Final report

by Suhail Ss

Submission date: 08-May-2022 01:27PM (UTC+0530)
Submission ID: 1831011627
File name: report.pdf (3.79M)
Word count: 8436
Character count: 45765

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING,

SCHOOL OF ENGINEERING AND TECHNOLOGY, SHARDA UNIVERSITY, GREATER NOIDA

Plant Leaf Disease Detection Using Deep Learning

A project submitted

In partial fulfillment of the requirements for the degree of

Bachelor of Technology in Computer Science and Engineering

By:

SUHAIL SAIFI(2018013282)

BAL MUKUND(2018007647)

DANISH ALAM(2018014062)

Supervised by:

Dr. Hoor Fatima, Assoc.Prof(CSE)

May,20022

CERTIFICATE

This is to certify that the report entitled "**Plant Leaf Disease Detection Using Deep Learning**" submitted by "Mr. SUHAIL SAIFI (2018013282), DANISH ALAM (2018007647), BAL MUKUND (2018014062)" to Sharda University, towards the fulfillment of requirements of the degree of "**Bachelor of Technology**" is record of bonafide final year Project work carried out by him in the "Department of Computer Science and Engineering, School of Engineering and Technology, Sharda University".

The results/findings contained in this Project have not been submitted in part or full to any other University/Institute for award of any other Degree/Diploma.

Signature of Supervisor

Name: Dr. Ruqaiya Khanam

Designation: Prof (CSE)

Signature of Head of Department
Name: Prof. (Dr.) Nitin Rakesh
Place:
Date:

Signature of External Examiner

Date:

Acknowledgement

A major project is a golden opportunity for learning and self-development. We consider our self very lucky and honored to have so many wonderful people lead us through in completion of this project.

First and foremost we would like to thank Dr. Nitin Rakesh, HOD, CSE who gave us an opportunity to undertake this project.

My grateful thanks to **Dr. Ruqaiya Khanam** for his guidance in my project work.

Dr. Ruqaiya Khanam, who in spite of being extraordinarily busy with academics, took time out to hear, guide and keep us on the correct path. We do not know where we would have been without his help.

The CSE department monitored our progress and arranged all facilities to make life easier. We choose this moment to acknowledge their contribution gratefully.

Name and signature of Students

SUHAIL SAIFI (2018013282)

DANISH ALAM (2018007647)

BAL MUKUND (2018014062)

ABSTRACT

Identification of plant disease plays an important role as it prevents stunted growth which causes bad effects on yields. As agriculture plays a vital role in the Indian economy and different countries of the world, so there is a need to prevent losses in terms of production, quality, and quantity of agriculture yields due to plant disease. Earlier farmers used to monitor plant disease with the naked eye which was time-consuming and required a lot of expertise such as being able to identify a disease and disease-causing agent. But nowadays with advancements in technology, smart farming, and automatic techniques, plant disease can be easily identified and proper diagnosis can be done. It reduces a lot of work of monitoring such as in the case of big farms. Also at an early stage, it detects the symptoms of plant disease when they first appear on leaves. This paper reflects the potential of one such method-Plant disease detection using machine learning, using which one can detect plant disease. It includes the use of image processing techniques with the help of a machine learning algorithm to get a clear and defined image or to extract some useful insight from it. Also, CNN is used for image classification. CNNs are equipped with input, output, and hidden layers which help in process and in image classification.

***Index Terms*–**Disease Detection, K-means clustering, image processing, unhealthy leaf detection

Contents

Chapter1: INTRODUCTION

1.1: Problem Definition

This 'Plant Leaf Disease Detection Using Deep Learning' project is a software algorithm that can effectively detect leaf disease on an image input. This project reflects the potential of image processing techniques with the help of a deep learning algorithm to get a clear and defined image or to extract some useful insight from it. Also, CNN is used for image classification. CNNs are equipped with input, output, and hidden layers which help in process and in image classification. The primary goal of this project is to create the "Leaf Disease Detection" expert system, which employs a Deep Learning approach to define any type of leaf disease.

Deep Learning (DL) is one of the various subclasses of Machine Learning, and it has so far gone into three different stages of developments. The first generation originated in 1943 and its name was MCP, and was only useful in linear classification problems. Back Propagation (BP) was the second generation of the neural network. It was invented by Hinton in 1986. Nonlinear classification and mapping were also solved with the help of sigmoid functions, but around 1991 some problems of gradient vanishing were pointed out. Deep Learning (DL) is the third generation of neural networks. It was introduced in 2006, and as of now many DL models or architectures are being used for image detections or recognition.

As we wanted to formulate a real-time Plant Disease Detection System, to infer between the efficacy of pre-trained models over Multi-Class Dataset we collated Plant Leaf Dataset from Kaggle which is open to use. The Dataset consists of 15 Classes with a diverse variety of Plant Leaves. TensorFlow and Keras have been induced for working upon Deep Learning Architectures. OpenCv is used for reading and processing images. Numpy for saving the images in array format. Sklearn for one hot encoding,training and testing split, calculating the performance of the model. Matplotlib and Seaborn for plotting graphs in a much more advanced way. Gradio, an open-source python library for creating web-apps has also been utilized for creating a web app for Plant Disease Detection.

As the Data has been split into Train, and Test set. The Training Accuracy, and Test Accuracy has been used for explaining the architecture's potency over the particular dataset. Also, the Confusion Matrix is used for scrutinizing the model's efficacy in real-time over different labels/classes.

1.2: Project Overview/ Requirement Specification

1.2.1: Motivation:

- To find the interactions between the diseases causing agents and the host plant in relation to the overall environment.
- To identify various diseases in plants.
- To implement a method for preventing the disease and providing management for reducing the losses/damages caused by diseases.
- Prevent diseases on plants for Farmers.
- Help out the pesticide company in predicting the new pesticide solutions.

1.2.2: Overview:

Plant disease detection has become a very big domain to work on, because many of the works have been done earlier using many different technologies and methods in this field. We so far have read some of the papers for our survey and have analyzed and have found many different results for different techniques and methods used. The papers were on the same topic but of different methodologies like few of them were based on MobileNetV2, DenseNet121 InceptionResNetV2 and VGG16 etc. We have analyzed that even after doing so much research in this domain, there is a large scope of research in the future also because every proposed model has some constraints, some disadvantages and some advantages. The need for a perfect model to detect the plant leaf disease is also in hunt and this is the reason that these models are not getting used on a large scale for disease detection. This is a very good and very helpful domain to work on because when farming is done on a very big scale, it is not possible for farmers to look for diseases on every leaf of the plants, so further research and findings on this domain can be very helpful.

1.3: HARDWARE REQUIREMENTS

Minimum Requirements	**Windows**
Operating System	Windows 7
Processor	Dual core, Intel i3
RAM	2 GB RAM
DISK Space	The amount of disc space available depends on the partition size and whether or not online help files are allowed. The MathWorks installer would tell you how much disc volume your partition needs.
Graphics Adapter	8-bit graphics adapter and display (for 256 simultaneous colors)
CD_ROM drive	For installation from CD

Recommended Requirements		**Windows**		
	Processor	**RAM**	**DISK Space**	**Graphics Adapter**
PYCHARM-Python IDE	Intel i3	2 GB	1 GB for Pycharm only. 5 GB for a typical installation	A 32-bit or 64-bit OpenGL capable graphics adapter is strongly recommended

1.4: SOFTWARE REQUIREMENTS

Browser Used	Google Chrome
Language Used	Python
Technology & libraries Used	Tensorflow Numpy CNN Keras Matplotlib Seaborn
Gradio, an open-source python library for creating web-apps	

- **Tensorflow** :- Tensorflow is an open source planning data data flow for all areas of operations. A number-related library is used for AI applications such as neural networks.

- **Google colab:-** Google collaboration (open-supply Jupyter Notebook interface with excessive GPU space). Google collaboration is an unencrypted Jupyter package that does not need to be set up and works perfectly in the cloud. With colab, you can really write and use code , save and analyze percentages. Access functional computing software, all at no cost in the browser. Jupyter Notebook is an effective way to duplicate and write in your Python code to analyze facts. Instead of rewriting and rewriting the entire code again and again. You can actually write code types and apply them at a time.

- **Seaborn:-** Seaborn is a python library for comprehension by viewing matplotlib. It provides an undeniable level of interaction in drawing attractive and useful measurable drawings. For a brief introduction to the ideas behind the library, you can look up the first notes or paper.

- **OS:-** The OS module in python enables enabling and completing enrollment(envelope), importing, modifying, and separating the current catalog, and more.

- **Matplotlib:-** Matplotlib is a python programming language library and extension of NumPy . Provides article-based API for installing episodes in applications using the most useful GUI toolboxes such as Tkinter,wxPython, Qt or GTK.

- **Pandas:-** Pandas explain() is used to view certain basic mathematical details such as percentage, total, std etc. of a data framework or a series of numerical values. When this method is used in a series of character units, it returns a different output.

- **NumPy:-** NumPy is a Python library used for running with collections. It additionally has the cappotenial to paintings withinside the area of direct polynomial calculations, 4 variables and frames. NumPy evolved in 2005 with the aid of Travis Oliphant. It is an open supply enterprise and you could use it sparingly. NumPy stands for Numerical Python.

- **Kaggle** was used to find an online dataset.

- **GitHub** and **stackoverflow** were used for reference in case of syntax editing errors.

- **Gradio:**- It is used for developing the Web App.

- **OpenCV:**- It is used to read and process images.

- **Sklearn:-** It is used for One hot encoding , train test split and calculating the performance of the model.

- **Language:-**
- **SDFFG**
  - **Python 3 -** we have used python ,which is R-mathematical programming language instead of MATLAB for following reasons:
    – Python records format improved in MATLAB
    – Python code is much more understandable than MATLAB
    – Keras (with backend of tensorflow) - keras is a neural community API that includes Tensorflow,CNTk, Theno etc. Python programs such as NumPy, Matplotlib, Pandas math and editing graphs.

Chapter2: Literature Survey

2.1: Existing System:

Paper [1] shows a Google Net model developed for 79 diseases of 14 different plant species in the experimental area and a complex field environment as a diagnostic tool. The advantage of this method is that the appearance of multiple pathogens in one leaf can be detected in many smaller images. The accuracy of this model was almost 94%.

Paper [2], Dechant et al. showed a new CNN structure to identify the apple leaf disease. Using an Inception network and Alex Net-precursor network, the network was formed. Alex Net model got replaced by Inception network which reduced the number of trainable parameters, thereby reducing storage requirements. A single CNN gave the experimented result which shows the accuracy rate of 90.8%. When we used two first-level classifiers, the accuracy rate became 95.9%, and the accuracy rate rose to 97.8%, when three first –level qualifiers were used. The performance of this network was compared with SVM, BP, Alex Net, Google Net, ResNet20 and VGG16. The accuracy of these models was 68.73%, 54.63%, 91.19%, 95.69%, 92.76% and 96.32%, compared to 97.62%, which is the accuracy of the proposed Alex Net-precursor + Cascade Inception network.

Paper [3], a well-known sub-field known as Few-Shot Learning (FSL) was introduced. FSL can solve neural network recognition problem and new classes that have not emerged from training; neural network recognition problem is difficult to integrate due to the small number of test samples, thus improving the accuracy of detection of small data sets. In databases where there are only a few images or many, the transfer learning method (TL) also has a problem with low visibility accuracy [23]. This is because it is a very difficult situation to learn different aspects of a deep network, and it leads to problems that are very difficult to integrate or over-install. Therefore, plant disease data sets containing single or small samples cannot support Deep Learning (DL) architecture training. On the other hand, the DL model (in-depth learning model) needs re-training in order to successfully recognize several new classes that do not appear in the training set. After extensive testing, the results showed that Fiw Shots Learning (FSL) could successfully achieve average 90% accuracy with the help of only 60 training images, which were much better than refined transmission learning, their accuracy was 73%.

