IMAGE BASED BARLEY LEAF FUNGAL DISEASE DETECTION USING CONVOLUTIONAL NEURAL NETWORK

MAMO GUDISA

HAWASSA UNIVERSITY, HAWASSA, ETHIOPIA

November, 2022

IMAGE BASED BARLEY LEAF FUNGAL DISEASE DETECTION USING CONVOLUTIONAL NEURAL NETWORK

MAMO GUDISA

MAJOR ADVISOR: DEGIF TEKA (PhD)

A THESIS SUBMITTED TO THE DEPARTMENT OF COMPUTER SCIENCE

HAWASSA UNIVERSITY

INSTITUTE OF TECHNOLOGY

IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE IN COMPUTER SCIENCE

HAWASSA, ETHIOPIA

November, 2022

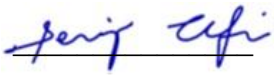
APPROVAL SHEET-I

This is to certify that the thesis entitled "**IMAGE BASED BARLEY LEAF FUNGAL DISEASE DETECTION USING CONVOLUTIONAL NEURAL NETWORK**" submitted in partial fulfillment of the requirements for the degree of Master's with specialization in Computer Science, the Graduate Program of the Department/School of Informatics, and has been carried out by Mamo Gudisa. Therefore, we recommend that the student has fulfilled the requirements and hence hereby can submit the thesis to the department.

_____	_____	_____
Name of major advisor	Signature	Date
_____	_____	_____
Name of co-advisor	Signature	Date

APPROVAL SHEET-II

We, the undersigned, members of the Board of Examiners of the final open defense by Mamo Gudisa have read and evaluated his/her thesis entitled "**IMAGE BASED BARLEY LEAF FUNGAL DISEASE DETECTION USING CONVOLUTIONAL NEURAL NETWORK**", and examined the candidate. This is, therefore, to certify that the thesis has been accepted in partial fulfillment of the requirements for the degree.

_____	_____	_____
Name of Major Advisor	Signature	Date
_____	_____	_____
Name of Internal Examiner-I	Signature	Date
_____	_____	_____
Name of Internal Examiner-II	Signature	Date
Dr. Dereje Teferi		January 11, 2023
Name of External examiner	Signature	Date
_____	_____	_____
SGS Approval	Signature	Date

ACKNOWLEDGMENTS

*I would like first to thank **GOD** for giving me strength and ability to learn, understand and complete this work. After that, I would like to thank my main advisor, **Dr. Degif Teka** for his close supervision and constructive suggestion during my thesis work. He has been devoting his time and providing ideas to carry out the research. I thank you for your encouragement starting from the beginning to the end of this thesis work.*

*Next, my deepest gratitude goes to **my parents** for providing me with unfailing support and continuous encouragement throughout my life.*

Finally, I would like to give my gratitude to people who are not mentioned in name but whose effort helped me much all along.

STATEMENT OF THE AUTHOUR

I hereby declare that this MSc thesis is my original work and has not been presented for a degree in any other university, and all sources of material used for this thesis / have been duly acknowledged.

Name: ……...……. Signature: …….……

Place: Institute of Technology, Hawassa University, Hawassa

Date of Submission: ……

Table of Contents

List of Figures

List of Tables

ABBREVIATIONS AND ACRONYMS

DL	*Deep Learning*
CNN	*Convolutional Neural Network*
ADAM	*Adaptive Learning Rate Optimization*
ML	*Machine Learning*
ANN	*Artificial Neural Network*
BCE	*Binary Cross-Entropy*
CNN	*Convolutional Neural Network*
DBN	*Deep Belief Networks*
FC	*Fully Connected*
GPU	*Graphics Processing Units*
HOT	*Histogram Of Template*
ML	*Machine Learning*
ReLU	*Rectified Linear Unit*
RGB	*Red, Green, And Blue*
RNN	*Recurrent Neural Network*
ANN	*Artificial Neural Network*
BCE	*Binary Cross-Entropy*
BCDD	*Barley crop disease detection*
KRC	*Kulumsa Research Center*
IAR	*Institute of Agricultural Research*

Abstract

Barley is one of the most grown crops in the east Arsi, west Arsi, and Bale zones of Oromia Region. It is one of the main sources of food and income in these and other areas of Ethiopia. However, barley crop was affected by fungal disease which reduces the production and main cause for the economic losses in agricultural industries in Ethiopia. For the betterment of human health, fungal diseases in leaf of barley crops must be controlled and effectively monitored. The earlier researchers have used hand-crafted-features for image classification and recognition with machine learning approach. Nowadays, the development in Deep Learning has allowed researchers to drastically improve the accuracy of object detection and classification. In this thesis, the researcher used a deep-learning approach which is Convolutional Neural Network algorithm to detect fungal disease of barley crop using leaf images collected from Arsi zone of Kulumsa research center and other images captured directly from different Barley farms. The researcher dataset contains two categories of barley crop: leaf rust and normal. The dataset contains 10,224 healthy and diseased images. From this, 80% of the images are used for training and the rest for testing the model. During training, data augmentation is used to generate more images to fit the proposed model. Additionally, many researchers agree that using data augmentation can also increase the performance of the model. The designed model is trained and tested using the collected dataset and compared with two pre-trained convolutional neural network models namely Mobile Net and InceptionV3. The model obtained 99.53% accuracy and it can be used as a practical tool for farmers to protect barley crop, against fungal diseases.

Keyword: *Image Classification, Convolutional neural network, fungal disease of Barley crop, Disease detection, deep learning.*

CHAPTER ONE

INTRODUCTION

1.1 Background of the study

The Ethiopian economy is mostly based on agriculture; almost 85% of the people rely on agriculture as their primary source of income [1]. Agriculture, in this perspective, is critical to Ethiopia's economy. Ethiopia is one of the African countries with a large potential for the creation of various agricultural crops. Ethiopians eat a variety of crops, including barley. As a result, Arsi and Bale are considered Ethiopia's barley Belts. Currently, malt barley production of a varieties; Beka, Holker, Miscal, and Walia has been grown in the area. The area covered by malt barley production in Arsi, which is the major malt supply producing area, has significantly increased (more than 85 thousand hectares per year since 2012) with average annual growth rate of 11.4%. Correspondingly, the production obtained in the same period increased to about 120 thousand tons since 2012, with average annual growth rate of 13.2 %, However now days this production is becoming decrease due to fungal disease The Arsi and West Arsi zones constitute over 85% of the total malting barley area in the country. [46] But barley production is currently influenced by biotic and abiotic factors, such as diseases and insect pests, which contribute to low yield and poor quality. The most serious issue among those elements is a fungus-caused disease.

The issue of effective crop disease prevention is inextricably linked to issues of sustainable agriculture and climate change [2]. According to research, climatic change can affect pathogen growth stages and rates, as well as host resistance, resulting in physiological alterations in host-pathogen interactions [3, 4]. The situation is made even more complicated by the fact that diseases are now more easily transmitted globally than ever before. New diseases can emerge in regions where they have never been seen before, and where there is, there is usually no local competence to combat them [5, 6, 7]. Inexperienced pesticide use can lead to micro-organisms developing long-term resistance, severely limiting their ability to fight back. One of the pillars of precision agriculture is timely and precise identification of crop diseases [8]. Crop diseases can be detected in a variety of methods. Some diseases have no visible symptoms, or the damage becomes apparent too late to intervene, necessitating a thorough investigation. However, because most diseases

exhibit themselves in the visible spectrum, a skilled professional's naked eye examination is the most common method for detecting crop diseases in practice. Crop pathologists can identify barley diseases, especially the fungal diseases (experts in the field of agriculture). However, getting specialists to identify the diseases is expensive for farmers who live far from where experts may be located; this is the area's biggest weakness. To address this issue and correctly diagnose fungal diseases, researchers developed a computerized model that can detect the disease using computer vision and deep learning techniques, based on symptoms that are immediately visible in the barley crops leaf. Therefore, an automated detection strategy for barley fungal infections was established in this work by utilizing deep learning approach.

1.2 Statement of problem

Barley is the main crop grown in Ethiopia central, eastern, and western areas [9]. However, there are a variety of problems that impact the crop throughout the cultivation process, such as diseases and pests, the most influential of which is fungal disease. The detection of barley fungal disease, like other crop diseases, needs careful attention and the expertise of professionals in the field [10]. Researchers in Ethiopia have discovered that the disease is causing significant output losses in regions where barley is grown [10]. In Ethiopia, up to 70% of barley crops are affected by the barley fungal disease. Farmers have been forced to cease barley cultivation due to the diseases, resulting in a catastrophic food scarcity in Oromia's highly populated zones. More than 20 percent of the country's farmers barley crops are directly affected by this disease [10]. Because the majority of Ethiopian farmers have no knowledge and do not receive accurate and full information regarding barley crop diseases, they require expert help. Furthermore, there is a scarcity of resources and competence in the field of barley pathology in areas where the crop is grown. Furthermore, crop pathologists are unable to reach every farm, and even when they do, they must rely on manual eye observation. The manual prediction approach is inaccurate, time-consuming, and requires a great deal of work. The relevance of employing computer vision to diagnose crop disease is not widely understood in this field. [12].

In other words, there are a lot of crops and diseases that are not addressed in the current technologies. Therefore, the researcher will need to extend the works to address more diseases and crops. There is a fact that some previously worked disease identification and classification researches are conducted by using some strict methods. For example, the images used for training

and testing are taken under a certain condition like in the laboratory within a proper lighting system, and strict angle of capture, collecting sample images from publicly available dataset instead of capturing real-world images from the field, use of traditional image processing techniques, and so on [13, 14, 15, 16].

Therefore, there is a need to develop an autonomous disease detection model that aids farmers in detecting diseases earlier and with more precision. To identify fungal diseases on leaf of barley crops, the researcher applies a deep learning method. The computer vision approach is a non-invasive technology that allows farmers to diagnose fungal disease in a consistent, relatively accurate, time-efficient, and cost-effective manner.

1.3 Research questions

Because of the difficulty of the disease, the solution to the difficulties described above require scientific probing. As a result, before coming up with a solution, the issues should be expressed as researchable questions. The following research questions were posed by the researcher:

1. How to build a CNN model that performs better result in detecting and classification of barley disease?
2. What are the hyperparameters that help in designing best performing CNN model for barley disease detection?
3. Which CNN model performs best in detecting of barley disease?

1.4 Objectives of the study

1.4.1 General objectives

The main objective of this thesis is to design and develop an automatic barley leaf fungal disease identification model by using a deep learning approach.

1.4.2 Specific objectives

Basically, the research aims to achieve the following specific goals:

- Review the literature for prior computer vision-based agricultural disease identifications
- Collect images of fungal diseased barley as well as healthy barley crops.
- To analyze the image dataset, choose a suitable approach and instrument.
- Develop a convolutional neural network model.

- Train the proposed model by using the collected dataset
- Evaluate the model

1.5 Significance of the study

The researcher agrees that technology is here to stay and that there is no way to escape using it. As a result, people need to employ technology in order to better their living situations. Deep learning for computer vision is a modern technology that is widely used for a variety of reasons. Because the physical appearance of barley is modest, detecting fungal disease with a necked eye takes a long time and effort, is less accurate, and can only be used in a narrow region. On the other hand, automatic disease detection techniques and approaches, take less time, require less work, more accurate, and cover larger regions.

In addition to these, the significance of this study is described as follows:

- First and foremost, this thesis will help agriculture specialists recognize the significance of computer vision in agriculture.
- Because the disease is recognized early and agricultural specialists are not needed, the study will aid in achieving a high output.
- The research aid in lowering the cost of production, which causes farmers to lose a lot of money owing to the over use of pesticides on their crops.
- It lowers the expense of hiring specialists to keep track of crops on vast farms.
- The findings of this study assist various agencies in taking appropriate action in cases of fungal disease.

Finally, this study may serve as reference material for the researchers who will conduct their research in computer vision especially researches related to crop disease identification.

1.6 Scope and limitation of the study

The scope of this study is limited to barley crops. The research primarily focused on the potential of deep learning in the agriculture industry to improve decision-making. As its main input, the model employs healthy and diseased leaf images of barley crops taken from a barley farm and research center. Images were taken at a variety of farms in the Oromia Region's Digalu/Tijo woreda Arsi zone and from Kulumsa Research Center. This study only looks for fungal diseases in barley leaf; it would not look for other diseases like Sigatoka (leaf spot), viruses, or temperature

change due to lack of budget, lack of time etc. Even while fungal disease affects other crops such as wheat and other crops, this study was limited to only barley leaf. There is no method to prescribe disease medicine or proper treatment once the disease has been detected in the crop, and measuring the severity of the disease is beyond the scope of this thesis.

1.7 Organization of the Thesis

The following is how the rest of the thesis is organized: The review of literature in the subject of crop disease detection connected to machine learning, deep learning, and investigations undertaken in those areas is presented in Chapter Two. The literature review section attempts to provide a concise overview of relevant publications.

The approach used to perform this thesis are briefly addressed in chapter three, covering data collecting methods, materials, and instruments employed, the selection of a model for the research, the suggested model's architectural design, and assessment procedures. The fourth chapter examines the challenge of detecting fungal disease in barley crops using infected and healthy leaf images of barley crop. In addition, several experimental outcomes are addressed in depth in this chapter. The investigation was finally completed, and conclusion and recommendation were provided in the final chapter.

CHAPTER TWO

LITERATURE REVIEW

The chapter reviews computer vision processes and techniques, deep learning and machine learning methodologies of the earlier studies as well as the barley crop and how typical neural networks function. Finally, the chapter finalized with a review of relevant literature and a discussion of the research gaps.

2.1 Barley crop

Barley is one of the most important food crops in Ethiopia. It is used primarily to produce flour for bread, chuko, Enjera and used widely in the production of many other cultural foods in Ethiopia especially in Arsi and Bale zone. Barley grains also used in the manufacture of alcoholic beverages and its straw is used as an animal feed and in the manufacture of carpets, packing, bedding, and others. Barley can be grown in an environment with wide variety of climate conditions but grows best in cool regions where the temperature is between 10 and 24°C (50–75°F). Barley will not grow at temperatures above 35°C (95°F). Barley will grow optimally in a deep, fertile,well-draining and well aerated soil at a pH level between 5.5 and 7.5. There are two types of barley that farmers grow in Ethiopia: food barley and malt barley. The majority of barley that farmers grow is food barley and it is the main ingredient for several staple dishes such as Enjera, Chuko, Dhanga, bread and etc. Recently, there has been an increasing demand for farmers to grow malt barley, which presently constitutes 10 percent of the total barley production. [24].

2.2 Fungal diseases affecting Barley crop

Septoria tritici blotch: The fungal disease Septoria tritici blotch forms tan, elongated blemishes on barley leaves. The degree of yellowing varies across kinds, although lesions may have a brown body bordered by yellow. The fungus's rows of black reproductive structures are essential diagnostic traits that may frequently be detected without magnification. Another name for this disease is "speckled leaf blotch." Temperatures ranging from 59 to 77 degrees Fahrenheit, as well as wet or humid weather lasting more than one day, are conducive to disease development. In the early spring, disease outbreaks are more common on lower leaves due to chilly, moist circumstances. As temperatures rise, the pathogen will begin to drop [18]. Figure 2.1 below illustrate leaf of barley crop affected by Septoria tritici blotch**.**

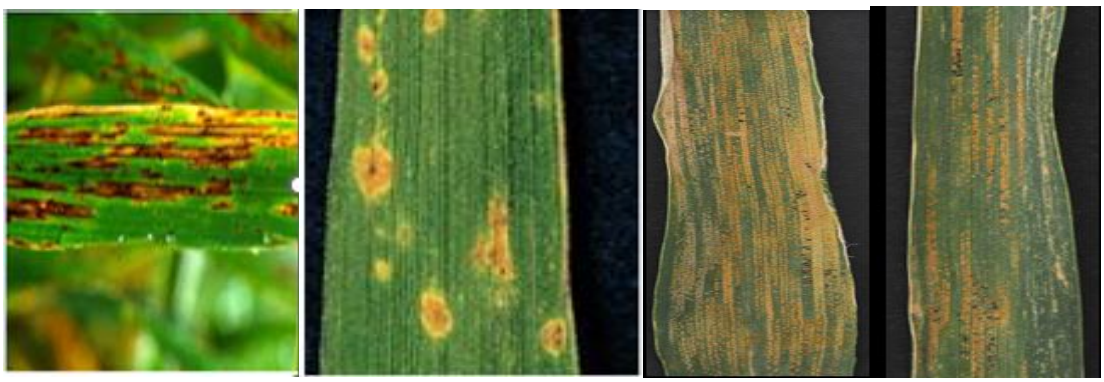


Figure 2.1: Barley crop affected by Septoria tritici blotch

Tan spot: Tan lesions with a yellow edge are a major diagnostic characteristic of tan spots. A black patch in the middle of mature tan spot lesions is common. As the lesions spread, they may meld together, resulting in vast regions of damaged leaf tissue. In the spring, the fungus that causes tan spot persists in the leftovers of past barley harvests, producing minute, black reproductive structures. [18]. Figure 2 illustrates leaf of barley crop affected by *Tan spot.*

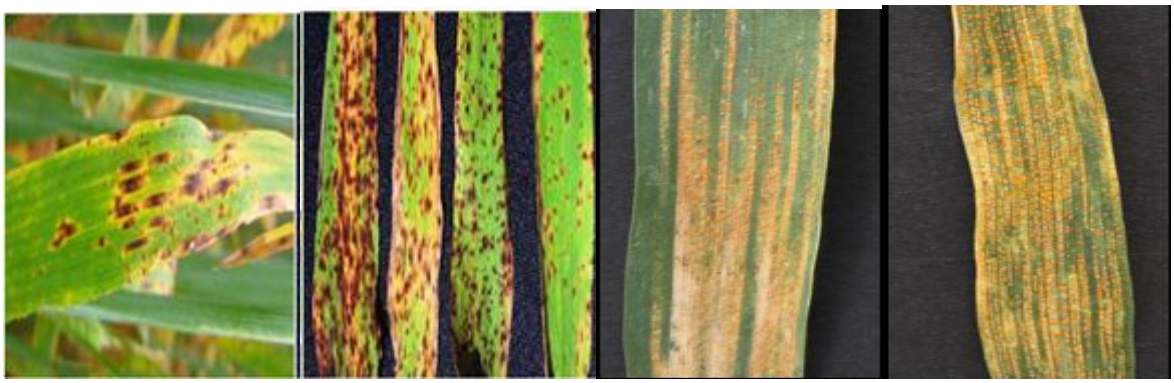


Figure 1.2: Barley crop affected by Tan spot.

