

**DESIGN AND DEVELOPMENT OF CNN BASED DISEASE  
DETECTION MODEL FOR MAIZE CROP USING  
HYPERSPETRAL IMAGES**

Thesis Submitted for the Award of the Degree of

**DOCTOR OF PHILOSOPHY**

in

**Computer Science and Engineering**

By

**M. Nagaraju**

**Registration No. 41800497**

Supervised By

**Dr. Priyanka Chawla**

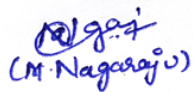
**Professor**



**LOVELY PROFESSIONAL UNIVERSITY  
PUNJAB  
2022**

## DECLARATION

I declare that the thesis entitled “**Design and Development of CNN Based Disease Detection Model for Maize Crop using Hyperspectral Images**” has been prepared by me under the supervision of Dr. Priyanka Chawla, Professor, School of Computer Science and Engineering, Lovely Professional University, India. No part of this thesis has formed the basis for the award of any degree or fellowship previously.



(M. Nagaraju)

**M. Nagaraju**

Reg.No 41800497

School of Computer Science and Engineering

Lovely Professional University, Punjab

Phagwara – 144002.

**Place: Phagwara**

**Date: 23-05-2022**

## **CERTIFICATE**

This is to certify that the thesis entitled “**Design and Development of CNN Based Disease Detection Model for Maize Crop using Hyperspectral Images**” submitted by **M. Nagaraju** to the Lovely Professional University, Punjab for the award of the degree of Doctor of Philosophy is a bonafide record of research work carried out by him under my supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.



**Dr. Priyanka Chawla**

Professor

School of Computer Science and Engineering

Lovely Professional University, Punjab

Phagwara – 144002.

**Place: Phagwara**

**Date: 23-05-2022**

## ABSTRACT

The primary thing in agriculture is the timely recognition of crop diseases. As the verification process is conducted manually it is becoming very difficult to recognize the disease categories. With an increase in cultivation, the early detection of disease symptoms can increase the productivity, address environmental and economic concerns. The most crucial thing in recognizing and classifying diseases efficiently depends on acquiring input datasets. The disease identification has become very critical due to lack of infrastructure. Automatic recognition of crop diseases using hyperspectral images (HSI) is a complex and a challenging task for sustainable cultivation. Recent developments in deep learning have shown a remarkable improvement in the early recognition and classification of crop diseases. With computer vision techniques, segmenting the disease areas on the images can make the classification process simple. Deep learning approach enables machines to adopt human behavior by learning image features from the input images. Convolutional Neural Networks is a comeback in applying deep learning that can effectively perform image processing, recognition, and classification as well. Since few decades, many researchers have proposed several algorithms to classify the diseases in crops with advancements in computer vision.

In recent times, crop diseases classification has been performed using various newly developed architectures on the publicly accessible datasets. With transfer learning approach, many researchers developed new CNN models by modifying the pre-trained one. The models have performed effectively while classifying the images collected from Plant Village and Kaggle web resources. But the same models have failed when applied for images captured in the real-time agricultural field.

In this research a convolutional neural networks-based approach has been followed to address seven different types of disease identification in maize crop. A systematic method has been followed to address the classification problem. First, a review of existing deep learning techniques, data acquisition, image processing techniques, image

augmentation techniques, neural network models and their performance have been conducted. Second, image acquisition and enrichment process to prepare training and testing image sets has been carried out. Third, a new CNN architecture named NPNet-19 is proposed, designed and developed from the scratch. The model is trained using 80% of images gathered from internet repositories. The model is tested with 20% of image acquired from real agricultural field.

During experiments, a total of 15,960 images have been considered where 13,300 images for training and 2660 images were used for classification. The images belong to seven unique disease classes including one healthy leaf. The model will be feed with the images after performing some preprocessing all the images like color, resizing to 224 width and 224 heights, rescaling to 1/.255 resolution, and JPG image format. Forth, the performance of NPNet-19 model has observed with basic hyperparameters like image set split ratio, image size, kernel size, number of epochs, optimizer, learning rate, loss function, batch size with standard values. Fifth, several experiments have conducted to evaluate and improve the efficiency of NPNet-19 model by adjusting several hyperparameters like image set split ratio, image size, kernel size, number of epochs, optimizer, learning rate, loss function, batch size with different values. Sixth, a comparative analysis has conducted to assess the model performance with four pre-trained models like CNN-SVM, DenseNet – 121, Shallow Net-8, and Inception V2. Seventh, the NPNet-19 model performance has compared with six existing transfer learning models like CNN-ST, Modified LeNet, Adoptive CNN, SoyNet, 9-layer and 13-layered architectures. All the experiments were conducted using own datasets. Eighth, the efficiency has evaluated by training and testing with only the images collected from plant village repository.

A total of fifty-five experiments have been conducted to validate the performance of NPNet-19. The experimental outcomes showed an accuracy of 97.5% on training image set and 88.72% of accuracy on testing image set when the model is designed with standard hyperparameters. During the first evaluation process several hyperparameters like image type as HSI, image size as 224 x 224, kernel size as 3 x 3, number of filters starting with 16, batch size as 32, classification type, optimizer as Adam, learning rate

of 0.001, activation functions as ReLU and softmax, number of layers as 19, and dropout percentage of 30% has been considered. The experimental approach explored the selection of several hyperparameters with different values. It is observed that the results show that the model trained and classified for 200 epochs improved the classification accuracy by 87.44%. The results show that the 168 x 168 and 224 x 224 input image sizes resulted an improved classification accuracy of 84.66% and 85.23%, respectively. The model has achieved 85.83% classification accuracy when an Adam optimizer (learning rate of 0.001) is used. But the other optimizers RMSprop and SGD has achieved only 81.95% and 79.66% results.

Several experiments have been performed to empirically compare the classification accuracy of the proposed model with CNN-SVM, DenseNet – 121, Shallow Net-8, and Inception V2 pre-trained models. The NPNet-19 proposed model has showed a 1.1%, 10.57%, 2.15%, and 1.74% improvement respectively. When compared to existing models like CNN-ST, Modified LeNet, Adoptive CNN, SoyNet, 9-layer and 13-layered architectures the proposed model has shown an improvement of 14.52%, 10.12%, 3.88%, 8.17%, 7.25%, and 3.39% in classification accuracy respectively. Finally, when the NPNet-19 is tested with images collected from the publicly available dataset (plant village) has obtained a very good classification accuracy of 97%. The results obtained are comparatively better than the pretrained and transfer learning CNN models and proved the novelty of the research. Therefore, it is even feasible that the model proposed in this work can also be applied to other crop diseases for transparent identification and classification purpose.

The significance of the present work is to deal with the deep learning technologies by implementing on plant diseases. Developing more sophisticated model and analyse the classification of maize crop diseases based on hyperspectral images is the main motto. The literature reviewed in this work provides the future research to explore and learn more deep learning capabilities while classifying the diseases in crops accurately. The present research can provide better experience and knowledge in designing a new CNN model to address the disease classification problem more efficiently. The selection of appropriate hyperparameters will also become easy and assist when dealing with multi-

class image classification problem. The results clearly shown that the proposed NPNet-19 model is best suitable to the maize crop disease classification and proved to be the best one even classifying the real-time images efficiently.

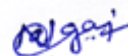
## **PREFACE / ACKNOWLEDGEMENT**

This thesis would not have been possible without the support of glorious people who motivated me during my doctoral study. I am thankful from my bottom of my heart toward the numerous persons who assisted me while conducting this study. First of all, I would like to thank my supervisor, Dr. Priyanka Chawla, for her worthy guidance, support, and suggestions, in every step of this research project during my Ph.D. journey. Dr. Priyanka Chawla has an optimistic personality with helpful nature, she has always made herself ready to clarify my doubts and it was a great opportunity to work under her supervision. She always shed light whenever I was feeling stuck in my path of research ambitions.

It is my privilege to thank my family members for their continuous encouragement throughout my research time. I am extremely thankful to the farmers of Telangana state for supporting and providing the necessary inputs.

I would like to express my gratitude toward the entire Lovely Professional University family for providing suitable infrastructure and environment for completing my research work in a time-bound manner. Also, I like to thank the Division of Research & Development and School of Computer Science and Engineering for their help and encouragement in my entire Ph.D. journey. Finally, I like to thank almighty God who helped me to achieve such a big milestone.

Date: 23-05-2022



M. Nagaraju



## TABLE OF CONTENTS

<b>Chapter No</b>	<b>Topic Name</b>	<b>Page No</b>
	Declaration	i
	Certificate	ii
	Abstract	iii
	Preface/Acknowledgement	vii
	Table of Contents	viii
	List of Tables	x
	List of Figures	xii
	Abbreviations	xiv
<b>1</b>	<b>INTRODUCTION</b>	<b>1</b>
1.1	Research Gaps	5
1.2	Research Objectives	7
1.3	Research Methodology	8
1.4	Organization of the thesis	11
<b>2</b>	<b>LITERATURE REVIEW</b>	<b>13</b>
2.1	Conduction	13
2.2	Literature	14
<b>3</b>	<b>DATA ACQUISITION AND ENRICHMENT</b>	<b>24</b>
3.1	Image Acquisition	25
3.2	Image Augmentations	28
3.3	Image Preprocessing Techniques	33
3.4	RGB to Hyperspectral Image Conversion	33
<b>4</b>	<b>DESIGN AND DEVELOPMENT OF A CNN MODEL</b>	<b>34</b>
4.1	Development of a CNN Model	34
4.2	Proposed CNN Model Design	41
<b>5</b>	<b>VERIFICATION AND VALIDATION OF THE PROPOSED MODEL</b>	<b>44</b>
5.1	Work Environment	44

5.2	Verification of the model	44
5.3	Classification Performance	48
5.4	Performance Analysis using Selective Hyperparameters	52
5.5	Comparative Analysis	76
<b>6</b>	<b>CONCLUSION</b>	<b>88</b>
6.1	Findings	90
6.2	Observations	92
6.3	Discussions	92
<b>7</b>	<b>FUTURE SCOPE</b>	<b>94</b>
	<b>REFERENCES</b>	<b>95</b>
	<b>LIST OF RESEARCH ARTICLES PUBLISHED</b>	<b>103</b>

## LIST OF TABLES

<b>S.No</b>	<b>Table No</b>	<b>Description</b>	<b>Page No</b>
1	2.1	Sources selected for literature	13
2	2.2	List of Search Keywords	13
3	2.3	Significant research contributions	21
4	3.1	Details of maize crop images	32
5	4.1	Proposed NPNet-19 model summary	42
6	5.1	Accuracy results of six different epochs	45
7	5.2	Confusion matrix of seven diseases classification	48
8	5.3	Classification results of different image set split ratios	49
9	5.4	Class-wise classification accuracies	50
10	5.5	Classification accuracies with three different kernel sizes	51
11	5.6	List of hyperparameters for tuning the model	53
12	5.7	Accuracies with different image sizes	53
13	5.8	Accuracies with different batch sizes	56
14	5.9	Accuracies with different number of epochs	59
15	5.10	Accuracies with different optimizers	61
16	5.11	Accuracies with different learning rates	64
17	5.12	Accuracies with different kernel sizes	67
18	5.13	Accuracies with different number of hidden layers	70
19	5.14	Confusion metric values with different image sizes	73
20	5.15	Confusion metric values with different image sizes	73
21	5.16	Confusion metric values with different number of epochs	73
22	5.17	Confusion metric values with different optimizers	74
23	5.18	Confusion metric values with different learning rates	74
24	5.19	Confusion metric values with different kernel sizes	75
25	5.20	Confusion metric values with different number of hidden layers	75
26	5.21	Performance results of pre-trained model and proposed model	77

27	5.22	Class-wise classification accuracies for pre-trained and proposed models	78
28	5.23	Performance evaluation of existing models and the proposed model	81
29	5.24	Confusion matrix for existing and proposed models	82
30	5.25	Confusion Matrix of four-class classification problem	86

## LIST OF FIGURES

<b>S. No</b>	<b>Figure No</b>	<b>Description</b>	<b>Page No</b>
1	1.1	Export report of maize crop around the world	1
2	1.2	Process showing the research workflow	2
3	1.3	Research Methodology of the proposed work	8
4	2.1	Step-by-Step review process of architectures	15
5	3.1	Four-Step process of data collection	24
6	3.2	Detailed methodology of Image acquisition process	25
7	3.3	Position Augmentations: (a) HV (b) HVS	30
8	3.4	Position Augmentation: (a) HVF (b) RR	30
9	3.5	Position and Color Augmentation: (a) RZ (b) RB	31
10	3.6	Data collection process	32
11	3.7	Seven Classes of maize crop disease categories (a) anthracnose leaf blight, (b) anthracnose stalk rot, (c) eyespot, (d) northern corn leaf spot, (e) southern rust, (f) gibberella stalk rot (g) healthy	32
12	4.1	Feature map extraction from an infected leaf image	36
13	4.2	First CNN layer development process	36
14	4.3	Sample images generated after adjustment with filters	36
15	4.4	Applying a rectifier function to remove non-linear image features	38
16	4.5	Activation function SoftMax	38
17	4.6	Different versions of image representations (a) original (b) rotated and (c) squashed	40
18	4.7	Proposed fully connected CNN Model	41
19	4.8	Proposed NPNet-19 CNN architectures	43
20	5.1	Performance Evaluation of the proposed model	44
21	5.2	Accuracy and loss curves of the training and testing image sets	46
22	5.3	Training and Testing Accuracy and Loss Curves	47
23	5.4	Performance improvement of the proposed model	47
24	5.5	Confusion matrix representations	49
25	5.6	Classification accuracies with different image sets split ratios	50

26	5.7	Performances measures with different kernel sizes	52
27	5.8	Accuracy and loss curves with different image sizes	54
28	5.9	Confusion matrices with different image sizes	55
29	5.10	Comparison of model performance with different image sizes	55
30	5.11	Accuracy and loss curves with different batch sizes	57
31	5.12	Confusion matrices with different batch sizes	58
32	5.13	Comparison of model performance with different batch sizes	58
33	5.14	Accuracy and loss curves with different number of epochs	59
34	5.15	Confusion matrices with different number of epochs	60
35	5.16	Comparison of model performance with number of epochs	61
36	5.17	Accuracy and loss curves with different optimizers	62
37	5.18	Confusion matrices with different optimizers	63
38	5.19	Comparison of model performance with optimizers	63
39	5.20	Accuracy and loss curves with different learning rates	65
40	5.21	Confusion matrices with different learning rates	66
41	5.22	Comparison of model performance with learning rate	66
42	5.23	Accuracy and loss curves with different kernel size	68
43	5.24	Confusion matrices with different kernel sizes	69
44	5.25	Comparison of model performance with different kernel sizes	69
45	5.26	Accuracy and loss curves with different number of hidden layers	71
46	5.27	Confusion matrices for different hidden layers	72
47	5.28	Comparison of model performance with different number of hidden layers	72
48	5.29	Performance evaluation process using Pre-Trained Models	76
49	5.30	Accuracy and Loss curves of pre-trained models	78
50	5.31	Performance measures of pre-training and proposed models	81
51	5.32	Performance measures of existing and proposed models	82
52	5.33	Training and Testing curves with plant village image sets	87
53	5.34	Confusion Matrix for without and with normalization values	87

## ABBREVIATIONS

AF	-	Activation Function
AI	-	Artificial Intelligence
ALB	-	Anthracnose Leaf Blight
ASR	-	Anthracnose Stalk Rot
BD	-	Bernoulli Distribution
BN	-	Batch Normalization
CM	-	Confusion Matrix
CNN	-	Convolutional Neural Network
CV	-	Computer Vision
D	-	Dropout
DCNN	-	Deep Convolutional Neural Networks
ES	-	Eyespot
FaaS	-	Farm-as-a-Service
FC	-	Fully Connected Layer
GSR	-	Gibberella Stalk Rot
H	-	Healthy
HD	-	Hyperspectral Data
HF	-	Horizontal Flips
HSI	-	Hyperspectral Imaging
HSV	-	Hue, Saturation, and Value
HV	-	Horizontal-Vertical
HVF	-	Horizontal-Vertical Flip
HVS	-	Horizontal-Vertical Shift
ID	-	Insufficient Data
ML	-	Machine Learning
MP	-	Max Pooling
MSVM	-	Multiclass Support Vector Machine

NA	-	Not Applied
NLB	-	Northern Leaf Blight
NM	-	Not Mentioned
NN	-	Neural Network
RB	-	Random Brightness
RF	-	Random Flips
ReLU	-	Rectifier Linear Unit
RGB	-	Red Green Blue
RR	-	Random Rotation
RZ	-	Random Zooming
SGD	-	Stochastic Gradient Descent
SR	-	Southern Rust
SVM	-	Support Vector Machine



## CHAPTER – 1

### INTRODUCTION

Maize is one of the most important crops exported from India, with a substantial increase in cultivation area coverage and production rate growth. The country contributes about 2% to 3% of total crop production worldwide. 80 percent of India's corn crop is exported, with 33 percent going to Indonesia, 27 percent to Malaysia, and 20 percent to Vietnam as shown in Fig 1.1. However, according to a statistical report [9] released by the Indian government, corn crop production was just 18.73 million tons in 2017-18, down 0.52 million tons from 2016-17. Improvement in the crop yield can help the country to improve in social and economic areas for the farmers and bring a rise in levels of foreign reserves.

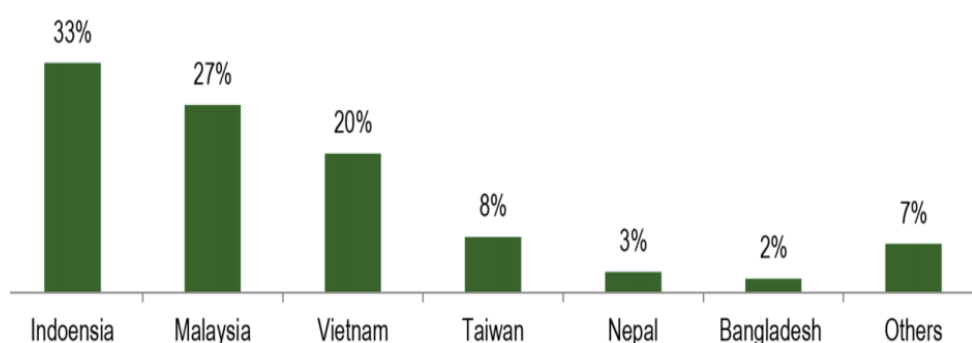


Fig 1.1. Export report of maize crop around the world

The crop loss is subject to critical weed diseases, insects, and infestations, which reduce the overall yield significantly. Because of biotic stresses, it is estimated that around 25% of maize yield was reduced during the last few years. The disease incidence is so difficult to predict and not possible or too late in some situations, due to lack of awareness. The early identification of disease symptoms for a crop is the main achievement to meet environmental and economic challenges. Increasing crop yield relies on reducing the yield loss that can further ensure food security for millions of people around the world. Consequently, the research must focus on several technical and non-technical aspects by targeting the issues that can be diagnosed, disease symptoms can be treated along with the prevention of epidemics even. The detection of

maize crop diseases is become difficult and requires a deep learning approach, and specific knowledge as identifying the symptoms was not developed well. This situation has focused on the importance of healthy and diseased images processing related to disease identification, diagnosis them in the early stages of development in such a way that the farmer can react in time. Simultaneously, the reliability of crop disease identification must be validated in real-time field environments. The complete process of validation and the workflow is shown in Fig 1.2.

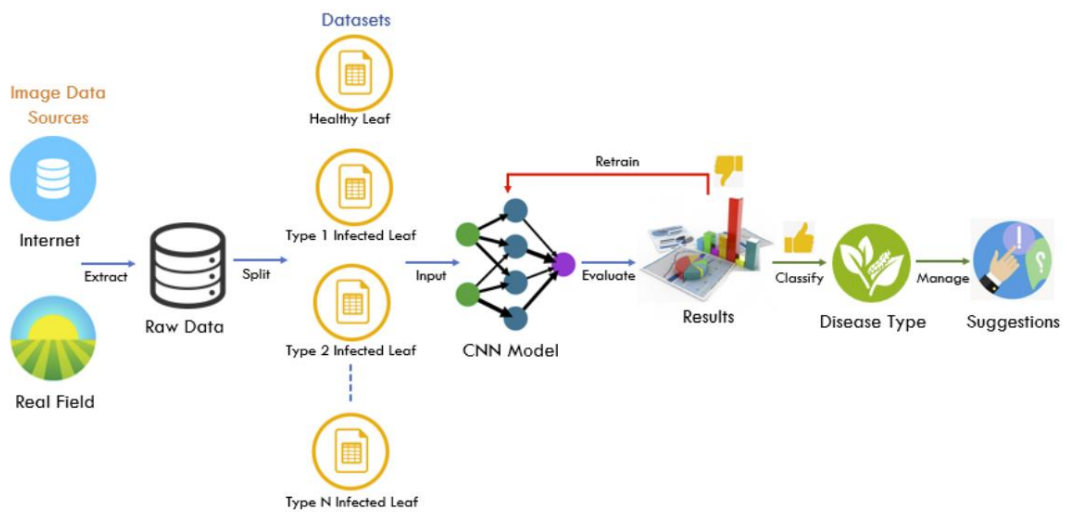


Fig 1.2. Process showing the research workflow

Since 1950's, computer experts in the field of Artificial Intelligence (AI) are trying to build machines that can able to understand and process the visual data. In the ensuing period this process is named as Computer Vision (CV). The primary functionality of computer vision is to define the process of image classification into various predefined classes. The field of computer vision is changing its shape to solve the most challenging problems. The concept of Deep Learning (DL) on the other side of CV and machine learning gaining numerous attentions in object recognition and AI in recent years.

The Machine Learning (ML) algorithms are part of Artificial Intelligence (AI) that provides computer intelligence by understanding and learning the underlying links between data and decision making without any specialized programming. Various ML algorithms like speech and vision were developed since the late 1990's but have generally failed to achieve greater performance.

A special category of Neural Network (NN) known as the Convolutionary Neural Network or simply CNN [1,2] is created because of the difficult nature CV activities. It can be the best ways of learning the recognition of images. CNN's success outside the academy attracted attention of researchers and industry around the globe. Enterprises like Facebook, Microsoft, Google, AT&T, and NEC have formed research consortiums to test the new CNN architectures. Most of the image processing competitors today are using deep CNN models only. CNNs have demonstrated the state-of-the-art achievements in image recognition, segmentation, detection and recovery activities [2].

In 1989, CNN architecture is highlighted first by LeCun's efforts to solve grid-based topological information [1] [3]. Later, the success of CNN models has been contributed in large part to its ability to reconstruct the hierarchy. The CNN's hierarchical structure emulates the layered and deep learning process in the human brain of the Neocortex that automatically extracts characteristics based on the underlying data. CNN learns during training by monitoring the input weight change through the propagation algorithm. The use of back propagation algorithms minimizes the cost function of a CNN model. CNN can extract low, medium and high features and a mixture of features at lower-level, mid-level, high-level and more abstract level also.

Deep architectures have more advantage when compared with shallow architectures in solving the problems in complex learning. Multiple layers stacking of several linear and non-linear unit helps to learn more complex representations at different abstraction levels across deep networks. Technology advances and therefore high computing resources available are however also considered as one of the primary reasons for the recent advancement of the deep CNNs. The architectures of deep CNN have resulted substantial performance improvement over models that are based on low and traditional vision.

The application in supervised analysis, deep CNNs can learn valuable representations from huge and unlabelled data. The mapping functions of CNN allows the extraction of invariant representations to be enhanced and therefore hundreds of category recognition functions can be performed. Recently, it was shown that different levels of usability, both high and low, can be converted into a 4-generic task to recognize with the theory of transfer learning [4][5]. The key characteristics of CNN include

hierarchical training, automated feature extraction, multiple classes and weight sharing [6–8, 10].

To overcome the difficulties, existing procedures must be better understood by monitoring, measuring, and analysing large amounts of agricultural data. It is also vital to understand both short-term crop management and large-scale ecosystems [11][12]. Big data analysis is one of the advanced deep learning approaches. In general, a deep learning model has at least three layers, with each layer including neurons that are linked to data features, producing more complicated information. Deep learning models learn input properties using a hierarchy of organised networks of neurons [13]. Recent research has focused on the evaluation of deep learning models based on hyperspectral imaging and data analysis to detect the diseases in plant [14-16].

The objective of this research is to propose an algorithm and generalize the presence of diseases like Anthracnose Leaf Blight (ALB), Anthracnose Stalk Rot (ASR), Eyespot (ES), Gibberella Stalk Rot (GSR), Healthy (H), Northern Leaf Blight (NLB), and Southern Rust (SR) by applying the deep learning techniques. The research proposed to develop and utilize a Deep CNN to perform disease-wise classification without the effect of other disease symptoms presented on the maize leaf.

## 1.1 Research Gaps Identified

- Adopting modern computer vision technology is insufficient for automatic disease detection.
- Environmental factors can have an impact on disease classification analysis while collecting input data.
- Disease symptoms are not well defined, making it difficult to segment diseased and healthy portions.
- Because of visual similarities in disease signs, existing approaches must rely on variances to differentiate.
- It was ruled out for assessing disease severity and management.
- The relevance of finding hyperspectral data from various sources of the internet is often not reliable.
- Image capturing is finding difficult rather than acquiring in laboratory conditions.
- The augmentation process to increase the image datasets and reduce overfitting during the training stage is performing very rarely.
- Small datasets cannot train a model to predict with a minimal error rate and cannot guarantee the model perfection.
- There is a challenging problem to process a huge volume of hyperspectral data represented in high dimensional imageries.
- In existing neural networks, a complexity was observed due to its typical structure while making decisions.
- Environmental conditions like light, angle, and practicality scale can also impact on analysing the disease classification.
- Fine-tuned procedure for model modification and optimization requires a higher level of learning.
- The computational workload in training a model is very high while using CPU's.

The chapter has reviewed the literature related to deep convolutional neural networks to identify the current state of research. The obtained literature has extended over various domains health and agriculture areas. The study of literature brought a need to review certain representative factors for further analysis. Since the factors identified were considered enough to represent the current scenario of CNN performance and a need for modelling the effective network system. The existing systems need to be improved to consider the effect of combined constituent subsystems while integrating them to apply towards disease prediction quantification.

## **1.2 Objectives of the proposed work**

1. To acquire images of healthy and diseased maize crop leaves for enriching trainable and testing datasets using different fine-tuning and augmentation techniques.
2. To design and develop CNN model for disease detection (such as anthracnose leaf blight, anthracnose stalk rot, eyespot, gibberella stalk rot etc.) based on leaf features of a maize crop.
3. Verification and validation of the proposed model.

### 1.3 Research Methodology

In this section, a complete step-by-step process of facilitating and implementing the CNN to model a network of convolutional layers for maize crop disease identification, classification and management and depicted in Fig 1.3.

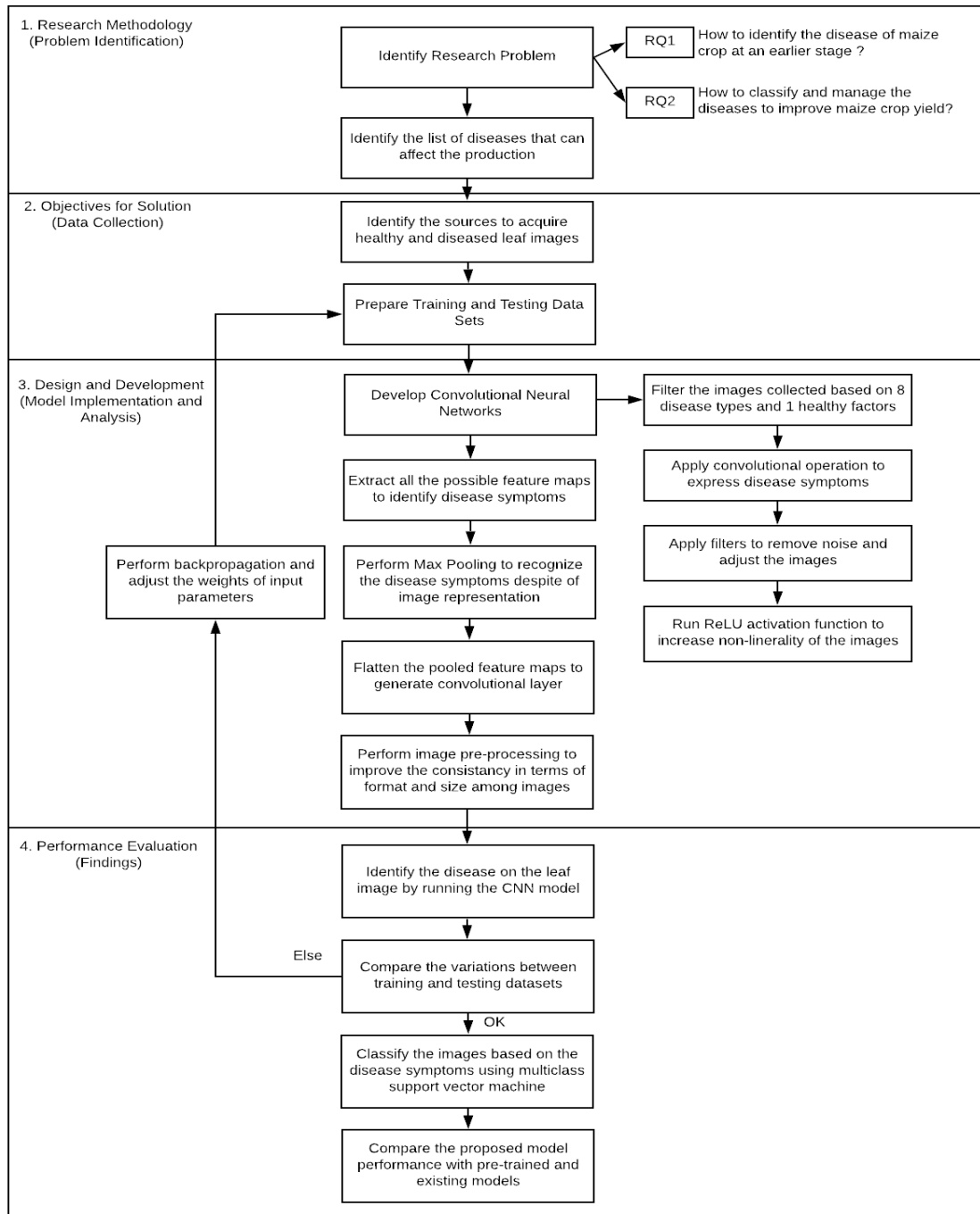


Fig 1.3. Research Methodology of the proposed work



A substantial image database must be created to develop an algorithm that can differentiate the disease at earlier stages for maize crop. A systematic approach is followed to acquire image datasets for all stages of disease recognition and classification.