Paper [4], Emergence of accurate techniques inside the field of leaf-primarily based image type has proven surprising consequences. This paper makes use of Random woodland to come across diseased leaf from the facts sets created. Random forests are as a whole, learning approach for classification, regression and other obligations that operate through constructing woodland of the choice bushes all through the schooling time not like choice trees, Random forests conquer the drawback of over fitting in their schooling facts set and it handles each numeric and categorical facts. For extracting features of a photo, they used Histogram of an orientated Gradient (HOG). typical they used device learning to train the big records units available to locate the ailment present in plant life. The histogram of oriented gradients (HOG) is an detail descriptor utilized as a part of computer vision and image processing for the sake of object detection. right here we are making usage of 3 thing descriptors: 1. Hu moments 2. Haralick texture3. colour Histogram. This paper also well-known shows the accuracy of numerous gadget gaining knowledge of models.

.

Various Machine learning models	Accuracy(percentage)
Logistic regression	65.33
Support vector machine	40.33
K-nearest neighbour	66.76
CART	64.66
Random Forests	70.14

Paper [5], implementation of machine learning algorithm is seen to identify and detect a wide variety of plants disease. Timely performance and high accuracy of Machine learning can be found with various backgrounds and disease coverage. Also has the ability to train itself and helps farmers to manage diseases effectively. Being platform agnostic Tensor Flow Framework is used for the CNN model implementation. Tensor Flow also known as data flow graphs because of its numerical operation. Graph nodes are represented by mathematical operation(s) and Multi-dimensional arrays are represented by the graph edges. In Tensorflow machine learning algorithm can be implemented easily and quickly. More than 80,000 images were used for training and validation with different backgrounds and images. It gives accurate information to farmers about the diseased plants. For training the model 60% images were used and the rest 40% for validation. Before detecting the disease interface used in KrishiMitr allows the user to select the plant type. To check its functioning, KrishiMitr was tested on a linux-based computer having a touch screen interface with a sensor for image acquisition. All codes of KrishiMitr are freely available on the internet.

Paper [6], this paper outlines a method for diagnosing plant disease based on colour, edge detection and histogram pairing. Matching histogram is based on colour scheme and edge detection methods. The methodology used in this study is divided into two main categories. The first phase deals with healthy sample training and disease sampling. The second phase deals with experimental sample training and produces results based on edge detection and histogram matching. Each leaf has its own characteristics on the edges. Some leaf borders are saw, some are smooth and some have waves and so on. Also, the midrib alignment and vein pattern of the leaves are different. Therefore, the canny edge detection algorithm is used.

Paper [7], et al. K. Muthukannan Found spot disease in leaves and other diseased leaf types using Neural Network Algorithms. Different methods were used to classify the diseased plant leaves such as Feed Forward Neural Network (FFNN), Learning Vector Quantization (LVQ) and Radial Basis Function Networks (RBF).. Simulation results have shown the effective result of the scheme. With the help of this work, a machine learning based system can be formed for the good crop quality that helps to boost the Indian Economy.[1] Et. al Malvika Ranjan in the paper Detected and Classified Leaf Disease using Artificial Neural Network that captures the images. Segmentation and Artificial neural network (ANN) result gave the colour feature like HSV which are used to train and distinguish the unhealthy samples appropriately. ANN gave an accuracy of 80% and shown effective experimental results. With the help of this method cotton leaf disease can be detected accurately. Et al. Syafiqah Ishakais have shown how to get useful insight and analytical data from leaf images which is used to classify healthy and diseased leaves of medicine plant using image processing method. To extract image and to get data, algorithm of adjusted contrast and segmentation is used. Experiment is performed using Artificial Neural Network. Final experimental result shows that the RBF network performance is better than MLP network. By using deep convolutional network et al. Srdjan Sladojevic developed a new approach towards plant disease detection model using leaf image classification. This method facilitated a simple and efficient practice to detect disease in leaves. Model could recognize thirteen different types of plant diseases out of samples. To implement this disease detection model necessary steps are mentioned in this paper which can be useful to experts in agriculture. Caffe, developed by Berkley Vision and Learning Centre, was involved in deep CNN training. This model gave accuracy between 91% and 98% for experimented results. Et al. Emanuel Cortes used 86,147 images of diseased plants dataset and performed deep convolutional network and semi supervised methods to train and to classify crop species of 57 different classes. Also gave better result on the unsupervised data was resent. It gave accuracy above 80% in the training phrase under 5 epochs with the learning rate of 1e-5.

Paper [8], the basic knowledge of in-intensity mastering additionally affords a comprehensive evaluation of the modern studies work performed at the analysis of plant leaf disease using in-depth studies. Whilst sufficient training statistics are available, in-depth studies methods are able to hit upon plant leaf diseases with higher accuracy. The significance of gathering big sets of information with excessive variability, data magnification, transfer readings, and mapping that enable CNN to enhance segment accuracy, in addition to the importance of small sample leaf disease and the significance of hyper-spectral pictures for early detection of the plant disease mentioned on the equal time, there are boundaries. Most of the proposed DL frames inside the books have accurate results to locate on their internet site, but the consequences aren't appropriate for different data units, which imply that the model has a vulnerable consistency. Consequently, better dynamic models of DL are needed to prepare information sets for numerous diseases. In a few studies, a Plant Village facts set was used to assess the performance of DL models. Even though this database contains many pictures of some plant species with their own diseases, it's been taken to the lab. Therefore, it is predicted that a first-rate database of plant illnesses will be installed in the real world. Even though some research use hyper spectral photos of diseased leaves and a few DL prices are used to diagnose plant leaf disease early, troubles affecting the giant use of HSI in early detection of plant diseases continue to be to be solved. that is, at the primary detection of plant diseases, it's miles difficult to obtain written data sets, or even experienced professionals can mark while the disease signs are not detected, and define pixels for undiagnosed disease, most importantly for HSI to detect plant disorder.

Paper [9], there are many advanced methods for diagnosing and classifying plant sicknesses using diseased plant leaves. but, there may be no effective and effective business remedy that can be used to diagnose diseases. In our work, we've got used four different DL models (InceptionV3, InceptionResnetV2, EfficientNetB0, MobileNetV2) to detect plant diseases using healthy images and plant leaves. To train and test the model, we used the standard Plant Village database with 53,407 images, all taken under laboratory conditions. This database contains 38 different classes of 14 healthy and diverse leaf pictures. After dividing the database into 80–20 (80% of total training data, 20% of complete images for testing), we found a very high accuracy rate of 99.56% in the EfficientNetB0 model. On average, less time was needed to train images in MobileNetV2 and EfficientNetB0 formats, and it took 565 and 545 s / epoch, respectively, for colour photographs. Compared to other in-depth learning methods, the in-depth learning model used has a better ability to predict both accuracy and loss. The time required for model training was significantly lower than other machine learning methods. In addition, the MobileNetV2 architecture is a deep convolutional neural network that limits the number of parameters and functionality as possible, and can be easily implemented on mobile devices.

Paper [10], the main goal of this paper is to detect the sickness and take precautionary measures to protect the crop before it's too late. By doing this prevents the loss to the farmers and helps to maintain the monetary growth. Also minimizes the use of pesticides. This paper shows the use of Histogram which is used to predict the features and give to the classification model. The histogram is used to represent the frequency distribution or used to summarize the continuous data on an interval scale. They came in handy to represent numerical data in a friendly manner. They identify the diseased leaf and send the messages to the farmers. In this paper, Tomato and Maize leaves are taken and their disease is identified and predicted accurately using SVM and ANN algorithms. SVM is a supervised learning algorithm used for regression problems. It is primarily used for classification problems and also helpful in detecting outliers. It is highly effective in dimensional spaces. SVM shows better results in terms of generalization capabilities. ANN is software made by the inspiration of biological neurons which can emulate relationships and dependencies in a dataset. It is highly effective in terms of predicting the outcome of a new dataset. In this paper, they have shown different methodologies which include stages such as Data collection, Preprocessing, Feature extraction, Image segmentation, classification phases. They have used the dataset which contains 200 maize and tomato leaf features. Out of which 50 are healthy leaf and 110 leaf pictures are for training and testing. Here the use of SVM and ANN gave the accurate result. SVM gave 60 to 70 percentages. ANN gave 80 to 85 percentages. This methodology has been tested correctly on python software.

Paper [11], this paper provides a survey on various techniques and different classifiers aspects which can be used to detect the rice disease plant. They have also proposed a CNN model and a hardware prototype, using these different rice diseases such as rice blast, rice blight, brown spots, leaf smut, and sheath blight. Also highlights different techniques and their comparison based on classifiers such as SVM, KNN, ANN, etc. SVM is a supervised learning algorithm used for regression problems. It is primarily used for classification problems and also helpful in detecting outliers. It is highly effective in dimensional spaces. SVM shows better results in terms of generalization capabilities. ANN is software made by the inspiration of biological neurons which can emulate relationships and dependencies in a dataset. KNN algorithm is mainly used for pattern classification tasks. It is a supervised learning algorithm that is an initial step in a machine learning setup. IT is used for both classifications as well as regression problems. It is highly effective in terms of predicting the outcome of a new dataset. The proposed method to detect the rice disease includes various stages such as Data acquisition which deals with the sample picture dataset and hardware module consisting of raspberry pi and multimedia sensors. Preprocessing and feature extraction is carried out. Then CNN classifier is used to identify the infected leaf.

CNN is a deep neural network class that helps in image visualization. CNN is primarily used for image classification as it follows a hierarchal model to give higher accuracy.CNN is used in deep learning to recognize objects in an image. This proposed work also uses libraries such as panda, NumPy, TensorFlow. Also shown that CNN proposed model gave accuracy about 96 percentages. Also, able to identify diseases such as rice blast, blight, brown spots, etc. correctly and successfully. This model shows promising results and helps farmers to maintain their monetary growth.

Paper [12], this paper shows a method to detect monocot and dicot plant disease. This paper mainly shows two categories of family plants such as Monocot and Dicot family types. Monocot plant is those which contains only one seed leaf and have straight and parallel vein structure. Examples: Wheat, corn, rice, etc. Dicot families are those plants that have two leaves. Dicot plants example is roses, geraniums, magnolias, etc. this paper shows how this method helps to detect disease with more accuracy. It includes three steps such as segmentation of leaf to segment the diseased leaf using the K-means algorithm. Segmentation is one of the techniques in machine learning in which an image is broken down into various image segments. This reduces the image complexity and thus helps in further image analysis. K-means algorithm is an unsupervised algorithm that helps to identify k number of centroids and facilitates to the allocation of data points to the nearest cluster. It is highly beneficial in business assumptions. It deals with vector quantization. Next comes feature extraction and the last step is a classification which is done using SVM. SVM is a supervised learning algorithm used for regression problems. It is primarily used for classification problems and also helpful in detecting outliers. It is highly effective in dimensional spaces. SVM shows better results in terms of generalization capabilities and also shows different techniques and their limitations. They have segmented the affected part and extracted the feature of the diseased leaf part. This paper helps farmers to maintain their monetary growth and detect the disease accurately.

Paper [13], this examine has utilized deep mastering competencies to reap computerized plant ailment detection structures. This system is based on a easy type mechanism which exploits the function extraction functionalities of CNN. For prediction ultimately, the model makes use of the completely connected layers. The researches become done using the publicly on hand series of 70295 images, and a hundred photos from experimental conditions and real environment. The machine has achieved a standard 98% testing accuracy on publicly on hand dataset, and carried out well on snap shots of Sukkur IBA university vegetation. it is concluded from accuracy that CNN is fairly appropriate for automated detection and prognosis of flowers. This machine may be integrated into mini-drones to stumble on diseases from flora in cultivated regions. Although this device is skilled on a Plant Village dataset with handiest 38 classes it can tell if the plant has a sickness or not as by some means symptoms are the same in all kinds of flowers. in addition, extra actual surroundings photographs may be introduced to the dataset to improve the accuracy on real-circumstance photographs of leaves and classify extra plant kinds in addition to disease kinds. inside the destiny, this system also can undertake a 3-layer method wherein the first layer detects if there's any plant in an image or not, 2d layer tells the plant kind and the 0.33 layer tells if there is any disorder or no longer and what kind of disease is there if any.

