Comparative Analysis of CNN, SVM, and ANN Models for Healthy and Unhealthy Leaf Identification

B.Sc. (Engineering) Thesis
Faculty of Engineering and Technology
Department of Information and Communication Engineering
B.Sc. (Engineering) Examination 2021
Course Code: ICE-4210
Course Title: Thesis

A thesis paper submitted to the Department of Information and Communication Engineering (ICE), Pabna University of Science and Technology (PUST) in partial fulfillment of the requirement for the degree of Bachelor of Science in Engineering in Information and Communication Engineering.

Submitted By
Md. Abu Yousuf
Roll No: 180640
Registration No: 1065323
Session: 2017-2018

Supervised By
Sohag Sarker
Associate Professor
Department of Information and Communication Engineering

Software Laboratory
Department of Information and Communication Engineering
Pabna University of Science and Technology
Pabna-6600, Bangladesh

DEDICATED

TO

MY BELOVED

PARENTS

CERTIFICATE

This is to certify that **Md. Abu Yousuf; Roll No: 180640;** Reg. No: 1065323; Session: 2017-18 has meticulously performed a thesis work entitled **"Comparative Analysis of CNN, SVM, and ANN Models for Healthy and Unhealthy Leaf Identification"** under my supervision for partial fulfillment of the requirement for the degree of Bachelor of Science in Engineering in Information and Communication Engineering. So far as I concern this is an original thesis that has been carried out for one year in the Department of Information and Communication Engineering, Pabna University of Science and Technology, Pabna-6600, Bangladesh. To the best of my knowledge, this paper has not been submitted to elsewhere prior submission to this department.

Sohag Sarker

Associate Professor

Department of Information and Communication Engineering

Pabna University of Science and Technology, Pabna-6600, Bangladesh.

DECLARATION

In accordance with rules and regulations of Pabna University of Science and Technology, I am Md. Abu Yousuf bearing Roll No: 180640 declaring that neither this thesis nor any part of this thesis has been submitted elsewhere for awarding of any degree and any material reproduced in this thesis has been properly acknowledged.

The Author
Md. Abu Yousuf
Roll No: 180640
Registration No: 1065323
Department of Information and Communication Engineering
Pabna University of Science and Technology, Pabna-6600, Bangladesh

Acknowledgement

First and foremost, I would like to thank ALLAH (omnipotent) for giving me strength, patience and ability to round off this study. I express my deepest gratitude and thanks to my honorable thesis supervisor Sohag Sarker, Associate Professor, Department of Information and Communication Engineering, Pabna University of Science and Technology, Pabna 6600, Bangladesh, for his excellent supervision and constructive advice, total support, constant inspiration and continuous encouragement which had contributed a lot to the completion of this project.

I am especially pleased and thankful to him for allowing me to work under his supervision.

I would like to convey my gratitude to honorable chairman, Dr. Md. Omar Faruk, Department of Information and Communication Engineering to carry out the present research work in the modern trend of Machine learning and deep learning and all the members of examination committee for their help to accomplish my project.

Last but not least, I would like to thank my parents for their blessing and love as well as my brothers and sisters for their constant support and encouragement.

Md. Abu Yousuf

Abstract

Over 70% of Bangladesh's population and 77% of its labor force reside in rural regions, directly or indirectly involved in agriculture. Therefore, the precise identification of diseased leaves holds paramount importance for our nation. Convolutional Neural Networks (CNNs), sophisticated deep learning algorithms with a storied development history, have achieved remarkable success across diverse domains, including computer vision, pattern recognition, and image classification. Simultaneously, Support Vector Machine (SVM) classifiers have garnered substantial attention, particularly in the domains of script recognition and image classification. Additionally, Artificial Neural Networks (ANNs) have become widely adopted and valuable models for various applications, including classification, clustering, pattern recognition, and prediction. As a machine learning (ML) model, ANNs are now capable of competing with traditional regression and statistical models in terms of utility. In this groundbreaking study, we embarked on an innovative investigation focused on the integration of three distinct classifiers: CNN, SVM, and ANN methodologies, to discriminate between healthy and unhealthy leaves. Our training and test datasets were meticulously curated from a bespoke repository. The experimental findings revealed remarkable results with a 99.72% accuracy in image recognition achieved through the CNN model, a notable 95.62% accuracy when employing ANN, and 94% accuracy using the SVM model. These achievements hold notable statistical significance when compared to the accuracies obtained in identifying healthy and unhealthy images within the dataset.

Table of Contents

List of Tables

List of Figures

CHAPTER 1
INTRODUCTION

1.1 Overview

Agriculture plays a pivotal role in the livelihood of Bangladesh, a nation where over 70% of the population and 77% of the labor force reside in rural regions and are closely tied to agrarian activities. For a country heavily reliant on agriculture, ensuring the health and productivity of crops is not merely an agricultural concern but a national imperative. Specifically, identifying and addressing the health of leaves is of paramount importance. Leaves serve as primary indicators of plant health and are a critical component of the nation's agricultural landscape [1]. In this era of rapid technological advancement, the integration of cutting-edge machine learning methodologies has revolutionized the field of agricultural research. Among these methodologies, Convolutional Neural Networks (CNNs), renowned for their sophisticated deep learning algorithms and storied development history, have demonstrated remarkable success across diverse domains. They excel in tasks related to computer vision, pattern recognition, and image classification, making them an ideal candidate for addressing agricultural challenges.

Simultaneously, Support Vector Machine (SVM) classifiers have garnered substantial attention in the domains of script recognition and image classification. Their ability to effectively discriminate between different classes makes them a valuable tool in the context of leaf health assessment.

Furthermore, Artificial Neural Networks (ANNs) have emerged as a widely used and useful model applicable to various fields, ranging from classification and clustering to pattern recognition and prediction. ANNs, a subset of machine learning models, have demonstrated the capability to compete with traditional regression and statistical models in terms of utility.

Recognizing the need for precise leaf health assessment in a nation where agriculture is the backbone of the economy, this study embarks on a novel investigation. It centers on the amalgamation of three distinct classifiers: CNN, SVM, and ANN methodologies. The primary objective is to discern between healthy and unhealthy leaves with a high degree of accuracy. To ensure the robustness and reliability of the study, both the training and test datasets were meticulously curated from a bespoke repository, reflecting the unique agricultural context of Bangladesh

The experimental findings have yielded promising results. The CNN model achieved a remarkable 99.72% accuracy in image recognition, underscoring the power of deep learning in this context. Notably, the SVM classifier achieved a noteworthy 94% accuracy, showcasing its effectiveness in leaf health assessment. Additionally, the ANN model demonstrated a substantial 95.62% accuracy, further highlighting the utility of artificial neural networks in this agricultural application. These achievements bear notable statistical significance when contrasted with the accuracies obtained in identifying healthy and unhealthy images derived from the dataset.

This research represents a significant stride in the quest to leverage advanced technology to safeguard the agricultural livelihoods of a nation, offering potential solutions to the pressing issue of leaf health assessment. In the following sections, we delve into the methodology, data sources, and detailed results that underpin these achievements, with the aim of contributing to the advancement of precision agriculture and crop management in Bangladesh and beyond.

1.2 History

The use of image processing to identify leaf diseases has been the subject of much research in the past, and this topic continues to draw in new investigators. Recent years have seen an increase in the use of image processing and machine learning for automatic crop disease identification.

P. Krithika et al., preprocessed using color-space conversion, contrast enhancement, and picture scaling. Using GLCM (Gray Level Co-Occurrence Matrix), K-Means clustering is carried out for segmentation and feature extraction. The multiclass SVM method was used for classification. Color space conversion was carried out by R. Meena et al., then an enhancement procedure was used. The leaf's primary hues are transformed into L∗A∗B∗. For segmentation, the K-Mean clustering technique is employed. For feature extraction and classification, respectively, the GLCM and SVM are employed. Bharat et al. employed a digital camera to capture the photos, and a median filter was applied to improve the photographs. Segmentation is accomplished by K-Mean clustering. For classification, SVM is employed. Segmentation is used, according to Pooja et al., to identify the regions of interest that contain the affected area. The k-Mean clustering approach is used, and the boundary and spot detection algorithm is used for Otsu's detection, which converts RGB to HSI before segmentation. Pre-processing was carried out by Rukaiyya et al. using contrast adjustment and normalization. Color transforms

are converted to YCBCR, and bi-level thresholding is carried out. To extract and classify features, the GLCM and HMM are utilized [2].

1.3 Related Work

The task of discriminating between healthy and unhealthy leaves in the context of precision agriculture has garnered substantial attention in recent years. Researchers and experts have employed various methodologies to address this critical challenge. In this section, we provide an overview of the relevant studies and approaches that have paved the way for our investigation.

1. Deep Learning in Agriculture

   The integration of deep learning techniques, particularly Convolutional Neural Networks (CNNs), has seen significant application in the domain of agriculture. Researchers have harnessed the power of CNNs to accurately detect and classify plant diseases based on leaf images. Notable studies have demonstrated the potential of CNNs in identifying a wide range of plant diseases, offering accurate and timely diagnosis for farmers.

2. Traditional Machine Learning Approaches

   Apart from deep learning, traditional machine learning techniques have also been explored for leaf health assessment. Support Vector Machines (SVMs) have emerged as strong contenders in image classification tasks. SVMs have been applied to distinguish between healthy and diseased leaves, achieving notable success in disease detection.

3. Integration of Multiple Models

   Some studies have delved into the fusion of multiple models, including CNNs, SVMs, and Artificial Neural Networks (ANNs), for enhanced leaf health assessment. These investigations have shown promise in leveraging the strengths of different classifiers to improve accuracy and robustness.

4. Leaf Datasets

   A crucial aspect of leaf health assessment is the availability of suitable datasets. Researchers have curated and made available diverse datasets comprising images of healthy and unhealthy leaves from various plant species. These datasets serve as valuable resources for training and evaluating machine learning models.

