machinery or equipment used for planting, cultivation, and harvesting.

**Market Prices:** Prices of potatoes at various stages of the supply chain, including farmgate prices and retail prices.

**Environmental Factors:** Any environmental factors affecting potato farming, such as climate change impacts or regulatory policies.

**Technology Adoption:** Adoption of modern technologies or innovations in potato farming, such as precision agriculture techniques or remote sensing.

**Economic Indicators:** Cost of production, profitability, and return on investment for potato farming operations.

**Socioeconomic Factors:** Socioeconomic variables influencing potato farming, such as land tenure systems or access to credit.

**Genetic Information:** Genetic traits or markers of potato varieties, including resistance to pests, diseases, or environmental stresses.

**Crop Rotation History:** History of previous crops grown on the land and its impact on potato yield and soil health.

**Carbon Footprint:** Measurement of greenhouse gas emissions associated with potato farming activities, including cultivation, transportation, and storage.

**Water Quality:** Analysis of water sources used for irrigation and its quality in terms of contaminants or salinity levels.

**Biodiversity:** Assessment of biodiversity within potato farming systems, including beneficial insects, wildlife habitats, and plant diversity.

**Post-Harvest Losses:** Quantity and causes of potato losses occurring after harvest, such as spoilage during storage or transportation.

**Value-Added Products:** Development and market demand for processed potato products such as chips, fries, and dehydrated flakes.

**Consumer Preferences:** Consumer preferences and trends related to potato products, including organic, locally sourced, or specialty varieties.

**Regulatory Compliance:** Compliance with local, national, and international regulations governing potato farming practices, including environmental standards and labor laws.

## CHAPTER 5

### CODING AND TESTING

```
import tensorflow as tf
from tensorflow.keras import models, layers
import matplotlib.pyplot as plt
from IPython.display import HTML
BATCH_SIZE = 32
IMAGE_SIZE = 256
CHANNELS=3
EPOCHS=50
import shutil
source_folder = '/content/plantvillage-dataset/plantvillage
dataset/color/Potato_healthy'
destination_folder = '/content/PlantVillage'
shutil.move(source_folder, destination_folder)
dataset = tf.keras.preprocessing.image_dataset_from_directory(
    "/content/PlantVillage",
    seed=123,
    shuffle=True,
    image_size=(IMAGE_SIZE, IMAGE_SIZE),
    batch_size=BATCH_SIZE
)
def get_dataset_partitions_tf(ds, train_split=0.8, val_split=0.1, test_split=0.1,
shuffle=True, shuffle_size=10000):
    assert (train_split + test_split + val_split) == 1
    ds_size = len(ds)
    if shuffle:
        ds = ds.shuffle(shuffle_size, seed=12)
    train_size = int(train_split * ds_size)
    val_size = int(val_split * ds_size)
    train_ds = ds.take(train_size)
    val_ds = ds.skip(train_size).take(val_size)
    test_ds = ds.skip(train_size).skip(val_size)
    return train_ds, val_ds, test_ds
train_ds, val_ds, test_ds = get_dataset_partitions_tf(dataset)
train_ds = train_ds.cache().shuffle(1000).prefetch(buffer_size=tf.data.AUTOTUNE)
val_ds = val_ds.cache().shuffle(1000).prefetch(buffer_size=tf.data.AUTOTUNE)
test_ds = test_ds.cache().shuffle(1000).prefetch(buffer_size=tf.data.AUTOTUNE)
resize_and_rescale = tf.keras.Sequential([
    layers.experimental.preprocessing.Resizing(IMAGE_SIZE, IMAGE_SIZE),
    layers.experimental.preprocessing.Rescaling(1./255),
])
data_augmentation = tf.keras.Sequential([
    layers.experimental.preprocessing.RandomFlip("horizontal_and_vertical"),
    layers.experimental.preprocessing.RandomRotation(0.2),
])
```

```

train_ds = train_ds.map(
    lambda x, y: (data_augmentation(x, training=True), y)
).prefetch(buffer_size=tf.data.AUTOTUNE)
len(train_ds)
input_shape = (BATCH_SIZE, IMAGE_SIZE, IMAGE_SIZE, CHANNELS)
n_classes = 3

model = models.Sequential([
    resize_and_rescale,
    layers.Conv2D(32, kernel_size = (3,3), activation='relu',
input_shape=input_shape),
    layers.MaxPooling2D((2, 2)),
    layers.Conv2D(64, kernel_size = (3,3), activation='relu'),
    layers.MaxPooling2D((2, 2)),
    layers.Conv2D(64, kernel_size = (3,3), activation='relu'),
    layers.MaxPooling2D((2, 2)),
    layers.Conv2D(64, (3, 3), activation='relu'),
    layers.MaxPooling2D((2, 2)),
    layers.Conv2D(64, (3, 3), activation='relu'),
    layers.MaxPooling2D((2, 2)),
    layers.Conv2D(64, (3, 3), activation='relu'),
    layers.MaxPooling2D((2, 2)),
    layers.Flatten(),
    layers.Dense(64, activation='relu'),
    layers.Dense(n_classes, activation='softmax'),
])
model.build(input_shape=input_shape)
model.compile(
    optimizer='adam',
    loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=False),
    metrics=['accuracy']
)
history = model.fit(
    train_ds,
    batch_size=BATCH_SIZE,
    validation_data=val_ds,
    verbose=1,
    epochs=20,
)
scores = model.evaluate(test_ds)
print(scores)
model.save("/content/MyModel SRM.h5")

