

# **Classification and Severity Prediction of Maize Leaf Diseases using Deep Learning CNN Approaches**

By

Malusi Sibiya

A thesis submitted in fulfilment of a PhD degree in Science, Engineering and Technology at the University  
of South Africa.

Supervisor: Prof. M. Sumbwanyambe

# Declaration

Name: MALUSI SIBIYA

Student number: 43447619

Degree: PhD in Science, Engineering and Technology

The exact wording of the title of the thesis or thesis as appearing on the copies submitted for examination:  
**Classification and Severity Prediction of Maize Leaf Diseases using Deep Learning CNN Approaches.**

I declare that the above thesis is my own work and that all the sources that I have used or quoted have been indicated and acknowledged by means of complete references.

I further declare that I submitted the thesis to originality checking software and that it falls within the accepted requirements for originality.

I further declare that I have not previously submitted this work, or part of it, for examination at UNISA for another qualification or at any other higher education institution.

SIGNATURE



DATE

\_\_\_\_June 2021\_\_\_\_

## List of Publications

1. Sibiya, M. and Sumbwanyambe, M., 2019a. A computational procedure for the recognition and classification of maize leaf diseases out of healthy leaves using convolutional neural networks. *Agri Engineering, 1* (1), pp. 119-131.

Link: <https://www.mdpi.com/2624-7402/1/1/9>

Status: Published

2. Sibiya, M. and Sumbwanyambe, M., 2019b. An algorithm for severity estimation of plant leaf diseases by the use of colour threshold image segmentation and fuzzy logic inference: a proposed algorithm to update a “leaf doctor” application. *Agri Engineering, 1* (2), pp. 205-219.

Link: <https://www.mdpi.com/2624-7402/1/2/15>

Status: Published

3. Sibiya, M. and Sumbwanyambe, M., 2021. Automatic Fuzzy Logic-Based Maize Common Rust Disease Severity Predictions with Thresholding and Deep Learning. *Pathogens, 10* (2), p. 131.

Link: <https://www.mdpi.com/search?authors=malusi+sibiya&journal=pathogens>

Status: Published

4. Northern Corn Leaf Blight Severity Predictions based on Colour and Sporulation of the Lesions by using Convolutional Neural Networks: A Maize Leaf Diseases Odyssey. *Agri Engineering 2021*.

Link: <https://www.mdpi.com/journal/agriengineering>

Status: Submitted

This is a thesis by publication of the above declared journal papers. The publication of these journal papers as thesis chapters is protected by the MDPI research and publication ethics available on <https://www.mdpi.com/ethics>

## Conference Proceedings

1. Sibiya, M. and Sumbwanyambe, M., 2019, January. An Embedded Fuzzy Logic Microcontroller for Plant Condition Monitoring: A Wireless Sensor Network Odyssey. In *2019 Southern African Universities Power Engineering Conference/Robotics and Mechatronics/Pattern Recognition Association of South Africa (SAUPEC/RobMech/PRASA)* (pp. 565-569). IEEE.

Link: <https://ieeexplore.ieee.org/abstract/document/8704748>

2. Sibiya, M. and Sumbwanyambe, M., 2019, August. Controlling of Microclimatic Parameters of a Greenhouse by using Fuzzy Logic and Wireless Sensor Networks. In *2019 International Conference on Advances in Big Data, Computing and Data Communication Systems (icABCD)* (pp. 1-6). IEEE.

Link: <https://ieeexplore.ieee.org/abstract/document/8850999>

3. Sibiya, M. and Sumbwanyambe, M., 2020, August. PH sensor using Fuzzy Logic on Arduino for the monitoring and control of acidity or alkalinity in the reservoir's irrigation water. In *2020 International Conference on Artificial Intelligence, Big Data, Computing and Data Communication Systems (icABCD)* (pp. 1-6). IEEE.

Link: <https://ieeexplore.ieee.org/abstract/document/9183824>

# Abstract

Maize (*zea mays*) is the staple food of Southern Africa and most of the African regions. This staple food has been threatened by a lot of diseases in terms of its yield and existence. Within this domain, it is important for researchers to develop technologies that will ensure its average yield by classifying or predicting such diseases at an early stage. The prediction, and to some degree classifying, of such diseases, with much reference to Southern Africa staple food (Maize), will result in a reduction of hunger and increased affordability among families. Reference is made to the three diseases which are Common Rust (CR), Grey Leaf Spot (GLS) and Northern Corn Leaf Blight (NCLB) (this study will mainly focus on these). With increasing drought conditions prevailing across Southern Africa and by extension across Africa, it is very vital that necessary mitigation measures are put in place to prevent additional loss of crop yield through diseases. This study introduces the development of Deep Learning (DL) Convolutional Neural Networks (CNNs) (note that in this thesis deep learning or convolution neural network or the combination of both will be used interchangeably to mean one thing) in order to classify the disease types and predict the severity of such diseases. The study focuses primarily on the CNNs, which are one of the tools that can be used for classifying images of various maize leaf diseases and in the severity prediction of Common Rust (CR) and Northern Corn Leaf Blight (NCLB). In essence the objectives of this study are:

- i. To create and test a CNN model that can classify various types of maize leaf diseases.
- ii. To set up and test a CNN model that can predict the severities of a maize leaf disease known as the maize CR. The model is to be a hybrid model because fuzzy logic rules are intended to be used with a CNN model.
- iii. To build and test a CNN model that can predict the severities of a maize leaf disease known as the NCLB by analysing lesion colour and sporulation patterns.

This study follows a quantitative study of designing and developing CNN algorithms that will classify and predict the severities of maize leaf diseases. For instance, in Chapter 3 of this study, the CNN model for classifying various types of maize leaf diseases was set up on a Java Neuroph GUI (general user interface) framework. The CNN in this chapter achieved an average validation accuracy of 92.85% and accuracies of 87% to 99.9% on separate class tests. In Chapter 4, the CNN model for the prediction of CR severities was based on fuzzy rules and thresholding methods. It achieved a validation accuracy of 95.63% and an accuracy 89% when tested on separate images of CR to make severity predictions among 4 classes of CR with various stages of the disease' severities. Finally, in Chapter 5, the CNN that was set up to predict the severities of NCLB achieved 100% of validation accuracy in classification of the two NCLB severity stages. The model also passed the robustness test that was set up to test its ability of classifying the two NCLB stages as both stages were trained on images that had a cigar-shaped like lesions. The three objectives of this study are met in three separate chapters based on published journal papers. Finally, the research objectives were evaluated

against the results obtained in these three separate chapters to summarize key research contributions made in this work.

## **Dedication**

Firstly, I thank God for giving me the courage to live up to my dreams. This thesis is dedicated to my wife, Bongiwe Sibiya, who supported me throughout the research journey and sleepless nights. Without your support, this would have been an impossibility

# Acknowledgements

This PhD thesis represents four years of research, learning, discovery, and collaboration. Without the support of several people and institutions, the thesis will not have been completed. Throughout the research, I received significant support and assistance from the following individuals and institutions:

1. The Council for Scientific and Industrial Research (CSIR) for supporting this project.
2. My supervisor, Prof. Mbuyu Sumbwanyambe for his extraordinary support throughout my PhD study and for his patience and knowledge while allowing me to extend my scientific research abilities. Thank you very much!
3. Mr. Bernardus C. Bothma of the Central University of Technology for his openness and sharing of ideas in programming Deep Learning models.
4. Prof. Kanzumba Kusakana and Prof. James Swart of the Central University of Technology for their pieces of advice on academic writing for journal publications.
5. Dr. Sifiso Xulu, a lecturer at the University of Free State (UFS), QwaQwa campus for the information he shared with me over the past 3 years on environmental sciences.



# Contents

Declaration .....	i
List of Publications .....	ii
Conference Proceedings.....	iii
Abstract .....	iv
Dedication .....	vi
Acknowledgements.....	vii
Contents .....	viii
List of Tables .....	xi
List of Figures .....	xii
List of Abbreviations and Acronyms .....	xvi
CHAPTER 1: Introduction .....	1
1.1 Background .....	1
1.2 Computer Vision System Pipeline .....	11
1.3 Deep Learning for Object Detection and Semantic Segmentation .....	13
1.4 Problem Statement .....	15
1.5 General Aim and Objectives .....	15
1.6 Research Methodology .....	16
1.7 Delimitation .....	17
1.8 Contributions to knowledge.....	17
1.9 Thesis structure .....	17
CHAPTER 2: Literature Review .....	19
2.1 Introduction.....	19
2.2 Underfitting and overfitting in ML and DL models .....	27
2.3 Optimization .....	28
2.4 Metrics for evaluating machine learning classification algorithms .....	29

2.5 Metrics for evaluating ML and DL regression algorithms .....	30
2.6 Chemical laboratory-based methods for plant disease detection.....	30
2.7 Theoretical alternatives.....	31
2.8 Fuzzy logic and Thresholding theoretical concepts.....	37
2.9 Summary.....	43
CHAPTER 3: Convolutional Neural Network for the Classification of Maize Leaf Disease.....	45
Abstract.....	45
3.1 Introduction and Related Works .....	45
3.2 Materials and methods .....	48
3.3 Results.....	53
3.4 Discussion and Conclusion.....	55
CHAPTER 4: Fuzzy Logic Rules for severity estimation of maize disease based on Thresholding .....	57
Abstract.....	57
4.1 Introduction and Related Works .....	57
4.2 Materials and methods .....	64
4.3 Results.....	71
4.4 Discussion and Conclusion.....	73
CHAPTER 5: Use of Convolutional Neural Networks for severity prediction of Northern Corn Leaf Blight based on lesion colour and sporulation.....	75
Abstract.....	75
5.1 Introduction and Related Works .....	75
5.2 Materials and methods .....	79
5.2.1 <i>Data Collection and Pre-processing</i> .....	79
5.3 Results.....	83
5.4 Discussion and Conclusion.....	87
CHAPTER 6: Evaluation of Objectives and Conclusion .....	89
6.1 Evaluating objectives and Key Research Contribution .....	89

Objective 1: To create and test a CNN model that can classify various types of maize leaf diseases .....	89
Objective 2: To set up and test a CNN model that can predict the severities of a maize leaf disease known as the maize CR.....	89
Objective 3: To build and test a CNN model that can predict the severities of a maize leaf disease known as the NCLB by analysing lesion colour and sporulation patterns .....	90
6.2 Recommendations and Suggestions for Future Research.....	90
6.3 Concluding Remarks.....	90
References.....	91

# List of Tables

Table 1. 1: A tabulated summary of a standard CNN Architecture and description of each layer.....	9
Table 1. 2: A tabulated summary of state-of-the-art CNN models.....	9
Table 2. 1: Definition of common non-linear activation functions. ....	23
Table 2. 2: A summary of chemical-based methods for plant leaf disease detection.....	31
Table 2. 3: Tabulated review of deep learning and machine learning approaches for plant leaf disease detection .....	34
Table 3. 1: Accuracy results of the CNN in the classification and recognition of maize leaf diseases and healthy leaves.....	54
Table 3. 2: Table of convoluted results and histogram results of some of the selected test images through the CNN.....	55
Table 4. 1: A summary of deep learning models used in plant disease classification. ....	60
Table 4. 2: A summary of methods for plant disease severity detection. ....	62
Table 4. 3: A tabulated summary of the materials that were used in the study. ....	64
Table 4. 4: Summary for the fine-tuning of pretrained models. ....	70
Table 4. 5: Model summary for a fine-tuned VGG-16 network to predict maize common rust disease severities. .....	70
Table 4. 6: Summary of model hyper parameter tuning and performance metrics. ....	71
Table 5. 1: Tabulated summary of the studies that investigated the use of DL for plant leaf disease detection and the related DL models used.....	78
Table 5. 2: A tabulated summary of the materials that were used in the study. ....	80
Table 5. 3: Parameter settings for model training and validation .....	82

# List of Figures

Figure 1. 1: (a) Northern corn leaf blight lesions are usually larger, tan to grey, and cigar-shaped and (b) The fungus causing northern corn leaf blight can produce large amounts of spores on the surface of lesions, giving them a dark or dusty appearance.....	1
Figure 1. 2: Multiple northern corn leaf blight lesions can develop on leaves and expand and become larger, eventually blighting entire leaves (Jackson-Ziems 2016).....	2
Figure 1. 3: Common rust of corn ( <a href="https://ohioline.osu.edu/factsheet/plpath-cer-02">https://ohioline.osu.edu/factsheet/plpath-cer-02</a> ) .....	3
Figure 1. 4: Diseased maize leaf with grey leaf spot (Ward,1999).....	4
Figure 1. 5: Classification and illustration of ML and DL concepts. ....	5
Figure 1. 6: A single layer feed forward NN. ....	5
Figure 1. 7: A NN with more than one hidden layer is called DL ANN. ....	6
Figure 1. 8: The Jordan network is shown with output activation values that are fed back to the input layer, to a set of extra neurons called the state units (Krose,1993).....	6
Figure 1. 9: A typical Kohonen network .....	7
Figure 1. 10: A standard CNN architecture. ....	8
Figure 1. 11: The human vision system using an eye and brain to sense and interpret an image (Elgendy, 2020). .....	12
Figure 1. 12: The components of a computer vision system used as a sensing device and an interpreting device (Elgendy, 2020).....	12
Figure 1. 13: Steps followed by the interpretation device of computer vision systems to make perceptions of the environment. ....	12
Figure 1. 14: Object detection is trained in detecting soda cans only, and so ignores other cans. ....	13
Figure 1. 15: Multiple instances of the same object are detected by the same model trained to detect one object instance. ....	13
Figure 1. 16: The object detection model was retrained to detect all three objects. ....	14
Figure 1. 17: The semantic segmentation network is viewed as a pixel-wise classifier.....	14
Figure 1. 18: Network architecture of semantic segmentation. ....	15
Figure 1. 19: DSR methodology followed in this study. ....	16
Figure 1. 20: Expanded DSR methodology followed in this study .....	17
Figure 2. 1: The concept of supervised learning.....	20
Figure 2. 2:The concept of unsupervised learning ( <a href="https://subscription.packtpub.com/book/big-data-and-business-intelligence/9781788397872/2/ch021v11sec33/unsupervised-learning">https://subscription.packtpub.com/book/big-data-and-business-intelligence/9781788397872/2/ch021v11sec33/unsupervised-learning</a> ).....	20

Figure 2. 3: ML algorithm selection platform ( <a href="https://www.programmersought.com/article/47434005192/">https://www.programmersought.com/article/47434005192/</a> ). .....	21
Figure 2. 4: Fully connected NN with two hidden layers and one output neuron. ....	23
Figure 2. 5: A typical CNN architecture. ....	26
Figure 2. 6: The convolution operation showing how one element of the feature is computed. ....	26
Figure 2. 7: The concept of Maximum Pooling. ....	27
Figure 2. 8: Overfitting and Underfitting in ML and DL (Patterson and Gibson, 2017). ....	27
Figure 2. 9: A confusion matrix used to explain ML and DL KPIs. ....	29
Figure 2. 10. The overall percentages of DL and ML algorithms that were used for plant leaf disease detection in this review. ....	32
Figure 2. 11: Architecture fuzzy logic system .....	37
Figure 2. 12: Membership functions of the input temperature .....	38
Figure 2. 13: Membership functions of the input temperature at a known temperature of 36 F°. ....	39
Figure 2. 14: A graphical presentation of general properties for fuzzy set membership functions. ....	39
Figure 2. 15: (a) Aorta and blood vessels shown by an MR angiaography image; (b) Intensity histogram of the image in (a);(c) The thresholded binary image with a threshold value $T_1=124$ , pixels labelled 1 (in white) corresponding to objects;(d) A binary image thresholded with a threshold value $T_2 =90$ (Zhou <i>et al.</i> , 2010b). .....	41
Figure 2. 16: (a) A blood vessel image; (b) histogram's intensity of image in (a); (c) The resulting thresholded image when optimal Otsu was used with a threshold value of $T_1=135$ ; (d) The resulting thresholded image when the conventional method was used with a threshold value $T_2=172$ (Zhou <i>et al.</i> , 2010b). ....	41
Figure 2. 17: (a) Original microscopic image of <i>C. aligns</i> , note it has an uneven background illumination; (b) Image intensity histogram (a); (c) Segmentation of image (a) using the Otsu threshold, where the threshold value is 117 and fails to locate all objects from the surrounding background; (d) Segmentation of image (a) using adaptive local thresholding (Zhou <i>et al.</i> , 2010b). ....	42
Figure 2. 18: (a) A grey level image of some randomly placed match sticks; (b) Intensity histogram of image in (a);(c) Multiple thresholded image with corresponding threshold values of $T_1= 45$ and $T_2= 134$ (Zhou <i>et al.</i> , <i>et al.</i> , 2010b). ....	43
Figure 3. 1: Immature GLS lesions on maize leaf appear as small tan spots, often with chlorotic borders. ...	46
Figure 3. 2: Mature GLS lesions on maize leaves are gray to tan in colour and distinctly rectangular in shape. .....	46
Figure 3. 3: Early lesions begin as flecks on leaves that develop into small tan spots (CR disease). ....	46

Figure 3. 4: More advanced disease development with spots of jagged appearance, turning into elongated brick-red to cinnamon-brown pustules (CR disease).....	46
Figure 3. 5: NCLB lesions usually large, cigar-shaped, and tan to gray. ....	47
Figure 3. 6: The fungus can cause NCLB to produce large amounts of spores on the surface of lesions, giving them a dark appearance.....	47
Figure 3. 7: Convolutional neural network architecture. ....	49
Figure 3. 8: Neuroph framework. ....	49
Figure 3. 9: Neural network classifier after the convolution and pooling were completed in the Neuroph library. ....	52
Figure 3. 10: Image recognition and classification test in the Neuroph framework.....	53
Figure 3. 11: GLS ( <i>Cercospora</i> ).....	53
Figure 3. 12: CR ( <i>Puccinia sorghi</i> ).....	53
Figure 3. 13: Healthy. ....	53
Figure 3. 14: NCLB ( <i>Exserohilum</i> ). ....	53
Figure 3. 15: Total network error graph during and after training of the CNN network on a Neuroph framework. ....	54
Figure 3. 16: Assignment of the weights to each of the 50 hidden layers of the CNN. ....	54
Figure 4. 1: The procedure of the proposed approach. ....	65
Figure 4. 2: Procedure explained in Figure 4.1.....	65
Figure 4. 3: Procedure explained in Figure 4.1, continued.....	65
Figure 4. 4: Assignment of maize common rust disease images to their severity classes using fuzzy decision rules. ....	67
Figure 4. 5: A sample of healthy maize images that were used for training in the Healthy class. ....	67
Figure 4. 6: The final arrangement of the training, validation, and test data sets for the prediction of the maize common rust disease severities by a fine-tuned VGG-16 network.....	68
Figure 4. 7: The VGG-16 architecture. ....	69
Figure 4. 8: Training loss against validation loss plots.....	72
Figure 4. 9: Training accuracy against validation accuracy plots.....	72
Figure 4. 10: The number of correctly classified images in each class of the VGG-16 network. ....	72
Figure 4. 11: A comparison of 2-dimensional array images with 3-dimensional array images. ....	73

Figure 5. 1: Field images of the NCLB in the Late Stage: (a) Large amounts of spores on the surface of lesions, with a dark or dusty appearance because of the fungus causing the NCLB; (b) As the lesions mature, their colour change to tan and the production of fungal spores has become visible in the middle of the lesions. ..76

Figure 5. 2: Field image of the NCLB on the Early Stage with.....76

Figure 5. 3: The tags were used to mark the maize crops from ..... which the NCLB images were taken. ....79

Figure 5. 4: Arrangement of the training and validation data sets for the CNN model intended to make severity predictions of the NCLB. ....79

Figure 5. 5: CNN model architecture for predicting the NCLB severities. ....81

Figure 5. 6: The general confusion matrix used to understand model performance key indicators. ....83

Figure 5. 7: Comparison plots of validation and testing accuracies of a model described in experiment 1....84

Figure 5. 8: Confusion matrix and metrics classification report of the model described in experiment 1. ....85

Figure 5. 9: Comparison of the training and validation accuracies in the model’s realistic performance point of view. ....86

Figure 5. 10: Confusion matrix and metrics classification report of the model described in experiment 2. ....87



## List of Abbreviations and Acronyms

ANN - artificial neural network  
ARC - agricultural research council  
AI - artificial intelligence  
CNN - convolutional neural network  
CR - common rust  
DL - deep learning  
DNN - deep neural network  
DSR - design science research  
ELISA - enzyme-linked immunosorbent assay  
FCM - flow cytometry  
FISH - fluorescence *in-situ* hybridization  
GAN - generative adversarial network  
GC-MS - gas chromatography-mass spectrometry (GC-MS)  
GLS - gray leaf spot  
GUI - graphical user interface  
IDE - integrated development environment  
IF - immunofluorescence  
ML - machine learning  
NCLB - northern corn leaf blight  
NN - neural network  
PCR - polymerase chain reaction  
PKIs - performance key indicators  
R-CNN - region based convolutional neural network  
SVM- Support Vector Machines  
IEEE- Institute of Electrical and Electronics Engineers  
SOM-Self Organising Network

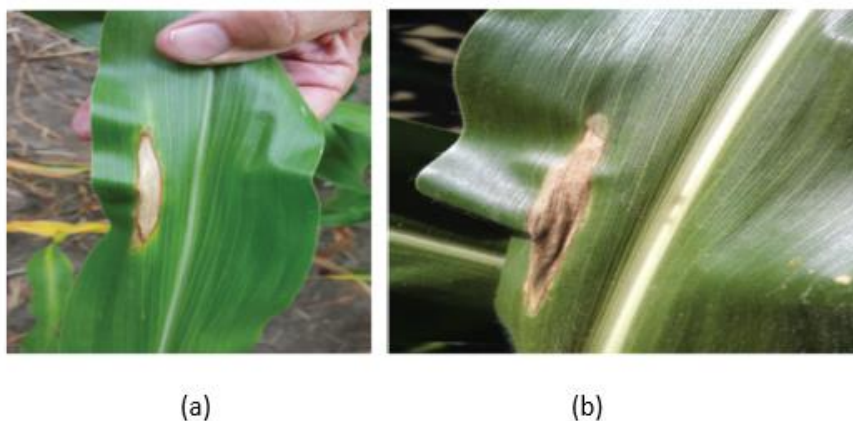
# CHAPTER 1: Introduction

## 1.1 Background

Plant diseases usually result in a reduced crop yield and that in essence affects the economies of many countries. For instance, in the works of Sankaran *et al.* (2010) it is highlighted that soybean rust, which is a fungal disease had caused a substantial economic loss, of which by just removing 20% of the infection would have benefitted the farmers a total of USD 11 million profit. There are so many ways of detecting and preventing plant leaf diseases in order to avert disastrous results like the one put forward by Sankaran *et al.* (2010). The most common of these are the chemical-based methods. However, there are limitations in terms of applying such chemical-based methods. The chemical-based methods, though conducive, are tedious and laborious in nature to solving the problem of detecting various plant diseases in commercial fields. As a matter of fact, such measures will require laboratory procedures which may be time consuming and costly. By and large, there are several diseases associated with maize (corn) but the most prevalent ones are the common rust (CR), northern corn Leaf Blight (NCLB) and gray leaf spot (GLS). These three diseases affect *Zea mays* (maize) crop yield in different ways and will be discussed in this thesis separately.

### 1.1.1 Northern Corn Leaf Blight (NCLB) Symptoms and Epidemiology

Northern corn leaf blight (NCLB) is a fungal disease of maize or corn (note that in this thesis corn or maize will be used interchangeably) and is caused by a fungus called *Exserohilum turcicum* (Jackson-Ziems 2016). The fungus is active and grows well in cold to moderate temperatures and high relative humidity. Studies indicate that NCLB is present in most of South Africa's maize-producing areas. The disease is also sporadically common in other moist maize-producing regions of the world. Yield loss due to this disease can be significant, up to 30-50% in susceptible hybrids when the disease develops early in the season, prior to the appearance of tasselling (Jackson-Ziems 2016). However, when the severity of the disease is minor or its development is delayed until well after silting, the effects on yield are usually minimal.



**Figure 1. 1:** (a) Northern corn leaf blight lesions are usually larger, tan to grey, and cigar-shaped and (b) The fungus causing northern corn leaf blight can produce large amounts of spores on the surface of lesions, giving them a dark or dusty appearance.

NCLB can be identified by relatively large grey or greenish elliptic or cigar form lesions that can develop on leaves, sheaths, or leaf sheaths (Figure 1.1). Lesions can range from 2.5 centimetres to over 17.5 centimetres in length and are oriented parallel to the foliar veins. Some maize hybrids may or may not result in dark-edged lesions. These are not limited by leaf veins in order to infect the plant. As the lesions mature, their colour can change to tan and the production of fungal spores can become visible in the middle of the lesions, resulting in a darker, dustier appearance. Because the lesions of the NCLB may appear similar to those of Goss's bacterial wilt and blight, an accurate diagnosis to effectively manage and minimize losses caused by the disease is necessary. The disease usually develops in the form of some scattered lesions in the lower canopy, which eventually develops lesions in the upper canopy, if the favourable conditions persist. One or more lesions may form on a leaf (Figure 1.2) and increase in size, frequently causing coalitions (union) and blights on larger areas or entire leaves. Symptoms may differ in self-pollinated maize seed varieties or in some resistant hybrids. For instance, lesions may appear smaller, yellow and/or spore-free in resistant hybrids (Jackson-Ziems 2016). The fungus that causes NCLB winters in infected leaves, sheaths and cobs from previous years. Spores produced on the residue or on diseased plants in the field can be transferred to new leaves higher on the plants or blown by the wind over long distances to neighbouring fields. Spores require 6-18 hours of water at the leaf surface for the germination and to some extent infect the leaf. The disease occurs more frequently during periods of high relative humidity and mild temperatures, which promote the production and germination of fungal spores. The growth of lesions takes 7 to 12 days after infection, depending on the hybrid susceptibility. The development of this disease may be unnoticed or unrecognised for one to two weeks when supported by favourable weather conditions. When this happens, lesions develop and may not be noticed until later in the latent phase. The severity of the disease (which is part of our study) increases as the lesions extend and grow, reducing the photosynthetic area, which reduces the filling (the period between anthesis and physiological maturity) and grain yield (Jackson-Ziems 2016).



**Figure 1. 2:** Multiple northern corn leaf blight lesions can develop on leaves and expand and become larger, eventually blighting entire leaves (Jackson-Ziems 2016).

### 1.1.2 Common Rust Symptoms and Epidemiology

Although some rust blisters may still be found in corn fields throughout the growing season, symptoms usually do not appear until after tasselling. These are distinguishable from other diseases by the development of darkish and reddish-brown pustules (*urodynia*) scattering over both the lower and upper surfaces of the maize leaves (Fig. 1.3) (Campus,2012). The pustules appear oval to elongated, and are usually small, less than 0.6 centimeters long, and are encircled by the epidermal layer of the leaf. If infections occur, when the leaves are still in the whorl, the pustules can develop in bands on the surface as the leaf expands in size (Dilliard, 1990).



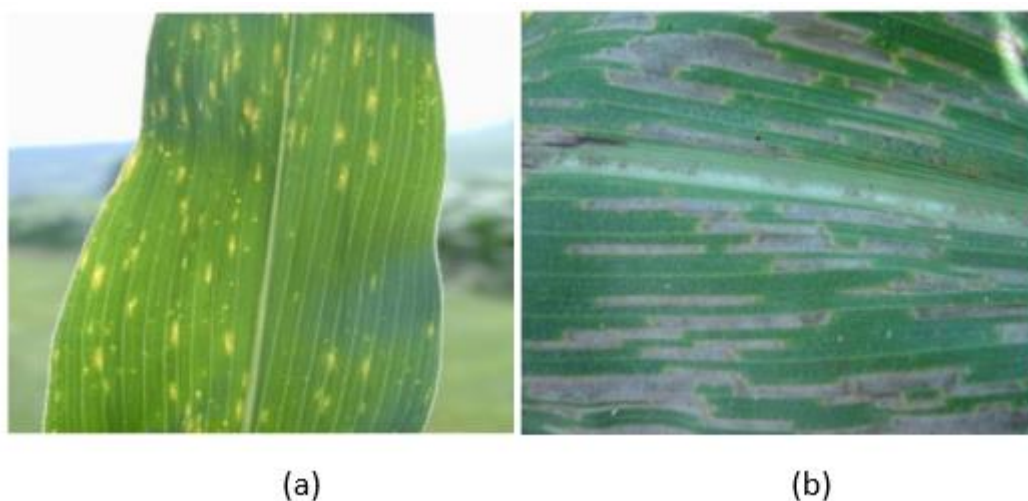
Figure 1. 3: Common rust of corn (<https://ohioline.osu.edu/factsheet/plpath-cer-02>)

In contrast to most other corn leaf diseases, this rust fungus does not wait out the winter season (overwintering) in plant residues. More often than not, spores are dispersed during the growing season where this fungus survives on corn or wood sorrel, the alternative host. Young leaves are generally more sensitive to infections than older leaves. The development and spread of rust are favoured by extended periods of cool temperatures ranging from 15° to 23° and high relative humidity (Dilliard, 1990). As a result, pustules develop on susceptible maize hybrids within 7 days of infection. Occasionally, chlorosis and death of leaves, and leaf sheaths occur with severe infections. The uredospore produced during the season are spread by the wind, spreading the pathogen to new leaves, plants and fields. As the maize plant matures, the pustules become brownish and blackish because of the development of darker pigmented teloids that replace urodynia and produce teliospores (Campus,2012).

### 1.1.3 Gray Leaf Spot Symptoms and Epidemiology

GLS of maize is caused by the fungus *Cercospora zea-maydis*. The disease is now recognised as one of the most maize yield limiting diseases in the world and certainly in the KwaZulu-Natal province, South Africa (Ward and Nowell, 1994). Not only is it a threat to corn production in commercial agriculture, but it also reduces maize yields on small farms. It was first identified in KwaZulu-Natal in 1989/90 and has since spread to neighbouring provinces and most maize-producing African countries (Ward and Nowell, 1994). Initial symptoms usually appear on the lower leaves of the maize plant. The immature lesions first appear as small tan spots about 1 to 3 mm in size and are irregular in shape. Tanning spots typically have yellow or chlorotic

borders and are easier to observe when the leaf is held towards the light (see Figure 1.4 (a)). Mature lesions are distinguished from other symptoms of the pathogen and are clearly rectangular in shape (5 to 70 mm long and 2 to 4 mm wide) and run parallel with leaf veins (Figure 1.4 (b)).



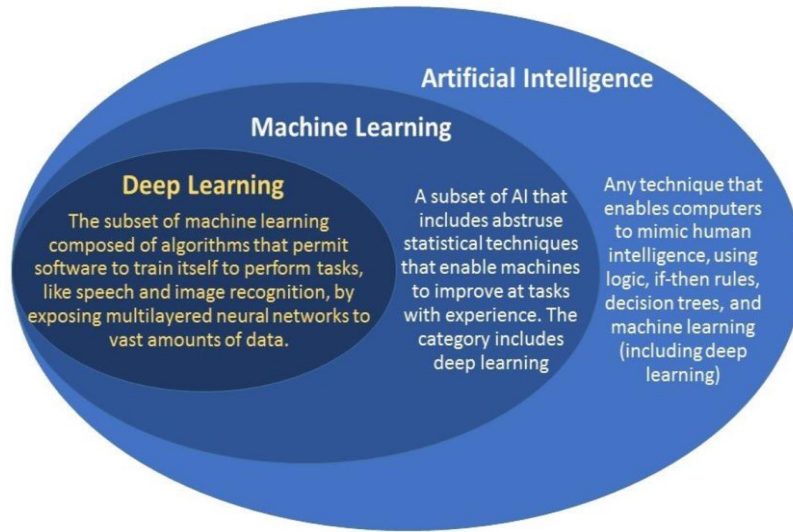
**Figure 1. 4:** Diseased maize leaf with grey leaf spot (Ward, 1999).

The GLS is highly dependent on favourable weather conditions. It requires frequent and prolonged periods of high humidity and warm temperatures (20°C to 30°C) to complete spore germination and the infection process. Spores (also known as conidia) are produced from infested residues of previous maize crops in spring under conditions of high humidity and these are windblown to infect the newly planted maize crop. The lower leaves are usually the site of primary infection.

#### ***1.1.4 Overview of Machine Learning Techniques***

Machine learning (ML) is a form of data processing that automates the development of analytical models. It is a branch of artificial intelligence focused on the premise that computers can learn from data, recognize patterns, and make decisions with little to no human input. Generally, there is a significant amount of literature of ML algorithms which can be classified according to the approach used in the learning process. The four major learning classes are supervised, unsupervised, semi-supervised and reinforcement learning (Aurélien, 2017). Supervised learning happens when ML algorithms get training data matched with the desired output. At this stage, the ML algorithm learns from the training data, and use the features learned to make predictions on unknown data. Unsupervised ML algorithms are not equipped with training data sets. In the case of unsupervised learning, real-world data is fed to algorithms and must learn from this data on its own. When classified as semi-supervised learning, ML algorithms work with a training dataset containing missing information, and still must learn from it. Lastly, the ML algorithms might have a reinforcement learning approach which occurs when the algorithms learn based on external feedback given either by a thinking entity, or the environment. An example of the ML algorithm using reinforcement learning could be the ML algorithm that plays games against a human opponent. Deep learning (DL) is a subfield of machine learning in the

artificial intelligence context that employs neural networks (NN) layers to learn from the training data. Figure 1.5 illustrates the concept of ML and DL.



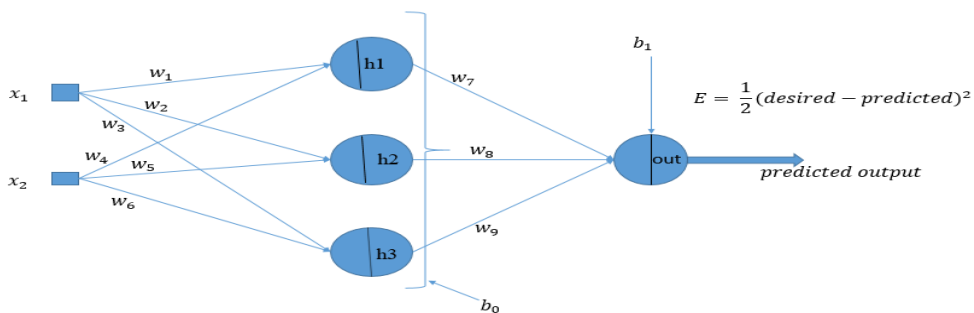
**Figure 1. 5:** Classification and illustration of ML and DL concepts.

There are three important types of NN as listed below:

- i. Artificial Neural Network (ANN): A typical ANN is dealt with in section 1.1.4.1
- ii. Convolutional Neural Network (CNN): This employs ANN with convolution and pooling layers. The type of data these can handle are images. An example of a typical CNN is dealt with in section 1.1.4.2.
- iii. Recurrent neural Network (RNN): These have a recurrent connection to the hidden state. They handle time series data, text data and audio data. An example of a typical RNN is dealt with in section 1.1.4.3.