5. Agricultural Significance

It is important to underscore the broader significance of this research. Agriculture is the backbone of economies in many countries, including Bangladesh. The ability to precisely identify leaf diseases is not only critical for crop health but also for food security and economic stability. Studies in this domain have the potential to make a substantial impact on agricultural practices and the livelihoods of rural populations.

In the context of this related work, our research seeks to build upon and contribute to the existing body of knowledge. By exploring the integration of CNNs, SVMs, and ANNs in leaf health assessment, we aim to offer a comprehensive and effective solution for the unique agricultural landscape of Bangladesh. Our work strives to further advance the field of precision agriculture and aid in the sustainable management of crops, benefiting both local farmers and the national economy.

Let's now examine some of the prior research that has been conducted in this area:

- Plant leaf disease detection using computer vision and machine learning
  Algorithms (2022). The accuracy of the proposed model is tested using SVM (88%), K-NN (97%) and CNN (99.6%) on tomato disordered samples [3].
- Unhealthy Region of Citrus Leaf Detection Using Image Processing Techniques (International Conference for Convergence of Technology - 2014). ANN based classifier has been used for classification with recognition rate up to 91% [4].
- GUI based Detection of Unhealthy Leaves using Image Processing Techniques (International Conference on Communication and Signal Processing, April 4-6, 2019, India) the average accuracy of detection in SVM and ANN are 85% and 97% respectively [5].
- High Accurate Unhealthy Leaf Detection [1].
- Recognition of Unhealthy Plant Leaves Using Naive Bayes
  Classifier (IOP Conference Series: Materials Science and Engineering) classification gain obtained by normal pixel value is 49.37% and the affected pixel value is 51.04%. The uncovering accuracy is improved to 97% by using classifier [6].
- Plant Leaf Detection and Disease Recognition using Deep Learning
  The plant variety and the type of illness the plant was infected with could be detected and recognized with up to 100% accuracy by the system, with the trained model achieving an accuracy rate of 96.5 percent [7].
- Plant leaf detection using modified active shape models

Two deformable models were developed with pepper leaves: Boundary-ASM and MLP-ASM. Matching processes are carried out by deforming the trained leaf models to fit real leaf images collected in the greenhouse. MLP-ASM detected 76.7 and 87.8% of overlapping and occluded pepper leaves respectively, while Boundary-ASM showed detection rates of 63.4 and 76.7%. The detection rates by the conventional ASM were 23.3 and 29.3% [8].

- Diseases Detection of Various Plant Leaf Using Image Processing Techniques: A Review

  Currently farmers are spraying pesticides to the plants but it effects human directly or indirectly by health or also economically. To detect these plant diseases many fast techniques need to be adopt [9].

1.4 Motivation

The agricultural sector plays a pivotal role in the socio-economic landscape of Bangladesh, with over 70% of the country's population and 77% of its labor force residing in rural regions and being directly or indirectly involved in agriculture. In this context, the accurate identification of diseased leaves is of paramount importance, as it directly impacts crop yield, food security, and the livelihoods of millions. Timely and precise identification of plant diseases can help mitigate losses and ensure sustainable agricultural practices.

Modern machine learning techniques have opened up new avenues for improving plant disease diagnosis and have the potential to revolutionize the agricultural sector. Convolutional Neural Networks (CNNs), Support Vector Machines (SVMs), and Artificial Neural Networks (ANNs) are powerful tools with established success in various domains, including computer vision, pattern recognition, and image classification. Leveraging these technologies to develop an integrated solution for distinguishing between healthy and unhealthy leaves holds great promise.

The significance of this study lies in its innovative approach, which combines the strengths of three distinct classifiers—CNN, SVM, and ANN methodologies. This integration aims to enhance the accuracy and reliability of leaf disease identification, providing a multifaceted solution to a real-world problem. By bringing together these different models, we aim to harness their collective potential to address the challenges posed by plant disease identification. Furthermore, the development of a robust and accurate system for leaf disease identification can greatly benefit farmers, agronomists, and researchers by facilitating early disease detection and enabling timely intervention. This can result in improved crop health, increased

agricultural productivity, and reduced dependence on chemical treatments, which, in turn, can contribute to more sustainable and environmentally friendly farming practices.

In summary, the motivation for this work stems from the urgent need to address plant diseases that affect the livelihoods of millions in Bangladesh. By harnessing the power of advanced machine learning techniques, this research contributes to the development of a robust solution that can benefit both the agricultural community and the nation as a whole.

1.5 Objectives

Convolution Neural Networks, Support Vector Machines, Artificial Neural Networks have been used in this context to find the best accuracy of the unhealthy leaf. Which serves several important goals to enhance their well-being and communication. Among these goals are:

1. Develop an Integrated Model: The primary objective of this research is to design and implement an integrated model that combines Convolutional Neural Networks (CNNs), Support Vector Machines (SVM), and Artificial Neural Networks (ANNs) for the precise discrimination between healthy and unhealthy leaves in the context of agriculture. It may make us and the Agro-worker stronger economically.

2. High Accuracy: Achieve high accuracy in leaf health assessment using the developed integrated model. The research aims to demonstrate the effectiveness of this amalgamation of classifiers by obtaining accurate and reliable results. Which ultimately will take us more stable for decision making.

3. Data Curation: Meticulously curate training and test datasets from a bespoke repository, ensuring that the data accurately reflects the unique agricultural context of Bangladesh. The datasets will be representative of various plant species and leaf conditions.

4. Model Comparison: Compare the performance of the CNN, SVM, and ANN methodologies for leaf health assessment. Analyze the strengths and weaknesses of each model, emphasizing their individual contributions and synergies within the integrated model.

5. Statistical Significance: Evaluate the statistical significance of the achieved results by contrasting the accuracies obtained from the integrated model with those derived from individual classifiers. The objective is to establish the superiority of the integrated approach.

6. Practical Application: Explore the practical applicability of the integrated model in real-world agricultural scenarios. Consider how this research can contribute to precision agriculture practices and the well-being of rural populations heavily dependent on agriculture in Bangladesh.

7. Contribution to Agriculture: Highlight the potential contributions of this research to the field of agriculture and crop management. Emphasize how accurate leaf health assessment can impact crop yield, food security, and the livelihoods of rural communities.

1.6 How the dataset is collected

The dataset used for this research was meticulously curated, drawing from multiple sources to ensure its diversity and suitability for the task of leaf health assessment in the context of precision agriculture.

- Kaggle Dataset: To kick start the dataset collection process, a selection of pre-existing datasets was acquired from Kaggle, a well-known platform for data science and machine learning resources. These datasets were carefully chosen to encompass a variety of plant species and leaf conditions, offering a broad representation of agricultural scenarios.
- UCI Machine Learning Repository: Additionally, data from the UCI Machine Learning Repository, a reputable source for machine learning datasets, was incorporated into the collection. This source provided supplementary datasets that enriched the diversity of the samples, contributing to a more comprehensive and robust dataset for the study.
- Personal Data Collection: Recognizing the importance of tailoring the dataset to the unique agricultural landscape of Bangladesh, a portion of the data was collected locally. Utilizing mobile devices, images of leaves from common local plant species were captured. This process allowed for the inclusion of indigenous plant species and leaf conditions specific to the study's geographic focus.

The collected data underwent preprocessing to ensure consistency and quality, including image standardization, labeling, and the creation of training and testing sets. This multifaceted approach to dataset collection aimed to reflect the agricultural context of Bangladesh and provide a reliable foundation for the subsequent application of machine learning models in leaf health assessment.

CHAPTER 2
MACHINE AND DEEP LEARNING TECHNIQUES

2.1 General

Due to its widespread use, deep learning-based image recognition has become more prominent in the field of feature extraction and detection. A number of methods for recognizing image and face recognition systems have been presented. The methods for transforming a real-time face camera view or image into an extractable machine form are sufficiently studied and documented.

2.2 CNN

The structure of CNN's visual cortex enhances its resemblance to the neuronal transmission network found in the human brain. Nerve cells only see and analyze their responses to stimuli from the visible spectrum. Spectral field overlaps should be arranged to occupy the whole display area. For the purpose of recognition, a CNN can identify the spatial relationships inside an image by relating it to its content [10]. In order to identify the distinctive characteristics that allow a picture to be identified, the CNN design offers a good match of the original image. During the training phase, the weights, parameters, and biases associated with the transformations from the original picture to feature vector are discovered in order to get a better understanding of the image's nature. The process of CNN is depicted in **Figure 2.1** and is composed of the following basic components: convolution, pooling, fully connected neural network or dense layer; moreover, some layer is added to prevent overfitting and under fitting of the model.

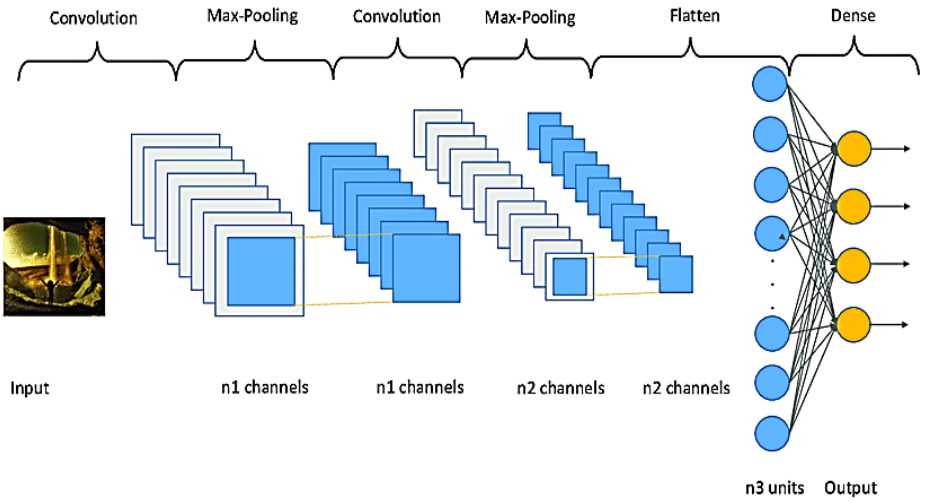


Figure 2.1 Process of CNN.