Powdery mildew: White fungal growths on leaves and leaf sheaths are signs of powdery mildew. Fungal growth is mostly restricted to the outer surfaces of crops, and it may be readily removed by wiping a finger across the afflicted parts. Dark reproductive structures may be mixed in with the white, cottony growth of the fungus in leaf regions. Disease growth is best between 59- and 72-degrees Fahrenheit with high humidity, and it is more common in thick foliage and heavily fertilized locations. The disease may persist on volunteer barley, and when barley growth begins

in the spring, powdery mildew symptoms can first develop on the older leaves [17,18]. Figure 2.3 illustrates leaf of barley crop affected by Powdery mildew.

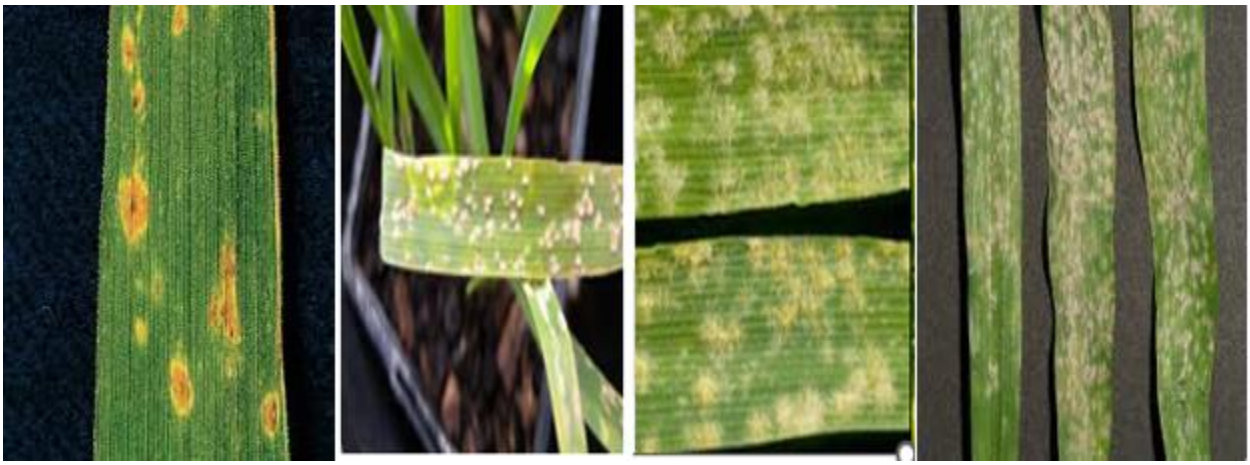


Figure 2.2: Barley crop affected by Powdery mildew.

Stripe rust: It produces a striped pattern of yellow blister-like lesions. The disease is most frequent on leaves, but when the disease is severe, it can also affect head tissue. Yellow rust is a term used to describe this disease. Temperatures between 50- and 64-degrees Fahrenheit, with periodic rain or dew, are favorable for disease growth. In years with chilly, moist springs, mild winters, and cool summers, disease levels can be high enough for spores to persist from season to season. Stripe rust may overwinter on leaf tissue, volunteer barley, and other grass hosts in temperatures as low as 23°F. The spores disintegrate quickly at temperatures over 59 °F. Figure 2.4 illustrates leaf of barley crop affected by Stripe rust.

Figure 2.3: Barley crop affected by Stripe rust.

Stem rust: causes blister-like lesions on leaves, leaf sheaths, and stems. Infection of glumesand awns is also possible. The reddish-brown spores of the fungus cause considerable tearing as they

burst through the outer layers of the crop tissues. Mature stem rust lesionsare more elongated than those of leaf rust [17, 18] but it is not well known in this research areas.

Figure 2.5 below illustrate leaves of health barley crops

Figure 2.4: Healthy Barley crops.

2.3 Artificial neural networks

Artificial neural networks (ANN) are a type of supervised machine learning technique that is commonly used in modeling. This thesis is primarily concerned with a deeper neural network known as a Convolutional Neural Network (CNN) which have many hidden layers which makes them to be different and powerful than the shallow neural networks.[29]

Artificial neural networks (ANNs) are statistical learning algorithms based on biological neural networks' features. They're used to solve a broad range of issues, from simple classification problems to speech recognition and computer vision. In the sense that they are implemented as a set of linked processing components, often termed nodes, that are functionally equivalent to biological neurons, ANNs are loosely based on biological neural networks. Artificial neural networks (ANN) are a type of supervised machine learning technique that is commonly used in modeling. Figure 2.6 illustrates a simple neural network with a single hidden layer.

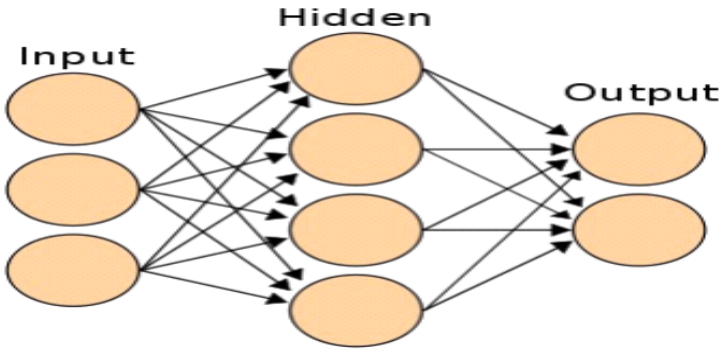


Figure 2.5: A simple neural network with one hidden layer.

2.3.1Activation Function

Activation functions are a critical part of the design of a neural network. The choice of activation function in the hidden layer controls how well the network model learns the training dataset. The choice of activation function in the output layer will define the type of predictions the model can make. An activation function in a neural network defines how the weighted sum of the input is transformed into an output from a node or nodes in a layer of the network. [29]

ReLU Function

Rectified linear unit has a derivative function and enables for back propagation while being computationally efficient, despite the fact that it seems to be a linear function. The key caveat is that the ReLU function does not simultaneously stimulate all of the neurons. Only if the result of the linear transformation is less than 0 will the neurons be silenced. [30]

$$F(x) = max\ (0, x) \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ (2.1)$$

Where F-represent activation function and X –is an input.

In this thesis, the most widely used activation function which is called Rectified Linear Unit (ReLU) has been used in the hidden layer of the network to make the proposed model more powerful and to learn complex features from data.

2.3.2.1 Needs of Activation function for Neural Network

The term "neural network" refers to a network made up of numerous layers of neurons. Nodes, which are used for categorization and prediction based on the data supplied as input to the network, can also be found among those neurons. An input layer, one or more hidden layers, and an output layer are all present. Each layer has nodes, and each node has a weight, which is taken into account while processing data from one layer to the next [36]. If a neural network does not employ an activation function, the output signal is just a basic linear function, which is a polynomial of degree one. Although a linear equation is simple and straightforward to solve, its complexity is restricted, and it lacks the capacity to learn and identify complicated data mappings. It is best to use a neural network without an activation function for simpler tasks like modeling complex kinds of unusual data like images, videos, audio, speech, text, and so on because it frequently behaves like a linear regression model with poor performance and power. [36].

2.4 Deep Learning

Deep learning is a machine learning area that uses a neural network as its architecture and bases its learning on a data representation algorithm rather than task-specific methods. [22 and 21] Deep learning approaches have been shown to outperform prior state-of-the-art machine learning techniques in a variety of domains in recent years, with computer vision being one of the most notable instances. Before the machine learning phase, the standard machine learning algorithm requires a separate hand-craft feature extraction. There is only one neural network phase in deep learning. The layers of the neural network are learning to recognize the essential elements of the input at the start of the network, and this data is fed forward to the subsequent levels for more network computation [20]. Deep learning is a very young approach that is continuously growing. Because of the availability of a big quantity of data and high-performance computing machine components such as GPU, deep learning now outperforms other standard machine learning algorithms [19]. Deep learning methods improve accuracy and performance by using multilayer (too many hidden layers) processing, and unlike typical machine learning methods, there is no explicit feature extraction. In other words, features are retrieved automatically from raw data in deep learning architecture and we can perform feature extraction and classification (it might be recognition depending on our problem) at once, therefore we only design a single model. The recurrent neural network (RNN), artificial neural network (ANN), and convolutional neural network (CNN) are three of the most often used supervised deep learning architectures.

RNN is a sort of supervised deep learning in which the previous step's output is used as input for the next step. For sequential data, the RNN deep learning technique is ideal. Image captioning, time-series analysis, natural-language processing, handwriting recognition, and machine translation are among applications where RNN is particularly useful. The hidden state, which memorizes certain information about a sequence, is the most important element of RNN.

The RNN process is broken down into four phases.

- The concealed state's output at time t-1 is fed into the input at time t.
- The output at time t is fed into the input at time t+1 in the same way.
- RNN can handle inputs of any substantial length.

- The RNN computation is based on past sequence data, and the model size does not scale with the amount of the input

By converting independent activations into dependents, the RNN reduces the complexity of raising parameters and understanding each prior output by feeding each output into the next hidden layer. [26] A convolutional neural network is a type of artificial neural network used to evaluate visual images in deep learning. The CNN architecture is a common deep learning architecture for a variety of computer vision applications, including image identification. It's a multilayer network that's modeled after the minimal vision system (visual cortex). In this thesis, CNN is used to identify fungal diseases in barley crops, and the detail is found in Section 2.5.

2.4.1 The inner workings of deep learning algorithms

Deep learning algorithms rely on neural networks, much as the human brain uses millions of neurons to process information. Deep learning executes operations based on the three levels below:

Input layer: The input layer contains input data from a dataset that are familiar with.

Hidden Layer: Just as we need to teach the brain via hidden neurons, we need to train the brain through hidden layers.

Output layer: the value that we want to classify. We take an observational feature and put it into a layer. That layer generates an output, which is then used as an input by the following layer, the hidden layer. This continues till the final output is obtained. The researcher separates the network further and add a lot of hidden layers depends on the complexity of the problem and connect everything just like the human brain interconnect everything and that’s how input values are processed through all hidden layers and then have output. That’s why this learning process is known as deep learning, because of the number of additional “Layers” we add to learn from the data. If we do not know it already, when a deep learning model is learning, it is simply updating the weights through an optimization function. A Layer is an intermediate row of so-called Neurons. [26] The deep learning categories are illustrated in Fig. 2.7.

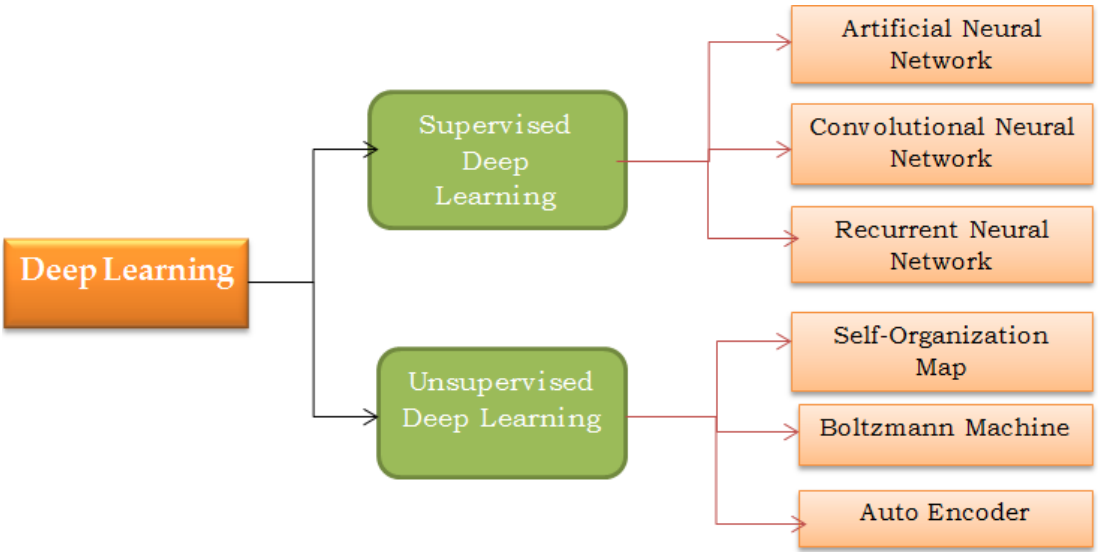


Figure 2.6: Category of deep learning algorithm

2.5 Convolutional Neural Network (CNN)

A Convolutional Neural Network (CNN) is a Deep Learning system that can take an input image, give relevance (learnable weights and biases) to various aspects/objects in the image, and distinguish between them. When compared to other classification methods, the amount of preprocessing required by a CNN is significantly less. Filters are hand-crafted in rudimentary approaches. CNN has the capacity to learn these filters/characteristics with adequate training. When it comes to complicated images with pixel dependencies throughout, the approach may display an average precision score when doing class prediction, but it will have little to no accuracy when it comes to incredibly basic binary images. Through the use of relevant filters, a CNN can successfully capture the spatial and temporal dependencies in an image. Because of the reduced number of parameters and the reusability of weights, the design provides a better fit to the image collection. In other words, the network may be trained to better recognize the image's complexity [24].

The architecture of CNN is illustrated in the following figure (Fig 2.8) [25].

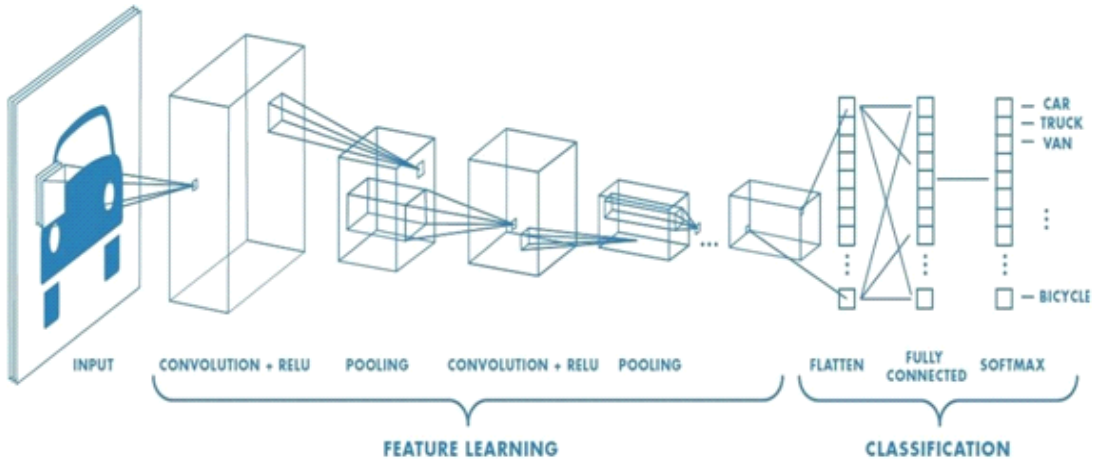


Figure 2.7: Example of CNN Architecture

This study uses images of infected and healthy barley crops as input, CNN is used to identify fungal disease in barley crops. CNN is employed in the classification process, and it is made up of several successive layers, each of which translates one volume of activation to another using distinct functions. [23]

2.5.1 Components of Convolutional Neural Networks

Convolutional neural networks are distinguished from other neural networks by their superior performance with image, speech, or audio signal inputs. They have three main layers. [23]

Convolutional Layer

The main building component of a CNN is the convolutional layer, which is where the majority of the computation takes place. It requires input data, a filter, and a feature map, among several other things. Let's suppose the input is a color image, which is made up of a 3D matrix of pixels. This indicates the input will have three dimensions: height, width, and depth, which match to the RGB color space of an image. A feature detector, also known as a kernel or a filter, will proceed over the image's receptive fields, checking for the presence of the feature. A "convolution" is the term for this process. The feature detector is a two-dimensional (2-D) weighted array that represents a part of the image. The filter size, which can vary in size, is usually a 3x3 matrix, which also determines the size of the receptive field. After that, the filter is applied to a part of an image, and a dot product between the input pixels and the filter is calculated. After that, the dot product is fed

into an output array. The filter then moves by a stride, and the procedure is repeated until the kernel has swept across the whole image. A feature map, activation map, or convolved feature is the ultimate result of a sequence of dot products from the input and the filter. Each output value in the feature map does not have to be connected to each pixel value in the input image, as seen in figure 2.1. It simply has to be connected to the receptive field, which is where the filter is applied. Convolution (and pooling) layers are often referred to as "partially connected" layers since the output array does not have to map directly to each input value. This property, however, can also be referred to as local connection. The feature detector's weights remain constant as it travels over the image, a technique known as parameter sharing. During training, some parameters, such as weight values, are adjusted via back propagation and gradient descent. [23]

However, there are three hyper parameters which affect the volume size of the output that need to be set before the training of the neural network begins. These include:

The number of filters affects the depth of the output. For example, three distinct filters would yield three different feature maps, creating a depth of three.

Stride is the distance, in pixels, that the kernel moves over the input matrix.

Zero-padding is when the filters don't suit the input image, it's frequently used. All elements outside of the input matrix are set to zero, resulting in a larger or equal-sized output. Padding comes in three varieties:

Valid padding: This is also known as no padding. In this case, the last convolution is dropped if dimensions do not align.

Same padding: This padding ensures that the output layer has the same size as the input layer

Full padding: This type of padding increases the size of the output by adding zeros to the border of the input.

A CNN adds a Rectified Linear Unit (ReLU) transformation to the feature map after each convolution operation, adding nonlinearity to the model. Finally, the convolutional layer turns an image into numerical values, which the neural network can analyze and extract meaningful patterns from. [23]

Pooling Layer

Down sampling, also known as pooling layers, which is a dimensionality reduction technique that reduces the number of factors in the input. The pooling process sweeps a filter across the whole

input, similar to the convolutional layer, however this filter does not contain any weights. Instead, the kernel uses an aggregation function to generate the output array from the values in the receptive field. Pooling may be divided into two categories:

- **Max pooling:** The filter selects the pixel with the highest value to transmit to the output array as it moves across the input. In comparison to average pooling, this method is used more frequently.
- **Average pooling:** The average value inside the receptive field is calculated as the filter passes over the input and sent to the output array. While the pooling layer loses a lot of information, it would provide a few benefits for the CNN. They assist in reducing complexity, increasing efficiency, and reducing the chance of over fitting. [23]

Fully-Connected Layer

The fully-connected layer's name is self-explanatory. In partially linked layers, the pixel values of the input image are not directly connected to the output layer, as previously stated. Each node in the output layer, on the other hand, links directly to a node in the previous layer in the fully-connected layer. This layer performs classification tasks based on the features retrieved by the preceding layers and their various filters. While convolutional and pooling layers often utilize ReLU functions to classify inputs, FC layers typically use a sigmoid activation function to provide a probability from 0 to 1. [23].