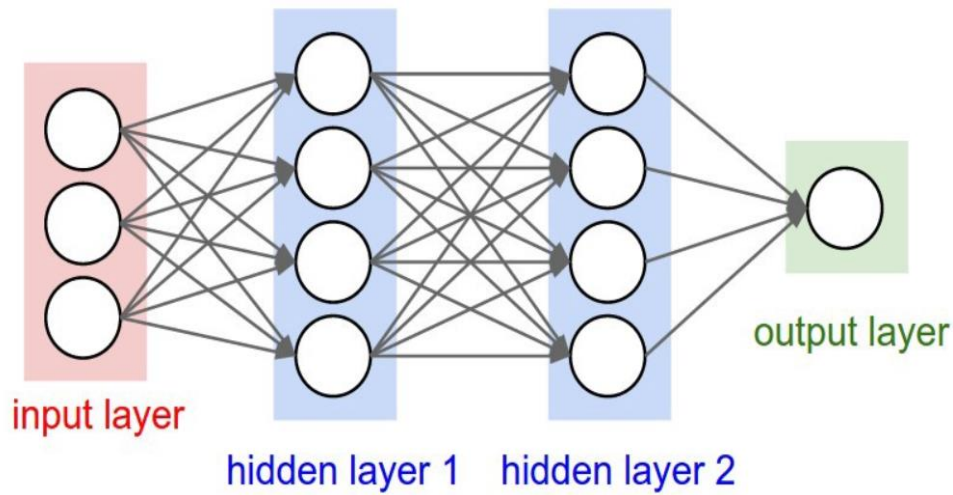
**1.1.4.1 Artificial Neural Networks**

A single layer feed forward artificial neural network (ANN) is shown in Figure 1.6



**Figure 1. 6:** A single layer feed forward NN.

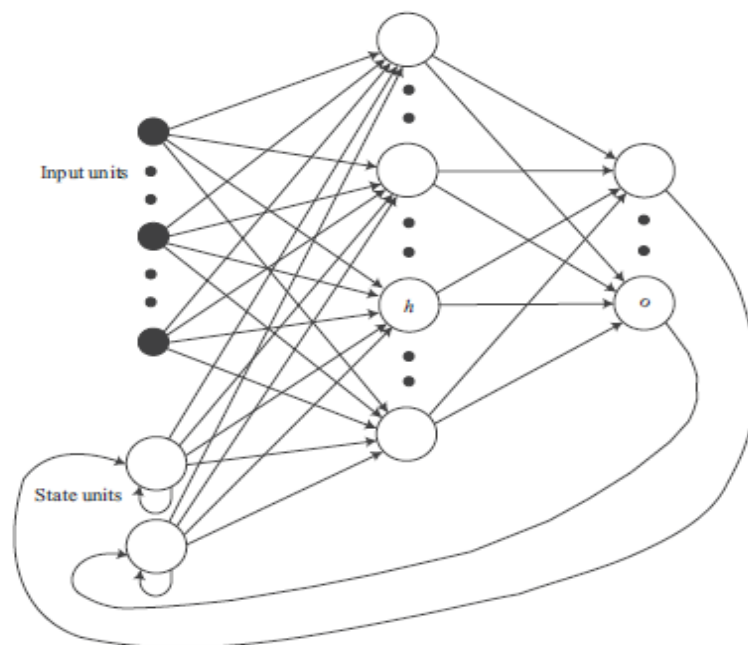
The word deep learning becomes significant when the hidden layers are added in the ANN shown in Figure 1.6. This implies that DL NN models have more than one hidden layer. An example of a DL ANN is shown in Figure 1.7.



**Figure 1. 7:** A NN with more than one hidden layer is called DL ANN.

### 1.1.4.2 Recurrent Neural Networks

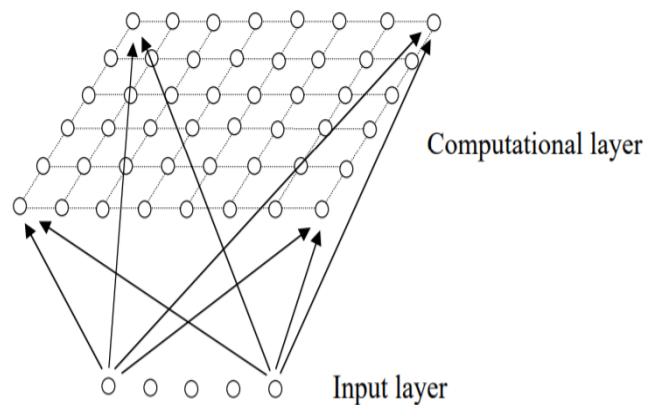
The most famous type of recurrent neural network (RNN) is the Jordan network. The Jordan network is a sort of repetitive (recurrent) neural system whereby the activation values of the output units are continually fed into the input layer through an arrangement of extra input units called the state units. There is a constant amount of state units as there are output units in the network. A set weight of 1 is available in associations between outputs and state units. This means that learning occurs only in the associations between the inputs and the hidden units as well as the hidden and output units. Figure 1.8 shows the RNN type called the Jordan network.



**Figure 1. 8:** The Jordan network is shown with output activation values that are fed back to the input layer, to a set of extra neurons called the state units (Krose,1993).

### 1.1.4.3 Self-Organising Neural Networks

These kinds of neural networks learn without the supervision of an external teacher. The unsupervised weight-adapting algorithms are usually based on some form of global competition between the neurons. These networks use competitive learning to learn. A particular kind of self-organising network (SOM) is known as Kohonen network. It consists of a feed forward structure that has a single computational layer arranged in columns and rows. Every neuron is full connected to all the source nodes in the input layer. Figure 1.9 shows a typical self-organising Kohonen network.



**Figure 1. 9:** A typical Kohonen network

The self-organisation process is governed by four major components named as follows:

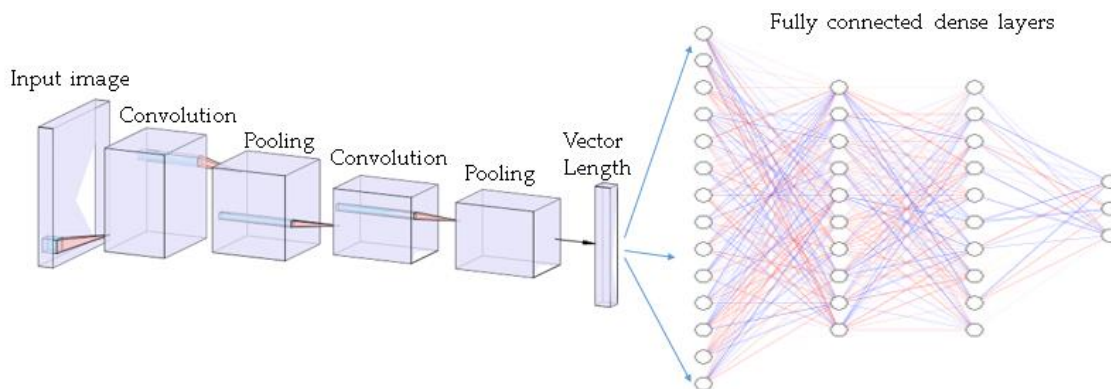
- i. Initialization: Small random values are used to initialize all the connection weights.
- ii. Competition: A neuron with the smallest value of the discriminant function is the one that is declared the winner.
- iii. A neuron that wins determines the spatial location of a topological neighbourhood of excited neurons, and therefore providing the basis for cooperation among neighbouring neurons.
- iv. Adaptation: The excited neurons decrease their single values of the discriminant function related to the input pattern through recommended adjustment of the associated connection weights, in a way that makes the response of the winning neuron to the subsequent application of a similar input pattern enhanced.

### 1.1.4.4 Machine Learning and Convolutional Neural Networks Applied in Plant Disease Classification.

A CNN is a type of artificial neural network used in image recognition and processing which is specially designed for processing pixels. A CNN utilizes a system much like a multi-layer perceptron that has been designed to reduce processing requirements. The layers of a CNN are made up of an input layer, an output layer and a hidden layer that includes several convolutional layers, bundled layers, fully connected layers, and normalization layers. The removal of limitations and increase in efficiency for image processing results in a system that is far more effective, simpler to train for image processing. A standard architecture of a CNN is



shown in Figure 1.10. It consists of layers that serve different purposes in extracting the features of the input image. The standard CNN architecture shown in Figure 1.10 is a basis of many state-of-the-art CNN architectures that are available in the literature. Among a few, these include widely used CNN models such as the VGG-16 and Google net. An input image can be either a colour image or grey image. These input images are tensors of shape (image height, image width, image channels). Colour images have three image channels of red, green and blue normally expressed as RGB channel.



**Figure 1. 10:** A standard CNN architecture.

The pixels for each of these channels can be varied from 0 to 255 to express the channel intensities. Grey images have only one channel that its pixels can be varied from 0 to 255. Grey to light-grey intensities can be achieved when the intensity settings approach 255, whereas darker intensities are achieved towards 0 intensity settings. The purpose of convolution is to extract features from an input image by sliding a filter along the windows of the image. Pooling merges semantically similar features into one, by preserving task-related information while removing redundant information. In a broader sense, the function of pooling is to continually reduce the image dimensionality while preserving important features, and hence reducing the number of parameters and computations in the network. This shortens the training time and controls overfitting. Fully connected dense layers learn the different features of the images and have special activation functions. Table 1.1 summarizes the layers of a standard CNN architecture.

Table 1. 1: A tabulated summary of a standard CNN Architecture and description of each layer.

CNN layers by order		Description of each layer
<b>Layer 1</b>	Input Reception	Input layers are where the raw input data of the image are loaded for processing in the network. The raw input data of the images could be in the format RGB (W x H x 3) or Greyscale (W x H).
<b>Layer 2</b>	First Convolutional layer and number of filters.	The convolutional layers are considered the core building blocks of CNNs. These convolutional layers transform the input data by using a patch of locally connecting neurons from the previous layer. The convolutional layer computes a dot product between the region of the neurons in the input layer, and the weights to which they are locally connected in the output layer. This will result to an output layer with the same, and smaller spatial dimensions. Filters perform the dot matrix with the input to get the activation maps. This is achieved through the process called convolution.
<b>Layer 3</b>	ReLU activation function	The ReLU layer applies an activation function that is element wise over the input data thresholding for example, $\max(0, x)$ at zero, that gives the same dimension output as the input to the layer. However, applying the ReLU function over input volume will only change the pixel values, but not the spatial dimensions of the input data in the output. ReLU layers do not possess parameters nor additional hyperparameters.
<b>Layer 4</b>	Pooling	Pooling layers are often placed between successive convolutional layers. The aim is to follow convolutional layers with pooling layers such that the spatial size (width and height) of the data representation is progressively reduced. Pooling layers reduce the data representation progressively throughout the network and that helps control overfitting. The pooling layer independently operates on every input's depth slice.
<b>More Convolutional layers and Pooling layers may be added depending on the required feature extraction effort.</b>		
<b>Layer 5</b>	Dense layers and activation functions	Dense layers compute class scores that will be used as output of the network. These are fully connected layers that perform transformations on the input data volume that are a function of the activations such as sigmoid, ReLU or Tanh in the input volume and the parameters (weights and biases of the neurons).

Some researchers were able to update the standard CNN architecture to the state-of-the-art CNN models. The state-of-the art CNN models proved to be having improved accuracies and were adopted by many industries for application in the business areas. The state-of-the-art CNNs became popular by virtue of the fact that they are reusable for classification applications they were not initially developed for. A select few of popular state-of-the-art CNN models are reviewed in Table 1.2 with the corresponding studies they were published in.

Table 1. 2: A tabulated summary of state-of-the-art CNN models

State-of-the-art CNN models	Study
VGG16	(Simonyan and Zisserman., 2014)
VGG19	(Simonyan and Zisserman., 2014)
Inception.	(Szegedy <i>et al.</i> , 2015)
Alex Net.	( Krizhevsky <i>et al.</i> , 2012)
MobileNet	(Howard <i>et al.</i> , 2017)
ResNet50	(He <i>et al.</i> , 2016)

Indeed, using artificial intelligence, with inclination to CNNs, modern robots with high tech computer vision would carry out an equivalent of such a job at a fraction of the cost and time. The CNNs proposed in this study are especially recommended for use in poor countries where there is a substantial lack of expertise and infrastructure to get timely information. Also, the chemical methods are tedious as they are microscopic and always require special equipment for necessary tests to be performed. ML and DL hold much promise to

address this problem and this study is devoted to test their performance in this regard. In this study, ML is simply explained as the ability of computers to learn without exactly making them learn through the hard coding. Popular ML categories are supervised learning and non-supervised learning. Supervised learning is the machine learning category of surmising a function from the labelled training data. In supervised learning, each example is a pair consisting of an input object (typically a vector) and the desired output value (also called the supervisory signal) (Caruana and Niculescu-Mizil, 2006). A supervised learning algorithm examines the training data and produces a prediction function, which can be utilized for mapping new examples. An ideal situation will take into account the algorithm to accurately decide the class labels for unseen occasions. This requires the learning algorithm, to sum up from the training data to unseen circumstances in a “sensible” manner. Unsupervised learning is that form of learning that endeavours to locate a concealed structure in the unlabelled data. Since the examples given to the learning algorithm are unlabelled, there is no error to assess a potential solution. For instance, and in line with this statement, Sladojevic *et al.*(2016) used deep learning to develop a model that was able to recognize 13 different types of plant leaf diseases out of healthy leaves, with the ability to distinguish plant leaves from their surroundings. Zheng *et al.* (2018) collected the canopy hyperspectral data of healthy wheat, that was wheat in the incubation period, and wheat in the diseased period in order to investigate a method to identify the variation in the two diseases. After data pre-processing, they built, three support vector machine (SVM) models for disease identification and six support vector regression (SVR) models for disease index (DI) inversion. The results showed that the SVM model based on wavelet packet decomposition coefficients with the overall identification accuracy of the training set was equal to 99.67% and that of the testing set equal to 82.00% was better than the other two models. Algorithms for extraction of colour and texture features were developed, which were in turn used to train support vector machine (SVM) and artificial neural network (ANN) classifiers (Pujari *et al.*, 2016). Pujari *et al.*(2016) presented in their work, a reduced feature set based approach for the recognition and classification of images of plant diseases. Ahmadi *et al.* (2017) used artificial neural network (ANN) analytical technique for discriminating and classifying fungal infections in oil palm trees at an early stage using raw, first, and second derivative Spectroradiometer datasets. Ahmadi *et al.* (2017) used machine learning for plant stress severity rating in soya beans. Naik *et al.* (2017) investigated 10 different classification approaches, with the best classifier being a hierarchical classifier with a mean per-class accuracy of ~96%. Other researchers such as, Chaudhary *et al.* (2016) presented an improved-RFC (Random Forest Classifier) approach for multi-class disease classification problem. In their approach, they used a combination of Random Forest machine learning algorithm, an attribute evaluator method and an instance filter method. The performance results confirmed that their proposed improved-RFC approach performed better than the Random Forest algorithm with the increase in disease classification accuracy up to 97.80% for multi-class groundnut disease dataset. The evolution of Artificial Neural Networks (ANNs) has found use in many applications in various areas of agriculture. For instance, a deep learning (DL) Convolutional Neural Network(CNN) architecture was applied

in the analysis of water pollution for agricultural irrigation resources(Chen *et al.*, 2020). Among these applications, there is a rapidly growing literature on use of DL for plant disease detection, which indicates that DL is a promising tool for the detection of plant diseases. Experiments on use of DL for classification of maize leaf disease from healthy leaves were carried out by Sibiya and Sumbwanyambe (2019a). Further studies on use of DL were performed by Sibiya and Sumbwanyambe (2021) to predict the severities of a maize leaf disease called the Common Rust. Seemingly, the use of CNNs for plant disease detection and severity by Sibiya and Sumbwanyambe (2019a, 2021) focused on maize leaves. There is no enough evidence in the literature for the use of DL to detect plant stem and root diseases, however, there is a large body of evidence for the detection of fruit diseases (Nikhitha *et al.*, 2019; Nasiri *et al.*, 2019).

## 1.2 Computer Vision System Pipeline

Visual ability with computers was inspired by the human vision system (Elgendy, 2020). A vision pipeline consists of two components for both human and computer vision systems. These components are called the sensing device and an interpreting device.

**Sensing device-human vision system:** An eye is a sensing device in case of a human vision system and is responsible for capturing images of the environment.

**Sensing device-computer vision system:** A camera is a good example in this regard. However, other examples of sensing devices for computer vision may be applicable depending on the AI for computer vision concerned. For instance, a camera, radar, X-ray, CT scan, Lidar, or a combination of devices may be used to provide the full scene of an environment to fulfill the task at hand. A combination of these may be found in autonomous vehicles, commonly known as driverless cars.

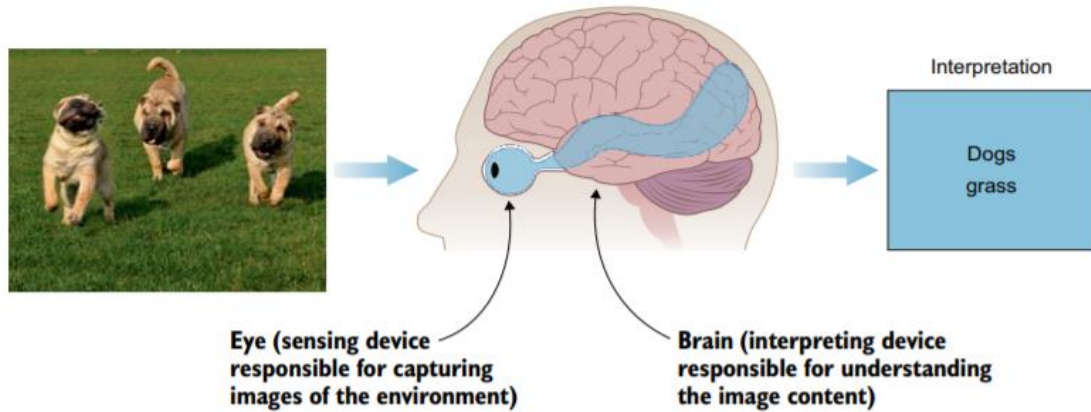
**Interpreting device-human vision system:** Brain is the interpreting device responsible for understanding the image content.

**Interpreting device-computer vision system:** CNNs use stages to learn about the contents in the image. These stages, when combined form the computer vision system. The four stages that form the computer vision system are explained in logical order as follows:

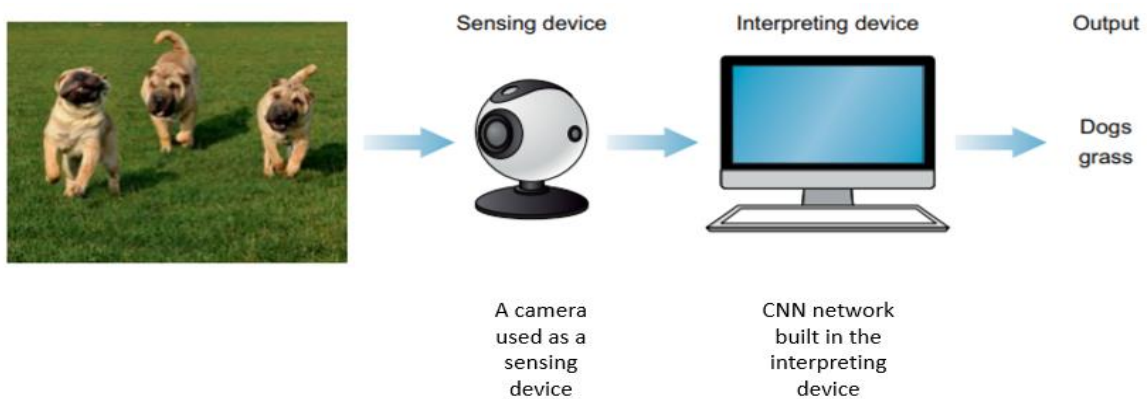
- i. **Input image:** This is achieved by means of sensing devices depending on the task at hand.
- ii. **Image processing:** In the case of CNNs, image processing can be achieved by performing various image processing techniques such as rotate, blur, edge enhancements and many others. These image processing techniques are performed to improve CNN's performance while avoiding overfitting.
- iii. **Extraction of features:** CNNs employ convolutions to extract features. Pooling is used to improve feature extraction. The convolutions and pooling may be available at several stages throughout the CNN network depending on the extent to which features must be extracted.

- iv. **A ML/CNN model used to interpret images:** The choice of the model used for making interpretations needs to be considered. For instance, a user defined CNN model is likely to underperform when compared to any of state-of-the-art models such as VGG-16 and Inception to mention a few.

Figures 1.11 and 1.12 illustrate the comparison of the human vision system with computer vision system, respectively.

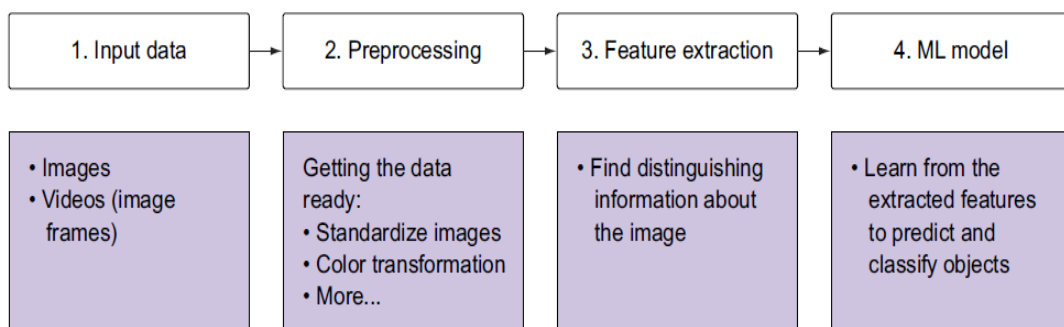


**Figure 1. 11:** The human vision system using an eye and brain to sense and interpret an image (Elgendy, 2020).



**Figure 1. 12:** The components of a computer vision system used as a sensing device and an interpreting device (Elgendy, 2020).

Figure 1.13 illustrates the summarized steps of the interpretation device in computer vision systems.



**Figure 1. 13:** Steps followed by the interpretation device of computer vision systems to make perceptions of the environment.

Step 3 in Figure 1.13 shows the feature extraction process. CNNs use an automated method for feature extraction called convolution. Traditional methods of feature extraction are histogram of oriented gradients (HOGs), Haar cascades, the scale invariant feature transform (SIFT), and speed up robust feature (SURF).

These traditional methods of feature extraction are hand crafted and passed to classifiers such as support vector machines and adaboost.

### 1.3 Deep Learning for Object Detection and Semantic Segmentation

#### 1.3.1 Object Detection

Object detection is a subject of computer vision whereby the algorithm used is capable of localizing and identifying the object. Depending on the number of objects to be identified in an image, object detection model is capable of detecting several objects in an image as long as it was trained to detect them. For instance, it can be seen in Figure 1.14 that a coca soda is only detected as the model was trained to only detect coca sodas. Also, Figure 1.15 illustrates a multi-detection of objects of the same class regardless of the number of times they appear in the image. Bounding boxes are used to localize and identify the objects that are supposed to be detected by a model.

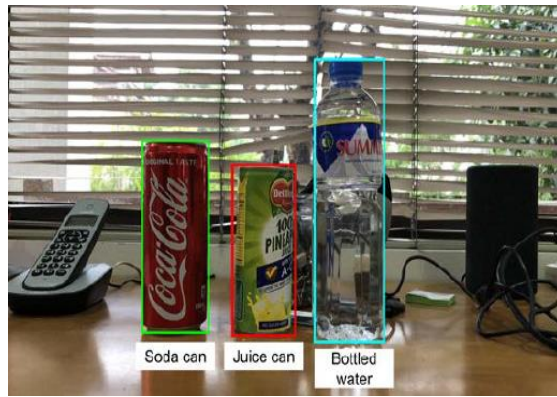


Figure 1.14: Object detection is trained in detecting soda cans only, and so ignores other cans.



Figure 1.15: Multiple instances of the same object are detected by the same model trained to detect one object instance.

Figure 1.16 illustrates a multi-detection of different class objects, when the model was trained to also detect the other two objects along with coca soda.

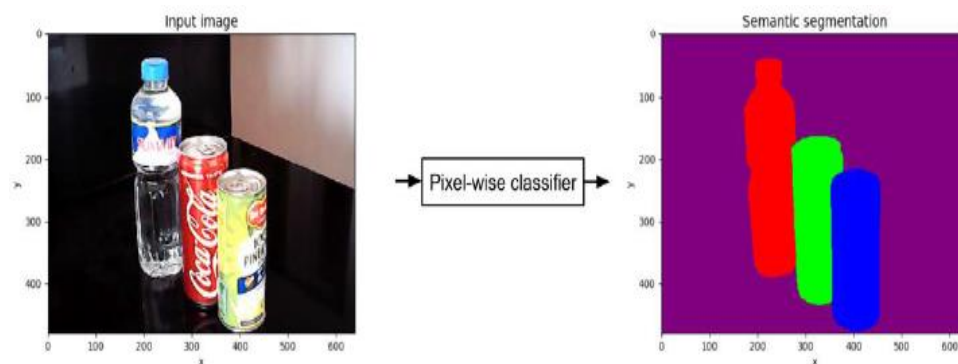


**Figure 1.16:** The object detection model was retrained to detect all three objects.

Object detection models are categorized into two types. The first type is the one that recognizes the regions of the object to detect the object. Examples of the latter mentioned model are R-CCNs and Faster R-CNNs. The second category uses anchor boxes that are spread all over the image while taking regression with respect to the ground truth bounding box of the object. Loss function is used to determine the box to be used as a bounding box of the object to be detected. Known models of the latter mentioned type of object detection are SSD and YOLO.

### 1.3.2 Semantic Segmentation

Semantic segmentation is no different from object detection explained in section 1.3.1, except that it uses pixels to localize and identify objects in an image. Figure 1.17 shows a pixel wise classifier using semantic segmentation.



**Figure 1.17:** The semantic segmentation network is viewed as a pixel-wise classifier.

The network architecture of semantic segmentation is shown in Figure 1.18. It has an RGB input image (for example, 640 x 480 x 3) and its output is a tensor with similar dimensions except that the last dimension is the number of object categories (for example, 640 x 480 x 4 for a 4-object category). For visualization purposes, the output is mapped into RGB by assigning a colour to each category.

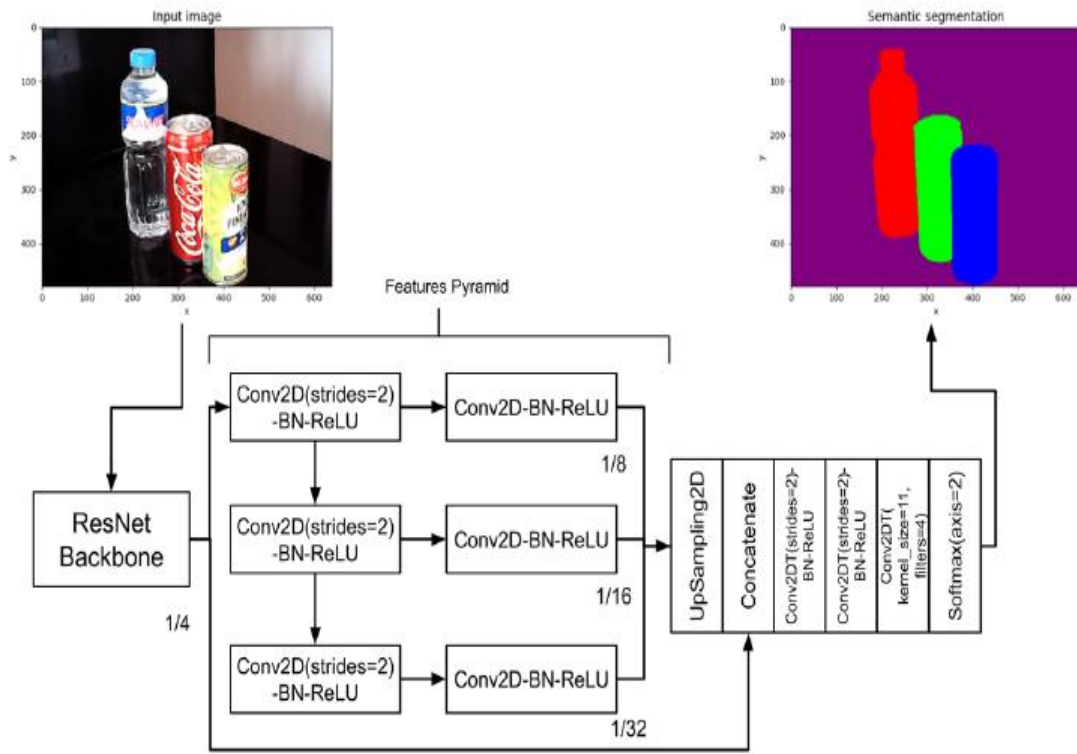


Figure 1.18: Network architecture of semantic segmentation.

## 1.4 Problem Statement

Maize diseases such as the NCLB have mostly affected the KwaZulu-Natal areas, threatened this industry, and resulted in a reduced maize yield. The chemical-based methods and other manual laboratory techniques, though conducive, are tedious and laborious in nature to solving the problem of detecting various plant leaf diseases in commercial fields as modern robots with high tech computer vision would, more especially in poor countries where there is a substantial lack of experts and infrastructure to get timely information. CNNs hold much promise to address this problem and this study is devoted to test their performance in this regard.

## 1.5 General Aim and Objectives

The general aim of this study is to introduce CNN approaches into classifying various types of maize leaf diseases (GLS, CR and NCLB), and to predict the severities of CR and NCLB thereof. Therefore, the three specific objectives of this study are listed below:

- i. To create and test a CNN model that can classify various types of maize leaf diseases.
- ii. To set up and test a CNN model that can predict the severities of a maize leaf disease known as the maize CR. The model is to be a hybrid model because fuzzy logic rules are intended to be used with a CNN model.
- iii. To build and test a CNN model that can predict the severities of a maize leaf disease known as the NCLB by analysing lesion colour and sporulation patterns.



The second and third objectives would lead to working models that would be used to classify and predict the severity of maize leaf diseases.

## 1.6 Research Methodology

In this study, a design and experimental procedure were followed in order to achieve the objectives set out in section 1.5. Primarily, this methodology followed that of the design science research (DSR) which is arranged in the following manner:

- i. **Data acquisition:** Mining of image data sets involved getting access to the PlantVillage repository. Other images for CR and NCLB were collected by means of a camera.
- ii. **Data cleaning and pre-processing:** The images were sorted and arranged according to their training classes and validation classes. This step was applied to all CNNs that were set up in this study.
- iii. **Model training and building:** The CNNs were trained at different epoch values on different architectures that were built using the Python-Keras library. The custom CNN architecture was set up for a model that was meant to classify various types of maize leaf diseases. The VGG-16 state-of-the-art network was used for setting up a model that predicted the severities of a maize CR disease. Once again, the custom CNN was set up to predict the NCLB severities.
- iv. **Testing data and model testing:** All the CNNs in this study were validated using the validation data sets that were used to observe the models' performances during the training process. This was meant for observing the possible occurrences of overfitting and underfitting in CNNs while they were training. To test the CNNs for robustness, separate class tests were performed to check the testing accuracies.

Figures 1.19-1.1.20 show the DSR methodology that will be followed in this study.

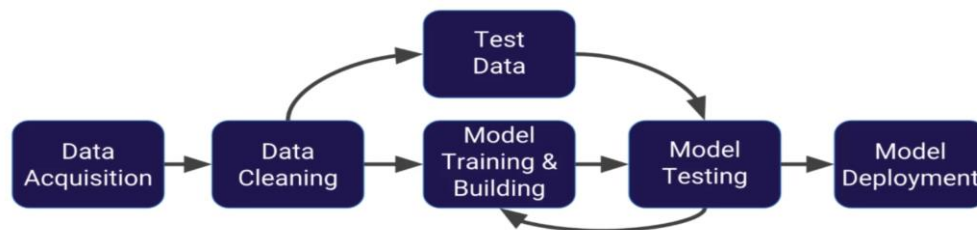


Figure 1. 19: DSR methodology followed in this study.

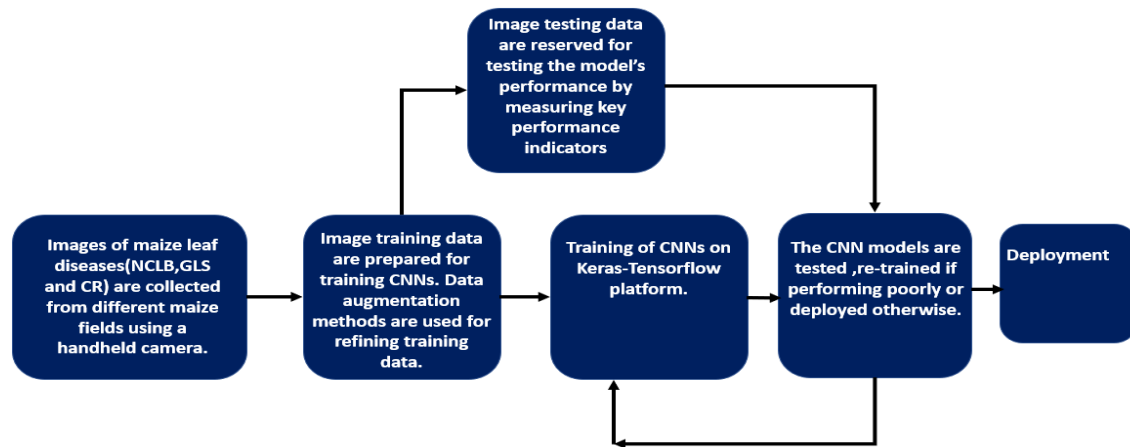


Figure 1. 20: Expanded DSR methodology followed in this study

The methodological approach used in each of the chapters that form this thesis follows the DSR methodology explained in Figures 1.19-1.20. As this is a thesis per publication, there is no separate methodological chapter because the methodologies used are explained in each chapter of the published literature.

## 1.7 Delimitation

This research work did not consider the following:

- i. Classification or severity prediction of images with maize stem diseases.
- ii. Classification or severity prediction of images with maize kernel diseases.

## 1.8 Contributions to knowledge

- i. Facilitated principles of a CNN model in order to develop a network for the classification of maize leaf diseases.
- ii. Development of a Hybrid model using FL rules and a CNN in order to classify and predict maize CR disease.
- iii. A novel approach to developing the training data sets for CNNs based on NCLB lesion colour and sporulation and an introduction of a CNN approach for predicting the severities of the NCLB disease.

## 1.9 Thesis structure

This thesis is organized into 6 Chapters. The main research results are presented in different Chapters (3 to 5) that were published as standalone papers but addresses the overall aim of this work.

**Chapter 1** presents the study background, underlines the research problem, state overall aim and objectives, and the general methodology employed in this study.

**Chapter 2** reviews the DL and ML for plant disease detection. This Chapter reviews the works on the DL and ML for plant disease detection and chemical based methods.

**Chapter 3** details the set-up of a customized CNN on a Java Neuroph framework to classify various types of maize leaf diseases. The emphasis will be on CNN model training, and testing.

**Chapter 4** introduces a novel approach of developing training data sets for severity prediction of maize CR by a hybrid model (Fuzzy logic and CNN). The focus will be on how Fuzzy decision rules were used with a CNN to achieve a model capable of making such predictions.

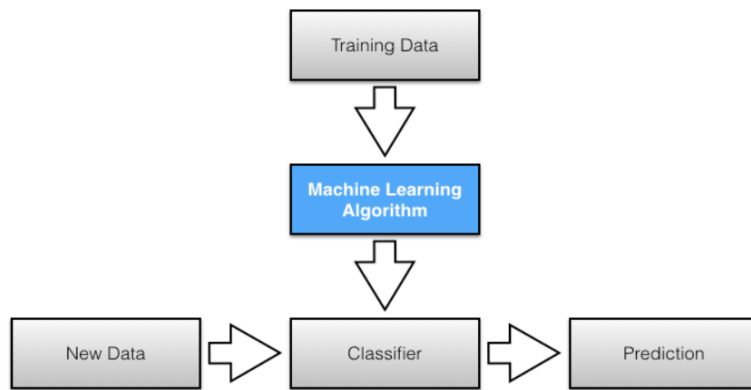
**Chapter 5** presents an approach to achieving a CNN that classifies NCLB by learning severity patterns based on NCLB lesion colour and sporulation.

**Chapter 6** evaluates the objectives by comparing them against the results obtained in Chapters 3 to 5, concludes the work of this thesis, and suggests future studies.