2.2.1 Convolution Layer

Convolution is primarily employed in image alteration processes including sharpening, feature augmentation, and smoothing of the camera. In this step, feature maps from the previous layer are convolutioned using memorizing kernels like as Gaussian or Gabor. The output maps are then produced by passing the resulting data through linear or non-linear activation functions such as Sigmoid, SoftMax, and ReLU. Convolution is an image processing technique that may be applied to sharpen, enhance, or augment pictures. To rebuild a picture, a matrix of numerical values known as a kernel is passed over the image.

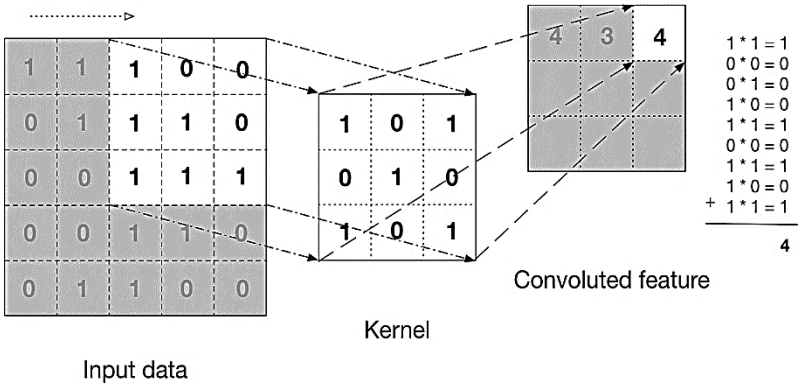


Figure 2.2 Illustration of Convolution Operation.

The first element in the output matrix is produced by multiplying the kernel element-by-element with the top-left pixels and adding the resulting values. It carries out this procedure all across the picture while shifting the filter to the right. The filter shape in RGB pictures is Fr x Fc x 3, while the input shape is (height x width x 3). The number of cells that the filter moves is called the "stride." Every filter segment is convolved with the relevant piece of the input

picture; the three values that result are then added together and regarded as the output. In both cases, we can apply more than one filter, and in such cases, we concatenate the result matrices. The final form in each of these cases is given as follows:

$(H - Fr + 1) \times (H - Fc + 1) \times N_c$----------eq^n (1)

The variables 'H' and 'Fr, Fc' in the convolution process represent the height and width of the input picture and filter, respectively. Concurrently, the variable 'N_c' denotes the quantity of filters involved in the procedure. In order to prevent picture shrinkage or loss of important information during the convolution process, padding is applied to the input image. There are two sorts of convolutions: one that maintains the input's form and the other that is a valid convolution without any padding. The identical convolution is used in the second kind, but padding is used to maintain the input's form. The following formula may be used to determine the output shape when padding is used:

$(H + 2PL - Fr + 1) \times (H + 2PL - Fc + 1) \times N_c$----------eq^n (2)

The remaining variables are the same as previously specified, and PL stands for the number of padding layers. Randomly initialized filters are used as learning parameters by CNNs, and then non-linearity is introduced by bias addition and ReLU activation. Regardless of the layer number, ReLU stops the same behavior. ReLU is used in this work. The activation function of the Rectified Linear Unit (ReLU) is expressed as follows:

$Out = max\ (0, Bias\_v)\ v$----------eq^n (3)

The layer output is denoted by Out, while the value obtained by integrating the bias value with the convolution output is represented by Bias_v. In this feature that triggers, change the input to a positive integer. This method reduces the dimensionality of the features by reshaping them in comparison to the original input.

If we have an input of size W x W x D and Dout number of kernels with a spatial size of F with stride S and amount of padding P, then the more simplify size of output volume can be determined by the following formula:

$W_{out} = \frac{W-F+2P}{S} + 1$----------eq^n (4)

This will yield an output volume of size Wout x Wout x Dout. More precisely, the convolutional operation with image and kernel to produce activation map is given below:

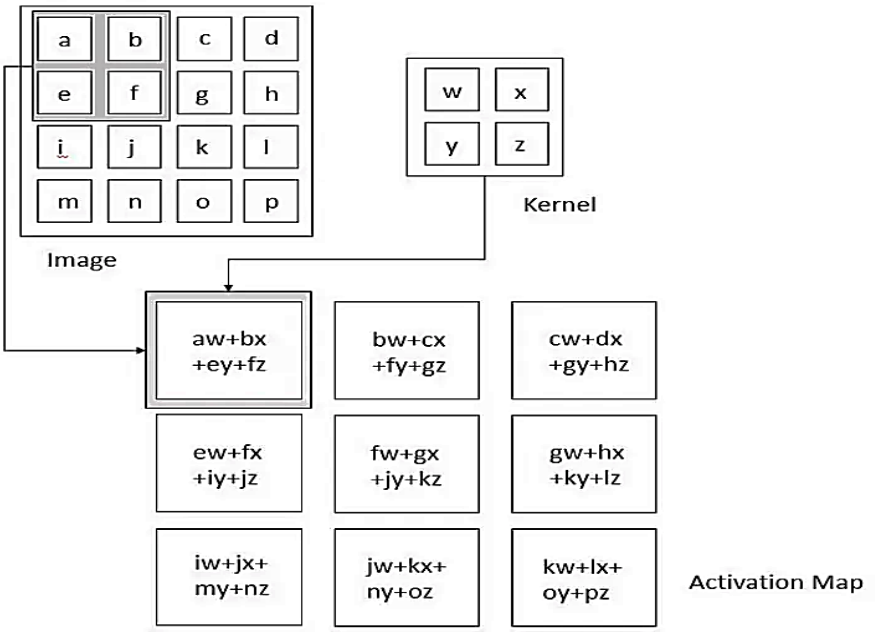


Figure 2.3 Convolutional Operation.

2.2.2 Pooling layer

By calculating a summary statistic from the outputs in the vicinity, the pooling layer substitutes the network's output at certain points. This aids in shrinking the representation's spatial size, which lowers the quantity of computation and weights needed. Each slice of the representation is handled independently for the pooling procedure.

The L2 norm of the rectangular neighborhood, the average of the rectangular neighborhood, and a weighted average depending on the distance from the center pixel are some examples of pooling functions. Nonetheless, the most widely used method is max pooling, which provides the neighborhood's maximum output.

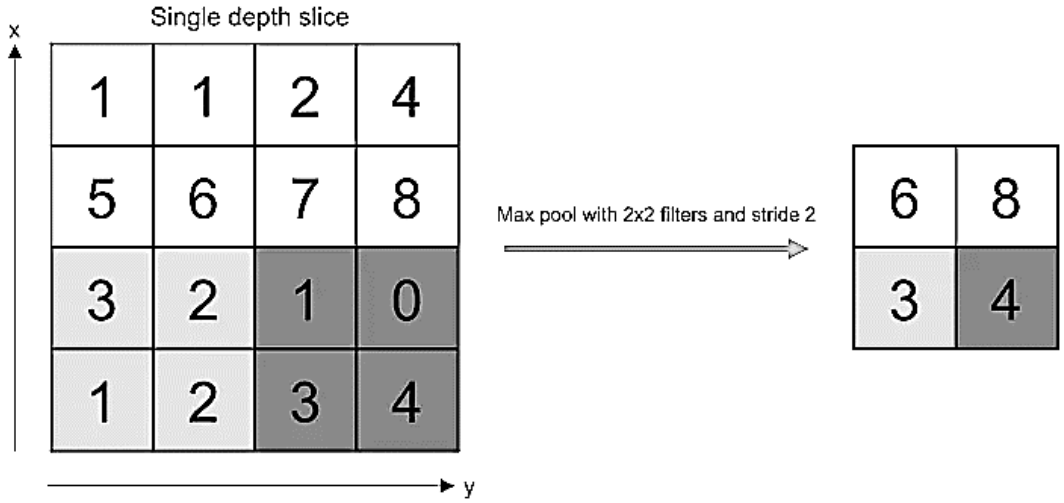


Figure 2.4 Pooling operation.

Assuming a pooling kernel of spatial size F, a stride of S, and an activation map of size W x W x D, the output volume may be calculated using the following formula:

$W_{out} = \frac{W-F}{S} + 1$----------eq^{n} (5)

An output volume measuring Wout x Wout x D will result from this. Pooling always offers some translation invariance, meaning that an item would always be recognized on the frame no matter where it appears.

Max Pooling and Average Pooling are the two varieties of pooling. The largest value from the area of the picture that the Kernel covers is returned by max pooling. Conversely, Average Pooling yields the mean of all the values from the area of the picture that the Kernel covers.

In addition, max pooling serves as a noise suppressant. In addition to dimensionality reduction, it does de-noising and completely eliminates the noisy activations. Conversely, Average Pooling just reduces dimensionality as a noise-suppressing technique. As a result, Max Pooling outperforms Average Pooling by a wide margin.

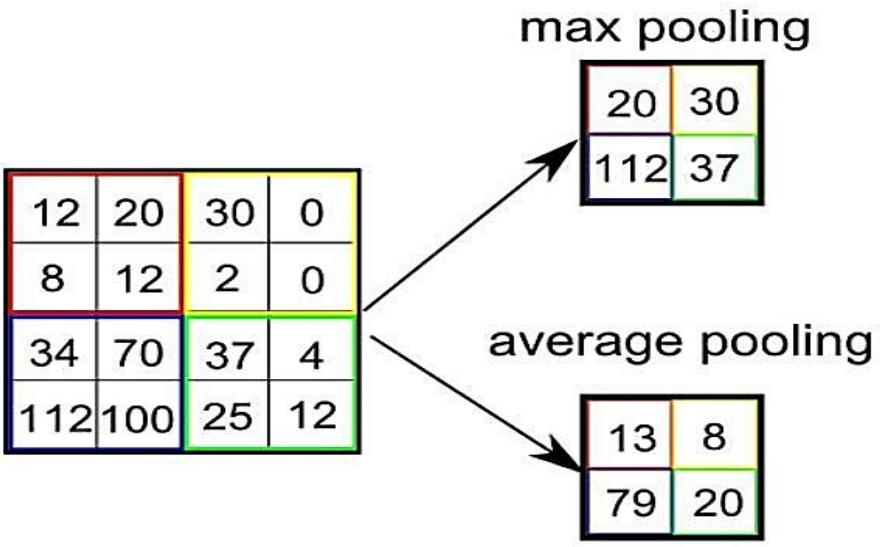


Figure 2.5 Types of pooling.