The fully-connected layer is illustrated in the following figure (Fig. 2.9) [34].

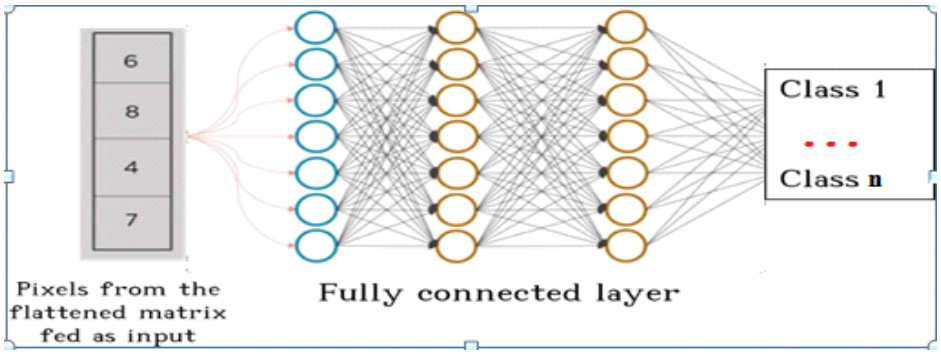


Figure 2.8: Example of Fully Connected layer[34]

2.6 Application of Convolutional Neural Network in crop disease Detection

Deep CNN has been found to be important in agriculture for detecting and classifying crop diseases using images of crops. Because of its great accuracy, CNNs are used for image classification and identification. The CNN uses a hierarchical model that starts by constructing a network, similar to

a funnel, and then outputs a fully-connected layer in which all neurons are connected to one another and the output is processed. The researcher next looks at some literature that shows how CNN architectures could be used to diagnose crop diseases. In [40], a deep CNN technique is used for crop disease classification using CNN and deep learning. The author developed a two-step deep learning-based comparison evaluation for crop disease classification. To begin, the best convolutional neural network (CNN) was identified through a comparison of well-known CNN designs, as well as modified and concatenated versions of several of the proposed DL models. Second, various deep learning optimization techniques were used to try to increase the performance of the best-obtained model. The author compared multiple CNNs using performance criteria such as validation accuracy/loss, F1-score, and the number of epochs required. The Crop Village dataset was used to train all of the selected DL architectures. The architecture trained with the convolutional neural network has the highest validation accuracy and F1-score of 99.81 percent and 0.9978, respectively, according to the author.

Another study on automated identification of crop diseases with limited data is also presented in [36]. As the authors presented these approaches require the collection and annotation of many images, which is difficult and costly process especially in the case of new or rare diseases. For this case, in this study, the authors developed and evaluated several methods for identifying crop diseases with little data. Convolutional Neural Networks (CNNs) are used due to their superior ability to transfer learning.

The authors of [39] used a Convolutional neural network to construct a mobile-based Cassava disease detection model (CNN). Where the usual technique is visual detection, which requires specific training, the Author models have the potential to enhance crop disease phenotyping. Models face new challenges owing to illumination and orientation in cases where a CNN is implemented on mobile devices. Model evaluation be undertaken in real-world situations, according to the authors, if such models are to be reliably integrated with computer vision products for crop disease morphological characteristics. To detect foliar indications of diseases in cassava, the authors train a CNN object detection model (Manihot esculenta Crantz). Finally, the authors built a mobile app to evaluate the model's performance on mobile images and video of 720 infected leaflets in a Tanzanian agricultural field.

2.7 Related works

This section reviews on how researchers of the past have made use of several standard algorithms in detection of crop disease. In general, many researches have been done for crop disease detection using image processing and machine learning approaches. However, the majorities of research in crop disease identification use traditional image processing approaches and follow a consistent sequence, which includes image capture, image preprocessing, image feature extraction, and ultimately classification [30, 31, and 33]. As a result, a number of researchers have proposed and/or created systems for detecting and classifying crop diseases using various methodologies; among them, the literature on machine learning and deep learning techniques are reviewed as follows.

An automated tool is presented in [28] to identify and classify banana leaf disease. They try to identify and classify disease caused by fungi and the diseases are known as banana Sigatoka and banana Speckle. In this study, a globally available dataset from the Crop Village project is used to collect the infected and healthy leaf images of banana. The leaves infected by the disease are determined based on the color difference between the healthy and the infected leaves. The authors perform preprocessing for the entire dataset by resizing each image to 60 × 60 pixels and by converting the images to gray scale. The authors perform feature extraction and classification by applying CNN algorithm. The trained model had 97.3% classification accuracy.

[39] Describes ANNs for banana crop classification and grading. A total of 35 diseased images of banana leaves captured in the field were used to train the NN in this experiment. By converting RGB images the color feature and Histogram of Template (HOT) features were extracted. A neural network is trained for classification feed-forward. The trained model correctly diagnoses five different banana crop diseases based on the provided images, as the authors mention in his study.

[38] Presents another article titled Classification of wheat leaf Septoria disease using image processing and machine learning techniques. The authors of this work employed image processing and machine learning extensively in several disease diagnostic methodologies. It has been used on images acquired by visible light cameras as well as images captured by technology that gathers information in an invisible wavelength, supporting professionals in selecting the appropriate measure and treatment. In this study, a digital camera image is used as the input, which is then enhanced using different preprocessing approaches, followed by a color-based segmentation method to separate the regions of interest, and then features are extracted using the Gray Level Co-occurrence Matrix. The authors used four different supervised learning algorithms to classify

the input image into two separate classes: "healthy" and "infected." In this study, the following models were used for comparison: Nave Bayes, k-Nearest Neighbor, Support Vector Machines, and Random Forest. Finally, the researcher indicated that his study with Random Forest model had a classification accuracy of 98.7%.

[39] Also uses a deep learning method to develop a bacterial wilt detection model on wheat crop. The created model was trained and evaluated using the gathered dataset, and the designed model was compared to several pre-trained convolutional neural network models, such as MOBILENET and InceptionV3. The collection comprises 4896 wheat images, both healthy and diseased. According to the author, 80 percent of the images are used for training and the remainder for testing the model. Data augmentation is performed during training to create more images that fit the proposed model. The results of the experiments show that the proposed approach is successful in detecting Wheat bacterial wilt infection. The suggested model for this investigation, as provided by the authors, was successfully classified the given image with the accuracy of 98.5%.
[36] Also presents another study on the automated detection of crop diseases with limited data. As stated by the authors, these methodologies need the collecting and annotation of numerous images, which is a time-consuming and expensive operation, particularly in the case of new or rare diseases. In this circumstance, the authors constructed and tested multiple approaches for diagnosing crop diseases with little data in this study. The authors used Convolutional Neural Networks (CNNs) for this study. Two baseline models, a Triplet network and a deep adversarial Metric Learning (DAML) method, were built using three CNN architectures (ResNet18, ResNet34, and ResNet50). These methods were trained on a huge source domain dataset and then fine-tuned to identify new diseases from a small number of images (between 5 and 50 per disease).
[37] Presents Detection of Rice Crop Diseases Using Convolutional Neural Network (CNN). According to this research, the CNN approach was used to identify infections in rice crops. In comparison to previous methodologies, this study produced better results. The accuracy for training data was 100 percent, while for testing data, it was 86.67 percent. According to the paper's author, the CNN model was employed in this study to detect three different forms of rice crop infections based on physical images of rice crop leaves: brown spots, leaf smuts, and bacterial leaf blight disease.

[41] Presents Leaf and spike wheat disease detection & classification using an improved deep convolutional architecture. This paper proposes a new method for classifying wheat diseases. A sophisticated deep learning model has been taught to reliably identify 10 types of wheat diseases. The proposed method has a 97.88 % testing accuracy. Furthermore, it outperforms the other two popular deep learning models – MOBILENET and RESNET50 – in terms of accuracy by 7.01 percent and 15.92 percent, respectively. The proposed method improves other criteria such as precision, recall, and f-score, according to the findings of the experiments.

[45] Presents Wheat yellow rust disease infection type classification using texture features. According to the researcher developing a method for processing color images of wheat spikes to effectively detect diseased regions using deep learning and image processing techniques to produce a reliable and cost-effective high output system for evaluating FHB in the field, according to the study. In a shadow situation, color images of wheat spikes at the milk stage were gathered and processed to create datasets, which were then used to retrain a deep convolutional neural network model via transfer learning. According to the study the coefficient of determination for the number of spikes counted by manual count and the model was 0.80, indicating that the model recognized spikes in the images quite well. The model was assessed, and the mean average precision for the testing dataset was 0.9201. On the basis of the results for spike detection, a new color feature was applied to obtain the gray image of each spike and a modified region-growing algorithm was implemented to segment and detect the diseased areas of each spike.

2.8 Proposed work

As mentioned in the previous sections, studies show that computer vision with a deep learning approach has been widely used in agriculture, particularly for crop disease identification, and has yielded interesting results, but that using deep learning to detect fungal disease in barley crops has not been well researched. As a result, the planned study was carried out using the steps below, which are not commonly used to identify barley crop diseases. The acquisition of images will be the initial phase. The images are obtained from barley dataset of Kulumsa research center and the images are captured by digital camera and Smartphones from the field. Preprocessing is the second phase, and its major objective is to enhance image data by eliminating undesired features, improving the image, and segmenting it. Using multiple segmentation methods such as thresholding, image segmentation is used to detect the image's boundaries. When the researcher

gets to the feature extraction step, extract several useful features from the image for disease detection, such as color and pattern. The fifth and most important phase is disease identification.

2.9 Research gap

Crop disease detection and classification utilizing image processing techniques and deep learning algorithms such as CNN is an active research topic with promising findings. Several crop disease detection studies have been undertaken in recent years utilizing machine learning and deep learning methodologies such as CNN models. In Ethiopia, barley is one of the major foods and factory crops that have been affected by a fungal disease. Despite the fact that numerous works have been established for the diagnosis and classification of various crop diseases, as far as the knowledge of the researcher is concerned no deep learning models have been developed to identify the most affecting barley disease, which is fungus. For this reason, an efficient CNN-based model is proposed to detect fungal diseases of leaf barley crops in this study.

CHAPTER THREE

RESEARCH METHODOLOGY

This chapter describes the approaches that were used to complete this thesis, including methods for implementing the model, data collecting, data preparation, software and hardware configuration, and techniques for evaluating the model. The full process of using deep learning to construct a model for detecting fungal disease in barley crops is detailed. In the subsections below, the entire procedure is broken down into multiple required phases, beginning with acquiring images for the identification process using a deep convolutional neural network (CNN).

3.1. Research flow

The research approach used in this thesis is experimental. The following phases of process flow are followed in order to achieve the goal of this thesis. The first phase entails determining the topic's domain, which includes examining several types of literature in order to fully understand the problem. The thesis general and specific objectives are established. The data preparation and thesis design are the topics of the second phase. Data is gathered from Kulumsa research center, labeled by agricultural professionals, and then classified into training, validation, and testing throughout the data preparation process. The model is created after the data has been prepared. The thesis is implemented in the third step, during which the planned model is implemented using appropriate tools and methodologies. With the appropriate data, the designed model is trained and tested. The performance of the model is evaluated while it is being trained. The model is evaluated with test data once it has been determined to be the best during evaluation. Finally, the model is compared to other models which have already trained.

The flow of this research is also shown in the figure below (Fig. 3.1).

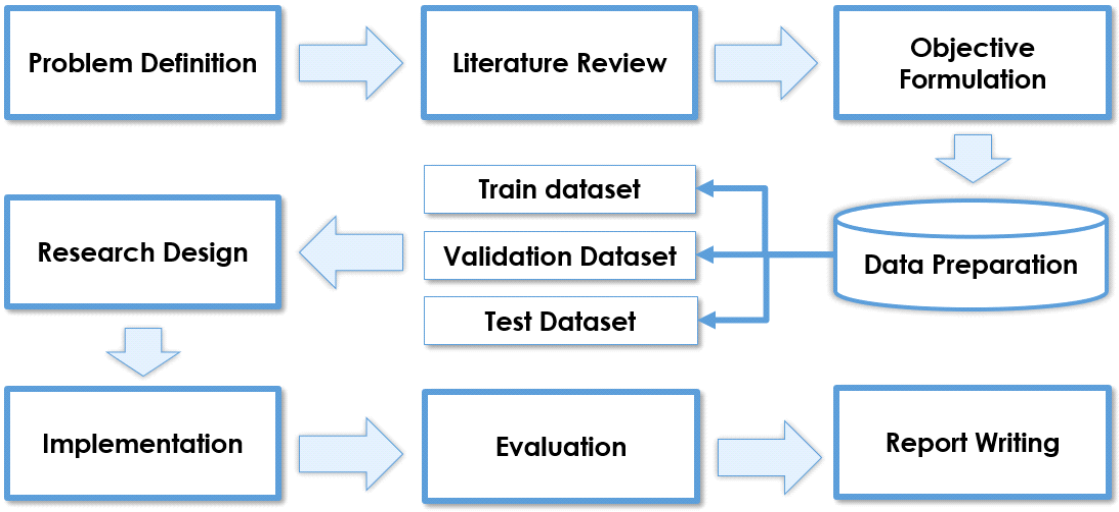


Figure 3.1: Research flow

3.2 Experimental Process

3.2.1 Data source and preparation

The most challenging task for the researcher is getting the data that is used to train the neural network model. Data preparation is needed to train and test the model. Barley crop leaf image data are used as the main input to the model. Barley leaf image data is collected from the Arsi Zone Institute of Agricultural Research (IAR) of Kulumsa Research Center (KRC).in addition images from different barley farms that are healthy and infected are also used for this thesis. So totally 10224 images are used. This can benefit the model by training it with different imaging properties and conditions.

3.2.2 Preprocessing image in the dataset

The features of a region contained in an image are extracted and recognized by a convolutional deep neural network, which then classifies the image based on the features. However, because the properties of the image data are not clearly differentiated and irregularities are present in the raw image data, the acquired raw image data does not guarantee effective model training in the manual image preprocessing approach. However, in the case of deep learning using the CNN method,

there is no need for further preprocessing on the dataset because the computer can learn the features from the raw pixels of the image.

However, the images in the prepared dataset are of varying sizes. Therefore, a size normalization operation is conducted on the dataset in order to achieve a comparable size of all images for the CNN technique and to lower the computational time of training because the model is trained on a typical PC with restricted hardware resources such as CPU, and memory. Finally, all of the dataset images are resized to 150 * 150 pixels.

3.2.3 Image Data Augmentation

Data augmentation is a method of artificially creating new training data from existing data. This is accomplished by using domain-specific approaches to transform examples from the training data into new and unique training examples. Sometimes, the amount of data available to the researcher is insufficient to adequately accomplish the classification task. In some circumstances, the researcher augments the data. In image-based deep learning tasks, augmentation is frequently employed to improve the amount and variability of training data. Only the training set should be augmented; the validation set should never be augmented. In addition, an efficient and convenient dataset, model, and training run may be used to test data augmentation techniques separately and together to evaluate if they result in a significant difference in model performance. [27]

Modern deep learning algorithms, such as the convolutional neural network (CNN), can learn patterns that are independent of where they appear in the image. However, augmentation can help with this transform-invariant method of learning by supporting the model in learning features that are also transform-invariant, such as left-to-right to top-to-bottom ordering, light levels in images, and so on. Various data augmentation techniques have been used on the original images in this thesis to obtain additional images for the existing data set. The researcher can either augment the data before feeding it into a model (offline augmentation) or during the training process. The types of data augmentation are represented in the following figure (Fig 3.2) [28].

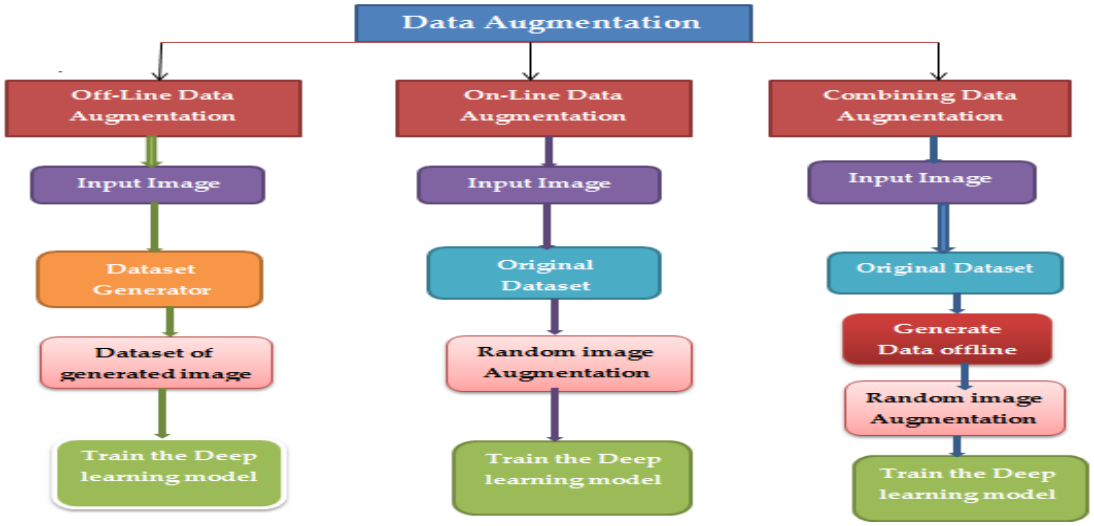


Figure 3.2: Types of data Augmentation [28]

3.2.3.1 Offline data augmentation

The researcher can save the augmented images on the disk using dataset generation or offline data augmentation. The approaches that may be used to execute online data augmentation can also be used to accomplish offline data augmentation, which is when the images are stored on a hard drive. After applying data augmentation techniques to each and every training image, augmented images are produced. As a result, the dataset becomes more varied and the model becomes more stable. This method may be used to increase the dataset's image quantity. [28]

3.3 Material and Tools

The researcher selects the appropriate software and hardware tools before starting this thesis. As a result, the researcher's chosen hardware and software tools are given below.

