A

GROUP PROJECT REPORT

ON

POTATO PLANT DISEASE DETECTION USING DEEP LEARNING

Dissertation Submitted In Partial Fulfillment Of The Requirement For

The Award Of

BACHELOR OF TECHNOLOGY

IN

INFORMATION TECHNOLOGY

By

Meghana Reddy Mynampati(20311A1234)

Chelukali Ashish Kumar(20311A1243)

Panga Saikiran(21315A1205)

DEPARTMENT OF INFORMATION TECHNOLOGY

SREENIDHI INSTITUTION OF SCIENCE & TECHNOLOGY

(An Autonomous Institution)

(AFFILIATED TO JNTU, HYDERABAD)

Yamnampet, Ghatkesar, R.R. Dist, Hyd-501301.

2023

<u>**POTATO PLANT DISEASE DETECTION USING DEEP LEARNING**</u>

Dissertation submitted in partial fulfillment of the Requirement for the award of

BACHELOR OF TECHNOLOGY

IN

INFORMATION TECHNOLOGY

By

Meghana Reddy Mynampati(20311A1234)

Chelukali Ashish Kumar(20311A1243)

Panga Saikiran(21315A1205)

Under the guidance of

Mr. S. Maruthi

Associate Professor

Department of IT

DEPARTMENT OF INFORMATION TECHNOLOGY

SREENIDHI INSTITUTION OF SCIENCE AND TECHNOLOGY

(An Autonomous Institution)

(AFFILIATED TO JNTU, HYDERABAD)

Yamnampet, Ghatkesar, R.R. Dist, Hyd-501301

2023

DEPARTMENT OF INFORMATION TECHNOLOGY

SREENIDHI INSTITUTION OF SCIENCE AND TECHNOLOGY

(An Autonomous Institution)

(Affiliated to JNT University; ISO Certified 9001:2000)

Yamnampet, Ghatkesar, Hyderabad-501301

Ph.Nos : 08415-200597, 08415-325444, 9395533303

CERTIFICATE

This is to certify that the Dissertation entitled "**POTATO PLANT DISEASE DETECTION USING DEEP LEARNING**" is bonafide workdone and submitted by Meghana Reddy Mynampati(20311A1234)**,** Chelukali Ashish Kumar(20311A1243) ,Panga Saikiran(21315A1205) in partial fulfillment of the requirement for the award of Degree of **Bachelor of Technology in Information Technology,** SREENIDHI INSTITUTE OF SCIENCE AND TECHNOLOGY**, Affiliated to Jawaharlal Nehru Technological University, Hyderabad** is a record of bonafide work carried out by us under the guidance and supervision from **January 2023** to **June 2023.**

The results presented in this dissertation have been verified and are found to be satisfactory. The results embodied in this dissertation have not been submitted to any other university for the awardof any other degree or diploma**.**

Project Co-ordinator
P. SREEDHAR
Associate Professor & Associate Head,
Department of IT, SNIST, Hyderabad

Head of the Department
Dr. SUNIL BHUTADA
Department of IT,
SNIST, Hyderabad

Project Internal Guide
S. MARUTHI
Associate Professor
Department of IT, SNIST, Hyderabad

ACKNOWLEDGEMENT

I thank **Dr**. **SUNIL BHUTADA**, **Head of the Department of IT** and **Mr. S. MARUTHI, Associate Professor** for supporting us as our **Internal Guide** and providing seamless knowledge over the past one year, and also for providing right suggestion at every phase of the development of our project.

I extend my profound gratitude to thank **Mr. P. Sreedhar**, **(Project Cordinator)**, Department of IT, SNIST, for his valuable suggestion and support throughout the course.

I express a whole hearted gratitude to **Dr. CV TOMY, Director,** and **Dr. Shiva Reddy, Principal, Sreenidhi Institute of Science and Technology** for providing us the conducive environment for carrying through our academic schedules and projects with ease.

I convey my heartful thanks to my Parents, friends, Technical and Non-Technical staff of the college for their constant support in the successful completion of the project.

Meghana Reddy Mynampati(20311A1234)

Chelukali Ashish Kumar(20311A1243)

Panga Saikiran(21315A1205)

POTATO PLANT DISEASE DETECTION USING DEEP LEARNING

ABSTRACT

The identification of plant disease in potato plant is an imperative part of crop monitoring systems. Computer vision and deep learning (DL) techniques have been proven to be state-of-the-art to address various agricultural problems. This research performed the complex tasks of localization and classification of the disease in potato plant leaves. Deep Learning rich libraries and user as well as developer friendly environment to work with, all these qualities make Deep Learning as the favourable method to get started with this problem. In this, we have used Deep Learning because of the advantages it offers to work with images especially in image classification to get improvised results. The methodology includes taking leaves of infected crops and label them as per the disease pattern. The images of infected leaves are processed pixel-based operations are applied to improve the information from the image. As a next step feature extraction is done followed by image segmentation and at the last classification of crop diseases based on the patterns extracted from the diseased leaves. In the future, the proposed detection methodology can also be adopted for other agricultural applications.

POTATO PLANT DISEASE DETECTION USING DEEP LEARNING

PREFACE

TABLE OF CONTENTS

1. INTRODUCTION

1.1 TASK AREA

Deep learning is a subset of machine learning and artificial intelligence that has gained significant attention and success in recent years. It involves training artificial neural networks with multiple layers, also known as deep neural networks, to learn and make intelligent decisions from vast amounts of data. Potato plant disease detection using deep learning involves applying artificial intelligence and deep learning techniques to automatically detect and classify diseases in potato plants based on images of their leaves or other plant parts. This technology aims to assist farmers and researchers in early disease identification, enabling timely interventions to prevent widespread crop losses and improve agricultural practices.

PROBLEM STATEMENT:

Problem statement is to classify whether a plant is healthy or not by using an image of apotato leaf. It is a multi-class classification problem. Following are the three classes:

- Late Blight- Potato leaves are more deteriorated.
- Early Blight- Potato leaves are in the early stage of disease.
- Healthy- Leaves are healthy.

DATASET USED:

Any supervised machine learning project starts with data collection process, there are basically three steps to collect dataset:

1) Collect and annotate data on your own.
2) Write web scrapping scripts to collect images from internet.
3) Buy data from third party vendors or use public repositories such as Kaggle.

Dataset is taken from Kaggle repository and it consists of images belonging to three different classes. Below is the link for dataset:

https://www.kaggle.com/datasets/arjuntejaswi/plant-village

Dataset contains three types of images :

Potato_healthy

Potato_Late_blight

Potato_Early_blight

1.2 EVALUATION

Potato plant disease detection using deep learning involves evaluating the trained model's performance and accuracy in identifying and classifying diseases in potato plants based on input images. The evaluation process is essential to ensure the reliability and effectiveness of the system in real-world agricultural applications.

Evaluation for potato plant disease detection using deep learning involves assessing the performance and accuracy of the trained deep learning model in correctly identifying and classifying diseases in potato plants based on input images. To evaluate the model, a separate dataset containing labeled images of healthy and diseased potato plants is used. This dataset is distinct from the data used for training and is referred toas the test set. The model is applied to the test set, and its predictions are compared to the ground truth labels to calculate various performance metrics.

Common evaluation metrics for disease detection tasks include accuracy, precision. Accuracy measures the overall correctness of the model's predictions, while precision represents the proportion of correctly identified diseased samples among all predicted diseased samples. Furthermore, confusion matrices can be generated to gain insights into the model's specific strengths and weaknesses in classifying different disease types. These matrices display the counts of true positive, true negative, false positive, and false negative predictions for each disease class.

2. LITERATURE SURVEY

2.1 MAJOR AREA

Potato plant disease detection using deep learning lies at the intersection of agriculture and artificial intelligence (AI), with a focus on precision farming and smart agricultural practices. Deep learning techniques have revolutionized the field of computer vision, enabling accurate and efficient analysis of images, which can be harnessed to detect and classify diseases in potato plants.

One of the primary applications of deep learning in this domain is early disease detection. By analyzing images of potato plants captured through drones or cameras, deep learning models can identify subtle signs of diseases that may not be visible to the naked eye.

Automated diagnosis is another key area enabled by deep learning. Trained models can recognize specific disease patterns and symptoms in potato plant images, providing accurate and consistent assessments of disease presence and severity. This reduces the reliance on human expertise, enables rapid diagnosis, and allows for data-driven decision-making in crop management.