### **1.3.1 Image pre-processing, augmentation and labelling**

During this process, the proposed Deep CNN's classifiers are pre-processed to improve the feature extraction of maize leaf image and increase consistency before the model was trained. The other most significant phase is to normalize the image with .jpg format and 224 x 224 pixels size and 32 x 32 dots per every inch. The type of image processing can be done using python code on the Anaconda Python Distribution.

Training any CNN requires substantial data. CNN must prepare with more data to obtain more features. If the number of images acquired (4,758) feels not enough for detection of diseases, then it is necessary to increase the images in the dataset by various methods to distinguish seven different diseases considered in this research. After the initialization of original healthy and diseased leaf images, some additional images were generated by rotating the existing images in various angles, mirroring each rotated image, and splitting the same image into several subparts with the same size etc. The process of dataset expansion can reduce overfitting during training phase [17]. Among the entire maize leaf images dataset, 80% are considered for training and the remaining 20% for testing. Proper labelling for all the images was done with disease acronym to confirm the accuracy of 7 classes in the image dataset by human experts based on the disease symptoms. It will help to differentiate the training dataset with the validation dataset.

### **1.3.2 Evaluating, Improving and Tuning the Deep CNN model**

The results obtained from the evaluation of the model must be strengthened further and be capable of recognizing the presence of seven different diseases and the absence of other conditions of maize crop. If the results vary between the trained dataset and the

tested dataset, then the entire process should be repeated (backpropagation) by changing the priorities of the input parameters or symptoms.

## **1.4 Organization of the thesis**

In chapter 1, the importance of computer vision, machine learning, deep learning and convolutional neural networks to perform the identification and classification of various diseases of maize crop is discussed. Understanding how the convolutional neural networks can contribute to solve the disease classification problem is discussed. The research objectives are also discussed along with the methodology followed during the completion of the present research.

In chapter 2, presents the literature reviewed in the field of agriculture specifically serving the identification of maize crop diseases is presented. The chapter also discussed the outline relevant to convolutional neural networks design and development including the architectures and their implementations.

In chapter 3, the process of image acquisition process is clearly discussed. The image enrichment into training and testing datasets is discussed. Various image pre-processing techniques to standardize the input images are discussed. Several image augmentation techniques are also discussed to explain how the process has increased the quantity of training dataset.

In chapter 4, the process of designing and developing a proposed convolutional neural network is discussed.

In chapter 5, the verification and validation of the proposed model using experimental approach is discussed. The performance evaluation of the proposed model based on the selection of suitable hyperparameters is discussed. The comparative performance analysis of the proposed model has been carried out with the pre-trained and existing model is also discussed.

In chapter 6, the conclusion of the thesis is presented by including the research problem, proposed approaches to address the research aims, outlining the major contributions, and summarizing the key findings. The observations made during the experimental

process is discussed as findings. The outcomes of every experiment conducted during this research is discussed clearly.

In chapter 7, the future scope of the thesis is discussed including some limitations and open challenges.

## CHAPTER – 2

### LITERATURE REVIEW

This section presents literature in the field of agriculture, specifically serving the purpose of identifying maize crop diseases. The analysis of research articles was done keeping in view to represent existing models in a more concise and informative manner. The purpose of this phrase is to outline the present study relevant to the convolutional neural network design. The goal is to enlighten regular principles and practices associated with active model development. The literature review was organized around a basic and consistent framework, which is completely based on individual models proposed.

#### 2.1 Conduction

The present thesis has collected numerous academic papers published between 2015 and 2021. To begin, a keyword-based search has conducted using the sources listed in Table 2.1 and the list of search keywords listed in Table 2.2.

Table 2.1. Sources selected for literature

Sources	Literature Type	Locations Identified
Journal Publications	Review articles Content Specific	Journal Databases Scientific Databases Free Downloadable Papers
Scholarly Books	Comprehensive Research Authors Perspectives Overviews Authoritative Historical	Digital Library Catalogues

Table 2.2. List of Search Keywords

Search Keywords		
Deep Learning Convolutional	Neural Network	Imaging Crop Diseases Disease
Prediction Image Data Sources	Artificial Intelligence	Convolutional Neural Networks
Disease Identification	Hyperspectral Imaging	Model Hyperparameters

Image Identification	Disease Identification	Disease Classification
Machine Learning Techniques	Smart Learning	Pre-trained model
Transfer Learning Techniques	Transfer Learning Models	Maize Crop datasets
Maize disease datasets	Image Pre-Processing	Image Augmentation Techniques
Fine-Tuning Techniques	CNN Models	Hyperspectral Data
Crop Diseases	Computer Vision	CNN Architectures
Artificial Intelligence	Activation Functions	Optimization Techniques

This subsection handle with reviewing, examining and organizing the selection process to evaluate the existing literatures based on plant diseases classification, deep learning techniques, and solutions of CNN model for the framed research questions.

RQ1. What are the internet sources that provides maize crop images?

RQ2. What are the deep learning techniques adopted to classify the different classes of maize crop diseases?

RQ3. What are the CNN architectures adopted by existing literatures to classify plant diseases?

RQ4. What are the pre-trained models contributing to the recognition of plant diseases using hyperspectral images?

RQ5. What are the existing models contributing to the classification of plant diseases using hyperspectral images?

## **2.2 Literature**

### **2.2.1 CNN Architectures**

The need of this literature is to identify several architectures developed in the area of computer vision and some open challenges. A systematic evaluation of academic articles published from 1962 to 2021 in various reputed articles has been conducted. Selected articles were divided into various domain areas combined into deep learning applications that represent a broad scope of available solutions in the area of computer vision. Further, the results can be used to feed machine learning algorithms and identify the situations for decision making. Based on the potential benefits of the architectures

of CNN models aims to understand the latest innovations in the field of computer vision and machine learning technologies. A step-by-step review of literature is conducted to identify the relevant academic and research journal articles for further analysis.

The study conducted a web-based searching strategy through several meta-search engines and digital libraries. The obtained information is analysed to rearrange it into several solutions. Table 2.1 describes the list of various sources from where the information has been collected and the search queries used to extract the relevant information from various books, articles, and other journals using the queries/keywords. To ensure information quality, articles published in reputed journals only were considered for further review process. Initially, 172 papers were identified and filtered by removing the duplicates and conference papers based on the field names listed in Table 2.2 and finally selected 42 papers for review. The process flow followed during review of architectures is shown in Fig 2.1.

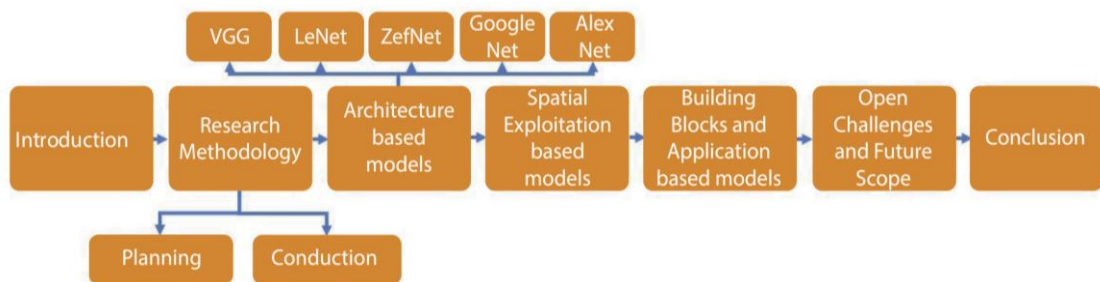


Fig 2.1. Step-by-Step review process of architectures

A network model for recognising and categorising maize crop diseases was entirely constructed using CNN by leveraging the principles of deep learning. The model can distinguish between three conditions: northern corn leaf blight, common rust, and grey leaf spot [3]. Deep learning frameworks were used to create an automated in-field disease diagnosis system for maize crops. The technique has integrated disease detection for training images in wild conditions [19]. To reduce the time consuming for diagnosis of maize crop northern leaf blight disease, a computational CNN was developed. The system can identify the conditions automatically by classifying small

portions of image regions and compare them with individual heat maps [20]. Improving the accuracy in identifying and reducing the parameters count of a maize crop disease is the motto. From sources like Plant Village and Google websites, nearly 500 maize crop images were collected and divided into nine categories with eight diseased classes and one healthy class. Two deep learning models like Google Net and CIFAR 10 were implemented to improve the precision and recognize leaf diseases in maize crop. The models are trained and tested with different types of maize images. Finally, a comparative study was made concerning the results obtained and analysed their efficiency [21]. From a publicly available repository over three datasets with 18,222 maize crop images were collected on Open-Source Framework [22].

By using deep structured concept of learning like CNN and using computer vision techniques, an integrated process for detecting leaf diseases was designed and by distinguishing them with the healthy one is designed. More than 50000 images of one healthy and five different classes of diseased leaves was acquired from a freely available Plant Village dataset. Among 900 images, 65% of images were used to train and 30% images for testing. The network was trained using an input hyperspectral dataset and attained accuracy in variation for different number of epochs and convolutional filter sizes [23]. A deep CNN is trained under controlled settings to identify 26 diseases among 14 crop types. The preparation was led with an open dataset of 54306 images including sound and sick harvest leaves [24]. Obtained datasets from 116 spectral signature mark through foliar tests in 4 levels of infections. The investigation has connected artificial neural systems procedures to segregate and arrange diseases in oil palm trees [25].

To encourage wheat crop infection conclusion more than 50000 annotated pictures including solid and harvest have been procured from Plant Village. In a transparent background, seven distinct maladies with 9230 images dataset were gathered from WDD2017 database [26]. About 93 images of cucumber downy mildew were obtained from web and nursery advancement base differentiated in Tianjin Academy of Agricultural Sciences utilizing a camera [26]. The hyperspectral information required to train and test the CNN were acquired from the field [27]. The dataset is gathered



from an open-access database with 50000 images of apple leaves for 30 class marks like early, intermediate and end stages infections [28]. As the neural system can't able to manage images, CNN design has been proposed with convolution and pooling layer for highlight extraction, order layer to characterize the images dataset. These layers are additionally associated with the convolutional and max-pooling layers [18]. A computerized framework for disease determination was created as different occurrences learning-based finding framework for wheat crop infections. The proposed framework was prepared with a start to finish dataset, and the outcomes are assessed to distinguish the exactness of perceiving the diseases and its classification [19].

A pipelined computational model of CNN was created and prepared to separate little bits of images to test the presence of NLB injuries. The expectations from the warmth maps are sustained to the CNN and trains to arrange the entire image and identify the presence of infection. The model is furnished with aerial vehicles with robotized crop phenotyping, viable reproducing for contamination opposition and limits the utilization of pesticides [20]. Two profound learning models are exhibited to improve the exactness and perceive leaf illnesses in maize crop. The models are prepared and tried with several kinds of maize pictures. At last, a near report was made as for the outcomes got and examined their effectiveness [21]. Pursued and planned a half and a half to deal with recognize crop infections by executing vision-based procedures. The system was prepared with the hyperspectral information dataset and accomplished exactness in variety for a few numbers of disease stages and size of the convolutional channel [23].

The model can ready to recognize 13 distinct assortments of plant diseases among the healthy leaves with a capacity to understand the surroundings of the plant leaves [14]. A coordinated framework named Farm-as-a-Service (FaaS) was created and assessed its presentation through investigation of foreseeing the sicknesses in the strawberry crop [29]. The DCNN model was created and led a procedure of perceiving the indications classification for four contaminations of a cucumber leaf. The design is a straightforward and quick way to handle little scale images [30]. An electronic model proposed has helped ranchers to distinguish the ailments in organic pomegranate

product. The framework works by transferring the image with the effectively-prepared dataset and recognizes the contaminations if any [31].

Image pre-processing venture on the dataset was performed to resizing for each picture to 60 \* 60 pixels and later changed over them into grayscale mode [32]. To improve the handling of images they were resized to 1824:1028 [33]. For early recognizable proof of infections, the input images were resized into not more than 224 pixels that may reduce the disease incipient that exists in small size. During the information accumulation, pictures whose goals are under 500 pixels were not considered. The other size pictures were resized into 256 \* 256 pixels that can diminish the time taking to prepare the CNN [14]. The most common and widely recognized picture sizes are 60 x 60, 96 x 96, 128 x 128, and 256 x 256 [34]. Image division is additionally one of the famous practices either to expand the number of pictures in a dataset or to encourage the profound learning process by distinguishing the areas of diseases [35-40].

First, among the other, convolutional activity is to extricate the image highlights. It safeguards the spatial relationship among the pixels by knowing and highlighting through smaller parts of an input picture. For convolution, the mathematical equation is

$$(f * g)(t) \stackrel{\text{def}}{=} \int f(\tau)g(t - \tau) d\tau \quad \infty - \infty \quad (1)$$

Noise filtering is one of the examination strategies used to alter the image and set it up for further handlings like upgrade, growth, division, and shading space change. The fundamental objective of the convolutional activity is to disengage the highlights like hone, obscure, edge recognition and improvement, and embellish from an information image. Each channel is connected to the leaf image to separate the red, green, and blue channels by processing a dot product between the input pixel and channel pixel [32].

In a large area of systems, the CNN layer carries on as highlight extractor from the image whose measurements were diminished further by pooling layer [34]. Pooling layers are utilized to acquire spatial invariance and lessen the goals of an element map. One element guide relates to another element guide of the previous layer [41]. ReLU

activation function is considered to add or improve the non-linearity to the convolutional network and is faster and better than the sigmoid function.

Here the function is represented as

$$f(x) = \begin{cases} x, & \text{if } x > 0 \\ 0, & \text{otherwise} \end{cases} \quad (2)$$

In [23], SoftMax function is applied at the output layer to estimate the probability distribution of a specific event among n various events. For an event  $x_i$ , SoftMax function can be represented as

$$F(x_i) = \frac{\text{Exp}(x_i)}{\sum_{j=0}^k \text{Exp}(x_j)}, \quad \text{Where } i=0,1,2, \dots, k \quad (3)$$

In general, backpropagation [24] adjusts the weights of the input variable and considered as the most powerful learning algorithm [75]. For every node  $j$  in the output, layer perform

$$\Delta[j] \leftarrow f'(in_j) \times (t_i - q_j) \quad (4)$$

Repeat for  $m$  from  $M-1$  to 1 for every node  $i$  in  $m$  layer as

$$\Delta[j] \leftarrow f'(in_i) \sum_j W_{i,j} \Delta[j] \quad (5)$$

Now, update the input variable weights for every

$$W_{i,j} \text{ in nw do as } W_{ij} \leftarrow W_{ij} + \alpha \times \Delta[j] \quad (6)$$

until the condition is terminated and finally return  $nn$ .

### **2.2.2 Image Augmentation Techniques**

Image augmentation has proved to be more effective method in image classification [42]. To amplify the count of images in a training dataset, data augmentation approaches was applied and thus improve the efficiency of CNN and deep learning sustainability. The most common methods of augmentation are rotation, zooming, shifting, modifications in color palette, and applying distortion. But these are not significantly enough to improve the efficiency of a CNN model or to handle the issue of overfitting [43]. Overfitting in a training dataset can be reduced by introducing distortion to the existing images. In the fully connected network, the regularization technique called dropout is implemented to reduce the overfitting [44]. In conjunction with deep learning architectures, data augmentation plays a vital role in handling the issue of overfitting [30].

Simple image rotation, affine and perspective transformations are the other image augmentation approaches to increase the quantity of images in a training set [14]. Affine transformation and image rotation techniques are used on the original dataset to increase its size as the number of images on the dataset are very few [46]. Rotation image augmentation is implemented on the original dataset and rotated the images in 90, 120, and 270 degrees. Later horizontal and vertical flipping is applied to produce twelve augmented images dataset [30]. After initializing the original dataset, additional versions of images are created by rotating them in 90, 120, and 270 degrees and mirroring them. Later, the centre portion of the image is cropped by the same size and processed them to grayscale images [21]. The most common techniques like horizontal or vertical flip, rotation at few degrees, inward or outward scaling, random crop, translation and noise injection are implemented to increase the dataset size [48]. Cropping of images is important that are captured in uncontrolled conditions with background complexity. This process can be implemented manually or automatically [49].

Removal of distortion can improve the images for further pre-processing. Color space conversion including cropping and filtering are the pre-processing techniques in this aspect [50]. HSV color space is another pre-processing augmentation that resembles the

properties of human color sensing [51]. This technique is best suited to handle image identification challenges that occurs due to unclear objects. Comparative Analysis of significant research contributions affirm to several image augmentation techniques for class balancing of crop images is shown in Table 2.3. The performance measure is used to evaluate the efficiency of different augmentation techniques. In Table 2.3, performance measure is calculated using the equation 7 derived from [75] as follows:

$$PM = \frac{x_2 - x_1}{x_1} \times 100 \quad (7)$$

where  $x_1$  is the initial input dataset size,  $x_2$  is the final dataset size, PM is the performance measure.

Table 2.3 Significant research contributions

RC. No	Authors and Year of publication	Size of Input Dataset	Traditional Augmentation Techniques	Conventional Augmentation Techniques	Final Dataset	Performance Measure (enhancement) in terms of multiples
1	Luis Perez et al [43], 2017	1000	shift, zoom in, zoom out, rotation, flip, distortion	Enhance, cezanne, monet, ukiyoe, van gogh, and winter	2000	100
2	Shanqing Gu et al [48], 2019	NM	Flip, rotate, crop, scale, whitening	NA	60000	ID
3	Srdjan Sladojevic et al [77], 2016	4483	Rotation	Affine Transformation and Perspective Transformation	33469	746.5
4	Ekin D et al [52], 2018	NM	Rotate, Invert, Contrast, Color, Brightness, Sharpness,	Shear X/Y, Translate X/Y, Equalize, Solarize, Posterize, Auto Contrast,	23591	ID

				Sample Pairing, Cutout		
5	Ramcharan, A et al [53], 2017	11670	Cropping	NM	15000	128.5
6	Juncheng Ma et al [30], 2018	1184	Rotate 90, 180, and 270 degrees, flip horizontal and vertical	NM	14208	1200
7	Dyrmann M et al [46], 2016	10413	NM	NM	50864	488.4
8	Xihai Zhang et al [21], 2018	500	Rotate 90, 180, and 270 degrees, crop, grayscale	NA	3060	612
9	Mohanty SP et al [45], 2016	NM	Color, and grayscale	Leaf segmentation	54306	ID
10	Barbedo JGA et al [47], 2016	NM	Grayscale	Color Transformation	1335	ID
11	Proposed Work	4758	<a href="#">Annexure 1</a>		65470	1375

ID-Insufficient Data, NA-Not Applied, NM- Not Mentioned

This section presents and summarizes work related to our proposed work. Many researchers have put a lot of effort to detect diseases at an early stage. Different types of plant diseases have been identified using advanced machine learning methods and algorithms. A method for diagnosing diseases in grape leaves based on identification is proposed [54]. The input images were gathered from a variety of sources, including the internet and the field. Later, to maximise the number of training images, image augmentation techniques were used. The accuracy of the proposed model is 97.22 percent, which is higher than some of the pre-trained models. An updated LeNet architecture based on CNN has been proposed [55]. The model was trained and tested

using images of maize leaf diseases from the Plant Village dataset, with a classification accuracy of 97.89%.

An improved CNN model for classifying five diseases in apple leaves is proposed. The images were taken from the original field and scored 78.8 mAP in classification [56]. To detect maize leaf diseases, an AlexNet-based architecture is proposed. When compared to baseline approaches, the model showed a 98.62% accuracy [57]. To classify diseases in corn is proposed an approach based on a support vector machine (SVM). During the successful classification of diseases, the algorithm achieved a 97% accuracy rate [58]. A classification model for identifying diseases in cucumber is proposed. The SVM algorithm-based model achieved a total classification accuracy of 98.13 percent, which included both healthy and diseased plants [59].

The study presented using the Fuzzy C-Means algorithm and K-fold cross-validation to detect the diseases in grapes. The model was capable of distinguishing the diseases with an overall accuracy of 98.6% [60]. An integrated CNN model for grape disease classification is suggested. The experimental results show that the model achieved 98.57% average accuracy [62]. To detect the pest in rice plants an adoptive CNN model is proposed and developed. As compared to existing pre-trained models, the architecture achieved a 93.3 % accuracy rate [63]. A 9-layer CNN model was proposed to identify 39 different plant diseases [61]. An adoptive CNN model to detect the pest in rice plants is proposed and the architecture has gained 93.3% accuracy and compared its performance with the existing pre-trained models [63]. A 9-layer CNN model to classify 39 different diseases of various plants is proposed. The model's experimental results appeared higher than conventional methods, with 96.46% accuracy [61].

## CHAPTER 3

### DATA ACQUISITION AND ENRICHMENT

The most considerable drawback while using deep learning models is the unavailability of large number of images in the training dataset. The size of the dataset can serve as the sufficient and proper input to the model during the training procedure. In reality, at least some hundreds of images are needed to use the model for efficient classification. The solution to increase the quantity of images in a dataset is applying image augmentations without losing the originality. The past researches have already proved that the image augmentation techniques can artificially increase the number of images in a training dataset. The techniques can improve the overall process of learning and model classification efficiency. Thus, two image acquisition approaches have been proposed to systematically obtain images from the internet sources and real agricultural field. Later, another algorithm is proposed to describe the procedure how the augmentation techniques can be applied to enhance the datasets.

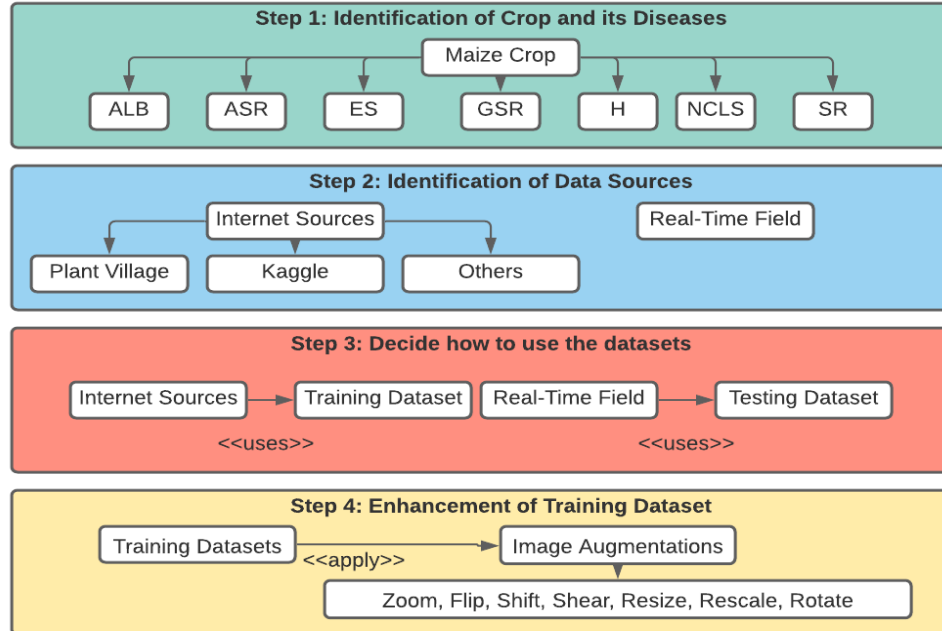


Fig 3.1 Four-step process of data collection

During this study, various augmentation techniques like zoom, flip, shift, shear, resize, rescale and rotate with different values has been investigated. Later, the techniques are applied to an original image and observed that the size is increased more than 350 times



higher than the original dataset size. The application of augmentation techniques proved that the suggested techniques are the best while dealing with unbalance datasets and reduce model overfitting. The overall process followed during data acquisition and enrichment process is shown in Fig 3.1.

### 3.1 Image Acquisition

For object recognition and image classification, appropriate image datasets are required at training and testing phase to run the model. Initially, a total of 4758 maize crop images have used from Kaggle and Plant Village dataset [73] for classification. The images captured from real-field were used for testing the model. The images downloaded from the internet has searched by object name or its class name on various sources databases, repositories and websites in English language. A clear and detailed methodology to acquire datasets has been depicted in Fig 3.2.

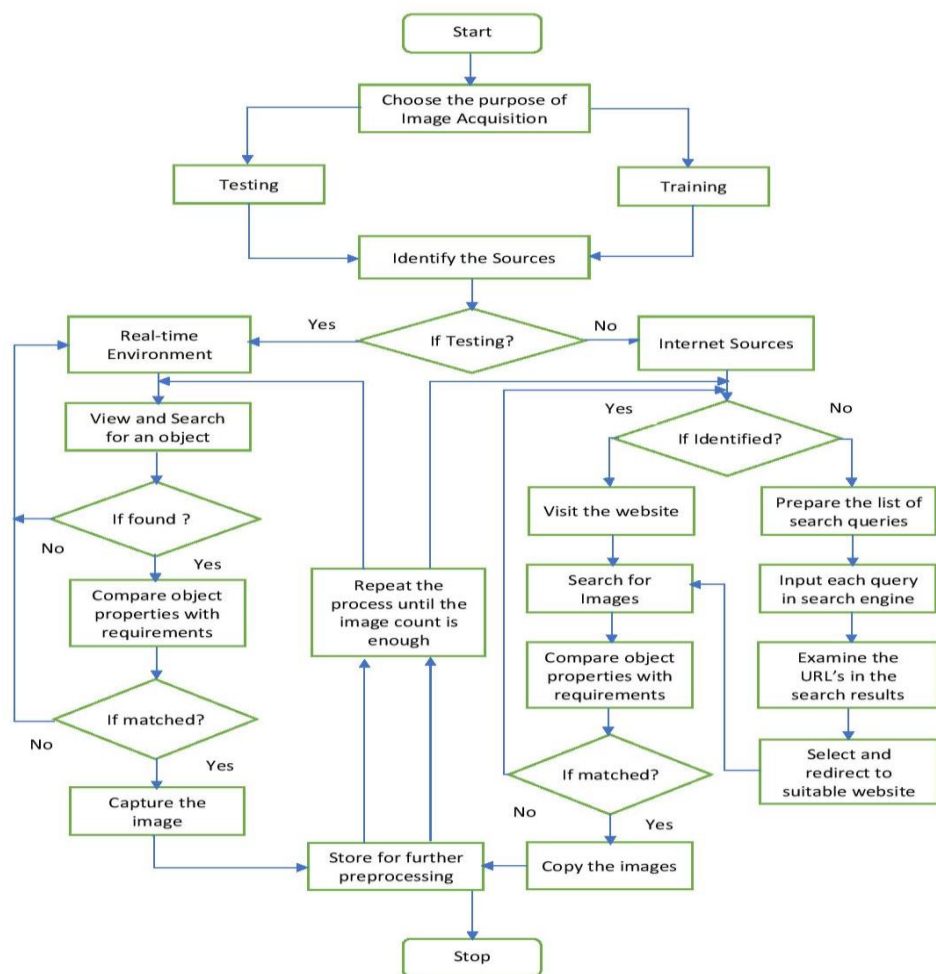


Fig 3.2. Detailed methodology of image acquisition process

In this sub-section, two image acquisition algorithms have been proposed to systematically acquire and enrich the datasets for learning process of the CNN model. Algorithm1 describes the process of collecting the images required to train the model. Algorithm2 describes the process of capturing the images required to test the model.

---

**Algorithm 1: Image Data Acquisition Process for Training**

---

Step1: **procedure** Image\_acquisition ( $X \rightarrow$  Images from Internet Sources)

Step2:  $max\_images \rightarrow$  Set the maximum number of images required

Step3:  $max\_classes \rightarrow$  Set the values as '7' for seven disease classes

Step4:  $object\_properties \rightarrow$  Set the object properties for each class

Step5: Identify the image sources in Web( $W$ )

Step6: repeat

Step7: **for**  $i < max\_images$  **do**

Step8:           **for**  $c \leq 7$  **do**

Step9:                   **for**  $m < max\_per\_class$  **do**

Step10:                           **If found**  $img$  **then**

Step11:                                   Visit ' $W$ ' for ' $DS(img)$ ' and perform the search process

Step12:                                   Identify  $img$   $X$  with required ' $obj\_properties$ '

Step13:                           **If found then**

Step14:                                   Compare ' $obj\_properties$ ' with requirements

Step15:                                   **else**

Step16:                                   Until  $img$   $X$  is found

Step17:                                   **If matched then**

Step18:   Identify  $img$  class  $x$  of captured image  $X$

Step19:                                   **else**

Step20:   Until  $X(obj\_properties)$  is matched

Step21:                                   Load  $X$  into dataset

Step22:                   **end for**

Step23: Until ' $max\_images$ ' is reached

Step24: **end for**

Step 25: Repeat

Step 26: Perform Image Augmentations

procedure augmentation\_process(*img<sub>x</sub>* → **original image**)

Randomly select an image *img<sub>x</sub>* from a training dataset *DS<sub>input</sub>*

Initialize ‘*img<sub>x</sub>*’ with pre-selected augmentation technique *t<sub>i</sub>* where *i*= 1 to *n*

*n* ← Set the number of ‘*img<sub>y</sub>*’ to be generated

*m* ← set the number of augmentations

Apply augmentations to obtain new image ‘*img<sub>y</sub>*’

Apply for position augmentation

position\_augment(*aug<sub>p</sub>*);

Store *img<sub>y</sub>* in *DS<sub>input</sub>* = { *DS<sub>input</sub>*, (*img<sub>x</sub>*, *img<sub>y</sub>*) }

Apply for color augmentation

color\_augment(*aug<sub>c</sub>*);

Store *img<sub>y</sub>* in *DS<sub>input</sub>* = { *DS<sub>input</sub>*, (*img<sub>x</sub>*, *img<sub>y</sub>*) }

Until *aug<sub>t1</sub>* to *aug<sub>tm</sub>* for each image of each image class

Step27: Repeat

Step28: Apply Image Fine Tuning

fine\_tune(*DS<sub>input</sub>* (*new*));

Store HSI (*img<sub>y</sub>*[*i*]) in *DS<sub>input</sub>*

Step29: Until *aug<sub>t1</sub>* to *aug<sub>tm</sub>* for each image of each image class

Step30: **end procedure**

---



---

**Algorithm 2: Image Data Acquisition Process for Testing**

---

Step1: **procedure** Image\_acquisition (Y → Images from real field)

Step2: max\_images → Set a value representing maximum number of images

Step3: max\_classes → Set a value of ‘7’ for seven disease classes

Step4: object\_properties → Set the object properties for each class

Step5: Identify the real-field environment

Step6: Visit the location

Step7: Repeat

Step8: **for i < max\_images do**

Step9:           **for c <= 7 do**

Step10:                   **If found *img* then**

Step11:                           Identify ‘Y’ with object

Step12: Compare the *object\_properties* with the requirements

Step13: **If matched then**

Step14: Capture ‘Y’

Step15: Identify the image class ‘y’ of ‘Y’

Step16: **else**

Step17: Until *object\_properties* are matched

Step18: Load Y into dataset

Step19: **end for**

Step20: Until *max\_images* are reached

Step21: **end for**

Step22: **end procedure**

---

The next process after image acquisition is to apply various augmentation techniques to enrich the training dataset with the existing images. The augmentation can help a CNN model to learn more image features that contribute for the better and more accurate image classification.