3.3.1 Software tools

Tensor Flow: Tensor Flow is an open-source machine learning platform that runs from beginning to end. It has a large, flexible range of tools, libraries, and community resources that allow researchers to advance the state-of-the-art in machine learning and developers to quickly construct and deploy ML applications. Tensor Flow comes with a Tensor Flow-specific version of the Keras API (in the keras module). It can be used on any desktop running Windows, Mac OS, or Linux, as

well as in the cloud as a service and on mobile platforms such as iOS and Android. Tensor Flow's design allows for data preprocessing, model creation, model training, and model evaluation. [27]

Keras:- Keras is a Python-based high-level neural network API that may be used with TensorFlow, CNTK, or Theano. It was created with the goal of allowing for rapid experimentation. A cornerstone of good research is being able to get from concept to outcome as quickly as possible. It supports CNN and RNN, as well as a mixture of the two [27], and enables easy implementation.

Anaconda: - Anaconda is a Python and R programming language package aimed at simplifying package management and deployment in scientific computing (data science, machine learning applications, large-scale data processing, predictive analytics, and so on). Data-science packages are included in the distribution, which are compatible with Windows, Linux, and Mac OS X. Anaconda, Inc., created by Peter Wang and Travis Oliphant in 2012 develops and maintains it. Anaconda Distribution or Anaconda Individual Edition is other Anaconda, Inc. products, whereas Anaconda Team Edition and Anaconda Enterprise Edition, both of which are not free, are other Anaconda, Inc. products. [27]

Visio The system architecture was designed with 2019[6]. With ready-made templates, this application was used to rapidly generate, collaborate, and share data-linked diagrams, aiding in the simplification of complicated information.

3.3.2 Hardware tools

The installation of a 64-bit Windows 10 operating system on a laptop with 8 GB of RAM, a 1 TB hard disk, and an Intel® CoreTM i5-5200U CPU (Central Processing Unit) processor, followed by the installation of the Anaconda IDE (Integrated Development Environment) and all necessary libraries, including Python 3.9, was the first step in turning the tide of this thesis. Raw color images of both healthy and diseased barley leaves were collected from research center and the others are captured by a smart phone (Infinix Hot10 with 13Mps) under normal conditions, that is, without taking into account the light sensor selection or the image's relative position to the camera.

3.4 Evaluation Metrics to Evaluate the Accuracy of Model

In general, there are several types of evaluation methods: The primary evaluation metrics are classification accuracy, logarithmic loss, confusion matrix, area under curve, F1 Score, mean absolute error, and mean squared error. When evaluated using a measure, the model may produce satisfying performance. When compared to alternative measures such as logarithmic loss or any other comparable metric, classification accuracy measurements may yield poor results [31]. Additionally, classification accuracy measures are employed, which is a recommended technique for classification problems with the same number of samples in all classes in the dataset. The number of correct predictions divided by the total number of input samples is the ratio. Taking all of this into account, the researcher used classification accuracy measures to assess the efficiency of the model.

Finally, the trained model is given the testing data (images that were not used in either the training or validation sets) to evaluate its performance and the model provides accuracy and loss of the testing data, which was never seen during the training.

CHAPTER FOUR

MODEL DESIGN AND EXPERIMENTAL PROCEDURE

This chapter explains how to implement the proposed model for fungal disease in barley crops into practice, as well as how to design it. This chapter focuses on core tasks including pre-trained models using different mechanisms, proposed model architecture, descriptions and classification in the proposed model.

4.1 Model Selection

Based on different literature in computer vision, namely in image classification, a supervised deep learning technique named CNN was chosen for this thesis. Because of its higher accuracy than other algorithms, CNN is employed for image classification and recognition. CNN uses a hierarchical model that builds a network similar to a funnel and then generates a fully-connected layer in which all neurons are linked to each other and an output is evaluated. Unlike the other deep learning algorithms, CNN discovers important features without the need for human intervention. Given a large number of classes of images, it can learn the key features of each class on its own [42].

Aside from the benefits listed above, the researcher chose CNN for the following reasons:

- CNN is simple to understand and use, and it has the highest accuracy of all image prediction algorithms.
- Previous research has shown that the CNN model converges faster than the machine learning (ML) model in terms of epochs.
- Before classification and prediction, most conventional machine learning algorithms require explicit extraction of the features used to study the image.
- Convolutional Neural Network (CNN) is used for any type of prediction problem, including those involving image data as input.

4.2 Experimental setup

The experimental setup is performed in the following order: First, all of the essential parameters and hyper parameters are configured. Second, the proposed model is trained to extract the most significant features from the dataset of barley images Third, the model classifies images of barley crops into two categories: those infected by fungal disease and healthy. Fourth, several evaluation measures such as classification accuracy with loss, training accuracy with loss, validation accuracy with loss, confusion matrix, precision, recall, and support are used to evaluate the model. Finally, the developed model is put to the test on a test dataset.

4.2.1 Input layer

The proposed model's input layer takes RGB images. since the input images are the color images, the shape of the input layer is configured as 150 ×150 × (N = 3) where N defined as a total number of channels. This layer simply sends the data to the first convolution layer and does not do any calculations. As a result, this layer does nothing but transmit the input to the first convolution layer. That is, there are no learnable features in this layer, and its parameter count is 0.

4.2.2 Convolution layer

The layers where filters are applied to the raw image are known as "convolutional layers." The proposed model, in this thesis includes a number of layers that transform conventional image sizes into a feature set for subsequent processing. To decrease the dimension and extract features from the barley images in the dataset, the proposed model uses four convolutional layers, each of which is coupled to four max-pooling layers.

The first convolution layer of the model accepts input images of size 150x150x3 using 16 filters with a size of 3x3x16, a step size of 1 pixel, and a fill value of 0. This process is followed by a Rectified Linear Units (ReLU) activation function that replaces all negative pixel values with 0 to introduce non-linearity into the network. The second convolution layer takes the output of the first convolution layer as input and filters it using 32 filters of size 3x3x32. The third convolution layer takes the output of the second convolution layer as input and filters it using 64 kernels of size 3x3x64. The fourth convolution layer takes the output of the third convolution layer as input and filters it using 32 kernels of size 3x3x32.

4.2.3 Pooling layer

There are four max-pooling layers with a kernel size of 2 × 2 × 3. All four max-pooling layers reduce the output of all four convolutional layers with a filter size of 2×2 with a stride value of 1. All pooling layers have no learnable features and they only perform down-sampling operation along the spatial dimension of the input volume, hence the number of parameters inthese layers is 0.

Layers	Filter size	Depth	Stride	Output Shape	# Parameters
Input	-	-	…	(150 ×150 × 3)	0
Conv1 + ReLU	3 × 3	16	1	(150,150,16)	448
maxPool 1	2 × 2	-	…	(75,75,16)	0
conv2 + ReLU	3 × 3	32	1	(37,37,32)	4640
maxPool 2	2 × 2	-	…	(37,37,32)	0
Conv3 + ReLU	3 × 3	64	1	(37,37,64)	18,496
maxPool 3	2 × 2	-	…	(18,18,64)	0
Conv4 + ReLU	3 × 3	32	1	(18,18,32)	18464
maxPool 4	2 × 2	-	…	(9,9,32)	0

4.2.4 Fully connected layer

This layer connects the information extracted from the previous steps (that is, the convolution layer and the pooling layer) to the output layer and finally classifies the input into the desired label. In the proposed model, there are three fully connected layers.

The first fully connected layer contains a total of 256 neurons, followed by the ReLU activation function, which accepts the output of the fourth maximum pooling layer. The second fully connected layer contains 100 neurons connected to all neurons in the first fully connected layer and the last fully connected layer, followed by the ReLU and value. The 0.2 dropout operation follows. The third fully connected layer is the last layer of the proposed layer.

The number of parameters in the CONV layer is as follows: ((m * n * d) + 1) * k), 1 is added for the bias term for each filter. The same equation can be written as: ((filter width shape * filter height shape * number of filters at previous level + 1) * number of filters).

4.2.5 Output layer

The output layer is the last of the model (4th FC layer) and has one neuron with a sigmoid activation function. That is because the model is designed to classify two classes, i.e., classification of input healthy and diseased images, called binary classification. A Sigmoid activation function is implemented to sum the output values equal to 1.0 and clip the individual outputs to a value between 0 and 1. The output of each fully connected layer is calculated according to equation below. [34]

$$Class\ output = w * x + b \quad \text{(4.1)}$$

*Where **w** is weighted and **x** is the values of the features extracted by the convolutional layer and **b** is bias.*

The following table (Table 4.1) shows the composition of the proposed classification phases

Table 4.1: *Composition of the proposed classification phases*

Layers	Output Shape	# Parameter
Flatten	2592	0
FC1 + ReLU	512	1327616
Dropout	512	0
FC2 + ReLU	100	51300
Dropout	100	0
FC3+ Sigmoid	2	404

4.3 Feature extraction phase of proposed model

The proposed model includes four convolutional layers (16, 32, 64, and 32, respectively), four rectified linear unit function (ReLU) max-pooling operations, and two fully connected layers and two output nodes and the last layer as two classification layers. Dropouts are included after being included in the fourth max-pooling layer to avoid over-fitting issues. The figure below (Figure 4.1) shows the feature extraction phase of the proposed model.

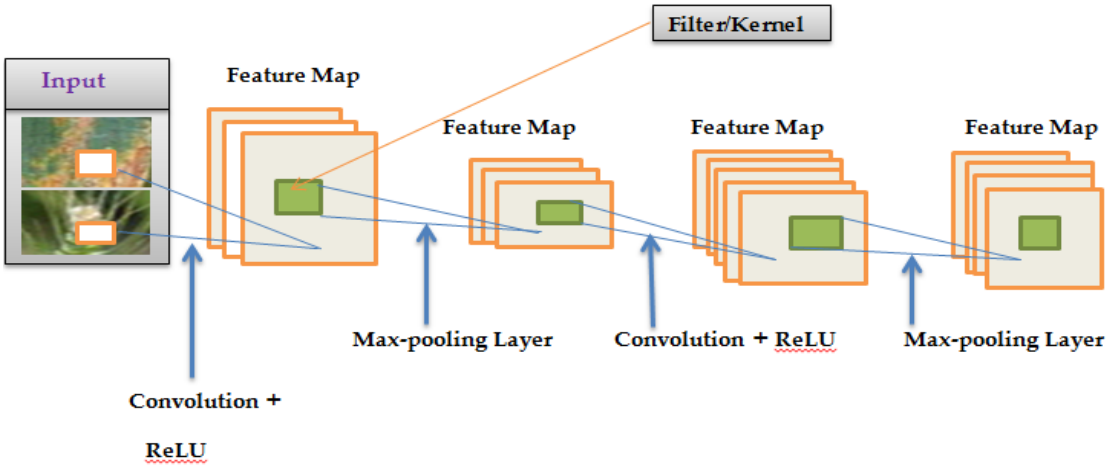


Figure 4.1: Feature Extraction Phase of the proposed model

Winners of the ILSVRC 2015 Image Classification Challenge have introduced a new concept called the Deep Residual Learning Framework. This allows to train hundreds of layers for compelling performance without worrying about performance saturation [44]. A training error for an entire module with an additional layer of ID is the same as an error without an additional layer. Therefore, the additional identity mapping layer is only used to learn features in addition to the inputs already available. You can use the input image size (W1), receptive field size (F), step size (S), and zero fill amount (P) to calculate the spatial size of the output volume for each slice [34]. The following equation shows the exact starting volume size for all layers in the proposed model.

$$Output\ size\ (W2) = \frac{(W1-F+2P)}{S} + 1 \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ (4.2)$$

*Where: **W1** is the size of the input volume, **F** is filter size, **P** is the number of zero paddings, and **S** is the stride.*

4.5 Classification layer using proposed model

Classification allows the model to analyze the input image and identify the class to which the input image applies (probability that the input image is part of the class). In the proposed model, there are a total of three fully connected layers, including the output layer. The main function of these layers is to classify the input images based on the key extracted features performed by the convolution layer. The first fully connected layer accepts the final output of the convolution layer 4, and then feeds the feature map to the second fully connected layer.

The figure below (Figure 4.2) shows the configuration of the proposed model classification phase.

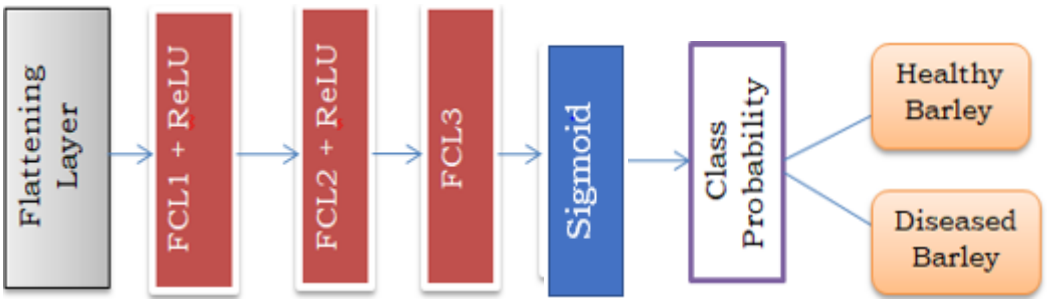


Figure 4.2: Configuration of the proposed Classification phase

4.6 Augmentation Parameters

Data augmentation is used in deep learning when we have limited data to train the model i.e., to enlarge dataset. The dataset available for this research is not sufficient to train the model, so additional input images are generated by different augmentation mechanisms described in table 4.2.

Table 4.2*: Data Augmentation techniques*

Augmentation parameter	***Augmentation Factor***
Horizontal Flip	*1(True)*
Shear Range	*0.3*
Width Shift Range	*0.2*
Height Shift Range	*0.2*
Zoom Range	*0.2*

4.7 Hyper Parameters configuration

Hyper-parameters are parameters whose values control the learning process and determine the values of model parameters that the learning algorithm ultimately learns. Hyper-parameters are used by the learning algorithm during training, but are not part of the resulting model.

The hyper-parameters selected for the model are described below.

Table 4.3*: Hyper Parameters configuration*

Parameters	***Configuration values***
Learning rate	0.001
Batch size	64
Epochs	30
Optimization algorithm	Adam
Activation function	ReLU and Sigmoid
Loss Function	Binary Cross-Entropy

4.8 Classification using Proposed Model

Classification is the process of extracting the most important features or representative pixels from an image. In the proposed model, image datasets of healthy and diseased barley are trained and pixels are taught directly to the network without manual intervention. As we can see in the figure above (Figure 4.1), there are a total of three fully connected layers, including the output layer. Classification layers are fully connected layers that provide classes that models can classify. The proposed model has a probability for each class of presented images. The final FC layer of the proposed model typically uses the sigmoid function to specify the classification. Then, after classifying the input images into disease and health based on the learning category of a particular class, the proposed model completed the detection of fungal diseases in barley crops. Then, finally, the test phase is also performed by giving the proposed trained model with an unseen barley image dataset.

4.9 Experiment with different Pre-Trained Models.

A persistence model is a form of CNN model that has been created by millions of people to handle similar issues. These models' fundamental assumptions are reasonable. Transfer information to a smaller record using models built for large datasets. Because these models are made up of millions of image functions and thousands of classes, they may be used to generalize image objects. Thus, even if the extracted features and classes differed from the original object, they were acquired from the preceding model [17, 19, 29, 31, 34]. Transfer channels are technologies that apply these concepts to real-world issues. Training these pre-trained models from scratch is computationally difficult and sometimes impossible. Two important components were included in majority of pre-trained models. The convolutional part (feature extraction component) includes the convolutional and pooling layers for extracting essential features from input images, as well as the classification phase, which calculates the input images to categorize based on the extracted features. Two pre-trained models were used for this job: Inception V3 and Mobile Net. They provided weights, and the models were trained using the barley dataset, with the results compared to the proposed model. During the training of these models, the convolution basis is either left unchanged or frozen, enabling just the classification base to be trained. There are three general methodologies for the transfer learning process [19].

The first method involves training the entire convolutional base and then modifying only the fully connected layers depending on a class. A subset of the convolutional base is trained in the second method. Putting all convolution bases to zero and training the classification base is the third strategy. Learning some of the ideas of convolution is the process of fine-tuning.

CHAPTER FIVE

RESULTS AND DISCUSSIONS

The experimental evaluation of the proposed model is the objective of this chapter. The evaluation is focused on the proposed model performance both independently and in comparison, to those of previously developed classical pre-trained models. The section describes how the research was carried out from beginning to end in the development of an effective discuss fungal disease detection model. Finally, experimental results are shown in a number of graphs and tables.

5.1 Experimental outcomes

To classify the input images, several experiments are conducted to determine the proposed model's classification performance. While the latter two trials employ the pre-trained CNN model, the first four experiments employ various techniques. The experimental procedure in this thesis, like every other CNN classification task, contains two basic steps. The first phase is training, and the second phase is testing. During the training phase, the input is passed with the output to the classifier to obtain the required class, and the various features of the image returned by the convolution base are then learned. In order to evaluate the classification model's performance, the test phase compares a trained model to a test or unseen dataset. The results of the tests are described in the next section. During the training phase, the proposed model is also validated using validation sets, confusion metrics, and numerous classification accuracy metrics such as fit, recall, F1 score, and support. The proposed model is based on the most well studied CNN model for crop disease detection, which can be performed on a minimal hardware resource and yields promising results. This model contains a total of seven layers, four of which are convolutional and three of which are fully connected layers. This model provides 150x150x3 pixels of color images. To train the model and get good results, the researcher used the cross-validation approach and the training / test strategy with learning rate of 0.001, activation’s function; ReLU and sigmoid, batch size; 64, optimization algorithm; Adam, Loss function; Binary cross-entropy and epochs 30. The experiments that were conducted on the suggested model to get better effects are described in detail in the following discussion. The model was trained on a total of 10,224 images, including extended images. The experiments that were conducted on the proposed model to get better effects are described in detail in the following discussion. The experiments that were conducted on the suggested model to get better effects are described in detail in the following discussion.