Deep learning-based disease detection systems also contribute to crop monitoring and management. Integrated into smart farming technologies, these systems can continuously monitor large fields of potato crops, providing real-time disease information to farmers. Moreover, the use of deep learning in potato plant disease detection fosters sustainable agriculture. By precisely identifying diseased plants, farmers can apply targeted pesticide treatments, reducing the overall use of chemicals and minimizing environmental impact.

Beyond agriculture, deep learning contributes to research and education in plant pathology. As data is collected from various farms and regions, it provides valuable insights into disease prevalence, distribution, and factors influencing outbreaks. This knowledge aids researchers in developing better disease management strategies and enhances the dissemination of information to farmers and agricultural experts.

2.2 SUB AREA

Potato plant disease detection using deep learning encompass specialized research directions and applications that focus on enhancing disease detection accuracy and addressing specific challenges. It includes transfer learning to leverage pre-trained models, anomaly detection for early identification of rare diseases, and multi-spectral imaging to integrate diverse spectral data. These sub-areas address specific challenges and contribute to more accurate and efficient disease detection systems, benefiting farmers, promoting sustainable agriculture, and supporting global food security.

Potato plant disease detection using deep learning uses CNN algorithm. CNN, or Convolutional Neural Network, is a specialized type of deep learning architecture primarily used for image and video recognition tasks. It is inspired by the visual processing mechanism of the human brain and designed to automatically and adaptively learn spatial hierarchies of features from input images.

CNNs are well-suited for image recognition tasks, including potato plant disease detection, as they can automatically learn hierarchical patterns and representations from raw image data. By stacking multiple convolutional, activation, and pooling layers, CNNs create increasingly abstract representations of the input, enabling effective feature extraction and classification

For potato plant disease detection, a Convolutional Neural Network(CNN) can be trained on a dataset of labeled images containing healthy and diseased potato plants. This simple CNN consists of convolutional layer in which a kernel or filter is applied which is used to detect the vertical or horizontal edges and then it is followed by Relu activation function. Next layer is max pooling layer in which a kernel is applied to extract the most prominent features from the image. Then followed by fully connected dense layers. Accuracy of the model is determined by the loss function in forward propagation and according to the loss value weights, filters and bias are updated in back propagation which may be through gradient descent or AdaDelta or adam.

Key components of a CNN include :

1. **Convolutional Layers**: These layers apply convolutional filters (kernels) to the input image to detect local patterns and features. Each filter slides across the image, computing dot product sand generating feature maps

2. **Activation Function**: Non-linear activation functions (e.g., ReLU - Rectified Linear Unit) introduce non-linearity into the network, enabling the model to learn complex relationships between features.

3. **Pooling Layers**: Pooling layers down sample feature maps, reducing the spatial dimensions and extracting the most salient features. Common pooling methods are max-pooling and average-pooling.

4. **Fully Connected Layers**: These layers take flattened feature maps and perform classification based on learned features. They connect all neurons from one layer to another, typically leading to the final output layer.

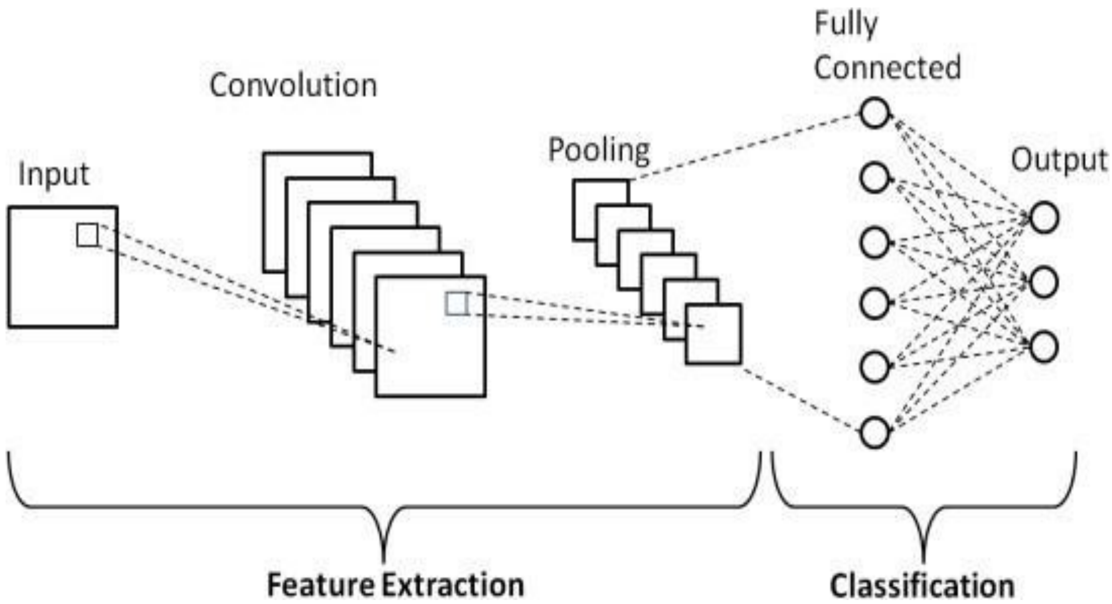


Fig 2.1 Block Diagram of CNN

2.3 EVOLUTION OF TASK

The evolution of tasks in potato plant disease detection has undergone remarkable advancements, fueled by the integration of cutting-edge technologies and methodologies over the years. This journey can be summarized in the following stages:

1. **Manual Visual Inspection**: In the early days of agriculture, disease detection in potato plants relied solely on manual visual inspection. While this approach was intuitive, it was labor-intensive, time-consuming, and subject to human error, limiting its scalability and accuracy.

2. **Traditional Image Processing Techniques**: With the advent of digital imaging in agriculture, traditional image processing techniques started to be applied for disease detection. These methods involved extracting handcrafted features from images of potato plants and applying rule-based algorithms to detect diseases. However, these techniques often struggled to handle variations in disease patterns and environmental conditions, leading to limited accuracy.

3. **Machine Learning Approaches**: As machine learning gained popularity in the 1990s, researchers began to explore the application of supervised learning algorithms to classify diseases in potato plants based on labeled image datasets. Machine learning algorithms, such as Support Vector Machines (SVM) and decision trees, showed promise in automating the detection process and achieving better accuracy than traditional techniques.

4. **Deep Learning and Convolutional Neural Networks (CNNs)**: The breakthrough of deep learning, particularly CNNs, revolutionized the field of image recognition, including potato plant disease detection. CNNs demonstrated the ability to automatically learn hierarchical and complex features directly from raw image data. They excelled at feature extraction and showed superior performance in disease classification tasks, significantly improving detection accuracy and robustness.

3. SYSTEM & IMPLEMENTATION

3.1 EXISTING SYSTEM

Potato plant disease detection using deep learning had made significant advancements in recent years. Before the widespread adoption of deep learning, potato plant disease detection predominantly relied on traditional computer vision and machine learning techniques. Some of the existing systems for potato plant disease detection before deep learning include Rule-Based Systems, Image Processing Techniques, Color-Based Segmentation, Texture Analysis, Machine Learning with Handcrafted Features, Expert Systems.

The commonly used technique is Support Vector Machines (SVM), which can be applied to classify potato plant diseases using image data. In this approach, a dataset of potato plant images with labeled disease types is collected. The images are preprocessed, resized, and converted to suitable formats. Next, relevant features are extracted from the images, representing them numerically. The dataset is then divided into training and testing sets. The training set is used to train the SVM classifier by finding the optimal hyperplane that separates the different disease classes in the feature space. SVM aims to maximize the margin between the classes, promoting better generalization to new data.

Another commonly used technique is K-Nearest Neighbors (KNN), which is a simple and effective machine learning algorithm for potato plant disease detection using image data. In this approach, a dataset of potato plant images with labeled disease types is collected. The images are preprocessed, and relevant features, such as color histograms, texture descriptors, and shape features, are extracted to represent them numerically

However, these techniques may face challenges with large-scale datasets and high-dimensional feature spaces, as it requires storing all training samples in memory for classification. For more complex and diverse image datasets, deep learning techniques like CNN have demonstrated superior performance, surpassing the traditional approaches.