### 3.2 Image Augmentations

The most critical problem facing by the researchers is gathering good quality and quantity of data. Image Augmentation (IA) is crucial to enrich the input dataset and also enhances the performance of any deep learning models. The insufficient data issue has brought the life to image augmentation. Data Augmentation is a process to increase the numbers of image samples in a dataset using the images that already exists. The augmentation process is the most important technique that can be performed manually or automatically. The most popular augmentation techniques implemented in this research are position augmentation and color augmentation.

Image Data Augmentation is necessary to determine the final size of the input dataset. The results after implementing IA techniques can improve the input dataset size from N to 2N. All the augmentation techniques like horizontal-vertical (HV), horizontal-vertical shift (HVS), horizontal-vertical flip (HVF), random rotation (RR), random zooming (RZ), and random brightness (RB) are applied to the images in the input space.

---

**Algorithm 3: Image Augmentation and Fine-Tuning Process**

---

**Function** `color_augment(DSinput):`

```
i=0
for c in image class do
    Apply color augmentation augc to a selected imgx.
    Select another augi.
    Generate the output imgy of augi.
end for
return augc [imgy]
```

**Function** `position_augment(DSinput):`

```
i=0
for c in image class do
    Apply position augmentation augp to a selected imgx.
    Select another augi.
    Generate the output imgy of augi.
end for
return augc [imgy]
```

**Function** `fine_tune(DSinput (new)):`

```
i=0
for c in image class do
    for each imgy[i] in DSinput(new)
        Regularize the size of imgy using ‘w’ and ‘h’
        Generalize the resolution of imgy using ‘res’
        Convert RGB (imgy[i]) to HSI (imgy[i])
        Increment ‘i’ in imgy[i]
    end for
return DS(HSI(imgy))
```

---

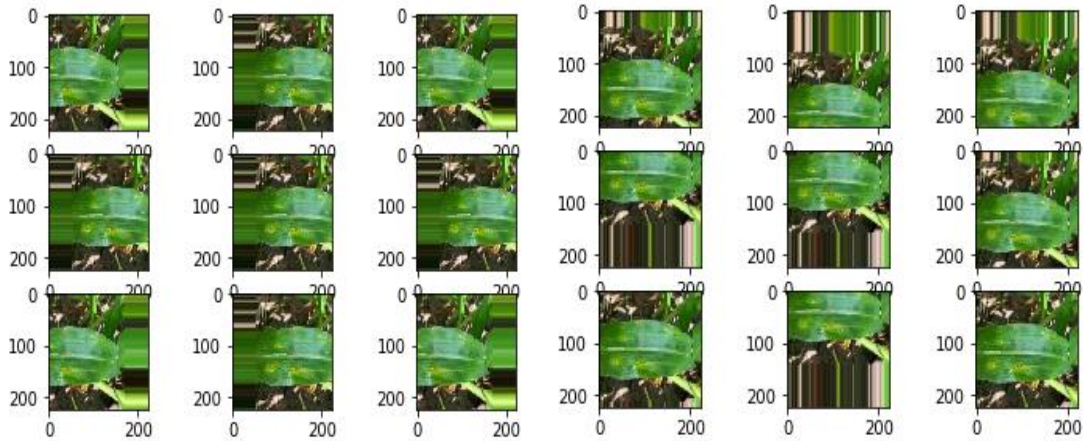


Fig 3.3. Position Augmentations: (a) HV (b) HVS

Fig. 3.3(a) represents a horizontal shifting (HS) augmentation technique with a range between  $[-200,200]$  pixels and generates new images. The image creates an instance for image augmentation and performs nine iterations plotting a copy at each iteration. The plotting results in a range of nine different randomly considered positive and negative HVS. The image corner is duplicated with the pixel values to fill the empty area of the image generated by the shift. Fig. 3.3(b) represents a vertical shift with a shift range of 0.5, specifying the shift percentage as the image height.

The HVF technique reverses the rows/columns pixels of an input leaf image. Fig. 3.4(a) demonstrates the horizontal shift by creating the nine augmented RGB images. The above figure shows the outcomes of the horizontal flips (HF) performed by applying the random flips (RF).

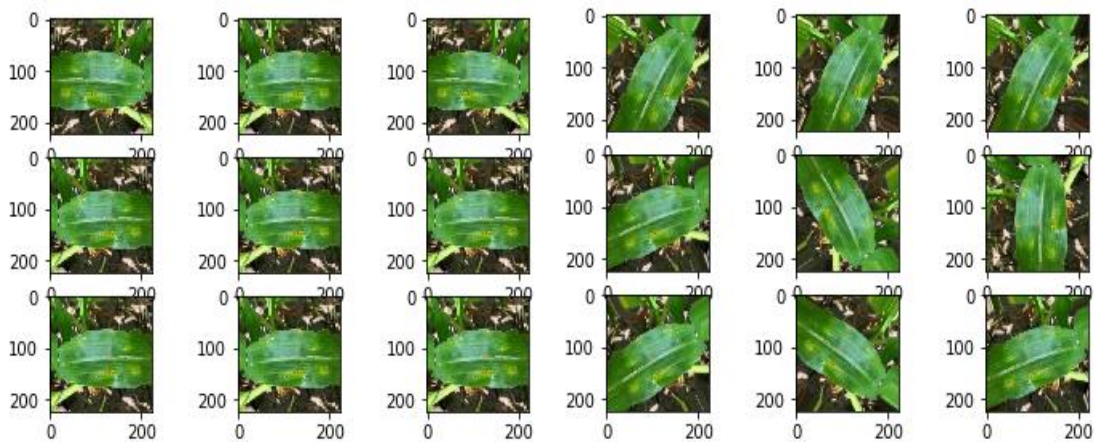


Fig. 3.4 Position Augmentation: (a) HVF (b) RR

RR augmentation technique, when applied rotates the image pixels and fills the remaining area of the frame with no pixels. Fig 3.4 (a)(b) demonstrates the random rotation technique by generating nine images rotated by selecting random angles between 00 and 900.

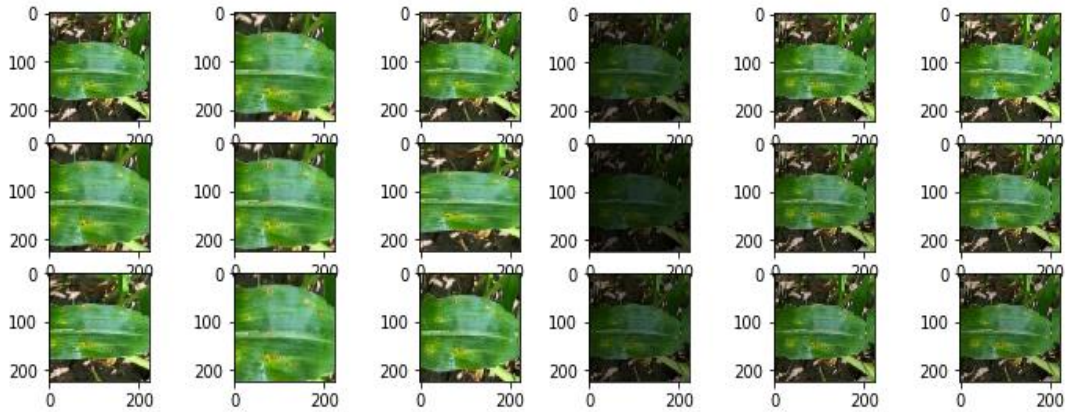


Fig. 3.5 Position and Color Augmentation: (a) RZ (b) RB

Fig 3.5(a) demonstrates the RZ augmentation technique. The technique generates nine different RGB images with different randomly zooming the input image by adding the new pixels around the image or taking the pixel values and interpolates. Fig 3.5(b) outputs the new images by applying the random zooming with values ranging from 50% zooming (0.5) and no zooming (1.0).

Fig 3.5(b) demonstrates the RB augmentation technique. This technique's application generates nine different RGB images with varying levels of lighting either by darkening, brightening, or both to the images with random values between 20% brightness (0.2) and no intensity (1.0).

The proposed algorithm<sup>3</sup> has generated 8542 more new images with the existing 4758 old images. After the augmentation process, the training set is ready with 13300 images to feed to the model. A total of 2660 images acquired from the real agriculture field were used to test the proposed CNN model to perform the classification of seven different disease classes. Among the classes, six are the common diseases and the other one is healthy leaf images of maize were considered. The process followed during images collection is shown in Fig 3.6. The complete details of the image's dataset are mentioned in Table 3.1 and some samples are shown in Fig 3.7.

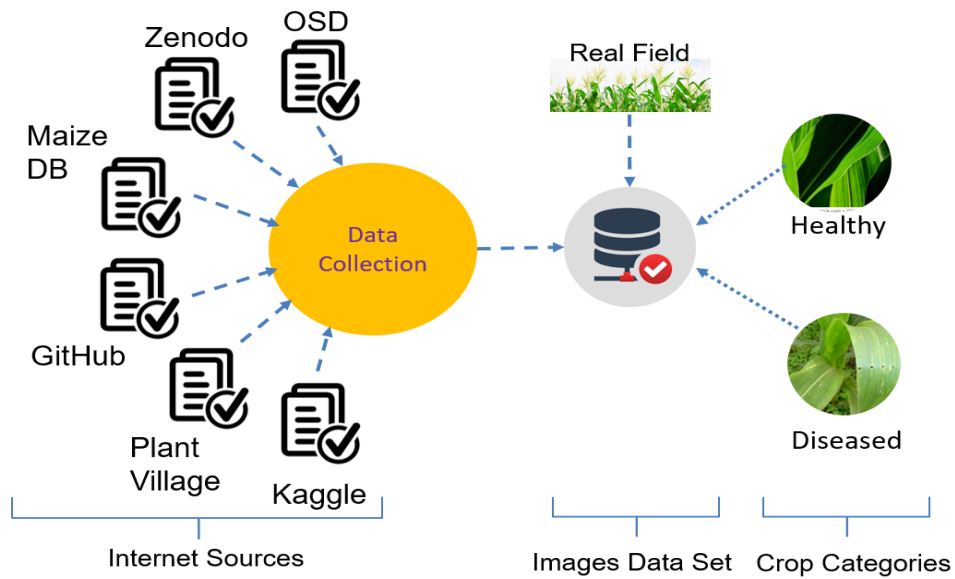


Fig 3.6. Data collection process

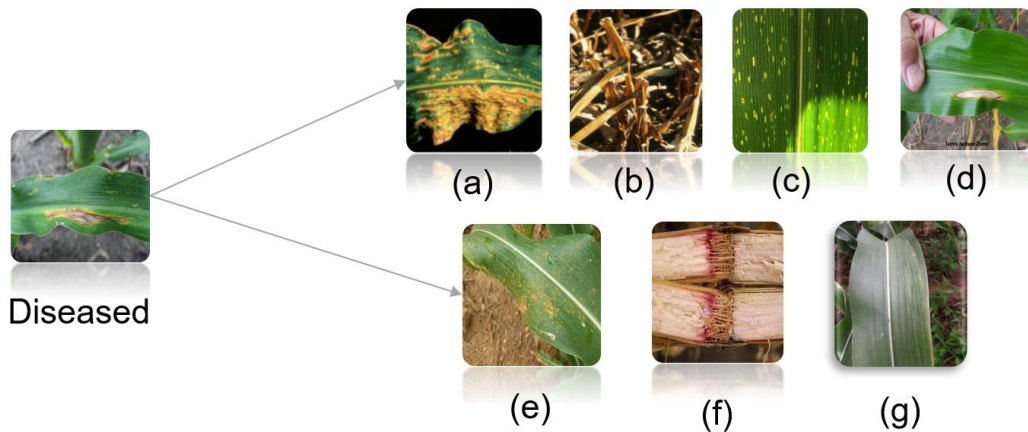


Fig 3.7 Seven Classes of maize crop disease categories (a) anthracnose leaf blight, (b) anthracnose stalk rot, (c) eyespot, (d) northern corn leaf spot, (e) southern rust, (f) Gibberella stalk rot (g) healthy

Table 3.1. Details of maize crop images

Disease Class	# Images used for training	# Images used for testing	Total Number
Anthracnose Leaf Blight (ALB)	1900	380	2280
Anthracnose Stalk Rot (ASR)	1900	380	2280
Eye Spot (ES)	1900	380	2280
Gabriella Stalk Rot (GSR)	1900	380	2280



Health (H)	1900	380	2280
Northern Corn Leaf Spot (NCLS)	1900	380	2280
Southern Rust (SR)	1900	380	2280
<b>Total</b>	<b>13300</b>	<b>2660</b>	<b>15960</b>

### 3.3 Image Pre-processing

Pre-processing is a technique for improving image quality by removing unwanted distortions and enhancing key features. It enables the proposed model to benefit from the learned features while understanding the image class. Image resize and rescale is used as pre-processing techniques. All of the images in the dataset have pre-processed to a regular size of 224 x 224 before being fed into the new model. To boost resolution, the dataset images are pre-processed to 1/.255 scaling.

### 3.4 RGB to Hyperspectral Image Conversion

A data-driven approach is followed to construct the hyperspectral images from RGB images. All the training and testing dataset RGB images are converted into hyperspectral images (HSI) with ' $n$ ' number of target spectral bands. Later, the HSI images are used to perform classification process.

## CHAPTER – 4

### DESIGN AND DEVELOPMENT OF A CNN MODEL

Identifying diseases in crops by accurately extracting the features is becoming more important and difficult task. The accurate recognition of diseases in crop is desired in the field of agriculture. A Convolutional Neural network or simply CNN is a special kind of multi-layer neural network model that performs the identification and classification of diseases in a systematic manner. CNN has been used to identify crop diseases in very few research so far. The motivation for proposing a new deep learning model is to help farmers identify maize crop diseases at the earlier stage only. An NPNet-19 model is designed, trained, and tested using a novel deep learning recognition and classification approach to identify diseases. The proposed deep learning model enhances the analytical outcomes by extracting features from raw images in a systematic manner. The model can recognize the diseases earlier by monitoring and classifying the images captured with a digital camera.

#### 4.1 Development of a CNN Model

In this sub-section, a detailed and a systematic process of developing a proposed neural network model is described. The development approach is a step-by-step process that clearly describes how the fully connected deep convolutional neural network is developed.

##### 4.1.1 Convolutional Layer

During this step, the development process of a convolution layer is clearly described. The first step is the convolution operation. It is a mathematical operation that can be applied on two functions ‘f’ and ‘g’ and outputs a third function ‘e’. The new function expresses how one disease symptom image is modified by the other.

$$(f * g)[n] = \sum_{m=-\infty}^{\infty} f[m]g[n - m] \quad (1)$$

for each value of ‘n’ and a dummy variable ‘m’. The convolutional operation has three elements, such as input symptom image, feature detection, and feature map. The input image is an infected one that can be represented as a pattern of 1’s and 0’s where 1 indicates infected portion and 0 indicates noninfected portion. The image is represented as a 3 x 3 matrix or cells for smaller images and 5 x 5 or sometimes 7 x7 matrices for a larger one and is treated as a feature detector. The matrix is also often referred as “filter” or “kernel”. A Feature detector is an image processing function usually performs to examine every pixel of an input image to determine the presence of an infection feature (add symptoms here) present at that pixel. The detector slides through the image with certain infection information and filter the cells (a small portion of an image) that are integral to it by excluding the remaining cells. From the input image, CNN creates many feature maps to obtain the first convolutional layer. The research develops more feature detectors and uses them to generate various feature maps that are further referred to as a convolutional layer. Through proper training, the CNN understands the different features that exist on the input leaf images and categorizes them into different disease classes based on their similarities.

Convolutions or convolutional layers are the essential elements of neural networks. The convolution layer consists of several filters, filter size, input shape, padding, stride, and activation function. The layer filters are independent and generate feature maps by convolving around the input image as shown in Fig 4.1. Here, we convolve an image of size  $I_w$  width and  $I_h$  height with a filter of  $w \times h$  and obtains an output feature map of size  $If_w$  and  $If_h$  as given in the equation 2 and 3 and shown in Fig 4.2.

$$If_w = \frac{I_w - w + 2pw}{sw} + 1 \quad (2)$$

$$If_h = \frac{I_h - h + 2ph}{sh} + 1 \quad (3)$$

In eqn (2) & (3),  $If_w$  and  $If_h$  are the width and height of the feature map generated after performing convolution of an image. Here,  $ph$  and  $pw$  are the height and width after zero padding,  $sw$  and  $sh$  are the strides in both horizontal and vertical directions respectively.

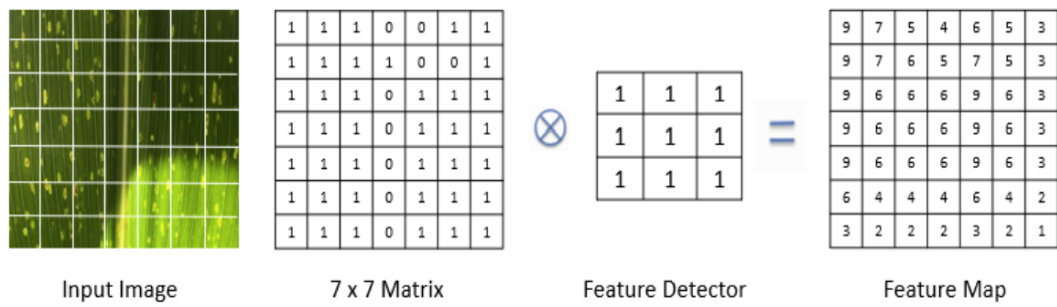


Fig 4.1. Feature map extraction from an infected leaf image

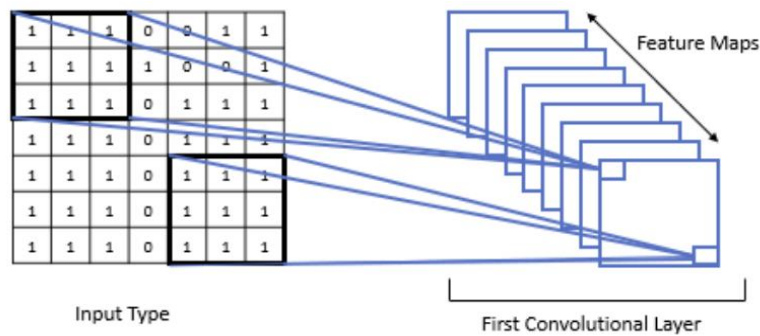


Fig 4.2. First CNN layer development process

Second process is applying the noise filter. Convolutional matrix is generated to adjust an image by reducing the noise. It is the concept called filter which is being applied to leaf images using sharpen, blur, edge detect, etc. These techniques are used to identify and alter the disease features in the same manner as the original one. Fig 4.3 shows some of the images adjust with a convolutional matrix using various filters. used



Fig 4.3. Sample images generated after adjustment with filters

### 4.1.2 Activation Functions

#### Step 1(c): ReLU Function

The most common nonlinear activation function used in convolutional layers is Rectified Linear Unit, or simply ReLU. The nonlinearity is calculated using equation 4 as follows:

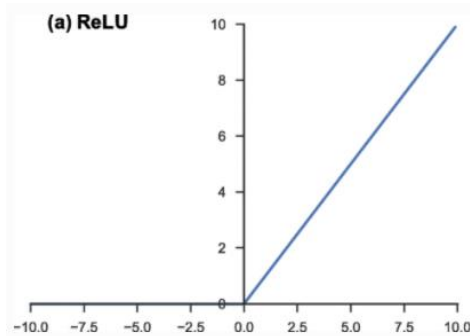
$$f(x)=\max (0, x) \quad (4)$$

While training a CNN, ReLU function is used to remove the linearity in an image because they have poor performance in terms of efficient identification of disease. Fig. 4.4(a) shows the graphical representation of ReLU activation function and the process how non-linearity is removed from the image feature matrix is shown in Fig 4.4 (b). When a rectifier function is applied to a diseased leaf image that removes the black elements from it and keeps all the positive values like grey and white color.

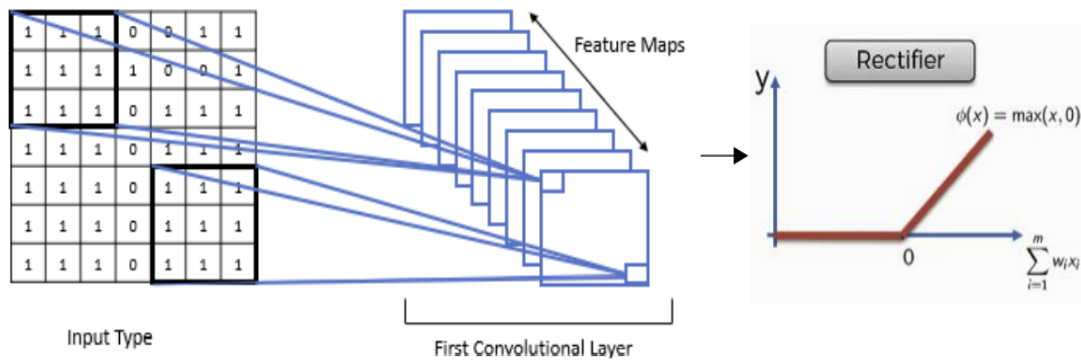
Third, categorical cross entropy is used for classification as there are more than two diseases being considered. The mathematical classification equation is denoted as

$$Y(x) = WT\Phi(x) + b \quad (5)$$

where  $Y(x)$  is the output classifier with positive and negative values. Positive values denote the symptom feature belongs to one disease class, whereas negative values denote another disease class. Here,  $WT$  and  $b$  denote weight and bias.



(a) Non-linear activation



(b) Application of ReLU

Fig 4.4. Applying a rectifier function to remove non-linear image features

The softmax activation function is applied to the classification of seven disease classes. This function normalizes the output image values to identify specific class probability. Fig 4.5 shows the multi-class activation graph of the function used during the classification process.

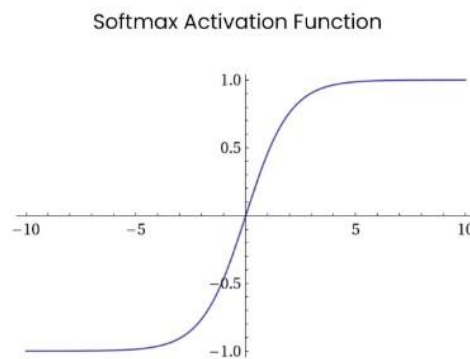


Fig 4.5 Activation function SoftMax

### 4.1.3 Batch Normalization

The Batch Normalization (BN) technique is implemented while designing the CNN to normalize the given inputs to a specific layer for every mini-batch. Adding this layer can stabilize learning and reduce the required number of epochs for training the network dramatically. In general, BN layer is a 4-dimensional tensor representing both the input and output that refers to  $I_{b,c,sd1,sd2}$  and  $O_{b,c,sd1,sd2}$ , respectively. Here  $b$  is a 'batch',  $c$  is the 'number of color channels',  $sd1$  and  $sd2$  are the 'spatial dimensions'. For all the

activations, BN applies a similar normalization in a provided channel as given in equation. 6 [64]:

$$O_{b,c,sd1,sd2} \leftarrow \gamma_c \frac{I_{b,c,sd1,sd2} - \mu_c}{\sqrt{\sigma_c^2 + \epsilon}} + \beta_c \quad (6)$$

where ' $\beta$ ' contains all channel ' $c$ ' activations, overall image features ' $b$ ' in the mini-batch including all  $sd1$ ,  $sd2$  spatial locations. Meanwhile, BN divides the common activation using the standard deviation ( $sd$ )  $\sigma_c$  plus numerical stability ' $\epsilon$ ' by following the  $\gamma_c, \beta_c$  channel-wise affine transformation parameters during the learning process.

#### 4.1.4 Max Pooling

As a convolution process, Max Pooling (MP) extracts the maximum value of the image area it convolves. The pooling layer accumulates the feature maps generated while convolving the kernel over an input image. The MP reduces the spatial size representation by minimizing the number of parameters and the dimensionality of an image. It means that the MP operation reduces the number of pixels in every feature map and produces the summarized version of the input image with detected features. Now, the output feature map will contain only the most prominent values of features of the previous feature map. This process improves the model performance capability due to the reduced number of learning parameters and taking less computational time. MP generates a filter for each layer as a single value using equation. 7 where  $Z_f$  is a feature vector, and ' $s$ ' is the number of strides [65].

$$Z_f = \max\{s\} = \max \{s1, s2, s3, \dots, sn\} \quad (7)$$

Max Pooling is concerned with training CNN in a way that it can be used to recognize the diseased leaf despite of all its representations like normal, rotate and squashed view. Fig 4.6 shows some sample images generated after performing random, rotate, and squashed transformation. The implementation of this process allows to detect the diseased image without worrying the difference in image textures, location, and the angle of an image captured or otherwise. The output of the previous step is a pooled feature map. This map is flattened into a column-wise representation that will be

inserted into CNN in further steps. Based upon the number of pooled features maps the number of layers in CNN depends.

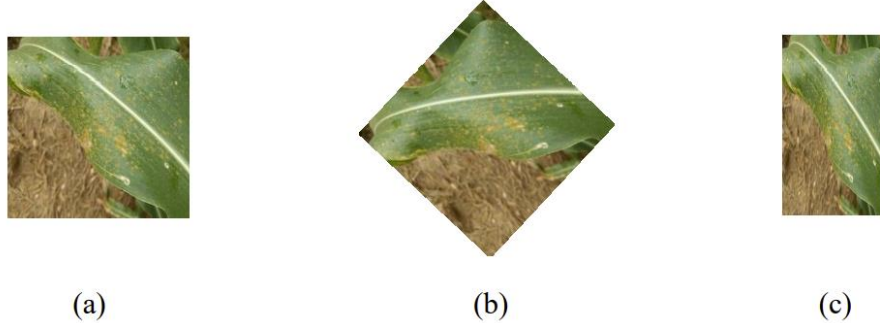


Fig 4.6. Different versions of image representations (a) original (b) rotated and (c) squashed

#### 4.1.5 Dropout

The dropout (D) layer allows reducing the problem of overfitting before performing the classification. First, the vector  $Zd$  is being facilitated by randomly dropping some elements with a probability of  $p$  provided by the Bernoulli Distribution (BD). In the softmax layer, the class-wise prediction is computed with a reduced vector and weights as given in the equation. 8 [64]:

$$O = Z_d W_s + d \text{ where } W_s \in R^{m \times k} \quad (8)$$

#### 4.1.6 Fully Connected Layer

Flatten or Fully Connected (FC) layer implies that all the previous network layer neurons connect to all the present network layer neurons. The number of neurons in the last layer will be the same as seven classes to be predicted as per the current work. In general, FC will have two parameters like weight  $w$  and bias  $b$ . The change in error during linear transformation is calculating using an equation. 9 as:

$$Z = W^T \cdot X + b \quad \text{where } X \text{ is the input} \quad (9)$$

In this step, a fully connected convolutional neural network is created with an input layer, hidden layers, and output layer. The input layer is the vector of input data collected from step 3. This step will bring the proposed Deep CNN to the next level with more complexity and precision. The role of this step is to take the features data



into a variety of attributes to make the CNN more capable of classifying seven different input diseased leaf images as of Fig 4.8.

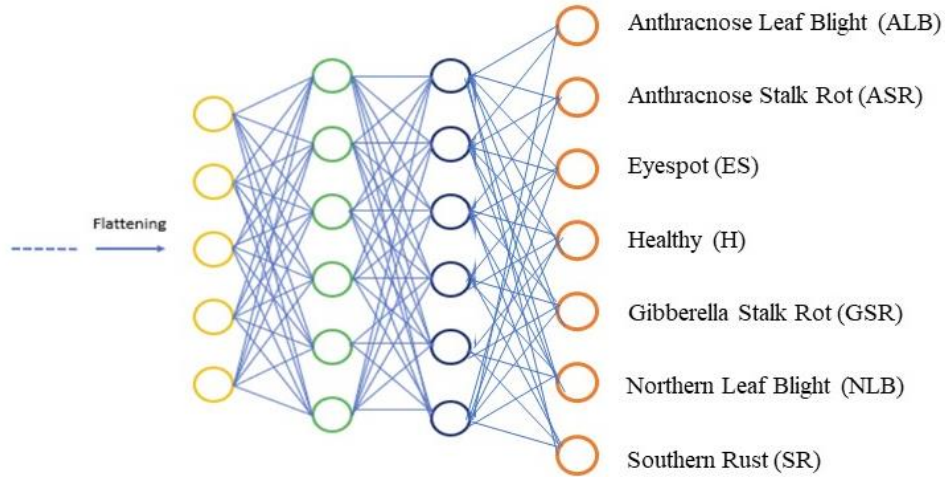


Fig 4.7. Proposed fully connected CNN Model

After the successful development of the model, the research continues with the implementation to identify the infected maize crop leaf, classifies them into seven classes based on the symptoms (features) and determines what type of infection the leaf is affected with. The complete process of implementation is described in the next sections.