5.2 Training using experimental strategy 1(Cross validation strategy)

In this section, eight Split was run using cross-validation techniques. Using Python scripts, this technique separates a part of the training dataset from the rest of the validation dataset. The 8-cross test with various trains, as well as the proposed model with various validations Split choices and Split numbers are depicted in Table 5.1. In this experiment, the entire experiments that were conducted on the suggested model to get better effects are described in detail in the following discussion. training experiments that were conducted on the suggested model to get better effects are described in detail in the following discussion. dataset is partitioned into eight different convolutions. 90% of the dataset is used for training, while the remaining 10% is used for testing. Each Split make up the validation set, which accounts for 10% of the overall dataset, whereas the 90% make up the rest of the dataset. Colors are used to train datasets that account for 90% of the total data. Each division (Division 1 to Division 8) was similarly trained in Epoch 30. As a result, we trained the proposed model using a total of 10 x 32(320) epochs. Furthermore, rather than developing a model, each component created the model with the highest level of verification accuracy and then select the most suitable model from a list of eight. As a result of this, a total of eight models were developed (Split 1 to Split 8). That is, after they have been trained, all 8 models will be developed with an epoch value of 30. The next section goes into the outcomes of each of the eight Splits (Split 1-8).

Split 1: experimental results are percent-aged in terms of training, validation, and classification accuracy, as well as training, validation, and classification loss. Table 5.1 illustrates the arrangement.

Table 5.1: Accuracy and Loss of training, validation, and classification test of Split on

Accuracy			**Loss**		
Training	Validation	Classification	Training	Validation	Classification
98.93%	98.03%	98.07%	0.31%	4.54%	0.93%

Figure 5.1 shows the outcomes of Split One proposed model in terms of training and validation accuracy, as well as training and validation loss.

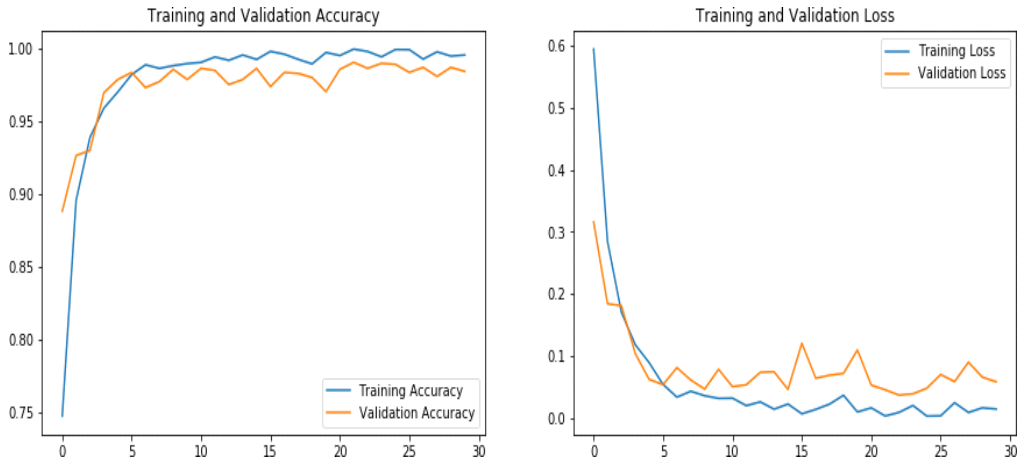


Figure 5.1: Training and validation accuracy and training and validation losses from Split1

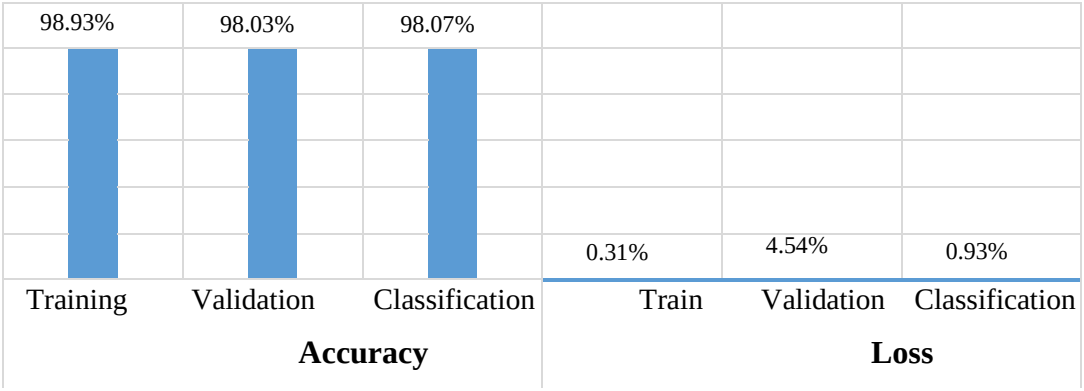


Table 5.2 consider all aspects experimental results for each class in the form of confusion matrices, fit rates, recalls, f1 values, supports, and confusion matrices.

Table 5.2: One-time Split ratio, recall, F1 score and supporting test data

Class	Precision	Recall	F1-Score	Support
Diseased Barley	0.98	0.99	0.98	380
Health Barley	0.99	0.97	0.98	430

Split 2: Table 5.3 shows the total experimental results of the proposed model in Split 2 in terms of training, validation, test accuracy, and training, validation increase. Decide on a percentage of loss.

Table 5.3: Split-Two training, validation, and test accuracy and loss

Accuracy			**Loss**		
Training	Validation	Classification	Training	Validation	Classification
98.96%	98.37%	98.33%	0.13%	2.65%	0.67%

Figure 5.2 depicts the proposed model's outcomes for Split two during training in terms of training versus validation accuracy and training against validation loss.

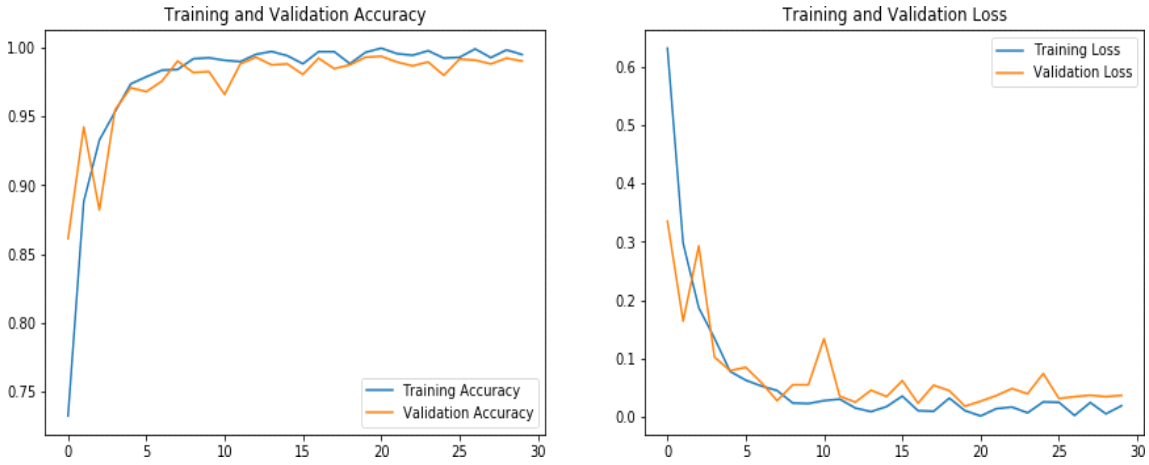


Figure 5.2: Training vs validation accuracy and Training vs validation loss of the Split two

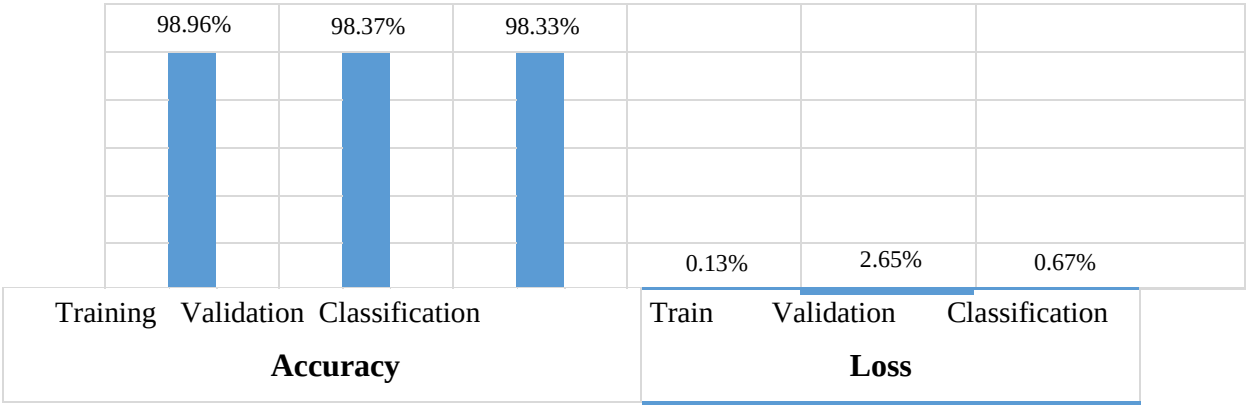


Table 5.4 consider all aspects experimental results for each class in the form of confusion matrices, fit rates, recalls, f1 values, supports, and confusion matrices.

Table 5.4: One-time Split ratio, recall, F1 score and supporting test data

Class	Precision	Recall	F1-Score	Support
Diseased Barley	0.99	0.99	0.99	380
Health Barley	0.99	0.98	0.98	430

Split 3: The total experimental findings obtained from the proposed model for Split three are provided in Table 5.5 in terms of Training, Validation, and Test accuracy, as well as Training, Validation, and Test loss in percentage form.

Table 5.5: Training, Validation, and Testing accuracy and loss of the Split three

Accuracy			Loss		
Training	Validation	Classification	Training	Validation	Classification
98.48%	98.10%	98.07%	1.84%	3.29%	0.93%

Figure 5.3 depicts the proposed model's outcomes for Split two during training in terms of training versus validation accuracy and training against validation loss.

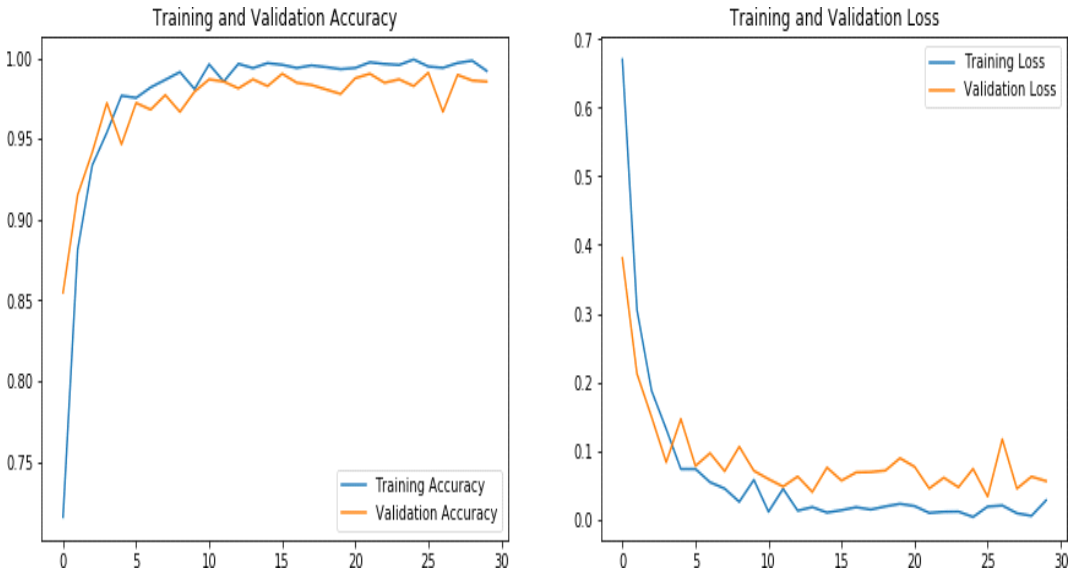
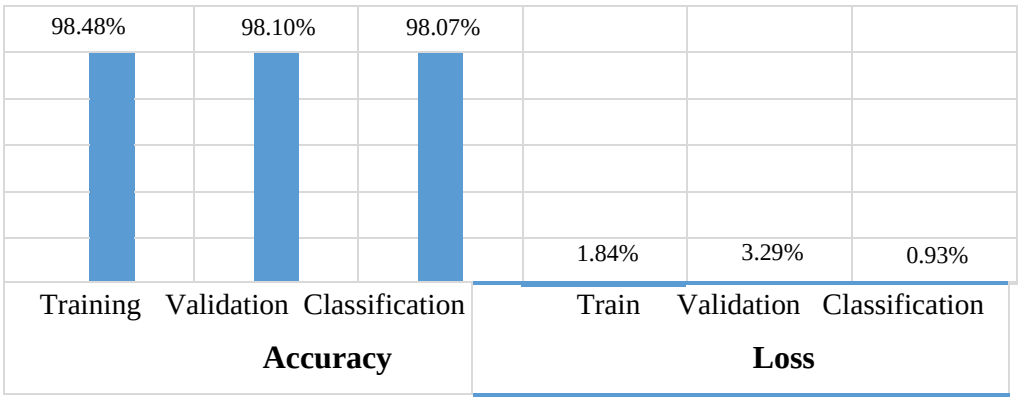


Figure 5.3: Training vs validation accuracy and Training vs validation loss of the Split three.



The detail experimental result of the proposed model is presented in the form of the confusion matrix, precision, recall, f1-scores, support, and confusion matrix for Split three for each class which is illustrated in Table 5.6.

Table 5.6: Precision, Recall, F1-Score, and Support test data of the Split three

Class	Precision	Recall	F1-Score	Support
Diseased Barley	0.98	0.98	0.98	380
Health Barley	0.98	0.98	0.98	430

Split 4: The total experimental findings obtained from the proposed model for Split four are provided in Table 5.7 in terms of Training, Validation, and Test accuracy, as well as Training, Validation, and Test loss in percentage form.

Table 5.7: Training, Validation, and Testing accuracy and loss of the Split four

Accuracy			Loss		
Training	Validation	Classification	Training	Validation	Classification
98.58%	97.75%	97.73%	1.15%	5.21%	1.27%

Figure 5.4 depicts the proposed model's outcomes for Split four during training in terms of training versus validation accuracy and training against validation loss.

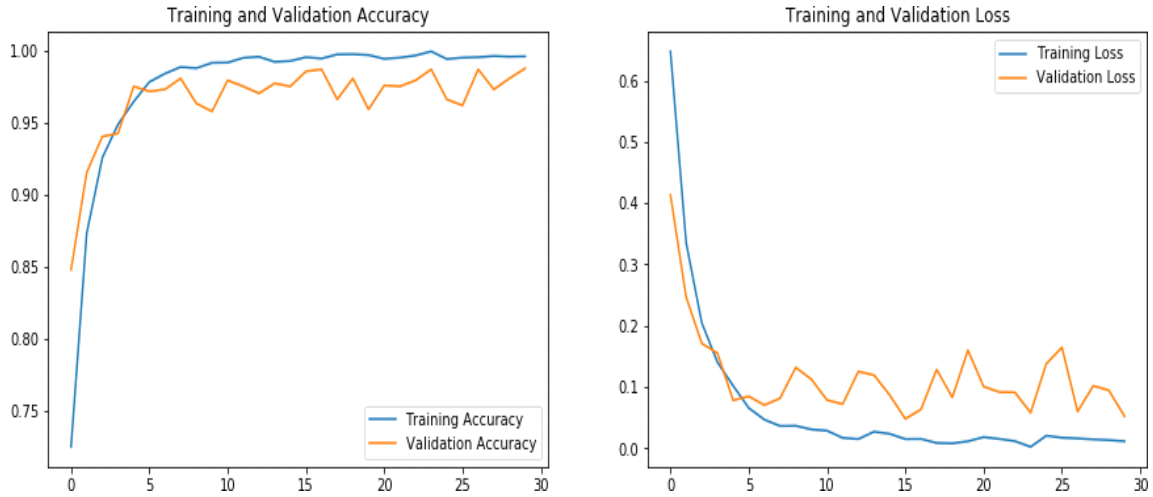
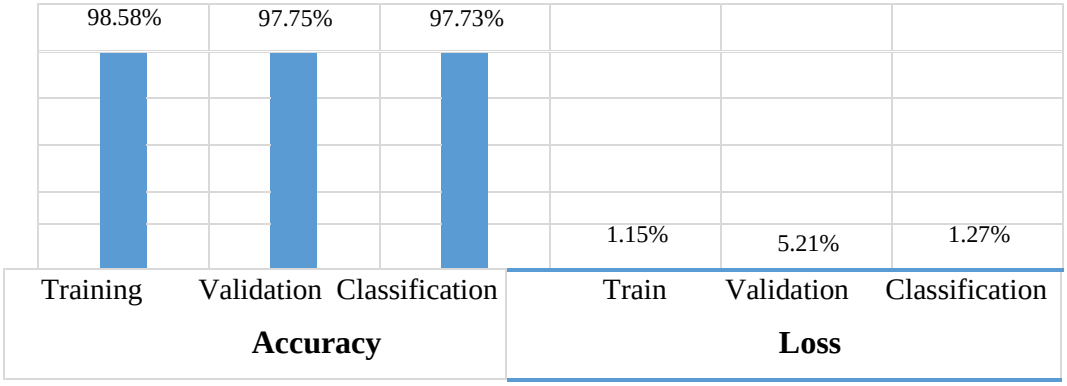


Figure 5.4: Training vs validation accuracy and Training vs validation loss of the Split four.



The detail experimental result of the proposed model is presented in the form of the confusion matrix, precision, recall, f1-scores, support, and confusion matrix for Split four for each class which is illustrated in Table 5.8.

Table 5.8: Precision, Recall, F1-Score, and Support Fold four for test data

Class	Precision	Recall	F1-Score	Support
Diseased Barley	0.98	0.97	0.98	380
Health Barley	0.98	0.98	0.98	430

Split 5: The total experimental findings obtained from the proposed model for Split four are provided in Table 5.9 in terms of Training, Validation, and Test accuracy, as well as Training, Validation, and Test loss in percentage form.

Table 5.9: Training, Validation, and Testing accuracy and loss of the Split five

Accuracy			Loss		
Training	Validation	Classification	Training	Validation	Classification
98.62%	97.96%	97.93%	1.40%	4.52%	1.07%

Figure 5.5 depicts the proposed model's outcomes for Split five during training in terms of training versus validation accuracy and training against validation loss.