3.2 PROPOSED SYSTEM

Potato plant disease detection using deep learning is a cutting- edge solution that harnesses the power of Convolutional Neural Networks (CNNs) to automatically detect and classify diseases in potato plants based on input images. The system is designed to provide accurate and efficient disease diagnosis, supporting farmers and researchers in making informed decisions to ensure healthy crop yields. The system's foundation lies in a diverse dataset of potato plant images, consisting of both healthy and diseased samples, capturing various disease manifestations and real- world scenarios. Preprocessing techniques are applied to standardize the image data, ensuring consistent input for the CNN model.

The heart of the system is the CNN architecture, engineered to learn intricate features and patterns indicative of different diseases from the image data. During the training phase, the model optimizes its internal parameters through backpropagation and gradient descent, enhancing its ability to distinguish between healthy and diseased plants.

Validation and testing of the trained model are crucial steps to assess its performance and generalization capabilities on independent datasets. This evaluation ensures the model's accuracy and robustness in real-world scenarios.

To make the system accessible to end-users, a user-friendly web interface or mobile application can be developed. Through this interface, users can effortlessly upload images of their potato plants, which are then processed by the deep learning model in real- time. The system provides rapid and accurate disease classification results, empowering farmers to take timely and targeted actions to mitigate the impact of diseases on their crops.

The proposed system's potential benefits extend beyond individual farms, contributing to sustainable agriculture practices and supporting global food security efforts. Continuous updates and improvements to the model based on new data and user feedback ensure its ongoing effectiveness in detecting and addressing emerging disease patterns.

3.3 FLOW DIAGRAM

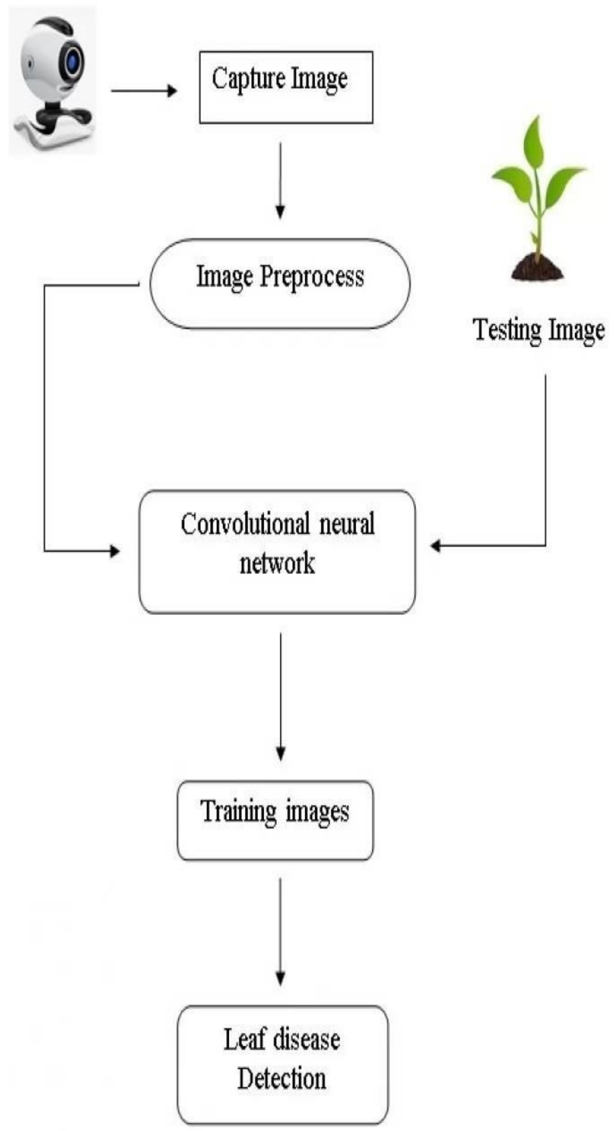


Fig 3.1 Flow diagram of plant disease detection using deep learning

3.4 PROCEDURE (ALGORITHM)

The process for potato plant disease detection using deep learning :

Step 1 : Data Collection and Preprocessing

- Collect a diverse dataset of potato plant images, including both healthy and diseased samples, and arrange them in labeled folders.
- Resize the images to a consistent size (e.g., 256x256 pixels) and normalize the pixel values to a range of [0, 1].

Code :

```
from tensorflow.keras.preprocessing.image import ImageDataGenerator
data_dir = "/Users/mynampativivekreddy/Desktop/PlantVillage"
image_size = (256, 256)

datagen = ImageDataGenerator(
rescale=1./255,
validation_split=0.1
)

train_dataset = datagen.flow_from_directory(
data_dir,
target_size=image_size,
batch_size=BATCH_SIZE,
class_mode='sparse',
subset='training'
)

val_dataset = datagen.flow_from_directory(
```

```
data_dir,
target_size=image_size,
batch_size=BATCH_SIZE,
class_mode='sparse',
subset='validation'
)
```

Step 2 : Data Augmentation

- Data Augmentation is a process that generates several realistic variants of each training sample, to artificially expand the size of the training dataset.

Code :

```
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)
```

Step 3 : Model Development

- Choose a suitable deep learning model architecture for image classification, such as a Convolutional Neural Network (CNN).
- Add convolutional layers with activation functions and pooling layers to themodel.

Code :

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense

model = Sequential([
 Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=input_shape),
 MaxPooling2D((2, 2)),
 Conv2D(64, kernel_size=(3, 3), activation='relu'),
 MaxPooling2D((2, 2)),
 Conv2D(64, kernel_size=(3, 3), activation='relu'),
 MaxPooling2D((2, 2)),
 Conv2D(64, kernel_size=(3, 3), activation='relu'),
 MaxPooling2D((2, 2)),
 Conv2D(64, kernel_size=(3, 3), activation='relu'),
 MaxPooling2D((2, 2)),
 Flatten(),
 Dense(64, activation='relu'), Dense(n_classes,
 activation='softmax')
 ])
```

Step 4: Model Compilation and Training

- Compile the model with an appropriate optimizer, loss function, and evaluationmetric.
- Train the model using the training dataset and validate it using the validationdataset.

Code **:**

```
model.compile(
optimizer='adam',
loss='sparse_categorical_crossentropy',
metrics=['accuracy']
)

history = model.fit( train_dataset,
validation_data=val_dataset,
epochs=EPOCHS,
verbose=1
)
```

Step 5: Model Evaluation and Visualization

- Evaluate the trained model using the test dataset.
- Visualize the training and validation accuracy and loss over epochs.

Code :

```
test_dataset = datagen.flow_from_directory(
data_dir,
target_size=image_size,
batch_size=BATCH_SIZE,
class_mode='sparse',
shuffle=False,
subset='validation';
)

scores = model.evaluate(test_dataset)
acc = history.history['accuracy']
```

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

loss =  history.history['loss']

val_loss = history.history['val_loss']


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()
```

Step 6: Prediction

- Use the trained model to make predictions on new unseen images.
- Visualize the predicted class labels and their corresponding confidence scores.

Code :

```
for images_batch, labels_batch in test_dataset:
for i in range(len(images_batch)):
first_image = images_batch[i].numpy().astype('uint8')
first_label = labels_batch[i].numpy()


plt.imshow(first_image)
```

```
plt.title("Actual Label: " + class_names[first_label])

batch_prediction=model.predict(np.expand_dims(images_batch[i],axis=0))
predicted_label = class_names[np.argmax(batch_prediction[0])] confidence =
round(100 * np.max(batch_prediction[0]), 2)

plt.title(f"Predicted Label: {predicted_label}, Confidence: {confidence}%")
plt.axis("off")
plt.show()
```

By following these steps and executing the provided code, we can implement potato plant disease detection using deep learning and obtain predictions for new images.