The i-th layer of a convolutional neural network is made up of the pooling layer and the convolutional layer. The number of these layers may be raised to capture even more minute details, but doing so will need more processing power, depending on how complex the pictures are.

2.2.3 Dense layer

The last layer of CNN uses either a dense layer or a fully linked neural network. All they are fully linked artificial neural networks. During the training phase, weights related to the network are calculated. After the convolution/pooling process is completed, a fully connected neural network determines which label best matches the image. This section establishes the link between the image's class and feature vector. The weights related to the network connection path are multiplied by the convolution/pooling operation's outputs. After that, the outcome is run via an activation function.

2.2.4 Dropout

Overfitting in the training dataset is typically a result of connecting every feature to the FC layer. Overfitting is the phenomenon when a certain model performs poorly when applied to fresh data since it performed so well on the training set. In order to solve this issue, a dropout Layer is used, in which a small number of neurons are removed from the neural network during training, reducing the size of the final model. Thirty percent of the nodes in the neural network are randomly removed after a dropout of 0.3. By simplifying the network and preventing overfitting, dropout enhances the performance of machine learning models. During training, it removes neurons from the neural networks. The **Figure 2.6** illustrates the dropout concept:

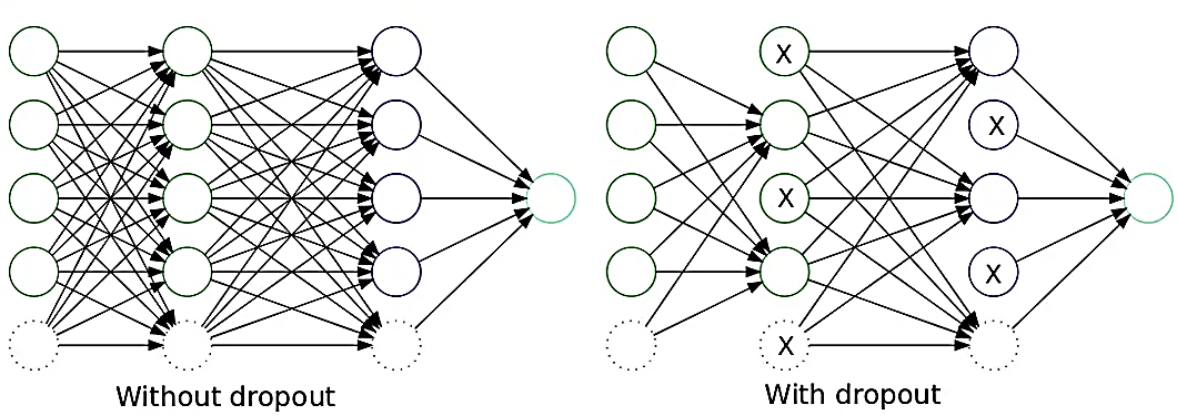


Figure 2.6 Characteristics with dropout and without dropout.

2.2.5 Activation Functions

Lastly, the activation function is among the CNN model's most crucial elements. They are employed in the learning and approximation of complicated and continuous relationships between network variable types. Put simply, at the end of the network, it determines which model information should shoot ahead and which ones shouldn't. It gives the network more nonlinearity. Numerous widely-utilized activation functions exist, including the Sigmoid, ReLU, SoftMax, and tanH functions. Every one of these features has a particular use. The sigmoid and SoftMax functions are recommended for a CNN model used for binary classification; for multi-class classification, SoftMax is typically employed. To put it simply, in a CNN model, activation functions decide whether or not to activate a neuron. It uses mathematical processes to determine the significance of the input to the work in terms of prediction [11].

2.2.5.1 Sigmoid

The mathematical version of the sigmoid non-linearity is σ (κ) = 1/ (1+e^κ). It converts a real-valued number into a range of values between 0 and 1.

However, a highly undesirable characteristic of the sigmoid is that the gradient nearly vanishes when the activation is at either tail. In backpropagation, an extremely tiny local gradient will essentially "kill" the gradient. Additionally, if the neuron receives only positive input, the Sigmond's output will include either positive or negative values, creating a zigzag pattern in the gradient updates for weight. The **Figure 2.7** illustrates sigmoid function:

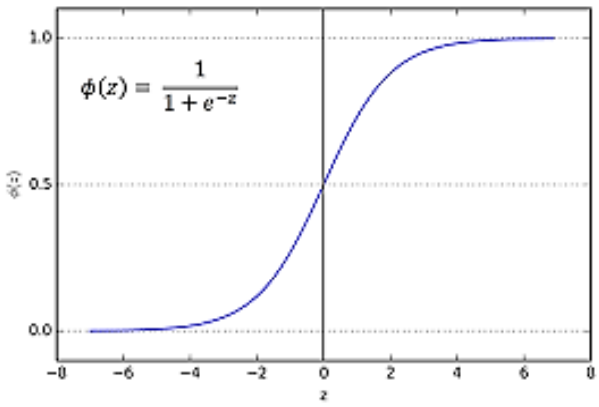


Figure 2.7 Sigmoid Function.

2.2.5.2 ReLU

In recent years, the Rectified Linear Unit (ReLU) has gained a lot of popularity. The function f(κ)=max (0, κ) is computed. Stated otherwise, the activation is only the threshold at zero. ReLU is more dependable than sigmoid and tanh, and it speeds up convergence by six times. The **Figure 2.8** illustrates sigmoid function:

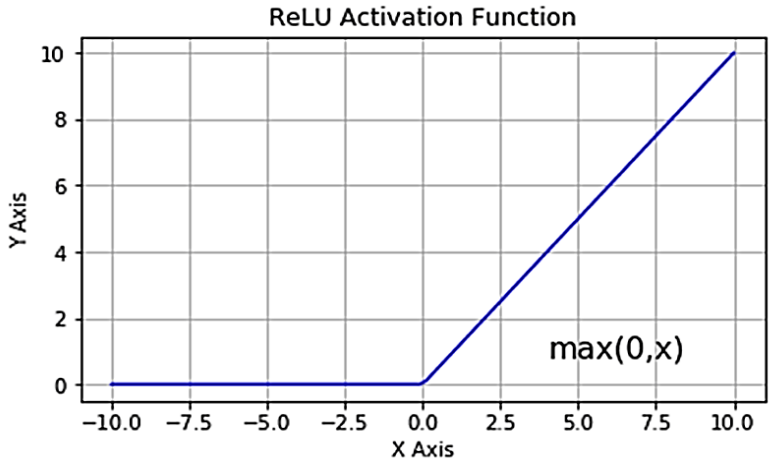


Figure 2.8 ReLU activation function.

2.2.5.3 SoftMax

Rather of utilizing ReLU, sigmoid, tanh, or another activation function, the SoftMax activation is typically given to the very last layer in a neural net. SoftMax is helpful since it effectively transforms the output of our neural network's last layer into a probability distribution. The **Figure 2.9** illustrates SoftMax function:

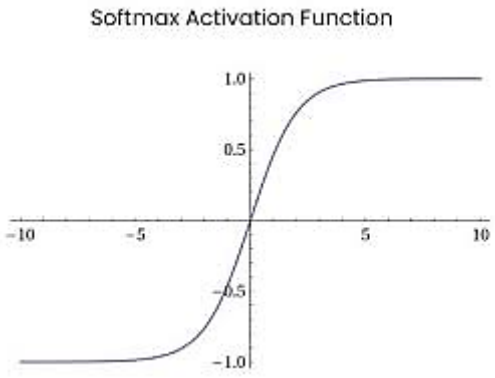


Figure 2.9 SoftMax activation function.

2.2.6 Batch Normalization

To begin with, remember that normalization is a pre-processing method for standardizing data. Stated differently, possessing disparate data sources within the same range. Our network may have issues if the data is not normalized before training, which would make training much more difficult and slow down the learning process. Let's say we have a car rental service, for instance. First, using data from rivals, we aim to forecast a reasonable pricing for every automobile. For each automobile, we have two features: the total number of kilometers driven and the age in years. These can vary greatly in both range and distance, from 0 to 30 years and 0 to hundreds of thousands of kilometers, respectively. These range disparities are undesirable for features since the value with the larger range may lead our models to overvalue them. To normalize our data, there are two primary approaches. Scaling it to a range of 0 to 1 is the easiest way to do this:

$$x_{normalized} = \frac{x-m}{x_{max}-x_{min}}$$ ----------eq^{n} (6)

m is the average value of the data collection, x_max is the greatest value, and x_min is the lowest value. X is the data point to normalize. This method is typically applied to data inputs. Wide ranges of non-normalized data points can lead to instability in neural networks. Relatively high inputs have the potential to cascade down to the layers, leading to issues like gradient explosions.

An alternative method of standardizing data involves utilizing the following formula to ensure that each data point has a mean of 0 and a standard deviation of 1. The formula is:

$$x_{normalized} = \frac{x-m}{s}$$ ----------eq^{n} (7)

Being x the normalization data point, m the data set mean, and s the data set standard deviation, every data point now resembles a typical normal distribution. With every feature on this scale, There won't be any bias, which will help our models learn more effectively. This last method is what we utilize in Batch Norm to normalize batches of data throughout the network.

Batch normalization is a normalizing method that is used between a neural network's layers as opposed to the raw data. Rather of using the entire data set, it is done along mini-batches. Learning is facilitated by its ability to expedite training and employ greater learning rates. Using the method outlined in the preceding section, we can define the Batch Normalization formula as follows:

$$z^{N} = \left(\frac{z-m_z}{s_z}\right)$$ ----------eq^{n} (8)

Where s_z is the output of the neurons' standard deviation and m_z is their mean [11].

