ISSN No : 2454-4221 (Print)
ISSN No : 2454-423X (Online)
International Journal of Research in Advanced Computer Science Engineering
A Peer Reviewed Open Access International Journal
www.ijracse.com

IMAGE BASED LEAF DISEASE DETECTION AND PREDICTION

S. Pratap [1], G. Sowmya [2], V. Sai Amrutha [3], A.V. Adithya [4], A. S. V. Rajesh [5].
1 Assistant Professor, Dept. of CSE, NS Raju Institute of Technology Visakhapatnam, Andhra Pradesh, India
2,3,4,5 Student, Dept of CSE, NS Raju Institute of Technology Visakhapatnam, Andhra Pradesh, India
spratap.cse@nsrit.edu.in, sowmyasrao24@gmail.com, vsaiamrutha1610@gmail.com, suryaadiyhya12@gmail.com, asvrajesh431@gmail.com.

Abstract
Plant Leaf Disease are a vital danger to food security, anyway their fast distinctive verification stays inconvenient in various pieces of the world due to the non participation of the significant establishment. Rise of exact methods in the field of leaf-based picture characterization has shown amazing outcomes. This paper utilizes Random Timberland in distinguishing among solid and unhealthy leaf from the informational collections made. Our proposed paper incorporates different periods of execution specifically dataset creation, include extraction, preparing the classifier and characterization. The made datasets of sick and solid leaves are all things considered prepared under Arbitrary Forest to arrange the ailing and sound pictures. For removing highlights of a picture we use Histogram of an Oriented Inclination (HOG). Generally speaking, utilizing AI to prepare the enormous informational collections accessible openly gives us an unmistakable method to distinguish the illness present in plants in an enormous scope

Keywords: Diseased and Healthy leaf, Random forest, Feature extraction, Training, Classification

1. Introduction

The agriculturist in provincial regions may think that it's tough to distinguish the illness which may be available in their harvests. it is not mild for them to visit agribusiness office and discover what the infection may be. Our precept objective is to differentiate the illness introduce in a plant by using watching its morphology with the aid of picture dealing with and gadget mastering. Pests and illnesses results inside the destruction of crops or component of the plant ensuing in decreased food manufacturing leading to food lack of confidence. Also, information about the pest control or manipulate and diseases are less in numerous less evolved countries. Poisonous pathogens, negative disorder control, drastic climate modifications are one of the key elements which arises in diminished food production. Diverse present day technologies have emerged to minimize postharvest processing, to improve agricultural sustainability and to maximize the productivity.

Cite this article as: S.Pratap, G.Sowmya, V.Sai Amrutha, A.V.Adithya, A.S.V.Rajesh., "Image Based Leaf Disease Detection And Prediction", International Journal of Research in Advanced Computer Science Engineering, (IJRACSE), Volume 6 Issue 11, April 2021, Page 14-30.

ISSN No : 2454-4221 (Print)
ISSN No : 2454-423X (Online)
International Journal of Research in Advanced Computer Science Engineering
A Peer Reviewed Open Access International Journal
www.ijracse.com

Diverse Laboratory primarily based procedures which include polymerase chain reaction, fuel chromatography, mass spectrometry, thermography and hyper spectral techniques have been hired for disorder identity. However, these techniques are not price effective and are excessive time eating. in recent times, server based totally and cellular based totally technique for disorder identity has been hired for ailment identity. numerous factors of these technologies being high resolution digital camera, high performance processing and considerable built in accessories are the brought blessings resulting in automated ailment recognition. current procedures together with machine gaining knowledge of and deep learning algorithm has been employed to boom the reputation price and the accuracy of the outcomes. various researches have taken region beneath the sector of gadget studying for plant disorder detection and analysis, such conventional machine mastering technique being random forest, synthetic neural network, support vector system(SVM), fuzzy good judgment, ok-way approach, Convolutional neural networks and so forth.…Random forests are as a whole, gaining knowledge of method for type, regression and different duties that operate by means of constructing a woodland of the selection timber throughout the education time. unlike selection trees, Random forets overcome the disadvantage of over fitting of their schooling facts set and it handles both numeric and categorical data. The histogram of orientated gradients (HOG) is an detail descriptor utilized as a part of computer imaginative and prescient and image processing for the sake of item detection. here we are making utilization of three component descriptors.

1.1 Introduction to Data Mining

Data mining integrates approaches and techniques from various disciplines such as machine learning, statistics, artificial intelligence, neural networks, database management, data warehousing, data visualization, spatial data analysis,probability graph theory etc. In short, data mining is a multi-disciplinary field.