## 4.2 CNN Model Design

CNN is a deep learning algorithm used for recognising and analysing visual imagery. Computer vision technology motivates CNN to learn features in an input image and allows to distinguish it from other images by allocating weight and bias to various objects. The CNN architecture is modelled with the human brain's neuron communication pattern. CNN is a multilayer perceptron with connectivity between adjacent layers of neurons. Every layer is a container for neurons, and layer N-1's input is a subset of neurons from layer N.

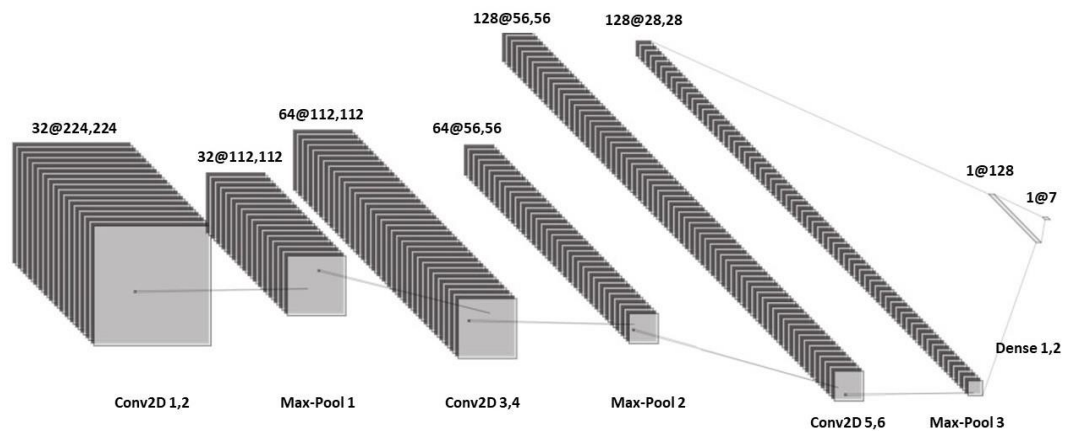
After trying a number of designs approaches the model has been developed by overcoming the problem of overfitting. The proposed NPNet-19 architecture is a CNN-based sequential model designed with several layers like convolutions, batch

normalization, max pooling, dropout, flatten, and dense imported using Tensor Flow 2.4.1 backend from the Keras library. The proposed model is a 3-level network architecture with a total of 19 internal layers. The first network level holds 2 convolutional layers with 32 filters (kernels) of size 3 x 3, one max-pooling layer, one batch normalization layer, and one layer with 30% dropout. The second network level holds 2 convolutional layers with 64 filters (kernels) of size 3 x 3, one max-pooling layer, one batch normalization layer, and one layer with 30% dropout. The third network level holds 2 convolutional layers with 128 filters (kernels) of size 3 x 3, one max-pooling layer, one batch-normalization layer, and one layer with 30% dropout. The model consists of 1 flatten layer and two dense layers. Among the dense layers, one layer is with 128 units and a ReLU non-linear activation function and the other layer is with 7 units and a 'softmax' multi-class classification function. Table 4.1 shows the summary of the proposed model. The three different architectures of the proposed CNN model are shown in Fig 4.8.

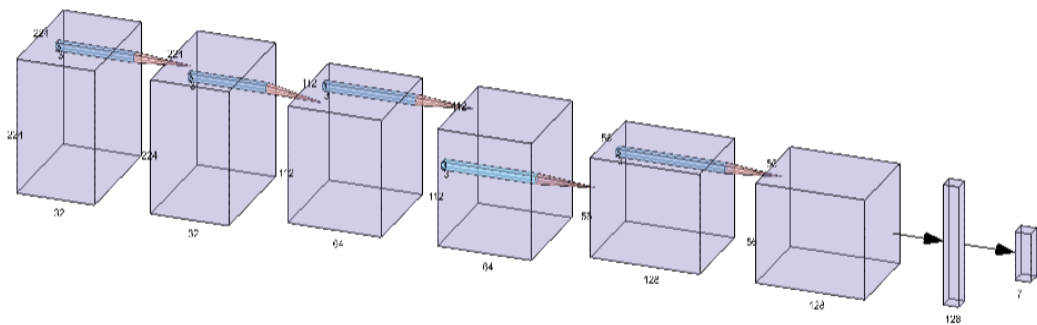
Table 4.1. Proposed NPNet-19 model summary

<b>Layer Category</b>	<b>Output Shape</b>	<b># of Param</b>
conv2d (Conv2D)	(218, 218, 32)	4736
conv2d (Conv2D)	(212, 212, 32)	50208
Batch Normalization	(212, 212, 32)	128
Max_Pooling2D	(106, 106, 32)	0
Dropout	(106, 106, 32)	0
conv2d (Conv2D)	(100, 100, 64)	100416
conv2d (Conv2D)	(94, 94, 64)	200768
Batch Normalization	(94, 94, 64)	256
Max_Pooling2D	(47, 47, 64)	0
Dropout	(47, 47, 64)	0
conv2d (Conv2D)	(41, 41, 128)	401536
conv2d (Conv2D)	(35, 35, 128)	802944
Batch Normalization	(35, 35, 128)	512
Max_Pooling2D	(17, 17, 128)	0

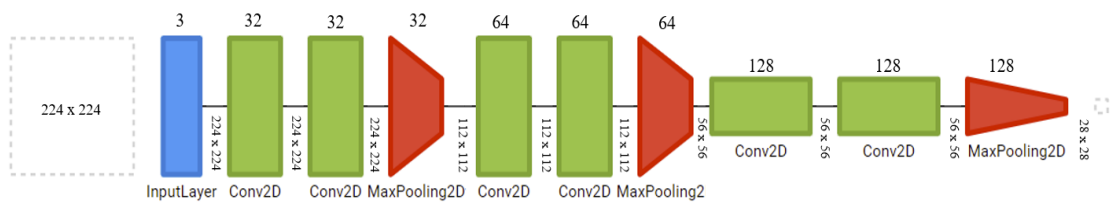
Dropout	(17, 17, 128)	0
Flatten	(36992)	0
Dense	(128)	4735104
Dropout	(128)	903
Dense	(7)	0



(a) LeNet style



(b) AlexNet style



(c) Conventional style

Fig 4.8. Proposed NPNet-19 Architecture styles

## CHAPTER – 5

### VERIFICATION AND VALIDATION OF THE PROPOSED MODEL

In this section, the performance of the proposed model is verified by changing the hyperparameter values. Next, the model performance has compared with the pre-trained models and other existing models. Images related to seven different classes of maize crop diseases is chosen for the experimental approach. A total of fifty-five experiments has carried out to validate the model performance. All the experiments have conducted using Keras library and Tensor Flow 2.4.1 backend.

#### 5.1 Work Environment

The experiment has been performed on a Windows 10 Operating System with Intel CPU @ 2.3GHz i7 Processor and 16GB RAM. The machine is accelerated by an NVIDIA GeForce RTX 2070 Super Max-Q GPU with 8GB GDDR6 card with CUDA 2560 Cores.

#### 5.2 Verification of the model

The evaluation of the proposed model has performed to determine the classification efficiency using several hyperparameters with their default values. The process of classification performance of the proposed model using training and testing image sets is shown in Fig 5.1.

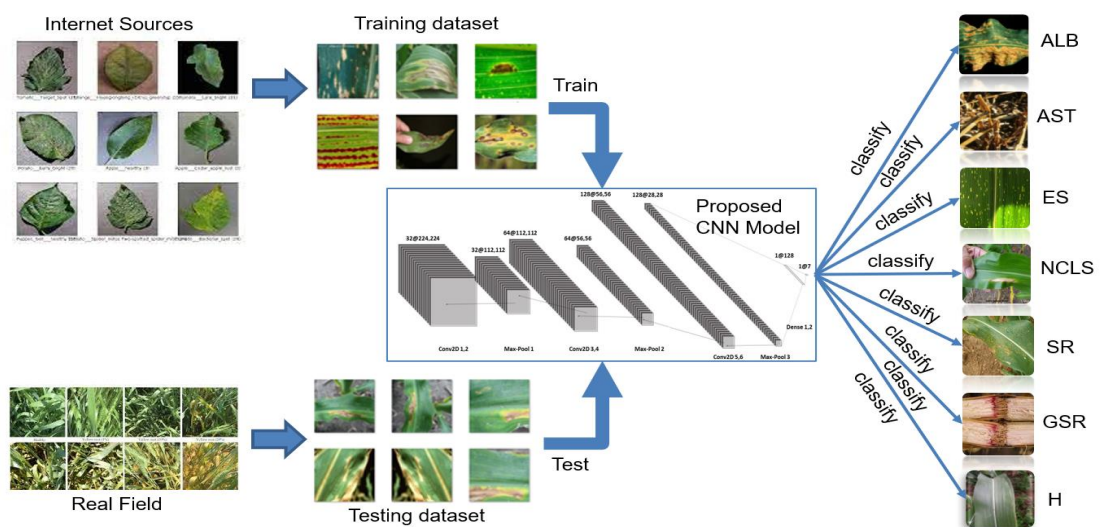


Fig 5.1. Performance evaluation process of the proposed model

### 5.2.1 Performance Evaluation

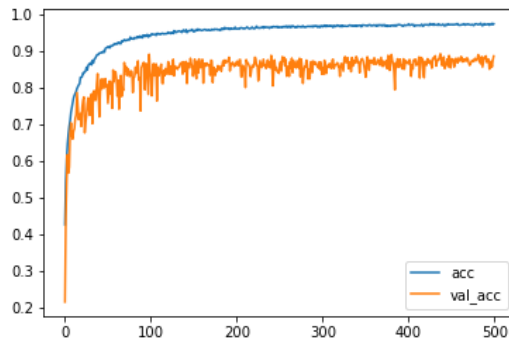
In this sub-section, the performance has observed by running the proposed model for different number of epochs. This process can allow to determine the best epoch value that can help to improve the model efficiency.

The experiment has carried out with different training and testing dataset split ratios. Initially, the experiment has conducted with a train and test dataset split ratio of 80:20 for the proposed model. To observe the performance, the new model has been trained and tested for different epochs like 35, 50, 100, 200, 300, and 500 with a batch size of 32. The performance evaluation results in terms of accuracies are shown in Table 5.1.

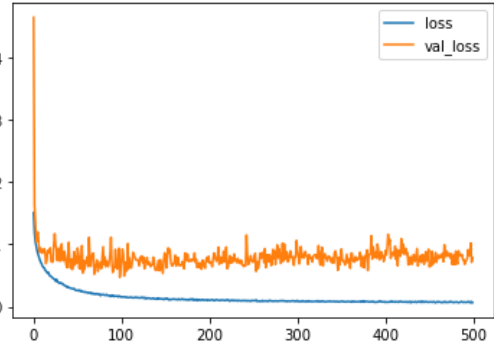
Table 5.1. Accuracy results of six different epochs

Train- Test ratio	Training and Testing Accuracies		
	# of epochs	Training Accuracy	Testing Accuracy
<b>80:20</b>	35	89.71%	70.64%
	50	91.29%	81.43%
	100	94.16%	83.42%
	200	95.71%	86.24%
	300	94.65%	83.46%
	500	97.51%	88.72%

The results mentioned in Table 5.1 has described that the classification accuracy obtained is high when the model is trained with dataset split ratio of 80:20 and with 500 epochs. Fig 5.2 shows the curves of training and validation accuracies including losses. Fig 5.3 shows the training/validation accuracy and loss curves of results mentioned in Table 5.1. The performance improvement of the proposed model over the number of epochs is shown in Fig 5.4.

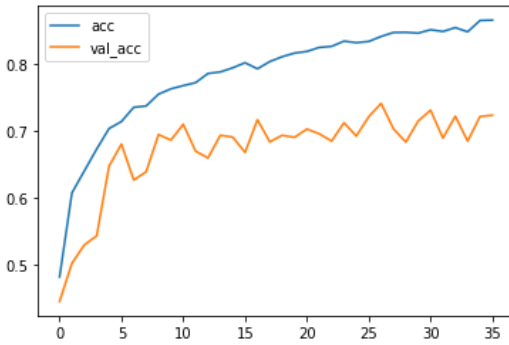


(a) Accuracies curves

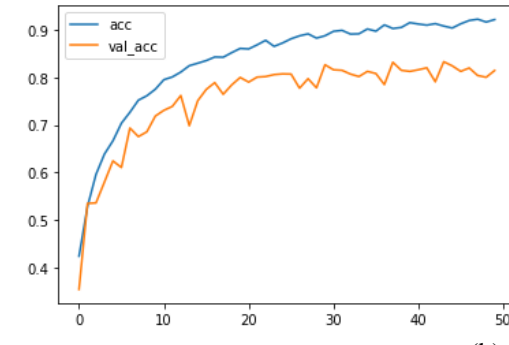
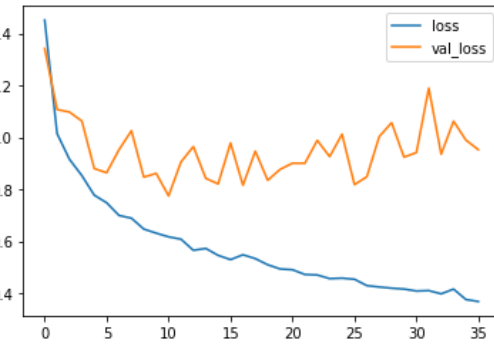


(b) Loss curves

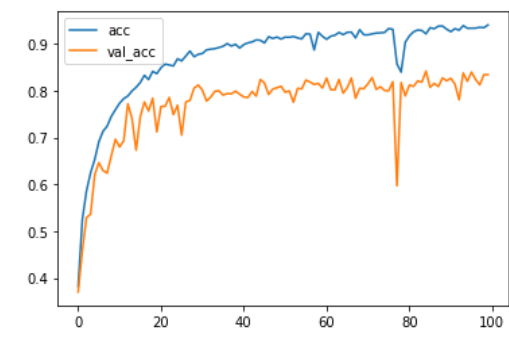
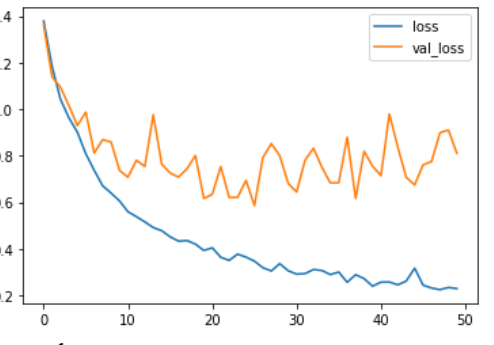
Fig 5.2. Accuracy and loss curves of the training and testing image sets



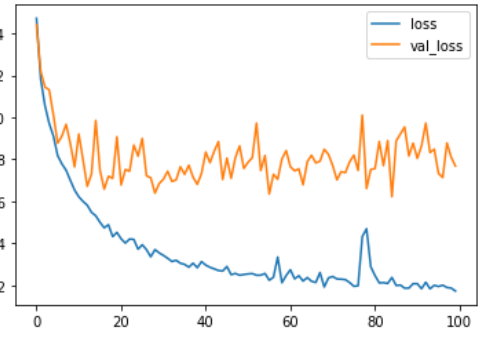
(a) 35 epochs



(b) 50 epochs



(c) 100 epochs



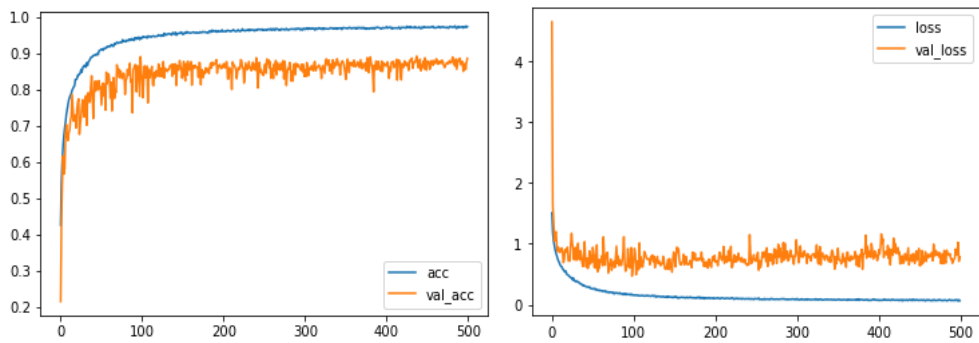
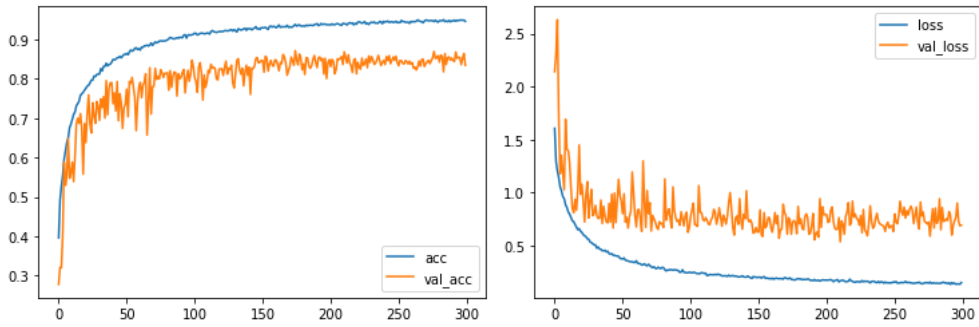
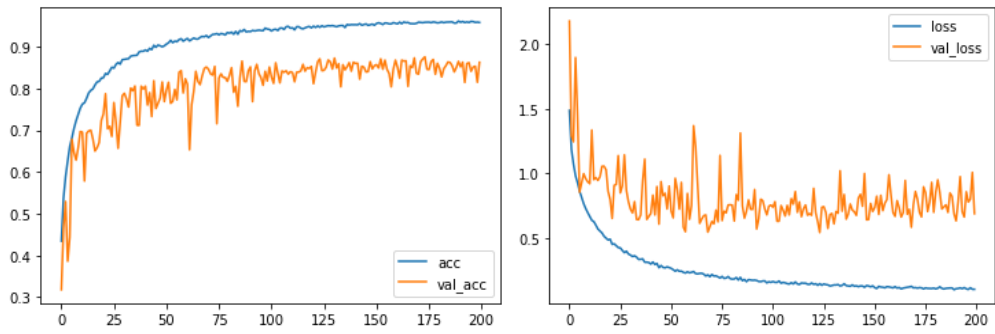


Fig 5.3. Training and Testing Accuracy and Loss Curves

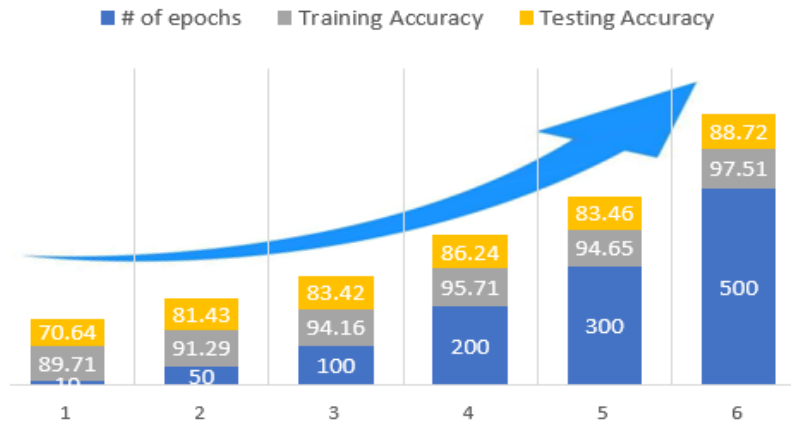


Fig 5.4. Performance improvement of the proposed model

### 5.3 Classification Performance

In this section, the evaluation of classification performance based on each disease class confusion matrix is presented. The confusion matrix provides a standard format for accuracy assessment by defining ‘n’ rows and ‘m’ columns using precision, recall, and F1-score. The number of class instances is represented by each row in the matrix, while the number of predicted class instances is represented by each column.

Table 5.2. Confusion matrix of seven diseases classification

<b>Class Index</b>	<b>Precision</b>	<b>Recall</b>	<b>F1-Score</b>	<b>Support</b>
<b>0</b>	88%	86%	87%	380
<b>1</b>	93%	100%	96%	380
<b>2</b>	81%	61%	69%	380
<b>3</b>	100%	93%	96%	380
<b>4</b>	90%	95%	93%	380
<b>5</b>	78%	87%	82%	380
<b>6</b>	91%	100%	95%	380
<b>Accuracy</b>			89%	2660
<b>Macro Avg</b>	89%	89%	88%	2660
<b>Weighted Avg</b>	89%	89%	88%	2660

Table 5.2 shows the precision, recall, and F1-score of seven maize crop disease classes. The top F1 scores obtained by the proposed model for disease type 1 (anthracnose stalk rot) and type 3 (gibberella stalk rot) is 96%. When compared with each class precision and recall value, the difference of type 0 disease (anthracnose leaf blight) is 88% ~ 86%. It describes that the model classified most images with type 0 disease as disease type 0. When compared with each class precision and recall value, the difference of type 2 disease (eyespot) is 81% ~ 61%. It describes that the model classified only few images with type 2 disease as disease type 2. The classification report in Fig 5.5 shows the confusion matrix with and without the normalization for results mentioned in Table 5.2.



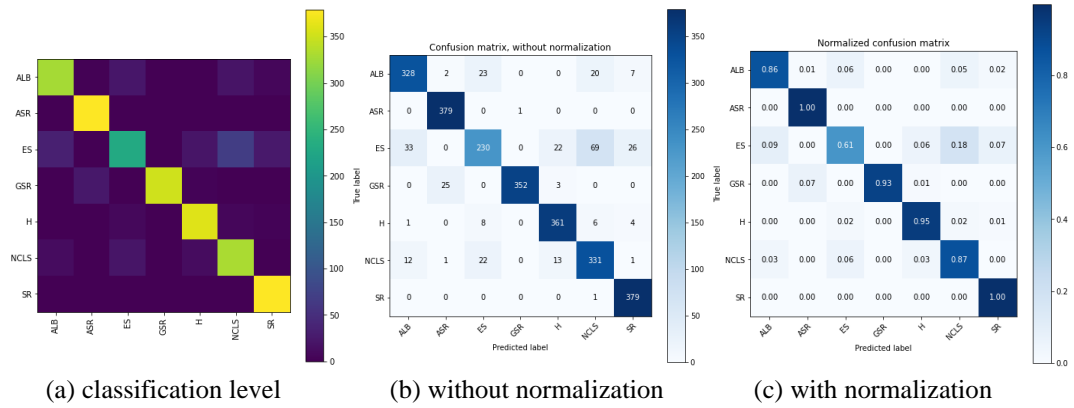


Fig 5.5. Confusion matrix representations

To assess the efficiency of the proposed model, a variety of experimental methods are used. To begin, additional training and testing dataset ratios are considered in the experimentation. The classification accuracy results for the proposed model are shown in Table 5.3 for three different train and test ratios as defined in Table 1. The class-wise performance values in Table 5.3 with 500 epochs is shown in Table 5.4.

Table 5.3. Classification results of different image set split ratios

Training-Testing Image set Ratio in %	Classification Accuracies	
	100 epochs	500 epochs
50-50	66%	72%
75-25	76%	81%
80-20	83%	89%

From Table 5.3, it is observed that the proposed model performed the classification problem well for the 80:20 dataset split ratio compared with split ratios 50:50 and 75:25. The comparative results are shown in Fig 5.6 and the class-wise accuracy values for different split ratios and 500 epochs are mentioned in Table 5.4.

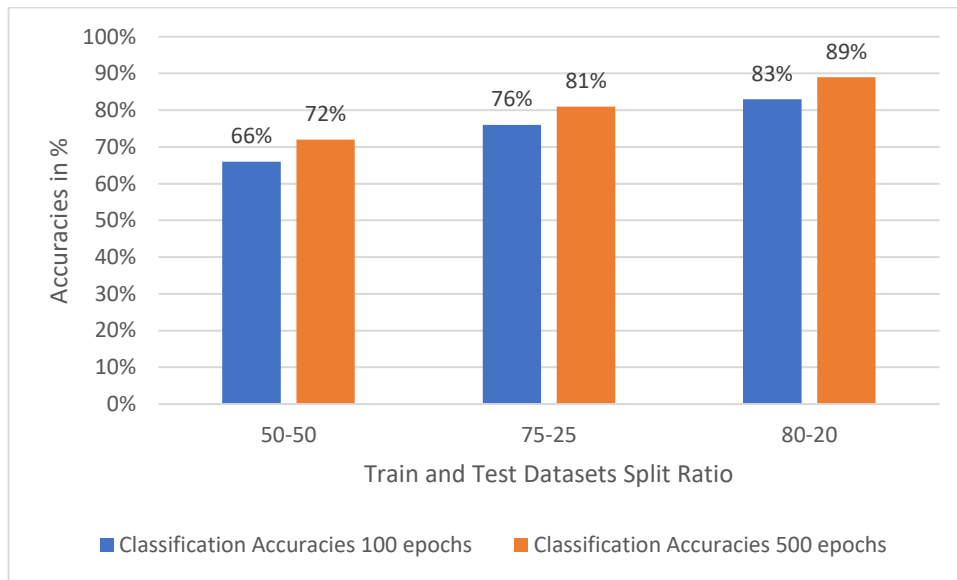


Fig 5.6. Classification accuracies with different image sets split ratios

Table 5.4. Class-wise classification accuracies

Image set Split Ratio in %	Classes	Classification Accuracy	Overall Average Classification Accuracy
50-50	ALB	61%	72%
	ASR	85%	
	ES	67%	
	GSR	90%	
	H	70%	
	NCLS	57%	
	SR	71%	
75-25	ALB	84%	81%
	ASR	95%	
	ES	63%	
	GSR	98%	
	H	72%	
	NCLS	67%	
	SR	86%	
	ALB	87%	
	ASR	96%	

80-20	ES	69	89%
	GSR	96	
	H	93	
	NCLS	82	
	SR	95	

The experimentation has so far been carried out with a 3 x 3 kernel scale. The proposed model is once again updated to increase classification accuracy by changing the kernel size from 3 x 3 to 5 x 5 and 7 x 7. Table 5.5 shows the results for the same dataset with various kernel sizes based on the confusion matrix. From Table 5.5, it is evident that both the kernel size 5 x 5 and 7 x 7 outperformed the 3 x 3 kernel size after evaluating for 100 epochs. The new model performed well with a 3% higher classification accuracy than the kernel size 3 x 3. Training the same model for 500 epochs, the kernel size 3 x 3 outperformed better than 5 x 5 and 7 x 7 kernel sizes. The performance evaluation results of the proposed model for different kernel sizes have presented in Table 5.5 and the same have been shown in Fig 5.7.

Table 5.5. Classification accuracies with three different kernel sizes

Training and Testing Ratio in %	# of epochs	Classification Accuracies		
		Kernel Size 3 x 3	Kernel Size 5 x 5	Kernel Size 7 x 7
50-50	100	66%	68%	66%
75-25	100	76%	75%	77%
80-20	100	83%	85%	85%
80-20	500	89%	85%	87%

Initially, the evaluation process of the proposed model has been carried out with different data split ratios and kernel sizes for 100 epochs. Later, the process has been continued for 500 epochs with a dataset split ratio of 80:20 and different kernel sizes to observe the improvement in accuracy. The green bars in figure 5.7 are showing the results observed during the experiment.

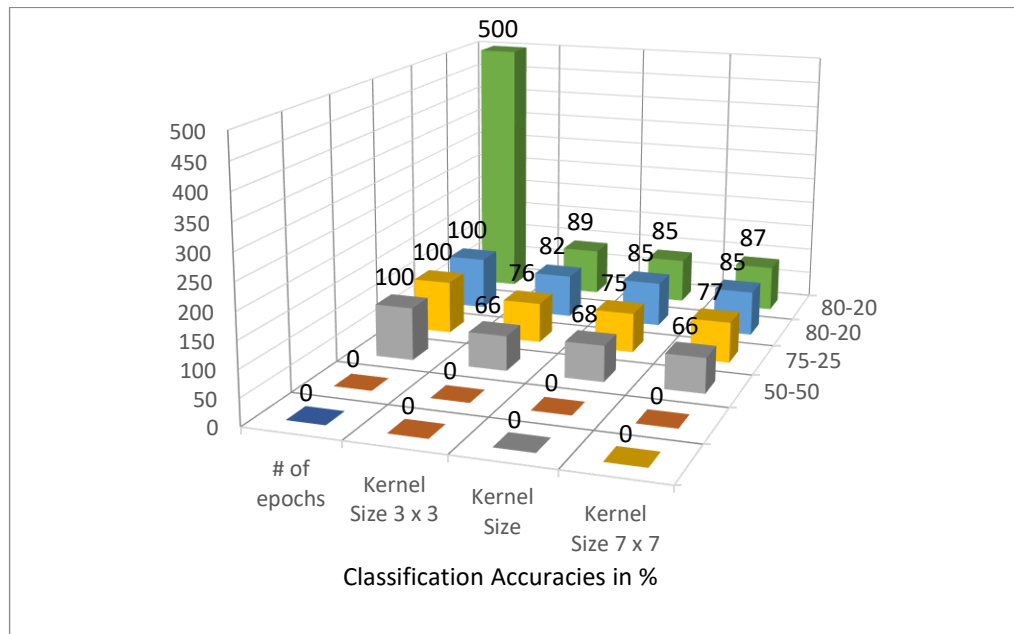


Fig 5.7. Performances measures with different kernel sizes

#### 5.4 Performance Analysis using selective hyperparameters

The selection of appropriate hyperparameters such as the input image size, optimizers, learning rate, kernel sizes, the number of hidden layers, batch size, number of epochs are the major barriers to apply CNN and requires considerable skill and experience. Internal dependencies in these hyper-parameters make tuning them more costly and time consuming. There is no standard procedure or technique that can help in selecting the suitable values for hyperparameters. The selection process should be done on the trial-and-error basis only. Through experimental approach it is observed that the inter dependencies among the hyperparameters can affect the performance of any CNN model through experimental approach. A total of fourteen experiments were conducted on selecting the suitable hyperparameters. During the experiments, the model is trained and tested for 100 epochs.