Paper [14], in this paper, they mentioned how ML in standard and DL in particular helped to diagnose diseases in plant life whilst illnesses are not identified nicely, they disrupt the harvest and eventually lead to long-term troubles, such as global warming or even famine. The proposed work summarizes many studies on the evolution of plant diseases and the identification of different ML strains. The proposed manuscript also reflects a well-received list of CV methods on this site, making it a comprehensive area of research that should be explored soon. [a] identification of disease is one of the main areas to pay attention. As each disease has many stage associated with it. Also systems must have the capabilities to suggest measures about specific disease stages. It helps to reduce damage percentage and boost the agricultural production. [b] Disease Quantification is another area. The extent of the damage caused by the disease must be identified. This plays an important role as remedial actions could be taken according to the disease severity. Helps to detect the infected proportion of a agricultural production. Also it helps to analyse whether specific chemicals are needed or not. As we know chemicals are harmful for the health. [c] Many Applications and articles are available about disease identification which is free to use. It does not give a fast and accurate result. So availability of this recognition model will help farmers to detect and identify disease which will help to increase to their revenue. [d] Dealing with the difficulty of the data, specifically during the training stage, the best option to look at is transfer learning- a heterogeneous domain. 3D convolution, temporal pooling, LSTMs, optical flow frames are some of the keywords which can be used while considering automatic plant disease detection.

Paper [15], high accuracy, training efficiency and unsupervised characteristics of deep learning is shown which used recognition model is. But in complex environment a lot of changes can be seen in accuracy and practicability during detection. To fix this problem TL, RPN and CV algorithm has been used. Due to this it's adaptability in complex environments has increased and also the accuracy of detection of disease. This paper reduces the quality requirements of the convolutional neural network on the data set to get the accurate results. This model helps to boost agricultural production and prevent crop losses. This paper helps to improve the existing theory and the accuracy. This model helps in sustainable development of smart agriculture. Recognition model based on deep learning can overcome the complexity of the environment and helped to improve the accuracy of identification but there are still some problems to be looked at. Neural network will be used in future research, to give zero initial sets corresponding to various leaves, which increases calculation limit for Chan–Vese algorithm(a iterative process) and speed up the training speed, ahead of time.

S.no	Findings		
	Title	Accuracy	Technologies Used
[1]	LEAF DISEASE DETECTION BY WELL -KNOWN DEEP LEARNING ARCHITECTURES	94%	Google Net, Deep Learning
[2]	NEW/MODIFIED DL ARCHITECTURE FOR LEAF-DISEASE DETECTION	97.62%	Alex Net-precursor, Cascade Inception
[3]	LEAF-DISEASE DETECTION BASED ON SMALL SAMPLES	90%	Few Shots Learning
[4]	Plant Disease Detection Using Machine Learning	89%	Various Models

[5]	KrishiMitr (Farmer's Friend): Using Machine Learning to Identify Diseases in Plants	91.3%	Tensor Flow
[6]	Mango Leaf Deficiency Detection using digital image processing and Machine Learning	90%	Image processing
[7]	Detecting Jute Plant Disease Using Image Processing and Machine Learning	89.99%	Colour co-occurrence methods, Multi SVM classifier
[8]	Detection Of Unhealthy Plant Leaves Using Image Processing and Genetic Algorithm with Arduino	93%	Genetic algorithm, Arduino, Masking the green pixel and colour co-Occurrence method.
[9]	Detection and measurement of paddy leaf disease symptoms using image processing	91,4%	ANN,FUZZY classification, SVM, K-means algorithm, colour co-occurrence methods.
[10]	Plant Disease Identification Using SVM and ANN Algorithms	92%	SVM and ANN Algorithms
[11]	Rice Plant Disease Detection and Classification Techniques	87%	SVM, KNN, ANN
[12]	A Survey on Recognition of Plant Disease with Help of Algorithm	93%	SVM
[13]	Plant Disease Detection using Deep Learning	96%	CNN
[14]	Detection of Plant Diseases Based on Image: From Classical Machine Learning to Deep Learning	94.3%	CNN
[15]	Deep Learning Based Plant Disease Identification	97.1%	RPN , CV , and TL algorithm

Working of the algorithm

CNN model - In deep learning , the convolution neural network (CNN / ConvNet) is a class of deep-seated networks, widely used to analyze visual images. Each neuron receives multiple inputs and takes a weighted load over you, which transmits it to the opening function and response output. CNNs are primarily used to classify images, group them together, and then create recognition.

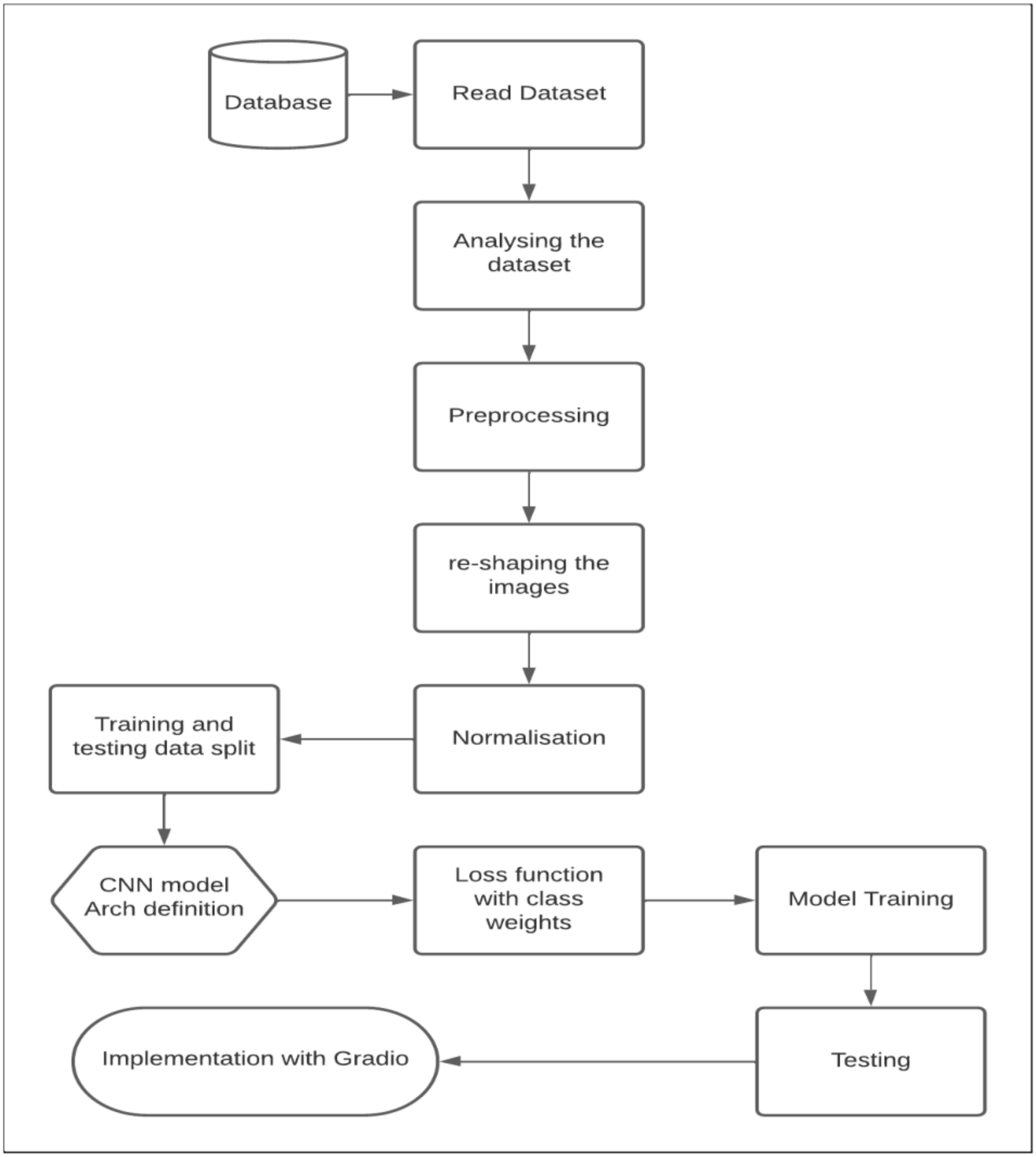
fig: CNN architecture

Layers of CNN model:

- **Convolution 2D**
- **MAX pooling 2D**
- **Dropout**
- **Flatten**
- **Dense**
- **Activation**

a. **Convolution 2D :** In convolution 2D extract the features from the entered image. It gives the output in matrix form.
b. **MAX pooling 2D:** In MAX pooling 2D it captures very large details on a modified performance map.
c. **Cessation:** Cessation is randomly determined by neurons that are skipped during training.
d. **Flatten:** Complete the feed output into a fully integrated layer. Provides statistics in the listing form.
e. **Density:** A linear function where each installation is attached to each exit in a weighted manner. It is compatible with an indirect activation feature.
f. **Getting started:** Use the sigmoid feature and expect a zero chance and 1. In the meger version we used binary entropy because we have zeros and layers1. We used the Adam optimizer for the compilation version.

2.2. Proposed System:

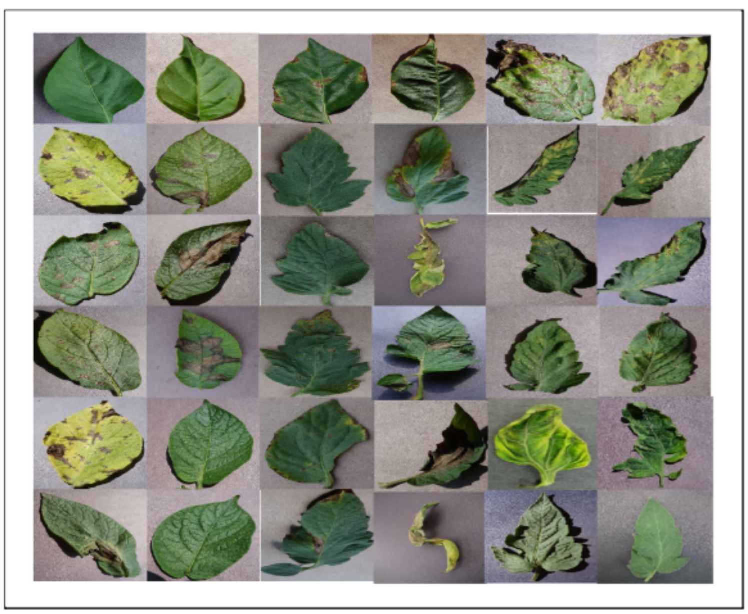


2.2.1: Technical flow

- Read Dataset
- Analyzing
- Preprocessing
- 1.Change to same shape
- 2.Normalisation
- Train Test Split
- CNN model architecture definition
- Loss function with class weights
- Model training
- Testing
- Implementation with Gradio

2.2.2: Dataset

As we wanted to formulate a real-time Plant Disease Detection System, to infer between the efficacy of pre-trained models over Multi-Class Dataset we collated Plant Leaf Dataset from Kaggle which is open to use. The Dataset consists of 15 Classes with a diverse variety of Plant Leaves elucidated in Figure. There are 20935 instances of Data for scrutinizing the model's performance. The illustration of the Dataset is shown in Figure.