1.1.1 Statistics

Statistics includes a number of methods to analyze numerical data in large quantities. Different statistical tools used in data mining are regression analysis, cluster analysis, correlation analysis and Bayesian network. Statistical models are usually built from a training data set. Correlation analysis identifies the correlation of variables to each other. Bayesian network is a directed graph that represents casual relationship among data found out using the Bayesian probability theorem. Given below is a simple Bayesian network where the nodes represent variables whereas edges represent the relationship between the nodes.

1.1.2 Machine Learning

Machine learning is the collection of methods, principles and algorithms that enables learning and prediction on the basis of past data. Machine learning is used to build new models and to search for a best model matching the test data. Machine learning methods normally use heuristics while searching for the model. Data mining uses a number of machine

ISSN No : 2454-4221 (Print)
ISSN No : 2454-423X (Online)
International Journal of Research in Advanced Computer Science Engineering
A Peer Reviewed Open Access International Journal
www.ijracse.com

learning methods including inductive concept learning, conceptual clustering and decision tree induction. A decision tree is a classification tree that decides the class of an object by following the path from the root to a leaf node. Given below is a
simple decision tree that is used for weather forecasting.

1.1.3 Database Oriented Techniques

Advancements in database and data warehouse implementation helps data mining in a number of ways. Database oriented techniques are used mainly to develop characteristics of the available data. Iterative database scanning for frequent item sets, attribute focusing, and attribute oriented induction are some of the database oriented techniques widely used in data mining. The iterative database scanning searches for frequent item sets in a database. Attribute oriented induction generalizes low level data into high level concepts using conceptual hierarchies.

1.1.4 Neural Networks

A neural network is a set of connected nodes called neurons. A neuron is a computing device that computes some requirement of its inputs and the inputs can even be the outputs of other neurons. A neural network can be trained to find the relationship between input attributes and output attribute by adjusting the connections and the parameters of the nodes.

1.1.5 Data Visualization

The information extracted from large volumes of data should be presented well to the end user and data visualization techniques make this possible. Data is transformed into different visual objects such as dots, lines, shapes etc and displayed in a two or three dimensional space. Data visualization is an effective way to identify trends, patterns, correlations and outliers from large amounts of data.

1.1.6 Summary

Data mining combines different techniques from various disciplines such as machine learning, statistics, database management, data visualization etc. These methods can be combined to deal with complex problems or to get alternative solutions. Normally data mining system employs one or more techniques to handle different kinds of data, different data mining tasks, different application areas and different data requirements.

1.2 Patterns in Data Mining

1. Association: The items or objects in relational databases, transactional databases or any other information repositories are considered.
2. Classification: The goal of classification is to construct a model with the help of historical data that can accurately predict the value. It maps the data into the predefined groups or classes and searches for the new patterns. For example: To predict weather on a particular day will be categorized into - sunny, rainy, or cloudy.
3. Regression: Regression creates predictive models. Regression analysis is used to make predictions based on existing data by applying formulas. Regression is very useful for finding (or predicting) the information on the basis of previously known information.

ISSN No : 2454-4221 (Print)
ISSN No : 2454-423X (Online)
International Journal of Research in Advanced Computer Science Engineering
A Peer Reviewed Open Access International Journal
www.ijracse.com

4. Cluster analysis: It is a process of portioning a set of data into a set of meaningful subclass, called as cluster. It is used to place the data elements into the related groups without advanced knowledge of the group definitions.
5. Forecasting: Forecasting is concerned with the discovery of knowledge or information patterns in data that can lead to reasonable predictions about the future.

1.3 Deep Learning Tutorial

Deep learning is based on the branch of machine learning, which is a subset of artificial intelligence. Since neural networks imitate the human brain and so deep learning will do. In deep learning, nothing is programmed explicitly. Basically, it is a machine learning class that makes use of numerous nonlinear processing units so as to perform feature extraction as well as transformation. The output from each preceding layer is taken as input by each one of the successive layers. Deep learning models are capable enough to focus on the accurate features themselves by requiring a little guidance from the programmer and are very helpful in solving out the problem of dimensionality. Deep learning algorithms are used, especially when we have a huge no of inputs and outputs. Since deep learning has been evolved by the machine learning, which itself is a subset of artificial intelligence and as the idea behind the artificial intelligence is to mimic the human behavior, so same is "the idea of deep learning to build such algorithm that can mimic the brain". Deep learning is implemented with the help of Neural Networks, and the idea behind the motivation of Neural Network is the biological neurons, which is nothing but a brain cell. Deep learning is implemented by the help of deep networks, which are nothing but neural networks with multiple hidden layers. we provide the raw data of images to the first layer of the input layer. After then, these input layer will determine the patterns of local contrast that means it will differentiate on the basis of colors, luminosity, etc. Then the 1st hidden layer will determine the face feature, i.e., it will fixate on eyes, nose, and lips, etc. And then, it will fixate those face features on the correct face template. So, in the 2nd hidden layer, it will actually determine the correct face here as it can be seen in the above image, after which it will be sent to the output layer. Likewise, more hidden layers can be added to solve more complex problems, for example, if you want to find out a particular kind of face having large or light complexions. So, as and when the hidden layers increase, we are able to solve complex problems

1.4.1 Deep Neural Networks

It is a neural network that incorporates the complexity of a certain level, which means several numbers of hidden layers are encompassed in between the input and output layers. They are highly proficient on model and process non-linear associations.