The hyperparameters used while training the model are presented in Table 5.6. During the model evaluation, the states in which the network presented the accuracy and loss for the training and testing datasets were saved. Later, the saved results were evaluated in terms of precision, recall and F1 score.

Table 5.6. List of hyperparameters for tuning the model

<b>Parameter</b>	<b>Value</b>
Image size	112 x 112, 168 x 168, and 224 x 224
Batch size	32, 64, and 128
Number of epochs	50, 100, and 200
Optimizers	Adam, RMSprop, and SGD
Learning rate	0.001, 0.01, and 0.005
Kernel size	3 x 3, 5 x 5, and 7 x 7
Number of hidden layers	15, 17, and 19
Loss function	Cross-Entropy

#### 5.4.1 Image Size

The resolution of the input images may impact the performance of image detection and classification. To understand the truth behind this statement, the proposed disease detection and classification CNN model has been trained by considering the input images with different resolutions for analysis. The model is trained on the images with varied resolutions like 112 x 112, 168 x 168 and 224 x 224 and evaluated its performance. The performance measure of the proposed model for 100 epochs, 32 batch size, 19 hidden layers and with three different image sizes are shown in Table 5.7.

Table 5.7. Accuracies with different image sizes

<b>Image Size</b>	<b>Kernel Size</b>	<b>Optimizer</b>	<b>Learning Rate</b>	<b>Training Accuracy</b>	<b>Testing Accuracy</b>
<b>112 x 112</b>	3 x 3	Adam	0.001 default	94.83%	85.83%
<b>168 x 168</b>	3 x 3	Adam	0.001 default	94.07%	84.66%
<b>224 x 224</b>	5 x 5	Adam	0.001 default	94.46%	85.26%

As shown in Fig 5.8 (a), the training process of the model is stable upto 10 epochs and started convergence at about 25 epochs and achieved an accuracy of 94.83% for an

image size 112 x 112. The training process is stable upto 5 epochs and started convergence at about 18 epochs and achieved an accuracy of 94.07% for an image size 168 x 168. The training process of the model is stable upto 10 epochs and started convergence at about 16 epochs and finally achieved an accuracy of 94.46% for an image size 224 x 224. As shown in Fig 5.8 (b), the model tested on input images of size 112 x 112 started to converge after 21 epochs and finally achieved an optimal classification performance of 85.83%. The model tested on input images of size 168 x 168 started to converge after 21 epochs and finally achieved an optimal classification performance of 84.66%. The model tested on input images of size 224 x 224 started to converge after 15 epochs and finally achieved an optimal classification performance of 85.26%.

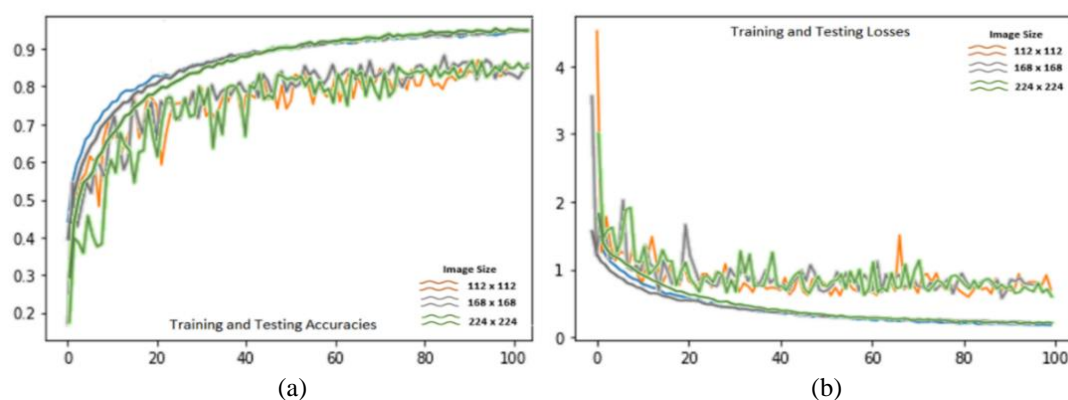


Fig 5.8. Accuracy and loss curves with different image sizes

Fig 5.9(a) shows the confusion matrix associated with results obtained with input image size of 112 x 112. The classification results of ASR, GSR, H, and SR were consistent for four disease classes with 99%, 97%, 90%, and 94% respectively except ALB, ES, and NCLS. The results presented a considerable sum of classification errors for disease classes ALB and NCLS with 20% and 19% respectively. Fig 5.9(b) shows the confusion matrix associated with results obtained with input image size of 168 x 168. The classification results of ASR, GSR, H, and SR were consistent for four disease classes with 99%, 93%, 93%, and 98% respectively except ALB, ES, and NCLS. The results presented a considerable sum of classification errors for disease class ALB with 24%. Fig 5.9(c) shows the confusion matrix associated with results obtained with input image size of 224 x 224. The classification results of ASR, H, and SR were consistent for three disease classes with 100%, 91%, and 94% respectively except ALB, ES, GSR, and

NCLS. The results presented a considerable sum of classification errors for disease classes GSR and NCLS with 24% and 26%. Fig 5.10 shows the performance measures chart with different image sizes.

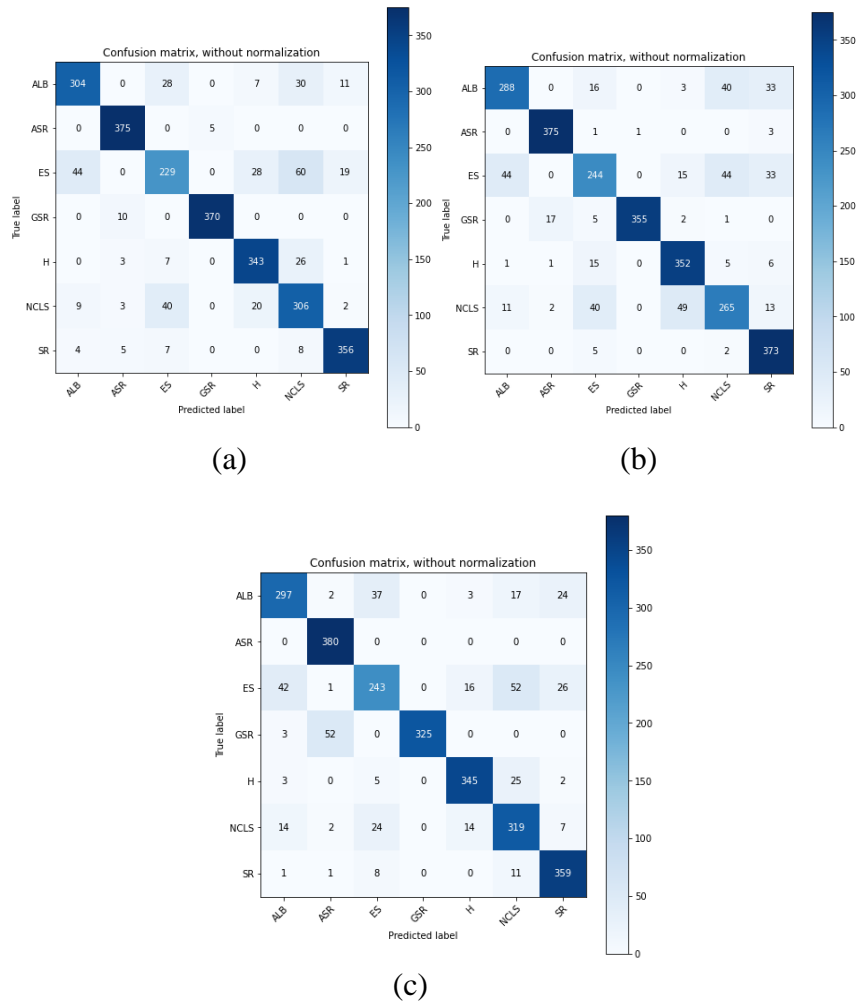


Fig 5.9. Confusion matrices with different image sizes

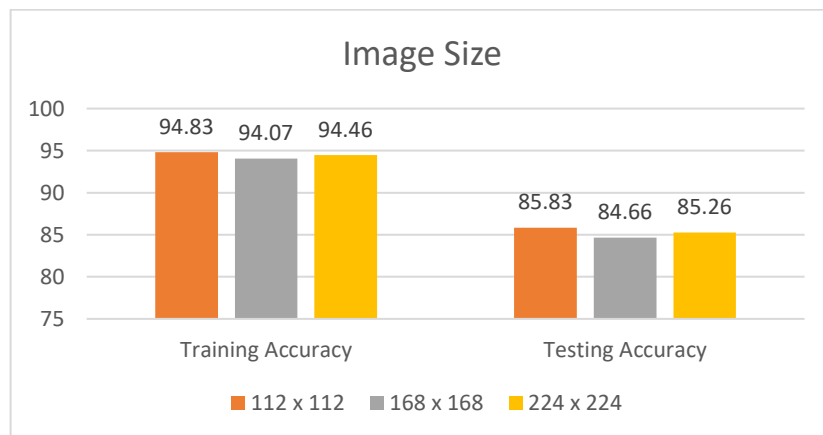


Fig 5.10. Comparison of model performance with different image sizes

### 5.4.2 Batch Size

The batch size hyperparameter can also influence the performance of a CNN model. This parameter supports the neural network to learn the image features more quickly and stable the learning process. The proposed CNN model is trained by feeding it with different batch sizes to determine its performance. The model is trained with different batch sizes like 32, 64, and 128 for analysis. The performance measure of the proposed model for 3 x 3 kernel size, 100 epochs, 19 hidden layers and with three different batch sizes are shown in Table 5.8.

Table 5.8. Accuracies with different batch sizes

<b>Image Size</b>	<b>Batch Size</b>	<b>Optimizer</b>	<b>Learning Rate</b>	<b>Training Accuracy</b>	<b>Testing Accuracy</b>
<b>112 x 112</b>	32	Adam	0.001 default	94.83%	85.83%
<b>112 x 112</b>	64	Adam	0.001	95.19%	80.23%
<b>112 x 112</b>	128	Adam	0.001 default	95.98%	83.91%

As shown in Fig 5.11 (a) the training process of the model is stable upto 10 epochs and started convergence at about 25 epochs and finally achieved an accuracy of 94.83% for batch size 32. The training process of the model is stable upto 5 epochs and started convergence at about 18 epochs and finally achieved an accuracy of 95.19% for batch size 64. The training process of the model is stable upto 10 epochs and started convergence at about 16 epochs and finally achieved an accuracy of 95.98%. As shown in Fig 5.11 (b), the model tested with batch size 32 started to converge after 21 epochs and finally achieved an optimal classification performance of 85.83%. The model tested with batch size 64 started to converge after 21 epochs and finally achieved an optimal classification performance of 80.23%. The model tested with batch size 128 started to converge after 15 epochs and finally achieved an optimal classification performance of 83.91%.



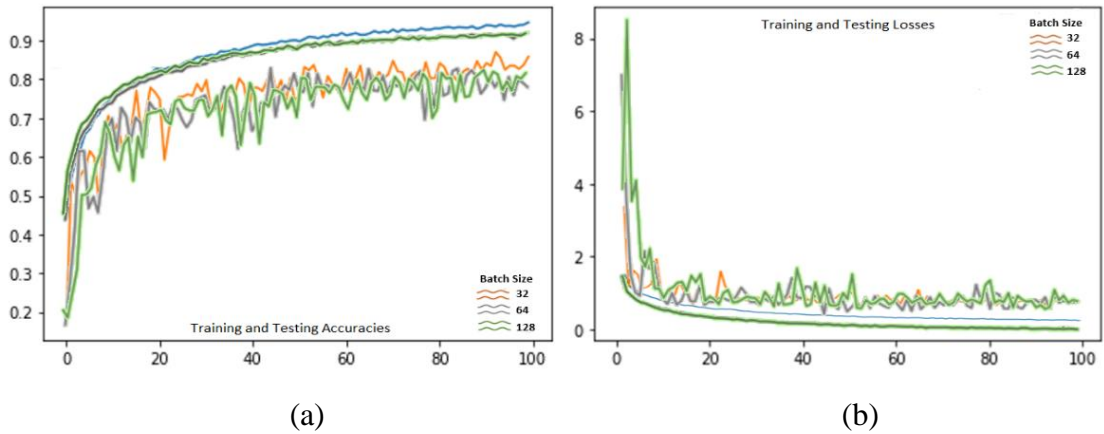
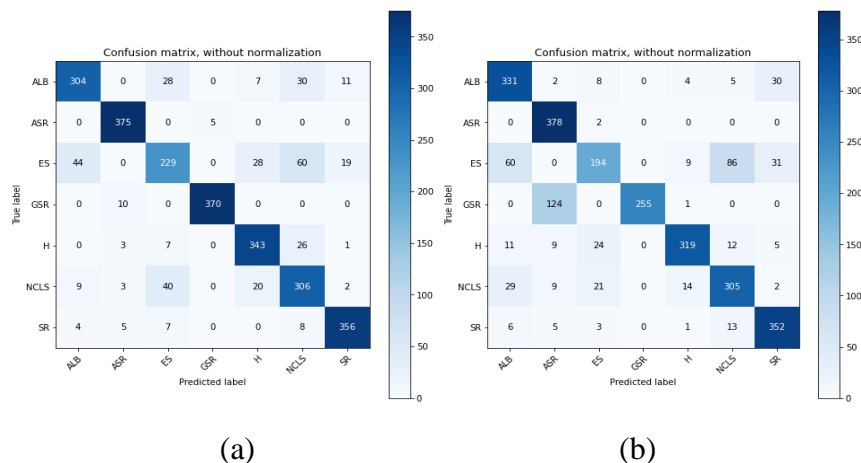
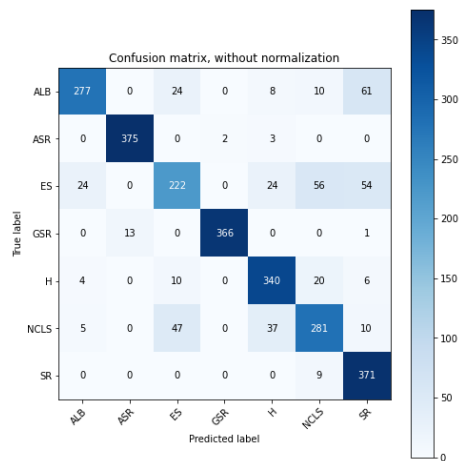


Fig 5.11. Accuracy and loss curves with different batch sizes

Fig 5.12(a) shows the confusion matrix associated with results obtained for batch size 32. The classification results of ASR, GSR, H, and SR were consistent for four disease classes with 99%, 97%, 90%, and 94% respectively except ALB, ES, and NCLS. The results presented a considerable sum of classification errors for disease classes ALB and NCLS with 20% and 19% respectively. Fig 5.12(b) shows the confusion matrix associated with results obtained for batch size 64. The classification results of ASR, and SR were consistent for two disease classes with 99% and 93% respectively except ALB, GSR, H, and NCLS. The results presented a considerable sum of classification errors for disease classes ALB, H, and NCLS with 13%, 16%, and 20%. Fig 5.12(c) shows the confusion matrix associated with results obtained with batch size 128. The classification results of ASR, GSR, and SR were consistent for three disease classes with 99%, 96%, and 98% respectively except ALB, ES, H, and NCLS. The results presented a considerable sum of classification errors for disease class H with 11%. Fig 5.13 shows the performance measures chart with different batch sizes.





(c)

Fig 5.12. Confusion matrices with different batch sizes

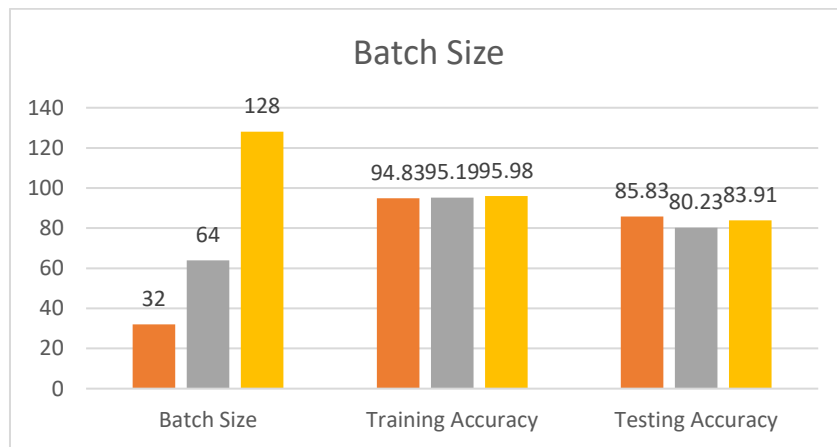


Fig 5.13. Comparison of model performance with different batch sizes

### 5.4.3 Number of Epochs

The performance of a CNN model can be tuned using one of the parameters like 'epoch'. The number of epochs can definitely influence the performance of the model. The proposed model is trained with 50, 100, and 200 optimal number of epochs and the results were analysed to determine the learning capability of a model. The performance measure of the proposed model for 3 x 3 kernel size, 32 batch size, 19 hidden layers and with three different number of epochs are shown in Table 5.9.

Table 5.9. Accuracies with different number of epochs

Image Size	Optimizer	Learning Rate	# of epochs	Training Accuracy	Testing Accuracy
112 x 112	Adam	0.001 default	50	90.15%	74.85%
112 x 112	Adam	0.001 default	100	94.83%	85.83%
112 x 112	Adam	0.001 default	200	96.16%	87.44%

As shown in Fig 5.14 (a), the training process of the model is stable upto 10 epochs and started convergence at about 25 epochs and finally achieved an accuracy of 90.15% with 50 epochs. The training process of the model is stable upto 5 epochs and started convergence at about 18 epochs and finally achieved an accuracy of 94.83% with 100 epochs. The training process of the model is stable upto 10 epochs and started convergence at about 16 epochs and finally achieved an accuracy of 96.16% with 200 epochs. As shown in Fig 5.14(b), the model tested with 50 epochs started to converge after 21 epochs and finally achieved an optimal classification performance of 74.85%. The model tested for 100 epochs started to converge after 21 epochs and finally achieved an optimal classification performance of 85.83%. The model tested with 200 epochs started to converge after 15 epochs and finally achieved an optimal classification performance of 87.44%.

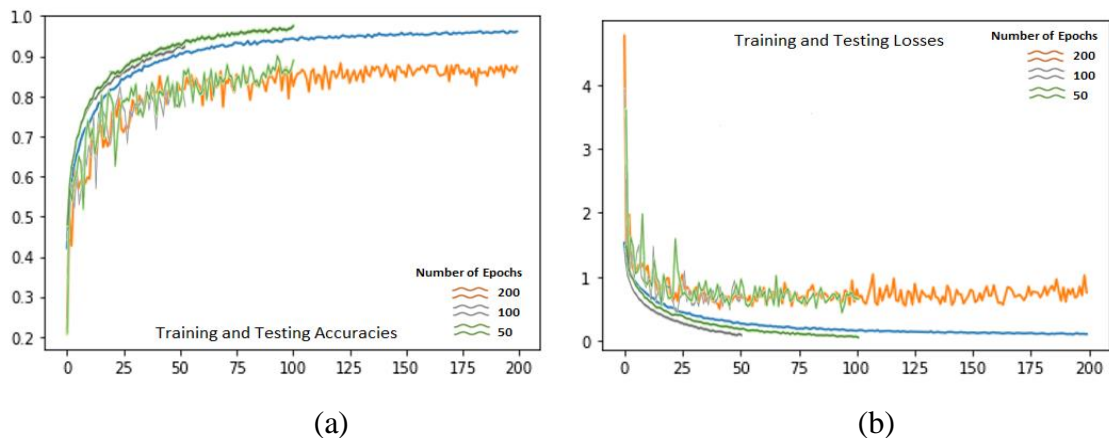


Fig 5.14. Accuracy and loss curves with different number of epochs

Fig 5.15(a) shows the confusion matrix associated with results obtained after running the model with 50 epochs. The classification result of GSR is consistent for one disease

class with 92% except ASR, ALB, ES, H, NCLS, and GSR. The results presented a considerable sum of classification errors for disease classes ASR, H, NCLS, and SR with 17%, 11%, 21%, and 15% respectively. Fig 5.15(b) shows the confusion matrix associated with results obtained after running the model with 100 epochs. The classification results of ASR, GSR, H, and SR were consistent for four disease classes with 99%, 97%, 90%, and 94% respectively except ALB, ES, and NCLS. The results presented a considerable sum of classification errors for disease classes ALB with 20% and 19% respectively. Fig 5.15(c) shows the confusion matrix associated with results obtained after running the model with 200 epochs. The classification results of ASR, GSR, H, and SR were consistent for four disease classes with 98%, 99%, 96%, and 94% respectively except ALB, ES, and NCLS. The results presented a considerable sum of classification errors for disease classes ALB and NCLS with 16% and 13%. Fig 5.16 shows the performance measures chart with different epochs.

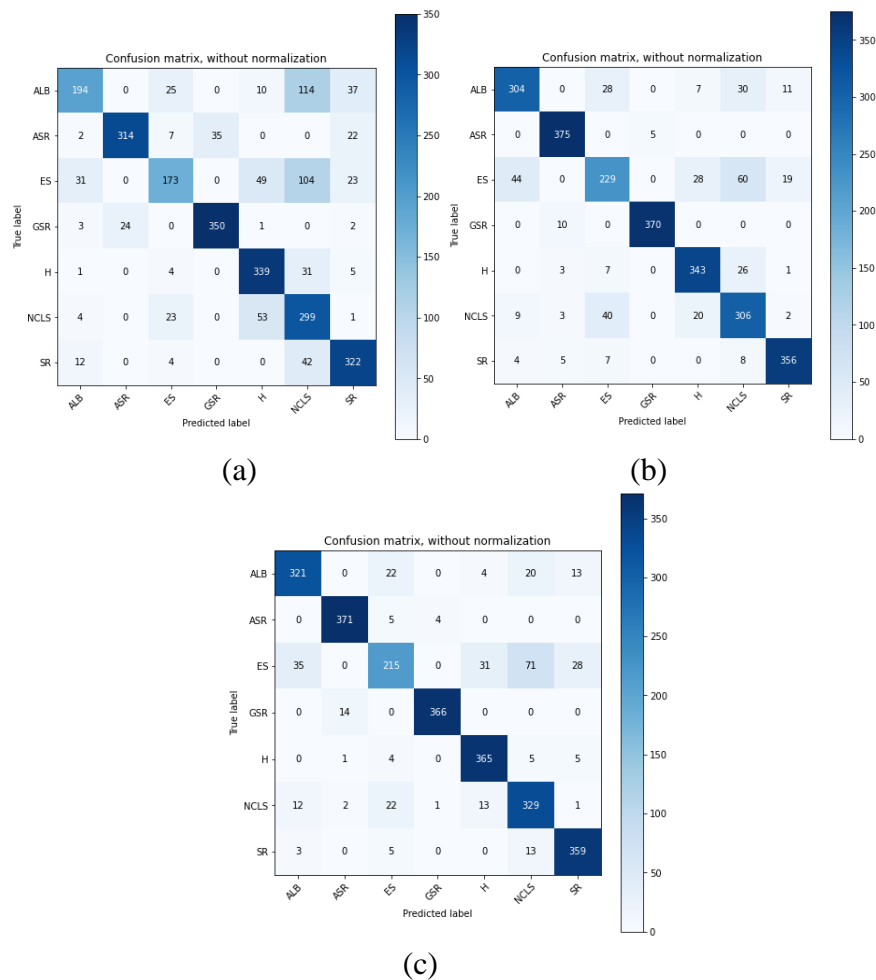


Fig 5.15. Confusion matrices with different number of epochs

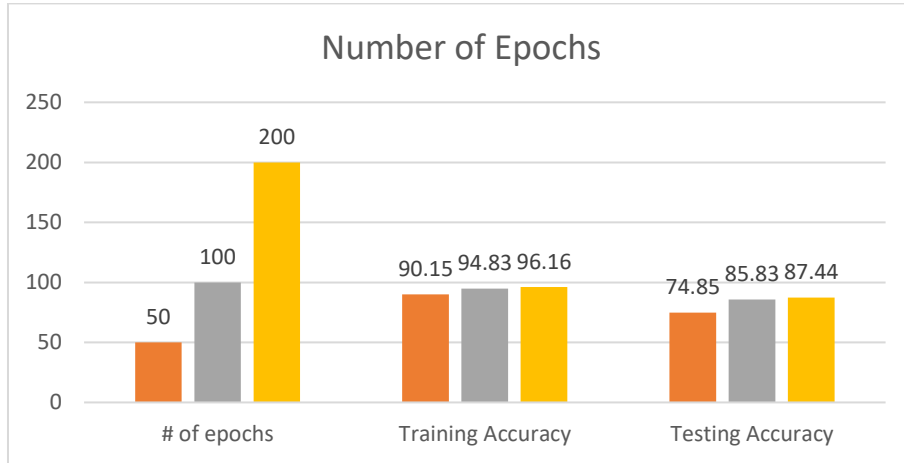


Fig 5.16. Comparison of model performance with different number of epochs

#### 5.4.4 Optimizers

The selection of optimization algorithm can show the difference in terms of learning time for any neural network model. The most popular optimization algorithms that are supported by deep learning like Stochastic Gradient Descent (SGD), RMSprop, and Adam are applied to the proposed model. The performance measure of the proposed model for 3 x 3 kernel size, 32 batch size, 100 epochs, 19 hidden layers and with three different optimizers are shown in Table 5.10.

Table 5.10. Accuracies with different optimizers

Image Size	Optimizer	Learning Rate	Training Accuracy	Testing Accuracy
112 x 112	Adam	0.001 default	94.83%	85.83%
112 x 112	RMSprop	0.001 default	90.83%	81.95%
112 x 112	SGD	0.01 default	89.99%	79.66%

As shown in Fig 5.17(a), the model tested using Adam optimizer started to converge after 21 epochs and finally achieved an optimal classification performance of 85.83%. The model tested using RMSprop optimizer started to converge after 21 epochs and finally achieved an optimal classification performance of 81.95%. The model tested using SGD optimizer started to converge after 15 epochs and finally achieved an optimal classification performance of 79.66%. As shown in Fig 5.17 (b), the training

process of the model is stable upto 10 epochs and started convergence at about 25 epochs and achieved an accuracy of 94.83% with Adam optimizer. The training process of the model is stable upto 5 epochs and started convergence at about 18 epochs and achieved an accuracy of 90.83% with RMSprop optimizer. The training process of the model is stable upto 10 epochs and started convergence at about 16 epochs and achieved an accuracy of 89.99% with SGD optimizer.

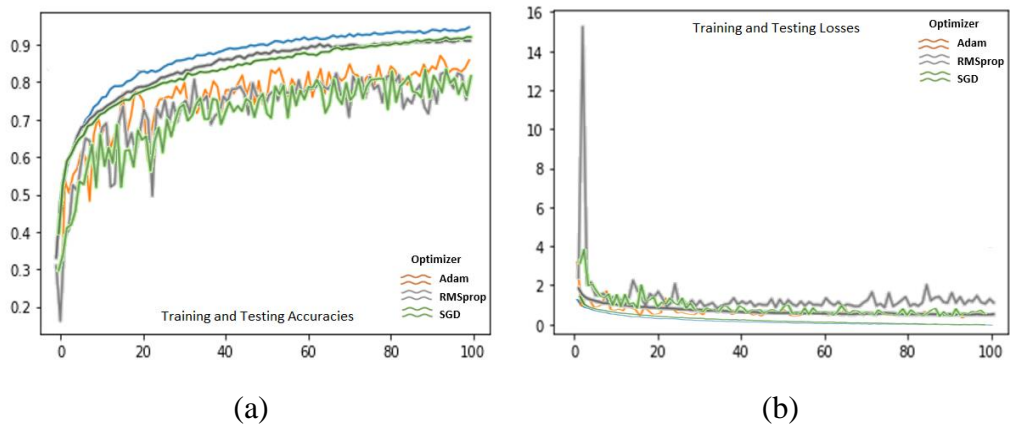
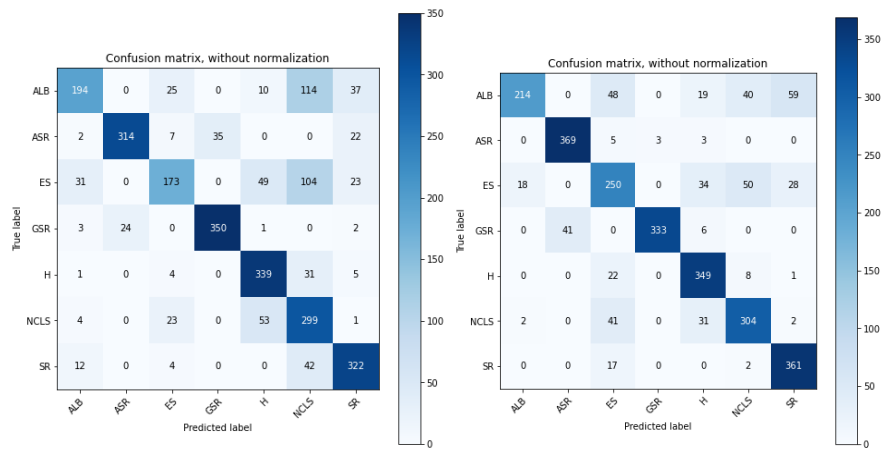


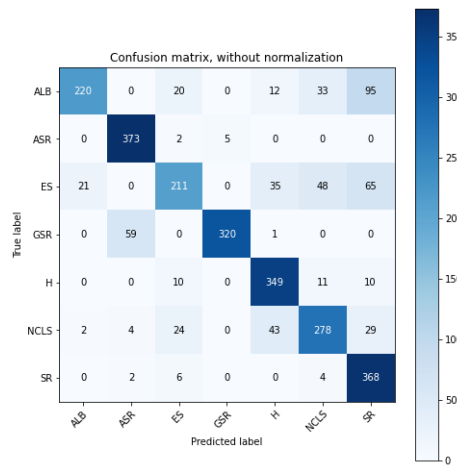
Fig 5.17. Accuracy and loss curves with different optimizers

Fig 5.18(a) shows the confusion matrix associated with results obtained using Adam optimizer. The classification result is consistent for one disease class GSR with 92% except ALB, ASR, ES, H, NCLS, and SR. The results presented a considerable sum of classification errors for disease classes ASR, H and SR with 17%, 11% and 15% respectively. Fig 5.18(b) shows the confusion matrix associated with results obtained using RMSprop optimizer. The classification results of ASR, H, and SR were consistent for three disease classes with 97%, 92%, and 95% respectively except ALB, ES, GSR and NCLS. The results presented a considerable sum of classification errors for disease class GSR and NCLS with 12% and 20% respectively. Fig 5.18(c) shows the confusion matrix associated with results obtained using SGD optimizer. The classification results of ASR, H, and SR were consistent for three disease classes with 98%, 92%, and 97% respectively except ALB, ES, GSR, and NCLS. The results presented a considerable sum of classification errors for disease class GSR with 26%. Fig 5.19 shows the performance measure chart with different optimizers.