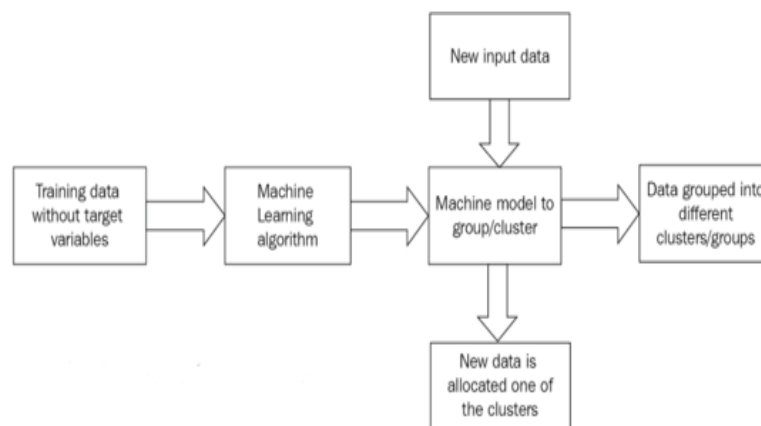
# CHAPTER 2: Literature Review

## 2.1 Introduction

As agriculture struggles to sustain the ever-growing global population, plant diseases cause significant production and economic losses. Savary *et al.* (2012) highlighted in their study that, in overall, yield losses due to plant pathogens were responsible for losses of approximately 20% to 40% of global agricultural productivity. In essence, this has a major impact on a farmer's income, food security, and the economy of a country. Prevention and control of plant diseases are therefore essential, more especially in poor countries where there is a substantial lack of experts nearby and infrastructure to get timely information. For instance, in the works of Sankaran *et al.* (2010) it is highlighted that soybean rust, which is a fungal disease had caused a substantial economic loss of which by just removing 20% of the infection would have benefitted the farmers a 11-million-dollar profit. Therefore, finding a fast, safe, automatic, less expensive, and accurate method to detect plant diseases is of great importance. There is evidence in the literature that ML and DL techniques have been applied to plant disease detection. In this chapter, we will present a review of such techniques. This study focuses on the use of CNNs, a DL approach to present various methods of maize leaf disease detection and severity prediction. Since DL is a subfield of ML, various algorithms of ML that are available in the literature for the detection of plant leaf diseases are also reviewed. The review approach used tabulates the studies, associated with DL or ML models, and related performance metrics. Before we look at the related works of ML and DL for plant leaf diseases detection, we first explain the ML and DL mathematical concepts. ML is the ability of computers to learn without exactly learning through hard coding. In a broader sense, ML is categorized into supervised learning and unsupervised learning. In supervised learning, the ML model is presented with the data to learn its features and later must be able to make predictions when a separate set of data is presented to the algorithm to make predictions. The model must avoid overfitting the data it is learning or its features otherwise it will perform poorly when making predictions on the test data sets. Underfitting is when the model cannot make sense out of the simple test data presented to it. The second category of ML is called unsupervised learning. In this case, the model learns from the data presented to it without any training data used to train it such that it can be able to make predictions. The unsupervised models group the information they find to be of the same category by forming clusters. However, since supervised ML is the most widely used technique of ML than unsupervised learning, this study mostly reviews the mathematical concepts of supervised ML algorithms. Figure 2.1 summarises the concept of supervised learning while the concept of unsupervised learning is summarized in Figure 2.2.



**Figure 2. 1:** The concept of supervised learning



**Figure 2. 2:**The concept of unsupervised learning (<https://subscription.packtpub.com/book/big-data-and-business-intelligence/9781788397872/2/ch02lv11sec33/unsupervised-learning>)

Figure 2.3 summarises the ML techniques and the algorithms associated with each technique. Basically Figure 2.3 is the ML platform for algorithm selection. For instance, everyone who would opt to use SVM in their ML projects would surely be working on supervised learning classification problems. Choosing the right ML algorithm can be overwhelming as most of the times, a “trial and error” approach is used. At times, algorithm selection depends on the size and type of the data, and the insights the data is meant to reveal.

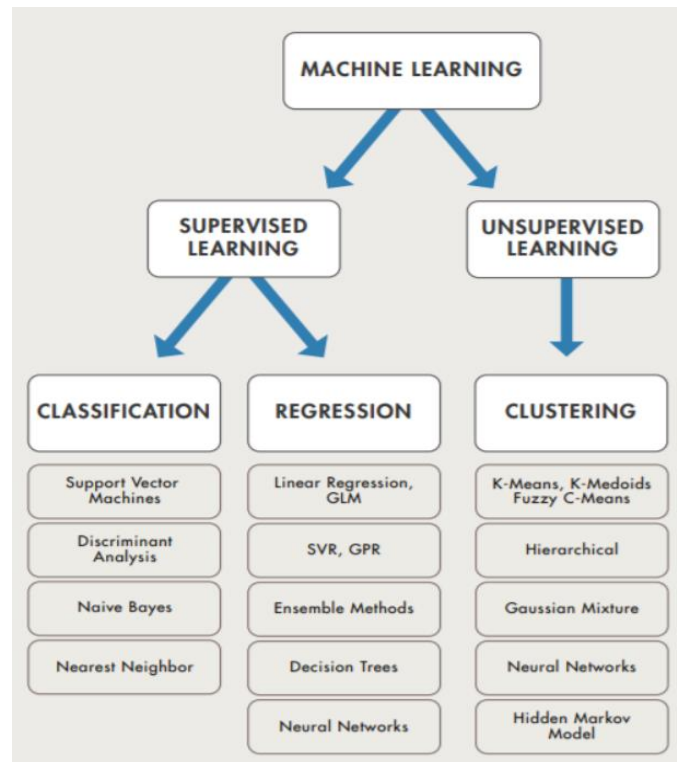


Figure 2. 3: ML algorithm selection platform (<https://www.programmingsought.com/article/47434005192/>).

There are several ML algorithms in the literature as shown in Figure 2.3, but to meet the general objective of this study, we need to review the DNN’s theoretical background and the mathematical concepts in detail. To do that, we first review the mathematical concepts behind ML’s linear regression since that form the basis of DNNs. It can be seen in Figure 2.3 that linear regression is a supervised ML algorithm. The regression algorithm makes a hypothesis by drawing a line of best fit through the data it is supposed to learn. Equation (2.1) shows the hypothesis and the learning parameters that are supposed to be properly adjusted to get the line of best fit.

$$h(x) = \theta_0 + \theta_1 x \tag{2.1}$$

where

$\theta$  = weight parameter to be adjusted

$x$  = a single input variable

In cases of more than one input variable, the hypothesis is presented as shown in equations (2.2), (2.3).

$$h(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \dots + \theta_i x_i \tag{2.2}$$

$$\therefore h(x) = \sum_{i=1}^n \theta_i x_i \tag{2.3}$$

The aim of the ML regression in this case is to reduce an error which is known as a loss function. This is done by making sure that the algorithm maps the input variables  $x_i$  to the output target variable  $y$ , such that the difference between the hypothesis  $h(x)$  and the predicted output target variable  $\hat{y}$  is minimal. A loss function of a single input variable is presented in equation (2.4).

$$J(\theta) = (h(x^i) - \hat{y}) \tag{2.4}$$

However, a loss function of the multi - input variable is presented in equation (2.5).

$$J(\theta) = \frac{1}{2} \sum_{i=1}^n (h(x^i) - \hat{y})^2 \tag{2.5}$$

To get the minimal error/loss function,  $\theta$  must be chosen such  $h(x) \approx y$ . To update  $\theta$ , an algorithm called gradient descent is usually used. The gradient descent is categorized into batch gradient descent and stochastic gradient descent. The difference between these two categories of gradient descent is that, in batch gradient descent, the algorithm goes through all training samples and calculate cumulative error before updating  $\theta$ . The stochastic gradient descent finds the error of each training sample and updates it before going to the next one. Equation (2.6) represents the batch gradient descent while equation (2.7) represents the stochastic gradient. In both equations,  $\alpha$  is the learning rate at which both algorithms update the  $\theta$ .

$$\theta = \theta - \alpha \frac{\partial}{\partial \theta} J(\theta) \tag{2.6}$$

whereas,

$$\theta_j = \theta_j - \alpha (h(x^i) - \hat{y}) \tag{2.7}$$

DL is a subfield of ML and its existence was inspired by the brain neurons connected by means of synapses. The idea of DL being similar to the brain neurons is still a controversial issue among many DL researchers. However, whatever the case might be, DL learning has many promising applications that have been tested by many researchers. Figure 2.4 shows the DL architecture of full connected NNs.

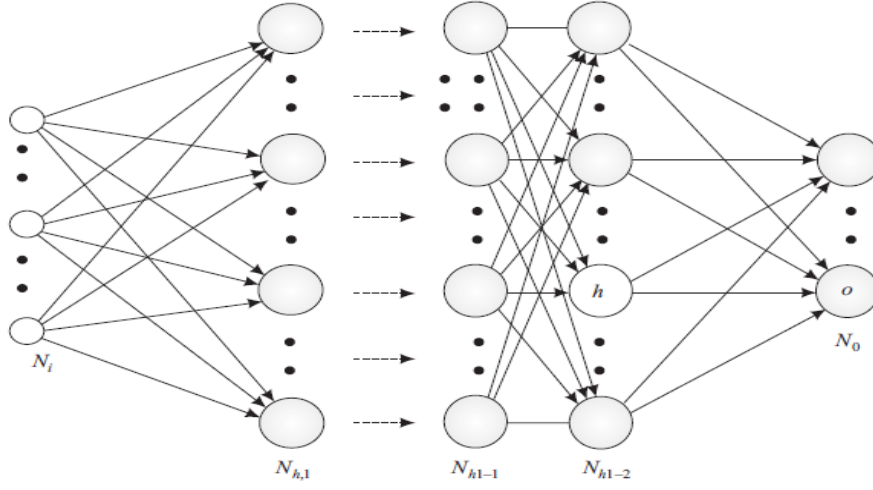


Figure 2. 4: Fully connected NN with two hidden layers and one output neuron.

The neurons are also known as the nodes. Each node of the NN learns the information using the concept of the linear regression explained in equations (2.1) to (2.7). The training samples are fed through the input layer and forward propagated towards the output. The predicted output will be compared against the target output and the error will be back propagated if it is huge, updating  $\theta$  of each node in the NN until  $h(x) \approx y$  close to  $\hat{y}$ . Back propagation is the concept of the batch and stochastic gradient descents that has just been explained. The learning by the nodes in a DNN such as the one shown in Figure 2.4 happens through the activation functions that each node possesses. Among many, the widely used non-linear activation functions in most DNNs are sigmoid and ReLU activation functions. However, Table 2.1 summarizes all the common non-linear activation functions available in the literature.

Table 2. 1: Definition of common non-linear activation functions.

<b>ReLU</b>	$ReLU(x) = \max(0, x)$	(2.8)
<b>Softplus</b>	$Softplus(x) = \log(1 + e^x)$	(2.9)
<b>Elu</b>	$Elu(x, a) = \begin{cases} x & \text{if } x \geq 0 \\ a(e^x - 1) & \text{otherwise} \end{cases}$ where $a \geq 0$ and is a tunable parameter	(2.10)
<b>Selu</b>	$Selu(x) = k \times elu(x, a)$ where $k$ $= 1.0507009873554804934193349852946$ and $a = 1.6732632423543772848170429916717$	(2.11)



---

<b>Sigmoid</b>	$Sigmoid(x) = \frac{1}{1 + e^x}$	(2.12)
----------------	----------------------------------	--------

---

<b>Tanh</b>	$Tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$	(2.13)
-------------	---	--------

---

The generalised delta rule for the learning of a DNN such as the one shown in Figure 2.4 is described as:

The activation is a differentiable function of the total input, given by

$$y_k^p = F(S_k^p) \tag{2.14}$$

in which

$$s_k^p = \sum_j w_{jk} y_j^p + \theta_k \tag{2.15}$$

In order to get the correct generalization of the delta rule, we need to set

$$\Delta_p w_{jk} = -\gamma \frac{\partial E^p}{\partial w_{jk}} \tag{2.16}$$

In this case,  $E^p$  is the error defined as the total quadratic error for pattern  $p$  at the output units:

$$E^p = \frac{1}{2} \sum_{o=1}^{N_o} (d_o^p - y_o^p)^2 \tag{2.17}$$

Where  $d_o^p$  is the desired output for unit 0 when pattern  $p$  is seen to be clamped in this sense.

$E = \sum_p E^p$  is set further to be a summed squared error. Thus, we can write

$$\frac{\partial E^p}{\partial w_{jk}} = \frac{\partial E^p}{\partial S_k^p} \frac{\partial S_k^p}{\partial E^p} \tag{2.18}$$

By means of equation (2.18) we can see that the second factor is

$$\frac{\partial S_k^p}{\partial E^p} = y_j^p \tag{2.19}$$

If we define

$$\delta_k^p = \frac{\partial E^p}{\partial S_k^p} \quad (2.20)$$

we will get an update rule which is equivalent to the delta rule, that results in a gradient descent on the error surface if weight changes are made according to:

$$\Delta_p w_{jk} = \gamma \delta_k^p y_j^p \quad (2.21)$$

We then figure out  $\delta_k^p$ . To compute the  $\delta_k^p$ , the chain rule is applied in order to write this partial derivative as the product of the two factors, one factor reflecting the change in error as a function of the output of the unit and one reflecting the change in the output as a function of changes in the input. Thus, we have

$$\delta_k^p = \frac{\partial E^p}{\partial S_k^p} = \frac{\partial E^p}{\partial y_k^p} \frac{\partial y_k^p}{\partial S_k^p} \quad (2.22)$$

If we compute the second factor by equation (2.14) we see that

$$\frac{\partial y_k^p}{\partial S_k^p} = F(S_k^p) \quad (2.23)$$

To compute the first factor of the equation (2.21), we consider two cases. First, assume that unit  $k$  is an output unit  $k = o$  of the network. In this case, it follows from the definition of  $E^p$  that

$$\frac{\partial E^p}{\partial y_o^p} = -(d_o^p - y_o^p) \quad (2.24)$$

Substituting this and equation (2.23) in equation (2.21), we get

$$\delta_o^p = (d_o^p - y_o^p) F'_o(S_o^p) \quad (2.25)$$

For any output unit  $o$ . Also, if  $k$  is not an output unit, but a hidden unit  $k = h$ , we do not know in depth the contribution of the unit to the output error of the network. The measure of the error can be written as a function of the net inputs from hidden to an output layer

$E^p = E^p(s_1^p, s_2^p, \dots, s_j^p, \dots)$  and we use the chain rule to write

$$\frac{\partial E^p}{\partial y_h^p} = \sum_{o=1}^{N_o} \frac{\partial E^p}{\partial S_o^p} \frac{\partial S_o^p}{\partial S_h^p} = \sum_{o=1}^{N_o} \frac{\partial E^p}{\partial S_o^p} \frac{\partial}{\partial y_h^p} \sum_{j=1}^{N_o} w_{ko} y_j^p = \sum_{j=1}^{N_o} \frac{\partial E^p}{\partial S_o^p} w_{ho} = \sum_{j=1}^{N_o} \delta_o^p w_{ho} \quad (2.26)$$

Substituting this in equation (2.21) yields

$$\delta_h^p = F(S_h^p) \sum_{j=1}^{N_o} \delta_o^p w_{ho} \quad (2.27)$$

Equations (2.25) and (2.27) make up a recursive procedure for computing all the  $\delta$ s for all units in the network, which are then sent to compute the weight changes according to equation (2.21). This procedure explains the generalised delta rule for a feed-forward network of non-linear nodes. The backward propagation of the DNN in Figure 2.4 is explained by the equation (2.6), whereby the error is back propagated to the network with the intention of updating the weight parameters of  $\theta$ . The activation functions to be used by these non-linear nodes may be chosen from the list of non-linear activation functions listed in Table 2.1. A different architecture of NN called CNN is used when the images are dealt with. Figure 2.5 shows a typical CNN.

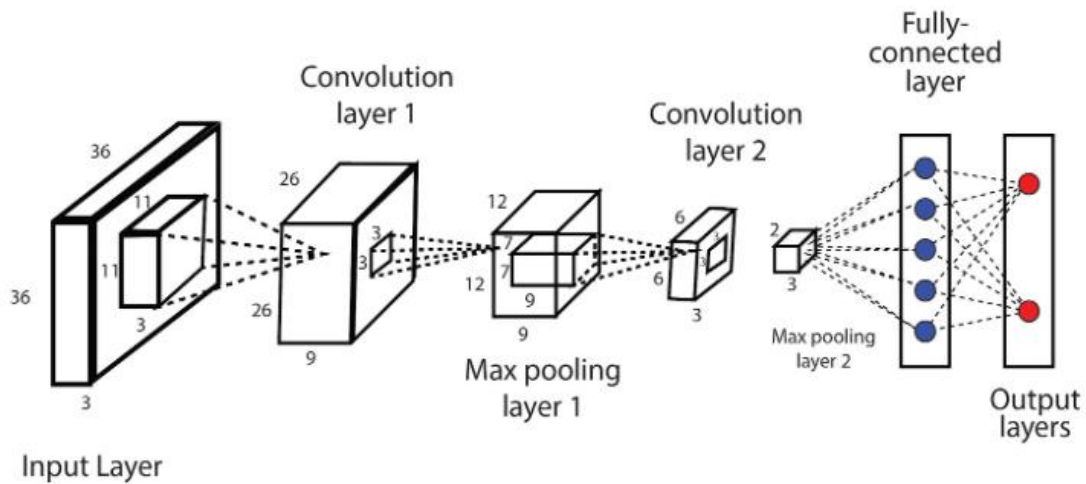


Figure 2. 5: A typical CNN architecture.

The CNN, as seen in Figure 2.5, has a layer called full-connected layer of which its operation is the one computed by equations (2.6), and (2.14) to (2.27). The Input layer is meant for the reception of the input image which is later received by convolution layer 1. The purpose of the convolution layers is to extract features from the image by performing the dot matrix between the image and the filters. Figure 2.6 shows the dot matrix operation between a 5x5 input image and a 3x3 filter/kernel. The resulting output is called the feature map.

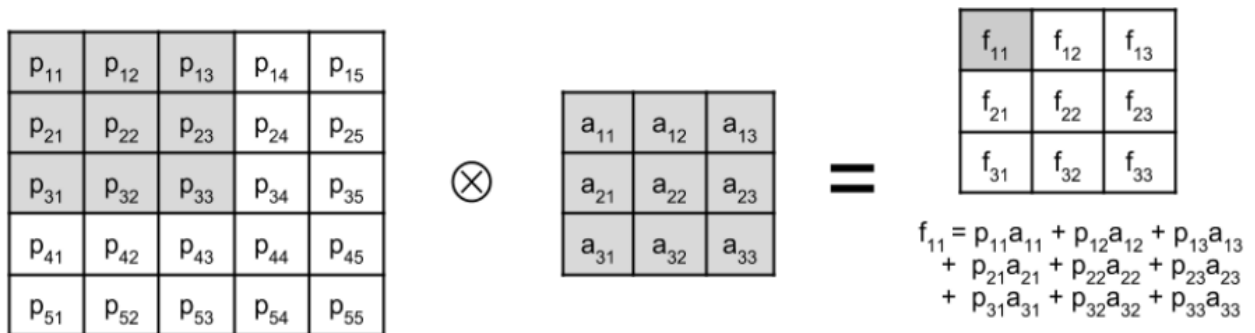


Figure 2. 6: The convolution operation showing how one element of the feature is computed.

In the resultant feature map after convolution, the value of a feature map item is greyed out. The resulting feature map can be seen to be smaller than the original entry image. This is due to the fact that convolution is only performed on valid elements. The filter/kernel cannot cross the boundaries of the image. Next, the feature map is received by the pooling layer. The concept of pooling is demonstrated in Figure 2.7 by using a pool size of 4 (pooling is done in every 4 cells of the feature map). The first pooling method is Average pooling and the second one is Maximum pooling, otherwise known as Max Pooling. The most widely used methods of pooling is the Max Pooling. It operates by using the pooling size on the feature points to result in one map point. The pooling size used in Figure 2.7 is 2x2 feature map.

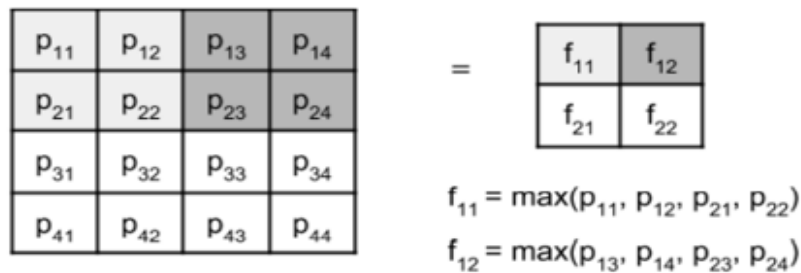


Figure 2. 7: The concept of Maximum Pooling.

The convolutions and pooling in CNNs may be repeated depending to an extent to which the features must be extracted. The pooling of a 4x4 feature map results in a 2x2 feature map that is flattened and fed to the fully connected layer that makes the classifications as shown in Figure 2.5. The mathematical concepts used in the fully connected layer are computed by equations (2.6), and (2.14) to (2.27).

## 2.2 Underfitting and overfitting in ML and DL models

**Underfitting model:** This happens when a fitted line by a ML or DL model does not approximate the data well and make it have better approximation of the data.

**Overfitting model:** This happens when we say a model has a low error rate for the training data, but it does not generalize well to the overall data in which we are interested.

**Appropriate model:** The appropriate model does not overfit nor underfit, it just generalizes well to training data. Figure 2.8 shows the behaviour of each of the above-mentioned models.

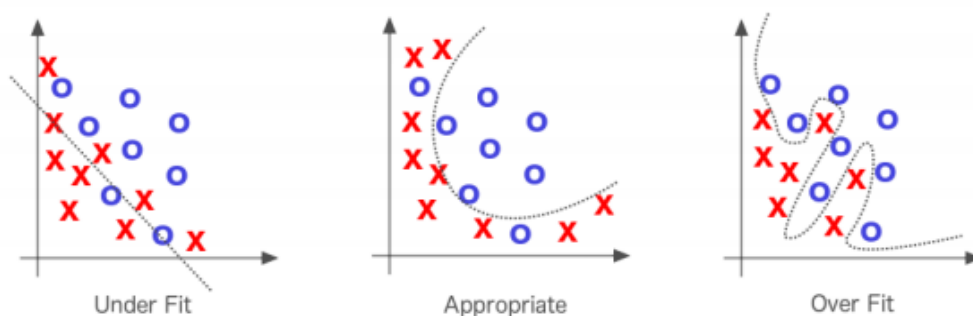


Figure 2. 8: Overfitting and Underfitting in ML and DL (Patterson and Gibson, 2017)

## 2.3 Optimization

The problem of overfitting and underfitting in any model can be corrected by tuning the parameters of the model. There are different types of parameters that can be tuned in ML and DL models. Hyper-parameters are the settings that must be provided to the ML and DL models. In the case of the CNNs that were dealt with in this study, these included the following;

### **Hyper-parameters related to network structure:**

- i. The order of the convolutional, pooling, and dropout layers
- ii. The type of activation function
- iii. The number of hidden neurons
- iv. The structure of pooling and convolutional layers

### **Hyper-parameters related to training algorithms:**

- i. Learning rate.
- ii. Momentum.
- iii. Number of epochs and batch size.

Other techniques of ML and DL optimization are regularization methods used for this study were as follows;

**Batch normalization:** Normally, we would apply a generic form of Normalization scaling to our image data values between 0-255 with values between 0 to 1 (this is done by dividing it by 255). The purpose is to reduce the influence of larger data points.

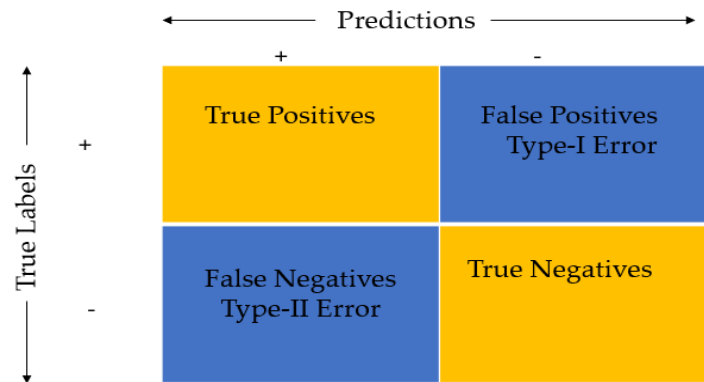
Before the training of any neural network, the weights are firstly randomized. If one of the weights becomes extremely large, its corresponding neuron's output will be very large cascading throughout the neural network causing instability causing instability. Batch Normalization normalizes the output from the activation functions of a selected layer (Ioffe and Szegedy, 2015). It then normalizes the output by multiplying it by a parameter and then adding another parameter to this result. The result is that all the activations leaving a batch normalization layer will have approximately zero mean. The weights now do not become imbalanced with extreme values since normalization is now included in the gradient process.

**Drop out:** The goal of drop out is to prevent overfitting. Dropout refers to dropping neurons (both hidden and visible) in a neural network with the aim of reducing overfitting. In training certain parts of the neural network are ignored during some forward and backward propagations. Dropout is an approach to regularizing neural networks which helps reducing interdependent learning amongst the neurons. Thus, the neural network learns more robust or meaningful features. In Dropout, we set a parameter 'P' that sets the probability of which neurons are kept or (1-P) for those that are dropped. Dropout almost doubles the time to converge in training (Heaton, 2013).

**Dataset Augmentation:** Data Augmentation is one of the easiest ways used to improve the models dealt with in this study. It was simply taking input image data sets, and made, slight variations to them in order to improve the amount of training data. This allowed the building of more robust models that did not over fit.

## 2.4 Metrics for evaluating machine learning classification algorithms

In ML and DL problems, it is always important to evaluate the performances of the chosen supervised classification models using performance key indicators (PKIs). These PKIs are explained in terms of a confusion matrix shown in Figure 2.9.



**Figure 2. 9:** A confusion matrix used to explain ML and DL KPIs

It is seen in Figure 2.9 that the confusion matrix consists of 4 outcome classes that may be predicted by the chosen classification ML or DL algorithms. True Positives are outcomes whereby the predictions made by the model correctly predict the positive classes. An example is when the model predicts that a patient has cancer and really a patient has cancer. Similarly, True Negatives are outcomes whereby the predictions made by the model correctly predict that a patient has no cancer and really a patient has no cancer. False Positives are outcomes whereby the model incorrectly predicts the positive classes, and False Negatives are outcomes where the model incorrectly predicts the negative classes. The resulting predictions of a confusion matrix may be used to derive the performance metrics presented in equations (2.28) to (2.31).

**Accuracy:** Accuracy in classification problems is the number of correct predictions made by the model divided by the total number of predictions. Accuracy is useful when the target classes are well balanced.

$$Accuracy = \frac{True\ Positives}{(True\ Positives + False\ Negatives + False\ Positives + True\ Negatives)} * 100 \tag{2.28}$$

**Recall:** This is the ability of the model to find all the relevant cases within a data set. The precise definition of recall is the number of true positives divided by the number of true positives plus false negatives.

$$Recall = \frac{True\ Positives}{(True\ Positives + False\ Negatives)} \tag{2.29}$$

**Precision:** This is the ability of the classification model to identify only the relevant data points. Precision is defined as the number of true positives divided by the number of true positives plus false positives.

$$Precision = \frac{True\ Positives}{(True\ Positives + False\ Positives)} \quad (2.30)$$

**F1-Score:** In cases where we want to find an optimal blend of precision and recall, we can combine the two metrics using what is called the F1 Score. All in all, the F1 Score is the harmonic mean of precision and recall taking both metrics into account.

$$F1 = 2 * \frac{Precision * Recall}{(Precision + Recall)} \quad (2.31)$$

## 2.5 Metrics for evaluating ML and DL regression algorithms

In this section, metrics for evaluating regression algorithms are examined. Several methods do exist of which some of them are outlined below:

**Use of Mean Absolute Error (MAE):** One of the ways of evaluating the performance of regression models is by computing the mean absolute error (MAE). We define the MAE as

$$MAE = \frac{\sum |y - \hat{y}|}{N} \quad (2.32)$$

Where  $y$  is the actual, and  $\hat{y}$  is the predicted value then  $|y - \hat{y}|$  is the absolute value of the difference between the predicted value and the actual value.  $N$  is the number of sample points.

**Use of Root Mean Square Error (RMSE):** Another evaluation metric for regression is the RMSE. It is calculated in the manner very similar to MAE, just that instead of taking the absolute value to get rid of the sign on the individual errors, the error is squared (This is because the square of a negative number is positive).

$$RMSE = \sqrt{\frac{\sum (y - \hat{y})^2}{N}} \quad (2.33)$$

Next, we review the studies that utilized chemical-based methods of plant disease detection (Stem and leaves).

## 2.6 Chemical laboratory-based methods for plant disease detection

Agriculture is one of South Africa's economic pillars. In particular, maize is the staple food to over a million South Africans. It is for this reason that chemical and manual laboratory-based methods have been developed

to enhance the production of maize. Several methods have been developed and the methods included, but were not limited to polymerase chain reaction (PCR), immunofluorescence (IF), fluorescence *in-situ* hybridization (FISH), enzyme-linked immunosorbent assay (ELISA), flow cytometry (FCM) gas chromatography-mass spectrometry (GC-MS) (Fang and Ramasamy, 2015). Table 2.2 summarizes the common chemical-based methods of plant disease detection in the literature.

Table 2. 2: A summary of chemical-based methods for plant leaf disease detection.

Chemical based method used	Study
Evaluation of antioxidants in stems and leaves of spinach.	(Singh <i>et al.</i> , 2016)
Detection of plant diseases using nanopore sequencing platform.	(Chalupowicz <i>et al.</i> , 2019)
Chemical control of plant diseases.	(Waard <i>et al.</i> , 1993)
Polymerase chain reaction (PCR)	(Minsavage <i>et al.</i> , 1994)
Polymerase chain reaction (PCR)	(Haelterman <i>et al.</i> , 2014)
Polymerase chain reaction (PCR)	(Saponari <i>et al.</i> , 2008)

## 2.7 Theoretical alternatives

Plant leaf disease-detecting systems are used to detect the types of plant leaf diseases to avoid agricultural losses. Distinctively, diverse modelling approaches such as Convolutional Neural Networks (CNN) and Support Vector Machines (SVM) have been used for plant leaf disease detection in several articles. Interestingly, there have been known shortcomings with some of the modelling approaches such as longer training times. Underpinned by such shortcomings researchers have considered other alternatives and probably in this study context, better methods of detecting plant leaf diseases. Strands of the literature show that various researchers and academics have used different algorithms, with some measure of success, for disease detection in plant leaves. A few articles that were reviewed in this study, dealt with different algorithms of DL that were used to model the detection of plant leaf diseases. Also, different values of accuracies for various DL architectures have been recorded in the literature. Could it be the use of different DL architectures that cause these discrepancies? With such a question in mind, it is the aim of this section to do a systematic review of the DL architectures that were used for the detection of plant leaf diseases in the literature. Amongst a few plant diseases that exist in different plants, this study restricts this review to the detection of plant leaf diseases using DL. This section shows how a systematic review of journal articles, conference proceedings and book chapters that was conducted to arrive at our performed to meet the objectives. In this section, we overlay the literature to draw up the conclusions on the DL for plant disease detection.

### 2.7.1 Study retrieval and selection

One of the steps to conduct a systematic review is study retrieval and selection (Denson and Seltzer, 2011). The reading process focused on finding the software framework for the deep learning model used to detect and classify plant leaf diseases. Given the software frameworks, the total percentage of the classification



accuracy for each study was considered to compare the performances of different software frameworks. The abstract and introduction reading approach for each paper was followed.

### 2.7.2 Exclusion criteria explained

To strengthen the validity of the review, we applied certain exclusion criteria to all the studies collected in order to meet with the main objective of this study. These criteria are presented next.

*Exclusion criteria 1:* All the studies had to be published journal articles, conference proceedings and book chapters, otherwise were excluded.

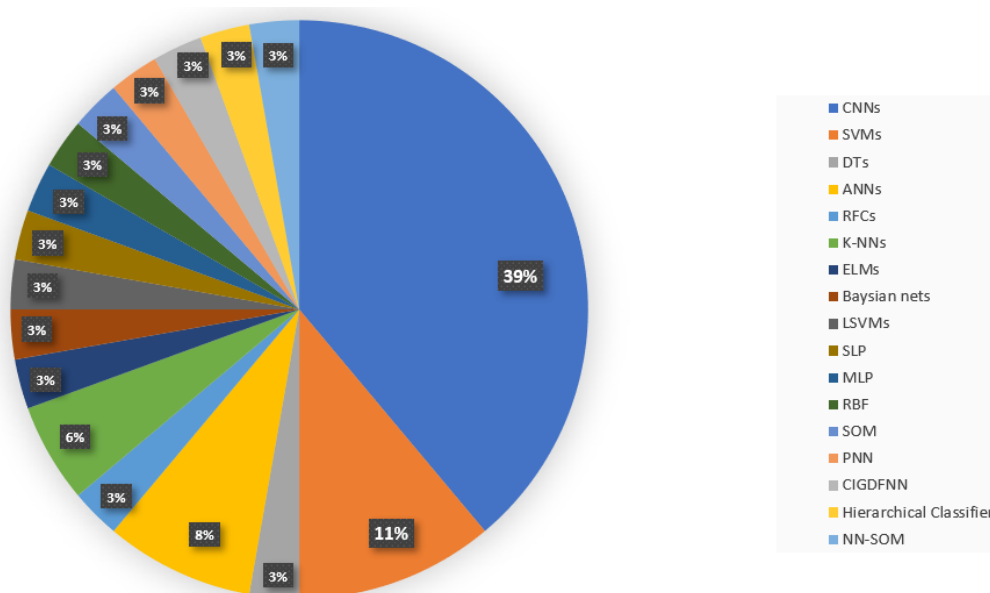
*Exclusion criteria 2:* Abstracts were not regarded as part of the review if they did not provide enough information.

*Exclusion criteria 3:* All the studies whereby the authors did not have access to, were not considered for review.

*Exclusion criteria 4:* All the studies were to be in English. The studies in any foreign language were excluded.

*Exclusion criteria 5:* Duplicate copies of studies were not included in this review. This means that all studies had to be unique.

*Exclusion criteria 6:* The studies that did not use DL and ML for the detection of plant leaf diseases were excluded. This means that all the studies that used DL and ML for the detection of stem or root diseases were excluded. Only studies that used deep learning models for the detection of leaf diseases were included in this review. The later mentioned exclusion criteria resulted in reviewing of the studies that involved both ML and DL for plant leaf diseases as summarized in Figure 2.10.



**Figure 2. 10.** The overall percentages of DL and ML algorithms that were used for plant leaf disease detection in this review.