1.4.2 Deep Belief Networks

A deep belief network is a class of Deep Neural Network that comprises of multi-layer belief networks.

1.4.3 Steps to perform DBN

ISSN No : 2454-4221 (Print)
ISSN No : 2454-423X (Online)
International Journal of Research in Advanced Computer Science Engineering
A Peer Reviewed Open Access International Journal
www.ijracse.com

With the help of the Contrastive Divergence algorithm, a layer of features is learned from perceptible units. Next, the formerly trained features are treated as visible units, which perform learning of features. Lastly, when the learning of the final hidden layer is accomplished, then the whole DBN is trained.

1.4.4 Recurrent Neural Networks

It permits parallel as well as sequential computation, and it is exactly similar to that of the human brain (large feedback network of connected neurons). Since they are capable enough to reminisce all of the imperative things related to the input they have received, so they are more precise.

2. System Analysis

2.1 Introduction

Plant diseases cause periodic outbreak of diseases which leads to large scale death and famine. It is estimated that the outbreak of helminthosporiose of rice in north eastern India in 1943 caused a heavy loss of food grains and death of a million people.Since the effects of plantdiseases were devastating, some of the crop cultivation has been abandoned. It is estimated that 2007 plant disease losses in Georgia (USA) is approximately $653.06 million (Jean, 2009). In India no estimation has been made but it is more than USA because the preventive steps taken to protect our crops are not even one-tenth of that in USA.

The Study of the System

To conduct studies and analyses of an operational and technological nature, and to promote the exchange and development of methods and tools for operational analysis as applied to defence problems.

2.2 Feasibility Study

Feasibility study is conducted once the problem is clearly understood. The feasibility study which is a high level capsule version of the entire system analysis and design process. The objective is to determine whether the proposed system is feasible or not and it helps us to the minimum expense of how to solve the problem and to determine, if the Problem is worth solving. The following are the three important tests that have been carried out for feasibility study. This study tells about how this package is useful to the users and its advantages and disadvantages, and also it tells whether this package is cost effective are not. There are three types of feasibility study, they are

a) Economic Feasibility.
b) Technical Feasibility.
c) Operational Feasibility.

2.2.1 Economic Feasibility

Economic feasibility attempts 2 weigh the costs of developing and implementing a new system, against the benefits that would accrue from having the new system in place. This feasibility study gives the top management the economic justification for the new system. A simple economic analysis which gives the actual comparison of costs and benefits are much more meaningful in this case. In addition, this proves to be a useful point of reference to compare actual costs as the project progresses. There could be various types of intangible benefits on account of automation. These could include increased

ISSN No : 2454-4221 (Print)
ISSN No : 2454-423X (Online)
International Journal of Research in Advanced Computer Science Engineering
A Peer Reviewed Open Access International Journal
www.ijracse.com

customer satisfaction, improvement in product quality better decision making timeliness of information, expediting activities, improved accuracy of operations, better documentation and record keeping, faster retrieval of information, better employee morale.

2.2.2 Technical Feasibility

Evaluating the technical feasibility is the trickiest part of a feasibility study. This is because, at this point in time, not too many detailed design of the system, making it difficult to access issues like performance, costs on (on account of the kind of technology to be deployed) etc. A number of issues have to be considered while doing a technical analysis. Understand the different technologies involved in the proposed system before commencing the project we have to be very clear about what are the technologies that are to be required for the development of the new system. Find out
whether the organization currently possesses the required technologies. Is the required technology available with the organization?

2.2.3 Operational Feasibility

Proposed project is beneficial only if it can be turned into information systems that will meet the organizations operating requirements. Simply stated, this test of feasibility asks if the system will work when it is developed and installed. Are there major barriers to Implementation? Here are questions that will help test the operational feasibility of a project: Is there sufficient support for the project from management from users? If the current system is well liked and used to the extent that persons will not be able to see reasons for change, there may be resistance. Are the current business methods acceptable to the user? If they are not, Users may welcome a change that will bring about a more operational and useful systems. Are there major barriers to Implementation? Here are questions that will help test the operational feasibility of a project. Have the user been involved in the planning and development of the project? Since the proposed system was to help reduce the hardships encountered. In the existing manual system, the new system was considered to be operational feasible.