2.3: Feasibility Study

Feasibility study is the first survey of users' knowledge that may estimate service needs, costs, benefits, and feasibility. The feasibility study examines the implementation and operation of the system due to a number of issues. Resources needed for use, such as computer equipment, personnel, and costs, are estimated in this section. Estimated costs are compared to available resources, and system cost analysis is performed. The main objective of a feasibility study is to determine the feasibility of the project, the feasibility of the project, the feasibility of the project. and the feasibility of the planning. Its purpose is to ensure that all project input data is available. As a result, we checked

- Technical feasibility
- Financial feasibility
- Social feasibility

2.3.1: Technical feasibility

Diagnosis and Classification of Plant Disease using Deep Learning is a complete project based on In-Depth Reading with minimal Machine Learning. Technologies and associated tools:

CNN

Tensorflow

Google Collaboratory

Seaborn

2.3.2 Resource Feasibility:

- Programming device (Laptop)
- Programming tools (freely available)
- Tensorflow
- Google Collaboratory
- Dataset
- Programming individuals

2.3.3 Financial Feasibility:

It will cost a very low amount for a project based on Deep Reading(simultaneous). Bug fixes and repair works will have lower costs associated with them. Customers will reap many benefits in addition to the costs associated with it.

2.3.4: Social Feasibility:

A component of the audit is to review the level of structure affirmation by the customer. This consolidates the course of setting up the customer to use the structure properly. The level of customer affirmation depends just upon the systems used to show the customer the structure and get that individual familiar with it.

Chapter3: System Analysis and Design

3.1. Software Requirement Specification:

Browser Used	Google Chrome
Language Used	Python
Technology & libraries Used	Tensorflow Numpy CNN Keras Matplotlib Seaborn
Gradio, an open-source python library for creating web-apps	

- **Tensorflow** :- Tensorflow is an open source planning data data flow for all areas of operations. A number-related library is used for AI applications such as neural networks.

- **Google colab:-** Google collaboration (open-supply Jupyter Notebook interface with excessive GPU space). Google collaboration is an unencrypted Jupyter package that does not need to be set up and works perfectly in the cloud. With colab, you can really write and use code , save and analyze percentages. Access functional computing software, all at no cost in the browser. Jupyter Notebook is an effective way to duplicate and write in your Python code to analyze facts. Instead of rewriting and rewriting the entire code again and again. You can actually write code types and apply them at a time.

- **Seaborn:-** Seaborn is a python library for comprehension by viewing matplotlib. It provides an undeniable level of interaction in drawing attractive and useful measurable drawings. For a brief introduction to the ideas behind the library, you can look up the first notes or paper.

- **OS:-** The OS module in python enables enabling and completing enrollment(envelope), importing, modifying, and separating the current catalog, and more.

- **Matplotlib:-** Matplotlib is a python programming language library and extension of NumPy . Provides article-based API for installing episodes in applications using the most useful GUI toolboxes such as Tkinter,wxPython, Qt or GTK.

- **Pandas:-** Pandas explain() is used to view certain basic mathematical details such as percentage, total, std etc. of a data framework or a series of numerical values. When this method is used in a series of character units, it returns a different output.

- **NumPy:-** NumPy is a Python library used for running with collections. It additionally has the cappotenial to paintings withinside the area of direct polynomial calculations, 4 variables and frames. NumPy evolved in 2005 with the aid of Travis Oliphant. It is an open supply enterprise and you could use it sparingly. NumPy stands for Numerical Python.

- **Kaggle** was used to find an online dataset.

- **GitHub** and **stackoverflow** were used for reference in case of syntax editing errors.

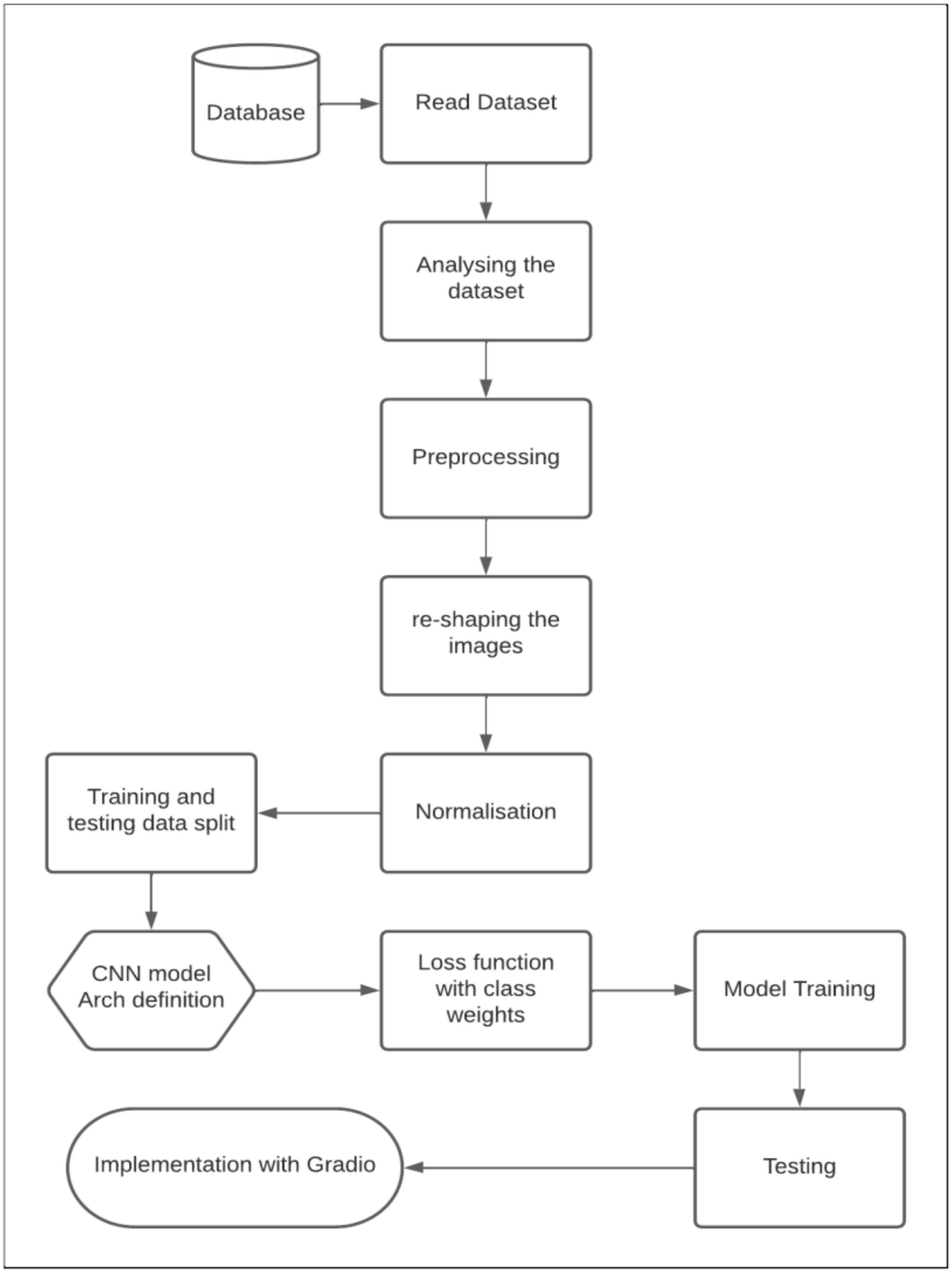
- **Gradio:**- It is used for developing the Web App.

- **OpenCV:**- It is used to read and process images.

- **Sklearn:-** It is used for One hot encoding , train test split and calculating the performance of the model.

- **Language:-**
- **SDFFG**
  - **Python 3 -** we have used python ,which is R-mathematical programming language instead of MATLAB for following reasons:
    – Python records format improved in MATLAB
    – Python code is much more understandable than MATLAB
    – Keras (with backend of tensorflow) - keras is a neural community API that includes Tensorflow,CNTk, Theno etc. Python programs such as NumPy, Matplotlib, Pandas math and editing graphs.

3.2. Flowchart:



3.1. Design and Test Steps:

The survey of literature gave us an impactful insight about the importance of considering the complexity of the utilized architectures. Thus, we moved on to the explication of our proposed outlook of Plant Disease Detection encapsulating varied modus operandi. The scrutiny of varied pre-trained models was done based on diverse parameters to inhibit computational complexity. There are certain factors which impacts the tuning of certain algorithms which can be elucidated as :

- High Number of Parameters in a model makes it Sluggish to be operated in real-time.
- High Number of Recurring Units, i.e., training over the same structure again and again. After a certain instance, the models perform extremely poor and causes overfitting.
- Utilization of Activation Functions which are complicated deteriorates the performance of the model.
- Deeper the Network more liable it is for causing Overfitting and thereby hindering the modus operandi.

As a result, it becomes prominent to consider the most feasible model for the applicability with greater authenticity. Thus, we collated the most famous Light-weight and Heavy-weight pre-trained models to scrutinize the minutes of the impact of depth of the models, number of parameters, and size of the model over a certain problem statement. We included MobileNetV2 and DenseNet121 as Light-weight Systems and InceptionResNetV2 and VGG16 as Heavy-weight Systems. The basis of selection of these models is elucidated in Table-I.

S.No.	**Pre-Trained Architectures**	**Model Size (In MB)**	**No. of Parameters (approx. in Millions)**	**No. of Layers**
1.	MobileNetV2	14	3.5	88
2.	DenseNet121	33	8	121
3.	InceptionResNetV2	215	55.8	572
4.	VGG 16	528	138	23

Now, as the model selection was carried out through explication over varied references, we tried to understand the architecture of each model. As a result, Figure.1, Figure.2, Figure.3, and Figure.4 demonstrate the graphical illustration of specifics of the model and the minutes of layers in those models.

A. MobileNetV2:

As per the Table-I, MobileNetV2 consists of 88 Layers with 3.5 Million Parameters overall and the size of the model is just 14MB. Thus, it has been considered as the Light-Weight Model when compared to other pre-trained models.

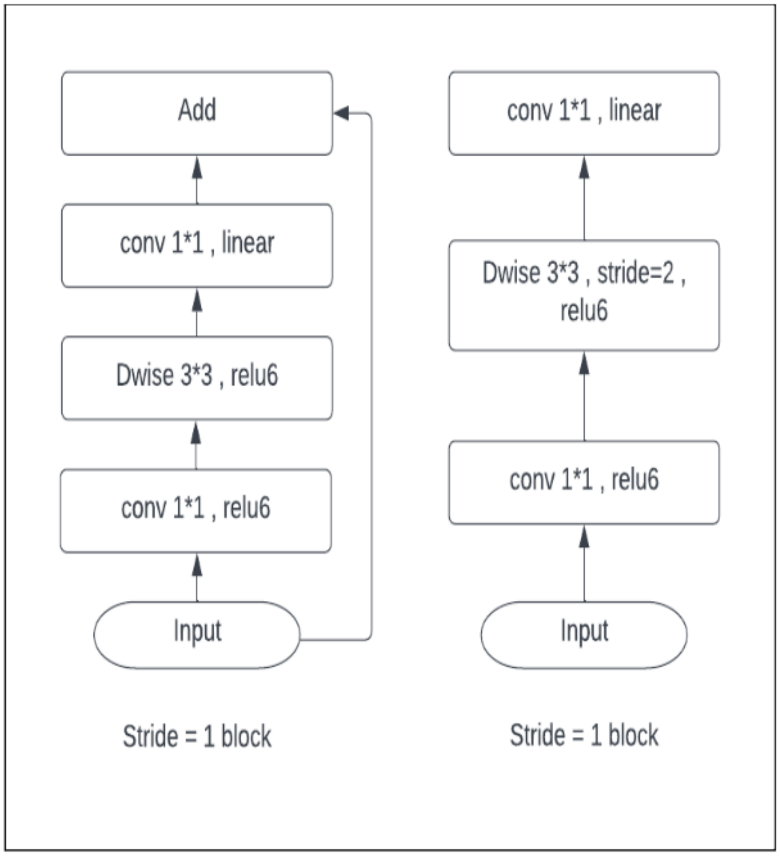


Fig .1. Demonstration of MobileNetV2 Architecture with Residual and Downsizing Block [16].

B. DenseNet121:

As per the Table-I, DenseNet121 consists of 121 Layers overall, along with 08 Million Parameters colligating a size of 33MB, making it also a Light-Weight System when compared to other pre-trained models.