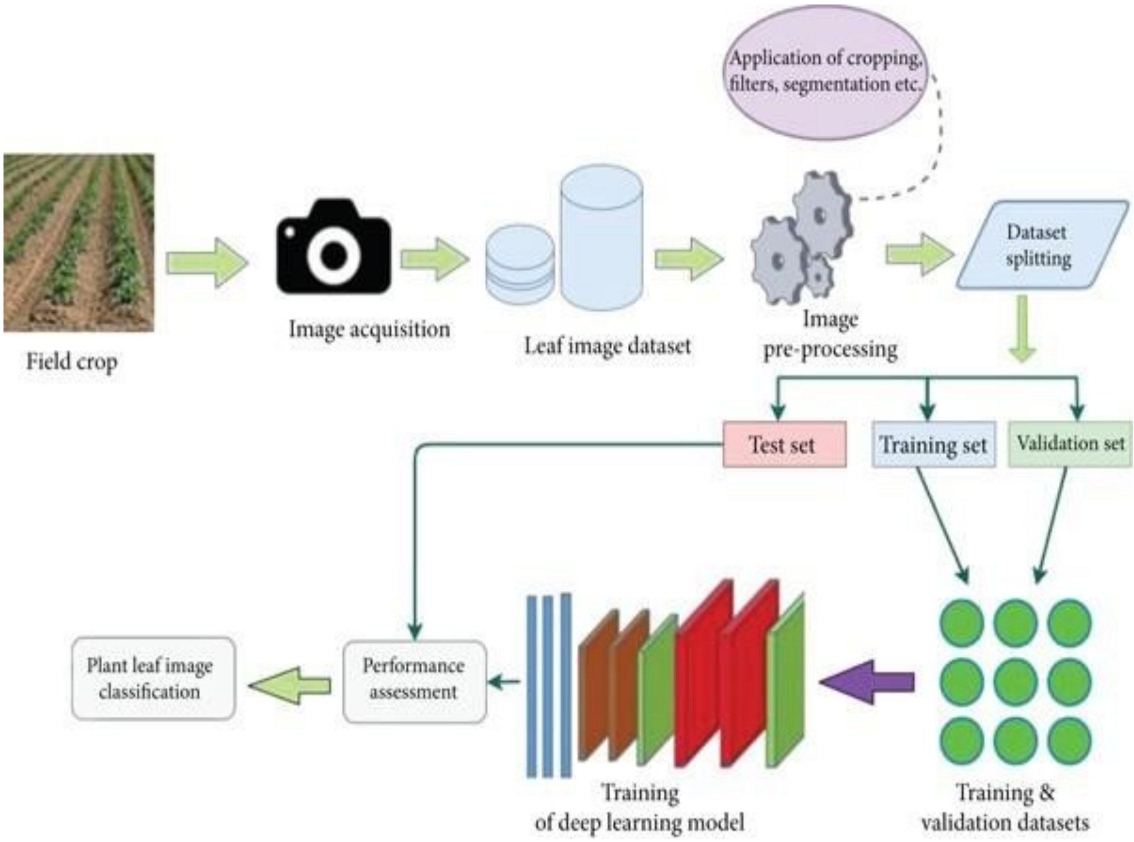


Fig 3.2 Approach to potato plant disease detection

3.5 SYSTEM SETUP

Setting up a system for potato plant disease detection using deep learning involves a series of steps to ensure the efficient and accurate detection of diseases in potato plants. The first step in the system setup is to determine the hardware requirements. Deep learning models, especially Convolutional Neural Networks (CNNs), require significant computational power for training. Therefore, it is essential to have a GPU (Graphics Processing Unit) to accelerate the training process. Additionally, a powerful CPU (Central Processing Unit) is required to handle data preprocessing and other tasks effectively. Sufficient RAM is also necessary to handle large datasets and model parameters efficiently.

Next, the system setup requires various software components. An appropriate operating system, such as Linux, is chosen for compatibility with deep learning libraries and tools. Python, a widely used programming language in the deep learning community, is installed, along with essential libraries like NumPy, Pandas, and Matplotlib for data manipulation and visualization. A deep learning framework, such as TensorFlow or PyTorch, is selected for building and training the deep learning model. Image processing libraries like PIL (Python Imaging Library) or OpenCV are also installed for image preprocessing and augmentation. Additionally, Jupyter Notebook or an integrated development environment (IDE) is set up to write and execute the code efficiently.

Data collection and preprocessing are crucial steps in the system setup. A diverse dataset of potato plant images, comprising both healthy and diseased samples, is collected and labeled accordingly. The images are preprocessed by resizing, normalizing, and augmenting the dataset to increase its size and improve the model's generalization capabilities.

Model development and training are essential aspects of the system setup. A suitable pre-trained CNN model or a custom architecture is chosen for disease detection. The dataset is split into training, validation, and test sets. The deep learning model is implemented using the selected framework and then trained on the GPU using the training set.

Following model development, evaluation, and validation are conducted. The trained model is evaluated using the validation set and appropriate metrics like accuracy, precision, recall, and F1-score. Error analysis is performed, and the model is fine-tuned based on the evaluation results to improve its performance.

Once the model is trained and validated, deployment and integration come into play. The trained model is saved for future use and deployment. A user-friendly interface is created, or the model is integrated into existing agricultural systems for real-time disease detection.

Finally, monitoring and maintenance are essential to ensure the system's ongoing performance. Regular monitoring of the system's performance is conducted to ensure accurate and reliable disease detection. Periodic updates to the model using new data help the system adapt to changing disease patterns and improve its accuracy over time. This setup enables effective potato plant disease detection using deep learning, providing valuable insights to farmers to make informed decisions and protect their crops effectively.

4. REQUIREMETS

4.1 SOFTWARE REQUIREMENTS

1. **Python**: Python is the programming language commonly used for deep learning tasks. Ensure you have Python installed on your system. You can download the latest version of Python from the official website.

2. **TensorFlow**: TensorFlow is an open-source deep learning framework developed by Google. It provides tools and libraries for building and training neural networks. Install TensorFlow using pip.

3. **Keras**: Keras is a high-level neural networks API that runs on top of TensorFlow. It simplifies the process of building and training deep learning models. Keras is usually included with TensorFlow, so you don't need to install it separately.

4. **NumPy**: NumPy is a Python library used for numerical computing and handling arrays. It is widely used in data manipulation and preprocessing for deep learning. Install NumPy using pip.

5. **Matplotlib**: Matplotlib is a plotting library used for visualizing data and generating graphs. It is helpful for visualizing the training and validation metrics during the model training process. Install Matplotlib using pip.

6. **Pillow**: Pillow is a library used for image loading and manipulation. It is a successor to the Python Imaging Library (PIL). Install Pillow using pip.

The Potato Plant Disease Detection Software, utilizing deep learning techniques, is a state-of-the-art solution designed to revolutionize potato farming by enabling accurate and efficient disease detection. Leveraging the power of deep learning algorithms, the software aims to assist farmers and agricultural experts in identifying various diseases in potato plants with unparalleled precision.

4.2 HARDWARE REQUIREMENTS

The hardware requirements for potato plant disease detection using deep learning can vary based on the complexity of the model, the size of the dataset, and the specific deep learning tasks involved. Here are the general hardware requirements for running deep learning models for potato plant disease detection:

1. **CPU:** A multi-core CPU is essential for training deep learning models efficiently. While deep learning can be run on a CPU, training times can be significantly reduced by using a powerful CPU with multiple cores.

2. G**PU** (Graphics Processing Unit): For deep learning tasks, especially when working with large datasets and complex models, a GPU is highly recommended. GPUs are capable of parallel processing, which speeds up the training process and enables the use of larger batch sizes.

3. **RAM**: Sufficient RAM is crucial for loading and processing large image datasets. The amount of RAM needed depends on the size of the dataset and the complexity of the deep learning model. At least 16GB of RAM is recommended, but more is better for handling larger datasets.

4. **Storage**: Deep learning models often require large amounts of storage space, especially when dealing with large datasets. Make sure you have ample storage space to store the dataset, model checkpoints, and other related files.

5. **Internet Connectivity**: Internet connectivity is necessary if you plan to use cloud- based services for training or accessing pre-trained models and datasets.

6. **Optional**: If you are dealing with extremely large datasets or complex models and require faster training times, you may consider using multiple GPUs or specialized hardware such as TPUs (Tensor Processing Units) for accelerated training.

4.3 FUNCTIONAL REQUIREMENTS

Functional requirements for potato plant disease detection using deep learning encompass the specific capabilities and features that the system must possess to effectively and accurately identify diseases in potato plants based on input images. These requirements are crucial in defining the scope and performance of the system. Functional requirements for potato plant disease detection using deep learning are essential to define the system's core functionalities and capabilities.

Firstly, the system must be capable of image classification, accurately distinguishing between healthy and diseased potato plant images based on their visual features and patterns. It should support model training with labeled datasets to learn disease detection effectively. The system should enable users to input new potato plant images for real-time prediction, providing a confidence level or probability score for each classification.

The accuracy of disease detection is crucial, and the system should strive for high precision. It must be scalable to handle large datasets and support multiple disease types, including common ones like late blight, early blight, and black scurf. Robustness is vital, allowing the system to handle variations in image quality and different stages of disease progression. Compatibility with various platforms and devices, such as smartphones and computers, is necessary for user accessibility.