2.3 Software Environment

Python is an easy to learn, powerful programming language. It has efficient high-level data structures and a simple but effective approach to object-oriented programming. Python's elegant syntax and dynamic typing, together with its interpreted nature, make it an ideal language for scripting and rapid application development in many areas on most platforms. The Python interpreter and the extensive standard library are freely available in source or binary form for all major platforms from the Python Web site, https://www.python.org/, and may be freely distributed. The same site also contains distributions of and pointers to many free third party Python modules, programs and tools, and additional documentation. The Python interpreter is easily extended with new functions and data types implemented in C or C++ (or other languages callable from C). Python is also suitable as an extension language for customizable applications.

ISSN No : 2454-4221 (Print)
ISSN No : 2454-423X (Online)
International Journal of Research in Advanced Computer Science Engineering
A Peer Reviewed Open Access International Journal
www.ijracse.com

This tutorial introduces the reader informally to the basic concepts and features of the Python language and system. It helps to have a Python interpreter handy for hands-on experience, but all examples are self-contained, so the tutorial can be read off-line as well. For a description of standard objects and modules, see The Python Standard Library. The Python Language Reference gives a more formal definition of the language. To write extensions in C or C++, read Extending and Embedding the Python Interpreter and Python/C API Reference Manual. There are also several books covering Python in depth. This tutorial does not attempt to be comprehensive and cover every single feature, or even every commonly used feature. Instead, it introduces many of Python's most noteworthy features, and will give you a good idea of the language's flavor and style. After reading it, you will be able to read and write Python modules and programs, and you will be ready to learn more about the various Python library modules described in The Python Standard Library.

2.3.1 The Python Standard Library

While The Python Language Reference describes the exact syntax and semantics of the Python language, this library reference manual describes the standard library that is distributed with Python. It also describes some of the optional components that are commonly included in Python distributions.

Python's standard library is very extensive, offering a wide range of facilities as indicated by the long table of contents listed below. The library contains built-in modules (written in C) that provide access to system functionality such as file I/O that would otherwise be inaccessible to Python programmers, as well as modules written in Python that provide standardized solutions for many problems that occur in everyday programming. Some of these modules are explicitly designed to encourage and enhance the portability of Python programs by abstracting away platform-specifics into platform-neutral APIs.

The Python installers for the Windows platform usually include the entire standard library and often also include many additional components. For Unix-like operating systems Python is normally provided as a collection of packages, so it may be necessary to use the packaging tools provided with the operating system to obtain some or all of the optional components

2.3.2 Dealing with Bugs

Python is a mature programming language which has established a reputation for stability. In order to maintain this reputation, the developers would like to know of any deficiencies you find in Python. It can be sometimes faster to fix bugs yourself and contribute patches to Python as it streamlines the process and involves less people. Learn how to contribute.

2.4 Requirements Specification

2.4.1 Functional Requirements

Functional requirements define the internal workings of the software: that is, the technical details, data manipulation and processing and other specific functionality that show how the use cases are to be satisfied. They are

ISSN No : 2454-4221 (Print)
ISSN No : 2454-423X (Online)
International Journal of Research in Advanced Computer Science Engineering
A Peer Reviewed Open Access International Journal
www.ijracse.com

supported by non-functional requirements, which impose constraints on the design or implementation.

2.4.1.2 HARDWARE REQUIREMENTS

Processor: Above 1.5GHZ
Hard Disk: 20GB
RAM: 4GB

2.4.1.3 SOFTWARE REQUIREMENTS

Language: PYTHON
OS: Windows10

2.4.1.4 Python Packages

- NumPy
- Pandas
- Matplotlib
- Keras
- TensorFlow

2.4.2 Non Functional Requirements

Non-functional requirements are requirements which specify criteria that can be used to judge the operation of a system, rather than specific behaviors. This should be contrasted with functional requirements that specify specific behavior or functions. Typical non-functional requirements are reliability, scalability, and cost. Non-functional requirements are often called the ilities of a system. Other terms for non-functional requirements are "constraints", "quality attributes" and "quality of service requirements". Reliability: If any exceptions occur during the execution of the software it should be caught and thereby prevent the system from crashing. Scalability: The system should be developed in such a way that new modules and functionalities can be added, thereby facilitating system evolution.

Cost: The cost should be low because a free availability of software package.

3. Design

3.1 Introduction

Logical design The logical design of a system pertains to an abstract representation of the data flows, inputs and outputs of the system. This is often conducted via modeling, using an over-abstract (and sometimes graphical) model of the actual system. In the context of systems design are included. Logical design includes ER Diagrams i.e. Entity Relationship Diagrams

Physical design The physical design relates to the actual input and output processes of the system. This is laid down in terms of how data is input into a system, how it is verified / authenticated, how it is processed, and how it is displayed as output.