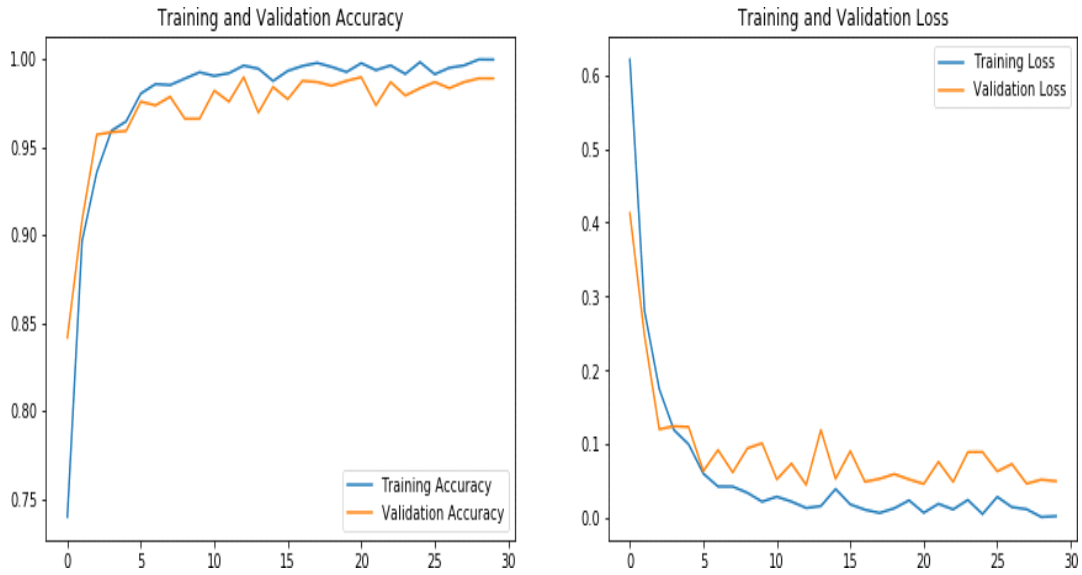
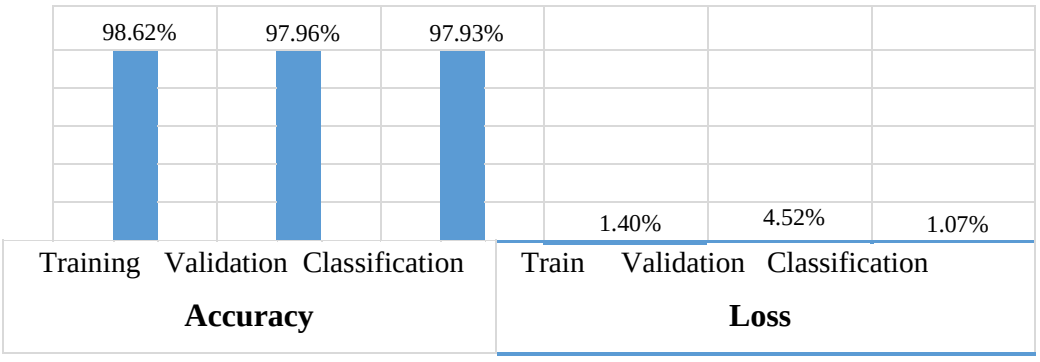


Figure 5.5: Training vs validation accuracy and Training vs validation loss of Split five



The detail experimental result of the proposed model is presented in the form of the confusion matrix, precision, recall, f1-scores, support, and confusion matrix for Split five for each class which is illustrated in Table 5.10.

Table 5.10: Precision, Recall, F1-Score, and Support Split five for test data

Class	Precision	Recall	F1-Score	Support
Diseased Barley	0.98	0.98	0.98	380
Health Barley	0.98	0.98	0.98	430

Split 6: The total experimental findings obtained from the proposed model for Split six are provided in Table 5.11 in terms of Training, Validation, and Test accuracy, as well as Training, Validation, and Test loss in percentage form.

Table 5.11: Training, Validation, and Testing accuracy and loss of the Split six

Accuracy			Loss		
Training	Validation	Classification	Training	Validation	Classification
98.28%	97.89%	97.87%	2.28%	5.10%	1.13%

Figure 5.6 depicts the proposed model's outcomes for Split six during training in terms of training versus validation accuracy and training against validation loss.

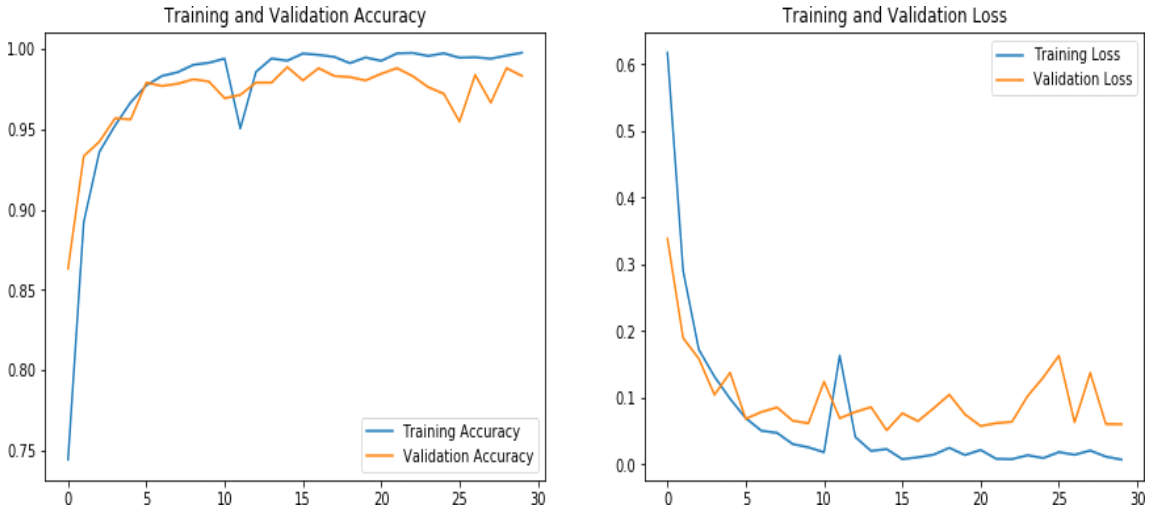
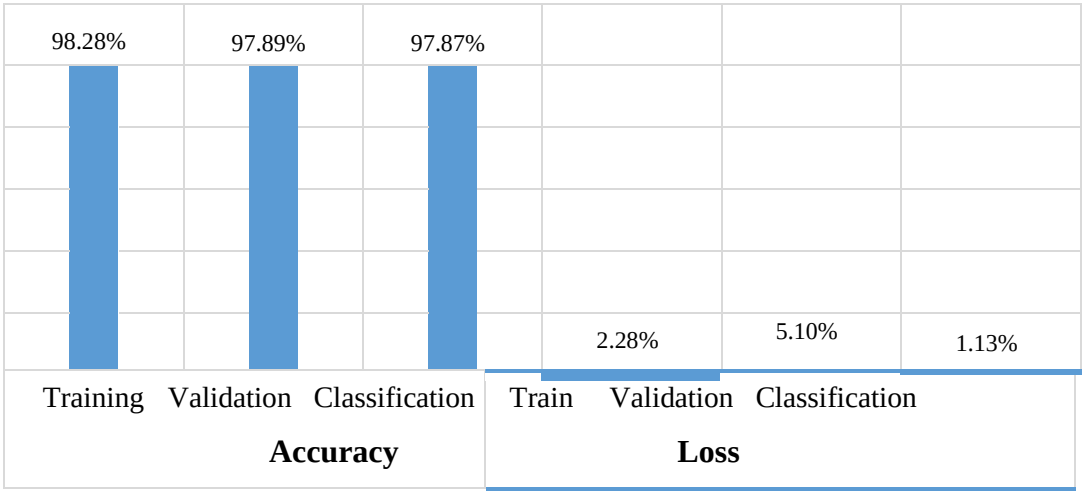


Figure 5.6: Training vs validation accuracy and Training vs validation loss of Split six.



The detail experimental result of the proposed model is presented in the form of the confusion matrix, precision, recall, f1-scores, support, and confusion matrix for Split six for each class which is illustrated in Table 5.12.

Table 5.12: Precision, Recall, F1-Score, and Support Split six for test data

Class	Precision	Recall	F1-Score	Support
Diseased Barley	0.98	0.98	0.98	380
Health Barley	0.98	0.97	0.98	430

Split 7: The total experimental findings obtained from the proposed model for Split seven are provided in Table 5.13 in terms of Training, Validation, and Test accuracy, as well as Training, Validation, and Test loss in percentage form.

Table 5.13: Training, Validation, and Testing accuracy and loss of the Split seven

Accuracy			Loss		
Training	Validation	Classification	Training	Validation	Classification
98.38%	98.31%	97.33%	2.04%	4.76%	0.67%

Figure 5.8 depicts the proposed model's outcomes for Split seven during training in terms of training versus validation accuracy and training against validation loss.

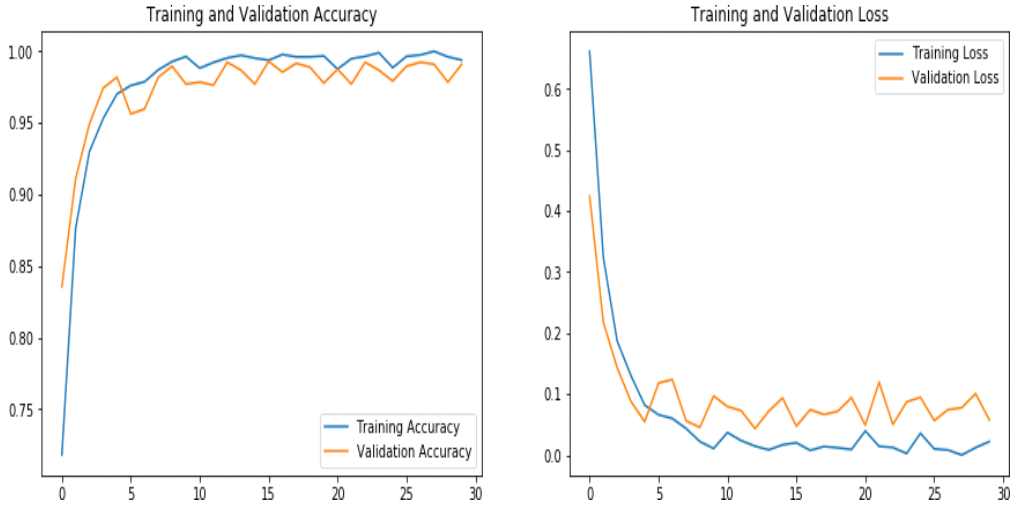
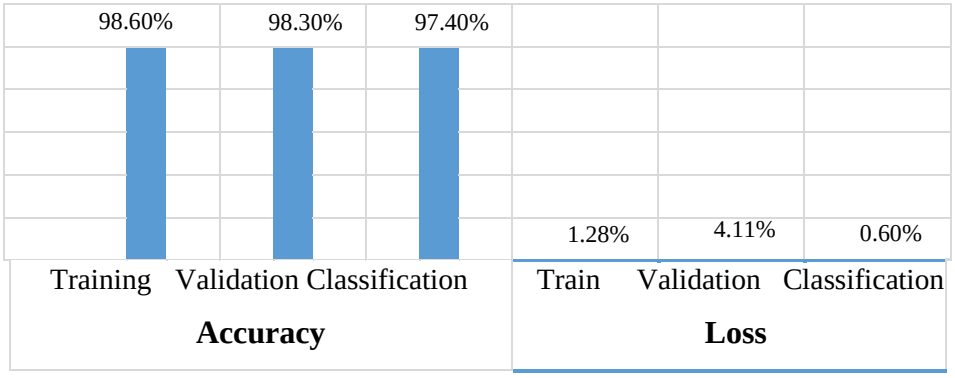


Figure 5.8: Training vs validation accuracy and Training vs validation loss of Split seven



The detail experimental result of the proposed model is presented in the form of the confusion matrix, precision, recall, f1-scores, support, and confusion matrix for Split seven for each class which is illustrated in Table 5.14.

Table 5.14: Precision, Recall, F1-Score, and Support Split seven for test data

Class	Precision	Recall	F1-Score	Support
Diseased Barley	0.99	0.99	0.99	380
Health Barley	0.99	0.98	0.99	430

Split 8: The total experimental findings obtained from the proposed model for Split eight are provided in Table 18 in terms of Training, Validation, and Test accuracy, as well as Training, Validation, and Test loss in percentage form.

Table 5.15: Training, Validation, and Testing accuracy and loss of the Split eight

Accuracy			Loss		
Training	Validation	Classification	Training	Validation	Classification
98.50%	**98.51%**	**99.53%**	**1.51%**	**2.37%**	**0.47%**

Figure 5.9 depicts the proposed model's outcomes for Split eight during training in terms of training versus validation accuracy and training against validation loss.

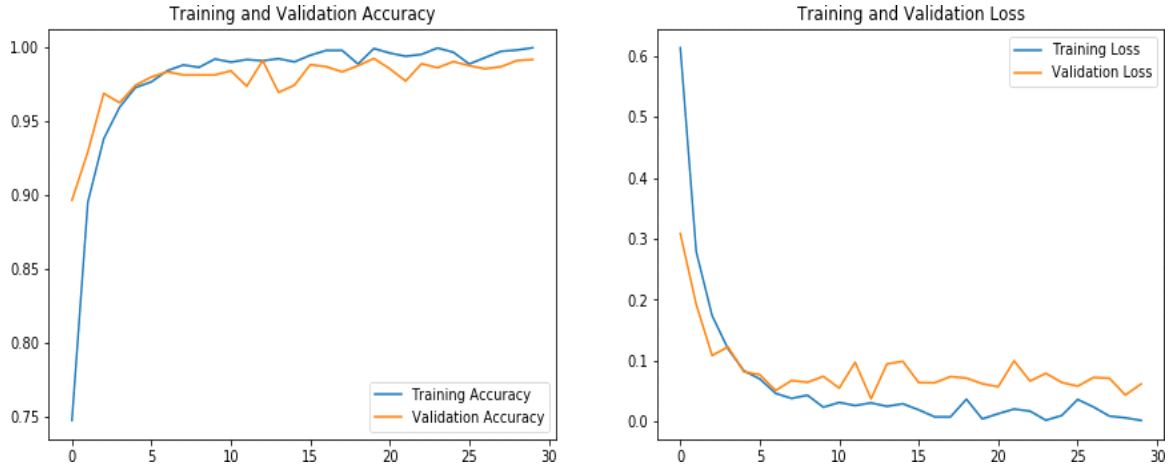
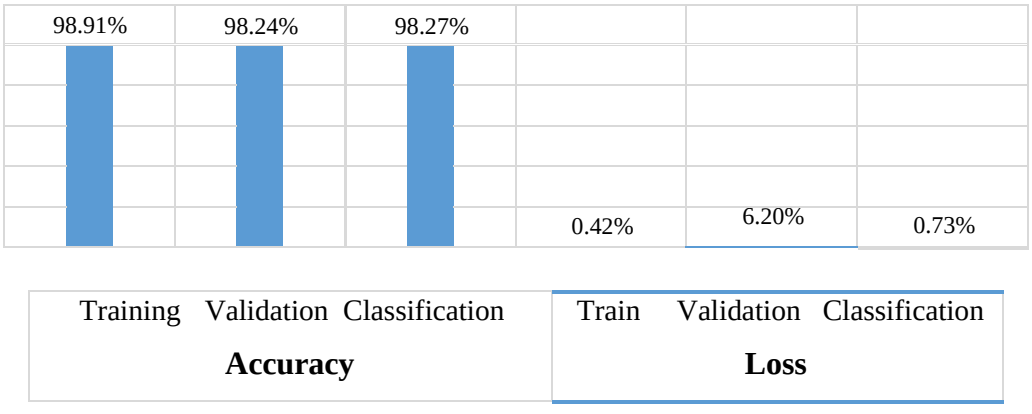


Figure 5.9: Training vs validation accuracy and Training vs validation loss of Split eight.



The detail experimental result of the proposed model is presented in the form of the confusion matrix, precision, recall, f1-scores, support, and confusion matrix for Split eight for each class which is illustrated in Table 5.16.

Table 5.16: Precision, Recall, F1-Score, and Support Split eight for test data

Class	Precision	Recall	F1-Score	Support
Diseased Barley	0.99	0.99	0.99	380
Health Barley	0.99	0.98	0.98	430

The findings of eight cross-validation strategy studies are summarized in the table below. Among the eight Splits, the last Split produces the best results, while split four produces the lowest results.

Table 5.17: Summary results of the proposed model using 8 Cross-validation approaches.

Splits	**Accuracy**			**Loss**		
	Training	**Validation**	**Classification**	**Training**	**Validation**	**Classification**
Split 1	98.93%	98.03%	98.07%	0.31%	4.54%	0.93%
Split 2	98.96%	98.37%	98.33%	0.13%	2.65%	0.67%
Split 3	98.48%	98.10%	98.07%	1.84%	3.29%	0.93%
Split 4	98.58%	97.75%	97.73%	1.15%	5.21%	1.27%
Split 5	98.62%	97.96%	97.93%	1.40%	4.52%	1.07%
Split 6	98.28%	97.89%	97.87%	2.28%	5.10%	1.13%
Split 7	98.38%	98.31%	97.33%	2.04%	4.76%	0.67%
Split 8	**98.50%**	**98.51%**	**99.53%**	**1.51%**	**2.37%**	**0.47%**

5.3 Training using experimental strategy 2

In this part, we use the train test Split technique to run three experiments using the proposed model at varying learning rates. Table 21 illustrates the % results for various accuracy and loss measures such as training, validation, and testing. Providing higher learning rates has much less accuracy than giving smaller learning rates, according to the results. In the proposed model, the learning rate of **0.001** provides us a good result among the three different learning rates.

Table 5.18: Experimental results of the proposed model using different learning rate

Learning Rate	Epoch	Accuracy			Loss		
		Training	Validation	Classification	Training	Validation	Classification
0.001	**30**	**98.50%**	**98.51%**	**98.53%**	**1.51%**	**2.37%**	**0.47%**
0.01	30	95.44%	97.36%	96.91%	1.261%	6.88%	2.09%
0.1	30	94.38%	96.06%	95.12%	16.06%	11.34%	2.88%

5.4 Training using experimental strategy 3

Two tests with the proposed model using different activation functions in the output layer are carried out in this part. Table 5.19 illustrates the % results for various accuracy and loss measures such as training, validation, and testing. The experimental findings demonstrate that for proposed binary classification problems, the sigmoid activation function is preferred, whereas the Soft-max activation function is better for multi classification. As a consequence, under the proposed model, using the sigmoid activation function yields the best results.