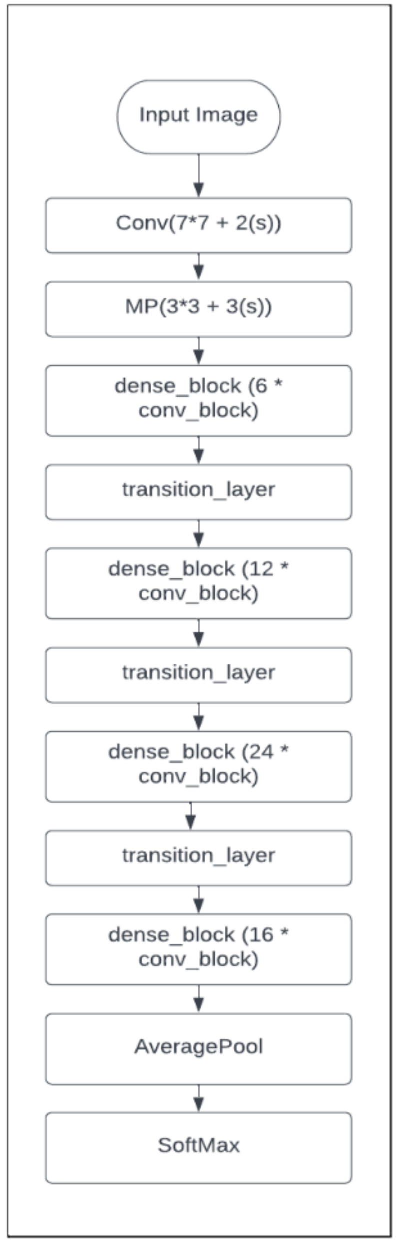


Fig .2. Demonstration of DenseNet121 Architecture [16].

C. InceptionResNetV2:

As per the Table-I, InceptionResNetV2 is the collation of Inception and ResNet Family of Pre-Trained architectures consisting of 572 Layers with 55.8 Parameters. The Size of the Model is 215MB making it a Heavy-Weight System when compared with other pre-trained models.

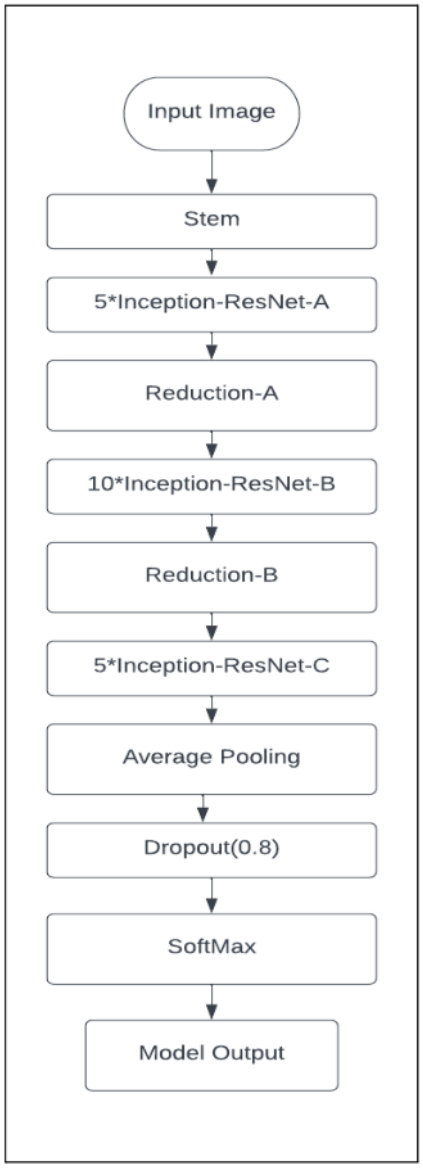


Fig .3. Demonstration of InceptionResNetV2 Architecture [16].

D. VGG16: As per the Table-I, VGG16 consists of 23 Layers with 138 Million Parameters. The size of the model stands to be 528MB, making it a Heavy-Weight System when compared to other pre-trained models.

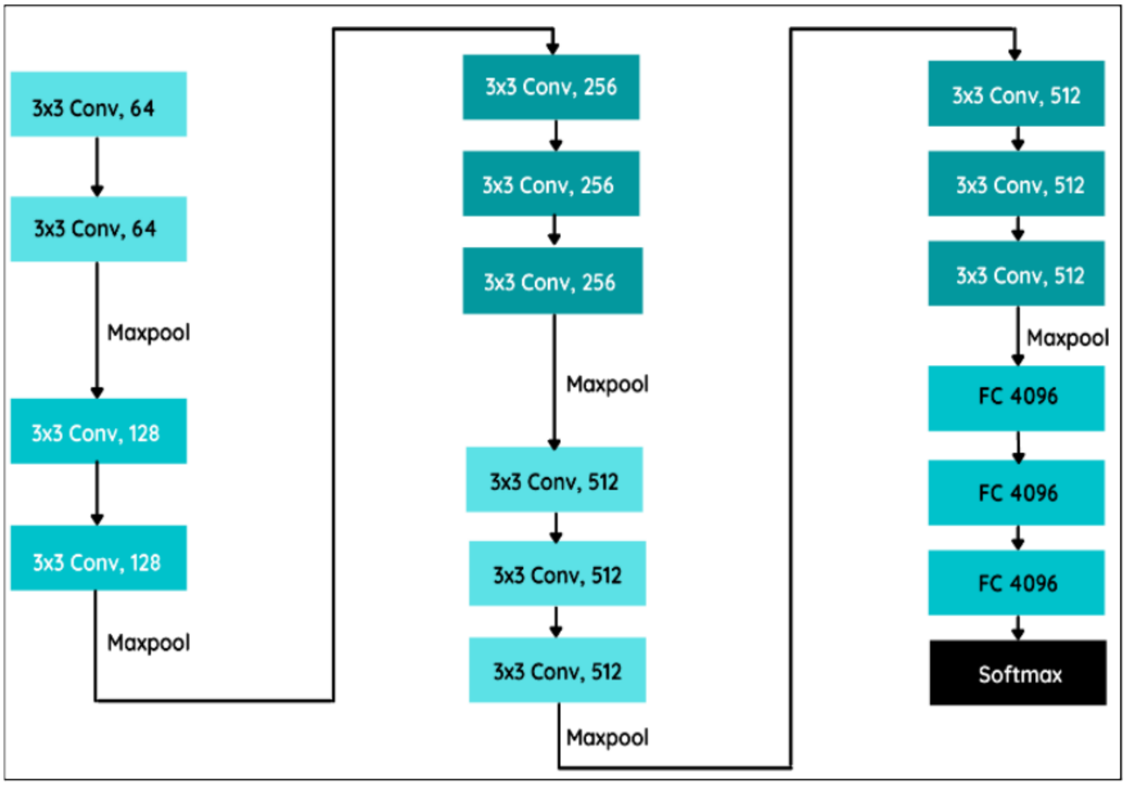


Fig .4. Demonstration of VGG16 Architecture [16].

These architectures stand out to be prominent for our modus operandi and aids us to understand the behavior of models for Multi-Class Dataset and thereby creating a real-time system for Plant Disease Detection.

3.1. Testing Process:

A. Dataset Utilized: As we wanted to formulate a real-time Plant Disease Detection System, to infer between the efficacy of pre-trained models over Multi-Class Dataset we collated Plant Leaf Dataset from Kaggle which is open to use. The Dataset consists of 15 Classes with a diverse variety of Plant Leaves elucidated in Figure.5. There are 20935 instances of Data for scrutinizing the model's performance. The illustration of Dataset is shown in Figure.5.

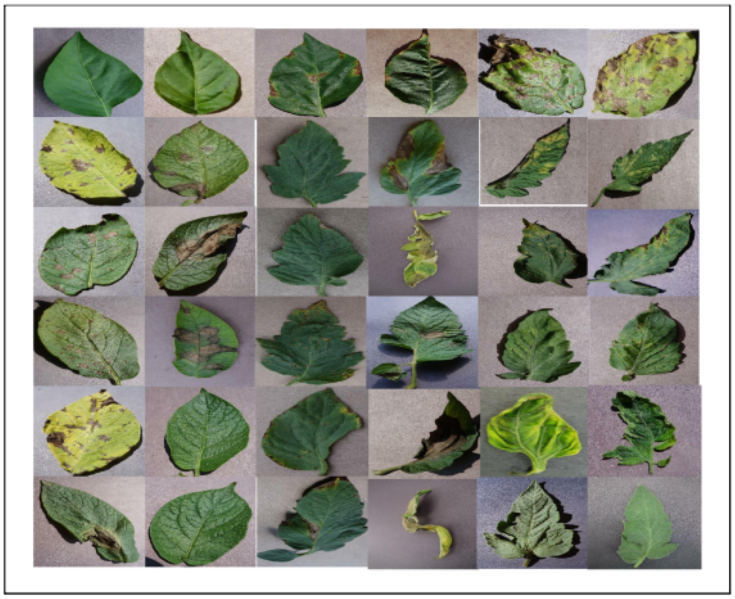


Fig .5. Demonstration of Plant Leaves Dataset.

B. Frameworks Utilized: TensorFlow and Keras have been induced for working upon Deep Learning Architectures. OpenCv is used for reading and processing images. Numpy for saving the images in array format. Sklearn for one hot encoding,training and testing split, calculating the performance of the model. Matplotlib and Seaborn for plotting graphs in a much more advanced way. Gradio, an open-source python library for creating web-apps has also been utilized for creating a web app for Plant Disease Detection.

C. Accuracy Parameters: As the Data has been split into Train, and Test set. The Training Accuracy, and Test Accuracy has been used for explaining the architecture's potency over the particular dataset. Also, the Confusion Matrix is used for scrutinizing the model's efficacy in real-time over different labels/classes.

D. Workflow: The dataset that we have taken was having 15 categories of plants. The dataset was very unbalanced, due to which we got many false positive predictions. So in order to handle the unbalanced dataset there are three ways,

1. Make an equal number of samples in all the categories by reducing the samples.
2. Generate more samples in those categories of plants which were having less number of samples.
3. Make use of class weights.

So we used the class weight method to balance the dataset. Working steps of class weight method - the class weight doesn't reduce or generate samples to make the dataset balanced, instead it makes the dataset balanced while training by providing an array, defining how much weight to be given to each category or class, such that the model trains or learns that the dataset is in equal distribution.