In Physical design, following requirements about the system are decided.

1. Input requirement,
2. Output requirements,
3. Storage requirements,
4. Processing Requirements,
5. System control and backup or recovery.

Put another way, the physical portion of systems design can generally be broken down into three sub-tasks:
1. User Interface Design
2. Data Design

ISSN No : 2454-4221 (Print)
ISSN No : 2454-423X (Online)
International Journal of Research in Advanced Computer Science Engineering
A Peer Reviewed Open Access International Journal
www.ijracse.com

3. Process Design

User Interface Design is concerned with how users add information to the system and with how the system presents information back to them. Data Design is concerned with how the data is represented and stored within the system. Finally, Process Design is concerned with how data moves through the system, and with how and where it is validated, secured and/or transformed as it flows into, through and out of the system. At the end of the systems design phase, documentation describing the three sub-tasks is produced and made available for use in the next phase. Physical design, in this context, does not refer to the tangible physical design of an information system. To use an analogy, a personal computer's physical design involves input via a keyboard, processing within the CPU, and output via a monitor, printer, etc. It would not concern the actual layout of the tangible hardware, which for a PC would be a monitor, CPU, motherboard, hard drive, modems, video/graphics cards, USB slots, etc. It involves a detailed design of a user and a product database structure processor and a control processor. The H/S personal specification is developed for the proposed system.

3.2 Input & Output Representation

3.2.1 Input Design

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:

- What data should be given as input?
- How the data should be arranged or coded?
- The dialog to guide the operating personnel in providing input.
- Methods for preparing input validations and steps to follow when error occur.

Objectives

Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system.

It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities. When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user will not be in maize of instant. Thus the objective

ISSN No : 2454-4221 (Print)
ISSN No : 2454-423X (Online)
International Journal of Research in Advanced Computer Science Engineering
A Peer Reviewed Open Access International Journal
www.ijracse.com

of input design is to create an input layout that is easy to follow.

3.2.2 Output Design

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output.important and direct source information to the user. Efficient and intelligent output design improves the system's relationship to help user decision-making. a. Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily and effectively. When analysis design computer output, they should Identify the specific output that is needed to meet the requirements.

Select methods for presenting information and Create document, report, or other formats that contain information produced by the system.

The output form of an information system should accomplish one or more of the following objectives.

- Convey information about past activities, current status or projections of the future.
- Signal important events, opportunities, problems, or warnings.
- Trigger an action.
- Confirm an action.

3.3 System Architecture

Architecture Flow:

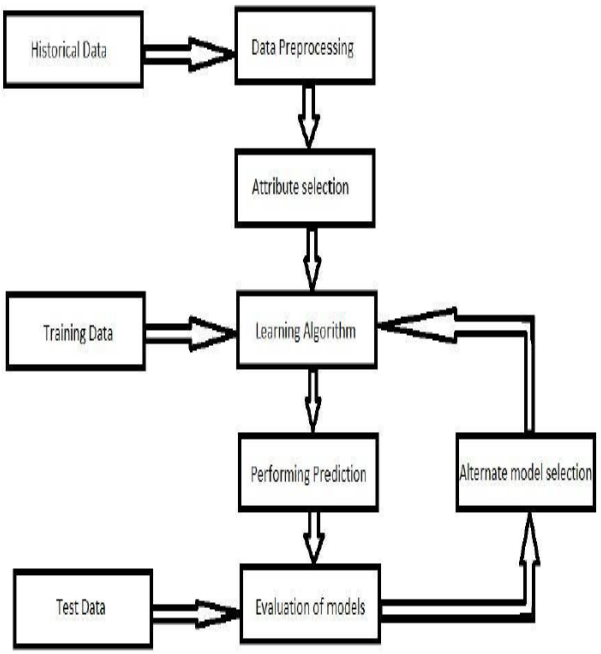
Below architecture diagram represents mainly flow of request from the users to database through servers. In this scenario overall system is designed in three tiers separately using three layers called presentation layer, business layer, data link layer. This project was developed using 3-tier architecture.

3.3.1 3-Tier Architecture:

The three-tier software architecture (three layer architecture) emerged in the 1990s to overcome the limitations of the two-tier architecture. The third tier (middle tier server) is between the user interface (client) and the data management (server) components. This middle tier provides process management where business logic and rules are executed and can accommodate hundreds of users (as compared to only 100 users with the two tier architecture) by providing functions such as queuing, application execution, and database staging. The three tier architecture is used when an effective distributed client/server design is needed that provides (when compared to the two tier) increased performance, flexibility, maintainability, reusability, and scalability, while hiding the complexity of distributed processing from the user. These characteristics have made three layer architectures a popular choice for Internet applications and net-centric information systems..