2.3 SVM

A great deal of study has been done recently on support vector machines (SVMs) and how they are used in many scientific domains. SVMs are among the most effective and reliable regression and classification algorithms in a variety of application domains. In pattern recognition, one of the most well-liked and active study fields among scientists, the SVM has proved significant. Other applications of SVM, such as SVM for big data sets, SVM for multiclassification, and SVM for imbalanced data sets, have been developed as a result of research in some sectors where SVMs perform poorly. SVM has also been used with other cutting-edge techniques, such as evolve algorithms, to improve classification and optimize parameters. Research and applications in a number of scientific and technical fields have recognized SVM algorithms [12].

2.3.1 SVMs' theoretical foundation

In pattern classification, getting a model that maximizes the performance given the training data is the main goal. Using traditional training techniques, the models are determined such that every input-output pair is accurately categorized inside its own class. But if the classifier

is over fit for the training set, the model starts to learn from the training set by memorization rather than by generalization, which weakens the classifier's capacity to generalize.

SVM is primarily used to partition a training set of classes using a surface that optimizes the margin between them. To put it another way, SVM makes it possible to maximize a model's capacity for generalization. Instead of minimizing the mean squared error on the set of training data, as is frequently the case with empirical risk minimization methods, the Structural Risk Minimization principle (SRM) allows for the minimization of a bound on the generalization error of a model.

Support vector machines—when training data are linearly separable in the input space—and situations where they are not—are covered in this research.

2.3.2 Case of linear separability

An SVM needs n instances to be trained. Every example comprises two elements: an input vector (x_i) and its corresponding label (y_i). Consider the following: Let X be a training set.

$(x1, y1), (x2, y2), \ldots \ldots, (X_n, Y_n)$ ----------eq^n (9)

i.e., $X = \{xi, yi\}_{i=1}^{n}$ where $x_i \in R^d$ and $y_i \in (+1, -1)$. For reasons of visualization, we will consider the case of a two-dimensional input, i.e. $x \in R^2$. The data are linearly separable and there are many hyperplanes that can perform the separation.

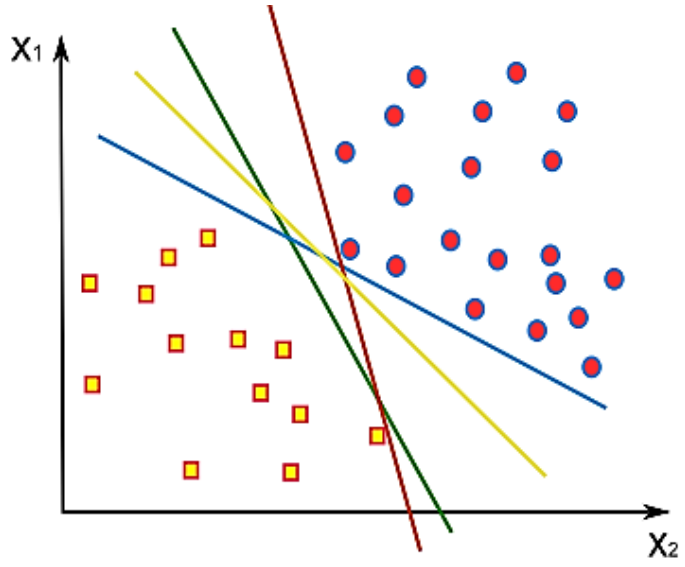


Figure 2.10 Separation hyperplanes.

A number of decision hyperplanes that precisely divide the input data set are displayed in Fig. 2.10. It is evident that an endless number of hyperplanes might accomplish this task. The separation hyperplane's and the maximum margin hyperplane's locations, however, determine

the generalization capacity. Optimal separation hyperplane is the name given to this hyperplane. The Equation defines the decision level, or the hyperplane dividing the input space is defined by,

$w^t x_i + b = 0$----------eq^n (10)

The linearly separable case in the feature space is the most basic case of SVM. We can optimize the geometric margin by setting the functional margin $kappi = 1$ (also known as Canonical Hyperplane), therefore, the linear classifier is $y_i = 1$.

$w.x^+ + b = 1$---------- -------eq^n (11.a)

$w.x^- + b = -1$----------eq^n (11.b)

These can be summed into a set of inequalities:

$y_i(w.x_i) + b \geq 1 \forall i$

The geometric margin of $x^+ \mathcal{Y}\ x^-$ is

$y_i = \frac{1}{2}(\frac{w}{||w||}.x^+ - \frac{w}{||w||}.x^-)$----------eq^n (12)

$= \frac{1}{2||w||}[(w.x^+) - (w.x^-)]$

$= \frac{1}{||w||}$

Where b represents the bias and w is the ideal separation hyperplane. Margin is the distance measured between the hyperplane and the training data point nearest to the hyperplane. If the ideal separation hyperplane is chosen as the separation hyperplane, the capacity for generalization is maximized. Reducing the norm of the vector of weights is the process of optimizing the geometric margin. In order to solve the quadratic programming issue, we look for the best hyperplane as well as two parallel hyperplanes (H1 and H2).

There is no data between the two hyperplanes, and the distance between H1 and H2 is maximum. Certain data points may be over H1 and others over H2, depending on how far apart H1 and H2 are from one another. These data points are referred to as support vectors. The other points can be added, removed, or altered without crossing the planes H1 and H2, and they won't affect the classifier's ability to generalize in any way because they directly contribute to the definition of the separation hyperplane; therefore, an SVM's solution can be found solely by using this limited set of support vectors.

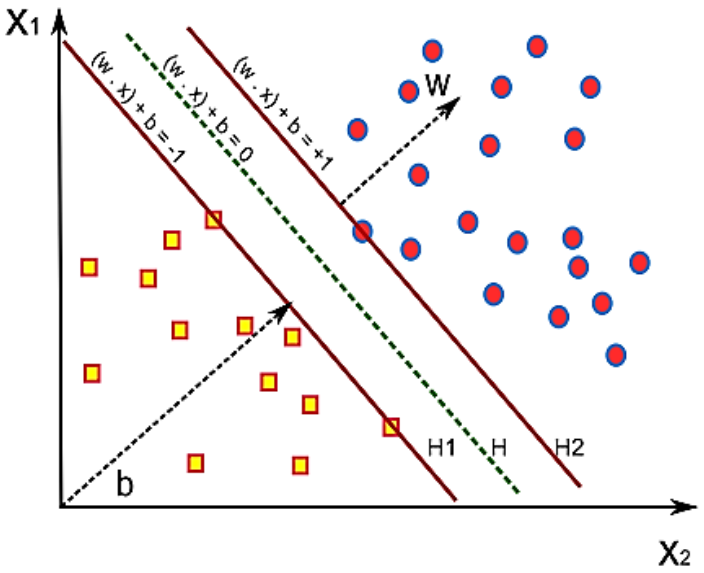


Figure 2.11 Optimal Classifier.

w; x and b, where w is a vector perpendicular to the hyperplane, can be used to represent any hyperplane. The geometric representation of the quadratic programming issue is displayed in Figure 2.11, along with the hyperplanes H1 and H2 and H (the optimum separator). Thus, the first optimization issue is as follows.

Proposition 1. For the linearly separable case $S = [(x_1, y_1) \ldots (x_l, y_l)]$, if (w, b) is the solution

$$\min_{w,b}(w.w) = ||w||^2$$

Subject to: $\gamma_i(w.x_i) + b \geq 1$

Then the maximal margin is given by $\gamma = \frac{1}{||w||}$----------eq^n (13)

We apply the Lagrange formulation to transform this into the dual issue. This is being done for two reasons. The first is that Lagrange multipliers, which are considerably simpler to deal with, will be used in place of the criteria listed in Eq. (5). The training data will only be present as the dot product between vectors in the second reformulation of the issue. This is an essential characteristic that will enable the process to be broadly applied in the nonlinear scenario. In this way, the Lagrangian is given by:

$$L(w, b, \alpha) = \frac{1}{2}(w.w) - \sum_{i=1}^{l} \alpha_i [y_i((w.x_i) + b - 1)]$$----------eq^n (14)

Where α_i are the Lagrange's multipliers.

The dual is found in these two steps: first, taking the derivative with respect to w and b

$$\frac{\partial L(w,b,\alpha)}{\partial w} = w - \sum_{i=1}^{i} \alpha_i y_i x_i = 0 \rightarrow w = \sum_{i=1}^{l} \alpha_i y_i x_i$$ ----------eq^n (15)

$$\frac{\partial L(w,b,\alpha)}{\partial b} = -\sum_{i=1}^{l} \alpha_i y_i = 0 \rightarrow \sum_{i=1}^{l} \alpha_i y_i = 0$$----------eq^n (16)

And second, substituting Equations (7) and (8) in the original Lagrangian (6)

$$L(w,b,\alpha) = \frac{1}{2}(w.w) - \sum_{i=1}^{l}\alpha_i[y_i((w.x_i) + b - 1)]$$----------eq^{n}(17)

$$= \frac{1}{2}(\sum_{i=1}^{l}\alpha_i \,.\, \sum_{i=1}^{l}\alpha_i y_i x_i) - \sum_{ij=1}^{l}\alpha_i\alpha_j y_i y_j((x_j.x_i) + b) - \sum_{i=1}^{l}\alpha_i$$

$$= \frac{1}{2}\sum_{ij=1}^{l}\alpha_i y_i \alpha_j y_j (x_i.x_j) - \sum_{ij=1}^{l}\alpha_i y_i \alpha_j y_j (x_j \,.\, x_i) - \sum_{i=1}^{l}\alpha_i y_i b + \sum_{i=1}^{l}\alpha_i$$

$$= -\frac{1}{2}\sum_{ij=1}^{l}\alpha_i y_i \alpha_j y_j \,(x_i \,.\, x_j) + \sum_{i=1}^{l}\alpha_i$$----------eq^{n}(18)

The data points with $\alpha_i > 0$ are called "support vectors" and these vectors define the hyperplanes H1, H2. At all the other training data $\alpha_i = 0$. The training data's most important components are the support vectors, which are the ones that are closest to the decision hyperplane.