Some of the studies considered for the thesis study include work by authors such as Sladojevic. For example, researchers like Sladojevic and others, have modelled plant leaf disease detection systems using different machine learning algorithms with exceptionally accurate results. (2016), Mohanty *et al.* (2016) and Amara *et al.* (2017). In the literature, most studies, with the objective of estimating the severity of foliar diseases, use

manual laboratory procedures. Manual lab procedures require skilled plant pathologists. A study led by Fang and Ramasamy (2015) examined direct methods for manual laboratory detection of foliar diseases. Sannakki *et al.* (2011) were the first to estimate the severity of leaf diseases using computerized vision. In their study, they used image processing and fuzzy logic to assess the severity of leaf diseases. They used the K-means clustering method to segment leaves from diseased plants. By some fuzzy logical inference, Sannakki *et al.* (2011) were able to estimate the severity of leaf diseases detected by grading them using MATLAB. Wang *et al.* (2017) addressed an issue of estimation of the severity of plant diseases using deep learning. Their study used a process that was completely automatic because it avoided labour-intensive engineering functions and threshold-based segmentation. They used the Plant Village image dataset to estimate the severity of fine-grained diseases, like apple rot. In their study, they used a range of deep convolutional neural networks and the most effective model being the VGG16 model with 90.4% accuracy on the holding test set. Sun *et al.* (2015) examined the results of the tomato disease severity comparison using image segmentation and visual estimates using a scale of categories for genetic testing of resistance. As part of their investigation, image processing and analysis were performed using image analysis software called ASSESS V2.2. They used the tint, saturation intensity, colour space and filter middle in the colour panel. Another study compared, visual estimates with image analysis measurements, was conducted by Jarroudi *et al.* (2015) to determine the severity of Septoria Late Blight in Winter wheat. In their study, they used a program called ASSESS V2.0 to process digital images. They chose a classic sign and a tint, saturation and intensity colour space to differentiate the rest of the leaf from the blue background. To develop and test their method for analyzing digital images using the Scion Image or NIH public domain software and quantified leaf colour. In their work, they provided step-by-step instructions to use the Scion software in order to measure the percentages of green and red in the sheets. Those were the colours of particular importance for the assessment of plant health. Wijekoon *et al.* (2008) used the method proposed by Murakami (2005) to quantify fungal infection of plant leaves by analyzing digital images. Barbedo (2014) studied an automatic method to detect and measure leaf disease symptoms using digital image processing. The Barbedo method (2014) was designed to be fully automatic, eliminating the possibility of human error by reducing the time required to measure the severity of the illness. In this method, RGB images have been converted to L\*a\*b format. The L\*a\*b format was chosen because channel A revealed different types of symptoms. The channel was used from the RGB to L\*a\*b conversion point until the algorithm was completed. All pixels in channel A outside of the masks were then converted to zero. Patil and Bodhe (2011) proposed segmenting diseased sugarcane lesions by means of rectangular segmentation methods. In their study, they compared the segmented area to the leaf area. The aim of their approach was to eliminate the use of pesticides, so that they were only applied according to the total calculated severity of the disease. Other methods of image segmentation using the KNN (K-Nearest Neighbours) segmentation were used by researchers like Pallottino *et al.* (2018). In their work, Pallottino *et al.* (2018) used the KNN algorithm for the implementation of the old tractor tillage process that recognized plant weeds using

a camera adapted for data acquisition. Owomugisha and Mwebaze's proposed method used a linear SVC classifier to classify leaf diseases, according to levels (Owomugisha and Mwebaze, 2017). The levels were classified as Healthy class, Level 2 severity, level 3 severity, and level 4 severity. Bock *et al.* (2009) used ASSESS V1.0 for its advantage of being able to perform a colour threshold in the image of the diseased area. In their work, they used the raters to estimate the severities of foliar citrus canker symptoms at a predetermined range of the percentage of infections. In this study, the development of the CNNs and their training and testing is detailed in Chapters 3 to 5 that address the objectives of this study. Reviewing subsequent and more recent literature there is evidence that DL and ML have a promising application in the detection of plant leaf diseases. Table 2.3 summarizes additional studies with some of the later reviewed DL and ML approaches for plant leaf disease detection available in the literature.

Table 2. 3: Tabulated review of deep learning and machine learning approaches for plant leaf disease detection

<b>DL methods for plant leaf disease detection</b>			
<b>Focus on the affected area of the plant species</b>	<b>Research aims</b>	<b>DL model and performance metrics</b>	<b>Study</b>
Soybean leaves.	To identify soybean leaf diseases and stresses.	CNN. Accuracy of 94.13% was achieved.	(Ghosal <i>et al.</i> , 2018)
Cotton leaves	To identify the diseased leaf spot of cotton.	Cross Information Gain Deep Forward Neural Network (CIGDFNN). Accuracy of 95% was achieved.	(Revathi and Hemalatha, 2013)
Plant leaves in general.	To collect and classify leaves of infected crops, according to the disease.	CNN. Accuracy of 98.59% was achieved.	(Dhaka and Shakya, 2018)
Palm tree leaves.	To discriminate and classify fungal infections in oil palm trees at an early stage using raw, first, and second derivative spectra-radiometer datasets.	ANNs. Accuracy of 83% at 540nm; Accuracy of 100% at 550nm.	(Ahmadi <i>et al.</i> , 2017)
Plant leaves of different crops.	Detecting and classifying plant leaf diseases.	CNN. Accuracy of Caffe net was 93.35%; Accuracy of Alex net was 85.53%; Accuracy of Google net was 99.34%.	(Mohanty <i>et al.</i> , 2016)
Banana leaves.	Automating the process of classifying banana leaf diseases.	CNN. The following accuracies were achieved depending on the image data split. 20%training; 80% testing (Colour) =98.61%. 40%training; 60% testing (Colour) =98.61%. 50%training; 50% testing (Colour) =99.72%. 60%training; 40% testing (Colour) =96.76%. 80%training; 20% testing (Colour) =92.88%. 20%training; 80% testing (Gray) =94.44%. 40%training; 60% testing (Gray) =97.57%. 50%training; 50% testing (Gray) =85.28%. 60%training; 40% testing (Gray) =92.82%. 80%training; 20% testing (Gray) =85.94%.	(Amara <i>et al.</i> ,2017)

Wheat, avocado, rice, and cotton leaves.	To understand the performance of different Neural networks in classification of plant leaf diseases.	Single-Layer Perceptron (SLP), Multilayer Perceptron (MLP), Radial-Basis Function (RBF), Kohonen's Self-Organising Map (SOM), Probabilistic Neural Network (PNN) and CNN.  The following classification accuracies were achieved: MLP, SOM =99%; RBF, MLP =98%; NN-SOM =97%; PNN= 95%; CNN= 99.3%; CNN=96.3%.	(Golhani <i>et al.</i> , 2018)
Areca nuts (same procedure as leaf classification).	To apply neural networks and image processing techniques for detecting and classifying the quality of Areca nuts.	ANN. A classification accuracy of 90.9% was achieved.	(Huang, 2012)
Plant leaves in general.	To detect and classify the plant leaf diseases using image processing and neural network technique.	CNN. A classification accuracy of 90.9% was achieved.	(Brahimi <i>et al.</i> , 2018)
Cassava leaves.	To identify and classify three types of cassava leaf diseases.	CNN. A classification accuracy of 93% was achieved.	(Ramcharan <i>et al.</i> , 2017)
Grapevine, Apple, Peach and Pear leaves.	To develop plant disease recognition model, based on leaf image classification.	CNN. A classification accuracy of 96.3% was achieved.	(Sladojevic <i>et al.</i> , 2016)
Various plant leaf diseases.	To classify plant leaf diseases.	ANN. A classification accuracy of 87.48% was achieved.	(Pujari <i>et al.</i> , 2016)
Maize leaves.	To classify various diseases of maize leaves.	CNN. An average accuracy of 92.85% and accuracies of 87-99.9% on separate class tests	(Sibiya and Sumbwanyambe, 2019a)
Maize leaves.	To predict the severities of maize Common Rust by classifying the tested images into stages of severity.	CNN. It achieved a validation accuracy of 95.63% and accuracy 89% when tested on separate images of CR to make severity predictions among 4 classes of CR with various stages of the disease' severities	(Sibiya and Sumbwanyambe, 2021)
Various leaf diseases	To improve the Accuracy of Plant Leaf Disease Classification	Improved CNNs. 98% accuracy for both testing and validation sets	(Nerkar and Talbar, 2020)
Various plant leaves with diseases such as black rot, bacterial plaque, and rust.	To introduce a mathematical model of plant leaf disease detection and recognition based on deep learning, which improves accuracy, generality, and training efficiency	CNN-Resnet 101. It is shown by the results that the accuracy of the method was 83.57%, which was better than the traditional method.	(Guo <i>et al.</i> , 2020)
Various plant leaves.	To investigate the impact of the dataset in plant leaf disease detection.	CNN. Classification accuracies were varied depending on the size of the training, number of classes, plant species, and whether the backgrounds were removed or not. Depending on these parameters, the accuracies ranged from the minimum of 50% to 100%	(Barbedo, 2018)
Various plant leaves.	To classify plant leaf diseases based on their lesions and spots	CNN. Classification accuracies were varied depending on the size of the training, number of classes, plant species, and whether the backgrounds were removed or not. Depending on these parameters, the accuracies ranged from the minimum of 31% to 100%	(Barbedo, 2019)

---

**ML methods for plant leaf disease detection**

---

<b>Research aims</b>	<b>Focus on the affected area of plant species</b>	<b>ML model and performance metrics</b>	<b>Study</b>
To capture the symptoms of Cotton Leaf Spot images and categorize the diseases using support vector machine.	Cotton leaves.	Support Vector Machines (SVM). SVM accuracy for wavelength transform was 97%. SVM accuracy for Texture feature extraction-GLCM was 97.2%	(Patil and Zambre, 2014)
To investigate the potential of using hyperspectral imaging for detecting different diseases on tomato leaves.	Tomato leaves.	Extreme Learning Machine (ELM) model. The following classification accuracies were achieved by the ELM. Healthy = 100% Early Blight = 100% Late Blight = 100%	(Xie <i>et al.</i> , 2015)
To identify and assess the two diseases, the canopy hyperspectral data of healthy wheat, wheat in incubation period, and wheat in diseased period.	Wheat leaves.	SVMs. The model's training accuracy was on separate test data was 82%.	(Wang <i>et al.</i> , 2015)
To use hyperspectral imaging in the determination of coffee rust.	Coffee leaves.	SVM, Decision Trees (DTs), and K-Nearest Neighbors (K-NNs). The following accuracies were achieved to correctly classify the healthy class. SVM accuracy was 94.70%; K-NNs accuracy was 93%; DTs accuracy was 90.30%.	(Castro <i>et al.</i> , 2018)
To detect plant leaf diseases and their severities.	Cassava leaves.	Linear Support Vector Machines (LSVM), KNNs and Extra trees. The following accuracies were achieved to correctly the diseased leaf areas: LSVM accuracy was 80% on colour, and 99.98% on colour with ORB feature extraction; KNNs accuracy was 44.68% on colour, and 100% on colour with ORB feature extraction; Extra trees accuracy was 48.94% on colour, and 99.88% on colour with ORB feature extraction;	(Owomugisha and Mwebaze, 2017)
To classify the leaf brown spot and the leaf blast diseases of rice plant based on the morphological changes of the plants caused by the diseases.	Rice leaves.	Bayesian networks and SVM. Bayes classifier achieved an accuracy of 79.5%; SVM achieved a classification accuracy of 68.1%.	(Phadikar <i>et al.</i> , 2012)
To classify different plant diseases.	Ground nut leaves.	Improved Random Forest classifier. It was confirmed by the performance results that the proposed improved-RFC approach performed better than the Random Forest algorithm with an improved classification accuracy of up to 97.80% for a multi-class groundnut disease dataset.	(Chaudhary <i>et al.</i> , 2016)
To detect stress severity.	Soybean leaves.	Classification trees. In this study, 10 different classification approaches, with the best hierarchical classifier with a mean per-class accuracy of approximately 96%.	(Naik <i>et al.</i> , 2017)

---

## 2.8 Fuzzy logic and Thresholding theoretical concepts

Part of this study focuses on hybridizing the CNN, thresholding and Fuzzy Logic (FL) decision rules into a hybrid system for predicting the severity of CR maize disease. A hybrid system is the one that combines at least two intelligent technologies. The primary point of the idea of hybridization is to defeat the shortcoming in one system while applying it and drawing out the quality of another strategy to discover an answer by joining them. Fuzzy logic and artificial neural networks have been used in many engineering applications and sequels of articles were published in these areas of artificial intelligence. Next, a review of FL and thresholding techniques is performed.

### 2.8.1 The Concept of Fuzzy Logic

Fuzzy logic is an approach to computing based on "degrees of truth" rather than traditional "true or false" (1 or 0) Boolean logic on which the modern computer is based (Zadeh, 2015). Fuzzy logic control is an intelligence that computers can use to achieve control objectives without modelling the system using complex math formulas. In fuzzy logic control, a series of "if" and/or "else" statements are used to achieve control goals. Figure 2.11 illustrates a fuzzy logic system.

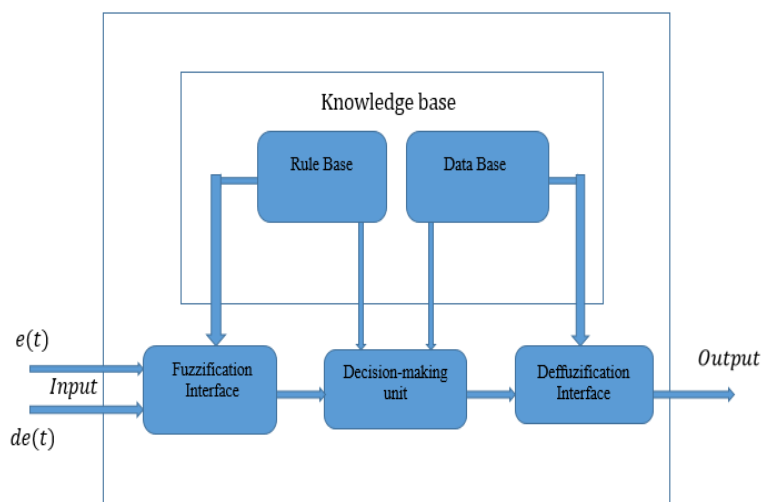


Figure 2. 11: Architecture fuzzy logic system

The literature highlights that there are many papers in which fuzzy image processing systems have been designed. For instance, Schröder *et al.* (2014) modelled imaging deflagration detection methods with a fuzzy classification. The results identified the fire, according to its normal flicker rate. Some, like Celik *et al.* (2007) developed fuzzy logic, an improved generic colour model to classify fire pixels. The performance of the model was tested on two large sets of images; one set contained fire while the other set contained no fire but had regions similar to fire colour. One was holding the fire, while the other was not holding the fire, but had zones like the colour of the fire. The model achieved a correct fire detection rate of 99.00% with a false alert rate of 9.50% (Celik *et al.*, 2007). The fuzzy logic of AI provides a high degree of reasoning flexibility. It is basically

a method of reasoning that resembles human reasoning. This approach resembles the way humans make decisions, and it includes all the intermediary possibilities between "YES" and "NO". In that sense, fuzzy logical reasoning provides acceptable reasoning and helps to deal with uncertainty in engineering. The fuzzy logical architecture contains all the rules and requirements offered by human experts to control the decision-making system. In a study by Behera *et al.* (2018), it is mentioned that fuzzy logic was invented by Lotfi Zadeh. It is Lotfi Zadeh, who observed that humans have a different range of possibilities between “YES” and “NO”, which is what made their logical thinking differ from that of the computers as they make binary decisions. It is for these reasons why Behera *et al.* (2018) were able to use multi class support vector machines (SVM) with K-means clustering for the classification of plant. In their work, they integrated a fuzzy logic to calculate the severity of orange disease, and their hybrid model reached an accuracy of 90%. Using fuzzy logic, others like Sannakki *et al.* (2012) proposed an image processing approach for automatically classifying the spread of disease on plant leaves. Rastogi, Arora, and Sharma used MATLAB to conduct segmentation and classification based on K-averages, percent infection calculation, and disease classification using the fuzzy logic toolkit (Rastogi et al., 2015).

**2.8.1.1 Fuzzification of Crisp Inputs**

Fuzzification is the way toward changing a genuine scalar value into a fuzzy value. This is accomplished with the distinctive sorts of fuzzifiers (membership functions). To explain the concept of FL, this study focuses on temperature cooling system that is controlled by fuzzy inference rules as an example. Fuzzy linguistic variables are utilized to present qualities traversing a specific range. Fuzzification can be utilized with fuzzy linguistic variables as shown in the following example:

**Example 1**

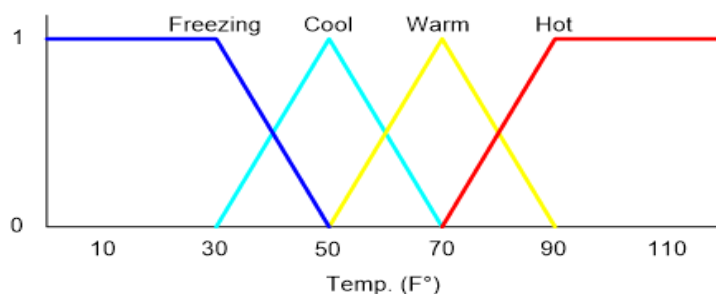
Temp: {Freezing, Cool, Warm, Hot}

Question: What is the temperature?

Answer: It is warm.

Question: How warm is it?

Figure 2.12 shows the membership function of the input temperature for a FL system.



**Figure 2. 12:** Membership functions of the input temperature

Imagine if a question was to be asked by referring to the fuzzified input of Figure 2.12. What is the degree of coolness at 36 F°?

Answer: It is 70% Freezing and 30% Cool.

Therefore, the degree of membership  $\mu(x)$  plays an important value in finding the fuzzified value at a particular value of the temperature. The illustrations in Figure 2.13 attempts to answer the latter asked question.

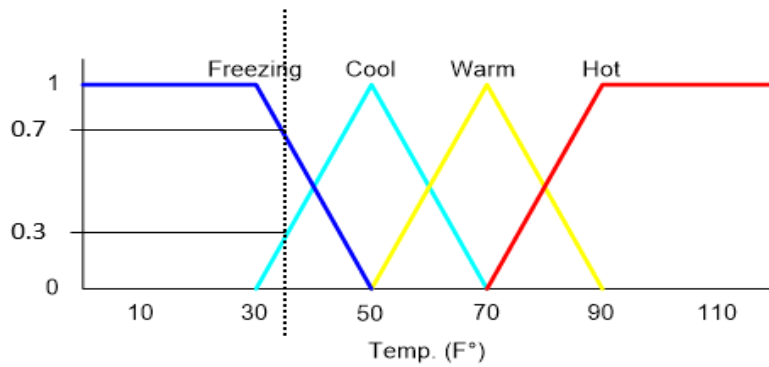


Figure 2. 13: Membership functions of the input temperature at a known temperature of 36 F°.

From the interpretation of the membership functions of Figure 2.13, the temperature value can be determined by

$$Temperature = \frac{\mu_{Cool}(x)}{Cool} + \frac{\mu_{Freezing}(x)}{Freezing} \tag{2.34}$$

To answer the latterly question

$$Temperature = \frac{\mu_{Cool}(x)}{Cool} + \frac{\mu_{Freezing}(x)}{Freezing} = \frac{0.3}{36} + \frac{0.7}{36} = 0.0277 \text{ (}^\circ\text{F)}$$

Membership functions have the following general properties (Davim, 2012):

Core:  $x \mid \mu_{\tilde{A}}(x) = 1$

Support:  $x \mid \mu_{\tilde{A}}(x) \geq 0$

Boundary:  $x \mid 0 < \mu_{\tilde{A}}(x) < 1$

Figure 2.14 illustrates the general properties of membership functions in a graphical presentation.

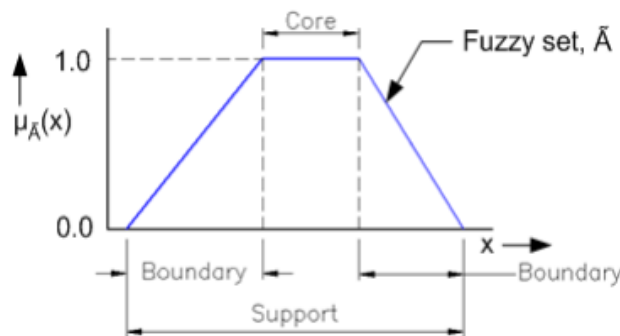


Figure 2. 14: A graphical presentation of general properties for fuzzy set membership functions.



### 2.8.1.2 Rule-Based Fuzzy Inferencing

This is the stage of the fuzzy logic control system that constitutes rule-based decisions. For example, a rule to switch the heater (output) ON when very low temperatures (input) are reached would be designed as follows:  
*If the temperature cools and freezing, then the heater ON*

A series of “if” (antecedent) and “then” (consequent) statements could be used in order to write fuzzy rules for the fuzzy decision-making unit. The decision-making unit is also known as a fuzzy inference unit.

### 2.8.1.3 Defuzzification

The output of the inference procedure is a fuzzy set, determining a probability conveyance of the (control) action. In the online control, a non-fuzzy (crisp) control action is generally required. Thus, one must defuzzify the fuzzy control action (output) induced from the fuzzy thinking calculation, specifically:

$$z_o = defuzzifier(C) \quad (2.35)$$

where  $z_o$  is the crisp action and *deffuzifier* is the defuzzification operator. A defuzzification is a process to select a representative element of the fuzzy output  $C$  inferred from the fuzzy control algorithm.

Definition:

Let  $A$  be the fuzzy set given by  $A = \{(x_k, \mu_A(x_k)), k = 1, 2, \dots, n\}$

where  $x_k \in R$ . Then the *centroid defuzzifier*  $\delta$  is given by

$$\delta = \frac{\sum_{k=1}^n x_k \mu_A(x_k)}{\sum_{k=1}^n \mu_A(x_k)} \quad (2.36)$$

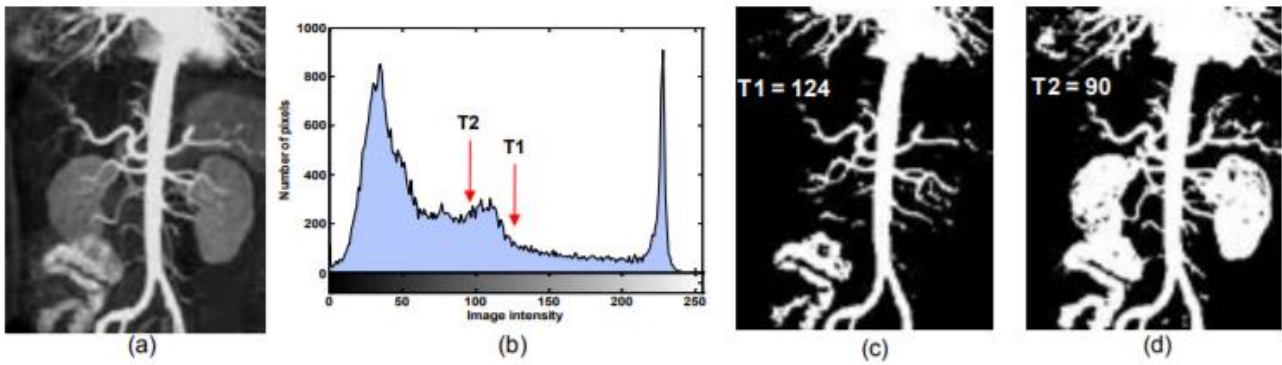
It might be seen that the centroid defuzzifier can be utilized just when  $X \subseteq R$ , the set of real numbers. This defuzzifier is the weighted average of the given qualities, weighted by the particular membership values.

### 2.8.2 The Fundamentals of Thresholding

**Histogram thresholding:** One of the famous techniques for monochrome image segmentation is histogram thresholding. For instance, suppose the intensity histogram shown in Figure 2.15 corresponds to an image  $f(x, y)$ , consisted of light object on the dark background. The objects from the background could be extracted by selecting a threshold  $T$  that separates histogram. Any point  $(x, y)$  for which  $f(x, y) \geq T$  (e.g.,  $T=124$ ) is called an object pointer; other than that, the point is called the background point. This implies, the thresholded image is defined as

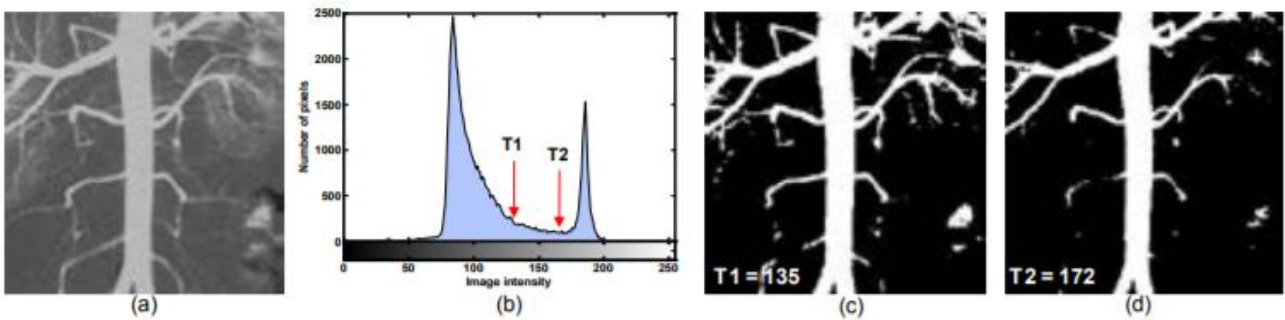
$$g(x, y) = \begin{cases} 1 & \text{if } f(x, y) \geq T \\ 0 & \text{if } f(x, y) \leq T \end{cases} \quad (2.37)$$

Pixels labelled 0 correspond to the background while those labelled 1 correspond to objects.



**Figure 2. 15:** (a) Aorta and blood vessels shown by an MR angiography image; (b) Intensity histogram of the image in (a);(c) The thresholded binary image with a threshold value T1=124, pixels labelled 1 (in white) corresponding to objects;(d) A binary image thresholded with a threshold value T2 =90 (Zhou *et al.*, 2010b).

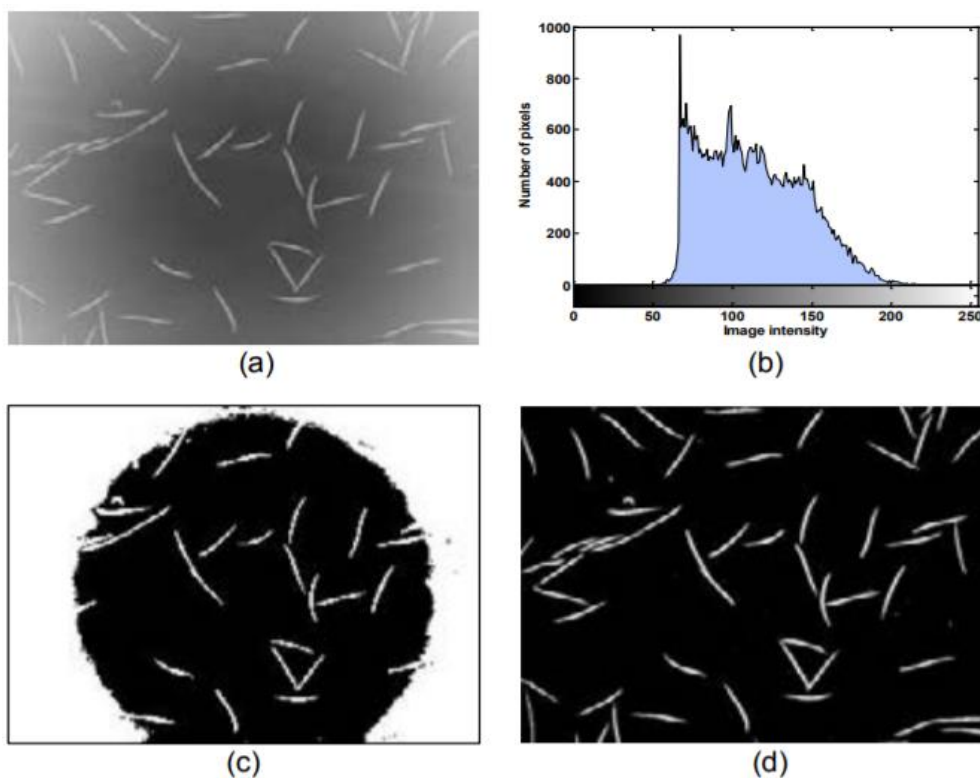
**Global thresholding:** This type thresholds either global or local (Zhou *et al.*, 2010b). There are quite a few approaches to implementing optimal thresholding. The general methodology follows the consideration of pixels, foreground, and background, as belonging to two classes or clusters. The aim is to pick a threshold in a manner that each pixel on each side of the threshold is closer in value to the mean of the pixels on that side of the threshold than the mean of the pixels on the other side of the pixels (Zhou *et al.*, 2010b). Since the algorithms automatically proceed without the intervention of the user, they are said to be unsupervised. A method proposed by Otsu (1979) is a good example of such a technique, in which the optimal threshold is chosen as the one that maximises  $\frac{\delta_B^2}{\delta_T^2}$ , with  $\delta_T^2$  the total variance. Since the Otsu is unsupervised and offers optimal thresholding, it was chosen in a study presented in chapter 4 of this thesis. Figure 2.16 compares the Otsu and conventional methods of thresholding.



**Figure 2. 16:** (a) A blood vessel image; (b) histogram's intensity of image in (a); (c) The resulting thresholded image when optimal Otsu was used with a threshold value of T1=135; (d) The resulting thresholded image when the conventional method was used with a threshold value T2=172 (Zhou *et al.*, 2010b).

**Adaptive local thresholding:** A satisfactory segmentation result is obtained when the background is uniform. In the literature there are many background corrections, but they may not result in an image that is suitable for thresholding (Zhou *et al.*, 2010b). It is possible that the transition between the background and object diffuses, making an optimal threshold level difficult to find. Also, a trivial change in the threshold level may have great impact in later analyses. To circumvent the problem of varying background, adaptive local thresholding can be used, or as refinement to coarse global thresholding (Gonzalez and Woods, 2002; Zhou *et al.*, 2010a).

When using adaptive local thresholding, each pixel is deemed to have a  $n \times n$  neighbourhood around it from which a threshold value is calculated and the pixel set to black or white, according to whether it is above or below this threshold,  $T_L$ . Also, the size of the neighbourhood,  $n$ , will have to be large enough to cover enough background and foreground pixels, such that the effect of noise is minimal. In fact, this size of the neighbourhood,  $n$ , does not have to be too large that uneven illumination becomes noticeable within the neighbourhood (Zhou *et al.*, 2010b). Figure 2.17 illustrates an example of an adaptive local thresholding versus the Otsu thresholding on an image that consists of a variable background.

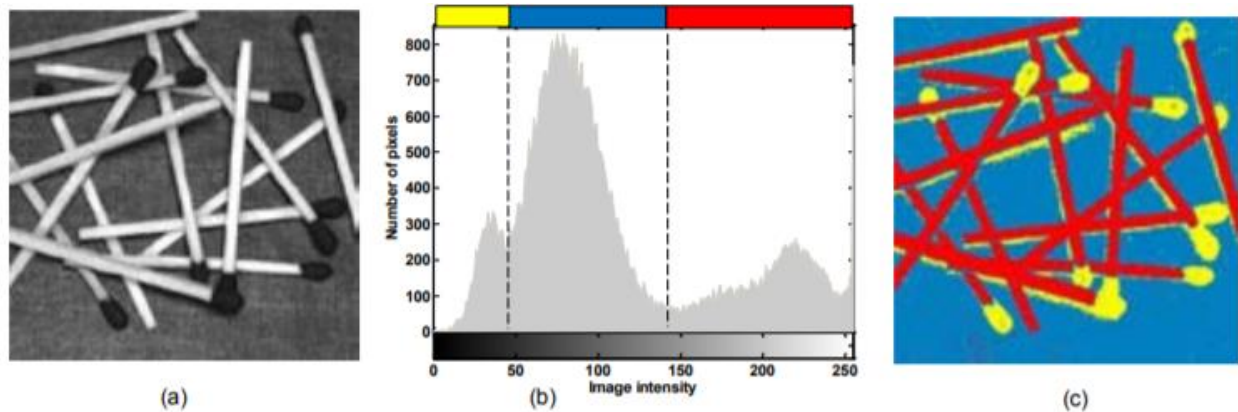


**Figure 2. 17:** (a) Original microscopic image of *C. aligns*, note it has an uneven background illumination; (b) Image intensity histogram (a); (c) Segmentation of image (a) using the Otsu threshold, where the threshold value is 117 and fails to locate all objects from the surrounding background; (d) Segmentation of image (a) using adaptive local thresholding (Zhou *et al.*, 2010b).

The main disadvantage of histogram-based methods is that they do not take into account the shape information, and the result can be difficult to conceive, particularly in cases of low signal-to-noise ratios (Zhou *et al.*, 2010b). Additionally, peak noise or contamination from other points can be categorized into the spot, resulting in errors in the estimated intensity values.

**Multiple thresholding:** The purpose of a single threshold is to segment the image into only two regions which are the background and foreground. However, more commonly, the objective is to segment the image into multiple regions by use of multiple thresholds (Zhou *et al.*, 2010b). The technique of multiple thresholding considers that an image consists of the various regions that correspond to the grey level ranges. As shown in Figure 2.18, it is seen that the histogram of an image can be separated by means of peaks (modes) that correspond to the various regions. A thresholding value that is used to separate the objects is the one

corresponding to valley between two adjacent peaks. The selection of an appropriate threshold value determines the success of thresholding.



**Figure 2. 18:** (a) A grey level image of some randomly placed match sticks; (b) Intensity histogram of image in (a);(c) Multiple thresholded image with corresponding threshold values of  $T_1= 45$  and  $T_2= 134$  (Zhou *et al.*, 2010b).

There is evidence in the literature that a plethora of studies employed the Otsu threshold method for many different applications. For instance, Sibiya and Sumbwanyambe (2019b) used the Otsu method in their study for severity detection of maize leaf diseases along with FL techniques. In their extended study, Sibiya and Sumbwanyambe (2021) also used the Otsu method and FL as a means of image segmentation for the preparation of image data that were used for training a CNN model to predict the maize CR severity. Various approaches have been proposed in using Otsu method for thresholding. For instance, it was proven in a study by Fang *et al.* (2009) that Otsu algorithm can calculate the high threshold value, which is significant to the Canny algorithm, and then this threshold value can be used in the Canny algorithm to detect the object's edge. Recent investigations in a study by Yang *et al.* (2020) have provided evidence of improved Otsu threshold segmentation algorithm.

## 2.9 Summary

This literature review has revealed evidence in the use of both ML and DL for plant leaf disease detection. Figure 2.10 illustrates the overall percentage of DL and ML algorithms that were used for plant leaf disease detection in this review. The mathematical concepts dealt with for ML and DL prove that DL is a subfield of ML. For instance, the regression algorithm of ML is also used by the nodes in the ANNs. Also, the DL uses the same gradient descent methods to update its weights like ML regression algorithm. There are a lot of ML algorithms in the literature. These involve algorithms such as Decision Trees, Support Vector Machines and Forest Trees to mention a few. It can be seen in Figure 2.10 that some of these ML and DL algorithms were used in the studies related to the approaches proposed in this study. The choice of the ML or DL algorithm to use can be determined by several factors. For instance, some ML and DL algorithms can handle different types of data. Some can handle images or numerical data while others can handle both. The CNN approaches proposed in this study can only handle images. The type of images used in CNNs can either be grey scale

images or RGB colour images, but this study focuses only on colour RGB images due to the nature of the image features being studied. The concept of computer vision was explained by comparing a human eye and a camera, and how features are extracted by CNN models. A fuzzy logic technique that forms part of Chapter 4 has been reviewed and an example has been used to explain the fuzzy logic concept in detail. In this chapter, the Otsu threshold method is explained along with other methods of thresholding since Chapter 4 also involves the use of the Otsu threshold method to develop the CNN Hybrid system for maize CR disease severity prediction.

# CHAPTER 3: Convolutional Neural Network for the Classification of Maize Leaf Disease

The following chapter is based published work by:

Sibiya, M. and Sumbwanyambe, M., 2019a. A computational procedure for the recognition and classification of maize leaf diseases out of healthy leaves using convolutional neural networks. *Agri Engineering, I* (1), pp. 119-131.