Implementation

10

After the Implementation of the proposed outlook, we got phenomenal results for the model that we have selected for building the Plant Disease Detection System which was VGG16 and a worthy insight was procured through it. Figure.7 and Figure.8 represent the graphical perspective of the model in terms of Accuracy and Loss.

```
[ ] from google.colab import drive
    drive.mount('/content/drive')

    Mounted at /content/drive
```

```
[ ] import os
    import IPython.display as ipd
    import matplotlib.pyplot as plt
    import numpy as np
    import pandas as pd
    import IPython.display as ipd
    import time
    from collections import Counter, OrderedDict
    import cv2
    import random
    import seaborn as sns
    from pathlib import Path
    import xml.etree.ElementTree as ET
    from natsort import natsorted

    from sklearn.preprocessing import LabelEncoder
    from sklearn.model_selection import cross_val_score
    from sklearn.model_selection import train_test_split
    from sklearn import metrics
    from sklearn.utils import class_weight
    from sklearn.utils.class_weight import compute_class_weight

    import keras, tensorflow
    from keras.utils.np_utils import to_categorical
    from keras.utils.np_utils import to_categorical
    from keras.models import Sequential
    from keras.layers import Dense, Dropout, Flatten, Conv2D, Conv1D, MaxPooling2D, MaxPooling1D, LeakyReLU
    from tensorflow.keras.optimizers import Adam,SGD,Adagrad,Adadelta,RMSprop,Adamax, Nadam
    from keras.callbacks import ModelCheckpoint
    from keras.preprocessing.image import ImageDataGenerator
```

Read Dataset

```
path = '/content/drive/MyDrive/PlantVillage'
folders = []
for i in os.listdir(path):
  folders.append(i)
  print(i, len(os.listdir(os.path.join(path, i))))
```

```
Tomato_Late_blight 1909
Tomato_Early_blight 1000
Tomato_healthy 1591
Tomato__Tomato_mosaic_virus 373
Tomato_Leaf_Mold 952
Pepper__bell___Bacterial_spot 1007
Tomato_Spider_mites_Two_spotted_spider_mite 1676
Potato___Early_blight 1000
Tomato__Tomato_YellowLeaf__Curl_Virus 3214
Tomato_Bacterial_spot 2127
Pepper__bell___healthy 1478
Potato___healthy 152
Potato___Late_blight 1000
Tomato__Target_Spot 1404
Tomato_Septoria_leaf_spot 1771
```

```
from natsort import natsorted
imgs = []
labels = []
shapes = []
dir = path
leng = []
folders = natsorted(os.listdir(path))
#folders.remove('.ipynb_checkpoints')
```

```
leng = []
folders = natsorted(os.listdir(path))
#folders.remove('.ipynb_checkpoints')
for folder in folders:
  print(folder)
  for files in tq(natsorted(os.listdir(os.path.join(dir, folder)))):
    try:
      image = cv2.imread(os.path.join(dir, folder, files))
      shapes.append(image.shape)
      image = cv2.resize(image, (96, 96))
      labels.append(folder)
      imgs.append(image)
    except Exception as e:
      print(e)

imgs = np.array(imgs)
labels = np.array(labels)
Counter(shapes)
```

```
Pepper__bell___Bacterial_spot
100% 1007/1007 [00:15<00:00, 261.10it/s]
Pepper__bell___healthy
100% 1478/1478 [00:18<00:00, 241.05it/s]
Potato___Early_blight
100% 1000/1000 [00:13<00:00, 268.78it/s]
Potato___Late_blight
100% 1000/1000 [00:12<00:00, 260.68it/s]
Potato___healthy
100% 152/152 [00:01<00:00, 171.45it/s]
Tomato_Bacterial_spot
100% 2127/2127 [00:29<00:00, 270.03it/s]
```

Calculating Class Weight

```
[ ] yy = np.array(yy)
    print(yy.shape)

    ()
```

```
[ ] from sklearn.utils import class_weight
    print(yy.shape)
    yy = yy.reshape(20653)
    print(yy.shape)
    print(np.unique(yy))
    class_weights = class_weight.compute_class_weight(class_weight='balanced', classes=np.unique(yy), y=yy)
    class_weights
```

One Hot Encoding

```
[ ] le=LabelEncoder()
    y=le.fit_transform(labels)
    y=to_categorical(y,15)
    y.shape
    labels = 0
```

Normalization

```
imgs = imgs/255
imgs.shape

(20653, 96, 96, 3)
```

```
[ ] from google.colab.patches import cv2_imshow
    cv2_imshow(imgs[11000]*255)
```

Train Test Split

+ Code + Text

```
[ ] from sklearn.model_selection import train_test_split
    np.random.seed(24)
    x_train,x_test,y_train,y_test=train_test_split(imgs,y,test_size=0.25,)
    print(x_train.shape)
    print(x_test.shape)
    print(y_train.shape)
    print(y_test.shape)
    imgs = y = 0

    (15489, 96, 96, 3)
    (5164, 96, 96, 3)
    (15489, 15)
    (5164, 15)
```

Loss Function

```
[ ] def lossFunc(true, pred):

            axis = -1 #if channels last
            weightsList = [1.38007355, 0.93094272, 1.37593333, 1.37593333, 9.05219298,
             0.6468892 , 1.37593333, 0.72076131, 1.44530812, 0.77692452,
             0.82096261, 0.9800095 , 0.42877324, 3.68882931, 0.86482296]

            classSelectors = K.argmax(true, axis=axis)

            classSelectors = [K.equal(K.cast(i,tf.int64), classSelectors) for i in range(len(weightsList))]
            #print('classSelectors2', classSelectors)
            #for i in classSelectors:

            classSelectors = [K.cast(x, K.floatx()) for x in classSelectors]
```

VGG Arch

```
[ ] ################################################################
    from tensorflow.keras.applications import VGG16, VGG19
    from tensorflow.keras.models import Sequential
    from tensorflow.keras.models import Sequential
    from tensorflow.keras import layers
    from keras.layers import Dense, Flatten, Conv1D, Dropout
    from keras.layers import MaxPooling1D
    from keras.layers.embeddings import Embedding
    from keras.preprocessing import sequence
    from keras.utils.vis_utils import plot_model
    from keras.layers.merge import concatenate
    from keras import *
    from keras.metrics import categorical_accuracy


    # Block 1
    from keras import layers
    from keras.layers import Activation, BatchNormalization
    from tensorflow.keras.losses import Huber
    input_img = Input((96, 96, 3))
    x = layers.Conv2D(64, (3, 3),
                      padding='same',
                      name='block1_conv1')(input_img)
    x = BatchNormalization()(x)
    x = Activation('relu')(x)
```

```
Total params: 15,946,319
Trainable params: 15,937,871
Non-trainable params: 8,448
_________________________________________________________________
```

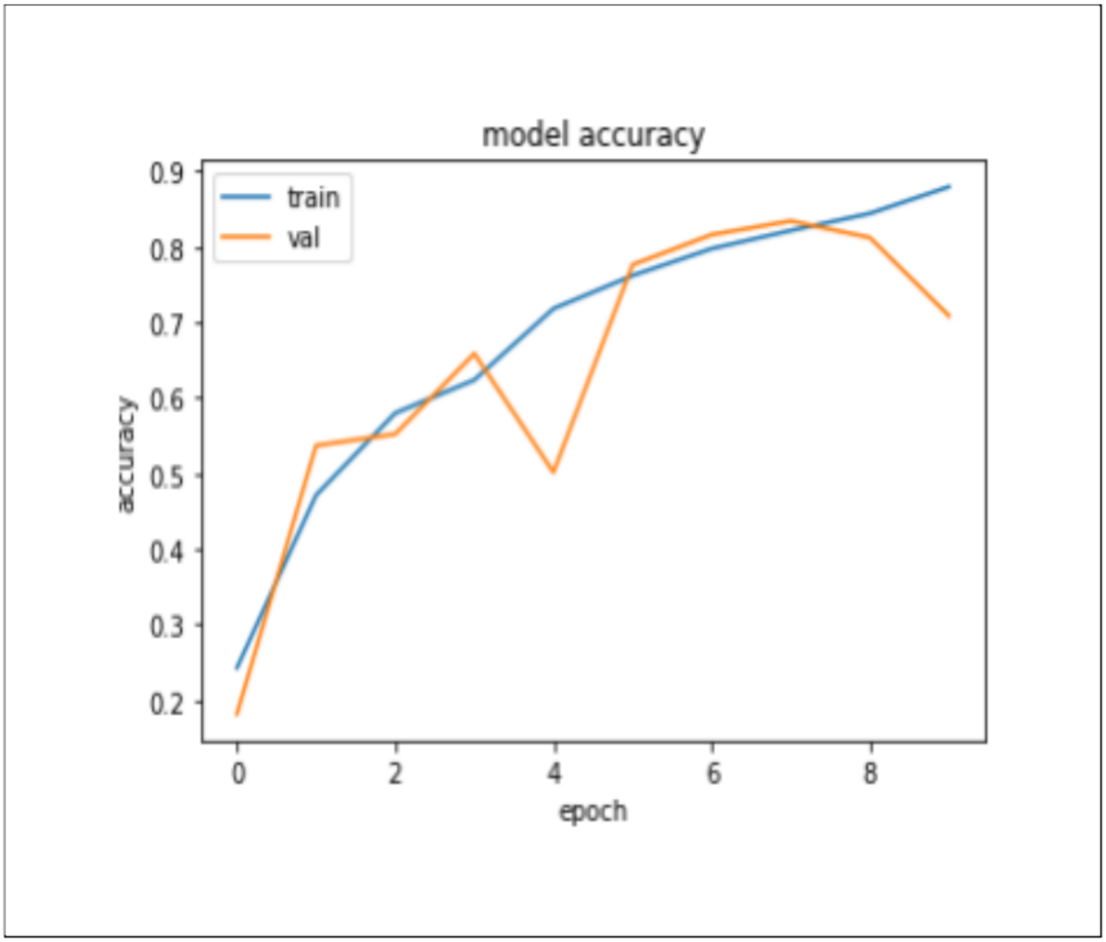
```python
start = time.time()
#model = load_model('./rb_model.h5')
red_lr= ReduceLROnPlateau(monitor='val_loss',patience=4,verbose=2,factor=0.001,min_delta=0.01)
filepath = r"/content/drive/MyDrive/mod/rb_model11.h5"
check=ModelCheckpoint(filepath = filepath, verbose = 1, save_best_only = True)

History = model.fit(x_train, y_train , epochs=10, verbose = 1 ,validation_data=(x_test, y_test),
                       callbacks = [check, ],
                       batch_size = 32, shuffle=True,
                       )

print(time.time()-start)
```

```
Epoch 1/10
485/485 [==============================] - ETA: 0s - loss: 2.4276 - acc: 0.2429
Epoch 1: val_loss improved from inf to 4.54341, saving model to /content/drive/MyDrive/mod/rb_model11.h5
485/485 [==============================] - 73s 123ms/step - loss: 2.4276 - acc: 0.2429 - val_loss: 4.5434 - val_acc: 0.1818
Epoch 2/10
484/485 [============================>.] - ETA: 0s - loss: 1.8057 - acc: 0.4715
Epoch 2: val_loss improved from 4.54341 to 1.51000, saving model to /content/drive/MyDrive/mod/rb_model11.h5
485/485 [==============================] - 57s 118ms/step - loss: 1.8058 - acc: 0.4715 - val_loss: 1.5100 - val_acc: 0.5372
Epoch 3/10
484/485 [============================>.] - ETA: 0s - loss: 1.3838 - acc: 0.5801
Epoch 3: val_loss improved from 1.51000 to 1.39498, saving model to /content/drive/MyDrive/mod/rb_model11.h5
485/485 [==============================] - 58s 119ms/step - loss: 1.3850 - acc: 0.5800 - val_loss: 1.3950 - val_acc: 0.5523
Epoch 4/10
484/485 [============================>.] - ETA: 0s - loss: 1.2679 - acc: 0.6240
Epoch 4: val_loss improved from 1.39498 to 1.18419, saving model to /content/drive/MyDrive/mod/rb_model11.h5
485/485 [==============================] - 59s 121ms/step - loss: 1.2680 - acc: 0.6239 - val_loss: 1.1842 - val_acc: 0.6588
Epoch 5/10
484/485 [============================>.] - ETA: 0s - loss: 0.9563 - acc: 0.7184
Epoch 5: val_loss did not improve from 1.18419
485/485 [==============================] - 57s 117ms/step - loss: 0.9563 - acc: 0.7183 - val_loss: 2.3492 - val_acc: 0.5017
```