Remark 1. By training the original data set, support vector machines (SVMs) create a hyperplane that precisely divides the data and is specified by a limited number of support vectors. Finding the same hyperplane defined by the same collection of support vectors would result from eliminating (or moving about without crossing) any other points (non-support vectors) and repeating the training. As a result, the following is the original optimization issue.

2.3.3 Karush-Kuhn-Tucker conditions

Because they specify the prerequisites for obtaining an ideal solution to a general optimization problem, Karush-Kuhn-Tucker conditions (KKT) are crucial to the theory of optimization.

Theorem 1. Given an optimization problem with convex domain $\Omega \subseteq \mathbb{R}^n$,

Minimize $f(w), w \,\epsilon\, \Omega$

s.t. $g_i(w) \leq 0, i = 1,\ldots..,k,$----------eq^{n}(19)

$h_i(w) = 0, i = 1,\ldots..,m,$----------eq^{n}(20)

2.3.4 Soft margin hyperplanes

When the data are linearly separable, that is, when there are no crossings in the training data set, the learning issue that was previously described holds true. These issues do arise occasionally in real life, though. However, there are some situations when the linear separation hyperplane can produce satisfactory outcomes even in the presence of crossings in the data set. However, in the situation of intersection, the above-mentioned quadratic programming techniques cannot be applied since the condition $y_i((w\,.x_i) + b) \geq 1, \forall I$ can not be satisfied in the case of intersection. The intersection's points cannot be properly categorized, and any incorrectly classified data x_i will cause its associated α_i to drift toward infinity [10].

2.4 Kernels

The ideal hyperplane in an SVM is chosen to optimize the model's capacity for generalization. Even if the hyperplanes are optimally determined—that is, the original input space is converted into a highly dimensional space known as "feature space" in order to maximize the space between classes—the classifier obtained may not have a high degree of generalization ability if the training data are not linearly separable.

The basic idea in designing non-linear SVMs is to transform the input vectors $x \epsilon R^n$ into vectors $\Phi(x)$ of a highly dimensional feature space F (where Φ represents the mapping: $(\mathbb{R}^n \rightarrow \mathbb{R}^f)$ and solve the problem of linear classification in the feature space

$x \epsilon \mathbb{R}^n \rightarrow \Phi(x) = [\emptyset_1(x), \emptyset_2(x), \ldots\ldots, \emptyset_n(x)]^T \epsilon \mathbb{R}^f$ ----------eq^n (21)

2.5. Mercer's condition

Mercer's theorem determines the conditions of functions to be kernels. Given a finite input space $X = \{x_1, \ldots\ldots\ldots, x_n\}$ and assuming that $K(x, z)$ is a symmetric function of then

$K = (K(x_i \,.\, x_j)_{ij=1}^n$ ----------eq^n (22)

Since K is symmetric there exists an orthogonal matrix V such that K ¼ VKV0, where K is the diagonal matrix that contains the eigenvalues [12].

2.5 ANNs

Artificial Neural Networks, often referred to as neural networks or simply ANNs, are a class of machine learning models inspired by the structure and function of biological neural networks in the human brain. ANNs are at the forefront of contemporary machine learning and have witnessed immense growth in applications across various domains, including image and speech recognition, natural language processing, recommendation systems, and much more. They are a fundamental component of deep learning, a subset of machine learning concerned with neural networks featuring multiple layers, known as deep neural networks [13].

Structure and Components of Artificial Neural Networks:

1. Neurons (Nodes): The fundamental building blocks of ANNs are artificial neurons, also called nodes or perceptron's. Each neuron receives input data, performs computations, and produces an output. Neurons are organized into layers, typically divided into three categories: input layer, hidden layers, and output layer.
2. Weights and Biases: The connections between neurons are characterized by weights and biases. Weights determine the importance of the input, and biases help in adjusting

the output. Training an ANN involves adjusting these weights and biases to learn patterns and relationships in the data.

3. Activation Functions: Activation functions introduce non-linearity into the neural network, enabling it to model complex relationships. Common activation functions include the sigmoid, hyperbolic tangent (tanh), and rectified linear unit (ReLU) functions.

Feedforward and Backpropagation:

In a feedforward neural network, data flows in one direction, from the input layer through the hidden layers to the output layer. The network processes the input and produces an output. Training an ANN is typically accomplished through backpropagation, a process where the network's predictions are compared to the actual target values, and errors are propagated backward through the network to adjust the weights and biases, optimizing the network's performance.

Deep Learning and Deep Neural Networks:

While traditional ANNs contain a single hidden layer, deep neural networks comprise multiple hidden layers, giving rise to deep learning. Deep learning has revolutionized the field of artificial intelligence by enabling the modeling of intricate, high-dimensional data. Deep neural networks are capable of learning hierarchical features, making them highly effective in tasks such as image and speech recognition.

CHAPTER 3
DATASET DESCRIPTION

3.1 Dataset and It's importance

The selection and curation of the dataset utilized in this research are of paramount significance. A dataset represents the bedrock upon which machine learning models are constructed, and its quality and relevance directly impact the efficacy and practical applicability of the research. In the context of leaf health assessment for precision agriculture in Bangladesh, the dataset is not merely a technical necessity; it serves as a vital bridge between the research findings and the agricultural landscape of the region [14].

- Foundation for Machine Learning Models

  Machine learning models, including Convolutional Neural Networks (CNNs), Support Vector Machines (SVM), and Artificial Neural Networks (ANNs), rely on data for learning and prediction. The dataset provided the foundational training and testing data for these models, enabling them to recognize patterns and make informed classifications of leaf health. Without a comprehensive and diverse dataset, the performance of these models may be compromised.

- Real-World Relevance

  Given the agricultural context of Bangladesh, it was imperative that the dataset reflect the local plant species, leaf conditions, and agricultural scenarios. Incorporating data from existing sources such as Kaggle and the UCI Machine Learning Repository was the initial step, allowing for a broader representation of leaf conditions. However, the dataset's relevance to the region was further enhanced by the inclusion of data collected locally, using mobile devices to capture images of indigenous plant species. This strategy ensured that the dataset was more representative of the unique agricultural landscape of Bangladesh.

- Generalization and Practical Applicability

  A well-structured and diverse dataset facilitates the generalization of machine learning models. It allows the models to recognize patterns and relationships that extend beyond the data within the dataset, thus enabling practical applications in real-world scenarios. In the context of precision agriculture, this is essential as the research outcomes should not only demonstrate high accuracy but also be readily applicable to the challenges faced by local farmers.

- Ethical Considerations

  Ethical data sourcing and minimizing biases are vital considerations in contemporary research. A carefully curated dataset was instrumental in addressing ethical concerns, ensuring that the data used in the study was collected and handled with integrity and fairness.

- Contributions to the Field

  This dataset, compiled from various sources and enhanced with locally collected data, is also a contribution to the field of precision agriculture. By making it available for further research and application, this work seeks to empower researchers and practitioners with a valuable resource tailored to the agricultural context of Bangladesh [15].

3.2 My Dataset

The dataset employed in this research is a comprehensive and meticulously curated resource designed to facilitate the task of leaf health assessment for precision agriculture in Bangladesh. The dataset parameters are as follows:

- Number of Images: The dataset comprises a substantial collection of 55,500 images. This extensive sample size ensures the diversity and representativeness necessary for training and evaluating machine learning models.
- Number of Categories: The dataset is structured into two distinct categories, which are essential for the task of leaf health assessment. The categories are defined as 'Healthy' and 'Unhealthy,' representing the primary classifications for leaf conditions.
- Image Size: All images within the dataset are standardized to a resolution of 100x100 pixels. This uniform image size simplifies the processing and analysis of data while maintaining consistency in the dataset.
- Image Format: Images in the dataset are represented in the RGB (Red, Green, and Blue) format. The RGB color space is the most common choice for images and is well-suited for visual recognition tasks, such as leaf health assessment.
- Label Format: The labels associated with each image are represented as strings. The labels 'Healthy' and 'Unhealthy' serve as categorical descriptors, facilitating the classification of leaves based on their health condition.

Figure 3.1 Some of my dataset's images.

This dataset's extensive size, two-category classification, standardized image size, RGB format, and categorical labeling make it a robust resource for the research. Its diversity and relevance to the local agricultural context of Bangladesh ensure that the dataset provides a strong foundation for the training and evaluation of machine learning models in the domain of leaf health assessment.

In the subsequent sections of this research, the dataset will be employed to train and test the integrated model, evaluating its efficacy in accurately classifying leaves as 'Healthy' or 'Unhealthy.' The large number of images and clear categorization serve to enhance the model's accuracy and its applicability to real-world agricultural scenarios, reinforcing the dataset's significance in this research.

CHAPTER 4
TOOLS AND TECHNOLOGY USED

4.1 Development Tools

For performing this research, I used two developing platforms.

- Colab (Online platform with GPU support)
- Kaggle.com (Online platform with GPU support
- Jupyter Notebook (Offline platform with CPU support)

4.1.1 Google Colab (Colabratory)

Google Colab, short for Colaboratory, is a cloud-based platform that provides free access to a Jupyter Notebook environment with access to high-performance GPUs (Graphics Processing Units). It is an invaluable tool for researchers and data scientists, offering a convenient way to develop, execute, and share machine learning and data analysis projects. With its seamless integration with Google Drive and the ability to run Python code in a browser, Google Colab simplifies collaborative research and computational tasks, making it an ideal choice for this thesis research [16].

4.1.2 Kaggle.com

Under Google LLC, Kaggle is an online community for data scientists and machine learning practitioners as well as a platform for data science competitions. With Kaggle, users may collaborate with other data scientists and machine learning experts, search and post datasets, explore and develop models in a web-based data science environment, and participate in contests to solve data science problems. I used this platform for training the model and extract the feature.