Website: <https://www.mdpi.com/2624-7402/1/1/9>

Status: Published

## Abstract

Leaf diseases can affect plant leaves to a certain extent, because plants can collapse and die completely. These diseases may significantly reduce the market supply of vegetables and fruit and result in a weak agricultural economy. A variety of laboratory methods to detect foliar diseases have been used in the literature. These methods take time and cannot be used in large fields for the detection of foliar diseases. This study concentrates on the principles of the convolutional neuronal network (CNN) in order to model a network of image recognition and classification of these diseases. Neuroph was used to train a CNN network, which recognized and classified images of corn leaf diseases which have been collected using a smart camera. A new method of CNN training and methodology has been employed to accelerate the rapid and easy implementation of the system in practice. The developed model recognized three different types of diseases of corn leaves from healthy leaves. Diseases of maize known as Northern Corn Leaf Blight (*Exserohilum*), Common Rust (*Puccinia sorghi*) and Grey Leaf Spot (*Cercospora*) were selected for this study. The reason for that is because, the latter mentioned corn leaf diseases have hit most parts of the Southern Africa as the literature reveals.

## 3.1 Introduction and Related Works

Grey Leafspot (GL) is caused by a fungus called *Cercospora zea-maydis*. Today, it is considered one of the diseases that limit maize yield the most in the world. The GL has caused a significant threat to corn production in many parts of the eastern United States and, more recently, large areas of the United States (Ward *et al.*, 1999). Its symptoms are usually seen on lower leaves (Ward *et al.*, 1999). The GLS maturation stages of corn are shown in Figures 3.1 and 3.2.



**Figure 3. 1:** Immature GLS lesions on corn leaves appear as small brown blotches, often with chlorotic borders.

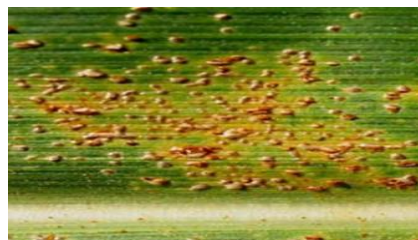


**Figure 3. 2:** The mature GLS lesions on the corn leaves are grey to tan and distinctly rectangular in shape.

Common rust (CR), otherwise caused by the pathogen *Puccinia sorghi*, is favored by cool temperatures (16-23°C) and high relative humidity (100%) (Campus 2012). Functional foliar zone and photosynthesis are reduced by disease damage. Spotting occurs on the upper and lower surface of leaves (Dilliard 1990). Figures 3.3 and 3.4 show early CR lesions and further disease development, respectively.



**Figure 3. 3:** Early lesions begin with spotting of leaves that develop into small tanning spots (CR disease).



**Figure 3. 4:** Further development of the disease with spots of serrated appearance, transforming into long brick red to cinnamon brown pustules (CR disease).

Cornfield leaf blight is a disease caused by a fungus called *Exserohilum turcicum*. The development of NCLB is influenced by cold to moderate temperatures and high relative humidity. Symptoms are identified by relatively large grey cigar-shaped lesions that can develop on leaves (Jackson 2008). The stage changes in NCLB due to fungal concentration are shown in Figures 3.5 and 3.6.



**Figure 3. 5:** Typically large, cigar-shaped, beige to grey lesions of NCLB.



**Figure 3. 6:** The fungus can cause NCLB to produce large quantities of spores on the surface of the lesions, resulting in a dark appearance.

In the literature, several machine learning algorithms have been used for identifying and detecting plant leaf diseases. With Open CV-dependent feature extraction strategies, the results of the created model reached somewhere within the range of 91% and 98% accuracy and 93% on average for isolated class tests (Sladojevic et al., 2016). Using deep learning methods, a public dataset of 54306 images of ill and healthy sheets was collected under controlled conditions and utilized for the training of a deep convolutional neural network in order to classify 14 crop species and 26 diseases. The feasibility of this approach was demonstrated by a trained model that achieved 99.35% precision on a set of pending tests (Mohanty et al., 2016). A study was conducted to apply the artificial neural network (ANN) analysis technique to distinguish and classify fungal infections in oil palms (Ahmadi et al., 2017). The datasets of the raw, first- and second-derivative spectral radiometers were used very early. They were obtained from 1.16 spectral signatures of leaf samples at four disease levels (T1: healthy, T2: slightly infected, T3: moderately infected, and T4: seriously infected) (Ahmadi et al., 2017). A web-based tool was proposed that helped farmers in the identification of fruit diseases by uploading fruit images to the system. The system used well-trained datasets of pomegranate fruit. The images to be analyzed were given by the user to undergo several processing steps in order to detect the severity of the illnesses by comparing them with the images of the dataset formed. Experimental results of the proposed approach demonstrated 82% of accuracy in determining pomegranate disease (Bhange and Hingoliwala, 2015). Very little is known about the implementation of CNN for plant disease recognition built on a framework that utilises graphical user interface (GUI) to train the network by use of raw images with feature extractions embedded in the program's library. The main objective of this study was to design a system that would recognise and classify the maize leaf diseases out of healthy leaves by means of facilitated CNN. The development and novelty of the proposed model lay in its simplicity; background images and healthy leaves were in accordance with other classes, empowering the model to distinguish between diseased leaves and sound ones or from the environment by utilizing deep CNN. The use of machine learning algorithms for



detecting leaf diseases is prevalent in the literature. It was found that machine learning models for predicting these plant leaf diseases differed in precision. A variety of techniques are currently used to recognize plant leaf diseases through the application of computer vision. One of them is disease detection by colour feature extraction from images. CNNs are known to be more effective in classifying and identifying images by extracting colour elements. For such reasons, Sladojevic *et al.* (2016) conducted CNN's in-depth training on recognition and classification of plant diseases. The results of the tests on the created model achieved an accuracy of 91% and 98%, for separate class tests, 96.3% on average. Using an open dataset of 54,306 images of diseased and healthy plant leaves collected under controlled conditions, Mohanty *et al.* (2016) trained a deep CNN to distinguish 14 crop species and 26 diseases. The trained model obtained an accuracy of 99.35% over a set of pending tests, demonstrating the plausibility of this approach. A study in comparison of support vector machine (SVM) and ANN, was performed by D.Pujari *et al.*(2016). Algorithms for colour extraction and texture characteristics have been developed and therefore used to form SVM and ANN classifiers. The study provided an approach based on a reduced set of characteristics for the recognition and classification of plant disease images. The results showed that the SVM classifier was progressively reasonable in identifying and classifying plant diseases. A SVM classifier was 92.17% compared to a ANN classifier with an accuracy of 87.4%. Recently, wheat disease detection using leaf images and data processing techniques has been widely used to help farmers monitor large acres. Researchers like Dixit and Ema have used SVM to recognize and classify wheat diseases. In their study, they elaborated on the key issues and challenges related to wheat leaf detection through the SVM (Dixit and Nema, 2018). The objective of this study is to model a facilitated CNN based on the graphical user interface for the recognition and classification of corn leaf diseases. In this study, the use of image recognition in the Neuroph Studio framework made it possible to design CNN with feature extraction functions embedded in the program library. The overall system's accuracy showed a progressive 99.9% for the classification of NCLB (*Exserohilum*), 91% of GLS (*Cercospora*), 87% of the CR (*Puccinia sorghi*), and 93.5% of healthy leaves using a maize data set of 100 images for each disease class, and 100 images for healthy class.

## **3.2 Materials and methods**

### **3.2.1 The Concept of CNN**

The regular NN (neural network) is not equipped to deal with images. If a regular NN were to be used with the images, then each pixel of the image would have to be connected to its neuron resulting in a network which would be computationally expensive. CNN processes images in different ways, while pursuing the overall idea of regular NN. Figure 3.7 illustrates an architecture for CNN.

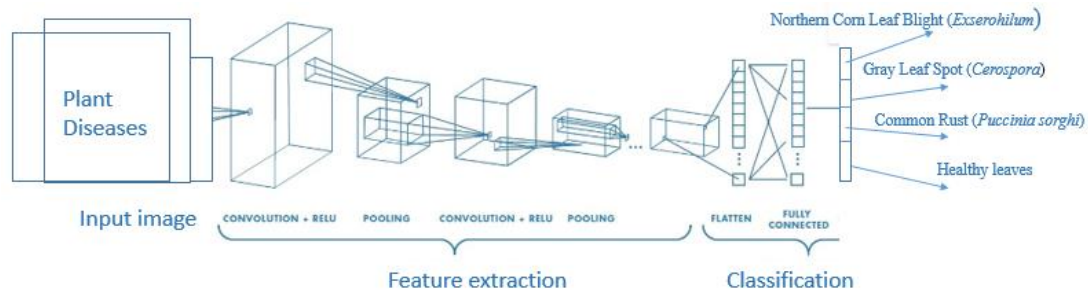


Figure 3. 1: The architecture of Convolutional neural network.

Convolution and pooling (feature retrieval): Convolution layers have a set of usable filters that are a matrix (width (W), height (H) and depth (D)). The input image is considered as a matrix and the filter is imagined sliding through the input image matrix to get the convoluted image which is the filtered image of the actual input image. If a filter is applied to the incoming image, the result will be a lower output matrix than the original image. Padding plays a major role if we are to get the same output size as the input size. Pooling is another important concept of CNN. Pooling is a form of non-linear descending sampling function which, in the case of CNNs, is applied to convoluted images. Among several non-linear functions that can perform pooling, maximum pooling is the most frequent. Classification: The input images are organized by a fully connected layer right after the convolution and max pooling layers. The neurons of a fully connected layer have connections to all the activations of the previous layer, as seen in the normal neuronal network. Figure 3.7 illustrates the diseases of corn plants that have been identified and categorized by a fully connected layer of the proposed CNN.

### 3.2.2 Materials

The Neuroph framework includes the Java neural network library and the built-in neural networks, and the Java IDE based on the NetBeans – Neuroph Studio platform. This is an integrated environment for creating and deploying neuronal networks for Java programs. Neuroph supports common neural network architectures, and it is very flexible so it can be easily expanded to meet specific needs. Figure 3.8 shows the basic structure for frame packages.

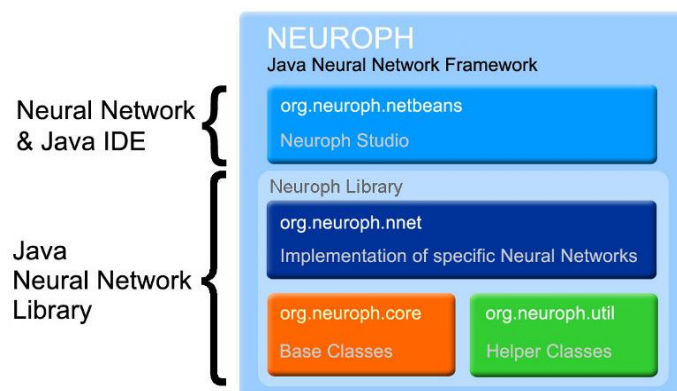


Figure 3. 2: The typical framework of the Neuroph platform.

### 3.2.3 Methodology

A CNN was developed on the Windows PC platform. The framework of the Neuroph Studio was used to construct a CNN with 50 hidden layers for the recognition and classification of diseases of corn leaves on healthy leaves. The network was formed using 100 colour images of each disease and healthy leaves with resolution parameters configured like 10x20x3 (width-to-height channels). Using the image resolution parameters mentioned above, the CNN was modelled to have 600 inputs across the neural array classifier input layer. Out of 100 images available for each class to be recognized, 70% were used for training and 30% for testing the CNN. Backpropagation was used as the learning algorithm for forming the CNN network. A mathematical model of the proposed CNN was implemented as shown below: The order made by the CNN was a tensor of order 3 as its input. Input images that were used for testing of possible disease conditions were images with H rows, W columns and 3 channels (RGB colour channels). A summary of the proposed CNN model is shown in equation (3.1).

$$x^1 \rightarrow w^1 \rightarrow x^2 \rightarrow \dots \rightarrow x^{L-1} \rightarrow w^{L-1} \rightarrow x^L \rightarrow w^L \rightarrow \quad (3.1)$$

Equation (3.1) illustrates how the proposed CNN model operates layer by layer. The input  $x^1$  was an image of a maize leaf disease or a sound disease with tensors of order 3. The  $x^1$  was the input to the first layer's input collectively known as tensor  $W^1$ . The output of the first layer was  $x^2$  which also acted as an input to the second layer processing. The processing proceeded until all layers in the CNN had been finished, which gave an output of  $x^L$ . An extra layer, however, was added for the propagation of errors back to learning parameter values in the CNN. The last layer was a loss layer. As an equation (3.2) illustrates, a single loss function is used.

$$z = \frac{1}{2} \| t - x^L \|^2 \quad (3.2)$$

In equation (3.2),  $t$  denotes the corresponding target value for the input  $x^L$ . The loss function formula in equation (3.2) was used to measure the discrepancy between CNN prediction  $x^L$  and  $t$ . Alternatively, the output prediction of the CNN was given as shown in equation (3.3).

$$\text{arg}_i \max x_i^L \quad (3.3)$$

The loss layer was not needed in the prediction, but was useful in learning of CNN parameters using a set of diseased and healthy maize leaf data sets as training sets. The proposed CNN model used stochastic gradient descending (SGD) to learn the model's parameters. Rather than generating a prediction, we compared the prediction with the  $t$  target corresponding to the input. Loss  $Z$  was then a supervisory signal, guiding the way the model parameters should be updated. The manner in which SGD changed the parameters is presented in equation (3.4).

$$w^i \leftarrow w^i - \eta \frac{\partial z}{\partial w^i} \quad (3.4)$$

where  $\eta$  represents the learning rate. The learning rate was chosen to be 0.01. Equation (3.5) has a superscript “time” index (e.g., training epochs/iterations).

$$(w^i)^{t+1} = (w^i)^t - \eta \frac{\partial z}{\partial (w^i)^t} \quad (3.5)$$

The activation functionality used in the convolution layer was the corrected linear unit (ReLU). The ReLU can be considered as a truncation performed separately for each element of the input. ReLU is shown as equation (3.6).

$$j_{i,j,d} = \max \{0, x_{i,j,d}^l\} \quad (3.6)$$

The  $l$ -th layer, had inputs that formed an order 3 tensor  $x^l$  with  $x^l \in R^{H^l \times W^l \times D^l}$ . Thus, for this reason we needed a triplet index set  $(i^l, j^l, d^l)$  to locate any specific element in  $x^l$ . According to equation (3.6), it is evident that in equation (3.7),

$$\frac{\partial y_{i,j,d}}{\partial x_{i,j,d}^l} = \llbracket x_{i,j,d}^l > 0 \rrbracket \quad (3.7)$$

where  $\llbracket . \rrbracket$  was the indicator function, being 1 if its argument was true, and 0 otherwise. Hence, we had

$$\left[ \frac{\partial z}{\partial x^l} \right]_{i,j,d} = \begin{cases} \left[ \frac{dz}{dy} \right]_{i,j,d} & \text{if } x_{i,j,d}^l > 0 \\ 0 & \text{otherwise} \end{cases} \quad (3.8)$$

$y$  is an alias of  $x^{l+1}$ . An alias means that one variable can be remodelled into another shape. Specifically, the max function  $(0, x)$  cannot be differentiated to  $x = 0$ . Focusing on the convolution layer, the standardized kernel was used to collect input images of diseased and healthy corn leaves. The convolution procedure is explained in equation (3.9).

$$(x, y) = \sum_{m=-\frac{M}{2}}^{\frac{M}{2}} \sum_{n=-\frac{N}{2}}^{\frac{N}{2}} h(m, n) f(x - m, y - m) \quad (3.9)$$

$h(m, n)$  is a filtering mask of size  $M \times N$ . Each element of this filter mask represents the weights used in the linear blend. It was at this point that the ReLU activation function was used with the convoluted input images

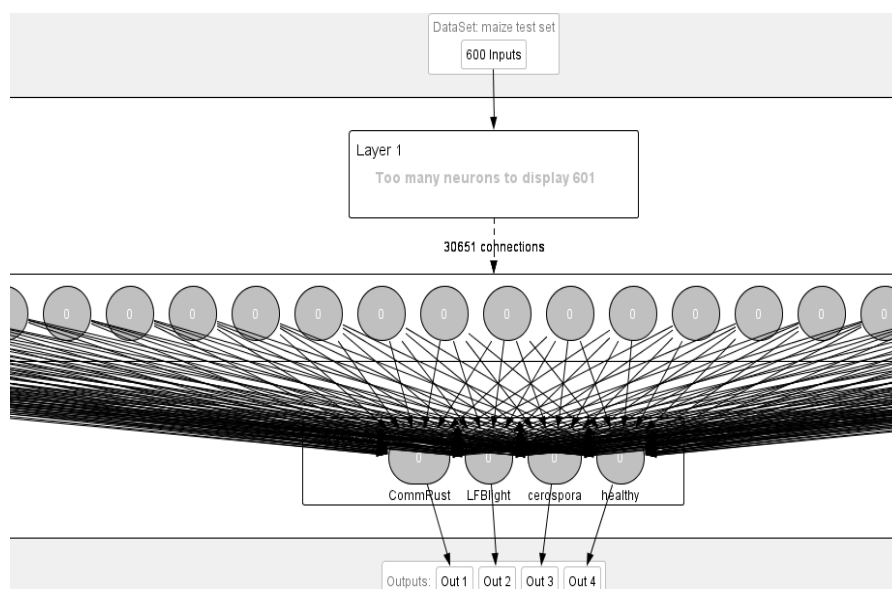
of diseased and healthy corn leaves to provide non-linearity. The filter that was used for the convolution of the input images in this study is illustrated by equation (3.10).

$$Kernel_{normalised} = \begin{bmatrix} -1 & -1 & -1 & -1 & -1 \\ -1 & -1 & -1 & -1 & -1 \\ -1 & -1 & 24 & -1 & -1 \\ -1 & -1 & -1 & -1 & -1 \\ -1 & -1 & -1 & -1 & -1 \end{bmatrix} \quad (3.10)$$

As described above in this study, pooling was used to sample the convoluted non-linear image to form an input signal entered into a fully connected classifier.

### 3.2.4 Data Collection and Testing of the CNN

The images to be analysed for possible existence of the maize diseases were captured using a Google Pixel 3 smart phone camera in a maize field and saved in a file located on Google Drive. To analyze the data collected in the field of maize, the Neuroph CNN was used on the computer station. Figure 3.9 illustrates a fully connected CNN classifier.



**Figure 3. 9:** The neural array classifier after convolution and maximum pooling has been completed in the Neuroph library.

Figure 3.10 shows how the Google Drive file was accessed to select one of the field images that was analyzed for the possible existence of corn leaf diseases. The output window of the result shows that the CR had a strong probability of 0.8 followed by the NCLB with a probability of 0.5. By continually accessing a file downloaded from Google Drive containing the field data, the field images collected were all analyzed for the possible existence of the diseases.



**Figure 3. 10:** Image recognition and classification test in the context of the Neuroph.



**Figure 3. 3:** GLS (*Cercospora*).



**Figure 3. 4:** CR (*Puccinia sorghi*).



**Figure 3. 5:** Healthy.

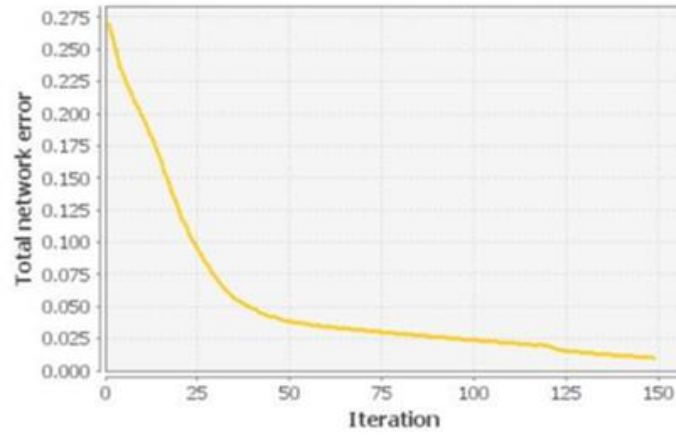


**Figure 3. 6:** NCLB (*Exserohilum*).

Figures 3.11-3.14 present some of the test images of the corn test dataset that were analyzed using the proposed CNN model.

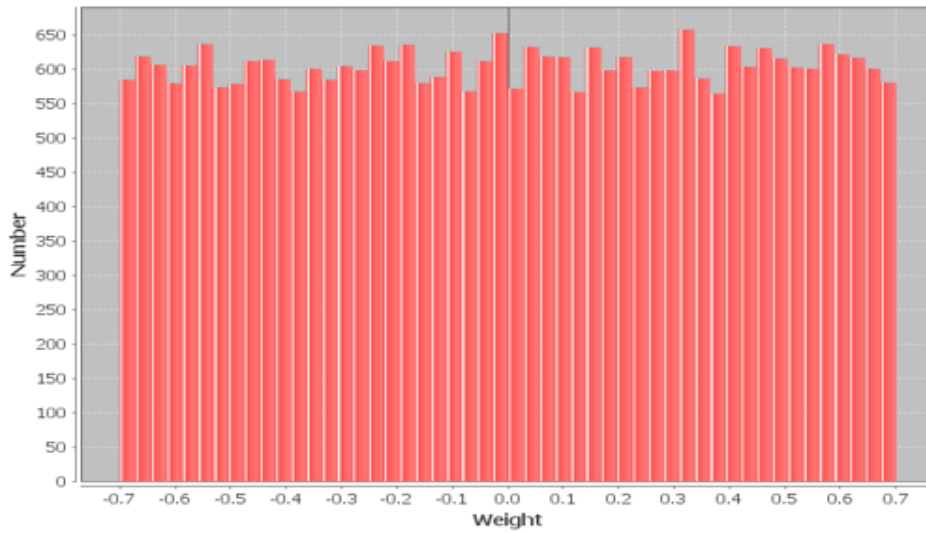
### 3.3 Results

Figure 3.15 shows the CNN training iterations performed to minimize the error from 0.275 to a reduced error of 0.001. As shown in Figure 3.15, an approximate total of 150 iterations was carried out to reduce the error to 0.01.



**Figure 3. 15:** Graph of total network errors during and after training the CNN network on a Neuroph frame.

Figure 3.16 illustrates the weight assignment for each of the 50 hidden layers that were used to build the CNN for the recognition and classification of maize diseases out of healthy leaves.



**Figure 3. 16:** Weight allocation for each of CNN's 50 hidden layers.

Table 3.1 provides results for separate class tests. The CNN was very precise in recognizing and classifying the NCLB (*Exserohilum*) with an exceptional accuracy of 99.9%. Some of the test images showed the properties of GLS (*Cercospora*) and CR (*Puccinia sorghi*). This is one reason why CNN underperformed when it comes to classifying the two diseases.

Table 3. 1: Accuracy of CNN results in classifying and recognizing diseases of healthy corn leaves and leaves.


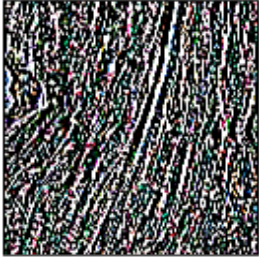
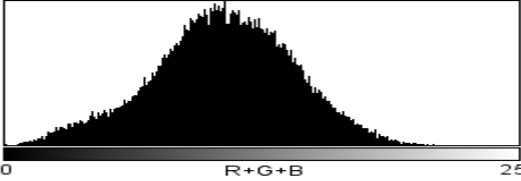

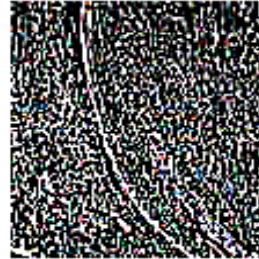
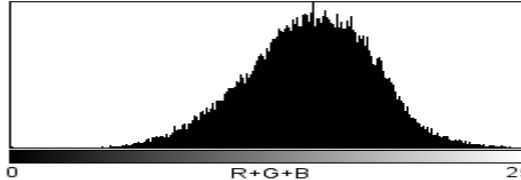


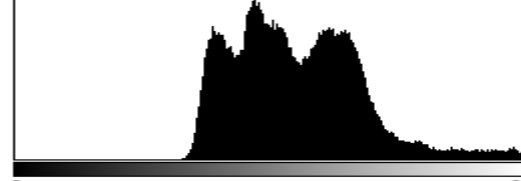

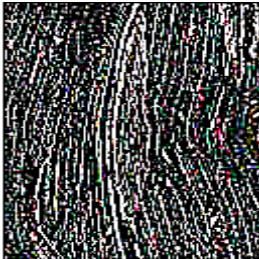
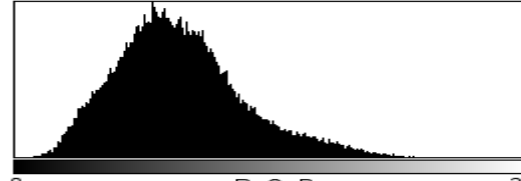
Type of Maize Disease	Total Percentage of Training Images	Total Percentage of Testing Images	CNN Classifier Accuracy
NCLB.	70%	30%	99.9%
GLS.	70%	30%	91%
CR.	70%	30%	87%
Healthy.	70%	30%	93.5%

The CNN's overall accuracy was determined by equation (3.11).

$$\text{Overall accuracy} = \frac{\sum \text{classifier accuracies}}{\text{Average}} \times 100, \quad (3.11)$$

The overall accuracy for the CNN classifier was 92.85%. Test results for the proposed CNN model were in the range of 87% and 99.9% of tests of distinct classes and 92.85% on average. Table 3.2 shows the results of the convoluted input images as well as the results of the histograms of the input images via the proposed CNN.

Table 3. 2: Array of convoluted results and histogram results of some of the test images selected via CNN.

Input Image	Convolution Result	Histogram Result
		 Count: 66048      Min: 0 Mean: 107.868      Max: 235 StdDev: 35.426      Mode: 109 (814)
		 Count: 42510      Min: 0 Mean: 148.179      Max: 255 StdDev: 32.122      Mode: 150 (583)
		 Count: 165675      Min: 79 Mean: 140.507      Max: 255 StdDev: 32.954      Mode: 119 (2293)
		 Count: 79779      Min: 0 Mean: 81.567      Max: 217 StdDev: 32.306      Mode: 68 (1180)

### 3.4 Discussion and Conclusion

These results further strengthened our hypothesis that the CNN could recognize and classify corn leaf diseases with an overall accuracy of 92.85%. The CNN was trained using about 150 iterations to reach a minimum error of 0.01. This proved that the model was rapid in learning training data. In separate class tests, the model obtained accuracies of 99.9% for NCLB, 91% for GLS, 87% for CR and 93.5% of healthy corn leaves. This



was achieved by using a batch of 100 images for each disease-class and 100 healthy images to train the CNN. Various images of each lot were used to conclude the feasibility of the research and the results obtained with respect to accuracy. This study focused on three maize disease types that were caused by biotic stresses. However, the proposed CNN would also be used to recognize the diseases that could be the cause of the abiotic stresses if it were to be trained with the data collected from the abiotic stressed plants. Future research is proposed to determine the performance of the CNN, which would be formed by biotic stress data and then tested against abiotic stress data. In the literature, there are numerous machine learning studies that explain the detection of plant leaf diseases. However, so far, none have investigated the detection of corn diseases in large open corn fields. Using deep CNN built in the Neuroph studio and a Google Pixel 3 smart phone we managed to conduct a study to detect the three types of maize diseases that occurred in the large open maize fields. The corn field was divided into 10x10 square meter section areas and the data was collected using a Google Pixel 3 smartphone of each section area. The database used for storing the collected data was Google Drive. During the training and testing of CNN, the data were extracted from Google Drive and used for training or testing CNN. The proposed method eliminates the use computation methods and cameras for data acquisition. The proposed method is more accurate than other methods that utilize data acquisition computational methods as the data was collected by the user from any angle of the leaves. Another advantage is that our proposed CNN was built in a GUI platform. This will enable people who are not familiar with high level programming languages, such as Python, MATLAB and C to mention a few, to build a CNN from scratch. In this study, we have reviewed related works in the literature for the plant leaf disease classification algorithms of machine learning. The Neuroph Studio framework was used as an IDE to build a more facilitated deep CNN whereby the convolution and pooling feature extractions were embedded in the Neuroph library. The proposed CNN was formed and tested using the data sets on the Plant Village website. The overall accuracy of 92.85% of CNN has shown its feasibility. The CNN was also tested using the data collected for maize and the results were compared with the results obtained when using the test data. It is recommended that the researchers who wish to use the proposed CNN in this study use the resolution settings of  $10 \times 20 \times 3$  (height  $\times$  width  $\times$  RGB). Future research is proposed to determine the performance of CNN when training and testing is performed with grayscale images.

# CHAPTER 4: Fuzzy Logic Rules for severity estimation of maize disease based on Thresholding

The following chapter is based on works published based by:

Sibiya, M. and Sumbwanyambe, M., 2021. Automatic Fuzzy Logic-Based Maize Common Rust Disease Severity Predictions with Thresholding and Deep Learning. *Pathogens*, 10 (2), p. 131.

Link: <https://www.mdpi.com/search?authors=malusi+sibiya&journal=pathogens>

Status: Published

Sibiya, M. and Sumbwanyambe, M., 2019b. An algorithm for severity estimation of plant leaf diseases by the use of colour threshold image segmentation and fuzzy logic inference: a proposed algorithm to update a “leaf doctor” application. *Agri Engineering*, 1 (2), pp. 205-219.

Website: <https://www.mdpi.com/2624-7402/1/2/15>

Status: Published

## Abstract

Many applications of plant pathology have been strengthened by the advancement of computer-based artificial intelligence (AI). For example, many experts and researchers had used pre-trained convolutional neural networks (CNN) such as VGG-16, Inception and Google Net for the detection of plant diseases. The trend for using AI for the detection of plant diseases has grown to such an extent that some researchers have been able to use AI to also detect their severity. The goal of this study is to introduce a new methodology that is reliable in detecting the severity of common corn rust infection using a CNN deep learning model. To do so, images of diseased corn leaves (common rust disease) were segmented to extract the percentage of diseased leaf surface. These calculated percentages were used to develop fuzzy decision rules for allocating Common Rust images based on their severity classes. The four severity classes were then used to prepare a VGG-16 network to automatically categorize common rust test images by their severity classes. Trained with images developed by using the approach proposed in this study, the VGG-16 network's validation accuracy was 95.63% and a testing accuracy of 89%. These accuracies were achieved when the VGG-16 was tested on images of the Common Rust disease, classifying them into four classes of Common Rust severity named as the Early stage, Middle stage, Late Stage and Healthy stage.

## 4.1 Introduction and Related Works

Researchers who have used CNN models to characterize the severity of leaf disease have so far used training datasets organized in classes that rely on human observations to make decisions. Not only is this method biased, but it is unreliable because some human decisions may be inaccurate due to possibilities such as visual

impairment. In a study by Wang *et al.* (2017) for example, VGG-16 was used to classify the severity in apple diseases. In this study, Wang *et al.* (2017) depended on a botanist's decision on the severity stages of diseased apple leaves. The following discretionary powers were taken by the botanist to make decisions regarding severity classes: the healthy-stage leaves are free of spots; the early-stage leaves have small circular spots with diameters less than 5 mm; the middle-stage leaves have more than three spots with at least one frog-eye spot enlarging to irregular or lobed shape; the end-stage leaves are so heavily infected that they will drop from the tree. Through these decisions made by their botanist, they developed the training datasets used to train the VGG-16 network with 90.4% test accuracy. The proposed approach was not generic in that the methods used to attribute images of diseased plants to their severity classes were only available for apple leaf diseases. In addition, the assignments according to severity were based on a human eye, so in this study, a new approach to using the decisions of computer fuzzy rules is introduced. These fuzzy rules were used to assign Common Rust corn images to their classes according to severity classes to train the VGG-16 network. The method proposed in this study is based on the image threshold for common rust-affected corn leaves. The percentages of the diseased leaf area were used to develop fuzzy decision rules that allowed the training images to be attributed to their severity classes. After the development of severity classes, the VGG-16 network was trained and refined to classify common rust images of corn. The image training data were categorized into four common rust severity classes: early stage, intermediate stage, late stage, and healthy stage. With this approach in hand, the RGB images of Common Rust were first converted into a grey scale using Otsu threshold segmentation. The Otsu method segmented the images into two classes of dark intensity (Background) pixels and bright intensity (Fore ground) pixels that could be used. The Agricultural Research Council (ARC), South Africa, was the principal source of information on Common Rust Disease (CR) that we intended to use in this study. Common corn rust results from a fungus called *Puccinia sorghi*. The CR is favoured by cool, wet weather conditions of about 59 to 72°F. On numerous occasions, it targets coastal regions in Durban, South Africa. Advances in technology and the emergence of artificial intelligence (AI) have made it possible for scientists and researchers to detect plant diseases using CNNs. CNN is an automated learning algorithm which is known for its application in computer vision applications. For instance, Zhang *et al.* (2018) developed a model for abnormally identifying the breast using a nine-layer convolutional neural network. However, in the context of this study, work on deep learning for the detection of plant diseases after the images have been reviewed. A Faster R-CNN architecture was updated by modifying the parameters of a CNN model. This Faster R-CNN architecture has been developed to automatically detect leaf spot disease (*Cercospora beticola* Sacc) in sugar beet, as proposed by (Ozguven and Adem, 2019). Their proposed model was formed and tested with 155 images to detect the severity in sugar beet. The total accuracy of the prediction was observed to be 95.48%. Cruz *et al.* (2017) developed a vision-based system to detect symptoms of leaf burn on leaves of *Olea europaea* infected with *Xylella fastidiosa*. (2017). In this work, the algorithm found low-level characteristics from raw data to automatically detect veins and colours that lead to symptomatic leaves. A foliar burn was

found with a true positive rate of 98.60 1.47%. The model was developed with a convolutional neural network that was trained in the stochastic gradient descent method (Cruz *et al.*, 2017). The literature shows that the deep learning models that were used to detect plant diseases had different performance accuracies that were determined by the methods of setting and regulating the model parameters. Chen *et al.* (2020). used the pre-trained networks with an image network, a famous dataset that labelled images ready to be used as training data. Their approach has improved performance over other leading-edge techniques and achieved at least 91.83% validation accuracy when trained on the public dataset. With a complex background in the included images, the proposed approach achieved a normal 92.00% accuracy for the rice seedling image class (Chen *et al.* (2019) used deep learning to identify potato tuber disease. The baseline architecture selected for this issue was a CNN developed by the Visual Geometry Group (VGG) (Ozguven and Adem, 2019). In their model, several new dropout layers were added to the VGG architecture to deal with problems of overfitting, especially due to the relatively small data set. A VGG array with 224x224 image resolution was used in the study. The CNN comprised eight learnable layers, the first five of which were convolutional, followed by three fully connected layers and ending with a soft max layer (Oppenheim *et al.*, 2019). Training CNNs usually requires large amounts of labelled data are usually required to train CNNs such that they perform better in making classifications (Oppenheim *et al.*, 2019). Therefore, two methods were used for data augmentation: mirroring creates additional examples by randomly flipping the images used in training; cropping was also used, cropping the image randomly to different sizes, while keeping the cropped image's minimum size to 190 × 190, helped to achieve data diversity (Oppenheim *et al.*, 2019). Arsenovic *et al.* (2019) conducted a study around solving the current limitations on the use of deep learning for the detection and classification of plant diseases. In their work, they used two approaches to address the question of d augmentation. The first one was the augmentation by means of traditional methods, and the second one, the art style Generative Adversarial Networks (GANs) (Arsenovic *et al.*, 2019). A summary of the deep learning in classifying plant diseases without considering their severity is presented in Table 4.1.

Table 4. 1: A summary of deep learning models used in plant disease classification.