```



PlantDisease11.ipynb X

C:\Users\Abhi&gt; AppData\Local\Microsoft\Windows\INetCache\IE\8CP09MF8&gt; PlantDisease11.ipynb&gt; Potato Disease Classification

+ Code + Markdown ▶ Run All ⏮ Restart ⏭ Clear All Outputs ⏮ Go To | Variables Outline

Python 3.12.1

# Potato Disease Classification

```
import tensorflow as tf
from tensorflow.keras import models, layers
import matplotlib.pyplot as plt
from IPython.display import HTML

#pip install opendatasets
import opendatasets as od
#ponnurumahesh - 5d7a32f441b2590379a895c2c402b0d9
od.download("https://www.kaggle.com/datasets/abdallahalidev/plantvillage-dataset")
```

Python

```
BATCH_SIZE = 32
IMAGE_SIZE = 256
CHANNELS=3
EPOCHS=50
```

Python

```
dataset = tf.keras.preprocessing.image_dataset_from_directory(
    "PlantVillage",
    seed=123,
    shuffle=True,
    image_size=(IMAGE_SIZE, IMAGE_SIZE),
    batch_size=BATCH_SIZE)
```

Python

Open a Remote Window

52 files belonging to 3 classes.

Go Live



Search

ENG  
IN21:10  
01-05-2024

```
def predict(model, img):
    img_array = tf.keras.preprocessing.image.img_to_array(images[i].numpy())
    img_array = tf.expand_dims(img_array, 0)

    predictions = model.predict(img_array)

    predicted_class = class_names[np.argmax(predictions[0])]
    confidence = round(100 * (np.max(predictions[0])), 2)
    return predicted_class, confidence
```

[45]

Python

```
plt.figure(figsize=(15, 15))
for images, labels in test_ds.take(1):
    for i in range(9):
        ax = plt.subplot(3, 3, i + 1)
        plt.imshow(images[i].numpy().astype("uint8"))

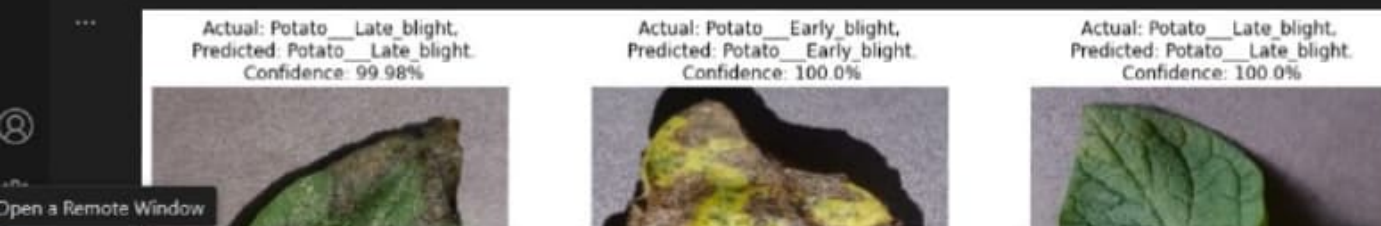
        predicted_class, confidence = predict(model, images[i].numpy())
        actual_class = class_names[labels[i]]

        plt.title(f"Actual: {actual_class},\n Predicted: {predicted_class},\n Confidence: {confidence}%")

        plt.axis("off")
```

[46]