(a)

(b)



(c)

Fig 5.18. Confusion matrices with different optimizers

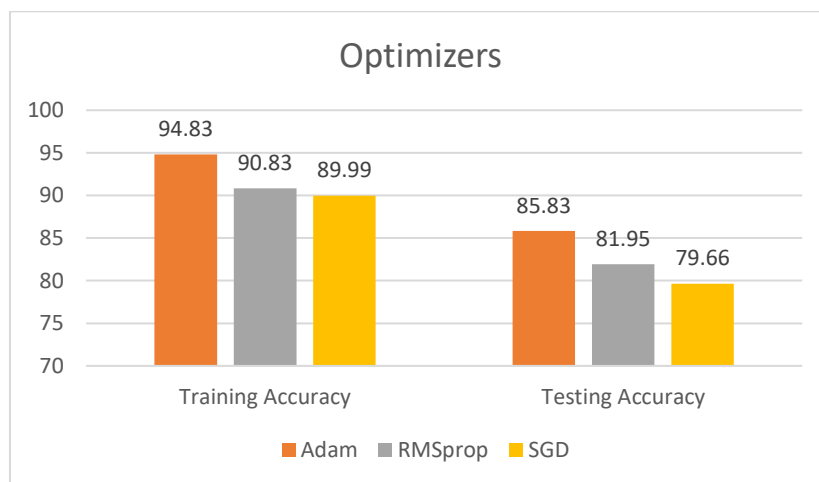


Fig 5.19. Comparison of model performance with different optimizers

### 5.4.5 Learning Rate

The adjustment of the learning rate parameter value can also impact the performance of classification. The learning rate hyperparameter can control the change in model each time when the weights are updated for each estimated error. Here, the Adam optimizer is selected and experiments are conducted by changing the value of learning rate to 0.001, 0.01, and 0.005. The performance measure of the proposed model for 3 x 3 kernel size, 32 batch size, 100 epochs, 19 hidden layers and with three different learning rates are shown in Table 5.11.

Table 5.11. Accuracies with different learning rates

<b>Image Size</b>	<b>Optimizer</b>	<b>Learning Rate</b>	<b>Training Accuracy</b>	<b>Testing Accuracy</b>
<b>112 x 112</b>	Adam	0.001 default	94.83%	85.83%
<b>112 x 112</b>	Adam	0.01	75.26%	76.88%
<b>112 x 112</b>	Adam	0.005	89.22%	81.50%

As shown in Fig 5.20(a), the training process of the model is stable upto 10 epochs and started convergence at about 25 epochs and finally achieved an accuracy of 94.83% with learning rate 0.001. The training process of the model is stable upto 5 epochs and started convergence at about 18 epochs and finally achieved an accuracy of 75.26% with learning rate 0.01. The training process of the model is stable upto 10 epochs and started convergence at about 16 epochs and finally achieved an accuracy of 89.22% with learning rate 0.005. As shown in Fig 5.20(b), the model tested using Adam optimizer and learning rate of 0.001 started to converge after 21 epochs and finally achieved an optimal classification performance of 85.83%. The model tested using Adam optimizer and learning rate of 0.01 started to converge after 21 epochs and finally achieved an optimal classification performance of 76.88%. The model tested using Adam optimizer and learning rate of 0.005 started to converge after 15 epochs and finally achieved an optimal classification performance of 81.50%.



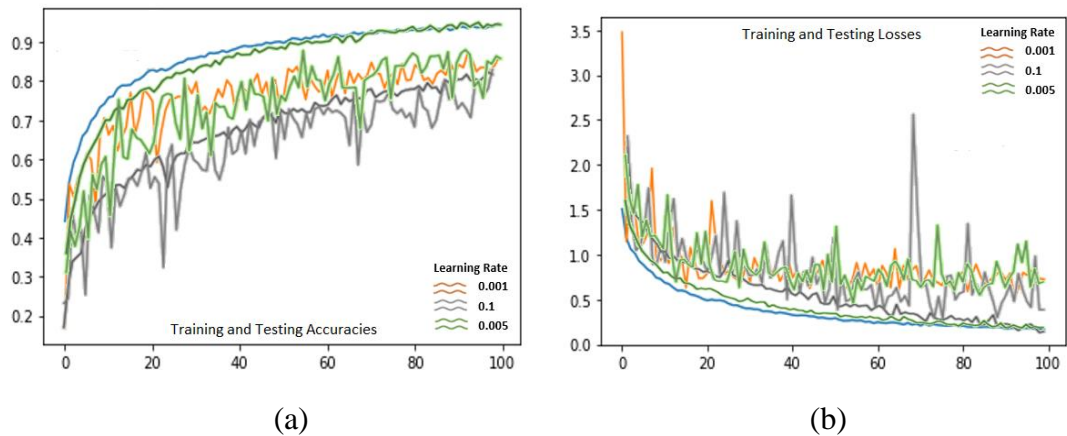
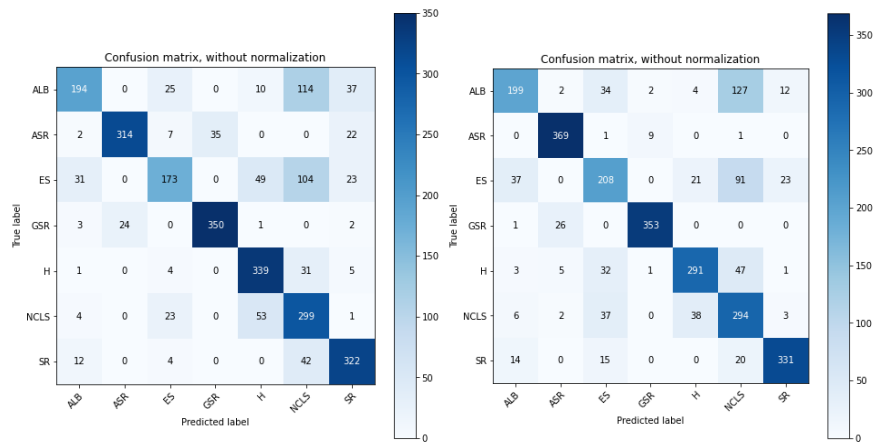


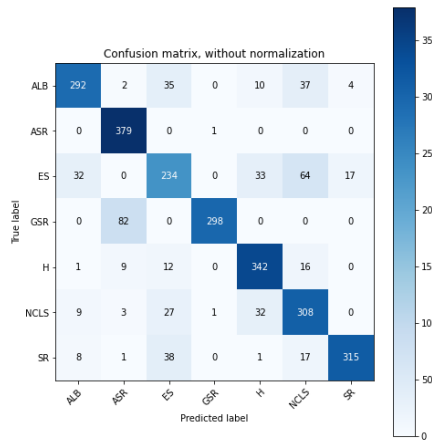
Fig 5.20. Accuracy and loss curves with different learning rate

Fig 5.21(a) shows the confusion matrix associated with results obtained using Adam optimizer and learning rate of 0.001. The classification results of GSR is consistent for one disease class with 92% except ASR, ALB, ES, H, NCLS and SR. The results presented a considerable sum of classification errors for disease classes ASB, H and SR with 17%, 11% and 15% respectively. Fig 5.21(b) shows the confusion matrix associated with results obtained using Adam optimizer and learning rate of 0.01. The classification results of ASR and GSR were consistent for two disease classes with 97% and 93% respectively except ALB, ES, H, NCLS, and GS. The results presented a considerable sum of classification errors for disease class H, NCLS and SR with 23%, 23%, and 13%. Fig 5.21(c) shows the confusion matrix associated with results obtained using Adam optimizer and learning rate of 0.005. The classification results of ASR and H were consistent for two disease classes with 100% and 90% respectively except ALB, ES, GSR, NCLS and SR. The results presented a considerable sum of classification errors for disease classes GSR, NCLS, and SR with 19% and 17%. Fig 5.21 shows the confusion matrix with and without normalization and Fig 5.22 shows the performance measure chart with different learning rates.



(a)

(b)



(c)

Fig 5.21. Confusion matrices with different learning rates

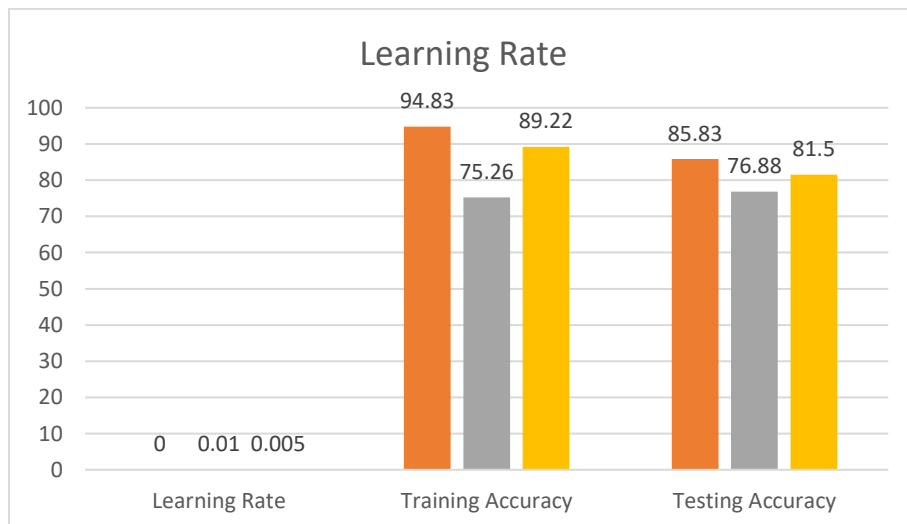


Fig 5.22. Comparison of model performance with different learning rate

#### 5.4.6 Kernel Size

Kernel size manages to reduce the model weights while learning deeper. A fully connected layer connects each output in terms of kernel size. Even the number of weights depends on the kernel size. Here, the experimentation is conducted with three different kernel sizes like 3 x 3, 5 x 5, and 7 x 7 to analyse the model performance. The performance measure of the proposed model for 32 batch size, 100 epochs, 19 hidden layers and with three different kernel sizes is shown in Table 5.12.

Table 5.12. Accuracies with different kernel sizes

<b>Image Size</b>	<b>Kernel Size</b>	<b>Optimizer</b>	<b>Learning Rate</b>	<b>Training Accuracy</b>	<b>Testing Accuracy</b>
<b>112 x 112</b>	3 x 3	Adam	0.001 default	94.83%	85.83%
<b>224 x 224</b>	5 x 5	Adam	0.001 default	94.46%	85.26%
<b>224 x 224</b>	7 x 7	Adam	0.001 default	94.58%	84.59%

As shown in Fig 5.23(a), the training process of the model is stable upto 10 epochs and started convergence at about 25 epochs and finally achieved an accuracy of 94.83% with kernel size 3 x 3. The training process of the model is stable upto 5 epochs and started convergence at about 18 epochs and finally achieved an accuracy of 94.46% with kernel 5 x 5. The training process of the model is stable upto 10 epochs and started convergence at about 16 epochs and finally achieved an accuracy of 94.58% with kernel size 7 x7. As shown in Fig 5.23 (b), the model tested using kernel size of 3 x 3 started to converge after 21 epochs and finally achieved an optimal classification performance of 85.83%. The model tested on input images using kernel size of 5 x 5 started to converge after 21 epochs and finally achieved an optimal classification performance of 85.26%. The model tested on input images using kernel size of 7 x 7 started to converge after 15 epochs and finally achieved an optimal classification performance of 84.59%.

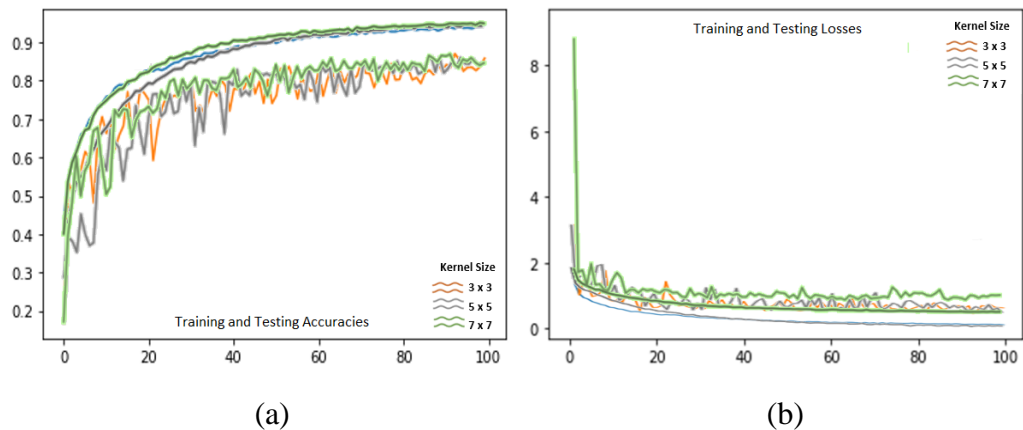
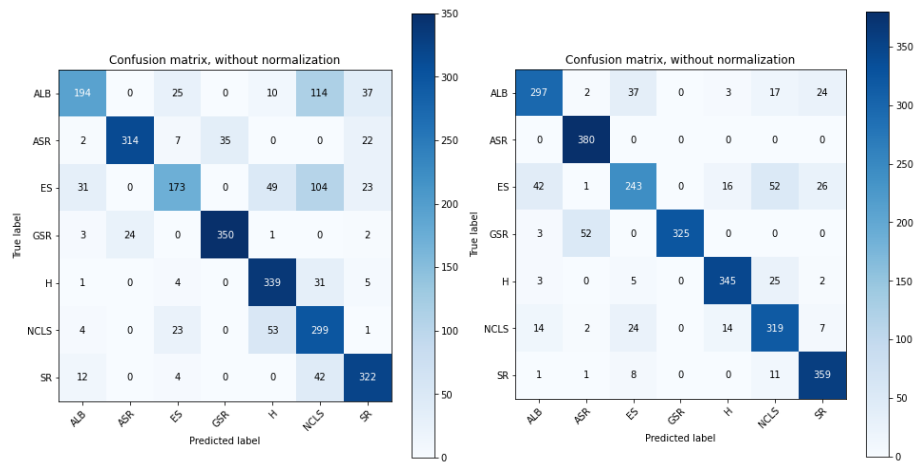


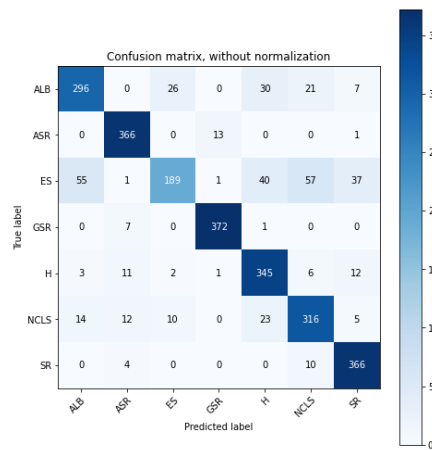
Fig 5.23. Accuracy and loss curves with different kernel size

Fig 5.24(a) shows the confusion matrix associated with results obtained using kernel size of 3 x 3. The classification result of GSR is consistent for one disease classes with 92% except ASR, ALB, ES, H, NCLS and SR. The results presented a considerable sum of classification errors for disease classes ASR, H and SR with 17%, 11% and 15% respectively. Fig 5.24(b) shows the confusion matrix associated with results obtained using kernel size of 5 x 5. The classification results of ASR, H and SR were consistent for three disease classes with 100%, 91% and 94% respectively except ALB, ES, GSR and NCLS. The results presented a considerable sum of classification errors for disease class ALB, GSR and NCLS with 22%, 14%, and 16%. Fig 5.24(c) shows the confusion matrix associated with results obtained using kernel size of 7 x 7. The classification results of ASR, GSR, H, and SR were consistent for four disease classes with 96%, 98%, 91% and 96% respectively except ALB, ES and NCLS. The results presented a considerable sum of classification errors for disease classes ALB and NCLS with 22% and 17%. Fig 5.25 shows the performance measure chart for different kernel size.



(a)

(b)



(c)

Fig 5.24. Confusion matrices with different kernel sizes

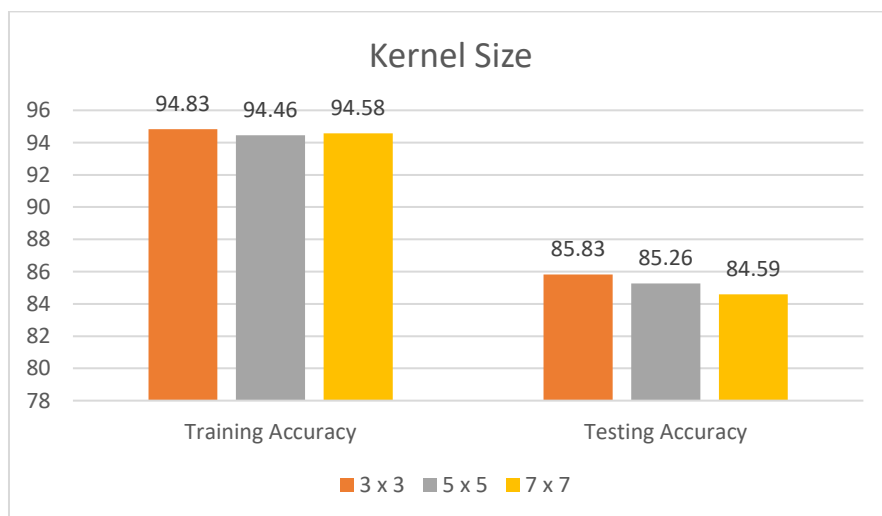


Fig 5.25. Comparison of model performance with different kernel sizes

#### 5.4.7 Number of hidden layers

Few numbers of hidden layers would work if the input is less complex and fewer features. To obtain an optimal solution a higher number of layers need to be selected. To determine the required number of layers and obtain better classification, the experimentation is conducted with 15, 17, and 19 layers. The performance evaluation of the proposed model for 3 x 3 kernel size, 32 batch size, 100 epochs and with three different hidden layers are shown in Table 5.13.

Table 5.13. Accuracies with different number of hidden layers

<b>Image Size</b>	<b>Optimizer</b>	<b>Learning Rate</b>	<b># of hidden layers</b>	<b>Training Accuracy</b>	<b>Testing Accuracy</b>
112 x 112	Adam	0.001 default	15	89.75%	72.07%
112 x 112	Adam	0.001 default	17	90.92%	78.72%
112 x 112	Adam	0.001 default	19	94.83%	85.83%

As shown in Fig 5.26(a), the training process of the model are stable upto 10 epochs and started convergence at about 25 epochs and finally achieved an accuracy of 89.75% with 15 hidden layers. The training process of the model are stable upto 5 epochs and started convergence at about 18 epochs and finally achieved an accuracy of 90.92% with 17 hidden layers. The training process of the model are stable upto 10 epochs and started convergence at about 16 epochs and finally achieved an accuracy of 94.83% with 19 hidden layers. As shown in Fig 5.27(b), the model tested using 15 hidden layers started to converge after 21 epochs and finally achieved an optimal classification performance of 72.07%. The model tested using 17 hidden layers started to converge after 21 epochs and finally achieved an optimal classification performance of 78.72%. The model tested using 17 hidden layers started to converge after 15 epochs and finally achieved an optimal classification performance of 85.83%.

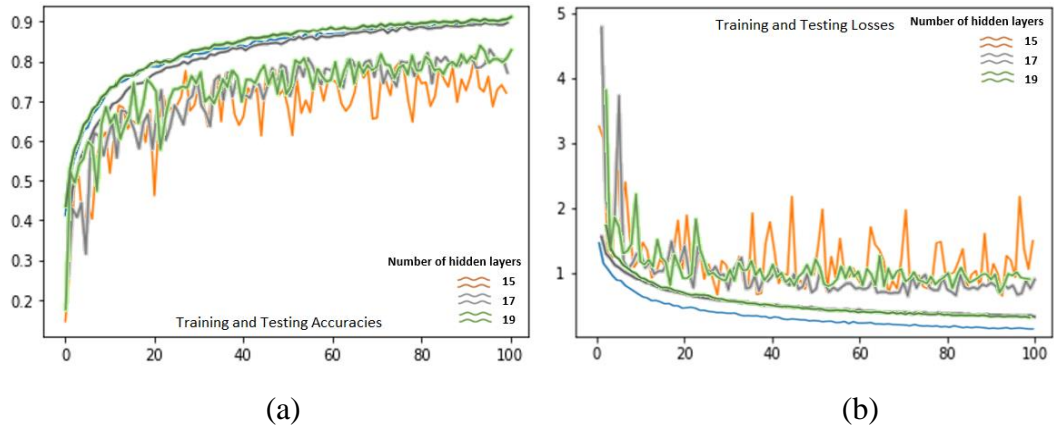
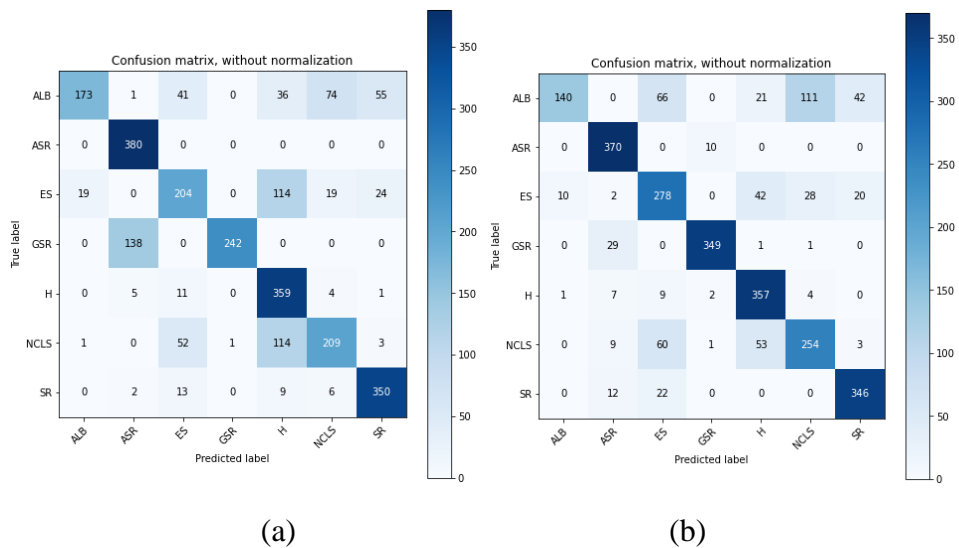
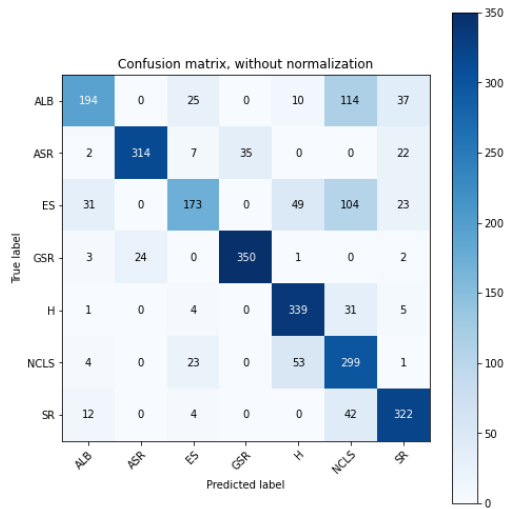


Fig 5.26. Accuracy and loss curves with different number of hidden layers

Fig 5.27(a) shows the confusion matrix associated with results obtained using 15 hidden layers. The classification results of ASR, H, and SR were consistent for three disease classes with 100%, 94% and 92% respectively except ALB, ES, GSR and NCLS. Fig 5.28(b) shows the confusion matrix associated with results obtained using 17 hidden layers. The classification results of ASR, GSR, H, and SR were consistent for four disease classes with 97%, 92%, 94%, and 91% respectively except ALB, ES, and NCLS. Fig 5.27(c) shows the confusion matrix associated with results obtained using 19 hidden layers. The classification results of GSR are consistent for one disease class with 92% except ASR, ALB, ES, H, NCLS and SR. The results presented a considerable sum of classification errors for disease classes ASR, H and SR with 17%, 11% and 15%. Fig 5.28 shows the performance measure chart for different hidden layers.





(c)

Fig 5.27. Confusion matrices for different hidden layers

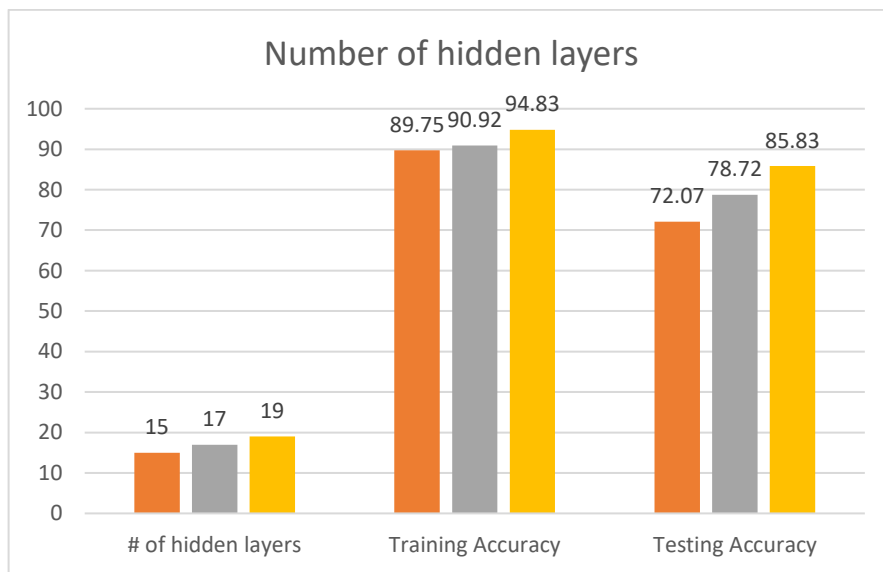


Fig 5.28. Comparison of model performance with different number of hidden layers



Table 5.14 Confusion metric values with different image sizes

Image Size	112 x 112							168 x 168							224 x 224						
Disease Class	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
<b>Precision</b>	0.84	0.95	0.74	0.99	0.86	0.71	0.92	0.84	0.95	0.75	1.00	0.84	0.74	0.81	0.82	0.87	0.77	1.00	0.91	0.75	0.86
<b>Recall</b>	0.80	0.99	0.60	0.97	0.90	0.81	0.94	0.76	0.99	0.64	0.93	0.93	0.70	0.98	0.78	1.00	0.64	0.86	0.91	0.84	0.94
<b>F1 Score</b>	0.82	0.97	0.66	0.98	0.88	0.76	0.93	0.80	0.97	0.69	0.96	0.88	0.72	0.89	0.80	0.93	0.70	0.92	0.91	0.79	0.90

Table 5.15 Confusion metric values with different image sizes

Batch Size	32							64							128						
Disease Class	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
<b>Precision</b>	0.84	0.95	0.74	0.99	0.86	0.71	0.92	0.76	0.72	0.77	1.00	0.92	0.72	0.84	0.89	0.97	0.73	0.99	0.83	0.75	0.74
<b>Recall</b>	0.80	0.99	0.60	0.97	0.90	0.81	0.94	0.87	0.99	0.51	0.67	0.84	0.80	0.93	0.73	0.99	0.58	0.96	0.89	0.74	0.98
<b>F1 Score</b>	0.82	0.97	0.66	0.98	0.88	0.76	0.93	0.81	0.83	0.61	0.80	0.88	0.76	0.88	0.80	0.98	0.65	0.98	0.86	0.74	0.84

Table 5.16 Confusion metric values with different number of epochs

# of epochs	50							100							200						
Disease Class	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6

<b>Precision</b>	0.79	0.93	0.73	0.91	0.75	0.51	0.78	0.84	0.95	0.74	0.99	0.86	0.71	0.92	0.87	0.96	0.79	0.99	0.88	0.75	0.88
<b>Recall</b>	0.51	0.83	0.46	0.92	0.89	0.79	0.85	0.80	0.99	0.60	0.97	0.90	0.81	0.94	0.84	0.98	0.57	0.96	0.96	0.87	0.94
<b>F1 Score</b>	0.62	0.87	0.56	0.92	0.81	0.62	0.81	0.82	0.97	0.66	0.98	0.88	0.76	0.93	0.85	0.97	0.66	0.97	0.92	0.80	0.91

Table 5.17 Confusion metric values with different optimizers

<b>Optimizer</b>	<b>Adam</b>							<b>RMSprop</b>							<b>SGD</b>						
<b>Disease Class</b>	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
<b>Precision</b>	0.84	0.95	0.74	0.99	0.86	0.71	0.92	0.91	0.90	0.65	0.99	0.79	0.75	0.80	0.91	0.85	0.77	0.98	0.79	0.74	0.65
<b>Recall</b>	0.80	0.99	0.60	0.97	0.90	0.81	0.94	0.56	0.97	0.66	0.88	0.92	0.80	0.95	0.58	0.98	0.56	0.84	0.92	0.73	0.97
<b>F1 Score</b>	0.82	0.97	0.66	0.98	0.88	0.76	0.93	0.70	0.93	0.66	0.93	0.85	0.78	0.87	0.71	0.91	0.65	0.91	0.85	0.74	0.78

Table 5.18 Confusion metric values with different learning rates

<b>Learning Rate</b>	<b>0.001</b>							<b>0.1</b>							<b>0.005</b>						
<b>Disease Class</b>	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
<b>Precision</b>	0.84	0.95	0.74	0.99	0.86	0.71	0.92	0.77	0.91	0.64	0.97	0.82	0.51	0.89	0.85	0.80	0.68	0.99	0.82	0.70	0.94
<b>Recall</b>	0.80	0.99	0.60	0.97	0.90	0.81	0.94	0.52	0.97	0.55	0.93	0.77	0.77	0.87	0.77	1.00	0.62	0.78	0.90	0.81	0.83

<b>F1 Score</b>	0.82	0.97	0.66	0.98	0.88	0.76	0.93	0.62	0.94	0.59	0.95	0.79	0.61	0.88	0.81	0.89	0.64	0.88	0.86	0.75	0.88
-----------------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------

Table 5.19 Confusion metric values with different kernel sizes

<b>Kernel Size</b>	<b>3 x 3</b>							<b>5 x 5</b>							<b>7 x 7</b>						
<b>Disease Class</b>	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
<b>Precision</b>	0.84	0.95	0.74	0.99	0.86	0.71	0.92	0.82	0.87	0.77	1.00	0.91	0.75	0.86	0.80	0.91	0.83	0.96	0.79	0.77	0.86
<b>Recall</b>	0.80	0.99	0.60	0.97	0.90	0.81	0.94	0.78	1.00	0.64	0.86	0.91	0.84	0.94	0.78	0.96	0.50	0.98	0.91	0.83	0.96
<b>F1 Score</b>	0.82	0.97	0.66	0.98	0.88	0.76	0.93	0.80	0.93	0.70	0.92	0.91	0.79	0.90	0.79	0.94	0.62	0.97	0.84	0.80	0.91

Table 5.20 Confusion metric values with different number of hidden layers

<b># of hidden layers</b>	<b>15</b>							<b>17</b>							<b>19</b>						
<b>Disease Class</b>	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
<b>Precision</b>	0.90	0.72	0.64	1.00	0.57	0.67	0.81	0.93	0.86	0.64	0.96	0.75	0.64	0.84	0.84	0.95	0.74	0.99	0.86	0.71	0.92
<b>Recall</b>	0.46	1.00	0.54	0.64	0.94	0.55	0.92	0.37	0.97	0.73	0.92	0.94	0.67	0.91	0.80	0.99	0.60	0.97	0.90	0.81	0.94
<b>F1 Score</b>	0.60	0.84	0.58	0.78	0.71	0.60	0.86	0.53	0.91	0.68	0.94	0.84	0.65	0.87	0.82	0.97	0.66	0.98	0.88	0.76	0.93

## 5.5 Comparative Analysis

To evaluate the classification efficiency of the proposed work, this sub-section examined at state-of-the-art deep learning models and existing frameworks. We have performed forty-one experiments to test the accuracy of proposed model and validate its results.