Table 5.19: Experimental results of the proposed model using different activation functions

Activation function	**Epoch**	**Accuracy**			**Loss**		
		Training	Validation	Classification	Training	Validation	Classification
Soft-Max	30	96.41%	96.14%	96.97%	10.00%	5.53%	2.03%
Sigmoid	**30**	**98.31%**	**98.48%**	**97.65%**	**4.3%**	**5.7%**	**6.23%**

5.5 Training using experimental strategy 4

In this section, several experiments are conducted with the proposed model using different epochs. Table 5.20 below shows us some of the results obtained concerning accuracy and loss metrics such as training, validation, and testing in the form of percentage separately. Results show that giving a higher epoch value during the training process gives better accuracy than that of smaller epoch values. Therefore, in this experiment, the epoch value of 30 gives us an optimal result in the proposed model.

Table 5.20: Experimental results of the proposed model using different epochs

Epoch Value	**Accuracy**			**Loss**		
	Training	Validation	Classification	Training	Validation	Classification
25	96.31%	96.48%	95.65%	4.3%	5.7%	6.23%
16	94.6%	95.2%	96.10%	15.5%	13.2%	8.6%
30	**98.50%**	**98.51%**	**99.53%**	**1.51%**	**2.37%**	**0.47%**

5.6 Comparison with other Pre-Trained Models

In this part, two experiments are conducted by applying transfer learning to compare the classification accuracy of two commonly researched classical pre-trained models, Mobile Net and InceptionV3, with the proposed model using the proposed dataset. Except for the convolution basis, a comparable dataset and hyper-parameter configuration are used in this implementation (feature extractor base). The Mobile Net architecture was chosen because it is an efficient model for mobile and embedded vision applications, while the Inceptionv3 model was chosen due to the complex feature.

5.6.1 Experimental results of InceptionV3 Model

One of the transfer learning strategies is used in this experiment to train the InceptionV3 model using the given dataset by retaining the convolution base and training fully collected layers. To be specific, the InceptionV3 model's networks are retrained using the proposed configured hyper-parameters shown in Table 5.21 without any fine-tuning in the convolution base, and only the output layers are modified to the proposed two classes. Table 5.21 shows the overall accuracy and

loss outcomes from the inceptionv3 model experiment in terms of classification accuracy and loss metrics for the training dataset, validation dataset, and test dataset, respectively.

Table 5.21: Overall classification accuracy and loss of InceptionV3 Model

Accuracy			Loss		
Training	Validation	Classification	Training	Validation	Classification
97.46%	96.71%	96.69%	1.70%	6.71%	2.31%

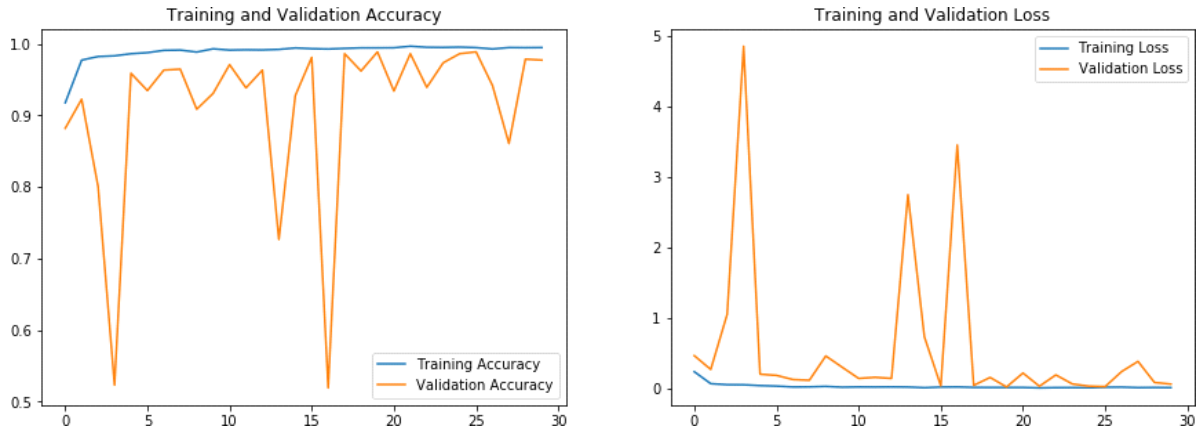


Figure 5.12: Training vs validation accuracy and Training vs validation loss inceptionv3 Model

The graphs in Figure 5.12 show us the results of the inceptionv3 model during training concerning training versus validation accuracy and training versus validation loss.

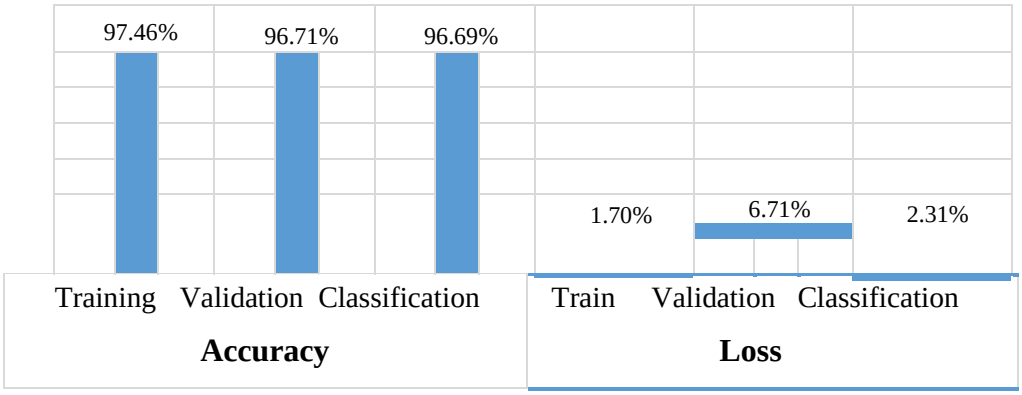


Table 5.22 illustrates the detail information of the precision, recall, F1-scores, and support for each class and their confusion matrix is plotted.

Table 5.22: Precision, Recall, F1-Score, and Support of InceptionV3 Model

Class	Precision	Recall	F1-Score	Support

Diseased Barley	0.926	1.00	0.967	430
Health Barley	1.00	0.930	0.96	520

5.6.2 Experimental results of Mobile-Net Model

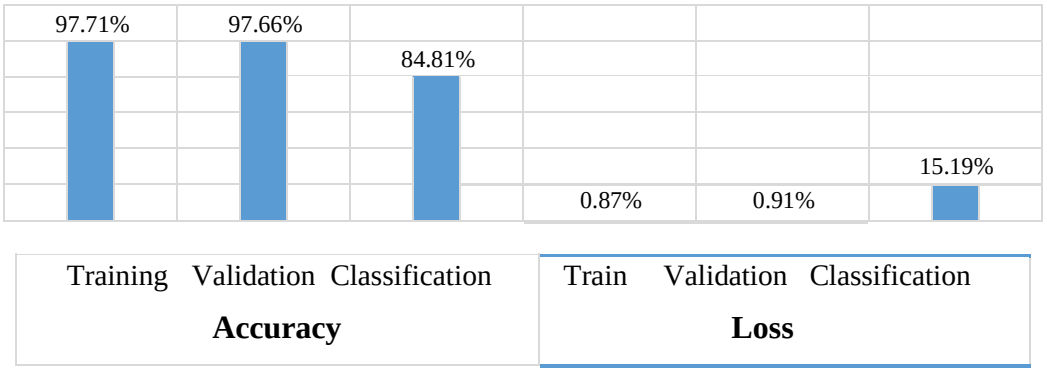
One of the transfer learning strategies is used in this experiment to retrain the Mobile Net model using the given dataset by freezing the convolution base and retraining fully collected layers. To be clear, the Mobile Net model's networks are retrained using the proposed configuration hyper-parameters shown in Table 5.23 without any fine-tuning in the convolution base, and only the output layers are trained to propose two classes.

Results and Evaluation of Mobile Net Model

The confusion matrix, precision, recall, f1-scores, and support concerning top experiment values for each class are used to describe the Mobile Net model's detailed experimental results. The table below also demonstrates the outcomes of the Mobile Net model during training in terms of training versus validation accuracy and training against validation loss. The validation accuracy decreases between some epochs; there is a gap between training and validation accuracy; this is due to the model being overfitted in some epochs. Table 5.23 shows the overall accuracy and loss findings from the Mobile Net model experiment in terms of classification accuracy and loss metrics for the training dataset, validation dataset, and test dataset, respectively.

Table 5.23: Overall classification accuracy and loss of Mobile Net Model

Accuracy			**Loss**		
Training	Validation	Classification	Training	Validation	Classification
97.71%	97.66%	84.81%	0.87%	0.91%	15.19%



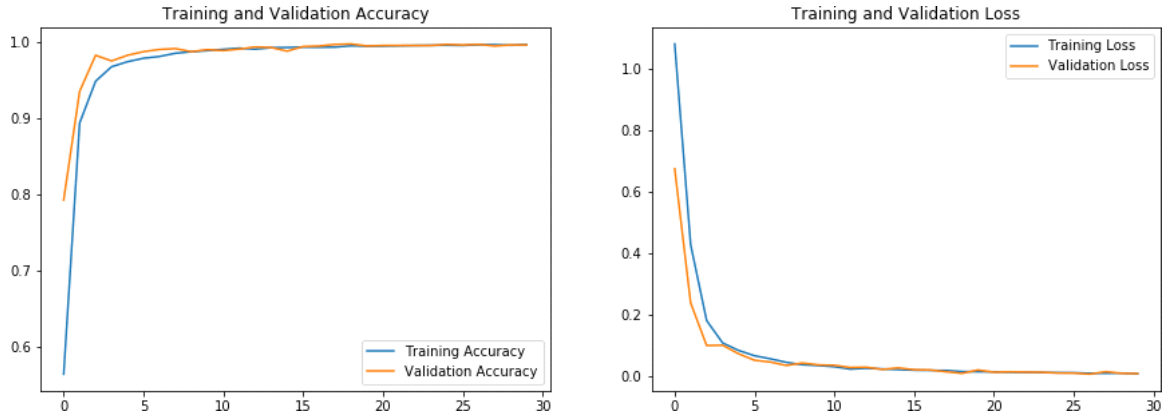


Figure 5.13: Training vs validation accuracy and Training vs validation loss Mobile Net Model

The graphs in Figure 22 above show us the results of the Mobile Net model during training concerning training versus validation accuracy and training versus validation loss. Table 5.24 illustrates the detail information of the precision, recall, F1-scores, and support for each class and their confusion matrix is plotted.

Table 5.24: Precision, Recall, F1-Score, and Support of Mobile Net model

Class	Precision	Recall	F1-Score	Support
Diseased Barley	0.54	1.00	0.70	430
Health Barley	1.00	0.66	0.80	520

5.7 Conclusion

The experiments are carried out using the proposed model as well as two additional typical CNN pre-trained models, InceptionV3 and Mobile Net, as detailed in the previous sections. The same dataset, software, hardware, and hyper-parameters setups are used in all experiments. All models are evaluated with a test dataset that is not observed during the model's training. The classification accuracy metrics are used to measure and compare the performance of the pre-trained classical models with the proposed model, and the new model achieves higher classification accuracy results. The percentages of the best training, validation, and testing accuracy of the proposed model are 98.50 percent, 98.51 percent, and 99.53 percent, respectively, as shown in the graph above. The Mobile Net model has 97.71 percent training accuracy, 97.66 percent validation accuracy, and 84.81 percent testing accuracy, respectively. The InceptionV3 model has training, validation, and testing accuracy of 97.45 percent, 96.71 percent, and 96.69 percent, respectively. When the

training, validation, and testing accuracy of the three experiments are compared, the proposed model performs better. These results show that the proposed model's detection and classification performance is superior to those of other models. The classification loss metrics are used to measure and compare the inconsistency of the pre-trained classical models with the proposed model.

CHAPTER SIX

CONCLUSION AND RECOMMENDATION

6.1 Conclusion

Different types of crops can be infected with several kinds of diseases. The naked eye observation of experts is the main approach used in practice for detection and identification of these diseases. But this needs continuous monitoring of experts. This evaluation process is however, tedious, time consuming, and less accurate. The main aim of this research is todesign and implement a model for a classification of barley leaf images with fungal disease using deep learning approach. The convolutional neural network algorithm was generally applied at the classification stage.

To make early detection and identification of the barley crop leaf diseases, this research has proposed and implemented a CNN based model that is used as a decision-making tool to help and support farmers and experts. The reason why this thesis work selects deep learning approach is that computer vision with deep learning methodologies has an outperformed in solving a total number of plant disease problems by extracting features without human intervention.In the proposed model, hyper-parameters are used to train the model and get good results, such as learning rate; 0.001, activation's function; ReLU and sigmoid, batch size; 64, optimization algorithm; Adam, Loss function; Binary cross-entropy. Then after a set of experiments, the proposed model finds good identification results. Experimental results show that the proposed model has higher performance, computationally efficient, simple and has a minimum classification accuracy of 97.40%, maximum classification accuracy of 99.53% and average classification of 98.065% which is higher than compared with that of existing classical CNN state-of-the-art models, such as Mobile Net and inception V3. This identification accuracy indicates that the proposed model is more effective and gives a better solution to identify barley leaf fungal diseases. Moreover, the proposed model can serve as a decision support tool to help farmers and experts around the area to identify fungal disease of barley leaf crop.

6.2 Recommendation

The sustainability of agriculture is important to Ethiopia's economy. Many agricultural products, including wheat, barley, bananas, tomatoes, and others, are vulnerable to many diseases. Image processing and deep learning technologies are important in identifying these diseases. No research has been conducted using CNN models and image analysis approaches to identify and classify fungal disease affecting leaf of barley crops. There are too many disease types that can impact many types of crops, hence only one disease and one part of barley crop was taken under consideration in this study. The researcher encourages others to continue their work in the field. Moreover, the facts, algorithms, rules, and feature analysis concepts used in this work can be implemented by researchers who are interested in designing and developing any system related to plant diseases classification or identification with related methodologies, with modification.

In addition to this, the researcher will make the model estimate the severity of the disease automatically to help farmers to decide whether to stop the disease or not. To apply this research in the field of agriculture the researcher recommends developing a mobile app that takes image of barley crop and gives automatic results about the severity of the disease in the taken image and giving helpful expert advice to the user.

References

[1] *Federal Democratic Republic of Ethiopia Central Statistical Agency (FDRECSA), "KEY FINDINGS OF THE 2014/2015 (2007 E.C.) AGRICULTURAL SAMPLE SURVEYS," Addis Ababa, 2015.*

[2] *K. A. Garrett, S. P. Dendy, E. E. Frank, M. N. Rouse, and S. E. Travers, "Climate change effects on crop disease: genomes toecosystems," Annual Review of Phytopathology, vol. 44, pp. 489–509, 2006.*

[3] *S. M. Coakley, H. Scherm, and S. Chakraborty, "Climate change and crop disease management," Annual Review of Phytopathology-ogy, vol. 37, no. 1, pp. 399–426, 1999.*

[4] *S. Chakraborty, A. V. Tiedemann, and P. S. Teng, "Climate change: potential impact on crop diseases," Environmental Pol- lution, vol. 108, no. 3, pp. 317–326, 2000.*

[5] *A. J. Tatem, D. J. Rogers, and S. I. Hay, "Global transport net-works and infectious disease spread," Advances in Parasitology, vol. 62, pp. 293–343, 2006.*

[6] *J. R. Rohr, T. R. Raffel, J. M. Romansic, H. McCallum, and P. J.Hudson, "Evaluating the links between climate, disease spread,and amphibian declines," Proceedings of the National Academyof Sciences of the United States of America, vol. 105, no. 45, pp.17436–17441, 2008.*

[7] *T. Van der Zwet, "Present worldwide distribution of fire blight,"in Proceedings of the 9th International Workshop on Fire Blight,vol. 590, Napier, New Zealand, October 2001.*

[8] *S. A. Miller, F. D. Beed, and C. L. Harmon, "Crop disease diagnostic capabilities and networks," Annual Review of Phy- topathology, vol. 47, pp. 15–38, 2009.*

[9] *S. W. Fanta and S. Neela, "A review on nutritional profile of the food from Barley: A staple diet for more than 25 per cent population in Ethiopia," Nutrition & Food Science, vol. 49, no. 5, pp. 824--843, 2019.*

[10] *("Efficiency of Intercropping System under Smallholder Farmers in Osun State, Nigeria,"2019)*

[11] *J. Barbedo and A. Garcia, "Digital image processing techniques for detecting, quantifying and classifying crop diseases," Springer Plus, vol. 2, no. 660, pp. 1-12, 2013.*

[12] *J. Amara, B. Bouaziz and A. Algergawy, "A Deep Learning-based Approach for Banana Leaf Diseases Classification," in BTW (Workshops), 2017, pp. 79-88.*

[13] *H. Al-Hiary, S. Bani-Ahmad, M. Reyalat, M. Braik and Z. ALRahamneh, "ast and accurate detection and classification of crop diseases," Machine learning, vol. 14, no. 5, pp. 31-38, 2011.*

[14] *D. Cui, Q. Zhang, M. Li, G. L. Hartman and Y. Zhao, "Image processing methods for quantitatively detecting soybean rust from multispectral images," Biosystems engineering, vol. 107, no. 3, pp. 186-193, 2010.*

[15] *G. Owomugisha, J. A. Quinn, E. Mwebaze and J. Lwasa, "Automated Vision-Based Diagnosis of Banana Bacterial Wilt Disease and Black Sigatoka Disease," International Conference on the Use of Mobile ICT in Africa, pp. 1-5, 2014.*

[16] *S. Patil and A. Chandavale, "A survey on methods of crop disease detection," International Journal of Science and Research (IJSR), vol. 4, no. 2, pp. 1392-1396, 2015.*

[17] *Wheat Leaf Identification, Kansas State University, Agricultural Experiment Station,2017.*

[18] *Wafaa M, Haggag, "Barley Diseases in Egypt and its management," Journal of Applied Sciences Research, vol. IX, no. 1, pp. 46-50, 2013.*

[19] *P. Josh and G. Adam, Deep Learning A Practitioner's Approach, Sebastopol: O'Reilly Media, 2017*

[20] *G. Ian, B. Yoshua and C. Aaron, Deep Learning, MIT Press, 2016.*

[21] *Y. LeCun, Y. Bengio and G. Hinton, "Deep learning," nature, vol. 521, no. 7553, pp. 436-444, 2015.*