Python



Open a Remote Window



Potato\_\_Early\_blight



Potato\_\_Early\_blight



Potato\_\_Early\_blight



Potato\_\_Late\_blight



Potato\_\_Early\_blight



Potato\_\_Early\_blight



Potato\_\_Late\_blight



Potato\_\_Early\_blight



Potato\_\_Late\_blight



Potato\_\_Early\_blight



Potato\_\_Early\_blight



Potato\_\_Early\_blight



FileEditSelectionViewGoRunTerminalHelp

←→Search

PlantDisease[1].ipynb X

C:\Users> Akhii > AppData > Local > Microsoft > Windows > INetCache > IE > 8CP09MF8 > PlantDisease[1].ipynb > M4Potato Disease Classification

+ Code + Markdown | Run All Restart Clear All Outputs Go To Variables Outline ... Python 3.12.1

[17]

train\_ds, val\_ds, test\_ds = get\_dataset\_partitions\_tf(dataset)

Python

[18]

len(train\_ds)

Python

...

54

[19]

len(val\_ds)

Python

...

6

[20]

len(test\_ds)

Python

...

8

[21]

train\_ds = train\_ds.cache().shuffle(1000).prefetch(buffer\_size=tf.data.AUTOTUNE)  
val\_ds = val\_ds.cache().shuffle(1000).prefetch(buffer\_size=tf.data.AUTOTUNE)  
test\_ds = test\_ds.cache().shuffle(1000).prefetch(buffer\_size=tf.data.AUTOTUNE)

Python

resize\_and\_rescale = tf.keras.Sequential([  
layers.experimental.preprocessing.Resizing(IMAGE\_SIZE, IMAGE\_SIZE),  
layers.experimental.preprocessing.Rescaling(1./255),

Open a Remote Window

0 0 12 0

Go Live

21:11  
01-05-2024

FileEditSelectionViewGoRunTerminalHelp

←→Search

PlantDisease[1].ipynb X

C:\Users> Akhii > AppData > Local > Microsoft > Windows > INetCache > IE > 8CP09MF8 > PlantDisease[1].ipynb > M4Potato Disease Classification

+ Code + Markdown | ▶ Run All ⏹ Restart ≡ Clear All Outputs ⌂ Go To 📄 Variables ≡ Outline ...

Python 3.12.1

val\_ds = test\_ds.take(6)  
len(val\_ds)

[14] Python

... 6

test\_ds = test\_ds.skip(6)  
len(test\_ds)

[15] Python

... 8

def get\_dataset\_partitions\_tf(ds, train\_split=0.8, val\_split=0.1, test\_split=0.1, shuffle=True, shuffle\_size=10000):  
 assert (train\_split + test\_split + val\_split) == 1  
  
 ds\_size = len(ds)  
  
 if shuffle:  
 ds = ds.shuffle(shuffle\_size, seed=12)  
  
 train\_size = int(train\_split \* ds\_size)  
 val\_size = int(val\_split \* ds\_size)  
  
 train\_ds = ds.take(train\_size)  
 val\_ds = ds.skip(train\_size).take(val\_size)  
 test\_ds = ds.skip(train\_size).skip(val\_size)  
  
 return train\_ds, val\_ds, test\_ds

[16] Python

Open a Remote Window

0 0 12 0

Go Live

Search

21:11  
01-05-2024

File Edit Selection View Go Run Terminal Help

Search

PlantDisease11.ipynb X

C:\Users\Abhi> AppData\Local\Microsoft\Windows\INetCache\IE\8CP09MF8> PlantDisease11.ipynb> Potato Disease Classification

+ code + Markdown Run All Restart Clear All Outputs Go To Variables Outline

Python 3.12.1

123

train\_ds, val\_ds, test\_ds = get\_dataset\_partitions\_tf(dataset)

Python

124

len(train\_ds)

Python

125

len(val\_ds)

Python

126

len(test\_ds)

Python

127

train\_ds = train\_ds.cache().shuffle(1000).prefetch(buffer\_size=tf.data.AUTOTUNE)

val\_ds = val\_ds.cache().shuffle(1000).prefetch(buffer\_size=tf.data.AUTOTUNE)

test\_ds = test\_ds.cache().shuffle(1000).prefetch(buffer\_size=tf.data.AUTOTUNE)