ISSN No : 2454-4221 (Print)
ISSN No : 2454-423X (Online)
International Journal of Research in Advanced Computer Science Engineering
A Peer Reviewed Open Access International Journal
www.ijracse.com

PROPOSED ALGORITHM



4. Implementation

Source Code

```
import tensorflow as tf
from tensorflow.keras.preprocessing import image
import os
import matplotlib.pyplot as plt
import numpy as np
!pip install kaggle
path='/root/.kaggle/'
file='kaggle.json'
if not (os.path.exists(path)):
os.mkdir(path)
if not (os.path.exists('/root/.kaggle/kaggle.json')):
with open(os.path.join(path, file), 'w+') as fp:
print(fp)
!echo '{"username":"sowmya516","key":"d95736bad1816576bbbf0ebf99194231"}' > /root/.kaggle/kaggle.json
! chmod 600 ~/.kaggle/kaggle.json
! kaggle datasets download -d vipoooool/new-plant-diseases-dataset
!unzip /content/new-plant-diseases-dataset.zip
data_dir='new plant diseases dataset(augmented)/New Plant Diseases Dataset(Augmented)/train'
test_dir='new plant diseases dataset(augmented)/New Plant Diseases Dataset(Augmented)/valid'
batch_size=128
target_size=(224,224)
epoch=100
from keras.preprocessing.image import ImageDataGenerator
data_train_generator=ImageDataGenerator(rescale=1./255,zoom_range=0.2,
width_shift_range=0.2,
height_shift_range=0.2,
shear_range=0.2,
fill_mode='nearest')
data_valid_generator = ImageDataGenerator(rescale=1./255)
data_train=data_train_generator.flow_from_directory(directory=data_dir,
shuffle=True,
target_size=target_size,
batch_size=batch_size)
data_test=data_valid_generator.flow_from_directory(directory=test_dir,
shuffle=True,
target_size=target_size,
batch_size=batch_size)
len_data=data_train.n+data_test.n
percentage_train_data= int((data_train.n/len_data)*100)
print(str(percentage_train_data) + ' %')
```

ISSN No : 2454-4221 (Print)
ISSN No : 2454-423X (Online)
International Journal of Research in Advanced Computer Science Engineering
A Peer Reviewed Open Access International Journal
www.ijracse.com

```
IMG_SHAPE = target_size + (3,)
base_model =
tf.keras.applications.MobileNetV2(input_shap
e=IMG_SHAPE,
include_top=False,
weights='imagenet')
class_names=data_train.class_indices
image_batch, label_batch =
next(iter(data_train))
feature_batch = base_model(image_batch)
print(feature_batch.shape)
model=tf.keras.Sequential([
tf.keras.applications.MobileNetV2(input_shap
e=IMG_SHAPE,
include_top=False,
weights='imagenet'),
46
tf.keras.layers.MaxPooling2D(2,2,padding='sa
me'),
tf.keras.layers.Conv2D(32, (3,3),
activation='relu'),
tf.keras.layers.BatchNormalization(),
tf.keras.layers.MaxPooling2D(2,2,padding='sa
me'),
tf.keras.layers.Flatten(),
tf.keras.layers.Dense(128,activation='relu'),
tf.keras.layers.Dropout(0.2),
tf.keras.layers.BatchNormalization(),
tf.keras.layers.Dense(64,activation='relu'),
tf.keras.layers.Dropout(0.2),
tf.keras.layers.BatchNormalization(),
tf.keras.layers.Dense(38,activation='softmax')
])
model.layers[0].trainable=False
model.compile(
optimizer=tf.keras.optimizers.Adam(),
loss='categorical_crossentropy',
metrics=['acc'])
path_checkpoint = "model_checkpoint.h5"
class myCallback(tf.keras.callbacks.Callback):
def on_epoch_end(self, epoch, logs={}):
if(logs.get('acc')>0.93 and
logs.get('val_acc')>0.93):
print("\nAkurasi telah mencapai >92%!")
self.model.stop_training = True
modelckpt_callback =
tf.keras.callbacks.ModelCheckpoint(
monitor="loss",
filepath=path_checkpoint,
verbose=1,
save_weights_only=True,
save_best_only=True,
)
history = model.fit(
data_train,
epochs=epoch,
validation_data=data_test,
callbacks=[myCallback(),
modelckpt_callback],
)
plt.plot(history.history['acc'])
plt.plot(history.history['val_acc'])
plt.title('ac - val acc ')
plt.xlabel('Epoch')
plt.legend(['acc','val acc'], loc='best')
plt.show()
import matplotlib.pyplot as plt
plt.plot(history.history['acc'])
plt.plot(history.history['val_acc'])
plt.title('loss - val acc ')
plt.xlabel('Epoch')
plt.legend(['loss','val acc'], loc='best')
plt.show()
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('loss - val loss ')
plt.xlabel('Epoch')
plt.legend(['loss','val loss'], loc='best')
```