The system's user interface should be intuitive, enabling easy interaction and image input. Proper documentation is essential, providing installation instructions, usage guidelines, and troubleshooting tips for users and developers. Lastly, the system must ensure security and privacy for user data, especially concerning sensitive agricultural information. By meeting these functional requirements, the potato plant disease detection system can offer accurate, efficient, and user-friendly disease diagnosis, supporting agricultural practices and improving crop management.

5. BEST & WORST INFERENCES

The deep learning-based potato plant disease detection system demonstrates several notable strengths that contribute to its effectiveness. One of the major advantages is its high accuracy in identifying various diseases affecting potato plants. By utilizing Convolutional Neural Networks (CNNs), the model can automatically learn and extract intricate patterns and features from input images, leading to precise disease classification. This accuracy is crucial for early disease detection, enabling farmers to implement timely and targeted interventions to prevent the spread of diseases and minimize crop losses. The system's ability to handle a diverse dataset with images of healthy and diseased plants further enhances its reliability and generalization capabilities.

Additionally, the data augmentation technique employed during the training process boosts the model's robustness. This ensures that the model can accurately classify potato plant diseases even when faced with variations in illumination, rotation, or other environmental factors. The user-friendly interface of the system makes it accessible to farmers and agricultural experts with minimal technical expertise. They can easily upload images of their potato plants and obtain reliable disease predictions, enabling informed decision-making for disease management. Ultimately, the better inferences of the system contribute to improved crop productivity, sustainable agriculture practices, and enhanced food security.

While the deep learning-based potato plant disease detection system has demonstrated significant promise, it is essential to acknowledge certain limitations and challenges. One of the main concerns is the requirement for a substantial amount of labeled training data. Collecting and annotating a diverse dataset of potato plant images can be time-consuming and labor-intensive, potentially limiting the accessibility of the system to regions with limited resources. Moreover, the accuracy of disease detection heavily relies on the quality and representativeness of the training data, which may vary across different regions and climates.

Another potential limitation lies in the interpretability of the deep learning model. CNNs are known for being complex black-box models, making it challenging to understand the specific features and patterns contributing to disease classification. This lack of interpretability might hinder the system's adoption by some farmers and experts who may seek more transparent and explainable models.

Furthermore, the system's performance could be affected by the presence of rare or novel diseases that were not adequately represented in the training dataset. In such cases, the model may struggle to accurately classify these unseen diseases, leading to potential misdiagnosis.

To address these limitations, ongoing research efforts should focus on developing techniques for data augmentation, transfer learning, and model interpretability. Additionally, collaboration between agricultural experts and deep learning researchers is crucial for improving the system's robustness and extending its applicability to diverse regions and disease scenarios. By addressing these challenges, the potato plant disease detection system can be further enhanced and solidify its position as an invaluable tool for supporting sustainable agriculture and crop health management.

6. OUTPUT SCREENS

```
import matplotlib.pyplot as plt
#Matplotlib is a python library used to create 2D graphs and plots by using python scripts.

import numpy as np
#NumPy is a Python library used for working with arrays.

import  Import "tensorflow" could not be resolved Pylance(reportMissingImports)
#PIL st  View Problem (⌥F8)   No quick fixes available   ed Python to deal with images

import tensorflow as tf
#The TensorFlow helps you implement best practices for data automation, model tracking, performance monitoring, and model retraining.

from tensorflow import keras
#Keras is a high-level, deep learning API developed for implementing neural networks

from tensorflow.keras import layers

from tensorflow.keras.models import Sequential
[1]  ✓ 4.4s                                                                Python
```

```
BATCH_SIZE = 50
IMAGE_SIZE = 256
CHANNELS=3
EPOCHS=20
[2]  ✓ 0.0s
```

```
data_dir = "/Users/mynampativivekreddy/Desktop/PlantVillage"
[3]  ✓ 0.0s
```

```
dataset = tf.keras.preprocessing.image_dataset_from_directory(
    data_dir,
    seed=123,
    shuffle=True,
    image_size=(IMAGE_SIZE,IMAGE_SIZE),
    batch_size=BATCH_SIZE
)
[4]  ✓ 0.1s
...  Found 2153 files belonging to 3 classes.
```

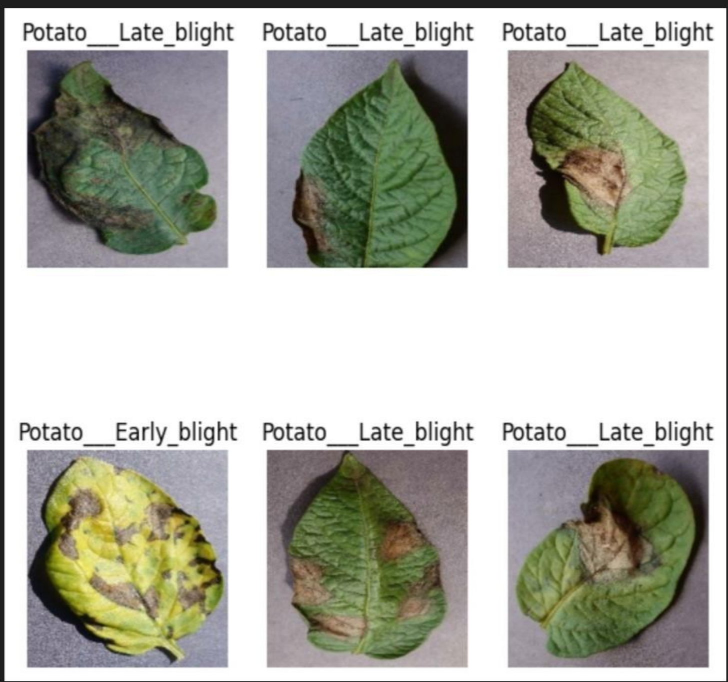
```
class_names = dataset.class_names
class_names
[5]  ✓ 0.0s
```

```
['Potato___Early_blight', 'Potato___Late_blight', 'Potato___healthy']
```

```
(variable) image_batch: Any
for image_                          t.take(1):
    print(image_batch.shape)
    print(labels_batch.numpy())
[6]  ✓ 0.1s
```

```
(50, 256, 256, 3)
[1 1 1 1 0 1 1 0 1 0 0 0 2 1 0 0 0 0 0 0 1 1 1 2 1 0 0 1 1 1 2 1 0 0 0 0 0
 0 1 1 0 1 1 0 1 0 0 1 0 2]
```

```
plt.figure(figsize=(7,7))
for image_batch, labels_batch in dataset.take(1):
    for i in range(6):
        ax = plt.subplot(2,3, i + 1)
        plt.imshow(image_batch[i].numpy().astype("uint8"))
        plt.title(class_names[labels_batch[i]])
        plt.axis("off")
[46]  ✓ 0.5s
```



```
len(dataset)
[8] ✓ 0.0s
··· 44

train_size = 0.8
len(dataset)*train_size
[9] ✓ 0.0s
··· 35.2

train_ds = dataset.take(12)
len(train_ds)
[10] ✓ 0.0s
··· 12

test_ds = dataset.skip(12)
len(test_ds)
[11] ✓ 0.0s
··· 32

val_size=0.1
len(dataset)*val_size
[12] ✓ 0.0s
··· 4.4
```

```
val_ds = test_ds.take(1)
len(val_ds)
[13] ✓ 0.0s
··· 1

+ Code   + Markdown

test_ds = test_ds.skip(1)
len(test_ds)
[14] ✓ 0.0s
··· 31

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
[15] ✓ 0.0s

train_ds, val_ds, test_ds = get_dataset_partitions_tf(dataset)
[16] ✓ 0.0s
```

```python
len(val_ds)
```
[17] ✓ 0.0s

```
4
```

```python
len(test_ds)
```
[18] ✓ 0.0s

```
5
```

```python
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)
```
[19] ✓ 0.0s