Python

128

resize\_and\_rescale = tf.keras.Sequential([

layers.experimental.preprocessing.Resizing(IMAGE\_SIZE, IMAGE\_SIZE),

layers.experimental.preprocessing.Rescaling(1./255),

Open a Remote Window

Go Live

21:11 01-05-2024

```

import tensorflow as tf
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
import numpy as np
import os

model = load_model('/content/MyModel SRM.h5')

# Define a function to make predictions
def predict_image(image_path):
    img = image.load_img(image_path, target_size=(256, 256))
    image1 = image.img_to_array(img)
    img = np.expand_dims(image1, axis=0)
    prediction = model.predict(img)
    print(prediction)
    print("predicted label:", class_names[np.argmax(prediction[0])])

# Test the model on an example image
test_image_path = '/content/sample_data/late_blight_1.jpeg' # change this to your image path
predicted_class = predict_image(test_image_path)

```



PlantDisease11.py:nb X

C:\Users&gt; Abhi\ &gt; AppData &gt; Local &gt; Microsoft &gt; Windows &gt; INetCache &gt; IE &gt; SCPOSMB &gt; PlantDisease11.py:nb &gt; M4 Potato Disease Classification &gt; plt.figure(figsize=(15,15))

Code + Markdown Run All Revert Clear All Outputs Go To Variables Outline

Python 3.12.1

import os

model = load\_model('/content/MyModel SRM.h5')

# Define a function to make predictions

def predict\_image(image\_path):

img = image.load\_img(image\_path, target\_size=(256, 256))

image1 = image.img\_to\_array(img)

img = np.expand\_dims(img, axis=0)

prediction = model.predict(img)

print(prediction)

print("predicted label:", class\_names[np.argmax(prediction[0])])

# Test the model on an example image

test\_image\_path = '/content/sample\_data/late\_blight\_1.jpeg' # Change this to your image path

predicted\_class = predict\_image(test\_image\_path)

Python

import numpy as np

for images\_batch, labels\_batch in test\_ds.take(1):

first\_image = images\_batch[0].numpy().astype('uint8')

first\_label = labels\_batch[0].numpy()

print("first image to predict")

plt.imshow(first\_image)

print("actual label:", class\_names[first\_label])

batch\_prediction = model.predict(images\_batch)

print(batch\_prediction)

print("predicted label:", class\_names[np.argmax(batch\_prediction[0])])

Python

Go Live

ENG  
IN

21:13

01-05-2024

`model.summary()`

model: "sequential\_2"

Layer (type)	Output Shape	Param #
sequential (Sequential)	(32, 256, 256, 3)	0
conv2d (Conv2D)	(32, 254, 254, 32)	836
max_pooling2d (MaxPooling2D)	(32, 127, 127, 32)	0
conv2d_1 (Conv2D)	(32, 125, 125, 64)	18496
max_pooling2d_1 (MaxPooling2D)	(32, 62, 62, 64)	0
conv2d_2 (Conv2D)	(32, 60, 60, 64)	36928
max_pooling2d_2 (MaxPooling2D)	(32, 30, 30, 64)	0
conv2d_3 (Conv2D)	(32, 28, 28, 64)	36928
max_pooling2d_3 (MaxPooling2D)	(32, 14, 14, 64)	0
conv2d_4 (Conv2D)	(32, 12, 12, 64)	36928
max_pooling2d_4 (MaxPooling2D)	(32, 6, 6, 64)	0
...		
Total params: 183,747		
Trainable params: 183,747		
Non-trainable params: 0		

File Edit Selection View Go Run Terminal Help

Search

PlantDisease11.ipynb X

C:\Users\Abhi\AppData\Local\Microsoft\Windows\INetCache\IE\8CP09MF8\PlantDisease11.ipynb> Potato Disease Classification

+ Code + Markdown Run All Restart Clear All Outputs Go To Variables Outline

Python 3.12.1

history.history.keys()

dict\_keys(['loss', 'accuracy', 'val\_loss', 'val\_accuracy'])

type(history.history['loss'])

list

len(history.history['loss'])

50

history.history['loss'][:5] # show loss for first 5 epochs

[0.8801848292350769,  
0.6833139228820801,  
0.3646925389766693,  
0.2776017189025879,  
0.24480397999286652]

acc = history.history['accuracy']  
val\_acc = history.history['val accuracy']  
history.history['loss']