Deep Learning Architecture/Image Database	Study Summary
Google Net	Twelve plant species were used, each with a different number of samples and diseases under changing circumstances (Barbedo 2018).
U-Net	After using U-Net to acquire images of the segmented leaves, the next step was to identify the types of plant diseases. Based on (Huang et al., 2019), it was an ideal image classification task (Huang et al., 2019).
Network (DCNN)	Using hyperspectral imaging of inoculated and simulated strain images, DCNN 3D had a classification accuracy of 95.73%. The infected class F1 score of 0.87 was achieved (Nagasubramanian <i>et al.</i> , 2019).
ImageNet	(Brahemi et al., 2017) used the ImageNet 1000 Class database with a predetermined model for classifying nine types of tomato disease.
Dense Net	A Light Deep Neural Array (DNN) approach which can operate on Internet of Things (IoT) devices with limited resources has been proposed (Ale et al., 2019).
9-Layer Deep CNN	The deep CNN model was built using an open data set comprising 39 different classes of plant leaves and background images. Six types of data augmentation methods were used: image flipping, gamma correction, noise injection, principal component analysis (PCA) colour augmentation, rotation, and scaling. The proposed model achieved a classifying precision of 96.46% (Geetharamani and Pandian 2019).
Deep Siamese convolutional network	The Deep Siamese Complex Network has been developed to address the issue of small image databases. Greater than 90% accuracy was achieved in detecting Esca disease, black rot and chlorosis on grape leaves (Goncharov et al., 2018).
CNN API written in Python	The model has been designed to detect and recognize several plant varieties, especially apples, corn, grapes, potatoes, sugarcane and tomatoes. The developed model obtained a 96.5% accuracy rate, and the system was able to register up to 100% accuracy in detecting and recognizing the plant variety and type of disease from which the plant has been infected (Militante et al., 2019).
Alex Net and Google Net	Using a public data set of 54306 images of infected and healthy plant leaves collected under controlled conditions, A deep convolution neural network was formed to identify 14 crop species and 26 diseases (or their absence). The trained model achieved an accuracy of 99.35% on a held-out test set, demonstrating the feasibility of this approach (Mohanty <i>et al.</i> , 2016).
Alex Net, VGG16, and VGG19	The experiments were carried out using data, including actual pictures of the disease and parasites from Turkey. Accuracy, sensitivity, specificity and F1 scores have all been calculated for performance evaluation (TÜRKOĞLU and Hanbay, 2019).
Alex Net, Alex Net OWTBn, Google Net, Over feta and VGG	Convolutional neural network models have been created in classification of plant diseases using basic foliar images of healthy and infected plants, using deep learning methods. The models were prepared with an open database of 87848 images containing 25 single plants in a group of 58 individual classes of mixtures of [plants, diseases], including healthy plants. Some model architectures have been formed, with the best performance achieving a 99.53% success rate in the identification of the corresponding [plant, disease] combination (or healthy plant) (Ferentinos, 2018).
CNN model similar to standard Le Net architecture	Apple leaf images, covering various diseases, along with healthy samples, from the PlantVillage dataset were used to validate the results. Image filtering, image compression and image generation techniques were used to acquire a wide range of training images and

	<p>fine-tune the system. The developed model achieved high precision scores in all classes, with a clear accuracy of 98.54% across the dataset, sampled and generated from 2,561 labelled images (Baranwal et al., 2019).</p>
<p>Alex Net and VGG-16</p>	<p>In this study, images of tomato leaves (6 illnesses and a healthy class) were obtained from the PlantVillage dataset and were supplied as inputs for two architectures based on deep learning, namely, Alex Net and VGG16 net. The classification accuracy achieved using Alex Net and VGG16 net was 97.49% and 97.23%, respectively, for 13,262 images (Rangarajan et al., 2018).</p>
<p>VGG 16, Inception V4, Res Net with 50, 101 and 152 layers and Dense Nets with 121 layers</p>	<p>In this paper, a strategic comparison of the Deep Learning architecture was carried out. The evaluated architectures consisted of VGG 16, Inception V4, Res Net with 50, 101 and 152 layers and Dense Nets with 121 layers. The data used in the experiment were 38 different classes, including pictures of diseased and healthy leaves of 14 plants from PlantVillage. Fast and accurate models for the identification of plant diseases were desired, so that accurate measurements can be implemented quickly. In the experiment, Dense Nets tended to constantly improve in precision with the growing number of epochs without signs of overfitting. In addition, Dense Nets needed a significantly smaller number of parameters and a reasonable computational time to achieve peak performance. It was 99.75% accurate to beat the rest of the architectures (Too et al., 2019).</p>
<p>Google Net</p>	<p>The work looked at the use of individual lesions and spots for the task, rather than thinking about the whole leaf. Each region had its own qualities; thus, the inconsistency of the data was extended without the need for additional images. This has also led to the identification of several diseases on the same leaf. However, proper segmentation of symptoms still had to be done manually, preventing complete automation. The accuracy obtained from this approach was, on average, 12% higher than that obtained from the original images. In addition, no culture was less than 75% accurate, although 10 diseases were considered. Even though the data base did not cover the full range of practical possibilities, these findings indicate that, if sufficient data were available, the deep learning technique is effective in detecting and recognizing plant diseases (Barbedo, 2019).</p>

Table 4.1 summarizes the classification of plant diseases without regard to their severity. Table 4.2 summarizes the work that was used for the prediction of plant disease severity.

Table 4. 2: A summary of methods for plant disease severity detection.

Methods Used for Plant Disease Severity Prediction	Study Summary
Using deep learning to automatically to automatically predict plant disease severity.	Various deep learning methods were developed by Wang, Sun and Wang, including a VGG-16 that was used to detect the plant severity in four steps. The VGG-16 outpermed the other models with a validation accuracy of 90.4% (Wang <i>et al.</i> , 2017).
Image processing to measure the symptoms of foliar disease.	A simple threshold method was performed to segment the leaf area and the lesion region area by using the triangle threshold method. The quotient of lesion area and leaf area were used to categorize the diseases with testing accuracy of 98.60% (Patil and Bodhe, 2011).
Use of digital image processing to measure leaf disease symptoms.	Image processing and measurements were used by to detect severities of plant leaf diseases. Using his method, he received an accuracy of 96% to detect the severities of the plant leaf diseases he dealt with. However, his method also has a disadvantage of limiting users who are not scientists or familiar with image processing (Barbedo,2014).
Plant disease incidence and severity measurements by use of machine learning.	Machine learning algorithms were employed by Owomugisha and Mwebaze. These were algorithms such as the nearest supporting vector machines and k neighbors for detecting plant disease incidents and severity measurements. Traditional machine learning algorithms that required the extraction of hand-made functions were used in this work. The handcrafted feature extraction algorithms they used for colour extraction in the images were SURF, SIFT, OBR and HOG (Owomugisha and Mwebaze, 2017).
Segmenting the affected zone.	In this work, two cascaded classifiers were used. The first classifier segmented the leaf from the background using local statistical features. Thereafter, using hue and luminance from the Hue-Saturation-Value colour space, another classifier was trained to detect disease and its stage of severity (Parikh <i>et al</i> , 2016).
Image segmentation and colour detection.	Different new boundaries, to be specific disease severity index (DSI), infection-per region (IPR), and disease-level parameter (DLP) for measuring the disease severity level and level-classification were formulated and derived (Shrivastava <i>et al.</i> , 2015).

Fuzzy logic of AI provides a platform for valuable reasoning. In simple terms, fuzzy logic is essentially a method of reasoning in a humane manner. No different from how humans make decisions, fuzzy logic implies all the intermediary possibilities between "YES" and "NO". Fuzzy logic reasoning provides an acceptable reasoning which can help address uncertainty in engineering. Conclusively, the fuzzy logic architecture consists of all the if-then rules and conditions that human experts use to control the decision-making system. In a study by Behera *et al.* (2018), it is reported that fuzzy logic was invented by Lotfi Zadeh. Lofti observed that, in contrast to computers, humans have a different spectrum of possibilities between "YES" and "NO". Because of that, Behera *et al.* (2018). have been able to use multi-class vector support machines (SVM) with clusters of K means for disease classification with 90% accuracy, and a fuzzy logic for determining the severity of orange disease. Other researchers such as Sannakki *et al.* (2011) proposed an image-based approach to automatically classify disease spread on plant leaves using fuzzy logic. Rastogi, Arora, and Sharma used

Matlab to perform K-means based segmentation and classification, percentage infection calculation, and disease grading using the fuzzy logic toolbox (Rastogi *et al.*, 2015). The threshold grouping method called the Otsu method, is based on selecting a threshold value to split the image into two classes in a manner which minimizes the variance within each class. Selecting a threshold value changes the difference between the two parts of the distribution, whereas distributions cannot be changed for obvious reasons. The key is to choose a threshold that minimizes the cumulative variance. The weighted total of variances for each class defines the variance in the class:

$$\sigma_{within}^2(T) = n_B(T)\sigma_B^2(T) + n_F(T)\sigma_F^2(T) \quad (4.1)$$

where:

$$n_B(T) = \sum_{i=0}^{T-1} p(i) \quad (4.2)$$

$$n_F(T) = \sum_{i=T}^{N-1} p(i) \quad (4.3)$$

$$\sigma_B^2(T) = \text{Background pixel variance} \quad (4.4)$$

$$\sigma_F^2(T) = \text{Foreground ground pixels} \quad (4.5)$$

The equations above require a very expensive calculation of the variance within the class for each class, and for each possible threshold value, which needs to be avoided. To reduce the computational cost, the class variance calculation which is a cheaper step can be defined as the variance within the class subtracted from the total.

$$\sigma_{Between}^2(T) = \sigma^2 - \sigma_{Within}^2(T) = n_B(T)[\mu_B(T) - \mu]^2 + n_o(T)[\mu_o(T) - \mu]^2 \quad (4.6)$$

where  $\sigma^2$  is the combined variance and  $\mu$  is the combined mean. Class variance is the weighted variance for cluster averages around the overall average. By substituting  $\mu = n_B(T)\mu_B(T) + n_o(T)\mu_o(T)$  and simplifying the result, we get

$$\sigma_{Between}^2(T) = n_B(T)n_o(T)[\mu_B(T) - \mu_o(T)]^2 \quad (4.7)$$

Thus, for each potential threshold, the algorithm separates the pixels into two clusters, as a function of the value (Otsu, 1979).



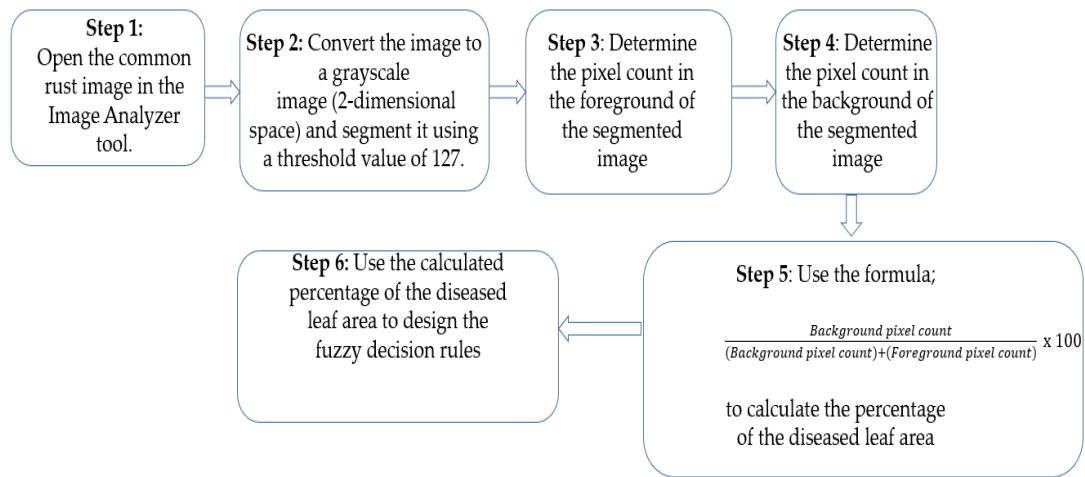
## 4.2 Materials and methods

Other researchers in the literature, for example, Weizheng *et al.* (2008), Chaudhary *et al.* (2012) and Zhou *et al.* (2019). Symptoms of plant diseases may include a detectable change in the colour, form or function of the plant in response to the pathogen Isleib (2012). This colour change causes the green pigment to be lost from the leaves. This proposed approach was conducted using the Otsu threshold segmentation method that was used to extract the percentages of infection from the infected leaf region (background pixels). These infection percentages were then used to derive fuzzy decision rules for attributing common rust images to their severity classes. The 4 severity classes that were developed were then used to train a VGG-16 network to automatically classify the test images of the Common Rust. This allowed the VGG-16 network to predict the severity of common corn rust among 3 classes, called Early-Stage, Middle-Stage and Late-Stage. The fourth stage, known as the healthy stage, was aimed at predicting healthy corn leaves. This is why the VGG-16 has been designed to be a 4-category classifier. The materials that were used in this study are tabulated in Table 4.3. Image Analyzer is basic, yet incredibly effective, free software for high-level analysis, editing and image optimization. As a free tool, Image Analyzer may be found at <https://image-analyzer.en.softonic.com/>. The PlantVillage data set has grown in popularity and has been used extensively by many machine learning researchers. The data used in this study are available at <https://github.com/spMohanty/PlantVillage-Dataset>.

Table 4. 3: A table summarising the materials used in the study.

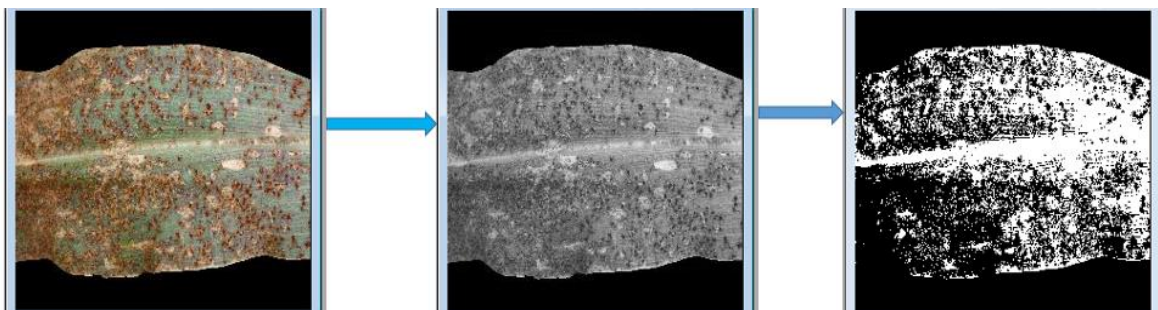
Hardware	Software	Data set
ASUS TUF Laptop with NVIDIA GeForce GTX 1650 4GB Graphics.	Image Analyzer, an open-source image processing tool.	A standard PlantVillage maize data set.
Samsung A-30 16 Mega Pixels rear camera	Jupyter notebook IDE	
	Python-Keras library	
	Python-Tensor flow library	
	Anaconda Package	

To carry out the Otsu threshold on the images, we used an open-source program called "Image Analyzer". Otsu thresholding assumes that there are two classes of pixels in the image which need separating, thus an Otsu global value of 127 was used under all conditions. Figure 4.1 sets out the procedure for the proposed approach.

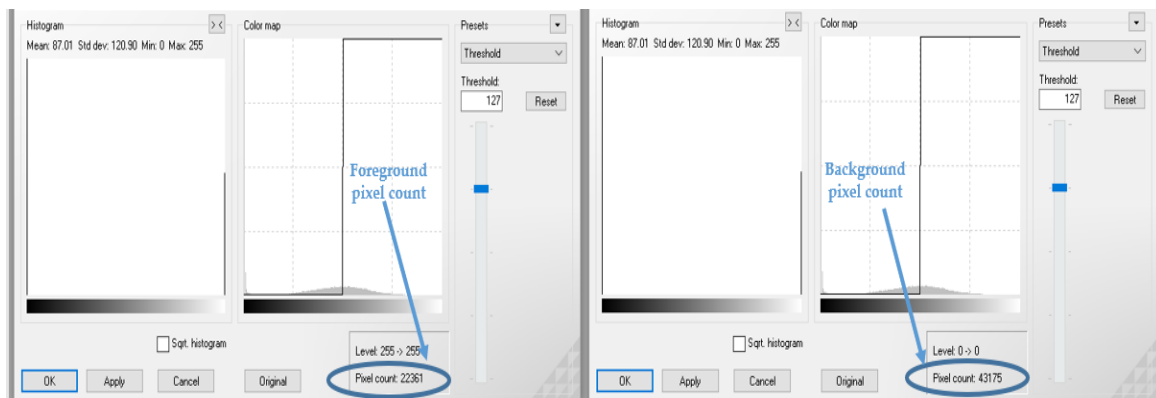


**Figure 4. 1:** The procedure of the proposed approach.

The procedure for the proposed approach is described in Figure 4.1 and shown in Figures 4.2 and 4.3 using a maize common rust disease assigned to the Late Stage as an example.



**Figure 4. 2:** Procedure outlined in Figure 4.1.



**Figure 4. 3:** Procedure described in Figure 4.1, cont'd.

Figures 4.2 and 4.3 provide a visual representation of the procedure explained in Figure 4.1 in four steps. Step 5 is explained using a formula under equation (4.8). The advantage of using colour image segmentation is that it is based on the colour features of the image pixels, assuming that homogeneous colours in the image correspond to separate clusters and hence meaningful objects in the image (Khattab *et al.*, 2014). However, for reasons of simplicity, in this study, we introduce the proposed approach, first converting the images into two-dimensional space before segmentation. Common rust images were segmented into grayscale using the Otsu method to a threshold value of 127 and percentages of infection in diseased leaf areas were calculated.

The latter was accomplished using equation (4.8). Then, once segmentation was complete, the dark intensities or background pixels displayed the areas of diseased leaves in the images. The non-diseased leaf areas, supposedly green, were presented by the light intensities or foreground pixels. The images used in this approach had a pixel size of 256x256. Furthermore, the images used for this approach should have a spatial dimension of 256 pixels in either the x-dimension or the y-dimension. Figure 4.2 shows how the spatial dimension of 256 pixels is covered by the image in the x-dimension. At least the background of the pictures should be black.

$$\%Diseased\ Leaf\ Area = \frac{Background\ pixel\ count}{(Background\ pixel\ count) + (Foreground\ pixel\ count)} \times 100 \quad (4.8)$$

Based on the fuzzy decision rules that will be explained below, an image of common rust disease is illustrated in Figures 4.2 and 4.3 would belong to a Late Stage because a percentage of 65.8% was calculated by means of equation (4.8) as follows:

$$\begin{aligned} \%Diseased\ Leaf\ Area &= \frac{43175}{43175 + 2232} \times 100 \\ &= 65.8\% \end{aligned}$$

Finally, step 6 involves deriving fuzzy decision rules for the attribution of common corn rust images to their severity classes. With the help of a specialist in plant pathology, fuzzy decision rules were established. These rules were established for the compilation of training datasets that were classified as Early Stage, Middle Stage and Late Stage. The technique employed to derive these rules of fuzzy logic was based on the experience of the plant pathologist. These techniques involved conventional methods of detecting the severity of plant diseases, such as observing the rate at which rust is dispersed over the leaf. The rules of fuzzy logic may vary depending on the plant species and the type of disease involved. With regard to the "Healthy Stage", the training data were compiled using healthy images within the PlantVillage dataset. The compilation of the training datasets was carried out using the fuzzy decision rules described below.

Design Rules for Healthy Prediction:

Rule 1: As for the "Healthy Stage", the training data were compiled by use of healthy images in the PlantVillage dataset.

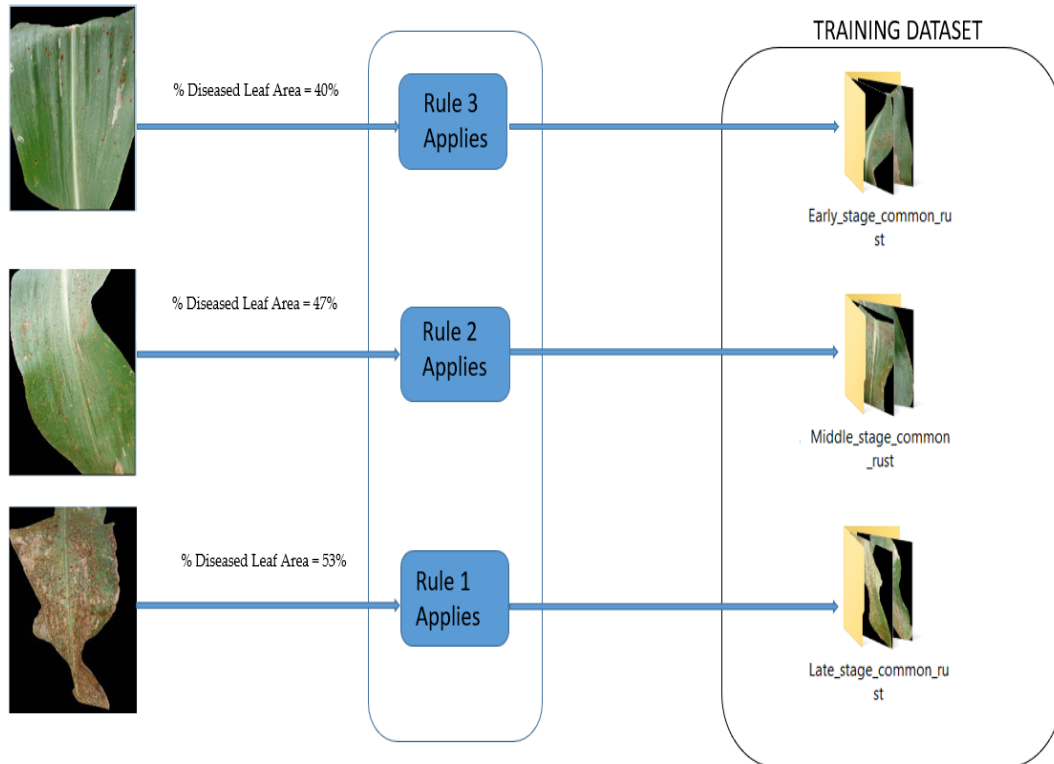
Design Rules for Common Rust Disease Severity Prediction:

Fuzzy decision Rule 1: *If %Diseased Leaf Area ≥50%, then, the image belongs to Late-stage training data set.*

Fuzzy decision Rule 2: *If 45% ≤ %Diseased Leaf Area < 50%, then, the image belongs to Middle stage training data set.*

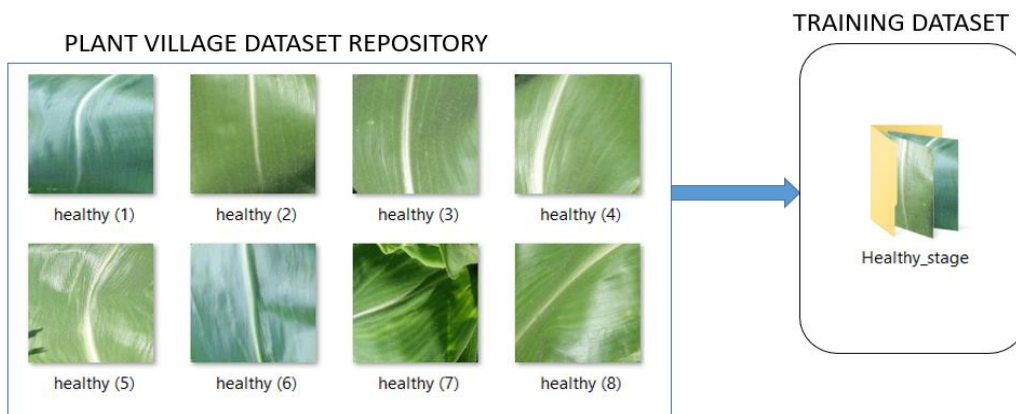
Fuzzy decision Rule 3: *If %Diseased Leaf Area <45%, then, the image belongs to Early-stage training data set.*

Figure 4.4 illustrates how common corn rust images are assigned to their severity classes using fuzzy decision rules. The same process was repeated for all common rust images until sufficient datasets were compiled to train the VGG-16 network. There were three severity classes for common corn rust and one in good health. The fourth healthy class was developed based on the PlantVillage data set using healthy corn images.



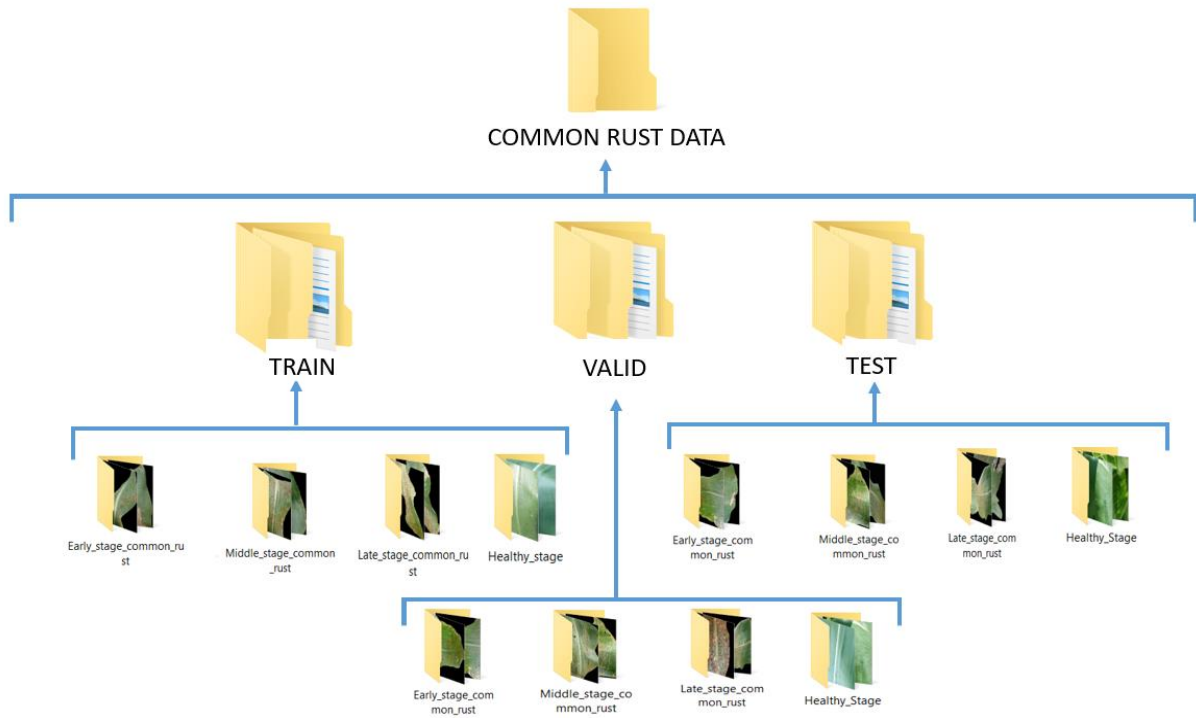
**Figure 4. 4:** Attributing common corn rust images to their severity classes using fuzzy decision rules.

Figure 4.5 shows a sample of healthy corn foliar images from the PlantVillage datasets that were used to train in the fourth class.



**Figure 4. 5:** An image sample of healthy corn that was used for training in the Healthy class.

By performing the fuzzy decision rules above to assign common rust images to their severity classes and forming a healthy class with healthy images of the Plant Village data set, eventually four classes of training data were achieved to train a VGG-16 for the prediction of maize Common Rust. Figure 4.6 shows the arrangement of the data sets that was followed to be able to train the VGG-16.



**Figure 4. 2:** The final arrangement of the training, validation, and test data sets for the prediction of the maize common rust disease severities by a fine-tuned VGG-16 network.

With different image backgrounds, the Plant Village data set we used was taken under nearly similar daylight conditions. Additional image data sets were gathered from camera Galaxy A-30 under normal daytime conditions. The images were taken during the summer in moderate weather in South Africa. The first series of images was taken between 5:00 and 6:00 a.m. in the rising sun. The second set of data was captured around noon, while the last was taken during sunset around 6:00 pm to 7:00 pm. During the tests, equal images were used for different brightness conditions.

#### ***4.2.1 Development of a Deep Learning Model by Fine-Tuning a VGG-16 Network and Theoretical Background***

Proposed by Simonyan and Zisserman of the Visual Geometry Group Lab of Oxford University in 2014, 2014), the VGG-16 network won the first and second prizes in the ImageNet Large Scale Visual Recognition contest for object localization and image classification categories, respectively. There were 200 classes under the object location category and 1,000 classes under the image classification category. The architecture for VGG-16 is depicted in Figure 4.7.

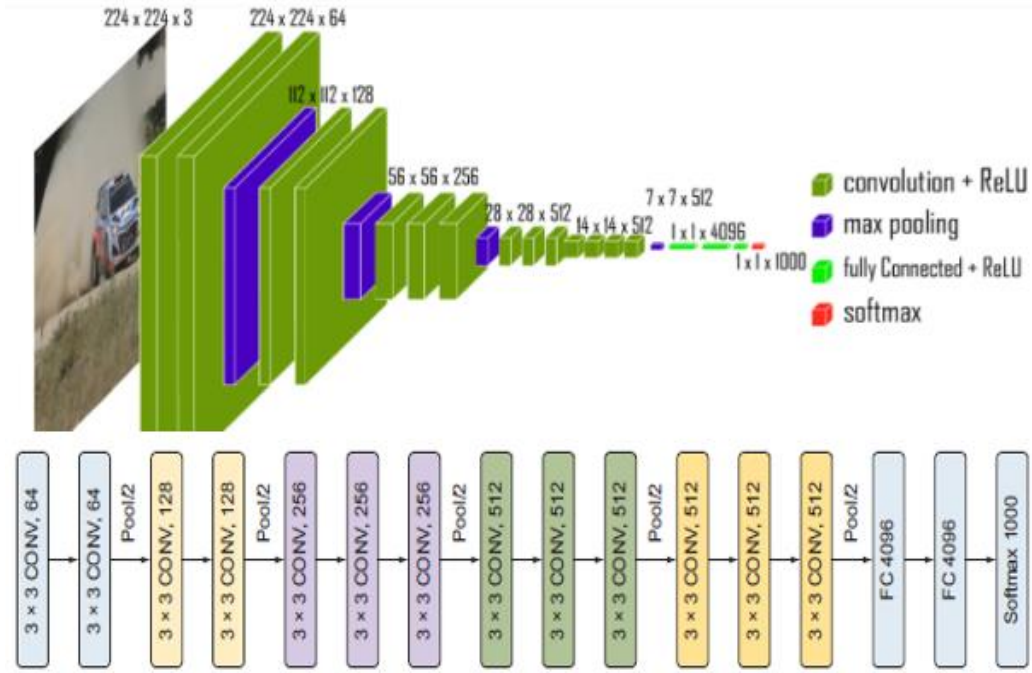


Figure 4. 3: The VGG-16 architecture.

Figure 4.7 indicates that the VGG-16 system has an input tensor of (224,224,3). This model processes the input image and produces a vector of 1000 predicted values (probability) as shown in equation (4.9).

$$\hat{y} = \begin{bmatrix} \hat{y}_0 \\ \hat{y}_1 \\ \vdots \\ \hat{y}_{999} \end{bmatrix}$$

(4.9)

The classification probability for a relevant class is determined by a Softmax function as illustrated in equation (4.10). Equation (4.10) shows the probability of prediction for the  $j$ th class from a sampling vector  $X$  and the weighting vector  $W$  with a Softmax function.

$$P(y = j|X) = \frac{e^{xW_j^T}}{\sum_{k=1}^K e^{xW_k^T}}$$

(4.10)

#### 4.2.2 Fine-tuning and Training a VGG-16 Network for Maize Leaf Disease Severity Prediction

There are four scenarios in which a pre-trained model can be fine-tuned. These scenarios are summarized in the following manner:

Scenario 1: The target datasets are small and very similar to the source datasets.

Scenario 2: The target datasets are broad and very similar to the source datasets.

Scenario 3: The target data set is small and very different than the source data set.

Scenario 4: The target dataset is vast and quite different from the source dataset.

The guidelines for the appropriate fine-tuning level to use in each of the scenarios are summarized in Table 4.4.

Table 4. 3: Summary for the fine-tuning of pretrained models.

Scenario	Size of the Target Data	Similarity of the New and Original Data Sets	The Approach Used
1	Small	Similar	The built-in network is used as a function extractor.
2	Large	Similar	Fine-tuning is done throughout the network.
3	Small	Very different	Fine tuning is performed based on earlier network activations.
4	Large	Very different	The tuning is done across the entire network.

The VGG-16 system developed for this study followed the procedure described in Table 4.4 in Scenario 3. This was accomplished by freezing all layers of a VGG-16 network, with the exception of the four upper layers. The resulting state-of-the-art model is summarized in Table 4.5.

Table 4. 4: Model summary for a fine-tuned VGG-16 network to predict maize common rust disease severities.

Layer Name	Type	Number of Filters	Number of Parameters
Input	Input layer	-	0
Block 1_Conv2D_1	Convolutional	64	1792
Block 1_Conv2D_2	Convolutional	64	36,928
Block 1_MaxPooling2D	Max Pooling	-	0
Block 2_Conv2D_1	Convolutional	128	73,856
Block 2_Conv2D_2	Convolutional	128	147,584
Block 2_MaxPooling2D	Max Pooling	-	0
Block 3_Conv2D_1	Convolutional	256	295,168
Block 3_Conv2D_2	Convolutional	256	590,080
Block 3_Conv2D_3	Convolutional	256	590,080
Block 3_MaxPooling2D	Max Pooling	-	0
Block 4_Conv2D_1	Convolutional	512	1,180,160
Block 4_Conv2D_2	Convolutional	512	2,359,808
Block 4_Conv2D_3	Convolutional	512	2,359,808
Block 4_MaxPooling2D	Max Pooling	-	0
Block 5_Conv2D_1	Convolutional	512	2,359,808
Block 5_Conv2D_2	Convolutional	512	2,359,808
Block 5_Conv2D_3	Convolutional	512	2,359,808
Block 5_MaxPooling2D	Max Pooling	-	0
Flatten	Layer	-	0
Dense (32 nodes)	Layer	-	802,848
Dropout (0.2)	Layer	-	0
Dense_1 (4 nodes)	Layer	-	132

A total of 15,517,668 parameters was contained by the model. Out of these model parameters, 802,980 were trainable, whereas 14,714,688 were not. As Table 4.5 illustrates, the fully connected layer consisted of a 32-

node dense layer and a ReLU activation function. In the first layer, 20% of the dropout was used. The "Softmax" activation function was used in the output layer of the fully connected layer. Each class was trained with 400 images and validated with 50 images. This made the model to be trained with 1600 images and 200 images for validation. Since a batch size of 32 was used, it means that the model took 1600/32 steps to train. The same applied to the validation steps as 200/32 validation steps were taken. The optimizer used was "Adam", it enabled the model to result in an outstanding validation accuracy of 95.63 %. At this point, the validation loss was 0.2. These performance metric results were achieved with a learning rate of 0.0001.

### 4.3 Results

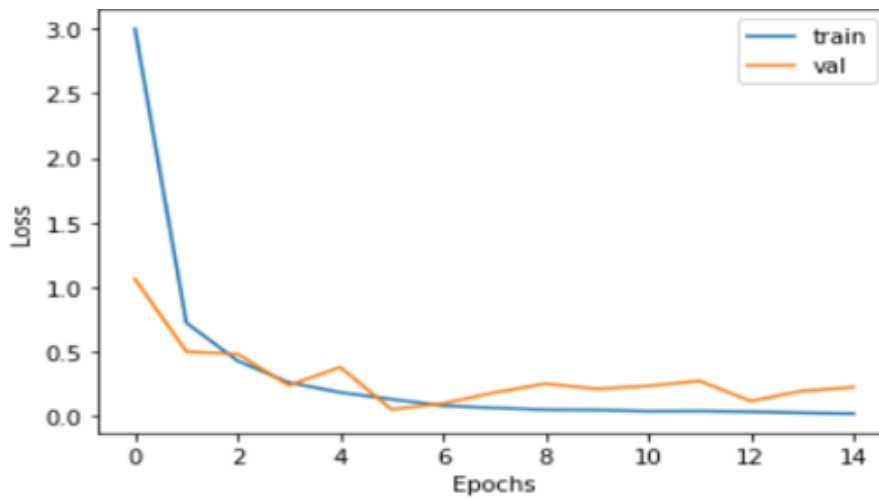
Table 4.6 summarizes the data sets, adjusted hyperparameters and performance parameters that define our VGG-16 network for Corn Rust severity prediction.