+ Code + Markdown

```python
resize_and_rescale = tf.keras.Sequential([
  layers.experimental.preprocessing.Resizing(IMAGE_SIZE, IMAGE_SIZE),
  layers.experimental.preprocessing.Rescaling(1./255),
])
```
[20] ✓ 0.0s

```python
data_augmentation = tf.keras.Sequential([
  layers.experimental.preprocessing.RandomFlip("horizontal_and_vertical"),
  layers.experimental.preprocessing.RandomRotation(0.2),
])
```
[21] ✓ 0.0s

```python
train_ds = train_ds.map(
    lambda x, y: (data_augmentation(x, training=True), y)
).prefetch(buffer_size=tf.data.AUTOTUNE)
```
[22] ✓ 0.3s

```python
input_shape = (BATCH_SIZE, IMAGE_SIZE, IMAGE_SIZE, CHANNELS)
n_classes = 3

model = 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)
```
[23] ✓ 0.0s

```
model.summary()
[24] ✓ 0.0s

Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 sequential (Sequential)     (50, 256, 256, 3)         0

 conv2d (Conv2D)             (50, 254, 254, 32)        896

 max_pooling2d (MaxPooling2  (50, 127, 127, 32)        0
 D)

 conv2d_1 (Conv2D)           (50, 125, 125, 64)        18496

 max_pooling2d_1 (MaxPoolin  (50, 62, 62, 64)          0
 g2D)

 conv2d_2 (Conv2D)           (50, 60, 60, 64)          36928

 max_pooling2d_2 (MaxPoolin  (50, 30, 30, 64)          0
 g2D)

 conv2d_3 (Conv2D)           (50, 28, 28, 64)          36928

 max_pooling2d_3 (MaxPoolin  (50, 14, 14, 64)          0
 g2D)
...
Total params: 183747 (717.76 KB)
Trainable params: 183747 (717.76 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

```
model.compile(
    optimizer='adam',
    loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=False),
    metrics=['accuracy']
)
[25] ✓ 0.0s                                                    Python
```

```
model_filename = 'model.h5'
model.save(model_filename)
[26] ✓ 0.0s                                                    Python

/Users/mynampativivekreddy/Library/Python/3.9/lib/python/site-packages/keras/src/engine/training.py:3000: UserWarning: You are saving your model as an HDF5 file
  saving_api.save_model(
```

```
history = model.fit(
    train_ds,
    batch_size=BATCH_SIZE,
    validation_data=val_ds,
    verbose=1,
    epochs=20,
)
```

[27] ✓ 24m 30.5s

```
Epoch 1/20
35/35 [==============================] - 81s 2s/step - loss: 0.8982 - accuracy: 0.8321 - val_loss: 0.7580 - val_accuracy: 0.5200
Epoch 2/20
35/35 [==============================] - 69s 2s/step - loss: 0.6895 - accuracy: 0.5091 - val_loss: 0.5527 - val_accuracy: 0.3300
Epoch 3/20
35/35 [==============================] - 69s 2s/step - loss: 0.5517 - accuracy: 0.4809 - val_loss: 0.6081 - val_accuracy: 0.6350
Epoch 4/20
35/35 [==============================] - 69s 2s/step - loss: 0.3935 - accuracy: 0.4662 - val_loss: 0.3192 - val_accuracy: 0.5150
Epoch 5/20
35/35 [==============================] - 69s 2s/step - loss: 0.3302 - accuracy: 0.4416 - val_loss: 0.4098 - val_accuracy: 0.5800
Epoch 6/20
35/35 [==============================] - 70s 2s/step - loss: 0.2487 - accuracy: 0.4656 - val_loss: 0.1136 - val_accuracy: 0.4550
Epoch 7/20
35/35 [==============================] - 64s 2s/step - loss: 0.2037 - accuracy: 0.4527 - val_loss: 0.1659 - val_accuracy: 0.5150
Epoch 8/20
35/35 [==============================] - 65s 2s/step - loss: 0.1475 - accuracy: 0.4586 - val_loss: 0.1550 - val_accuracy: 0.5150
Epoch 9/20
35/35 [==============================] - 80s 2s/step - loss: 0.1077 - accuracy: 0.4639 - val_loss: 0.1525 - val_accuracy: 0.5100
Epoch 10/20
35/35 [==============================] - 72s 2s/step - loss: 0.1373 - accuracy: 0.4674 - val_loss: 0.1380 - val_accuracy: 0.4400
Epoch 11/20
35/35 [==============================] - 72s 2s/step - loss: 0.1779 - accuracy: 0.4580 - val_loss: 0.3363 - val_accuracy: 0.5650
Epoch 12/20
35/35 [==============================] - 85s 2s/step - loss: 0.1273 - accuracy: 0.4674 - val_loss: 0.0558 - val_accuracy: 0.4650
Epoch 13/20
...
Epoch 19/20
35/35 [==============================] - 67s 2s/step - loss: 0.0458 - accuracy: 0.4668 - val_loss: 0.0793 - val_accuracy: 0.4900
Epoch 20/20
35/35 [==============================] - 71s 2s/step - loss: 0.0544 - accuracy: 0.4674 - val_loss: 0.1709 - val_accuracy: 0.5200
```

Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...

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

[28] ✓ 4.9s

```
5/5 [==============================] - 5s 657ms/step - loss: 0.2250 - accuracy: 0.5240
```

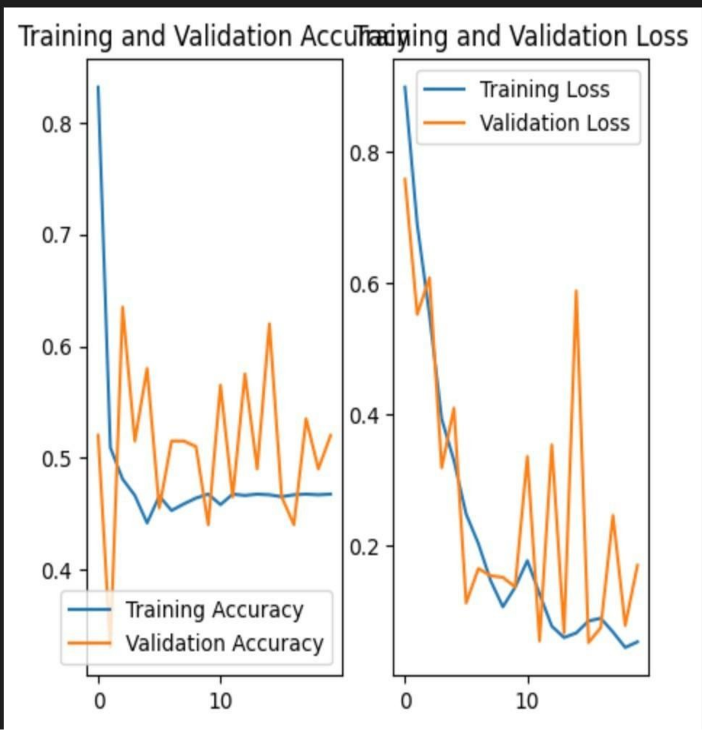
+ Code + Markdown

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

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

[29] ✓ 0.0s

```python
plt.figure(figsize=(5, 5))
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()
```

[47] ✓ 0.2s



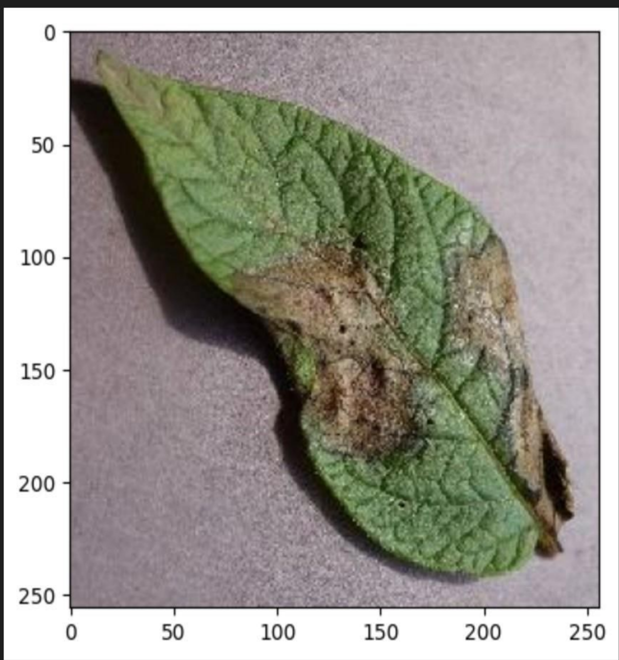

```python
def predict(model, img):
    img_array = tf.keras.preprocessing.image.img_to_array(images[i].numpy())
    img_array = tf.expand_dims(img_array, 0)

    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
```

[32] ✓ 0.0s

```
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 (variable) model: Any bel])

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

[31] ✓ 1.0s

```
first image to predict
actual label: Potato___Late_blight
2/2 [==============================] - 1s 183ms/step
predicted label: Potato___Early_blight
```