Open a Remote Window

Go Live

21:12  
01-05-2024



```
acc = history.history['accuracy']
val_acc = history.history['val_accuracy']

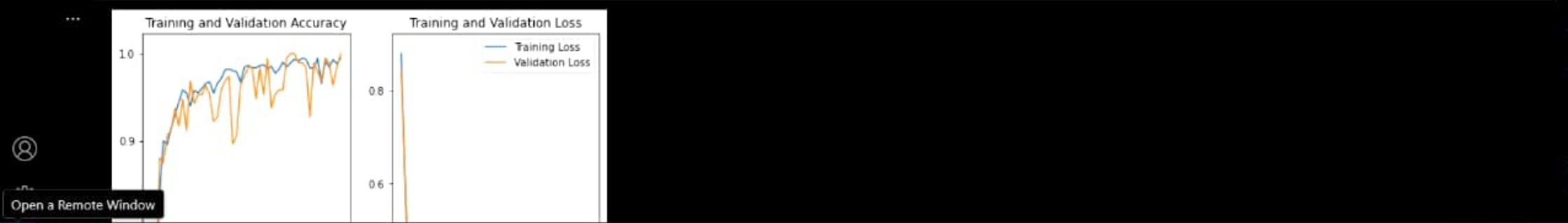
loss = history.history['loss']
val_loss = history.history['val_loss']
```

[42] Python

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

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

[43] Python



```
.. Found 2152 files belonging to 3 classes.
```

```
class_names = dataset.class_names
class_names
```

```
[6]
```

```
.. ['Potato__Early_blight', 'Potato__Late_blight', 'Potato__healthy']
```

```
> v
```

```
for image_batch, labels_batch in dataset.take(1):
    print(image_batch.shape)
    print(labels_batch.numpy())
```

```
[7]
```

```
.. (32, 256, 256, 3)
```

```
[1 1 1 0 0 0 0 0 1 1 1 1 0 1 0 1 1 1 0 1 0 1 0 0 1 0 0 1 1 2 0 0]
```

```
plt.figure(figsize=(10, 10))
for image_batch, labels_batch in dataset.take(1):
    for i in range(12):
        ax = plt.subplot(3, 4, i + 1)
        plt.imshow(image_batch[i].numpy().astype("uint8"))
        plt.title(class_names[labels_batch[i]])
        plt.axis("off")
```

```
[8]
```

File Edit Selection View Go Run Terminal Help Search

PlantDisease[1].ipynb X

C:\Users> Akhii > AppData > Local > Microsoft > Windows > INetCache > IE > 8CP09MF8 > PlantDisease[1].ipynb > M\*Potato Disease Classification

+ Code + Markdown | Run All Restart Clear All Outputs Go To | Variables Outline ... Python 3.12.1

```
model.compile(
    optimizer='adam',
    loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=False),
    metrics=['accuracy']
)
```

[32] Python

```
history = model.fit(
    train_ds,
    batch_size=BATCH_SIZE,
    validation_data=val_ds,
    verbose=1,
    epochs=50,
)
```

[33] Python

... Epoch 1/50  
54/54 [=====] - 20s 255ms/step - loss: 0.8802 - accuracy: 0.5341 - val\_loss: 0.8462 - val\_accuracy: 0.5938  
Epoch 2/50  
54/54 [=====] - 11s 196ms/step - loss: 0.6033 - accuracy: 0.7396 - val\_loss: 0.6225 - val\_accuracy: 0.6979  
Epoch 3/50  
54/54 [=====] - 9s 172ms/step - loss: 0.3647 - accuracy: 0.8403 - val\_loss: 0.3065 - val\_accuracy: 0.8802  
Epoch 4/50  
54/54 [=====] - 10s 176ms/step - loss: 0.2776 - accuracy: 0.8999 - val\_loss: 0.2702 - val\_accuracy: 0.8750  
Epoch 5/50  
54/54 [=====] - 10s 179ms/step - loss: 0.2448 - accuracy: 0.8953 - val\_loss: 0.1857 - val\_accuracy: 0.9062  
Epoch 6/50  
54/54 [=====] - 9s 174ms/step - loss: 0.2020 - accuracy: 0.9144 - val\_loss: 0.2987 - val\_accuracy: 0.9115  
Epoch 7/50  
54/54 [=====] - 10s 185ms/step - loss: 0.1751 - accuracy: 0.9288 - val\_loss: 0.1854 - val\_accuracy: 0.9375  
Epoch 8/50  
54/54 [=====] - 10s 180ms/step - loss: 0.1436 - accuracy: 0.9444 - val\_loss: 0.2273 - val\_accuracy: 0.9167  
Epoch 9/50  
54/54 [=====] - 10s 175ms/step - loss: 0.1128 - accuracy: 0.9583 - val\_loss: 0.1425 - val\_accuracy: 0.9479

Open a Remote Window

0 12 0

Go Live

ENG IN 21:12 01-05-2024

File Edit Selection View Go Run Terminal Help

← → Search

PlantDisease[1].ipynb X

C:\Users> Akhii > AppData > Local > Microsoft > Windows > INetCache > IE > 8CP09MF8 > PlantDisease[1].ipynb > M4Potato Disease Classification

+ Code + Markdown | Run All Restart Clear All Outputs Go To Variables Outline ...