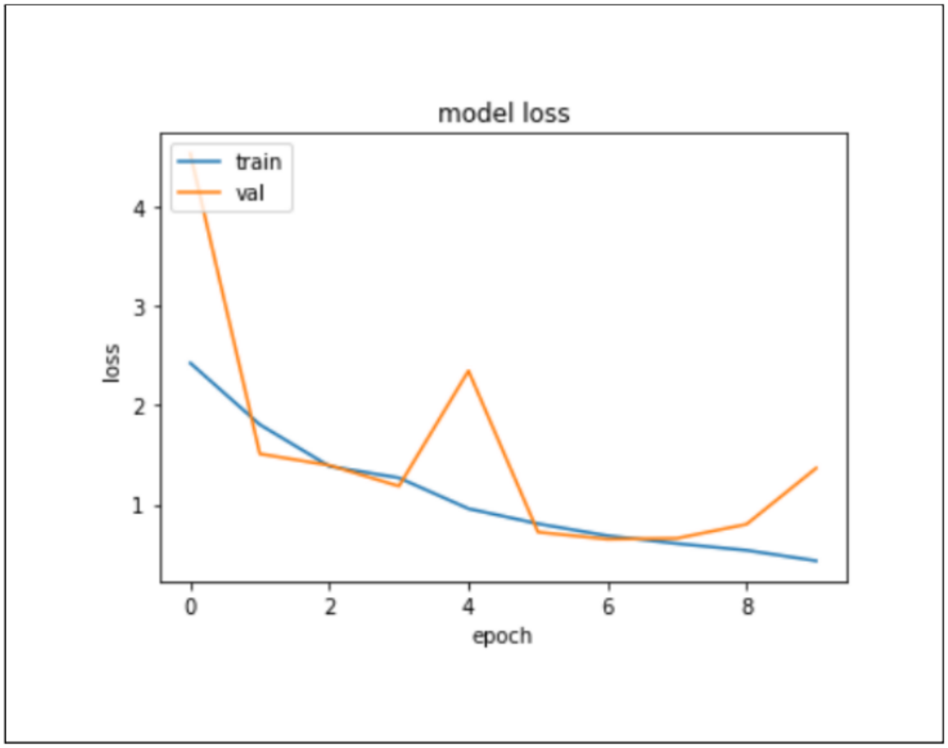
```
import gradio as gr
import tensorflow as tf
import requests
#Run the F1-score function and Loss Function
model = load_model(r'/content/drive/MyDrive/mod/rb_model11.h5', custom_objects={'lossFunc':lossFunc})

# Download human-readable labels for ImageNet.
labels = natsorted(os.listdir('/content/drive/MyDrive/PlantVillage'))

def classify_image(inp):
  img_final = inp.copy()
  inp = inp.reshape((1, 96, 96, 3))
  inp = inp/255
  pred = model.predict(inp)
  flag = labels[np.argmax(pred[0])]
  return(flag)

  #return(output)
image = gr.inputs.Image(shape=(96, 96))
#label = gr.outputs.Label(num_top_classes=3)
#iface = gr.Interface(image_mod, gr.inputs.Image(type="pil"), "image")

v = 'body{background-image: url("https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcTLAYw2YT6u7VSCoN3Yk4ZxEIPkNhTAkVaOPA&usqp=CAU"); background-re
v = 'body{background-image: url("https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcTLAYw2YT6u7VSCoN3Yk4ZxEIPkNhTAkVaOPA&usqp=CAU");}'

description = "<h1 style='color:white'>PLANT DISEASE DETECTOR</h1><b><h3 style='color:white';>Share a picture of the plant to get immediate results!"

gr.Interface(fn=classify_image, inputs=image, outputs="label", description=description, css=v).launch(debug=False)
```

Chapter4: Result

After the Implementation of the proposed outlook, we got phenomenal results for the model that we have selected for building the Plant Disease Detection System which was VGG16 and a worthy insight was procured through it. Below figures represent the graphical perspective of the model in terms of Accuracy and Loss.

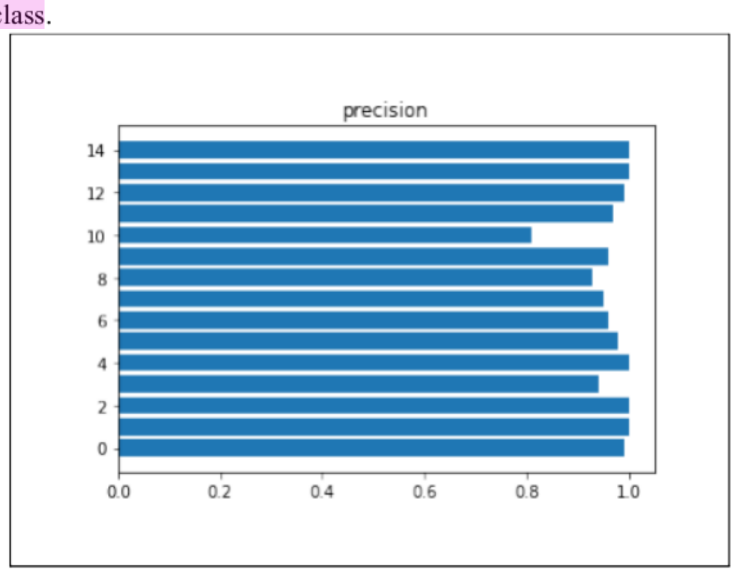


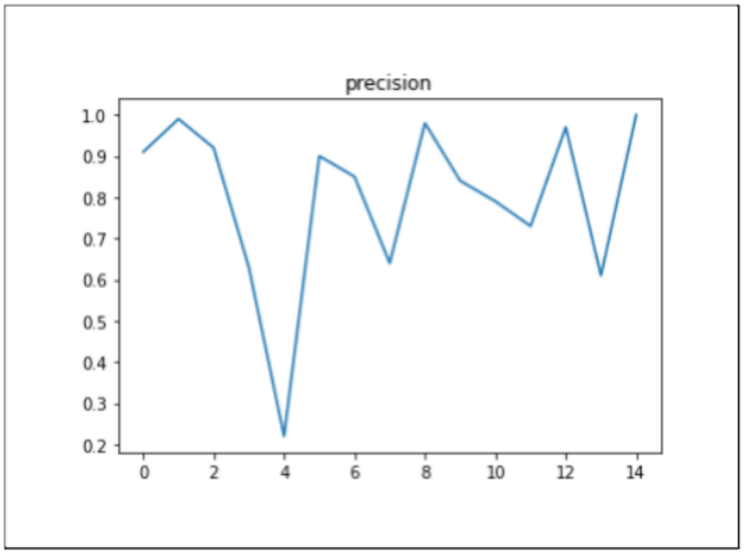
Accuracy Results for VGG16.

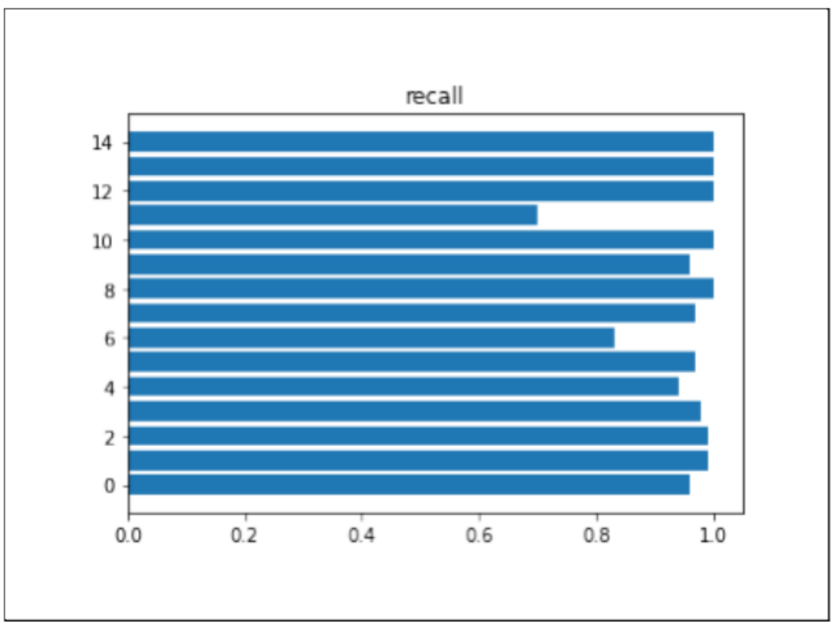


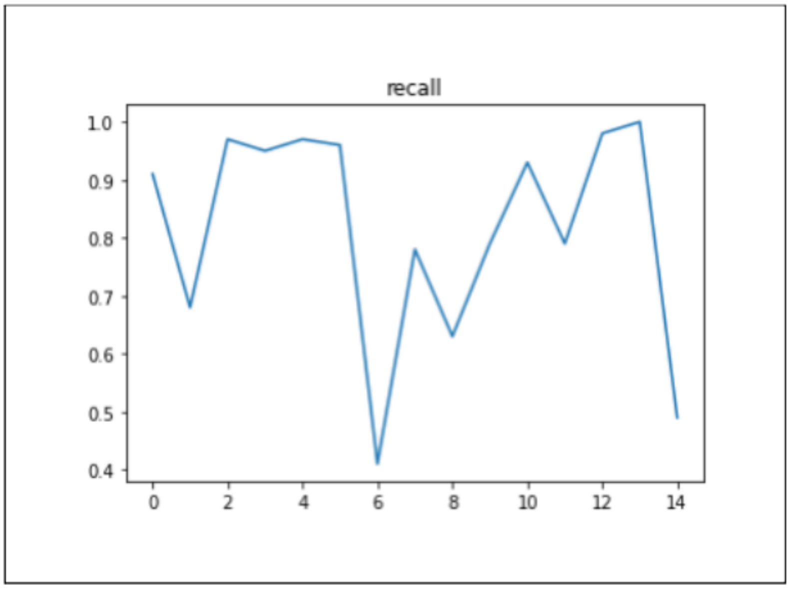
Loss Results for VGG16.

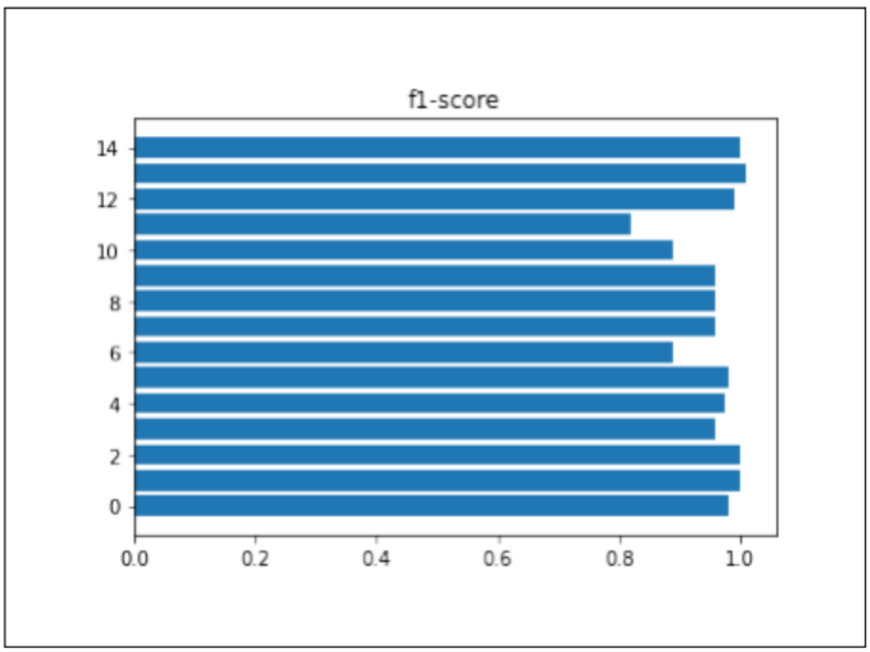
As we have used VGG architecture for training the model and from that we achieved 97% of accuracy, and made the calculation of precision, recall, F1-score for each class. Below figures demonstrate the precision ,recall and F1-score for each class.




precision
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0
2
4
6
8
10
12
14


recall
14
12
10
8
6
4
2
0
0.0
0.2
0.4
0.6
0.8
1.0


recall
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0
2
4
6
8
10
12
14


f1-score
14
12
10
8
6
4
2
0
0.0
0.2
0.4
0.6
0.8
1.0

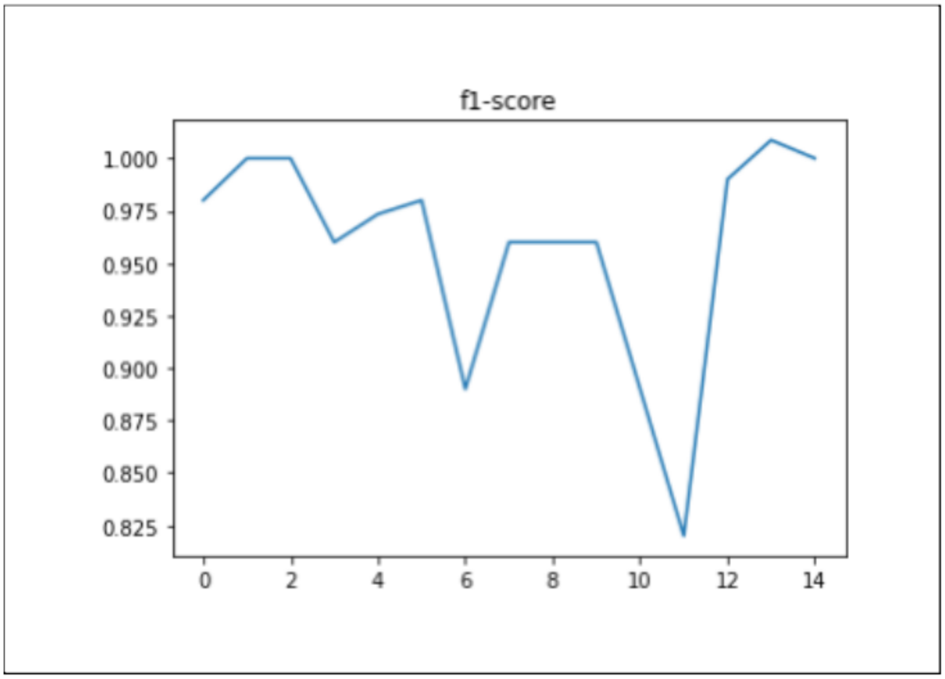


Table-II, illustrates the Accuracies obtained for Training, Validation and Test Data for the VGG16 model.