### 5.5.1 Pre-Trained Models

In this sub section, the experiments carried out to compare the performances of different pre-trained models [66-69] with the proposed has been reported. The evaluation process of the pre-trained models using own image sets is shown in Fig 5.29.

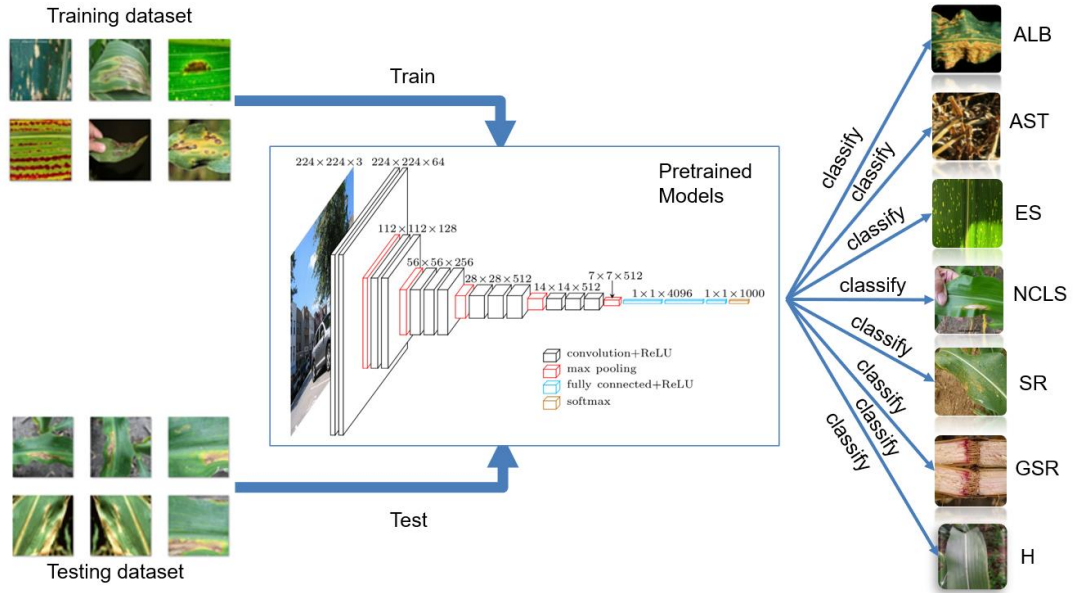


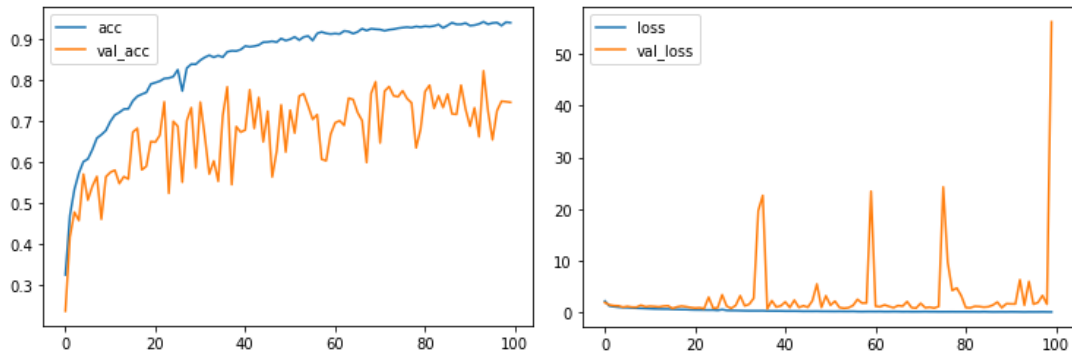
Fig 5.29 Performance evaluation process using Pre-Trained Models

The performance of all the models have been evaluated for 100 number of epochs. The training and testing accuracy results obtained after conducting the experiments are summarized in Table 5.14.

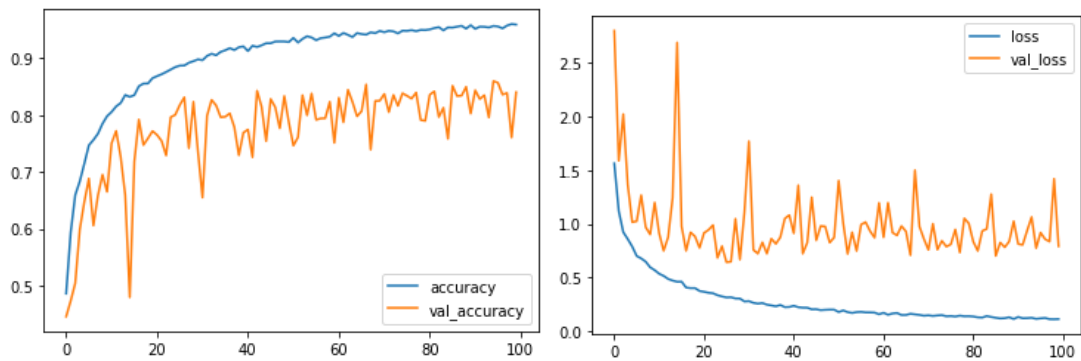
Table 5.21. Performance results of pre-trained model and proposed model

Reference	Model Name	Dataset	Training Accuracy	Testing Accuracy	# of Wrong Predictions
[66]	Dense Net 121	Own	94.04%	74.59%	676
[67]	Inception V2	Own	94.16%	83.42%	441
[68]	Shallow Net 8	Own	96.86%	83.01%	452
[69]	CNN-SVM	Own	94.15%	84.06%	421
	<b>Proposed NPNet-19 Model</b>	<b>Own</b>	<b>94.52%</b>	<b>85.16%</b>	<b>401</b>

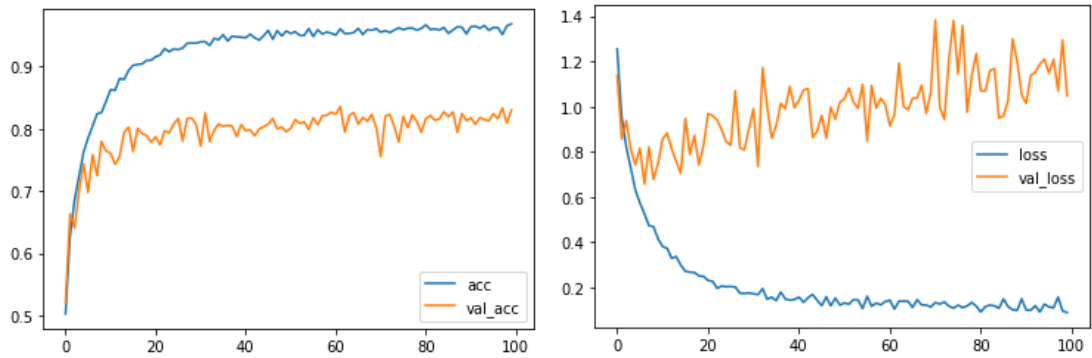
The findings show that the proposed classification system achieves improvements in accuracy of 10.57%, 1.74%, 2.15%, and 1.1%, respectively, as compared to pre-trained models [66-69]. The proposed model has shown a higher classification accuracy of 85.16% compared with the other pre-trained models. Table 5.6 displays the confusion matrix values after the evaluating operation and Fig 5.30 depicts the findings outlined in Table 5.14.



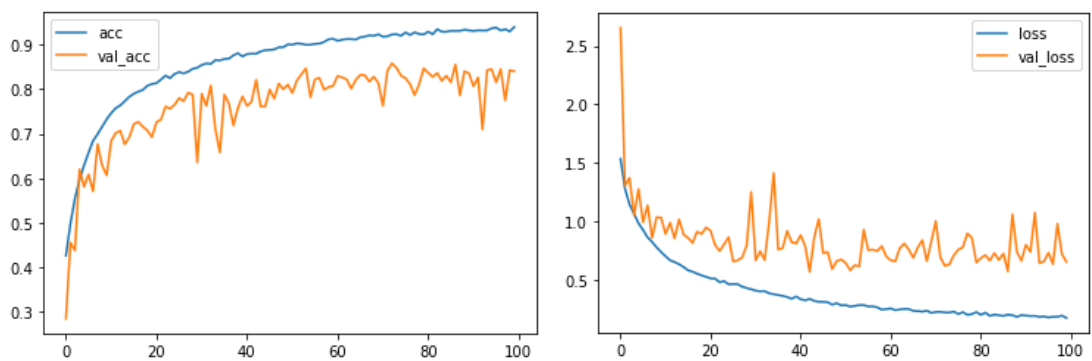
(a) DenseNet – 121



(b) Inception V2



(c) Shallow Net 8



(d) CNN-SVM

Figure 5.30 Accuracy and Loss curves of pre-trained models

Table 5.22. Class-wise classification accuracies for pre-trained and proposed models

References	Class Index	Precision	Recall	F1 Score
[66]	0	79%	61%	69%
	1	93%	96%	58%
	2	61%	56%	58%
	3	96%	94%	95%
	4	51%	79%	62%
	5	69%	64%	66%
	6	88%	72%	79%
	0	92%	99%	95%
	1	82%	64%	72%
	2	98%	98%	98%

<b>[67]</b>	3	83%	75%	79%
	4	71%	79%	75%
	5	71%	79%	75%
	6	75%	95%	75%
<b>[68]</b>	0	82%	84%	83%
	1	91%	100%	95%
	2	72%	60%	66%
	3	100%	92%	96%
	4	82%	79%	81%
	5	69%	79%	74%
<b>[69]</b>	6	87%	87%	87%
	0	90%	75%	82%
	1	88%	100%	93%
	2	76%	70%	73%
	3	100%	92%	96%
	4	87%	78%	82%
	5	69%	82%	75%
<b>Proposed NPNet-19 Model</b>	6	83%	91%	87%
	0	88%	82%	85%
	1	92%	98%	95%
	2	75%	73%	74%
	3	98%	93%	96%
	4	83%	89%	86%
	5	83%	77%	80%
6	85%	91%	88%	

Table 5.15 compares the precision, recall, and F1-score of seven maize crop diseases for the pre-trained models. The top F1 score obtained by [66] model on disease type 3 (GSR) is 95%. The difference between precision and recall values of type 0 disease (ALB) is 79% ~ 61%. It describes that only few images with type 0 disease are classified as disease type 0. The difference between precision and recall values of type 3 disease

(NLB) is 96% ~ 94%. It means that most of the images with type 3 disease are classified as disease type 3.

The top F1 score obtained by [67] model on disease type 2 (ES) is 98%. The difference between the precision and recall values of type 1 disease (ASR) is 88% ~ 64%. It describes that only few images with type 1 disease are classified as disease type 1. The difference between precision and recall values shows that type 2 disease (ES) is 98% ~ 98%. It means that all the images with type 2 disease are classified as disease type 2.

The top F1 score obtained by [68] model on disease type 3 (GSR) is 96%. The difference between precision and recall values of type 2 disease (ES) is 72% ~ 60%. It describes that only few images with type 2 disease are classified as disease type 2. The difference between precision and recall values of type 6 disease (NLS) is 87% ~ 87%. It means that all the images with type 6 disease are classified as disease type 6.

The top F1 score obtained by [69] model on disease type 3 (GSR) is 96%. The difference between precision and recall values of type 0 disease (ALB) is 79% ~ 69%. It describes that only few images with type 0 disease are classified as disease type 0. The difference between precision and recall values of type 3 disease (GSR) is 100% ~ 92%. It means that most of the images with type 3 disease are classified as disease type 3.

The top F1 score obtained by the proposed model on disease type 1 (ASR) is 96%. The difference between precision and recall values of type 0 disease (ALB) is 88% ~ 86%. It describes that most of the images with type 0 disease are classified as disease type 0. The difference between precision and recall values of type 2 disease (ES) is 81% ~ 61%. It means that only few images with type 2 disease are classified as disease type 2.

The proposed model has performed the classification of type 0, 1, 3, 4 and 5 with a higher level of accuracy. The overall observations describe that disease type 1,2 and 3 has obtained higher F1 score. Later, the performance of existing and proposed models is evaluated and the results obtained are shown in Table 5.16 and depicted Fig 5.31.



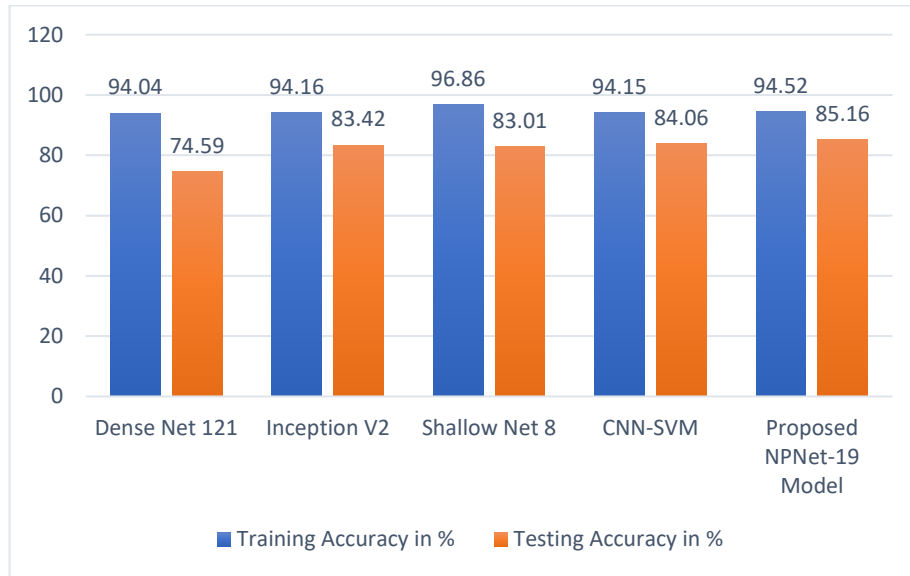


Fig 5.31. Performance measures of pre-training and proposed models

### 5.5.2 Transfer Learning Models

In this sub section, the experiments are carried out to compare the performance of six different existing model developed using transfer learning approach. Later, the performance of the models is compared with the proposed one.

Table 5.23. Performance evaluation of existing models and the proposed model

Reference	Model Name	Dataset	Training Accuracy	Testing Accuracy	# of Wrong Predictions
[57]	Modified LeNet	Own	89.17%	75.04%	664
[70]	CNN-ST	Own	82.93%	70.64%	781
[71]	SoyNet	Own	93.33%	76.99%	612
[63]	Adoptive CNN	Own	94.22%	81.28%	498
[61]	9-Layered	Own	93.17%	77.91%	579
[72]	13-Layered	Own	96.32%	81.77%	485
<b>Proposed Model</b>	<b>NPNet-19</b>	<b>Own</b>	<b>94.52%</b>	<b>85.16%</b>	<b>401</b>

In comparison to current transfer learning models [57, 61, 63, 70-72] the proposed method achieves an improvement in accuracy of 10.12%, 14.52%, 8.17%, 3.88%, 7.25%, and 3.39% as mentioned in Table 5.16. According to the findings, the proposed model outperforms other existing models in terms of classification accuracy. The

analysis described in Table 5.16 is shown in Fig 5.32. After comparing the performance of pre-trained models and the proposed model. Table 5.17 displays the confusion matrix values.

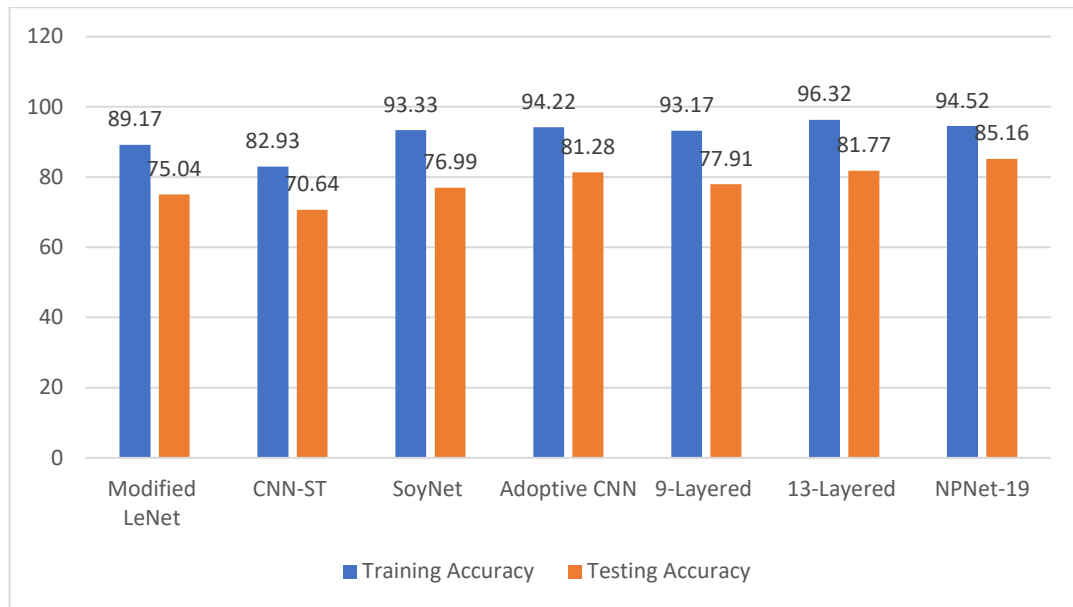


Fig 5.32. Performance measures of existing and proposed models

Table 5.24. Confusion matrix for existing and proposed models

Reference	Class Index	Precision	Recall	F1 Score
[57]	0	80%	70%	75%
	1	81%	90%	85%
	2	76%	55%	64%
	3	90%	92%	91%
	4	75%	82%	78%
	5	53%	78%	63%
	6	83%	58%	69%
[70]	0	87%	49%	83%
	1	79%	87%	83%
	2	63%	65%	64%
	3	88%	90%	89%
	4	75%	72%	74%

	5	48%	74%	58%
	6	74%	56%	64%
[71]	0	59%	53%	64%
	1	94%	95%	94%
	2	68%	63%	65%
	3	100%	84%	91%
	4	56%	83%	67%
	5	56%	83%	67%
	6	72%	94%	81%
[63]	0	86%	62%	72%
	1	95%	97%	96%
	2	65%	57%	61%
	3	97%	97%	97%
	4	81%	82%	82%
	5	65%	80%	72%
	6	82%	94%	88%
[61]	0	74%	76%	75%
	1	94%	96%	95%
	2	75%	57%	65%
	3	96%	98%	97%
	4	89%	67%	76%
	5	55%	84%	66%
	6	79%	69%	74%
[72]	0	79%	61%	69%
	1	93%	96%	94%
	2	61%	56%	58%
	3	96%	94%	95%
	4	51%	79%	62%
	5	69%	64%	66%
	6	88%	72%	79%
	0	88%	82%	85%

Proposed	1	92%	98%	95%
NPNet-19 Model	2	75%	73%	74%
	3	98%	93%	96%
	4	83%	89%	86%
	5	83%	77%	80%
	6	85%	91%	88

Table 5.17 compares the precision, recall, and F1-score of seven maize crop diseases for the pre-trained models. The top F1 score obtained by [57] model on disease type 3 (GSR) is 91%. The difference between precision and recall values of type 2 disease (ES) is 76% ~ 55%. It describes that only few images of type 2 disease are classified as disease type 2. The difference between precision and recall values of type 3 disease (GSR) is 90% ~ 92%. It means that most of the images of type 3 disease are classified as disease type 3.

The top F1 score obtained by [70] model on disease type 3 (GSR) is 90%. The difference between precision and recall values of type 2 disease (ES) is 87% ~ 49%. It describes that very few images with type 2 disease are classified as disease type 2. The difference between precision and recall values of type 3 disease (GSR) is 75% ~ 72%. It means that most of the images with type 3 disease are classified as disease type 3.

The top F1 score obtained by [71] model on disease type 1 (ASR) is 94%. The difference between precision and recall values of type 4 disease (H) is 56% ~ 83%. It describes that only few images of type 4 disease are classified as disease type 4. The difference between precision and recall values of type 1 disease (ASR) is 94% ~ 95%. It means that most of the images of type 1 disease are classified as disease type 1.

The top F1 score obtained by [63] model on disease type 3 (GSR) is 97%. The difference between precision and recall values of type 0 disease (ALB) is 86% ~ 62%. It describes that only few images of type 0 disease are classified as disease type 0. The difference between precision and recall values of type 3 disease (GSR) is 97% ~ 97%. It means that all the images of type 3 disease are classified as disease type 3.

The top F1 score obtained by [61] model on disease type 3 (GSR) is 97%. The difference between precision and recall values of type 3 disease (GSR) is 96% ~ 98%. It describes that most of the images of type 3 disease are classified as disease type 3. The difference between precision and recall values of type 2 disease (ES) is 75% ~ 57%. It means that only few images of type 2 disease are classified as disease type 2.

The top F1 score obtained by [72] model on disease type 3 (GSR) is 94%. The difference between precision and recall values of type 0 disease (ALB) is 79% ~ 61%. It describes that only few images of type 0 disease are classified as disease type 0. The difference between precision and recall values of type 3 disease (GSR) is 96% ~ 94%. It means that most of the images of type 3 disease are classified as disease type 3.

The top F1 score obtained by the proposed model on disease type 3 (GSR) is 96%. The difference between precision and recall values of type 3 disease (GSR) is 88% ~ 86%. It describes that most of the images with type 3 disease are classified as disease type 3. The difference between precision and recall values of type 2 disease (ES) is 81% ~ 61%. It means that very few images with type 2 disease are classified as disease type 2.

The results of the experiments show that the new model is capable of accurately detecting maize crop diseases in real-time agriculture fields. In comparison to pre-trained models and other transfer learning methods, this model has shown considerable better accuracy.

### **5.5.3 Plant Village Dataset**

Finally, the proposed model's classification efficiency is tested using images collected from the Plant Village dataset [73]. A total of 3852 maize crop images of four different classes including healthy are used for classification. The dataset is randomly distributed into training and testing datasets at a ratio of 80:20. The training dataset is used to train the NPNet-19. The testing dataset is then used to evaluate the NPNet-19 model performance. The performance of proposed model has been evaluated for 100 epoch and finally achieved a classification accuracy of 96.76%. The results have proved that the new model is efficient in classifying the images collected from the internet sources as well. The classification report, including the confusion matrix is shown in Table 5.18.

The training and validation classification accuracies including losses is shown in Fig. 5.33. The confusion matrix with and without normalization is shown in Fig 5.33.

Table 5.18 shows the precision, recall, and F1-score of four maize crop diseases. The top F1 score obtained by the proposed model on disease type 2 (H) is 100%. The difference between precision and recall values of type 2 disease (H) is 100% ~ 100%. It describes that all the images with type 2 disease are classified as disease type 2. The difference between precision and recall values shows that type 3 disease (NLB) is 99% ~ 88%. It means that only few images with type 3 disease are classified as disease type.

Table 5.25. Confusion Matrix of four-class classification problem

<b>Class Index</b>	<b>Precision</b>	<b>Recall</b>	<b>F1-Score</b>	<b>Support</b>
<b>0</b>	99%	99%	99%	193
<b>1</b>	89%	99%	94%	193
<b>2</b>	100%	100%	100%	193
<b>3</b>	99%	88%	93%	193
<b>Accuracy</b>			97%	193
<b>Macro Avg</b>	97%	97%	97%	772
<b>Weighted Avg</b>	97%	97%	97%	772
<b>Total Number of Test Cases: 772</b>				
<b>Number of Wrong Predictions: 25</b>				

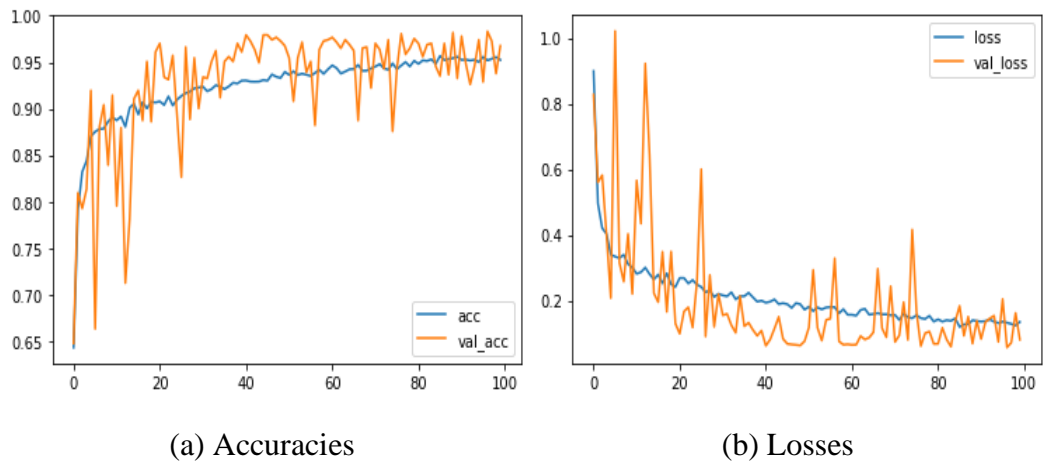


Fig 5.33. Training and Testing curves with plant village image sets

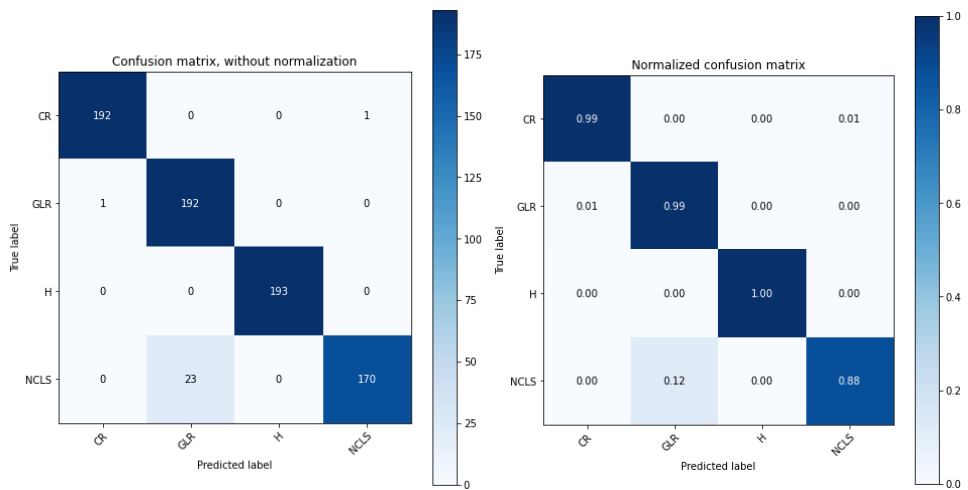


Fig 5.34. Confusion Matrix for without and with normalization values

## **CHAPTER – 6**

### **CONCLUSION**

In this research, a survey of deep learning techniques was conducted to see how useful they are in the agricultural area. We concentrated on the data sources, models used, pre-processing strategies used, and overall efficiency of the suggested CNN models. The results showed that most existing CNN models were limited in their ability to interpret unstructured raw image input. Deep learning systems require systematic engineering and expert design skills to extract features from unstructured data and convert them into feature vectors, which are then used by subsystems to recognise and categorise certain patterns in input data. The goal of this survey is to encourage researchers to use deep learning approaches for plant disease identification and classification using image analysis.