[22] *J. Schmidhuber, "Deep learning in neural networks: An overview," Neural networks, vol. 61, pp. 85-117, 2015.*

[23] *]. Saha, S. (2015). A Comprehensive Guide to Convolutional Neural Networks -the ELI5 way. In Towards Data Science.*

[24] *Weldeyohanis Kifle, S. (2016). Review on Barley Production and Marketing in Ethiopia. Journal of Economics and Sustainable Development Www.Iiste.Org ISSN, 7(9).*

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

[26] *[Online] Available: https://www.analyticsvidhya.com/blog/2021/05/introduction-to-supervised-deep-learning-algorithms/*

[27] *C. François, Deep Learning with Python, New York: Manning Publications, 2017.*

[28] *[Online]. Available: https://www.analyticsvidhya.com/blog/2021/06/offline-data-augmentation-for-multiple-images/*

[29] *[Online].Available: DISHASHREE GUPTA, (JANUARY 30, 2020) Fundamentals of Deep Learning –Activation functions.https://www.analyticsvidhya.com/blog/2020/01/fundamentals-deep-learning-activation-functions-when-to-use-them/*

[30] *[Online].Available: Pragati Baheti, 12 Types of Neural Networks Activation Functions: https://www.v7labs.com/blog/neural-networks-activation-functions#activation-function*

[31] *[Online].Available Aditya Mishra (Feb 24,2018), Metrics to Evaluate your Machine Learning Algorithm:https://towardsdatascience.com/metrics-to-evaluate-your-machine-learning-algorithm-f10ba6e38234*

[32] *F. Li, J. Justin and Y. Serena, "CS231n: Convolutional Neural Networks for Visual Recognition," Stanford University, Spring 2018. [Online]. Available: [Accessed 20 january 2019].*

[33] *K. P. Ferentinos, "Deep learning models for crop disease detection and diagnosis," Computers and Electronics in Agriculture, vol. 145, pp. 311-318, 2018.*

[34] *Yidnekachew Kibru Afework, B. (n.d.). "Developing bacterial wilt detection model on Barley crop using a deep learning approach", October 2019.*

[35] *Sharma, S., Sharma, S., & Athaiya, A. (2020). ACTIVATION FUNCTIONS IN NEURAL NETWORKS. International Journal of Engineering Applied Sciences and Technology, 04(12). https://doi.org/10.33564/ijeast.2020.v04i12.054.*

[36] *Afifi, A., Alhumam, A., & Abdelwahab, "A. Convolutional neural network for automatic identification of crop diseases with limited data. " 2021.https://doi.org/10.3390/crops10010028.*

[37] *Rasjava, A. R., Sugiyarto, A. W., Kurniasari, Y., & Ramadhan, S. Y. (2020). Detection of Rice Crops Diseases Using Convolutional Neural Network (CNN). Proceeding International Conference on Science and Engineering, 3. https://doi.org/10.14421/icse.v3.535.*

[38] *Kahsay, M. Classification of wheat leaf septoria disease using image processing and machine learning techniques a ",(2019).*

[39] *Ramcharan, A., McCloskey, P., Baranowski, K., Mbilinyi, N., Mrisho, L., Ndalahwa, M., Legg, J., & Hughes, D. P. (2019). A mobile-based deep learning model for cassava disease diagnosis. Frontiers in Crop Science, 10. https://doi.org/10.3389/fpls.2019.00272*

[40] *Saleem, M. H., Potgieter, J., & Arif, K. M. (2020). Crop disease classification: A comparative evaluation of convolutional neural networks and deep learning optimizers. Crops, 9(10). https://doi.org/10.3390/crops9101319*

[41] *Ennadifi, E., Laraba, S., Vincke, D., Mercatoris, B., & Gosselin, B. (2020). Wheat Diseases Classification and Localization Using Convolutional Neural Networks and GradCAM Visualization. 2020 International Conference on Intelligent Systems and Computer Vision, ISCV 2020. https://doi.org/10.1109/ISCV49265.2020.9204258*

[42] *[Online].Available: Google, "Machine Learning Crash Course Google Developers,"2019.https://developers.google.com/machinelearning/crashcourse/classification/true-false positive-negative.*

[43] *T. N. T. a. S. Kamlu, "Crop disease detection using different algorithms," in Proceedings of the Second International Conference on Research in Intelligent and Computing in Engineering , Vol.10, pp-103-106, 2017.*

[44] *Tigadi, B., & Sharma, B. (2016). Banana Crop Disease Detection and Grading Using Image Processing. International Journal of Engineering Science and Computing, 6(6), 6512–6516.*

[45] *Qiu, R., Yang, C., Moghimi, A., Zhang, M., Steffenson, B. J., & Hirsch, C. D. (2019). Detection of Fusarium Head Blight in wheat using a deep neural network and color imaging. Remote Sensing, 11(22). https://doi.org/10.3390/rs11222658*

[46] *Dewit Alemu, kaleb Kelemu, Berhane Lakew (2014). Value Chain Analysis of Malt Barley in Arsi and West Arsi Zones of Oromia Region.*

APPENDIX

```
import os

import numpy as np from PIL import Image import tensorflow as tf

import matplotlib.pyplot as plt

from tensorflow.keras.models import Sequential

from tensorflow.keras.preprocessing.image import ImageDataGenerator

from tensorflow.keras.layers import Dense, Conv2D, Flatten, Dropout, MaxPooling2D,lr

from     future     import absolute_import, division, print_function, unicode_literals

#8_fold split to create 8 Splits

 import os, os.path, shutil

import random

Dataset = "D:/BarleyDataSet/Training" classes = os.listdir(Dataset)

if not os.path.exists("D:/BarleyDataSet/K_fold"): os.mkdir("D:/BarleyDataSet/K_fold")

for folder_path in classes:

fold = 8

images = os.listdir(os.path.join(Dataset, folder_path)) random.shuffle(images)

print(len(images))

number_of_images = int(len(images)/fold)+1 fold_counter = 0

i = 0

curr_subdir = None

if os.path.exists("D:/BarleyDataSet/K_fold/" + folder_path): shutil.rmLeaf("D:/BarleyDataSet/K_fold/"
+ folder_path) os.mkdir("D:/BarleyDataSet/K_fold/" + folder_path)

else: os.mkdir("D:/BarleyDataSet/K_fold/" + folder_path) for f in images:

# Move file to current dir

f_base = os.path.basename(f) shutil.copy(os.path.join(Dataset,folder_path,f),
os.path.join(subdir_name,f))

i += 1
```

```
#To plot_confusion_matrix with its classification accuracy and Classification loss

 def plot_confusion_matrix(cm,

target_names, title='Confusion matrix', cmap=None, normalize=False):

import matplotlib.pyplot as plt import numpy as np

import itertools

accuracy = np.trace(cm) / np.sum(cm).astype('float')

misclass = 1 - accuracy

if cmap is None:

cmap = plt.get_cmap('Blues') plt.figure(figsize=(8, 6))

plt.imshow(cm, interpolation='nearest', cmap=cmap) plt.title(title)

plt.colorbar()

if target_names is not None:

tick_marks = np.arange(len(target_names)) plt.xticks(tick_marks, target_names, rotation=45)
plt.yticks(tick_marks, target_names)

if normalize:

cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis] thresh = cm.max() / 1.5 if normalize else
cm.max() / 2

for i, j in itertools.product(range(cm.shape[0]), range(cm.shape[1])): if normalize:

plt.text(j, i, "{:0.2g}".format(cm[i, j]), horizontalalignment="center", color="white" if cm[i, j] > thresh
else "black")

else:

plt.text(j, i, "{:,}".format(cm[i, j]), horizontalalignment="center", color="white" if cm[i, j] > thresh else
"black")

plt.tight_layout() plt.ylabel('True label')

plt.xlabel('Predicted label\naccuracy={:0.4f}; misclass={:0.4f}'.format(accuracy, misclass))

plt.show()

from keras.preprocessing.image import ImageDataGenerator import numpy as np
```

from sklearn.metrics import confusion_matrix, precision_recall_fscore_support from tensorflow.keras.preprocessing.image import ImageDataGenerator

from keras.preprocessing import image import tensorflow as tf

import os, os.path import shutil import random import keras

for i in range(1, 11):

train_dir = 'D:/BarleyDataSet/K_fold/train' validation_dir = 'D:/BarleyDataSet/K_fold/validation'

#To create sub/Class directories inside training directory

if os.path.exists("D:/BarleyDataSet/K_fold/train/HealthyBarleyLeaf"):
shutil.rmLeaf("D:/BarleyDataSet/K_fold/train/HealthyBarleyLeaf")
os.mkdir("D:/BarleyDataSet/K_fold/train/HealthyBarleyLeaf")

else: os.mkdir("D:/BarleyDataSet/K_fold//train/HealthyBarleyLeaf")

if os.path.exists("D:/BarleyDataSet/K_fold/train/InfectedBarleyLeaf"):
shutil.rmLeaf("D:/BarleyDataSet/K_fold/train/ InfectedBarleyLeaf ")
os.mkdir("D:/BarleyDataSet/K_fold/train/ InfectedBarleyLeaf ")

else: os.mkdir("D:/BarleyDataSet/K_fold/train/ InfectedBarleyLeaf ")

#To create sub/Class directories inside validation directory

if os.path.exists("D:/BarleyDataSet/K_fold/validation/HealthyBarleyLeaf"):
shutil.rmLeaf("D:/BarleyDataSet/K_fold/validation/HealthyBarleyLeaf")
os.mkdir("D:/BarleyDataSet/K_fold/validation/HealthyBarleyLeaf")

else: os.mkdir("D:/BarleyDataSet/K_fold/validation/HealthyBarleyLeaf")

if os.path.exists("D:/BarleyDataSet/K_fold/validation/ InfectedBarleyLeaf "):
shutil.rmLeaf("D:/BarleyDataSet/K_fold/validation/ InfectedBarleyLeaf ")
os.mkdir("D:/BarleyDataSet/K_fold/validation/ InfectedBarleyLeaf ")

else: os.mkdir("D:/BarleyDataSet/K_fold/validation/ InfectedBarleyLeaf ")

#To copy the training images to training sub/class directories

for j in range(1,9):

if j!=i:

```
for file in os.listdir('D:/BarleyDataSet/K_fold/Healthy Barley Leaf/fold_' + str(j)):
shutil.copy(os.path.join('D:/BarleyDataSet/K_fold/Healthy Barley Leaf/fold_' + str(j), file),

os.path.join("D:/BarleyDataSet/K_fold/train/HealthyBarleyLeaf", file)) # performs copy & overwrite


for file in os.listdir('D:/BarleyDataSet/K_fold/ InfectedBarleyLeaf /fold_' + str(j)):
shutil.copy(os.path.join('D:/BarleyDataSet/K_fold/ InfectedBarleyLeaf /fold_' + str(j), file),

os.path.join("D:/BarleyDataSet/K_fold/train/ InfectedBarleyLeaf ", file))

#Assign Path of each class/sub training directories

train_HealthyBarleyLeaf_dir = 'D:/BarleyDataSet/K_fold/train/HealthyBarleyLeaf' # directory with our
training Healthy BarleyLeaf pictures

train_BacterialWilt_dir = 'D:/BarleyDataSet/K_fold/train/ InfectedBarleyLeaf ' # directory with our
training BacterialWilt pictures

#To copy the validation images to validation sub/class directories

for file in os.listdir('D:/BarleyDataSet/K_fold/Healthy Barley Leaf/fold_' + str(i)):

shutil.copy(os.path.join('D:/BarleyDataSet/K_fold/Healthy Barley Leaf/fold_' + str(i), file),
os.path.join("D:/BarleyDataSet/K_fold/validation/HealthyBarleyLeaf", file))

 # Performs copy & overwrite

for file in os.listdir('D:/BarleyDataSet/K_fold/ InfectedBarleyLeaf /fold_' + str(i)):
shutil.copy(os.path.join('D:/BarleyDataSet/K_fold/ InfectedBarleyLeaf /fold_' + str(i), file),

os.path.join("D:/BarleyDataSet/K_fold/validation/ InfectedBarleyLeaf ", file))

#Assign Path of each class/sub validation directories

validation_HealthyBarleyLeaf_dir = "D:/BarleyDataSet/K_fold/validation/HealthyBarleyLeaf"
validation_BacterialWilt_dir = "D:/BarleyDataSet/K_fold/validation/ InfectedBarleyLeaf "

#To configure Batch_size, epochs, image size, learning rate

model.compile (loss='binary_crossentropy',

                    Optimizer=Adam(lr=0.001),

                        Metrics=[accuracy])

model_fit=model.fit (train_dataset,

                         steps_per_epoch=5,
```

epochs=30,

validation_data=validation_dataset)

#To augment the training data using following parameters

image_gen_train = ImageDataGenerator(rescale=1./255, rotation_range=45, width_shift_range=.15, height_shift_range=.15, horizontal_flip=True, zoom_range=0.5)

#To augment the validation data using the following parameters

image_gen_val = ImageDataGenerator(rescale=1./255, rotation_range=45, width_shift_range=.15, height_shift_range=.15, horizontal_flip=True, zoom_range=0.5)

 # Generator for normalizing our training image data

data train_image_generator = ImageDataGenerator(rescale=1./255)

 # Generator for normalizing our validation image data

validation_image_generator = ImageDataGenerator(rescale=1./255)

#To count a print the total number of training images with their number of classes

class train_data_gen = train_image_generator.flow_from_directory(batch_size=batch_size,

directory=train_dir, shuffle=True,

target_size=(IMG_HEIGHT, IMG_WIDTH), class_mode='binary')

#To count a print the total number of training images with their number of class

val_data_gen = validation_image_generator.flow_from_directory(batch_size=batch_size,

directory=validation_dir, target_size=(IMG_HEIGHT, IMG_WIDTH), class_mode='binary')

#Train the model 8(k) times

model = Sequential([Conv2D(16, 3,

padding='same', activation='relu',

input_shape=(IMG_HEIGHT, IMG_WIDTH ,3)),

MaxPooling2D(pool_size=(2, 2)),

Conv2D(32, 3,

padding='same', activation='relu'),

MaxPooling2D(pool_size=(2, 2)),

```
Conv2D(64, 3,

padding='same', activation='relu'),

MaxPooling2D(pool_size=(2, 2)),

Conv2D(32, 3,

padding='same', activation='relu'),

MaxPooling2D(pool_size=(2, 2)), Flatten(),

Dense(512, activation='relu'),

Dropout(0.2), Dense(100,

activation='relu'), Dropout(0.2),

Dense(4, activation=sigmoid)])

model.summary() model.compile(optimizer='adam',

loss='binary_crossentropy', metrics=['accuracy'])


# To check and save best model among 30 epochs that has maximum validation value

from tensorflow.keras.callbacks import ModelCheckpoint
filepath="D:/BarleyDataSet/K_fold/BarleyNetModel_fold_" + str(i) + ".hdf5"

checkpoint = ModelCheckpoint(filepath, monitor='val_acc', verbose=1, save_best_only=True,
mode='max') callbacks_list = [checkpoint]

history = model.fit_generator( train_data_gen, callbacks=callbacks_list, verbose=1,

    steps_per_epoch=total_train // batch_size, epochs=epochs, validation_data=val_data_gen,
    validation_steps=total_val // batch_size)

#To plot the training accuracy and validation accuracy with their loss

acc = history.history['acc']

val_acc = history.history['val_acc'] loss = history.history['loss'] val_loss = history.history['val_loss']
epochs_range = range(epochs)

plt.figure(figsize=(15, 5))

plt.subplot(1, 2, 1)
```

plt.plot(epochs_range, acc, label='Training Accuracy') plt.plot(epochs_range, val_acc, label='Validation Accuracy') plt.legend(loc='lower right')

plt.title('Training and Validation Accuracy')

plt.subplot(1, 2, 2)

plt.plot(epochs_range, loss, label='Training Loss') plt.plot(epochs_range, val_loss, label='Validation Loss') plt.legend(loc='upper right')

plt.title('Training and Validation Loss') plt.show()

plt.savefig("D:/BarleyDataSet/K_fold/plot figures from fold_" + str(i)+".png") #To plot confusion matrix and print class labels with its accuracy and loss

saved_model = tf.keras.models.load_model("D:/BarleyDataSet/K_fold/BarleyNetModel_fold_" + str(i) + ".hdf5")

image_gen_val = ImageDataGenerator(rescale=1./255)

val_data_gen = image_gen_val.flow_from_directory(batch_size=batch_size, directory=validation_dir, shuffle=False,

target_size=(IMG_HEIGHT, IMG_WIDTH), class_mode='binary')

pred = saved_model.predict(val_data_gen) y_pred = np.rint(pred)

max_index = np.argmax(y_pred, axis=1) print("Predicted Classes ") print(max_index)

print("\n")

y_true = val_data_gen.classes print("True Classes ") print(y_true)

print("\n")

cnf_matrix = confusion_matrix(y_true, max_index) print("Confusion Matrix")

print(cnf_matrix) print("\n")

print("precision, recall, fscore and support for each class") print(precision_recall_fscore_support(y_true, max_index, average=None)) print("\n")

classes=[' InfectedBarleyLeaf, 'Healthy Barley'] print("Plot Confusion Matrix")

plt.figure()

plot_confusion_matrix(cnf_matrix, target_names=classes, title='Confusi on matrix')

plt.savefig('Confussion Matrix of fold_'+ str(i) +'.png') print("\n")

filePath = "D:/BarleyDataSet/K_fold/validation/InfectedBarleyLeaf" if os.path.exists(filePath):

os.remove(filePath)

else:print(" Next step>>>>>>>")

#To test/predict class of single image

saved_model = tf.keras.models.load_model("D:/BarleyDataSet/K_fold/BarleyNetModel_fold_" + str(i) + ".hdf5") import numpy as np

from keras.preprocessing import image import tensorflow as tf

img_width, img_height = 150, 150

img = image.load_img('D:/BarleyDataSet/Validation/ InfectedBarleyLeaf (2500).png ', target_size = (img_width, img_height)) img = image.img_to_array(img)

img = np.expand_dims(img, axis = 0) pred = saved_model.predict(img) print(pred)KM

max_index = np.where(pred[0] == np.amax(pred[0])) if(max_index[0][0]==0):

print("InfectedBarleyLeaf ")

elif(max_index[0][0]==1):

print("Healthy Barley")