4.1.3 Jupyter Notebook

Using the free and open-source online tool Jupyter Notebook, researchers may produce and distribute documents with live code, equations, graphics, and narrative prose. It is a well-liked instrument for machine learning, scientific computing, and data science. Documents in Jupyter Notebooks are saved in (.ipynb) file format. These files are viewed in a web browser and contain both text and code. Because Jupyter Notebooks are interactive, you can run code and see the results right away. Furthermore, the Jupyter Notebooks incredibly versatile. I used Jupyter Notebook for detecting people's facial emotion by accessing camera or pre-recorded human video.

4.2 Technology Used

4.2.1 TensorFlow

An open-source software package called TensorFlow is used to compute numbers using data flow graphs. Although scientific computing has also made use of it, machine learning and deep learning are its main applications. Google was the original developer of TensorFlow, which was made available as open-source software in 2015 under the Apache License 2.0. TensorFlow's efficiency, scalability, and flexibility make it a popular option for deep learning and machine learning applications. Models may be trained and deployed on several platforms, including as CPUs, GPUs, and TPUs. A vast array of tools and frameworks are also available with TensorFlow for creating deep learning and machine learning models.

TensorFlow uses data flow graphs as one of its primary features. Computations are shown as a network of nodes and edges in data flow graphs. Every edge depicts the data flow between nodes, and every node represents a mathematical process. TensorFlow is very scalable and efficient because to data flow graphs. TensorFlow operates on explicit data flow graphs, which allow for computation optimization and parallelization. Additionally, distributed training is supported by TensorFlow, enabling you to train models across several devices [17].

4.2.2 OpenCV

A one of the free and open-source libraries for computer vision and machine learning is called OpenCV (Open-Source Computer Vision Library). It offers a plethora of features for object identification and recognition, video analysis, and picture processing. Python, Java, and other programming languages may be used with OpenCV, which is built in C++. A strong and adaptable tool for computer vision and machine learning is OpenCV. It has several uses and is employed by both practitioners and researchers [18].

CHAPTER 5
METHODOLOGY AND SYSTEM MODEL

5.1 Basic Approach

The research methodology for leaf health assessment comprises three sequential phases, each building upon the previous one. These phases are executed in the Google Colab environment, which provides a powerful cloud-based platform for developing and executing machine learning projects.

1. Image Detection:

In the initial phase, the focus is on image detection. This involves the acquisition and preprocessing of leaf images. The steps involved in this phase include:

Data Collection: The dataset is collected from various sources, including Kaggle, the UCI Machine Learning Repository, and locally captured images from mobile devices. This diverse dataset provides a rich source of leaf images.

Data Preprocessing: The collected images are standardized, resized to a consistent 100x100 pixel resolution, and formatted in the RGB color space. This ensures that all images are in a uniform format, ready for further analysis.

Data Labeling: Each image is labeled according to its leaf condition, with labels 'Healthy' or 'Unhealthy' represented as strings. This categorization facilitates the subsequent phases of the research.

2. Image Recognition:

With the preprocessed dataset in hand, the research proceeds to the image recognition phase. In this stage, machine learning models are employed to recognize patterns and features within the leaf images. Key steps include:

Model Development: Machine learning models, such as Convolutional Neural Networks (CNNs) and Support Vector Machines (SVMs), are developed to learn from the dataset. These models are designed to recognize and extract relevant features from the leaf images.

Training: The developed models are trained on the labeled dataset. They learn to differentiate between healthy and unhealthy leaves by recognizing visual cues and patterns within the images. The training process optimizes the model's performance for accurate recognition.

Evaluation: The trained models are evaluated using a testing dataset. The performance metrics, including accuracy, precision, recall, and F1-score, are used to assess the models' ability to

recognize leaf conditions. This phase determines the models' effectiveness in image recognition.

3. Image Classification:

The final phase of the research involves image classification, where the machine learning models are put to practical use in classifying new, unseen leaf images. The primary tasks in this phase include:

Real-World Application: The trained models are applied to classify leaf images in real-world scenarios, simulating their practical use in precision agriculture. The models provide automated classification of leaves as 'Healthy' or 'Unhealthy,' contributing to crop management decisions.

Validation: The model's classifications are validated against ground truth data to ensure their accuracy in practical applications. This validation phase confirms the reliability of the models for real-world leaf health assessment.

Contribution to Precision Agriculture: The research findings in this phase emphasize the significance of accurate image classification in precision agriculture, where the well-being of crops and the livelihoods of rural communities are directly impacted by the results.

The three sequential phases of image detection, image recognition, and image classification conducted in Google Colab constitute the core methodology of this research. This systematic approach is designed to advance the field of precision agriculture and contribute to the accurate assessment of leaf health in the context of Bangladesh.

5.2 Data Preparation

Data preparation is a critical initial step in building and training machine learning models for leaf health assessment. This phase involves setting up the environment and enhancing the dataset for effective model training.

5.2.1 Importing Required Libraries

NumPy: NumPy is a fundamental Python library for scientific computing. It provides support for working with arrays, matrices, and mathematical operations. In the context of our research, NumPy is used to handle and manipulate image data efficiently [19].

TensorFlow: TensorFlow is an open-source machine learning framework developed by Google. It offers a comprehensive set of tools for building and training machine learning models, including deep neural networks. TensorFlow is a primary library for constructing and training convolutional neural networks (CNNs) in our research [17].

Keras: Keras is an open-source deep learning API that is part of TensorFlow. It provides a high-level, user-friendly interface for building and training neural networks, including convolutional neural networks (CNNs). Keras is commonly used for rapid prototyping and experimentation with deep learning models [20].

OpenCV: OpenCV (Open-Source Computer Vision Library) is a powerful computer vision library that provides a wide range of functions and tools for image processing and analysis. It can be valuable for tasks such as image preprocessing and feature extraction.

Scikit-Learn: Scikit-Learn is a machine learning library for Python that offers a variety of tools for data preprocessing, feature selection, model evaluation, and more. It's particularly useful when working with traditional machine learning algorithms alongside deep learning.

Matplotlib and Seaborn: Matplotlib and Seaborn are popular Python libraries for data visualization. They enable you to create informative and visually appealing plots and charts to help visualize our data and research findings [21].

5.2.2 Data Augmentation

Data augmentation is a key technique used to diversify the dataset by applying various transformations to the images. These transformations include rotation, horizontal and vertical shifts, shearing, zooming, and flipping. Data augmentation is valuable for increasing the dataset's variability, which is crucial for training models to handle real-world variations in leaf images.

The application of data augmentation is part of the preprocessing stage. By creating augmented versions of the images, you ensure that the machine learning models have a more extensive and diverse training set. This, in turn, helps improve the models' ability to generalize and make accurate predictions on unseen data [22].

In the subsequent phases of our research, the carefully prepared dataset, with its varied and augmented data, will serve as the basis for training and evaluating machine learning models. This dataset diversity is essential for robust and accurate leaf health assessment in precision agriculture.

"Data augmentation" entails creating altered copies of pre-existing datasets in order to increase the train set's size. Typically, this procedure entails making small adjustments to the data or employing a deep neural network to generate new data points. By artificially growing the face emotion recognition dataset, the suggested strategy avoids overfitting, boosts model accuracy, and lowers the expenses related to labeling and cleaning the raw dataset. Since data

modification may generate variance that reflects the methods used to identify face emotions, a variety of techniques were used for data augmentation.

- Featurewise_center=False: Using feature-wise normalization this parameter sets the input mean to zero over the entire dataset.
- Samplewise_center=False: The sample-wise centre parameter sets each sample's mean to false per sample.
- Featurewise_std_normalization=False: Transform the input by dividing each feature by the dataset's standard deviation in a feature-wise manner.
- Samplewise_std_normalization=False: For each sample, divide the inputs by the standard deviation of the dataset using sample-wise.
- Zca_whitening=False: Apply ZCA whitening using zca_whitening.
- Rotation_range: We will rotate our training images by 15 degrees.
- Zoom_range: Randomly apply a 1% zoom to the training image.
- Height_shift_range and width_shift_range: We introduced variation in the dataset by randomly shifting the images in both height and width by 10%.
- Horizontal_flip and vertical_flip: We have set the random flipping of images for both vertical and horizontal orientations to false [23].

5.2.3 Loading Dataset

```
train_datagen = ImageDataGenerator (#Here the augmented data is stored in the train_datagen variable)
train_generator = train_datagen.flow_from_directory('/dataset/train',
                                        batch_size=128,
                                        class_mode='categorical',
                                        target_size= (100,100),
                                        )
test_generator = ImageDataGenerator(rescale=1./255).flow_from_directory(
                                        '/dataset/test',
                                        batch_size=128,
                                        class_mode='categorical',
                                        target_size=(100,100)
                                         )
```

```
val_generator = ImageDataGenerator(rescale=1./255).flow_from_directory(
                                    '/dataset/val',
                                    batch_size=128,
                                    class_mode='categorical',
                                    target_size= (100,100)
                                     )
```

Here the batch size 128 means the number of samples processed before the model is updated. By loading train, test, and validate data from the dataset, the given output is generated:

Found 35000 images belonging to 2 classes.

Found 10250 images belonging to 2 classes.

Found 10250 images belonging to 2 classes.

5.3 System Model

The CNN is made up of three different kinds of layers: convolutional layers, pooling, and fully connected layers. A CNN architecture is created when these layers are stacked. The proposed CNN-based deep learning model has 12 layers and is sequentially connected to all of them. Where there are 5 convolutional layers, 3 pooling layers, and 4 dense layers. CNN model for the first layer is an input layer that contains some parameters such as filter size, kernel size, input shape, activation function. It is defined in this research that the filter size is 64 and the kernel size is 3X3 also define the input shape (64, 64, 3) in which 64 x 64 indicates the image width and height and 3 represent the number of color channels here 3 for RGB image and also define the activation function in our model for ReLU. Then add two-dimensional convolutional layer which for define filter size 64 and activation function ReLU. Then add another special layer called pooling layer. This layer's main goal is to lower the convolved feature map size to save on computational costs. It contains two types of layers one is Max-Pooling layer another is AVG-Pooling layer. In this model, I add Max-Pooling layer with 2x2 pool-size. Then I add another two special layer Batch Normalization layer and Dropout layer which prevent the model from being over fit and under fit. Here Batch Normalization works for batch normalizing its own mean and standard deviation and Dropout which help to eliminate some neuron from particular layer. In this model I use 0.25 Dropout value which indicates the 25% neuron elimination. After this I add back-to-back two convolutional layers with filter size 128 other parameters same as the previous convolutional layer. Then add another convolutional layer with filter size 256. Then add another Max-Pooling layer with pool size 2x2 after this in addition Batch Normalization layer and also include Dropout layer with threshold value 0.25.