Python 3.12.1

```
import numpy as np
for images_batch, labels_batch in test_ds.take(1):

    first_image = images_batch[0].numpy().astype('uint8')
    first_label = labels_batch[0].numpy()

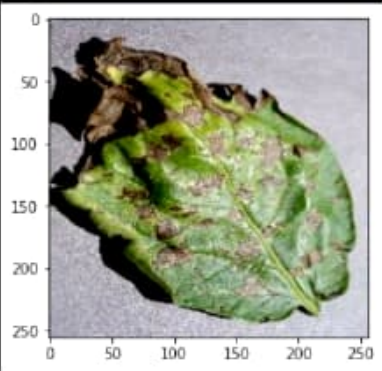
    print("first image to predict")
    plt.imshow(first_image)
    print("actual label:", class_names[first_label])

    batch_prediction = model.predict(images_batch)
    print("predicted label:", class_names[np.argmax(batch_prediction[0])])
```

[44]

Python

... first image to predict  
actual label: Potato\_Early\_blight  
predicted label: Potato\_Early\_blight

... 

Open a Remote Window

0 0 12 0

Go Live

21:12  
01-05-2024

File Edit Selection View Go Run Terminal Help

Search

PlantDisease[1].ipynb X

C:\Users> Akhii > AppData > Local > Microsoft > Windows > INetCache > IE > 8CP09MF8 > PlantDisease[1].ipynb > M4Potato Disease Classification

+ Code + Markdown | Run All Restart Clear All Outputs Go To Variables Outline ... Python 3.12.1

```
history.history.keys()
```

[38] Python

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

```
type(history.history['loss'])
```

[39] Python

```
list
```

```
len(history.history['loss'])
```

[40] Python

```
50
```

```
history.history['loss'][:5] # show loss for first 5 epochs
```

[41] Python

```
[0.8801848292350769,
0.6033139228820801,
0.3646925389766693,
0.2776017189025879,
0.24480397999286652]
```

```
acc = history.history['accuracy']
val_acc = history.history['val_accuracy']
```

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

Open a Remote Window

Go Live

ENG IN 21:12 01-05-2024



File Edit Selection View Go Run Terminal Help

← → Search

PlantDisease[1].ipynb X

C:\Users> Akhii > AppData > Local > Microsoft > Windows > INetCache > IE > 8CP09MF8 > PlantDisease[1].ipynb > M4Potato Disease Classification

+ Code + Markdown | ▶ Run All ⌂ Restart ≡ Clear All Outputs ⌂ Go To 📄 Variables ≡ Outline ...

Python 3.12.1

```
import numpy as np
for images_batch, labels_batch in test_ds.take(1):


    first_image = images_batch[0].numpy().astype('uint8')
    first_label = labels_batch[0].numpy()

    print("first image to predict")
    plt.imshow(first_image)
    print("actual label:", class_names[first_label])

    batch_prediction = model.predict(images_batch)
    print("predicted label:", class_names[np.argmax(batch_prediction[0])])
```

[44] Python

... first image to predict  
actual label: Potato\_\_Early\_blight  
predicted label: Potato\_\_Early\_blight

... 

Open a Remote Window

0 0 12 0

Go Live

21:12  
01-05-2024

Windows Taskbar

Search

Taskbar Icons: File Explorer, Microsoft Edge, Google Chrome, VS Code, etc.