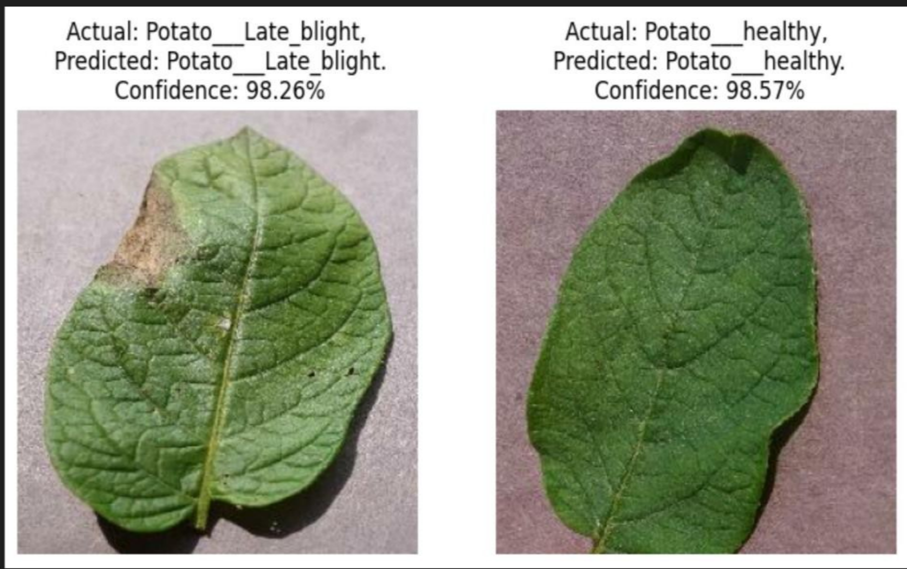

```
def predict(model, img):
    img_array = tf.keras.preprocessing.image.img_to_array(images[i].numpy())
    img_array = tf.expand_dims(img_array, 0)

    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
```

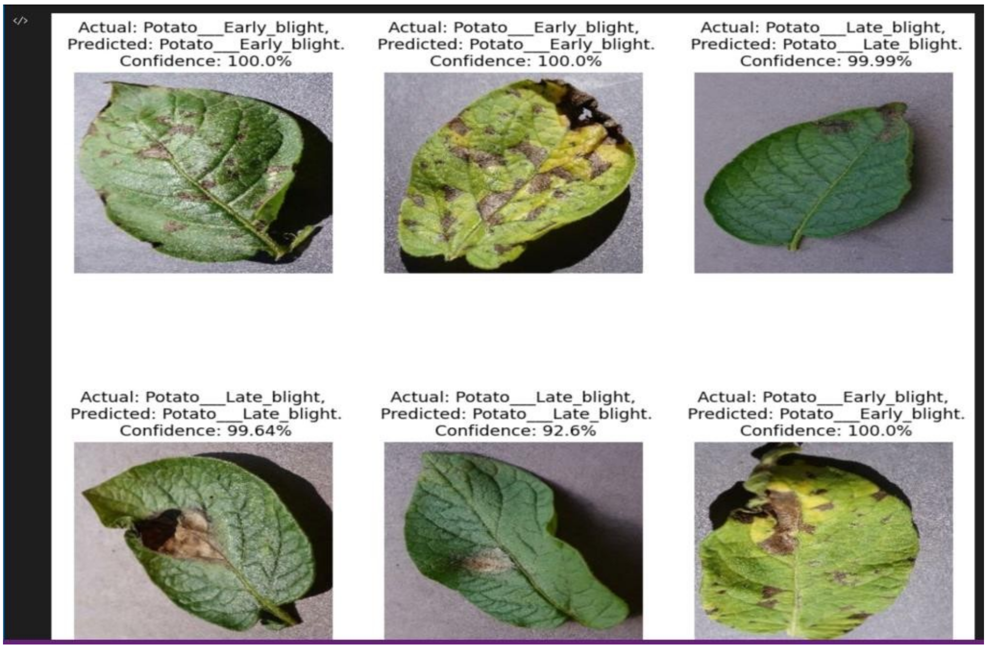
[32] ✓ 0.0s

```python
plt.figure(figsize=(9,9))
for images, labels in test_ds.take(1):
    for i in range(2):
        ax = plt.subplot(1, 2, 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")
```

[60] ✓ 0.4s

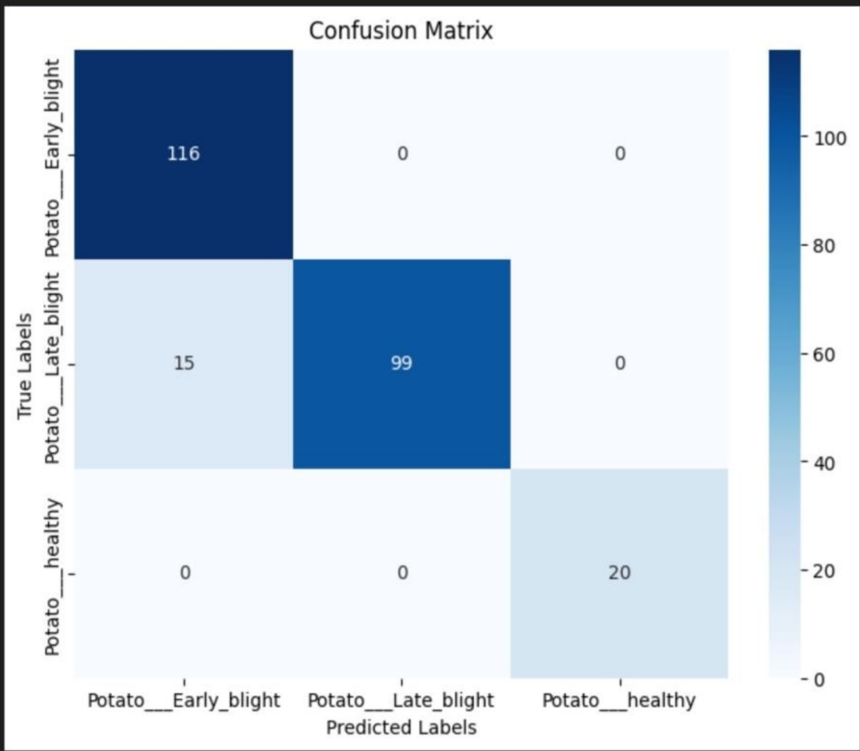
```
1/1 [==============================] - 0s 45ms/step
1/1 [==============================] - 0s 25ms/step
```





```
import seaborn as sns

plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt="d", cmap="Blues", xticklabels=class_names, yticklabels=class_names)
plt.xlabel("Predicted Labels")
plt.ylabel("True Labels")
plt.title("Confusion Matrix")
plt.show()
```

[75] ✓ 0.1s



7. OUTLINE

Potato plant disease detection using deep learning is an advanced technique to identify and classify diseases affecting potato crops. The process involves collecting a diverse dataset of potato plant images, including healthy and diseased samples. The dataset undergoes preprocessing steps to ensure consistency and variability, such as resizing and data augmentation. A Convolutional Neural Network (CNN) architecture is designed to effectively learn features from the images, consisting of convolutional and pooling layers.

The dataset is split into training, validation, and testing sets, where the training set is used to train the model, the validation set helps tune hyperparameters, and the testing set evaluates the model's performance on unseen data. During training, the CNN model learns to recognize patterns associated with different potato plant diseases through iterative weight updates using optimization algorithms.

Hyperparameter tuning is performed using the validation set to achieve optimal model performance. Once trained and validated, the model is evaluated on the testing dataset to measure accuracy and performance metrics. In real-world applications, the trained model can be used to predict the probability of new potato plant images belonging to each disease category, facilitating timely disease detection and management for farmers.

The ultimate goal is to deploy the model in production environments, integrating it into agricultural systems, web applications, or mobile apps. This provides farmers with a reliable and efficient tool for potato plant disease detection, leading to improved crop yield, reduced economic losses, and sustainable agriculture practices. The integration of deep learning technology into plant disease detection holds great promise for enhancing agricultural productivity and food security.