Table 4. 5: Summary of model hyper parameter tuning and performance metrics.

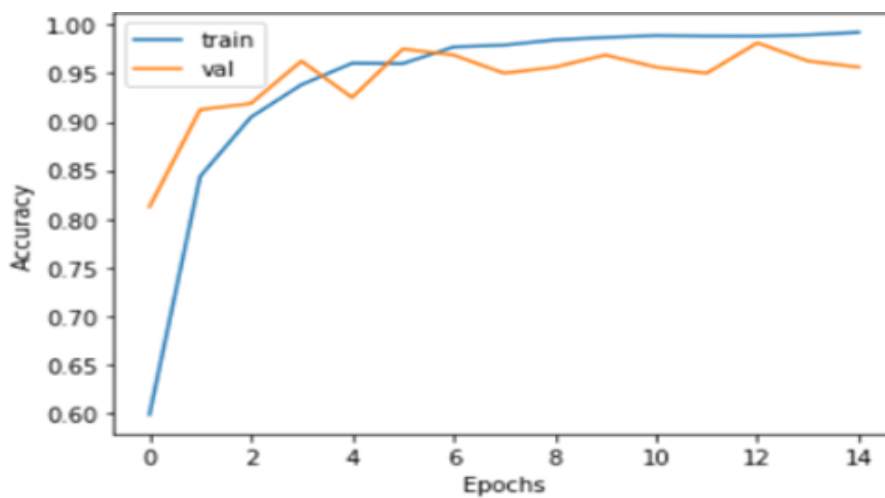
<b>Number of training images</b>	1600
<b>Number of validation images</b>	200
<b>Total images of the collected data set</b>	100
<b>Chosen batch size during training</b>	32
<b>Chosen optimizer and the learning rate</b>	Adam (Lr = 0.0001)
<b>Applied dropout in the fully connected layers</b>	20% of nodes in the first Dense layer of FC.
<b>Training Accuracy obtained by the model</b>	99.21%
<b>Validation Accuracy obtained by the model</b>	95.63%
<b>Test Accuracy obtained by the model</b>	89%
<b>Number of training epochs.</b>	15
<b>Recorded training loss after 15 epochs</b>	0.02
<b>Validation loss during training.</b>	0.2

Figures 4.8 and 4.9 show the graphs of the training parameters with respect to the validation parameters. Figure 4.8 shows the loss measurement plots, and Figure 4.9 shows the precision measurement plots. The primary cause of poor predictive performance in machine learning is either overfitting or underfitting of data (Elgendy, 2020). Underfitting means that the model is too simple and does not match the training data (Elgendy, 2020). Overfilling implies that the model is entirely complex, until it remembers the training data and does not generalize based on test/validation data it has never seen before (Elgendy, 2020). Consequently, Figures 4.8 and 4.9 indicate that the proposed model did not underfit nor overfit the training data. This is so because the two plots show that they are well generalised from the validation data. The model also achieved a high-test accuracy of 89% when tested on 100 images, with 25 images per class. It can also be seen in Figure 4.8 that the validation loss converges to 0.2 without oscillations. which was and reported that a learning rate of 0.0001 that we put into the Adam optimizer was ideal.



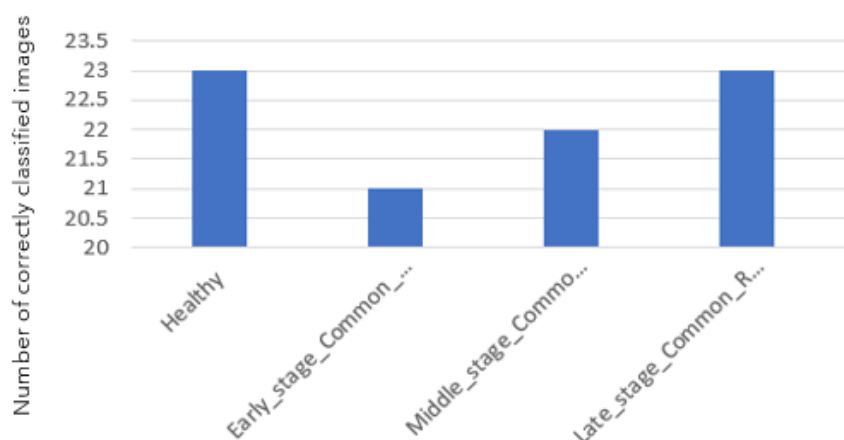


**Figure 4. 4:** Training loss against validation loss plots.



**Figure 4. 5:** Training accuracy against validation accuracy plots.

The test data sets were a set of camera images that were also allocated to their severity classes using the approach proposed in this study. A total of 100 images was used in the test experiments. Of the 100 images to be tested, each class included 25 images. Figure 4.10 shows the number of classifications that the VGG-16 managed to complete in each class.



**Figure 4. 6:** The number of correctly classified images in each class of the VGG-16 network.

Equation (4.11) shows how the testing accuracy of 89% was achieved by the VGG-16 network.

$$\text{Testing Accuracy} = \frac{\text{Number of correct classifications in each class}}{\text{Total number of testing images}} \times 100 \quad (4.11)$$

Using the information provided in Figure 4.10, the accuracy of the model testing was determined as follows:

$$\text{Testing Accuracy} = \frac{23 + 21 + 22 + 23}{100} \times 100 = 89\%$$

#### 4.4 Discussion and Conclusion

A new approach has been developed to assign images of common maize rust to their severity classes, guided by unclear decision rules and CNN approaches. The rules for the Fuzzy decision were developed using percentages of infected foliar zones. Before completing all the steps mentioned later, we first converted the colour images from a 3-dimensional array into a 2-dimensional array. RGB is a three-dimensional table which has three channels of colours Red, Green and Blue. Each of these channels is made up of 8-bit pixels that determine the colour intensity in different parts of the image. For instance, a green colour is a result of setting the intensities of the pixels in the same dimensional space to  $255G + 0B + 0R$ . The approach proposed in this study uses segmentation, which is actually time-consuming when conducted in the 3-dimensional space. The best way to achieve our objectives was to convert the colour images of a three-dimensional space into a two-dimensional space. Figure 4.11 shows the differences between two colour spaces that the image may take. We then segmented the greyscale images and calculated the percentages of diseased leaf areas in the maize common rust images. This allowed us to create fuzzy logic decision rules according to the guidelines of an experienced plant pathologist. For example, he mentioned that a dark common rust, which covers over 50% of the leaf area, is considered an advanced disease (Late stage). These fuzzy logic rules were then used to build a training dataset that was used to train the VGG-16 network. The image data severity classes of common corn rust developed in this way were used to train the fined-tuned VGG-16 which obtained a validation accuracy of 95.63% and a test accuracy of 89%.

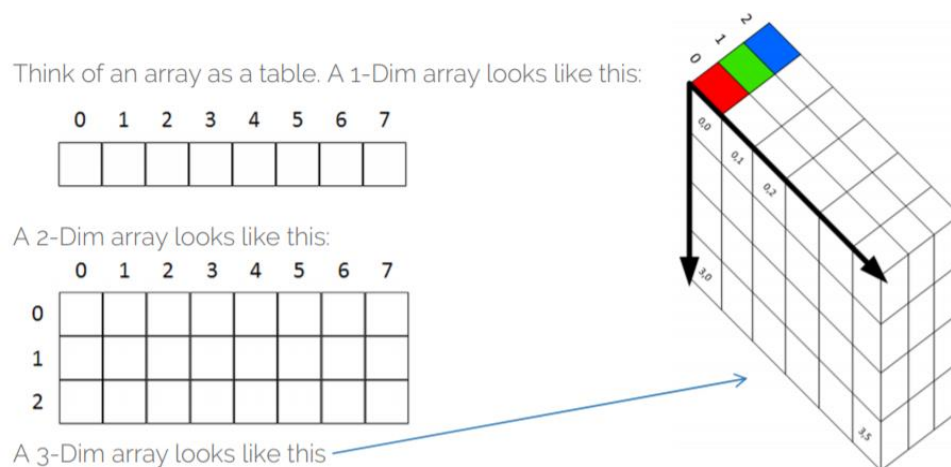


Figure 4. 7: A comparison of 2-dimensional array images with 3-dimensional array images.

The approach that was proposed by Wang *et al.* (2017) for predicting the severity of leaf disease using DL was biased to some extent. This is because the methods they used to attribute the common rust images were fully based on decisions that were made by human eye observations. The proposed approach is therefore unbiased as it uses the decisions of the computer-based fuzzy decision rules to attribute the common rust images to their severity classes for common corn rust disease. The work published by Sibiya and Sumbwanyambe (2019b) resulted in the development of training datasets using fuzzy logic and a threshold. As far as we know, this is the first report on the approach to predict the severity of common maize rust. Broadly translated, our findings indicate the proposed approach resulted in a model that had higher validation and testing accuracies in the prediction of maize common rust disease severity. Further investigation is advisable in order to validate various findings, which can be derived from the methods used in this study provided other types of diseases of corn leaves are considered.

# CHAPTER 5: Use of Convolutional Neural Networks for severity prediction of Northern Corn Leaf Blight based on lesion colour and sporulation

This chapter is based on a paper submitted by:

Sibiya, M. and Sumbwanyambe, M., 2021. Northern Corn Leaf Blight Severity Predictions based on Colour and Sporulation of the Lesions by using Convolutional Neural Networks: A Maize Leaf Diseases Odyssey, *Agri Engineering 2021*

Link: <https://www.mdpi.com/journal/agriengineering>

Status: Submitted

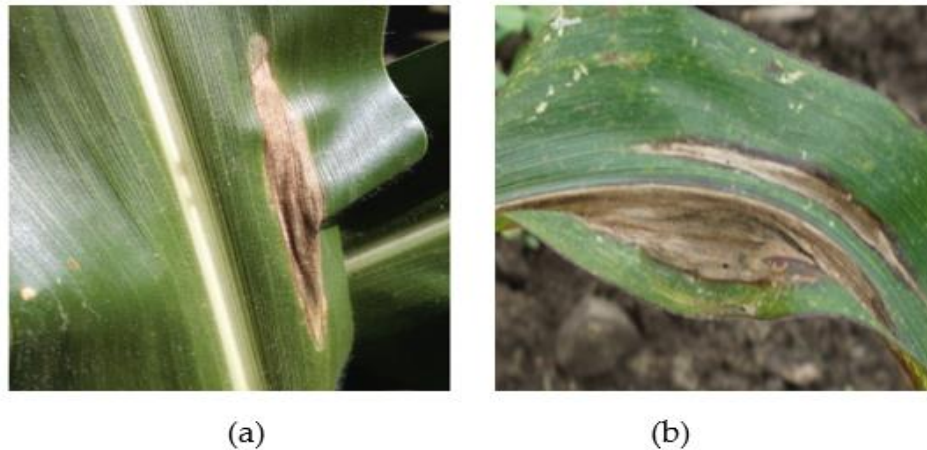
## Abstract

This study aims to introduce a new method of predicting the severities of the Northern Corn Leaf Blight (NCLB) by classifying images of this disease into two categories of “Early Stage” and “Late Stage” using Convolutional Neural Networks (CNN) approach. In the “Early Stage”, lesions of the NCLB initially appear smaller, more yellow, and/or not produce spores. In the “Late Stage”, lesions eventually turn tan coloured and may contain dark areas of fungal sporulation. In this study, a CNN model trained on image data sets with later mentioned lesion features of both stages was used to make predictions and achieved a high validation accuracy of 100% for experiment 1(model performance test), and 99% equivalent to 100% for experiment 2 (model robustness test). This is a promising tool for plant pathology researchers interested in understanding the severities of the NCLB samples collected. This may also be a great tool for computer vision of Agri-tech robots intended to collect information about the NCLB disease occurring in large maize fields.

## 5.1 Introduction and Related Works

This work is an extension of published works as explained in the discussion section. The severity of the Northern Corn Leaf Blight (NCLB) increases as the lesions expand and grow larger, resulting in a reduced photosynthetic surface area, which reduces grain filling and yield (University of Minnesota, 2021; Jackson-Ziems, 2016). The development of the disease at the beginning of the season has a greater potential for impact on yield due to the longer period and the greater leaf area affected, which reduces photosynthesis and subsequently the production of carbohydrates. The NCLB is caused by the fungus *Exserohilum turcicum*. Its development is encouraged by cool to moderate temperatures and high relative humidity. NCLB can be identified by relatively large cigar-shaped grey or greenish elliptic lesions that can develop on leaves, shells or sheath (Jackson-Ziems, 2016). NCLB lesions can range from 1 inch to over 6 inches long and are oriented parallel to the foliar veins. After 7 to 12 days of infection, the lesions develop and the fungus responsible for the NCLB can produce large amounts of spores (sporulation) on the surface of the lesions, causing them to

look dark or dusty. As the lesions mature, their colour may change to tan and the production of fungal spores may become visible in the middle of the lesions, leading to a darker, dustier appearance (see Figure 5.1). This manifests the late stage of the disease. However, at the early stages of this disease, lesions may appear smaller, more yellow, and/or not produce spores (Jackson-Ziems, 2016). Figure 5.2 shows the NCLB lesions at its early stage. Hybrids that are resistant to NCLB may have the same lesion features as those of NCLB at the early stage (Jackson-Ziems, 2016). The aim of this study is to introduce a novel method of predicting the severities of the NCLB by classifying the images of this disease into two classes of “Early-Stage” and “Late-Stage” using Convolutional Neural Networks (CNN).



**Figure 5. 1:** Field images of the NCLB in the Late Stage: (a) Large amounts of spores on the surface of lesions, with a dark or dusty appearance because of the fungus causing the NCLB; (b) As the lesions mature, their colour change to tan and the production of fungal spores has become visible in the middle of the lesions.



**Figure 5. 2:** Field image of the NCLB on the Early Stage with small lesions, more yellow, and/or not produce spores.

### 5.1.1 Management of Northern Corn Leaf Blight and applications of Deep Learning for the detection of leaf diseases.

The NCLB due to *Exserohilum turcicum* is a destructive disease of maize. In the literature, there is strong evidence that many researchers developed different strategies for managing the NCLB. For instance, in a study by Ding *et al.* (2015), the host resistance management strategy was used to minimize the detrimental effects

of NCLB on maize productivity. In their study, they evaluated a diverse maize population that comprised of 999 inbred lines across various environments for resistance to NCLB. The genomic regions associated with NCLB resistance in maize were identified by using a genome-wide association analysis that was conducted using 56,110 single-nucleotide polymorphism markers. The alleles (gene) significantly associated with NCLB resistance were identified by Single-marker, haplotype-based associations, and Anderson-Darling tests (Ding *et al.*, 2015). Other researchers controlled the NCLB by planting hybrids with good NCLB disease ratings (Pataky, 1985; Abera *et al.*, 2016; Technow *et al.*, 2013). Hybrid resistance can assist reduce disease severity by restricting the number or measure of NCLB lesions that develop, extending the incubation period (number of days between infection and lesion advancement) (University of Minnesota, 2021). In the literature, there is evidence of small to large scale studies that were conducted to manage NCLB disease by using NCLB resistant hybrids (Pataky, 1985; Abera *et al.*, 2016; Technow *et al.*, 2013). Other researchers such as Carpane *et al.* (2020), used fungicides to slow the disease spread and reduced the overall severity of the Northern CLB. This study also contributes toward the management of the NCLB as it is aimed at introducing a novel method of predicting NCLB severities by using the colour and sporulation of the lesions on images that were used to train a CNN model. This is a promising tool for plant pathology researchers interested in understanding the severities of the NCLB samples collected. This may also be a great tool for computer vision of Agri-tech robots intended to collect information about the NCLB disease occurring in large maize fields. As shown in Table 5.1, there is a large body of literature on the use of DL for the detection of plant leaf diseases.

Table 5. 1: Tabulated summary of the studies that investigated the use of DL for plant leaf disease detection and the related DL models used.

Deep Learning Architecture	Reference
Google Net	(Barbedo, 2018).
	(Huang <i>et al.</i> , 2019).
U-Net	
3D Deep Convolutional Neural Network (DCNN)	(Nagasubramanian <i>et al.</i> , 2019).
ImageNet	(Brahimi <i>et al.</i> , 2017).
Dense Net	(Ale <i>et al.</i> , 2019).
9-Layer Deep CNN	(Geetharamani and Pandian, 2019)
Deep Siamese convolutional network	(Goncharov <i>et al.</i> , 2018).
CNN API written in Python	(Militante <i>et al.</i> , 2019).
Alex Net and Google Net	(Mohanty <i>et al.</i> , 2016)
Alex Net, VGG16, and VGG19	(TÜRKOĞLU and Hanbay, 2019).
Alex Net, Alex Net OWTBn, Google Net, Over feta and VGG	(Ferentinos, 2018).
CNN model resembling the standard Le Net architecture	(Baranwal <i>et al.</i> , 2019).
Alex Net and VGG-16	(Rangarajan <i>et al.</i> , 2018).
VGG 16, Inception V4, Res Net with 50, 101 and 152 layers and Dense Nets with 121 layers	(Too <i>et al.</i> , 2019).
Google Net	(Barbedo, 2019)

## 5.2 Materials and methods

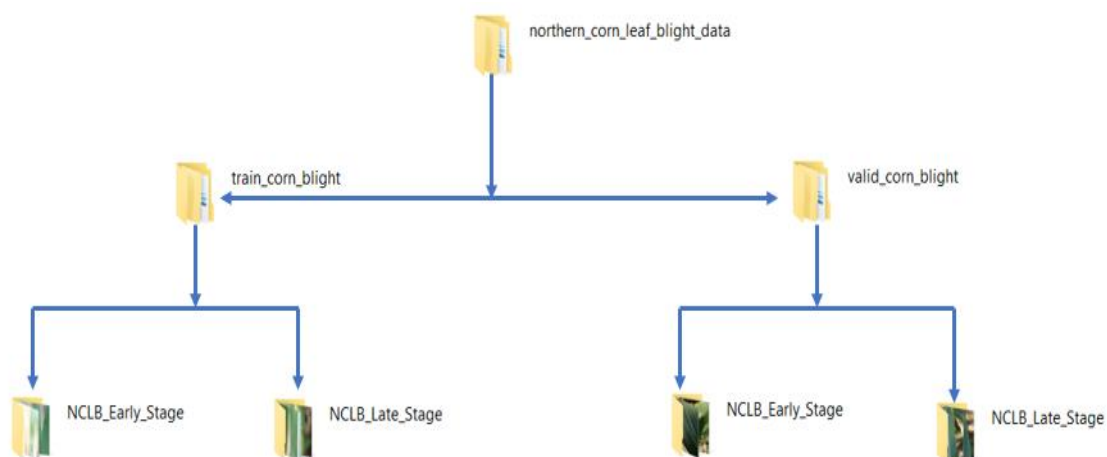
### 5.2.1 Data Collection and Pre-processing

The NCLB images were collected from 5 different maize fields located in the outskirts of Bloemfontein, South Africa. The maize crops from which the images had been collected by means of a camera were marked with tags as shown in Figure 5.3.



**Figure 5. 3:** The tags were used to mark the maize crops from which the NCLB images were taken.

In total, 3500 images were collected from the 5 maize fields. The collected images were a mix of images with features of both the NCLB Early and Late stages. Using the information in the literature about NCLB severity features (Jackson-Ziems, 2016), the images were classified into two classes of “NCLB Early-Stage Class” and “NCLB Late-Stage Class”. The literature on the identification of NCLB Early and Late stages was verified with the professionals from Plant Pathology Department at the University of Free State, South Africa. Out of the collected 3500 images, 1600 images were used for training the CNN model with each class trained on 800 images. Figure 5.4 shows the arrangement of the training and validation data sets for the CNN model intended to make predictions of the NCLB severities. Both the training and validation data sets were made of two classes. As it was mentioned earlier that each class of the training datasets contained 800 images, a different case applied to the validation data sets where each class contained 180 images.



**Figure 5. 4:** Arrangement of the training and validation data sets for the CNN model intended to make severity predictions of the NCLB.



### 5.2.2 Materials

A list of materials that were used in this study is summarized in Table 5.2.

Table 5. 2: A tabulated summary of the materials that were used in the study.

Hardware	Software	Dataset
ASUS TUF Laptop with NVIDIA GeForce GTX 1650 4GB Graphics.	Anaconda package.	Custom image data set for NCLB disease.
Samsung A-30 16 Mega Pixels rear camera.	Jupyter notebook IDE.	
	Python-Keras library.	
	Python-Tensorflow library.	

**Hardware:** ASUS TUF Laptop with NVIDIA GeForce GTX 1650 4GB Graphics was used as the training of a CNN with images may be time consuming if done with a PC or may even time out without completing the training process. The Samsung A-30 16 Mega Pixels rear camera was used for taking the NCLB images that were used in this study from 5 different maize fields.

**Software:** An Anaconda package which is recommended by most data-scientists were downloaded and installed into the GPU machine. The advantage of this package is that it comes with most useful libraries such as Matplotlib for plots and viewing of images. Other important libraries that come with this package are NumPy, Scikit-learn and Seaborn to mention a few. Tensorflow and Keras are Python libraries that work in coordination with each other and were used for computation of the CNN. The Jupyter notebook IDE (Integrated Development Environment) was used to its advantage of the code management and ability to make the plots within the IDE.

**Datasets:** The collected NCLB data sets were used for training in the CNN model that is proposed in this study for the prediction of the NCLB severities.

### 5.2.3 CNN Model Architecture

Figure 5.5 shows the deep CNN for that was used in this study for the prediction of the NCLB severities. Stage 1 that accepts the input images is linked to stage 2 that has the 2 classes of the predicted NCLB severities. In this model several regularization methods such as the batch normalization and drop out were used in different layers of the network to overcome overfitting. Also, overfitting was overcome using callbacks such as the early stopping, checkpoint and reduction of the learning rate. To put the model architecture on test, two experiments were conducted. The first experiment was meant for testing the model performance on a set image data set while the second experiment was for testing the model's robustness.

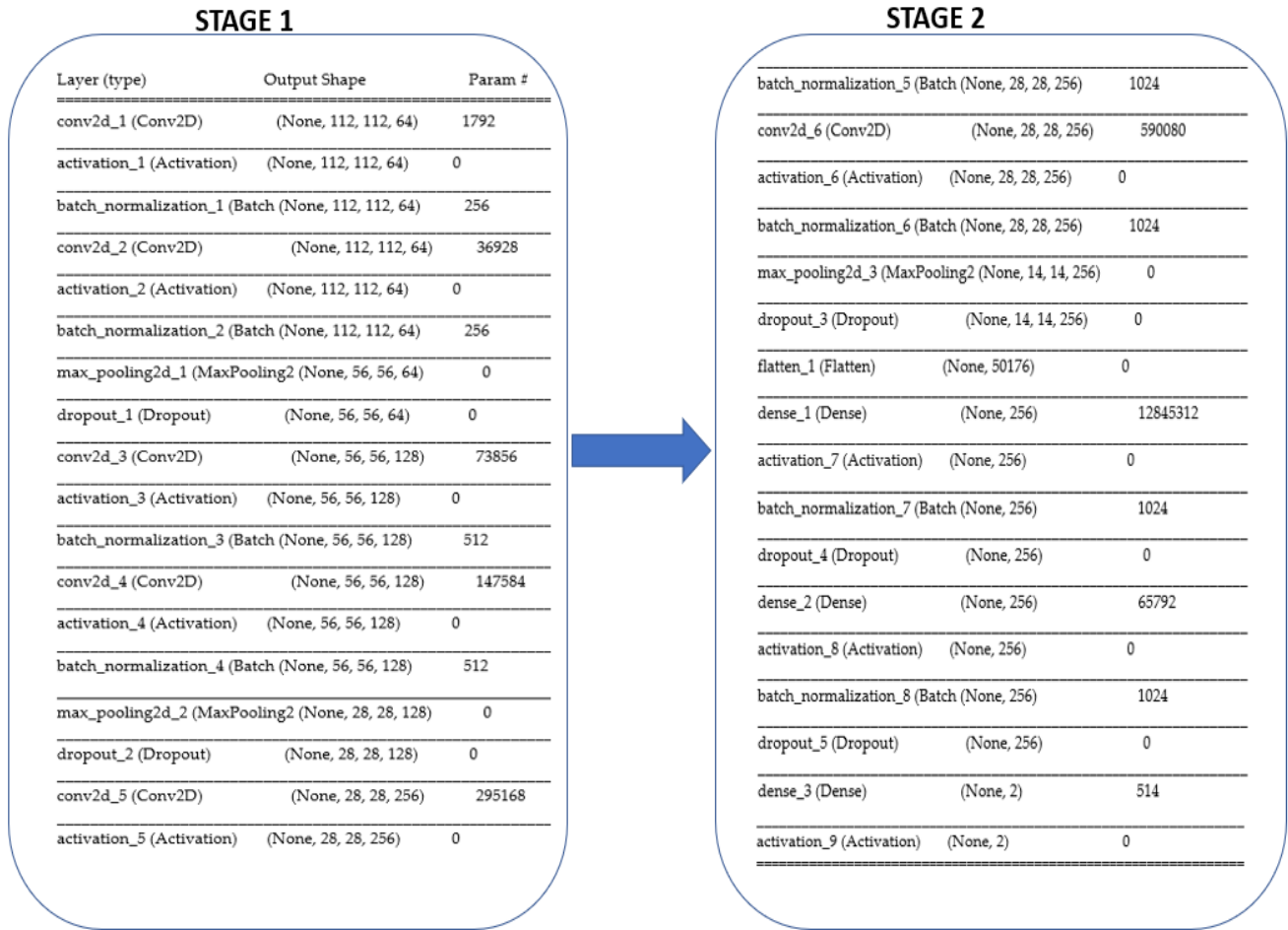


Figure 5.5: CNN model architecture for predicting the NCLB severities.

### 5.2.4 Experiment 1: Model Performance

This experiment was conducted by training the CNN model architecture shown in Figure 5.5 with 1600 images of NCLB. Since the model was developed to make its predictions between two classes, each class was trained with 800 images. The validation/testing data sets contained 360 NCLB images with 180 images belonging to each of the two classes. To test the model's performance, 4 key performance indicators (KPIs) were monitored. These were Accuracy, Precision, Recall and F1-Score. The resulting performances for this model are reported in the results section, however, it can be seen in the confusion matrix report in Figure 5.7 that the model achieved 100 % of accuracy. A summary of the model's important parameters for this experiment are tabulated in Table 5.3.

Table 5. 3: Parameter settings for model training and validation

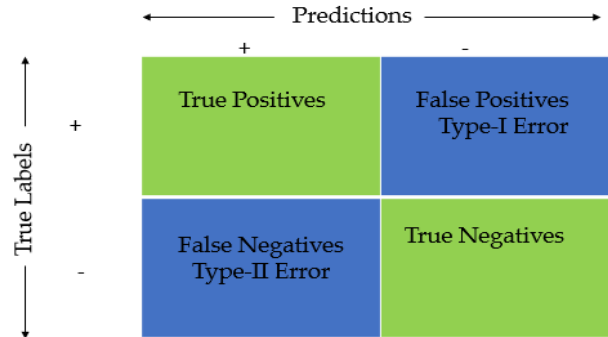
<b>Training Epochs</b>	10
<b>Training images NCLB Early Stage-Class</b>	800
<b>Training images NCLB Late Stage-Class</b>	800
<b>Validating/Testing images NCLB Early Stage-Class</b>	180
<b>Validating/Testing images NCLB Late Stage-Class</b>	180
<b>Learning rate (Lr)</b>	0.001

### ***5.2.5 Experiment 2: Model Robustness test***

Because the NCLB disease is recognised by the cigar like shaped lesions regardless of their severity stages, we decided to test the robustness of the model. This was achieved by taking two images that were supposed to be in the NCLB Late-Stage Class and used them to substitute the other two images in the NCLB Early-Stage class. Then the NCLB Late-Stage Class was balanced with other two correct images. This means, the NCLB Late-Stage class contained 180 correct images while NCLB Early-Stage Class contained 178 correct images and two images that supposedly belonged to the NCLB Late-Stage Class. The model was deemed to be robust because it misclassified the two wrong images that were put in the NCLB Early-Stage. This can be observed by taking a closer look at the confusion matrix report in the results section. Also, it can be seen in the results section, that the model did not misclassify the images when the correct images were used for validation in their correct classes. The fact that the model misclassified the two images resulting in 99% of accuracy means that the model achieved 100% of accuracy in the robustness test because the misclassification was expected to happen to see the model's ability to understand the features of the images that it was trained on.

### 5.3 Results

Before the experiment results are unpacked in detail, the model’s key performance indicators are explained. As mentioned before, the model’s key performance indicators that were monitored in this study are Accuracy, Precision, F1-Score and Recall. To understand these key performance indicators, the confusion matrix in Figure 5.6 is used to explain each of them.



**Figure 5. 6:** The general confusion matrix used to understand model performance key indicators.

**Accuracy:** Accuracy in classification problems is the number of correct predictions made by the model divided by the total number of predictions. Equation (5.1) shows the formula for calculating the accuracy based on the confusion matrix in Figure 5.6.

$$Accuracy = \frac{True\ Positives}{(True\ Positives + False\ Negatives + False\ Positives + True\ Negatives)} * 100 \tag{5.1}$$

Accuracy is useful when the target classes are well balanced, so, in this this study the balanced data sets were used, hence it was important to also focus on the accuracy.

**Recall:** This is the ability of the model to find all the relevant cases within a data set. The precise definition of recall is the number of true positives divided by the number of true positives plus false negatives. Equation (5.2) shows the formula for calculating the recall based on the confusion matrix in Figure 5.6.

$$Recall = \frac{True\ Positives}{(True\ Positives + False\ Negatives)} \tag{5.2}$$

**Precision:** This is the ability of the classification model to identify only the relevant data points. Precision is defined as the number of true positives divided by the number of true positives plus false positives. Equation (5.3) shows the formula for calculating the precision based on the confusion matrix in Figure 5.6.

$$Precision = \frac{True\ Positives}{(True\ Positives + False\ Positives)} \tag{5.3}$$

**F1-Score:** In cases where we want to find an optimal blend of precision and recall, we can combine the two metrics using what is called the F1 Score. All in all, the F1 Score is the harmonic mean of precision and recall taking both metrics into account as shown in equation (5.4).

$$F1 = 2 * \frac{Precision * Recall}{(Precision + Recall)} \tag{5.4}$$

### 5.3.1 Experiment 1: Model Performance results

Figure 5.7 shows the comparison plots of a model described in experiment 1. The plots show that the validation accuracies are higher when compared with training accuracies. This is a good sign that the model was generalizing well to make the predictions and it did not overfit the training data. An overfitting model would result to higher training accuracies and lower validation accuracies that would tend to drift away from the training accuracies.

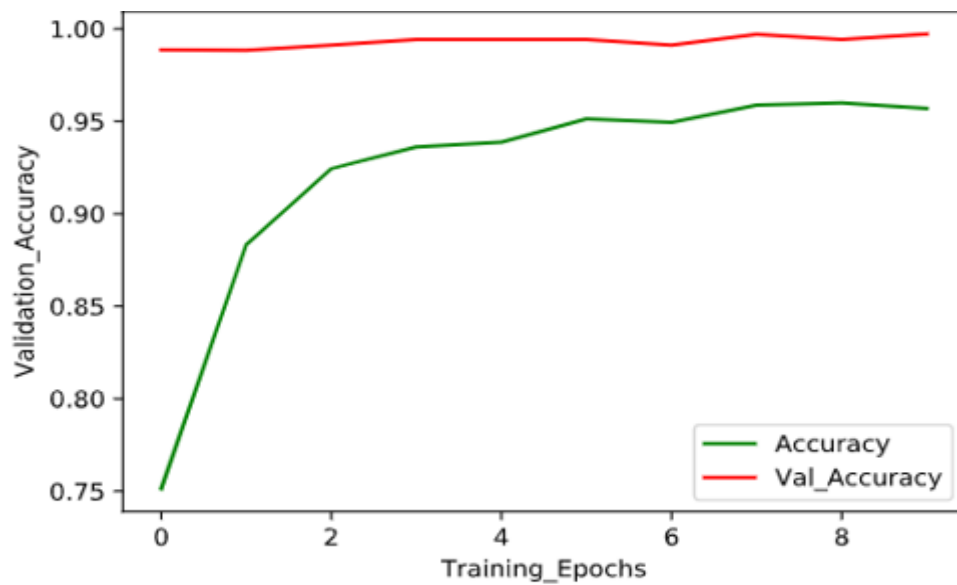
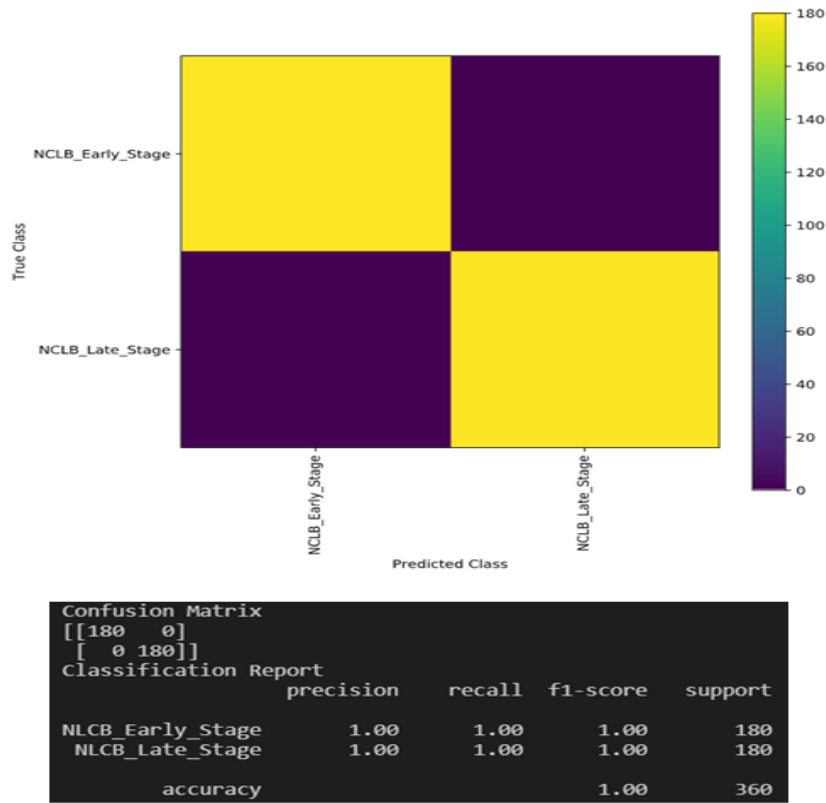


Figure 5. 7: Comparison plots of validation and testing accuracies of a model described in experiment 1.

Figure 5.8 shows the confusion matrix of the model predictions and the generated classification report of the performance metrics.