TABLE II. Analysis of pre-trained architectures based on their accuracies over Plant Leaf dataset.

Model	Training Accuracy (%)	Test Accuracy (%)
VGG16	97.51	97.33

Chapter5: Conclusion

Plant disease detection has become a very big domain to work on, because many of the works have been done earlier using many different technologies and methods in this field. We so far have read some of the papers for our survey and have analyzed and have found many different results for different techniques and methods used. The papers were on the same topic but of different methodologies like few of them were based on MobileNetV2, DenseNet121 InceptionResNetV2 and VGG16 etc. We have analyzed that even after doing so much research in this domain, there is a large scope of research in the future also because every proposed model has some constraints, some disadvantages and some advantages. The need for a perfect model to detect the plant leaf disease is also in hunt and this is the reason that these models are not getting used on a large scale for disease detection. This is a very good and very helpful domain to work on because when farming is done on a very big scale, it is not possible for farmers to look for diseases on every leaf of the plants, so further research and findings on this domain can be very helpful.

So after comparing the results of different models, VGG16 architecture is the best modus operandi as it works better with small data and large data also and after our research we got the accuracy of 99% in some of the present classes, like the category Tomato_Target_spot is the least accuracy scored which is 82% the category Pepper_bell_healthy, Potota_b, etc are the highest accuracy scored which is more than 99%

As a result, it's necessary to scrutinize over varied architectures for finalizing the base model for any modus operandi. The efficacy we got stands out as the best in assuaging this system and more dataset can be amalgamated for greater explication as an outlook.

Chapter6: References

[1] J. G. A. Barbedo, ‘‘Plant disease identification from individual lesions and spots using deep learning,’’ Biosyst. Eng., Vol. 180, PP 96–107, Apr, 2019

[2] Liu, Y. Zhang, D.-J. He, and Y.-X. Li, “Identification of apple leaf diseases based on deep convolutional neural networks,’’ Symmetry, Vol. 10, No. 1, pp 11–19. Dec, 2018.

[3] H. Larochelle, ‘‘Few-shot learning,’’ in Computer Vision: A Reference Guide, 2nd ed., K. Ikeuchi, Ed. Tokyo, and Tokyo: Univ. Tokyo Press, 2014.

[4] Shima Naresh, Ramchandra Habbar et al. “Plant Disease Detection Using Machine Learning”, IEEE, 13 Aug 2018.

[5] Parul Sharma, YPL Singh et al “KrishiMitr (Farmer’s Friend): Using Machine Learning to Identify Diseases in Plants”, IEEE, 7 Jan 2019.

[6] Shital Bankar, Ajita Dubey et al. “Plant Disease Detection Techniques Using Canny Edge Detection & Color Histogram in Image Processing”, International Journal of Computer Science and Information Technologies, Vol. 5 (2) , 2014, 1165-1168.

[7] ZN Reza et al. “Detecting jute plant disease using image processing and machine learning”, IEEE, 9 March 2017.

[8] MS Arya, K Anjali et al. “Detection of unhealthy plant leaves using image processing and genetic algorithm with Arduino”, IEEE, 14 June 2018.

[9] R T Narmadha et al. “Detection and measurement of paddy leaf disease symptoms using image processing”, IEEE. 23 Nov 2017.

[10] M Poobalasubramnium et al. “Plant Disease Identification and Detection Using Support Vector Machines and Artificial Neural Networks”, Artificial Intelligence and Evolutionary Computations in Engineering Systems (pp.15-27). Jan 2020.

[11] Tejas Tawde, Kunal Deshmukh et al. “Rice Plant Disease Detection and Classification Techniques, A Survey”, IJERT, volume 10, July 2021.

[12] Vittal S Gutte et al. “A Survey on Recognition of Plant Disease with Help of Algorithm” IJESC, Volume 6 Issue No. 6, Juy 2016.

[13] Murk Chohan, Adil Khan et al. “Plant Disease Detection using Deep Learning”, IJRTE. ISSN: 2277-3878, Volume-9 Issue-1, May 2020.

[14] RU Khan, A Khan et al. “Image-Based Detection of Plant Diseases: From Classical Machine Learning to Deep Learning Journey”, Hindwani, Volume 2021 |Article ID 5541859. June 2021.

[15] Yang Juo, Zing Jhang et al. “Plant Disease Identification Based on Deep Learning Algorithm in Smart Farming”, Hindwani, Volume 2020 |Article ID 2479172. Aug 2020.

[16] S. A. Dwivedi and A. Attry, "Juxtaposing Deep Learning Models Efficacy for Ocular Disorder Detection of Diabetic Retinopathy for Ophthalmoscopy," 2021 6th International Conference on Signal Processing, Computing and Control (ISPCC), 2021, pp. 352-357, doi: 10.1109/ISPCC53510.2021.9609368.

Final report

ORIGINALITY REPORT

14%	13%	10%	7%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Sharda University Student Paper	4%
2	Lili Li, Shujuan Zhang, Bin Wang. "Plant Disease Detection and Classification by Deep Learning—A Review", IEEE Access, 2021 Publication	2%
3	www.jetir.org Internet Source	2%
4	Submitted to Jaypee University of Information Technology Student Paper	2%
5	www.mdpi.com Internet Source	1%
6	downloads.hindawi.com Internet Source	1%
7	www.ijres.org Internet Source	1%
8	www.siftdesk.org Internet Source	1%

9	"Second International Conference on Image Processing and Capsule Networks", Springer Science and Business Media LLC, 2022 Publication	1 %
10	Subhash Arun Dwivedi, Yves Attry Kalin, Ganpati Kumar Roy, Kanika Singla. "Real-Time Detection for Normal, Glaucomatous and Diabetic Retinopathy Eyes for Ophthalmoscopy using Deep Learning", 2022 9th International Conference on Computing for Sustainable Global Development (INDIACom), 2022 Publication	1 %
11	www.ijcsit.com Internet Source	1 %
12	Shima Ramesh, Ramachandra Hebbar, Niveditha M., Pooja R., Prasad Bhat N., Shashank N., Vinod P.V.. "Plant Disease Detection Using Machine Learning", 2018 International Conference on Design Innovations for 3Cs Compute Communicate Control (ICDI3C), 2018 Publication	1 %
13	www.slideshare.net Internet Source	1 %
14	Parul Sharma, Yash Paul Singh Berwal, Wiqas Ghai. "KrishiMitr (Farmer's Friend): Using	1 %

Machine Learning to Identify Diseases in Plants", 2018 IEEE International Conference on Internet of Things and Intelligence System (IOTAIS), 2018
Publication

15	www.ijert.org Internet Source	<1 %
16	www.hindawi.com Internet Source	<1 %
17	www.ijrte.org Internet Source	<1 %
18	www.ijariit.com Internet Source	<1 %
19	"Communication Software and Networks", Springer Science and Business Media LLC, 2021 Publication	<1 %
20	sersc.org Internet Source	<1 %
21	"Communication, Networks and Computing", Springer Science and Business Media LLC, 2021 Publication	<1 %
22	Singh, Vijai, Varsha, and A K Misra. "Detection of unhealthy region of plant leaves using image processing and genetic algorithm",	<1 %

2015 International Conference on Advances in Computer Engineering and Applications, 2015.
Publication

23	www.irjmets.com Internet Source	<1 %
24	"Ubiquitous Intelligent Systems", Springer Science and Business Media LLC, 2022 Publication	<1 %
25	Rehan Ullah Khan, Khalil Khan, Waleed Albattah, Ali Mustafa Qamar. "Image-Based Detection of Plant Diseases: From Classical Machine Learning to Deep Learning Journey", Wireless Communications and Mobile Computing, 2021 Publication	<1 %
26	"Innovative Data Communication Technologies and Application", Springer Science and Business Media LLC, 2021 Publication	<1 %
27	"Machine Learning and Information Processing", Springer Science and Business Media LLC, 2020 Publication	<1 %
28	Lawrence C. Ngugi, Moataz Abelwahab, Mohammed Abo-Zahhad. "Recent Advances in Image Processing Techniques for Automated Leaf Pest and Disease Recognition	<1 %

- A Review", Information Processing in Agriculture, 2020
Publication

---

29 WWW.MDPI.COM
Internet Source
<1%

---

30 ethesis.nitrkl.ac.in
Internet Source
<1%

---

31 "Sixth International Conference on Intelligent Computing and Applications", Springer Science and Business Media LLC, 2021
Publication
<1%

---

32 www.ipl.org
Internet Source
<1%

---

33 www.irjet.net
Internet Source
<1%

---

34 Aamina Nooraiyeen. "Robotic Vehicle for Automated Detection of Leaf Diseases", 2020 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT), 2020
Publication
<1%

---

35 Ahmad Ayad, Melvin Renner, Anke Schmeink. "Improving the Communication and Computation Efficiency of Split Learning for IoT Applications", 2021 IEEE Global Communications Conference (GLOBECOM), 2021
<1%

Publication

36	Ankit Chhillar, Sanjeev Thakur, Ajay Rana. "Survey of Plant Disease Detection Using Image Classification Techniques", 2020 8th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO), 2020 Publication	<1 %
37	Mourad Moussa, Hazar El Ouni, Ali Douik. "Edge Detection Based on Fuzzy Logic and Hybrid Types of Shannon Entropy", Journal of Circuits, Systems and Computers, 2020 Publication	<1 %
38	www.ijitee.org Internet Source	<1 %
39	www.int-arch-photogramm-remote-sens-spatial-inf-sci.net Internet Source	<1 %
40	www.xajzkjdx.cn Internet Source	<1 %
41	"Krishimitr: A Cloud Computing and Platform for Disease Detection in Agriculture", International Journal of Innovative Technology and Exploring Engineering, 2019 Publication	<1 %
42	Mohsen Niazian, Gniewko Niedbała. "Machine Learning for Plant Breeding and	<1 %

Biotechnology", Agriculture, 2020
Publication

43	Nidhi Prashar, A. L. Sangal. "Chapter 54 Plant Disease Detection Using Deep Learning (Convolutional Neural Networks)", Springer Science and Business Media LLC, 2022 Publication	<1 %
44	Sk Mahmudul Hassan, Arnab Kumar Maji, Michał Jasiński, Zbigniew Leonowicz, Elżbieta Jasińska. "Identification of Plant-Leaf Diseases Using CNN and Transfer-Learning Approach", Electronics, 2021 Publication	<1 %

Exclude quotes	On	Exclude matches	< 6 words
Exclude bibliography	On		