Most of the data augmentation techniques are designed and applied manually that need knowledge and implementation time. Common augmentation methods to recognize images are manually designed and are dataset specific. Such methods based on affine and color transformations are very easy and fast to implement but they have not proved to be the best methods to increase the images in a training dataset. Moreover, the traditional augmentation methods do not create any new visualize features of an image that can improve the performance of a learning model significantly. The methods mentioned are not popular and all of them are not considered as the common one to increase the images in a training dataset. An image acquisition and augmentation method proposed in this research takes advantage of the techniques of transfer learning and increase of image size in terms of quantity. In this research, a wide variation of commonly used position and color augmentations is presented and proposed many new and advanced techniques. The first two algorithms developed in this research can be used to acquire the images from internet and real-world sources to train and test a CNN deep learning model. The next developed algorithm can improve the implementation strategy to acquire more images from an original image and increase the training dataset size. Later, we analyse 52 techniques of augmentation by applying them to an original input image. The results of the analysis are observed and proved that the proposed new



techniques will be the best way to deal with unbalanced datasets. The future scope of this work is to apply more augmentation techniques like mixing images, pooling, perspective transform, segmentation, convolutional and geometric that can bring enormous potential to improve the data insufficiency to train deep learning models.

Convolutional neural networks have accomplished remarkable improvement in strengthening researcher's interest in the field of vision-related tasks and machine learning. As a classification of artificial intelligence, CNN has become more dominant in various domains like natural language processing, image classification, motion detection, speech recognition, and many more. In this research, an extensive survey on CNN based on architectures, building blocks, spatial exploitations, and applications were done. Beyond surveying every aspect of CNN, we also reviewed the development of CNN's architecture based on design changes in processing units and therefore suggested the taxonomy of CNN architectures. In addition to differentiating CNNs into various classes, the research provides a better understanding of the evolution, implementation challenges, and possible directions.

This research has proposed a new CNN architecture named NPNet-19 for the detection and classification of six common maize crop diseases and healthy leaves. Based on 4758 collected maize crop images, 13330 images were created using image augmentation techniques. The deep separable convolutions and regularization techniques like batch normalization and dropout was applied to the model to alleviate overfitting. In this research, the potential performance efficiency of the proposed model has examined first by varying the parameters like train-test dataset split ratio, kernel size, and the number of epochs. Later, the performance of NPNet-19 has compared with the four pre-trained models like DenseNet-121, Inception V2, ShallowNet-8, and CNN-SVM. The proposed model has shown an improvement of 10.57%, 1.74%, 2.15%, and 1.1% respectively. Finally, the performance of the proposed model has compared with six existing models like Modified LeNet, CNN-ST, SoyNet, Adaptive CNN, 9-layer, and 13-layered architectures. The new model showed an improved classification accuracy of 10.12%, 14.52%, 8.17%, 3.88%, 7.25%, and 3.39% respectively. After analysing the results, it has been proved that NPNet-19 is outperformed while classifying real-time maize crop images.

The selection of appropriate hyperparameters such as the input image size, optimizers, learning rate, kernel sizes of convolutional layer, the number of hidden layers, batch size, number of epochs are the major barriers to apply CNN and requires considerable skill and experience. Internal dependencies in these hyper-parameters make tuning them more costly and time consuming. There is no standard procedure or technique that can help in selecting the suitable values for hyperparameters. The selection process should be done on the trial-and-error basis only. The present research has shown the inter dependencies among the hyperparameters that can affect the performance of any CNN model through experimental approach. It has been proved that the learning time of a model mainly depends on the hyperparameters like image size, kernel size, number of hidden layers and number of epochs. The selection of higher values for these parameters can increase the running time of the model and need a machine accelerated with a high-capacity GPU.

The experimental results also describe that running the model for a higher number of epochs can improve the classification accuracy while classifying real-time field data. The NPNet-19 model used in this study has achieved a classification accuracy of 87.44% when it is trained for 200 number of epochs. It is also describing that the selection of input image size with 168 x 168 and 224 x 224 have achieved a higher classification accuracy of 84.66% and 85.23% respectively. The model achieved a classification accuracy of 85.83% when it is optimized with an Adam optimizer and a learning rate of 0.001. The results are higher than the other optimization techniques like RMSprop with 81.95% and SGD with 79.66% only.

### ***6.1 Findings***

This sub section summarized and documented the experimental results and the observations made during the evaluation process.

In this research, NPNet-19 deep learning network were investigated for classifying maize crop diseases. First CNN hyperparameters and their values were explored and later investigated the model performance. The input image size of 224 x 224 has outperformed when compared with 112 x 112 and 168 x 168. The Adam optimizer performed better than RMSprop and SGD. Hence it is recommended to use Adam

optimizer while performing maize crop disease classification. The Adam optimizer with a learning rate of 0.001 performed well when compared with learning rates 0.01 and 0.005. The number of hidden layers should be at least 17 layers when feeding a model with an image size of 224 x 224. The kernel size should be at least 5 x 5 or 7 x 7 when training the model using images of size 224 x 224. The number of epochs should be more than 200 so that the model can learn more features and performs the classification well. The model outperformed when the image size of 224 x 224, batch size 32, 200 number of epochs, Adam optimizer with a learning rate of 0.001, kernel size 5 x 5, and 19 number of hidden layers hyperparameter values were selected. Computational power will be more when the model is trained with an image size of 224 x 224 and the kernel size of 7 x 7. It is adequate to provide graphical processing units (GPUs) to optimize the model configurations in reasonable period.

The images of crop leaf for a specific disease are very difficult to acquire. This fact is visible by observing the limited size of image datasets used in the literature. Only less than 30% of the studies have used large datasets ranging in thousands. The most common observation is using a small dataset of images during testing and large propagation during training. It is analysed that an efficient image augmentation is needed to balance the count of images among the different classes. Table 2.3. clearly mentioned various literatures consider for the comparative analysis and impact of various traditional and conventional augmentation techniques. The observations showed how augmentation techniques plays a vital role in improving the size of the datasets that can be further used to train any model to obtain efficient results in disease identification and classification. The techniques implemented by the literatures improved the dataset size at a minimum of 100 times and maximum of 1200 times than the size of original dataset. The present study has identified 52 techniques with different types like position transformation, color transformation, color augmentation and affine transmission. It is observed that after applying the techniques the dataset has increased 1375 times than the size of original input dataset which is 175 times higher than the existing techniques in the literature. The study also mentioned how the newly identified advanced augmentation techniques are best to improve that size of the input datasets.

## ***6.2 Observations***

A classification accuracy of 88.72% has achieved when the proposed model is trained for 500 epochs. The obtained accuracy is 5.3% higher than the results achieved after running the same model for 100 epochs. A classification accuracy of 89% has achieved when the proposed model is trained for 500 epochs on the dataset split ratio of 80:20. A higher accuracy of 7% has obtained when the model is trained for 100 epochs on the same dataset split ratio.

A classification accuracy of 85% has achieved when the proposed model is trained with kernel size 5 x 5 and 7 x 7 and for 100 number of epochs. The proposed model has obtained accuracy of 4% higher than the kernel size 3 x 3 is obtained. But the same kernel sizes have shown less performance of 2-4% when the same model is trained with kernel size 3 x 3 for 500 epochs.

The proposed model has achieved 97% classification accuracy when it is trained for only 100 epochs using the images collected from the Plant Village and Kaggle data sources. A classification accuracy of 85.16% has achieved when the model is trained and tested for 100 epochs. When compared with four pre-trained model the proposed model has achieved a higher average accuracy of 3.76%. A classification accuracy of 85.16% has achieved when the proposed model is trained and tested for 100 epochs. The proposed model has achieved an average accuracy of 7.89% that is higher than the accuracy obtained by the other six existing models.

## ***6.3 Discussions***

In connection to the observations in the previous section, we provide few discussions. It is observed that the image size 224 x 224 can make the model learn more features by using all the pixel values during the classification process. It is proved that the kernel size 3 x 3 is the standard and most effective one while dealing with real-time images. It is best practice to use a small number of filters for the first convolutional layers to make the model learn more image features. Forth, A higher classification accuracy of the proposed model can be obtained by running it at least for 500 epochs. Adam optimizer has proved to be more effective again as an optimization function during multi-class image classification.

The dropout regularization technique and image augmentations can reduce model overfitting while identifying new testing dataset images. Regression predictive modelling technique can be used to evaluate the performance of the proposed model in terms of error metrics. The metrics values of the training and testing datasets generated by the proposed model are same for both the datasets and shows that most of the actual images of the crop are correctly predicted as the same actual class.

It has been observed that the CNN models have performed well on the datasets extracted from the internet sources with a higher level of classification accuracies. But the same models have shown low accuracy levels when testing with real-time images.

In future research, we are planning to conduct more experiments based on several hyperparameters like different image size, kernel size, optimizer, learning rate, number of epochs, loss function and number of hidden layers. This approach can help the future research in selecting suitable hyperparameters while designing a new model. It will also be possible to investigate how these hyperparameters can affect the performance of a model.

## **CHAPTER – 7**

### **FUTURE SCOPE**

CNN has achieved good reliability in results either by its form or by a topology grid. CNN's vision function is one of the shortcomings. CNN may provide little robustness in the face of noise and other image changes. CNN learning requires strong hardware resources like GPU. CNN architecture is expected to be a field of research in the future. Attention is one of the key mechanisms for the capture of image information in the human visual system. CNN can increase model's representation efficiency by using generative learning capabilities. Deep CNN has many hyperparameters including activation function, kernel length, layer-set neurons, etc. The choice of hyperparameters and their measurement time is very difficult to set parameters in deep learning. The hyperparameters tuning is a tedious process of intuition-driven which may not be described explicitly. In this regard, the use of genetic algorithms can optimize the hyperparameter automatically through random searches and guidance of previous searches.

Researchers might benefit from familiarity with cutting-edge research methodology more generally. The research goal is to identify and predict numerous crop diseases at an early stage of infection using deep learning technologies in agriculture. When it comes to plant disease identification and classification, it's been found that DL does both. DL applications on disease recognition/classification and many others involving neural networks should be examined now. More deep learning approaches are expected to improve their performance in the future. Researchers will eventually have to use datasets obtained from real-world experiments to test their newly designed or developed models.

## REFERENCES

- [1] LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E, Hubbard, W., and Jackel, L. D. (1989). Backpropagation applied to handwritten zip code recognition. *Neural Comput.*, 1(4), pp. 541–551.
- [2] X. Liu, Z. Deng, and Y. Yang, “Recent progress in semantic image segmentation,” *Artif. Intell. Rev.*, vol. 52, no. 2, pp. 1089–1106, 2019.
- [3] Ian Goodfellow, Y. Bengio, and A. Courville, “Deep learning,” *Nat. Methods*, vol. 13, no. 1, p. 35, 2017.
- [4] A. S. Qureshi and A. Khan, “Adaptive Transfer Learning in Deep Neural Networks: Wind Power Prediction using Knowledge Transfer from Region to Region and Between Different Task Domains,” *arXiv Prepr. arXiv1810.12611*, 2018.
- [5] A. S. Qureshi, A. Khan, A. Zameer, and A. Usman, “Wind power prediction using deep neural network based meta regression and transfer learning,” *Appl. Soft Comput. J.*, vol. 58, pp. 742–755, 2017.
- [6] Q. Abbas, M. E. A. Ibrahim, and M. A. Jaffar, “A comprehensive review of recent advances on deep vision systems,” *Artif. Intell. Rev.*, vol. 52, no. 1, pp. 39–76, 2019.
- [7] Y. Guo, Y. Liu, A. Oerlemans, S. Lao, S. Wu, and M. S. Lew, “Deep learning for visual understanding: A review,” *Neurocomputing*, vol. 187, pp. 27–48, 2016.
- [8] W. Liu, Z. Wang, X. Liu, N. Zeng, Y. Liu, and F. E. Alsaadi, “A survey of deep neural network architectures and their applications,” *Neurocomputing*, vol. 234, no. October 2016, pp. 11–26, 2017.
- [9] Available Online at: [Web copy of AR \(Eng\) 3.pdf \(agricoop.nic.in\)](#)
- [10] Abbas, Ibrahim, and Jaffar, (2019). A comprehensive review of recent advances on deep vision systems. *Artificial. Intell. Rev.*, 52(1), pp. 39–76.
- [11] Kamilaris A, Gao F, Prenafeta-Boldu´ FX, Ali MI (2016) Agri-IoT: a semantic framework for internet of things-enabled smart farming applications. In: 3rd world forum on the internet of things (WFIoT) IEEE. Reston, pp 442–447.
- [12] Kamilaris A, Assumpcio A, Blasi AB, Torrellas M, Prenafeta-Boldu´ FX (2017) Estimating the environmental impact of agriculture by means of geospatial and

big data analysis: the case of Catalonia. From Science to Society. Springer, Luxembourg, pp 39–48.

- [13] Lowe A, Harrison N, French AP (2017) Hyperspectral image analysis techniques for the detection and classification of the early onset of plant disease and stress. *Plant Methods* 13(1):80.
- [14] Sladojevic S, Arsenovic M, Anderla A, Culibrk D, Stefanovic D (2016) Deep neural networks-based recognition of plant diseases by leaf image classification. *Comput Intell Neurosci* 2016:1–11.
- [15] Go´mez-Chova L, Tuia D, Moser G, Camps-Valls G (2015) Multimodal classification of remote sensing images: a review and future directions. *Proc IEEE* 103(9):1560–1584.
- [16] Wang G, Sun Y, Wang J (2017) Automatic image-based plant disease severity estimation using deep learning. *Comput Intell Neurosci* 2017:1–8.
- [17] D. M. Hawkins, “The problem of overfitting,” *J. Chem. Inf. Comput. Sci.*, vol. 44, no. 1, pp. 1\_12, 2004.
- [18] Malusi Sibiya, Mbuya Sumbwanyambe, “A Computational Procedure for the Recognition and Classification of Maize Leaf Disease out of Health Leaves using Convolutional Neural Networks”, *Agri Engineering*, 2019, pp.119-131.
- [19] Jiang Lu, Jie Hu, Guannam Zhao, Fenghua Mei, Changshui Zhang, “An In-Field Automatic Wheat Disease Diagnosis System, Elsevier 2017, pp.1-15.
- [20] Chad Dechant, Tyr Wiesner-Hanks, “Automated Identification of Northern Leaf Blight Infected Maize Plants from Field Imagery using Deep Learning”, *Phytopathology*, 2015, pp.1-26.
- [21] Xihai Zhang, Yue Qiao, Fanfeng Meng, Chengguo Fan and Mingming Zhang, “Identification of Maize Leaf Disease using Improved Deep Convolutional Neural Networks”, *IEEE Access* 2018, pp.30370-30377.
- [22] Tyr Wiesner-Hanks, Ethan L.Stewart, Nicholas Kaczmar, chad Dechant, Harvey Wu, Rebecca Nelson, Hod Lipson, and Michael A. Gore, “Image set for Deep Learning: Field Images of Maize A noted with Disease Symptoms”, *BMC Research Notes*, 2018, pp.1-3.
- [23] Aditya Khamparia, Gurinder Saini, Deepak Gupta, Ashish Khanna, Shrastic Tiwari, Victor Hugo, DeAlbuquerque, “Seasonal Crops Disease Prediction and



- Classification using Deep Convolutional Encoder Network”, *Circuits, Systems, and Signal Processing*, 2019, pp.1-14.
- [24] Fengle Zhua, Mengzhu Hea, Zengwei Zheng, Data augmentation using improved cDCGAN for plant vigor rating, *Computers and Electronics in Agriculture*, 175, 2020, 105603:105612.
- [25] Ahmadi.P, Muharam, F.M, Ahmad.K, Mansor.S.I.A, “Early Detection of Ganoderma Basal Stem Rot of Oil Palms using Artificial Neural Networks Spectral Analysis”, *Plant Dis*, 2017, 101, pp.1009-1016.
- [26] Juncheng Ma, Keming Du, Lingxian Zhang, Feixiang Zheng, Jinxiang Chu, Zhongfu Sun, “A segmentation method for greenhouse vegetable foliar disease spots images using color information and region growing”, *Computers and Electronics in Agriculture* 142, 2017, pp. 110–117.
- [27] Xuebing Bai, Xinxing Li, Zetian Fu, Xiongjie Lv, Lingxian Zhang, “A fuzzy clustering segmentation method based on neighbourhood grayscale information for defining cucumber leaf spot disease images”, *Computers and Electronics in Agriculture* 136, 2017 pp. 157– 165.
- [28] Guan Wang, Yu Sun, and Jianxin Wang, “Automatic Image-Based Plant Disease Severity Estimation Using Deep Learning”, *Computational Intelligence and Neuroscience Volume 2017*, pp. 1-8.
- [29] Sehan Kim, Meonghun Lee and Changsun Shin, “IoT-Based Strawberry Disease Prediction System for Smart Farming”, *Sensors* 2018, pp. 1-17.
- [30] Juncheng Ma, Keming Du, Feixiang Zheng, Lingxian Zhang, Zhihong Gong, Zhongfu Sun, “A recognition method for cucumber diseases using leaf symptom images based on deep convolutional neural network”, *Computers and Electronics in Agriculture*, 2018, pp. 18-24.
- [31] Manisha Bhange, H.A. Hingoliwala, “Smart Farming: Pomegranate Disease Detection Using Image Processing”, *Procedia Computer Science*, Vol. 58, 2015, pp. 280 – 288, Available: [www.sciencedirect.com](http://www.sciencedirect.com).
- [32] Jihen Amara, Bassem Bouaziz, and Alsayed Algergawy, “A Deep Learning-based Approach for Banana Leaf Diseases Classification”, *Lecture Notes in Informatics (LNI)*, Gesellschaft für Informatik, Bonn, 2017, pp.79.

- [33] Ulzii-Orshikh Dorj, Malrey Lee, Sang-seok Yun, “A yield estimation in citrus orchards via fruit detection and counting using image processing”, *Computers and Electronics in Agriculture* 140, 2017, pp. 103–112.
- [34] Andreas Kamilaris, Francesc X. Prenafeta-Boldú, “Deep learning in agriculture: A survey”, *Computers and Electronics in Agriculture* 147, 2018, pp. 70–90.
- [35] Ienco, D., Gaetano, R., Dupaquier, C., Maurel, P., “Land Cover Classification via Multi-temporal Spatial Data by Recurrent Neural Networks”. arXiv preprint arXiv: 1704.04055, 2017.
- [36] Rebetez, J. et al., “Augmenting a convolutional neural network with local histograms— a case study in crop classification from high-resolution UAV imagery”. *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*. Bruges, Belgium, 2016.
- [37] Grinblat, G.L., Uzal, L.C., Larese, M.G., Granitto, P.M., “Deep learning for plant identification using vein morphological patterns”. *Comput. Electron. Agric.* 127, pp. 418– 424.
- [38] Sa, I., Ge, Z., Dayoub, F., Upcroft, B., Perez, T., McCool, C., “Deep fruits: A fruit detection system using deep neural networks”. *Sensors* 16 (8), 1222.
- [39] Chen, S.W., Shiva Kumar, S.S., D’Cunha, S., Das, J., Okon, E., Qu, C., Kumar, V., “Counting apples and oranges with deep learning: a data-driven approach”. *IEEE Rob. Autom. Lett.* 2 (2), pp. 781–788.
- [40] Bargoti, S., Underwood, J., 2016. “Deep Fruit Detection in Orchards”. arXiv preprint arXiv: 1610.03677.
- [41] Wang, Z.M., Cao, H.J. and Fan, L. “Method on Human Activity Recognition Based on Convolutional Neural Networks”. *Computer Science*, 43, 2016, pp. 56-58.
- [42] Y. Xu, R. Jia, L. Mou, G. Li, Y. Chen, Y. Lu, and Z. Jin. “Improved Relation Classification by Deep Recurrent Neural Networks with Data Augmentation”, 2016.
- [43] [Luis Perez](#), and [Jason Wang](#), “The Effectiveness of Data Augmentation in Image Classification using Deep Learning ”, *Computer Vision and Pattern Recognition*, pp. 1- 8, 2017.

- [44] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton, “ImageNet Classification with Deep Convolutional Neural Networks”, *Communications of the ACM*, Vol.60, No.6, pp. 1-9, 2017.
- [45] Mohanty SP, Hughes D, and Salathe M, “Using Deep Learning for Image-Based Plant Disease Detection”, *Front Plant Sci*, Vol. 7, pp. 1–10, 2016.
- [46] Dyrmann, M., Karstoft, H., and Midtby, H.S., “Plant Species Classification using Deep Convolutional Neural Network”, *Biosyst. Eng.* 151, pp. 72–80, 2016.
- [47] Barbedo JGA, Koenigkan LV, and Santos TT, “Identifying Multiple Plant Diseases using Digital Image Processing”, *Biosystems Engineering*, 147, pp.104–116, 2016.
- [48] Shanqing Gu, Manisha Pednekar, Robert Slater, “Improve Image Classification Using Data Augmentation and Neural Networks”, *SMU Data Science Review*, Vol.2, No.2, pp. 1-44, 2019.
- [49] Prasad S, Peddoju SK, and Ghosh D, “Multi-resolution mobile vision system for plant leaf disease diagnosis”, *Signal Image Video Process*, Vol. 10(2), pp. 379–388, 2016.
- [50] Sukhvir Kaur, Shreelekha Pandey, Shivani Goel, “Plants Disease Identification and Classification Through Leaf Images: A Survey”, *Computational Methods in Engineering*, pp. 1-24, 2018.
- [51] Zhang SW, Shang YJ, and Wang L, “Plant Disease Recognition based on Plant Leaf Image”, *Journal of Animal Plant Science*, Vol. 25 (Suppl. 1), pp. 42–45, 2015.
- [52] Ekin D. Cubuk, Barret Zoph, Dandelion Mané, Vijay Vasudevan, Quoc V. Le, “Auto Augment: Learning Augmentation Strategies from Data”, *arXiv:1805.09501v3 [cs.CV]*, pp. 1-14, 2019.
- [53] Ramcharan, A., Baranowski, K., McCloskey, P., Ahamed, B., Legg, J., and Hughes, D. “Transfer learning for image-based cassava disease detection”. *Front. Plant Sci.* 8:1852, 2017. doi: 10.3389/fpls.2017.01852.
- [54] Bin Liu<sup>1</sup>, Zefeng Ding, Liangliang Tian, Dongjian He, Shuqin Li, and Hongyan Wang (2020), Graph Leaf Disease Identification using Improved Deep Convolutional Neural Networks, *Frontiers in Plants*, Vol.11: 1082-1096.  
<https://doi.org/10.3389/fpls.2020.01082>

- [55] Ramar Ahila Priyadharshini, Selvaraj Arivazhagan, Madakannu Arun, Annamalai Mirnalini (2019), Maize leaf disease classification using deep convolutional neural networks, *Neural Computing and Applications* <https://doi.org/10.1007/s00521-019-04228-3>:1-9.
- [56] Peng Jiag, Yuehan Chen , Bin Liu, Dongjian He, and Chunquan Liang (2019), Real-Time Detection of Apple Leaf Diseases Using Deep Learning Approach Based on Improved Convolutional Neural Networks, *IEEE Access*, Vol.7: 59069-59081.
- [57] Mingjie LV, Guoxiong Zhou, Mingfang He, Aibin Chen, Wenzhuo Zhang, and Yahui Hu (2020), Maize Leaf Disease Identification Based on Feature Enhancement and DMS-Robust Alexnet, *IEEE Access*, Vol. 8:57952-57968.
- [58] Akbarzadeh, S., Paap, A., Ahderom, S., Apopei, B., and Alameh, K. (2018). Plant Discrimination by Support Vector Machine Classifier Based on Spectral Reflectance. *Comput. Electron. Agric.* 148: 250–258. DOI: 10.1016/j.compag.2018.03.026.
- [59] Zhang, S., Zhang, S., Zhang, C., Wang, X., and Shi, Y. (2019). Cucumber Leaf Disease Identification with Global Pooling Dilated Convolutional Neural Network. *Comput. Electron. Agric.* 162: 422–430. DOI: 10.1016/j.compag.2019.03.012.
- [60] Mohammadpoor, M., Nooghabi, M. G., and Ahmedi, Z. (2020). An Intelligent Technique for Grape Fanleaf Virus Detection. *Int. J. Interact. Multimedia Artif. Intell.* 6: 62–67. DOI: 10.9781/ijimai.2020.02.001.
- [61] Geetharamani, G., and Pandian, A. J. (2019). Identification of Plant Leaf Diseases Using a Nine-layer Deep Convolutional Neural Network. *Comput. Electr. Eng.* 76: 323–338. DOI: 10.1016/j.compeleceng.2019.04.011.
- [62] Ji, M., Zhang, L., and Wu, Q. (2019). Automatic grape leaf diseases identification via UnitedModel based on multiple convolutional neural networks. *Inf. Process. Agric.* 1–9. DOI: 10.1016/j.inpa.2019.10.003.
- [63] Chowdhury R. Rahman, Preetom S. Arko, Mohammed E. Ali, Mohammad A. Iqbal Khan, Sajid H. Apon, Farzana Nowrin, and Abu Wasif (2020), Identification

- and recognition of rice diseases and pests using convolutional neural networks, *BioSystems Engineering*, Vol. 194: 112-120.
- [64] Johan Bjorck, Carla Gomes, Bart Selman, Kilian Q. Weinberger (2018), Understanding Batch Normalization, arXiv:1806.02375v4 [cs.LG].
- [65] Víctor Suárez-Paniagua and Isabel Segura-Bedmar, Evaluation of pooling operations in convolutional architectures for drug-drug interaction extraction, *BMC Bioinformatics* 2018.
- [66] Too, E.C.; Yujian, L.; Njuki, S. Yingchun, L (2019). A comparative study of fine-tuning deep learning models for plant disease identification. *Comput. Electron. Agric.* 161: 272–279.
- [67] Jiang, P.; Chen, Y.; Liu, B.; He, D.; Liang, C (2019). Real-Time Detection of Apple Leaf Diseases Using Deep Learning Approach Based on Improved Convolutional Neural Networks. *IEEE Access*, 7: 9069–59080.
- [68] Bin Liu, Yun Zhang, DongJian He, and Yuxiang Li (2018), Identification of Apple Leaf Diseases Based on Deep Convolutional Neural Networks, *Symmetry*:10-11; doi:10.3390/sym10010011.
- [69] Yingqiong Peng, Muxin Liao, Hong Deng, Ling AO, Yuxia Song, Weiji Huang, Jing Hua (2020), CNN-SVM: A classification method for fruit fly image the complex background, *IET Research Journal, IET Cyber-Physical Systems: Theory & Applications*:1-5.
- [70] Alexandre Barbosaa, Rodrigo Trevisanb, Naira Hovakimyana, Nicolas F. Martin (2020), Modeling yield response to crop management using convolutional neural networks, *Computers, and Electronics Agriculture* 170, pp. 1-8.
- [71] Aditya Karlekara, Ayan Seal (2020), SoyNet: Soybean leaf diseases classification, *Computers, and Electronics in Agriculture* 172, 105342, 2020:1-9.
- [72] Zhang, YD., Dong, Z., Chen, X. *et al.* Image based fruit category classification by 13-layer deep convolutional neural network and data augmentation. *Multimedia Tools and Applications*, **78**, 3613–3632 (2019). <https://doi.org/10.1007/s11042-017-5243-3>

- [73] Available online at [https://www.kaggle.com/emmarex/plant\\_disease](https://www.kaggle.com/emmarex/plant_disease)
- [74] Bansilal, S. (2017). The application of the percentage change calculation in the context of inflation in Mathematical Literacy. *Pythagoras*, 38(1).
- [75] A.D. Almási, S. Woźniak, V. Cristea, Y. Leblebici, T. Engbersen, “Review of advances in neural networks: neural design technology stack”, *Neurocomputing* 174, 2016, pp. 31–41.
- [76] Jingyao Zhang, Yuan Rao, Chao Man, Zhaohui Jiang and Shaowen Li, Identification of cucumber leaf diseases using deep learning and small sample size for agricultural Internet of Things, *International Journal of Distributed Sensor Networks* 2021, Vol. 17(4), 1:13.
- [77] Srdjan Sladojevic, Marko Arsenovic, Andras Anderla, Dubravko Culibrk, and Darko Stefanovic, “Deep Neural Networks Based Recognition of Plant Diseases by Leaf Image Classification”, *Computational Intelligence and Neuroscience*, pp. 1-12, 2016.

## LIST OF ARTICLES PUBLISHED/UNDER REVIEW

1. A research article titled “Systematic review of deep learning techniques in plant disease detection” is published in International Journal of System Assurance Engineering and Management, January 2020. [doi.org](#)
2. A conference paper titled “Deep CNN Architectures for Learning Image Classification: A Systematic Review, Taxonomy and Open Challenges” is presented and published in IGDTUW Proceedings of 2nd International Conference on Artificial Intelligence and Speech Technology, November 2020.
3. A conference paper titled “Plant Disease Classification using DCCN-19 Convolutional Neural Networks” is presented in IEEE 9<sup>th</sup> International Conference on Reliability, Infocom Technologies and Optimization (ICRITO'2021), September 2021.
4. A conference paper titled “Plant Disease Classification using DCCN-19 Convolutional Neural Networks” is published in IEEE 9<sup>th</sup> International Conference proceedings on Reliability, Infocom Technologies and Optimization (ICRITO'2021), November 2021.  
<https://ieeexplore.ieee.org/document/9596200>
5. A research article titled “Convolutional Network Model Based Leaf Disease Detection using Augmentation Techniques” is published in Expert Systems: The Journal of Knowledge Engineering, Wiley Publications, November 2021. <https://doi.org/10.1111/exsy.12885>
6. A research article titled “Performance Improvement of Deep Learning Models using Image Augmentation Techniques” is published in Multimedia Tools and Applications, Springer Journal, January 2022. <https://doi.org/10.1007/s11042-021-11869-x>
7. A research article titled “Maize Crop Disease Detection using NPNet-19 Convolutional Neural Network” is under review with Neural Computing and Applications, Springer Journal.
8. A research article titled “An Effective Image Augmentation Approach for Maize Crop Disease Recognition and Classification” is under review with Songklanakarin Journal of Science and Technology.

9. A research article titled “Convolutional Neural Network for Maize Crop Diseases Classification using Hyperparameter Optimization” is under review with Information Processing in Agriculture Journal.