**Figure 5. 8:** Confusion matrix and metrics classification report of the model described in experiment 1.

Taking a closer look at the classification report results in Figure 5.8, it can be seen that the model did not misclassify the NCLB classes. Using the formulas in equations (5.1) to (5.4), the metrics generated in the classification report are calculated as follows:

$$Accuracy = \frac{180}{(180 + 0 + 0 + 0)} = 1.0 \tag{5.5}$$

$$Recall = \frac{180}{(180 + 0)} = 1.0 \tag{5.6}$$

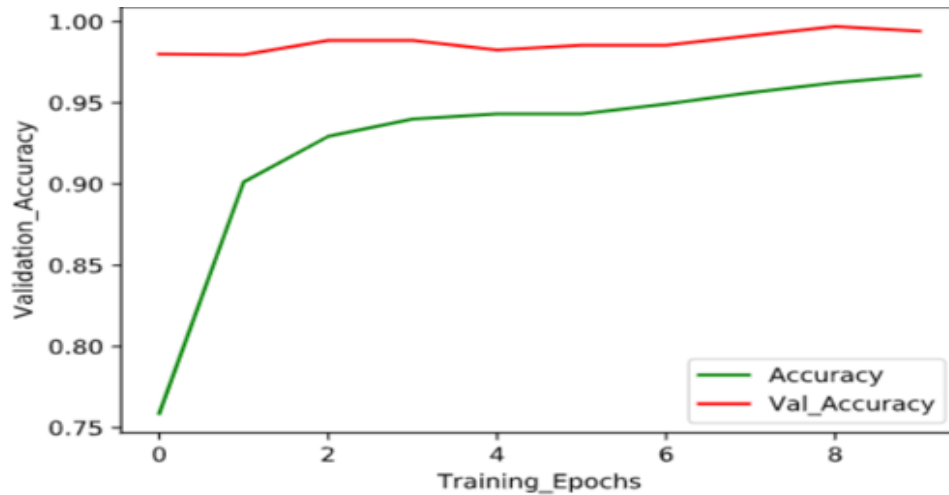
$$Precision = \frac{180}{(180 + 0)} = 1.0 \tag{5.7}$$

$$F1 = 2 * \frac{1.0 * 1.0}{(1.0 + 1.0)} = 1.0 \tag{5.8}$$

### 5.3.2 Experiment 2: Model Robustness test results

As explained in the methodology section, the purpose of this experiment was to test the model’s robustness. This was achieved by putting two images of NCLB in the class they did not belong to while ensuring the balanced class of 180 images. If the model was robust, it was expected to misclassify the two images and give an accuracy of 99 % in the model’s realistic classification point of view. However, this model’s 99%

classification accuracy was equivalent to 100 % of accuracy in the model’s robustness test point of view. Figure 5.9 shows the comparison of the training and validation accuracies in the model’s realistic performance point of view. So, with the model being able to misclassify the two NCLB images it was deemed to be robust in prediction of the NCLB severities.



**Figure 5. 9:** Comparison of the training and validation accuracies in the model’s realistic performance point of view.

The model’s ability to misclassify the two images deemed the model to robust in making the NCLB severity predictions. Since in this experiment the purpose was to test the model’s robustness, the performance metric that is closely looked is the Accuracy. Accuracy of 99% was expected from a robust model. However, this accuracy measure was not seen as a drop in the model’s performance in accuracy because one of the NCLB classes was validated with 178 correct images and 2 images that did not belong to that class but supposedly belonged to the second class. When the model was validated under these validation data set conditions, it “misclassified” the two images. In the latter sentence, the word “misclassified” is written in quotes because the model did not really misclassify these two images, but it assigned them to the class to which they were supposed to belong based on their image features. In this sense, the model’s 99% of accuracy is 100% of accuracy at the model’s robustness test point of view. So, the robustness accuracy test in this regard is calculated as shown in equation (5.9).

$$Accuracy (Robustness test) = \frac{178}{(178 + 0 + 2 + 0)} = 0.99 \equiv 1.0 \tag{5.9}$$

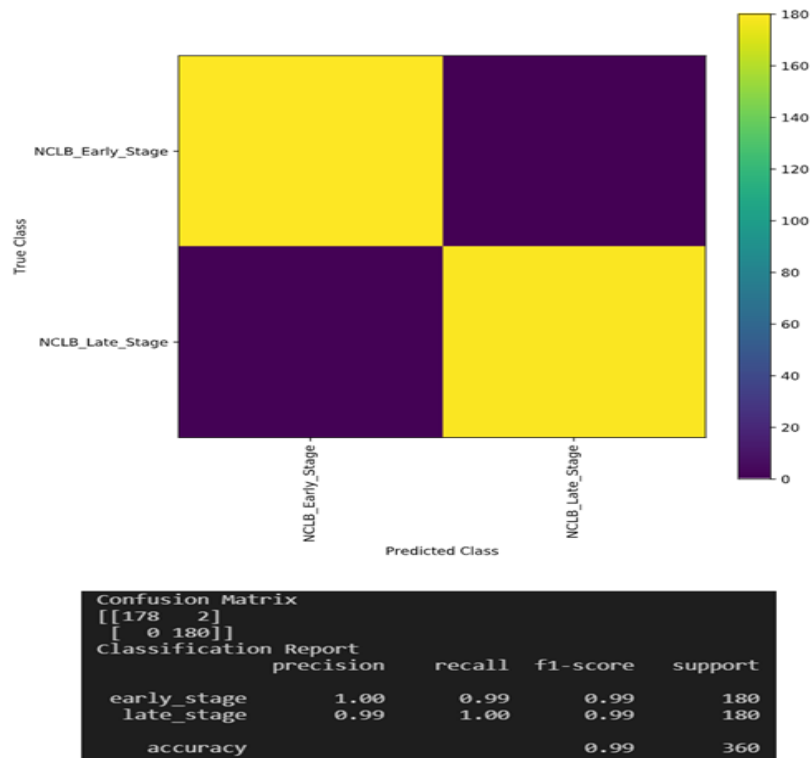


Figure 5. 10: Confusion matrix and metrics classification report of the model described in experiment 2.

## 5.4 Discussion and Conclusion

Most studies on the plant leaf disease classification used CNNs to classify plant leaf diseases of different plant species or same plant species of different disease types (Ferentinos, 2018; Baranwal *et al.*, 2019; Rangarajan *et al.*, 2018; Too *et al.*, 2017; Barbedo, 2019). In this study, CNN was used to classify foliar disease severities of the same plant species and same disease type. Here, we studied the maize, and the disease type was the NCLB. This novel approach of predicting the severities of the NCLB using a CNN was inspired by the fact that NCLB affects the KwaZulu-Natal regions, South Africa, affecting the country’s seasonal maize yield (Craven, 2020). The aim of this study is to introduce a novel method that can be used by plant pathology researchers who would want to predict the NCLB severities hence replacing tedious laboratory methods. Also, the CNN model proposed here for NCLB severity predictions is a promising tool for computer vision of Agri-tech robots intended to collect information about the NCLB disease occurring in large maize fields. The developed deep learning model is inspired by state-of-the-art VGG-16 model as its structure resembles the VGG-16 network (Simonyan and Zisserman., 2014). The model was compensated for overfitting by using regularization methods such as batch normalization, early stopping, learning rate decay and drop out. As a result of these regularization methods, the model achieved high classification accuracy of 100% and passed the robustness test explained in experiment 2. This work is an extension of our previously published works on a research project of maize leaf diseases (Sibiya and Sumbwanyambe, 2019a; Sibiya and Sumbwanyambe, 2021). Initially the aim was to classify maize leaf diseases using a GUI framework (Sibiya and Sumbwanyambe, 2019a). In this work, the CNN was able to classify various maize leaf diseases and achieved



appreciable validation accuracy of 92.85% (Sibiya and Sumbwanyambe, 2019). In our previous works we used CNN, Fuzzy decision rules, and thresholding to predict the severity of maize Common Rust (Sibiya and Sumbwanyambe, 2019). Out of the maize leaf disease that were classified, we then selected the Common Rust and Northern Corn Leaf Blight to be the disease that their severities had to be predicted as they mostly affect the maize yield of Southern Africa. The prediction of maize Common Rust was introduced in a published study where CNN was used with fuzzy decision rules and thresholding to make the predictions of this disease (Sibiya and Sumbwanyambe, 2019). In this study, the novel approach to predicting the severities of the Northern Corn Leaf is introduced as part of a continuing research on maize leaf diseases. We conclude that the CNN model can be used to make severity predictions of the NCLB disease. The training data sets for the severity classes were compiled as guided by the image features of each class (Jackson-Ziems, 2016). The image features that were used to assign each NCLB image to their appropriate training and validation class were confirmed by the professional pathologist at the University of Free State, South Africa. The model proposed in this study is a promising tool for Plant Pathology researchers interested in understanding the severities of the NCLB samples collected. This may also be a great tool for computer vision of Agri-tech robots intended to collect information about the NCLB disease occurring in large maize fields.

## **CHAPTER 6: Evaluation of Objectives and Conclusion**

In general, this chapter provides the framework of the entire thesis, tying up the key results in order to reflect on the overall contribution of CNNs on classification and severity prediction of maize leaf diseases. This chapter begins by evaluating research objectives and summarize key research contributions made in this work. It then presents further research recommendations and provides concluding remarks.

### **6.1 Evaluating objectives and Key Research Contribution**

The aim of the research was to introduce Deep Learning CNN approaches for the classification and severity prediction of maize leaf diseases. The remarkable results that emerged from the study were of the CNN models that were developed for classifying various maize leaf diseases and severity prediction of maize leaf diseases. The research results are critically evaluated by revisiting each study objective. The key areas where this study made an original contribution are highlighted by also revisiting each study objective.

#### **Objective 1: To create and test a CNN model that can classify various types of maize leaf diseases**

The study offered valuable insights into the development of a CNN model to classify various types of maize leaf diseases using a GUI platform. The CNN model was developed to make predictions of three different maize leaf diseases.

##### **Contributions:**

1. Facilitated principles of a CNN model in order to develop a network for the classification of maize leaf diseases.

#### **Objective 2: To set up and test a CNN model that can predict the severities of a maize leaf disease known as the maize CR**

The study demonstrated a novel approach of using CNN to predict the severities of maize CR disease. The CNN model was trained on images of maize CR disease with different severities of the disease in order to make severity predictions. The testing of the model was done using separate testing data sets.

##### **Contributions:**

1. Development of a Hybrid model using FL rules and a CNN in order to classify and predict maize CR disease.

### **Objective 3: To build and test a CNN model that can predict the severities of a maize leaf disease known as the NCLB by analysing lesion colour and sporulation patterns**

The study demonstrated a novel approach of using CNN to predict the severities of NCLB disease. The CNN model was trained on images of the NCLB disease with different severities of the disease in order to make severity predictions. The model was tested on separate testing data sets.

#### **Contributions:**

1. A novel approach to developing the training data sets for CNNs based on NCLB lesion colour and sporulation.
2. Introduction of a CNN approach for predicting the severities of the NCLB disease.

## **6.2 Recommendations and Suggestions for Future Research**

Based on the findings of this thesis, further studies, which take the following recommendations into account, will need to be undertaken.

1. The results of this study are only limited to CNN algorithms, therefore ML algorithms that only work on numerical data were never tested for similar approaches introduced in this study. Therefore, it would be interesting if plant pathologists would intervene and propose suggestions on the availability of numerical data for plant leaf diseases.
2. It is recommended that research be conducted on how agricultural vehicles and robots may be equipped with the CNN models proposed in this study.

## **6.3 Concluding Remarks**

The results presented in this thesis clearly demonstrate that CNNs are a promising tool not only for the classification of various maize leaf diseases, but for various plant species leaf diseases at large (Sibiya and Sumbwanyambe, 2019a). Initially, it was thought that severity prediction of plant leaf diseases using CNNs was impossible until this study proved it possible by using Fuzzy decision rules and thresholding to manipulate training data sets (Sibiya and Sumbwanyambe, 2021). The idea of using Fuzzy logic and thresholding to develop CR training data sets for severity prediction with a CNN was inspired by the works of (Sibiya and Sumbwanyambe, 2019b). It is also demonstrated in this study that the image data sets of plant leaf lesions and sporulation could be manipulated to develop training datasets for a CNN model capable of making disease severity prediction of that plant species.

## References

- Caruana, R. and Niculescu-Mizil, A., 2006, June. An empirical comparison of supervised learning algorithms. In *Proceedings of the 23rd international conference on Machine learning* (pp. 161-168).
- Zheng, Q., Huang, W., Cui, X., Shi, Y. and Liu, L., 2018. New spectral index for detecting wheat yellow rust using Sentinel-2 multispectral imagery. *Sensors*, 18 (3), p. 868.
- Fang, Y. and Ramasamy, R. P. ,2015. Current and prospective methods for plant disease detection, *Biosensors*, 5 (3), pp. 537–561.
- Chen, H., Chen, A., Xu, L., Xie, H., Qiao, H., Lin, Q. and Cai, K., 2020. A deep learning CNN architecture applied in smart near-infrared analysis of water pollution for agricultural irrigation resources. *Agricultural Water Management*, 240, p. 106303.
- Sibiya, M. and Sumbwanyambe, M., 2019a. A computational procedure for the recognition and classification of maize leaf diseases out of healthy leaves using convolutional neural networks. *AgriEngineering*, 1 (1), pp. 119-131.
- Sibiya, M. and Sumbwanyambe, M., 2021. Automatic Fuzzy Logic-Based Maize Common Rust Disease Severity Predictions with Thresholding and Deep Learning. *Pathogens*, 10 (2), p. 131.
- Nikhitha, M., Sri, S.R. and Maheswari, B.U., 2019, June. Fruit Recognition and Grade of Disease Detection using Inception V3 Model. In *2019 3rd International conference on Electronics, Communication and Aerospace Technology (ICECA)* (pp. 1040-1043). IEEE.
- Nasiri, A., Taheri-Garavand, A. and Zhang, Y. D. ,2019. Image-based deep learning automated sorting of date fruit, *Postharvest Biology and Technology*.
- Singh, P.K., Tiwari, S.K., Rai, N., Rai, K. and Singh, M., 2016. Antioxidant and phytochemical levels and their interrelation in stem and leaf extract of water spinach (*Ipomea aquatica*). *Indian Journal of Agricultural Sciences*, 86 (3), pp. 347-54.

Chalupowicz, L., Dombrovsky, A., Gaba, V., Luria, N., Reuven, M., Beerman, A., Lachman, O., Dror, O., Nissan, G. and Manulis-Sasson, S., 2019. Diagnosis of plant diseases using the Nanopore sequencing platform. *Plant Pathology*, 68 (2), pp. 229-238.

de Waard, M.A., Georgopoulos, S.G., Hollomon, D.W., Ishii, H., Leroux, P. and Ragsdale, N.N., 1993. Chemical control of plant diseases: problems and prospects. *Annual review of Phytopathology (USA)*.

Minsavage, G.V., Thompson, C.M., Hopkins, D.L., Leite, R.M.V.B.C. and Stall, R.E., 1994. Development of a polymerase chain reaction protocol for detection of *Xylella fastidiosa* in plant tissue. *Phytopathology*, 84 (5), pp. 456-461.

Haelterman, R.M., Tolocka, P.A., Roca, M.E., Guzmán, F.A., Fernández, F.D. and Otero, M.L., 2015. First presumptive diagnosis of *Xylella fastidiosa* causing olive scorch in Argentina. *Journal of Plant Pathology*, 97 (2).

Saponari, M., Manjunath, K. and Yokomi, R.K., 2008. Quantitative detection of Citrus tristeza virus in citrus and aphids by real-time reverse transcription-PCR (TaqMan®). *Journal of Virological Methods*, 147 (1), pp. 43-53.

Ghosal, S., Blystone, D., Singh, A.K., Ganapathysubramanian, B., Singh, A. and Sarkar, S., 2018. An explainable deep machine vision framework for plant stress phenotyping. *Proceedings of the National Academy of Sciences*, 115 (18), pp. 4613-4618.

Revathi, P. and Hemalatha, M. (2013) 'Identification of cotton diseases based on cross information gain\_deep forward neural network classifier with PSO feature selection, *International Journal of Engineering and Technology*, 5 (6), pp. 4637-4642.

Dhakal, A. and Shakya, S., 2018. Image-based plant disease detection with deep learning. *International Journal of Computer Trends and Technology*, 61 (1).

Ahmadi, P., Muharam, F.M., Ahmad, K., Mansor, S. and Abu Seman, I., 2017. Early detection of *Ganoderma* basal stem rot of oil palms using artificial neural network spectral analysis. *Plant disease*, 101 (6), pp. 1009-1016.

- Mohanty, S. P., Hughes, D. P. and Salathé, M., 2016. Using Deep Learning for Image-Based Plant Disease Detection, in *Frontiers in Plant Science*.
- Amara, J., Bouaziz, B. and Algergawy, A., 2017. A Deep Learning-based Approach for Banana Leaf Diseases Classification, In *BTW (Workshops)*, pp. 79–88.
- Golhani, K., Balasundram, S.K., Vadamalai, G. and Pradhan, B., 2018. A review of neural networks in plant disease detection using hyperspectral data. *Information Processing in Agriculture*, 5 (3), pp. 354-371.
- Huang, K. Y. (2012) 'Detection and classification of areca nuts with machine vision', *Computers and Mathematics with Applications*, 64(5), pp. 739–746.
- Brahimi, M., Arsenovic, M., Laraba, S., Sladojevic, S., Boukhalifa, K. and Moussaoui, A., 2018. Deep learning for plant diseases: detection and saliency map visualisation. In *Human and machine learning* (pp. 93-117). Springer, Cham.
- Ramcharan, A., Baranowski, K., McCloskey, P., Ahmed, B., Legg, J. and Hughes, D.P., 2017. Deep learning for image-based cassava disease detection. *Frontiers in plant science*, 8, p. 1852.
- Sladojevic, S., Arsenovic, M., Anderla, A., Culibrk, D. and Stefanovic, D., 2016. Deep neural networks based recognition of plant diseases by leaf image classification. *Computational intelligence and neuroscience*, 2016.
- D.Pujari, J., Yakkundimath, R. and Byadgi, A. S., 2016. 'SVM and ANN Based Classification of Plant Diseases Using Feature Reduction Technique', *International Journal of Interactive Multimedia and Artificial Intelligence*, 3(7), p. 6.
- Nerkar, B. and Talbar, S., 2020. 'Fusing Convolutional Neural Networks to Improve the Accuracy of Plant Leaf Disease Classification', *Current Journal of Applied Science and Technology*.
- Guo, Y., Zhang, J., Yin, C., Hu, X., Zou, Y., Xue, Z. and Wang, W., 2020. Plant Disease Identification Based on Deep Learning Algorithm in Smart Farming. *Discrete Dynamics in Nature and Society*, 2020.
- Barbedo, J. G. A., 2018. 'Impact of dataset size and variety on the effectiveness of deep learning and transfer learning for plant disease classification', *Computers and Electronics in Agriculture*, 153(August), pp. 46–53.

- Barbedo, J.G.A., 2019. Plant disease identification from individual lesions and spots using deep learning. *Biosystems Engineering*, 180, pp. 96-107.
- Patil, P. S. P. and Zambre, M. R. S., 2014. Classification of Cotton Leaf Spot Disease Using Support Vector Machine, 4 (5), pp. 92–97.
- Xie, C., Shao, Y., Li, X. and He, Y., 2015. Detection of early blight and late blight diseases on tomato leaves using hyperspectral imaging. *Scientific reports*, 5 (1), pp. 1-11.
- Wang, H., Qin, F., Liu, Q., Ruan, L., Wang, R., Ma, Z., Li, X., Cheng, P. and Wang, H., 2015. Identification and disease index inversion of wheat stripe rust and wheat leaf rust based on hyperspectral data at canopy level. *Journal of Spectroscopy*, 2015.
- Castro, W., Oblitas, J., Maicelo, J. and Avila-George, H., 2018. Evaluation of expert systems techniques for classifying different stages of coffee rust infection in hyperspectral images. *International Journal of Computational Intelligence Systems*, 11 (1), pp. 86-100.
- Owomugisha, G. and Mwebaze, E. (2017) ‘Machine learning for plant disease incidence and severity measurements from leaf images’, in *Proceedings - 2016 15th IEEE International Conference on Machine Learning and Applications, ICMLA 2016*, pp. 158–163.
- Phadikar, S., Sil, J. and Das, A.K., 2012. Classification of rice leaf diseases based on morphological changes. *International Journal of Information and Electronics Engineering*, 2 (3), pp. 460-463.
- Chaudhary, A., Kolhe, S. and Kamal, R. ,2016. An improved random forest classifier for multi-class classification, *Information Processing in Agriculture*.
- Naik, H.S., Zhang, J., Lofquist, A., Assefa, T., Sarkar, S., Ackerman, D., Singh, A., Singh, A.K. and Ganapathysubramanian, B., 2017. A real-time phenotyping framework using machine learning for plant stress severity rating in soybean. *Plant methods*, 13 (1), pp. 1-12
- Simonyan, K. and Zisserman, A., 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V. and Rabinovich, A., 2015. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1-9).

Krizhevsky, A., Sutskever, I. and Hinton, G.E., 2012. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, pp. 1097-1105.

Ward, J.M., Stromberg, E.L., Nowell, D.C. and Nutter Jr, F.W., 1999. Gray leaf spot: a disease of global importance in maize production. *Plant disease*, 83 (10), pp. 884-895.

Campus, P., 2012. Inoculation methods and disease rating scales for maize diseases. (Revised). *Directorate of Maize Research, New Delhi*.

Dillard, H.R. and Seem, R.C., 1990. Incidence-severity relationships for common maize rust on sweet corn. *Phytopathology*, 80 (9), pp. 842-849.

Jackson, T.A., 2008. G08-1680 Rust Diseases of Corn in Nebraska.

Bhange, M. and Hingoliwala, H.A., 2015. Smart farming: Pomegranate disease detection using image processing. *Procedia Computer Science*, 58, pp. 280-288.

Dixit, A. and Nema, S., 2018. Wheat Leaf Disease Detection Using Machine Learning Method-A Review. *International Journal of Computer Science and Mobile Computing*, 7 (5), pp. 124-129.

Wang, G., Sun, Y. and Wang, J., 2017. Automatic image-based plant disease severity estimation using deep learning. *Computational intelligence and neuroscience*, 2017.

Zhang, Y.D., Pan, C., Chen, X. and Wang, F., 2018. Abnormal breast identification by nine-layer convolutional neural network with parametric rectified linear unit and rank-based stochastic pooling. *Journal of computational science*, 27, pp. 57-68.

Jackson-Ziems, T.A., 2016. Northern Corn Leaf Blight.

Ozguven, M.M. and Adem, K., 2019. Automatic detection and classification of leaf spot disease in sugar beet using deep learning algorithms. *Physica A: Statistical Mechanics and its Applications*, 535, p. 122537.



Ding, J., Ali, F., Chen, G., Li, H., Mahuku, G., Yang, N., Narro, L., Magorokosho, C., Makumbi, D. and Yan, J., 2015. Genome-wide association mapping reveals novel sources of resistance to northern corn leaf blight in maize. *BMC plant biology*, 15 (1), pp. 1-11.

University of Minnesota, Northern corn leaf blight, 2021, viewed 2 March 2021, <<https://extension.umn.edu/corn-pest-management/northern-corn-leaf-blight>>.

Pataky, J.K., 1985. Relationships among reactions of sweet corn hybrids to Goss' wilt, Stewart's bacterial wilt, and northern corn leaf blight. *Plant disease*, 69 (10), pp. 845-848.

Abera, W., Hussein, S., Derera, J., Worku, M. and Laing, M., 2016. Heterosis and combining ability of elite maize inbred lines under northern corn leaf blight disease prone environments of the mid-altitude tropics. *Euphytica*, 208 (2), pp. 391-400.

Technow, F., Bürger, A. and Melchinger, A.E., 2013. Genomic prediction of northern corn leaf blight resistance in maize with combined or separated training sets for heterotic groups. *G3: Genes/ Genomes/ Genetics*, 3 (2), pp. 197-203.

Cruz, A.C., Luvisi, A., De Bellis, L. and Ampatzidis, Y., 2017. Vision-based plant disease detection system using transfer and deep learning. In *2017 asabe annual international meeting* (p. 1). American Society of Agricultural and Biological Engineers.

Chen, J., Chen, J., Zhang, D., Sun, Y. and Nanekaran, Y.A., 2020. Using deep transfer learning for image-based plant disease identification. *Computers and Electronics in Agriculture*, 173, p. 105393.

Oppenheim, D., Shani, G., Erlich, O. and Tsror, L., 2019. Using deep learning for image-based potato tuber disease detection. *Phytopathology*, 109 (6), pp. 1083-1087.

Arsenovic, M., Karanovic, M., Sladojevic, S., Anderla, A. and Stefanovic, D., 2019. Solving current limitations of deep learning based approaches for plant disease detection. *Symmetry*, 11 (7), p. 939.

Carpane, P.D., Peper, A.M. and Kohn, F., 2020. Management of Northern Corn Leaf Blight using Nativo (Trifloxistrobin+ Tebuconazole) Fungicide Applications. *Crop Protection*, 127, p. 104982.

Huang, S., Liu, W., Qi, F. and Yang, K., 2019, August. Development and validation of a deep learning algorithm for the recognition of plant disease. In *2019 IEEE 21st International Conference on High Performance Computing and Communications; IEEE 17th International Conference on Smart City; IEEE 5th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)* (pp. 1951-1957). IEEE.

Nagasubramanian, K., Jones, S., Singh, A.K., Sarkar, S., Singh, A. and Ganapathysubramanian, B., 2019. Plant disease identification using explainable 3D deep learning on hyperspectral images. *Plant methods*, 15 (1), pp. 1-10.

Brahimi, M., Boukhalifa, K. and Moussaoui, A., 2017. Deep learning for tomato diseases: classification and symptoms visualization. *Applied Artificial Intelligence*, 31 (4), pp. 299-315.

Ale, L., Sheta, A., Li, L., Wang, Y. and Zhang, N., 2019, December. Deep learning based plant disease detection for smart agriculture. In *2019 IEEE Globecom Workshops (GC Wkshps)* (pp. 1-6). IEEE.

Geetharamani, G. and Pandian, A., 2019. Identification of plant leaf diseases using a nine-layer deep convolutional neural network. *Computers & Electrical Engineering*, 76, pp. 323-338.

Goncharov, P., Ososkov, G., Nechaevskiy, A., Uzhinskiy, A. and Nestsiaenia, I., 2018, October. Disease detection on the plant leaves by deep learning. In *International Conference on Neuroinformatics* (pp. 151-159). Springer, Cham.

Militante, S.V., Gerardo, B.D. and Dionisio, N.V., 2019, October. Plant Leaf Detection and Disease Recognition using Deep Learning. In *2019 IEEE Eurasia Conference on IoT, Communication and Engineering (ECICE)* (pp. 579-582). IEEE.

TÜRKOĞLU, M. and Hanbay, D., 2019. Plant disease and pest detection using deep learning-based features. *Turkish Journal of Electrical Engineering & Computer Sciences*, 27 (3), pp. 1636-1651.

Ferentinos, K.P., 2018. Deep learning models for plant disease detection and diagnosis. *Computers and Electronics in Agriculture*, 145, pp. 311-318.

Baranwal, S., Khandelwal, S. and Arora, A., 2019, February. Deep learning convolutional neural network for apple leaves disease detection. In *Proceedings of International Conference on Sustainable Computing in Science, Technology and Management (SUSCOM)*, Amity University Rajasthan, Jaipur-India.

- Rangarajan, A.K., Purushothaman, R. and Ramesh, A., 2018. Tomato crop disease classification using pre-trained deep learning algorithm. *Procedia computer science*, 133, pp.1040-1047.
- Too, E.C., Yujian, L., Njuki, S. and Yingchun, L., 2019. A comparative study of fine-tuning deep learning models for plant disease identification. *Computers and Electronics in Agriculture*, 161, pp. 272-279.
- Otsu, N., 1979. A threshold selection method from gray-level histograms. *IEEE transactions on systems, man, and cybernetics*, 9 (1), pp. 62-66.
- Craven, M., Morey, L., Abrahams, A., Njom, H.A. and van Rensburg, B.J., 2020. Effect of northern corn leaf blight severity on Fusarium ear rot incidence of maize. *South African Journal of Science*, 116 (11-12), pp. 1-11.
- Patil, S.B. and Bodhe, S.K., 2011. Leaf disease severity measurement using image processing. *International Journal of Engineering and Technology*, 3 (5), pp. 297-301.
- Barbedo, J.G.A., 2014. An automatic method to detect and measure leaf disease symptoms using digital image processing. *Plant Disease*, 98 (12), pp. 1709-1716.
- Parikh, A., Raval, M.S., Parmar, C. and Chaudhary, S., 2016, October. Disease detection and severity estimation in cotton plant from unconstrained images. In *2016 IEEE International Conference on Data Science and Advanced Analytics (DSAA)* (pp. 594-601). IEEE.
- Shrivastava, S., Singh, S.K. and Hooda, D.S., 2015. Color sensing and image processing-based automatic soybean plant foliar disease severity detection and estimation. *Multimedia Tools and Applications*, 74 (24), pp. 11467-11484.
- Behera, S.K., Jena, L., Rath, A.K. and Sethy, P.K., 2018, April. Disease classification and grading of orange using machine learning and fuzzy logic. In *2018 International Conference on Communication and Signal Processing (ICCSP)* (pp. 0678-0682). IEEE.
- Sannakki, S.S., Rajpurohit, V.S., Nargund, V.B., Kumar, A. and Yallur, P.S., 2011. Leaf disease grading by machine vision and fuzzy logic. *Int J*, 2 (5), pp. 1709-1716.
- Rastogi, A., Arora, R. and Sharma, S., 2015, February. Leaf disease detection and grading using computer vision technology & fuzzy logic. In *2015 2nd international conference on signal processing and integrated networks (SPIN)* (pp. 500-505). IEEE.

Weizheng, S., Yachun, W., Zhanliang, C. and Hongda, W., 2008, December. Grading method of leaf spot disease based on image processing. In *2008 international conference on computer science and software engineering* (Vol. 6, pp. 491-494). IEEE.

Chaudhary, P., Godara, S., Cheeran, A.N. and Chaudhari, A.K., 2012. Fast and accurate method for leaf area measurement. *International journal of computer applications*, 49 (9), pp. 22-25.

Zhou, G., Zhang, W., Chen, A., He, M. and Ma, X., 2019. Rapid detection of rice disease based on FCM-KM and faster R-CNN fusion. *IEEE Access*, 7, pp. 143190-143206.

Denson, N. and Seltzer, M.H., 2011. Meta-analysis in higher education: An illustrative example using hierarchical linear modeling. *Research in Higher Education*, 52 (3), pp. 215-244.

Isleib, J., 2012. Signs and symptoms of plant disease: Is it fungal, viral or bacterial. *Michigan State University Extension*.

He, K., Zhang, X., Ren, S. and Sun, J., 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770-778).

Khattab, D., Ebied, H.M., Hussein, A.S. and Tolba, M.F., 2014. Color image segmentation based on different color space models using automatic GrabCut. *The Scientific World Journal*, 2014.

Elgendy M. Deep Learning for Vision Systems. Manning Publications; 2020 Nov 10.

Sibiya, M. and Sumbwanyambe, M., 2019b. An algorithm for severity estimation of plant leaf diseases by the use of colour threshold image segmentation and fuzzy logic inference: a proposed algorithm to update a “leaf doctor” application. *Agri Engineering*, 1 (2), pp. 205-219.

Ioffe, S. and Szegedy, C., 2015, June. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning* (pp. 448-456). PMLR.

Heaton, J., 2013. *Artificial Intelligence for Humans. Volume 1: Fundamental Algorithms*. CreateSpace Independent Publishing: North Charleston, SC, USA.

- Savary, S., Ficke, A., Aubertot, J.N. and Hollier, C., 2012. Crop losses due to diseases and their implications for global food production losses and food security.
- Sankaran, S., Mishra, A., Ehsani, R. and Davis, C., 2010. A review of advanced techniques for detecting plant diseases. *Computers and Electronics in Agriculture*, 72 (1), pp. 1-13.
- Ward, J.M.J., Birch, E.B. and Nowell, D.C., 1994. Grey leaf spot on maize. *Pietermaritzburg: Cedara Agric. Develop. Inst.*
- Zhou, H., Wu, J. and Zhang, J., 2010a. *Digital Image Processing: Part I*. Bookboon.
- Zhou, H., Wu, J. and Zhang, J., 2010b. *Digital Image Processing: Part II*. Bookboon.
- Gonzalez, R.C. and Woods, R.E., 2002. Digital image processing.
- Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M. and Adam, H., 2017. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*.
- Fang, M., Yue, G. and Yu, Q., 2009. The study on an application of Otsu method in canny operator. In *Proceedings. The 2009 International Symposium on Information Processing (ISIP 2009)* (p. 109). Academy Publisher.
- Yang, P., Song, W., Zhao, X., Zheng, R. and Qingge, L., 2020. An improved Otsu threshold segmentation algorithm. *International Journal of Computational Science and Engineering*, 22 (1), pp. 146-153.
- Sun, H., Wei, J., Zhang, J. and Yang, W., 2014. A comparison of disease severity measurements using image analysis and visual estimates using a category scale for genetic analysis of resistance to bacterial spot in tomato. *European Journal of Plant Pathology*, 139 (1), pp. 125-136.
- El Jarroudi, M., Kouadio, A.L., Mackels, C., Tychon, B., Delfosse, P. and Bock, C.H., 2015. A comparison between visual estimates and image analysis measurements to determine *Septoria* leaf blotch severity in winter wheat. *Plant Pathology*, 64 (2), pp. 355-364.

- Murakami, P.F., 2005. *An instructional guide for leaf color analysis using digital imaging software* (Vol. 327). US Department of Agriculture, Forest Service, Northeastern Research Station.
- Wijekoon, C.P., Goodwin, P.H. and Hsiang, T., 2008. Quantifying fungal infection of plant leaves by digital image analysis using Scion Image software. *Journal of microbiological methods*, 74 (2-3), pp. 94-101.
- Pallottino, F., Menesatti, P., Figorilli, S., Antonucci, F., Tomasone, R., Colantoni, A. and Costa, C., 2018. Machine vision retrofit system for mechanical weed control in precision agriculture applications. *Sustainability*, 10 (7), p. 2209.
- Bock, C.H., Cook, A.Z., Parker, P.E. and Gottwald, T.R., 2009. Automated image analysis of the severity of foliar citrus canker symptoms. *Plant disease*, 93 (6), pp. 660-665.
- Faggella, D., 2018. What is machine learning?-an informed definition.
- Mitchell, R., Michalski, J. and Carbonell, T., 2013. *An artificial intelligence approach*. Berlin: Springer.
- Aurélien, G., 2017. Hands-on machine learning with scikit-learn & tensorflow. *Geron Aurelien*.
- Baylor, D., Breck, E., Cheng, H.T., Fiedel, N., Foo, C.Y., Haque, Z., Haykal, S., Ispir, M., Jain, V., Koc, L. and Koo, C.Y., 2017, August. Tfx: A tensorflow-based production-scale machine learning platform. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 1387-1395).
- Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V. and Vanderplas, J., 2011. Scikit-learn: Machine learning in Python. *The Journal of machine Learning research*, 12, pp. 2825-2830.
- Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S. and Darrell, T., 2014, November. Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the 22nd ACM international conference on Multimedia* (pp. 675-678).
- Seide, F., 2017, February. Keynote: The computer science behind the Microsoft cognitive toolkit: an open source large-scale deep learning toolkit for Windows and Linux. In *2017 IEEE/ACM International Symposium on Code Generation and Optimization (CGO)* (pp. xi-xi). IEEE.

L. A. Zadeh, "Fuzzy logic - A personal perspective," *Fuzzy Sets Syst.*, 2015.

Schröder, T., Krüger, K. and Kümmerlen, F., 2014. Image processing based deflagration detection using fuzzy logic classification. *Fire safety journal*, 65, pp. 1-10.

Celik, H. Demirel, H. Ozkaramanli, and M. Uyguroglu, "Fire detection using statistical color model in video sequences," *J. Vis. Commun. Image Represent.*, 2007.

Davim, J.P. ed., 2012. *Statistical and computational techniques in manufacturing*. Springer Science & Business Media.

Naik, H.S., Zhang, J., Lofquist, A., Assefa, T., Sarkar, S., Ackerman, D., Singh, A., Singh, A.K. and Ganapathysubramanian, B., 2017. A real-time genotyping framework using machine learning for plant stress severity rating in soybean. *Plant methods*, 13 (1), pp. 1-12.

Ward, J.M., 1999. Gray leaf spot of maize

Kröse, B., Krose, B., van der Smagt, P. and Smagt, P., 1993. An introduction to neural networks.

Patterson, J. and Gibson, A., 2017. *Deep learning: A practitioner's approach*. " O'Reilly Media, Inc."