8. CONCLUSION

Potato plant disease detection using deep learning has shown promising results and offers several advantages for the agricultural industry. The use of deep learning models has the potential to revolutionize disease monitoring and management in potato crops. By leveraging large datasets and powerful neural networks, these models can achieve high accuracy in detecting various diseases, enabling early detection and timely intervention. One of the major advantages of using deep learning for disease detection is its ability to automatically learn and extract relevant features from images without the need for hand- crafted features. Another significant benefit is the scalability of deep learning models. Once trained, these models can process large volumes of data rapidly, making them suitable for real-time disease monitoring across vast agricultural areas. In conclusion, potato plant disease detection using deep learning has the potential to revolutionize the agricultural sector by significantly improving disease management, increasing crop yield, and contributing to global food security. Continued advancements and innovations in deep learning technologies hold great promise for the future of potato farming and beyond. The benefits of using deep learning for potato plant disease detection include improved accuracy, scalability, and potential for automation, which can lead to increased crop yield and reduced losses. Additionally, deep learning models can generalize well to new and unseen potato plant images, making them adaptable to different farming scenarios.

9. FUTURE SCOPE

The future scope for potato plant disease detection using deep learning is highly promising, offering significant advancements in agricultural practices. One key area of development is improving the accuracy and robustness of deep learning models. By utilizing larger and diverse datasets, exploring novel architectures, and optimizing hyperparameters, the models can achieve better performance, even in challenging environmental conditions and against rare diseases.

Integration with edge computing and Internet of Things (IoT) devices is another avenue for future growth. Deploying lightweight models on IoT sensors and drones allows real-time disease monitoring in the fields, reducing data transmission and enhancing responsiveness in disease detection.

Transfer learning and few-shot learning will be explored to adapt pre-trained models to potato disease detection, making it easier for farmers to implement disease detection on a smaller scale with limited labeled data.

Overall, the future holds immense potential for leveraging deep learning in potato plant disease detection, enabling more efficient disease management, crop health monitoring, and fostering sustainable agricultural practices for improved food security.

10. REFERENCES

- E. Hirani, V. Magotra, J. Jain and P. Bide, "Plant Disease Detection Using Deep Learning," 2021 6th International Conference for Convergence in Technology (I2CT), Maharashtra, India, 2021, pp. 1-4, doi: 10.1109/I2CT51068.2021.9417910.

- A. Sharma, K. Lakhwani and H. Singh Janeja, "Plant Disease Identification Using Deep Learning: A Systematic Review," 2021 2nd International Conference on Intelligent Engineering and Management (ICIEM), London, United Kingdom, 2021, pp. 222-227, doi: 10.1109/ICIEM51511.2021.9445277.

- Chohan, Murk, et al. "Plant disease detection using deep learning." *International Journal of Recent Technology and Engineering* 9.1 (2020): 909-914.

- Islam, Md Ashraful, and Md Hanif Sikder. "A Deep Learning Approach to Classify the Potato Leaf Disease." *Islam, MA, & Sikder, MH (2022). A Deep Learning Approach to Classify the Potato Leaf Disease. Journal of Advances in Mathematics and Computer Science* 37.12 (2022): 143-155.

APPENDIX – C

ABSTRACT

Sreenidhi Institute of Science andTechnology

Department of Information Technology

GROUP PROJECT

Batch No : 08		**Title of the Group Project**
ROLL NO	**NAME**	
20311A1234	Meghana Reddy Mynampati	POTATO PLANT DISEASE DETECTION USING DEEP LEARNING
20311A1243	Chelukali Ashish Kumar	
21315A1205	Panga Saikiran	

ABSTRACT

The identification of plant disease is an imperative part of crop monitoring systems. Computer vision and deep learning (DL) techniques have been proven to be state-of-the-art to address various agricultural problems. Deep Learning rich libraries and user as well as developer friendly environment to work with, all these qualities make Deep Learning as the favourable method to get started with this problem. The methodology includes taking leaves of infected crops and label them as per the disease pattern. The images of infected leaves are processed pixel based operations are applied to improve the information from the image. As a next step feature extraction is done followed by image segmentation and at the last classification of crop diseases based on the patterns extracted from the diseased leaves. In the future, the proposed detection methodology can also be adopted for other agricultural applications.

Student 1: Meghana Mynampati **Student 2:** Chelukali Ashish **Student 3:** Panga Saikiran

Project Coordinator	**Internal Guide**	**HOD-IT**
Mr. P. SREEDHAR	**Mr. S. MARUTHI**	**Dr SUNIL BHUTADA**
Associate Professor &	Associate Professor	Professor
Associate Head	Department of IT	Department of IT
Department of IT		

APPENDIX – D

CORRELATION BETWEEN THE GROUP PROJECT-1 AND THE PROGRAM OUTCOMES(POS), PROGRAM SPECIFIC OUTCOMES(PSOS)

BATCH NO : 08		Title of the Group Project
ROLL NO	**NAME**	
20311A1234	Meghana Reddy Mynampati	POTATO PLANT DISEASE DETECTION USING DEEP LEARNING
20311A1243	Chelukali Ashish Kumar	
21315A1205	Panga Saikiran	

Table 1: Group Project correlation with appropriate POs/PSOs (Please specify level of Correlation,H/M/L against POs/PSOs)

H	High	M	Moderate	L	Low

SREENIDHI INSTITUTE OF SCIENCE ANDTECHNOLOGY DEPARTMENT OF INFORMATIONTECHNOLOGY Projects Correlation with POs/PSOs														
PO1	**PO2**	**PO3**	**PO4**	**PO5**	**PO6**	**PO7**	**PO8**	**PO9**	**PO10**	**PO11**	**PO12**	**PSO1**	**PSO2**	**PSO3**
M	**L**	**L**	**H**	**H**	**L**	**M**	**H**	**M**	**H**	**H**	**H**	**H**	**H**	**M**

Student 1: Meghana Mynampati **Student 2:** Chelukali Ashish **Student 3:** Panga Saikiran

Project Coordinator
Mr. P. SREEDHAR
Associate Professor &
Associate Head
Department of IT

Internal Guide
Mr. S. MARUTHI
Associate Professor
Department of IT

HOD-IT
Dr. SUNIL BHUTADA
Professor
Department of IT

APPENDIX – E

DOMAIN OF THE PROJECT AND NATURE OF THE PROJECT

<table>
<tr><th colspan="2">BATCH NO : 08</th><th rowspan="2">Title of the Group Project</th></tr>
<tr><th>ROLL NO</th><th>NAME</th></tr>
<tr><td>20311A1234</td><td>Meghana Reddy Mynampati</td><td rowspan="3">POTATO PLANT DISEASE DETECTION USING DEEP LEARNING</td></tr>
<tr><td>20311A1260</td><td>Chelukali Ashish Kumar</td></tr>
<tr><td>21315A1203</td><td>Panga Saikiran</td></tr>
</table>

Table 2: Nature of the Group Project (Please tick √ Appropriate for your project)

<table>
<tr><th rowspan="2">Batch No.</th><th rowspan="2">Title Of The Group Project</th><th colspan="4">Nature of Group Project</th></tr>
<tr><th>Product</th><th>Application</th><th>Research</th><th>Others (please specify)</th></tr>
<tr><td>08</td><td>POTATO PLANT DISEASE DETECTION USING DEEP LEARNING</td><td></td><td>√</td><td></td><td></td></tr>
</table>

Student 1: Meghana Mynampati **Student 2:** Chelukali Ashish **Student 3:** Panga Saikiran

Project Coordinator
Mr. P. SREEDHAR
Associate Professor &
Associate Head
Department of IT

Internal Guide
Mr. S. MARUTHI
Associate Professor
Department of IT

HOD-IT
Dr. SUNIL BHUTADA
Professor
Department of IT

Table 3: Domain of the Group Project (Please tick √Appropriate for your project)

Batch No.	Title of the Group Project	Domain of the Group Project				
		ARTIFICIAL INTELLIGENCE, MACHINE LEARNING(Python technology) and DEEP LEARNING	COMPUTER NETWORKS, INFORMATION SECURITY, CYBER SECURITY	DATA WAREHOUSING,DATA MINING, BIG DATA ANALYTICS	CLOUD COMPUTING, INTERNET OF THINGS	SOFTWARE ENGINEERING,IMAGE PROCESSING
08	POTATO PLANT DISEASE DETECTION USING DEEP LEARNING	√				

Student 1: Meghana Mynampati **Student 2:** Chelukali Ashish **Student 3:** Panga Saikiran

Project Coordinator
Mr. P. SREEDHAR
Associate Professor &
Associate Head
Department of IT

Internal Guide
Mr. S. MARUTHI
Associate Professor
Department of IT

HOD-IT
Dr. SUNIL BHUTADA
Professor
Department of IT