System Tray: ENG IN, Network, Volume, Date/Time

## CHAPTER 6

### SCREENSHOTS AND RESULTS

```
54/54 [=====] - 10s 178ms/step - loss: 0.0353 - accuracy: 0.9896 - val_loss: 0.0092 - val_accuracy: 1.0000
Epoch 38/50
54/54 [=====] - 9s 173ms/step - loss: 0.0206 - accuracy: 0.9936 - val_loss: 0.0079 - val_accuracy: 1.0000
Epoch 39/50
54/54 [=====] - 9s 171ms/step - loss: 0.0307 - accuracy: 0.9913 - val_loss: 0.0209 - val_accuracy: 0.9896
Epoch 40/50
54/54 [=====] - 9s 175ms/step - loss: 0.0143 - accuracy: 0.9948 - val_loss: 0.0240 - val_accuracy: 0.9896
Epoch 41/50
54/54 [=====] - 9s 170ms/step - loss: 0.0196 - accuracy: 0.9936 - val_loss: 0.0441 - val_accuracy: 0.9844
Epoch 42/50
54/54 [=====] - 9s 173ms/step - loss: 0.0382 - accuracy: 0.9832 - val_loss: 0.2912 - val_accuracy: 0.9271
Epoch 43/50
54/54 [=====] - 9s 172ms/step - loss: 0.0416 - accuracy: 0.9832 - val_loss: 0.0425 - val_accuracy: 0.9896
Epoch 44/50
54/54 [=====] - 9s 171ms/step - loss: 0.0162 - accuracy: 0.9948 - val_loss: 0.0567 - val_accuracy: 0.9792
Epoch 45/50
54/54 [=====] - 9s 170ms/step - loss: 0.0990 - accuracy: 0.9653 - val_loss: 0.0892 - val_accuracy: 0.9688
Epoch 46/50
54/54 [=====] - 9s 171ms/step - loss: 0.0243 - accuracy: 0.9919 - val_loss: 0.0174 - val_accuracy: 0.9948
Epoch 47/50
54/54 [=====] - 9s 170ms/step - loss: 0.0476 - accuracy: 0.9844 - val_loss: 0.0217 - val_accuracy: 0.9896
Epoch 48/50
54/54 [=====] - 9s 170ms/step - loss: 0.0184 - accuracy: 0.9931 - val_loss: 0.1227 - val_accuracy: 0.9635
Epoch 49/50
54/54 [=====] - 10s 184ms/step - loss: 0.0298 - accuracy: 0.9884 - val_loss: 0.0528 - val_accuracy: 0.9844
Epoch 50/50
54/54 [=====] - 11s 196ms/step - loss: 0.0189 - accuracy: 0.9948 - val_loss: 0.0064 - val_accuracy: 1.0000
```

```
scores = model.evaluate(test_ds)
```

```
8/8 [=====] - 1s 14ms/step - loss: 0.0063 - accuracy: 1.0000
```

Fig.6.1 Training and validation accuracy

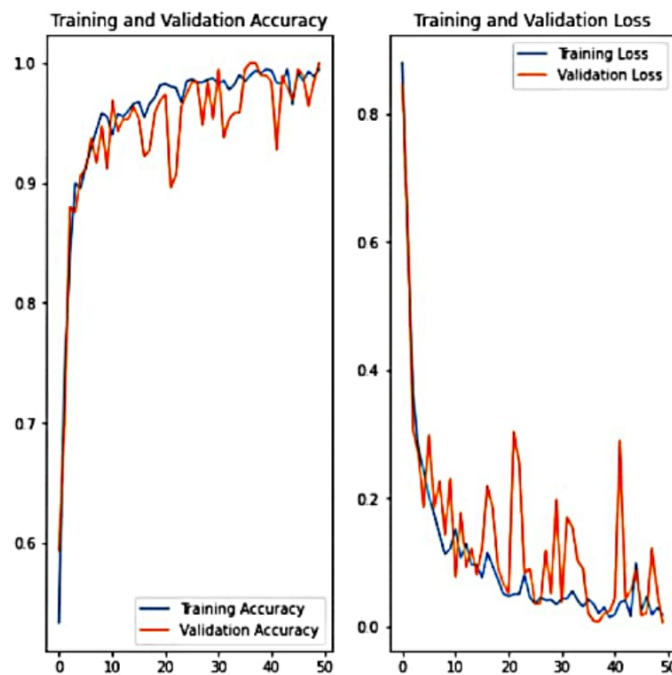


Fig.6.2 Training and Validation Graphs

[Grab your reader's attention with a great quote from the document or use this space to emphasize a key point. To place this text box anywhere on the page, just drag it.]