ISSN No : 2454-4221 (Print)
ISSN No : 2454-423X (Online)
International Journal of Research in Advanced Computer Science Engineering
A Peer Reviewed Open Access International Journal
www.ijracse.com

```
plt.show()
def getList(dict):
list = []
for key in dict.keys():
list.append(key)
return list
pesticides=["Bonide® Orchard","fireblight",
"Immunox", "None", "None",
"Potassium bicarbonate", "None",
"Fungicides", "Numerous fungicides",
"fungicide Dithane M-45", "None",
"Mancozeb",
"carbaryl", "captan", "None",
"No Cure", "chlorothalonil", "None",
"Clorox® bleach",
```

5.Testing

5.1 Introduction

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

5.2 TYPES OF TESTS

5.2.1 Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

5.2.2 Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components. Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects. The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level interact without error.

5.2.3 Functional testing

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and

ISSN No : 2454-4221 (Print)
ISSN No : 2454-423X (Online)
International Journal of Research in Advanced Computer Science Engineering
A Peer Reviewed Open Access International Journal
www.ijracse.com

technical requirements, system documentation, and user manuals. Functional testing is centered on the following items: Valid Input : identified classes of valid input must be accepted. Invalid Input : identified classes of invalid input must be rejected. Functions : identified functions must be exercised. Output : identified classes of application outputs must be exercised. Systems/Procedures : interfacing systems or procedures must be invoked. Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

5.2.4 System Testing

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

5.2.5 White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

5.2.6 Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

5.2.7 Unit Testing

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

5.3 Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

5.4 Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

5.5 Features to be tested

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

ISSN No : 2454-4221 (Print)
ISSN No : 2454-423X (Online)
International Journal of Research in Advanced Computer Science Engineering
A Peer Reviewed Open Access International Journal
www.ijracse.com

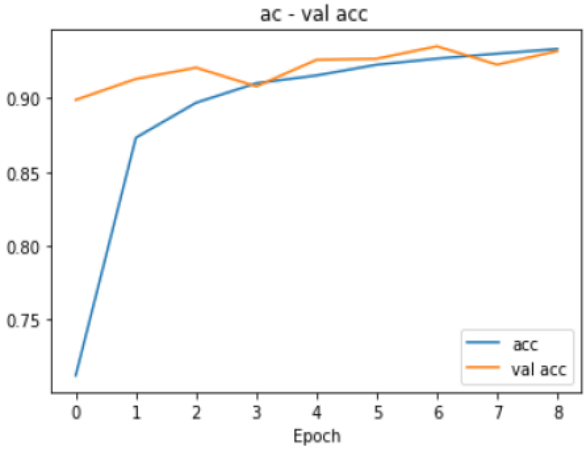
5.6 Test Results:

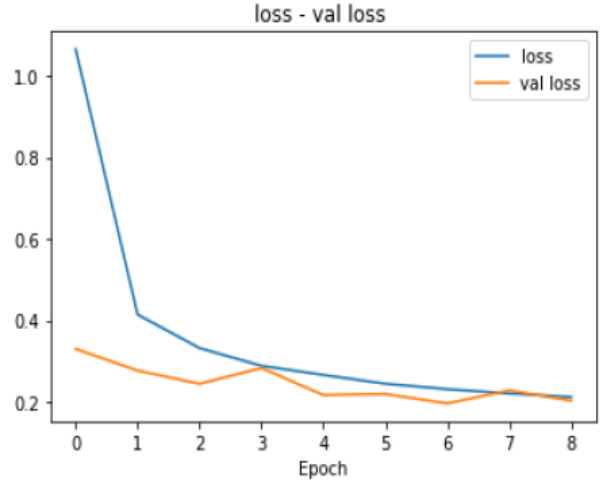
All the test cases mentioned above passed successfully. No defects encountered.

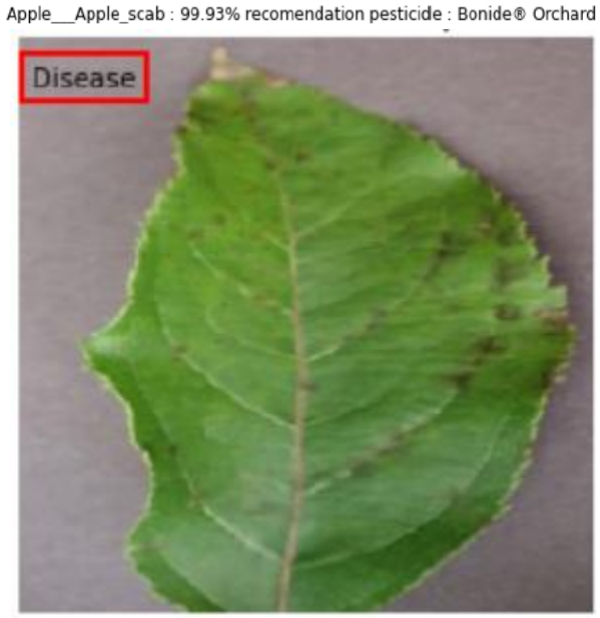
5.7 Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