## CHAPTER 6

Actual: Potato\_\_Late\_blight,  
Predicted: Potato\_\_Late\_blight.  
Confidence: 99.98%



Actual: Potato\_\_Early\_blight,  
Predicted: Potato\_\_Early\_blight.  
Confidence: 100.0%



Actual: Potato\_\_Late\_blight,  
Predicted: Potato\_\_Late\_blight.  
Confidence: 100.0%



Actual: Potato\_\_Late\_blight,  
Predicted: Potato\_\_Late\_blight.  
Confidence: 100.0%



Actual: Potato\_\_Late\_blight,  
Predicted: Potato\_\_Late\_blight.  
Confidence: 99.92%



Actual: Potato\_\_Early\_blight,  
Predicted: Potato\_\_Early\_blight.  
Confidence: 99.87%



Actual: Potato\_\_Late\_blight,  
Predicted: Potato\_\_Late\_blight.  
Confidence: 99.88%



Actual: Potato\_\_healthy,  
Predicted: Potato\_\_healthy.  
Confidence: 99.53%



Actual: Potato\_\_Late\_blight,  
Predicted: Potato\_\_Late\_blight.  
Confidence: 88.21%



### 6.3 Testing Sample Images



## **CHAPTER 7**

### **CONCLUSION AND FUTURE ENHANCEMENTS**

#### **6.1. CONCLUSION**

In conclusion, our research marks a substantial leap forward in the realm of potato crop disease management. Achieving an outstanding validation accuracy of 99.59%, our Convolutional Neural Network (CNN) model surpasses conventional methods, addressing a critical gap in agricultural research. The innovative integration of this model into a user-friendly React Native mobile application empowers farmers with a practical tool for timely disease prediction. This fusion of advanced technology and accessible applications not only enhances precision in agriculture but also contributes to the democratization of agricultural innovation. By significantly improving disease identification accuracy and providing practical solutions for farmers, our research promises to revolutionize potato farming practices, reduce economic losses, and fortify global food security.

#### **6.2. FUTURE ENHANCEMENTS**

In future iterations, our research envisions several enhancements to further bolster the effectiveness and accessibility of potato crop disease management. Expanding the CNN model to accommodate multi-class classification would offer a more comprehensive assessment of crop health, while integrating real-time monitoring capabilities into the mobile application could provide farmers with timely alerts and insights. Customized disease prediction models tailored to specific geographical regions, coupled with an interactive decision support system, would empower farmers with personalized recommendations and best practices. Moreover, incorporating community engagement features and feedback mechanisms within the application could foster knowledge-sharing and continuous improvement. Strategic partnerships with agricultural stakeholders and the integration of additional machine learning algorithms could further optimize disease prediction accuracy and scale up the adoption of the mobile application, promising to revolutionize potato farming practices worldwide.

## REFERENCES

- [1] Sharma, Priyanka, B. K. Singh, and R. P. Singh. "Prediction of potato late blight disease based upon weather parameters using artificial neural network approach.", 9th International Conference on Computing, Communication and Networking Technologies, 2018.
- [2] Arshad F, Mateen M, Hayat S, Wardah M, Al-Huda Z, Gu YH, Al-antari MA. "PLDPNet: End-to-end hybrid deep learning framework for potato leaf disease prediction.", Alexandria Engineering Journal. 2023, pp.406-18.
- [3] Lakshmanarao, A., M. Raja Babu, and T. Srinivasa Ravi Kiran. "Plant disease prediction and classification using deep learning ConvNets.", 2021 International Conference on Artificial Intelligence and Machine Vision, 2021.
- [4] Van der Waals, J. E., Lise Korsten, and T. A. S. Aveling. "A review of early blight of potato." African Plant Protection 7.2, 2021, pp.91-102.
- [5] Ahmed N, Khan MA, Khan NA, Ali MA. "Prediction of potato late blight disease based upon environmental factors in Faisalabad. Pakistan.", J. Plant Pathol. Microbiol. S, 2015.
- [6] Henshall WR, Shtienberg D, Beresford RM. "A new potato late blight disease prediction model and its comparison with two previous models.", New Zealand Plant Protection, 2006, pp.150-154.
- [7] Sanjeev K, Gupta NK, Jeberson W, Paswan S. "Early prediction of potato leaf diseases using ANN classifier.", Oriental Journal of Computer Science and Technology, 2021, pp.129-134.
- [8] Mahum R, Munir H, Mughal ZU, Awais M, Sher Khan F, Saqlain M, Mahamad S, Tlili I., "A novel framework for potato leaf disease detection using an efficient deep learning model.", Human and Ecological Risk Assessment: An International Journal, 2023, 29(2), 303-326.