6. Results







6. Conclusion

In conclusion, MobileNet V2 has successfully implemented to classify various tomato plant diseases based on captured leaf images. The best classification performance is obtained when MobileNet V2 is trained using Adagrad with a batch size of 16. The experimental

ISSN No : 2454-4221 (Print)
ISSN No : 2454-423X (Online)
International Journal of Research in Advanced Computer Science Engineering
A Peer Reviewed Open Access International Journal
www.ijracse.com

results also prove that a learning rate of 0.001 and data division of 4:1 ratio between training and testing deliver the most accurate classification performance. For future work, all classes in the PlantVillage will be explored instead of just three diseases. The objective of this project is the detection, prediction and suggestions, classification of leaf diseases by capturing the images.This system will largely contribute in growth in the yield of the farms.

References

[1] S. Sato et al., "The tomato genome sequence provides insights into fleshy fruit evolution," Nature, vol. 485, no. 7400, pp. 635–641, 2012.

[2] A. Fuentes, S. Yoon, S. C. Kim, and D. S. Park, "A robust deep-learning-based detector for real-time tomato plant diseases and pests recognition," Sensors (Switzerland), vol. 17, no. 9, 2017.

[3] H. Durmus, E. O. Gunes, and M. Kirci, "Disease detection on the leaves of the tomato plants by using deep learning," 6th Int. Conf. Agro-Geoinformatics, Agro-Geoinformatics 2017, 2017.

[4] M. Nowicki, M. R. Foolad, M. Nowakowska, and E. U. Kozik, "Potato and Tomato Late Blight Caused by Phytophthora infestans: An Overview of Pathology and Resistance Breeding," Plant Dis., vol. 96, no. 1, pp. 4–17, 2012.

[5] I. M. Hanssen et al., "Seed transmission of Pepino mosaic virus in tomato," Eur. J. Plant Pathol., vol. 126, no. 2, pp. 145–152, 2010.

[6] W. Zaka-ud-din, Muhammad Aziz, Sumair Rashid, Junaid Ismail, "Classification of Disease in Tomato Plants ' Leaf Using Image Segmentation and SVM," Tech. Journal, Univ. Eng. Technol., vol. 23, no. 2, pp. 81–88, 2018.

[7] M. A. Zulkifley, B. moran, "Enhancement of robust foreground detection through masked GreyWorld and color cooccurrence approach," 3rd IEEE International Conference on Computer Science and Information Technology, vol. 4, pp. 131–136, 2010.

[8] A. K. Rangarajan, R. Purushothaman, and A. Ramesh, "Tomato crop disease classification using pre-trained deep learning algorithm," Procedia Comput. Sci., vol. 133, pp. 1040–1047, 2018.

[9] Q. Xiang, G. Zhang, X. Wang, J. Lai, R. Li, and Q. Hu, "Fruit image classification based on Mobilenetv2 with transfer learning technique," 3rd International Conference on Computer Science and Application Engineering, pp. 1-7, 2019.

[10] Z. Alom, T. M. Taha, C. Yakopcic, S. Westberg, S. Nasrin, and V. K. Asari, "The History Began from AlexNet: A Comprehensive Survey on Deep Learning Approaches," arXiv, vol. abs/1803.01164, pp. 1-39, 2018.

[11] M. A. Zulkifley and N. Trigoni, "Multiple-Model Fully Convolutional Neural Networksm for Single Object Tracking on Thermal Infrared Video," IEEE Access, vol. 6, pp. 42790-42799, 2018.

[12] M. A. Zulkifley, "Two streams multiple-model object tracker for thermal infrared video," IEEE Access, vol. 7, pp. 32383-32392, 2019.

[13] S. B. Jadhav, "Convolutional Neural Networks for Leaf Image-Based Plant Disease Classification," IAES International Journal of Artificial Intelligence (IJ-AI), vol. 8, no. 4, pp. 328–341, 2019.

ISSN No : 2454-4221 (Print)
ISSN No : 2454-423X (Online)
International Journal of Research in Advanced Computer Science Engineering
A Peer Reviewed Open Access International Journal
www.ijracse.com

[14] M. A. Zulkifley, S. R. Abdani, and N. H. Zulkifley, "Pterygium-Net: a deep learning approach to pterygium detection and localization," Multimedia Tools and Applications, vol. 78, no. 24, pp. 34563-34584, 2019.