After add Flatten layer which converts the two-dimension input from one dimension.it is mainly used at junction of convolutional layer and dense layer. Then add dense layer which is used to connect the neurons between two layers. The Dense layer, which also includes weights and biases, is utilized. These layers make up the final few layers of a CNN Architecture and are often positioned before the output layer. In our model we use three dense layers before the output dense layer. The first dense layer unit size 512, activation function is ReLU and also uses another hyper parameter called kernel regularize it also prevents the model from over-fit based on their threshold value. The second dense layer unit size is 256 and other parameters are same previous layer. The third dense layer unit size is 128 and other values same as previous layer. Then also used dropout layer. After this the last dense layer serve as model output. Which for unit size is the total number of classes for classified and also contain activation function as define SoftMax. In this research, CNN based proposed model has total number of parameters 9111367 and trainable number of parameters 9110471 out of total parameters and non-trainable number of parameters 896. All the parameters to establish my proposed Facial Emotion Recognition compound characters model are mentioned in the **Table 1** and also shows the architecture of my model in **Figure 5.1**.

Table 1: **Proposed CNN Model for internal parameter**

Layer (type)	Output Shape	Param #
Conv2D	(None, 64, 64, 64)	1792
Conv2D	(None, 64, 64, 64)	36928
MaxPooling	(None, 32, 32, 64)	0
Batch	(None, 32, 32, 64)	256
Dropout	(None, 32, 32, 64)	0
Conv2D	(None, 32, 32, 128)	73856
Conv2D	(None, 32, 32, 128)	147584
MaxPooling	(None, 16, 16, 128)	0
Batch	(None, 16, 16, 128)	512
Dropout	(None, 16, 16, 128)	0
Conv2D	(None, 16, 16, 256)	295168
MaxPooling	(None, 8, 8, 256)	0
Batch	(None, 8, 8, 256)	1024
Dropout	(None, 8, 8, 256)	0
Flatten	(None, 16384)	0
Dense	(None,512)	8389120
Dense	(None,256)	131328
Dense	(None,128)	32896
Dropout	(None,128)	0
Dense	(None,7)	903

Total params: 9,111,367

Trainable params: 9,110,471

Non-trainable params: 896

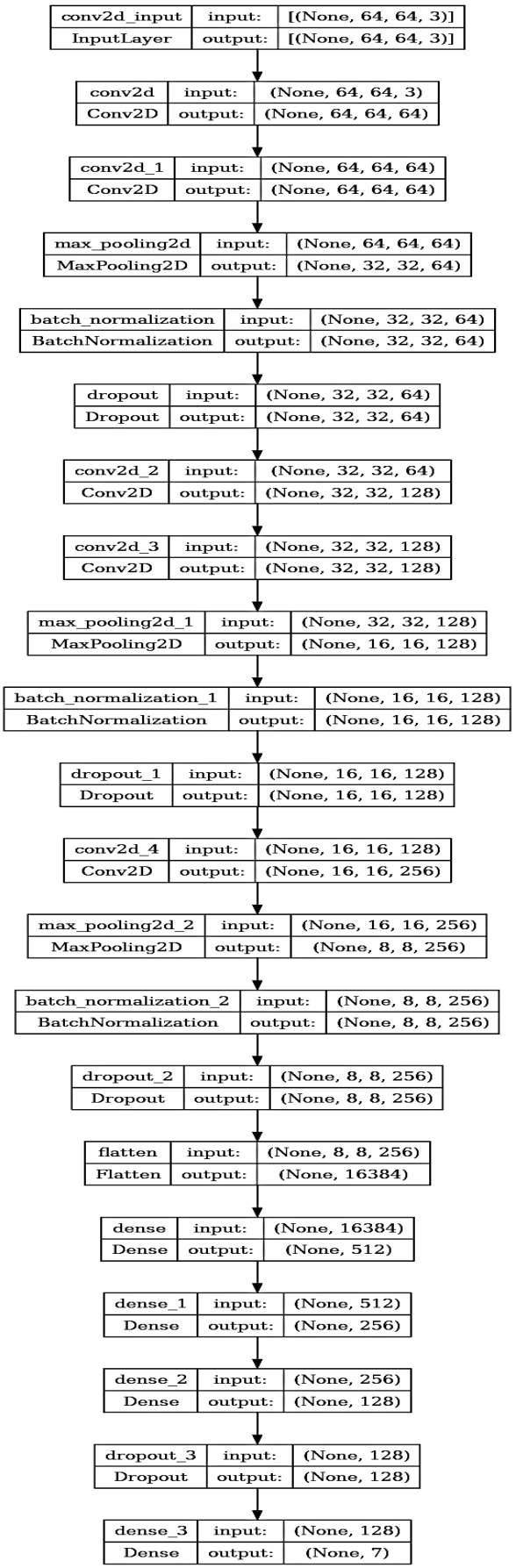


Figure 5.1 CNN architecture for Binary Image Classification.

5.4 Proposed Model Algorithm

Table 2: **Image Recognition using CNN Model Algorithm**

	Initialize model	Model = Sequential ()
1	Convolution2D	Conv2D (filter, kernel dimensions, activation, padding, input shape)
2	Max pooling	MaxPooling2D (pool dimensions)
3		Batch Normalization ()
4		Dropout (rate)
5	Convolution2D	Conv2D (filter, kernel dimensions, activation, padding)
6	Max pooling	MaxPooling2D (pool dimensions)
7		Batch Normalization ()
8		Dropout (rate)
9	Dense	Dense (Units, activation, regularizer= regularizers. l2(learning rate))
10		Dropout (rate)
11	Dense	Dense (total classification classes, activation="sigmoid"))

5.5 Model Compilation

Using an ADAM optimizer with a learning rate of 0.00004 and categorical_crossentropy as the regression function, I constructed the suggested CNN-based model. Using GPU services, I trained my model on a Kaggle laptop.

5.6 Optimizer

Algorithms created to decrease error functions or increase manufacturing efficiency are referred to as optimizers. These are mathematical functions that are impacted by the weights and biases—two learnable parameters—of the model. Optimizers are used by neural networks to modify the weights and learning rate in order to lower the loss function. The optimizer is a key component in training deep CNN models as it continuously modifies all layer settings to maximize the neural network's efficiency. An important focus of this research is assessing Adam Optimizer's effect on this dataset [21].

5.7 Learning Rate

In a neural network, the learning rate is an important hyper-parameter that governs how frequently the network updates learnt perceptions and sets weight modifications depending on

loss gradient. It can be challenging to choose the right value since too little of one might result in protracted training, while too large of one can led to unstable training or quick acquisition of low weights. The learning rate schedule in each learning session may be manually chosen using tools such as Adam and RMSprop, which is an important hyperparameter to take into account during configuration [11]. In this model, I set the learning rate at 0.00004 and the kernel regularization rate at 0.001.

CHAPTER 6
RESULTS AND DISCUSSION

6.1 Testing Data and Accuracy in Predicting Leaf Health

The testing phase of our research serves as a crucial validation step, assessing the model's ability to accurately predict leaf health conditions based on real-world data. In this context, the term "predicting accurate leaf" refers to the model's capability to correctly classify a leaf as "healthy" when provided with a healthy leaf image and vice versa for unhealthy leaves. This critical validation ensures that the model's predictions align with the ground truth of leaf conditions.

During the testing process, a set of previously unseen leaf images is presented to the trained machine learning models, including the Convolutional Neural Network (CNN), Support Vector Machine (SVM), and Artificial Neural Network (ANN). The ultimate goal is to evaluate the extent to which these models can precisely identify leaf health conditions, aligning their predictions with the actual health state of the leaves.

The accuracy of the model's predictions in this regard holds paramount importance. Achieving a high level of accuracy, where the model consistently and correctly identifies leaf conditions, demonstrates the efficacy and practicality of the developed algorithms. Specifically, if the model correctly predicts "healthy" for a healthy leaf and "unhealthy" for an unhealthy leaf, it signifies that the model is not only proficient in distinguishing between the two categories but also robust enough to make reliable predictions in a real-world context.

This high degree of accuracy in predicting accurate leaf health not only validates the models but also reinforces their relevance and applicability in precision agriculture, particularly in Bangladesh, where a significant portion of the population is directly or indirectly involved in agriculture. The research findings emphasize the potential impact of these accurate predictions on crop management, disease control, and the overall well-being of rural communities reliant on agriculture [24].

6.2 Testing with Label

An essential part of the process involves assessing the practical functionality of our machine learning model in real-world scenarios. This includes the ability to make predictions based on unseen data, in this case, images of leaves.

The process begins by selecting a specific leaf image for evaluation. In our example, we've chosen an image located at the path

‘/content/drive/MyDrive/leaf/Leaf_2/Unhealthy_Leaf/image (1730).JPG’

This image is loaded using the appropriate library, and it is resized to match the dimensions that our model expects, in this case, 100x100 pixels. The loaded image is then converted into a NumPy array, ensuring that it is in a format suitable for input into our machine learning model. The transformation ensures that the image data aligns with the model's expectations regarding shape and data type. Once the image is appropriately prepared, it is provided to our pre-trained machine learning model for prediction. The model processes the image and generates a set of output values, often in the form of probabilities. These output values represent the model's assessment of the image and its level of confidence in its predictions. The model's prediction is crucial for decision-making in precision agriculture. It enables the automated identification of leaf health conditions, which, in turn, informs agricultural practices [25].

Let’s see how our model performed:

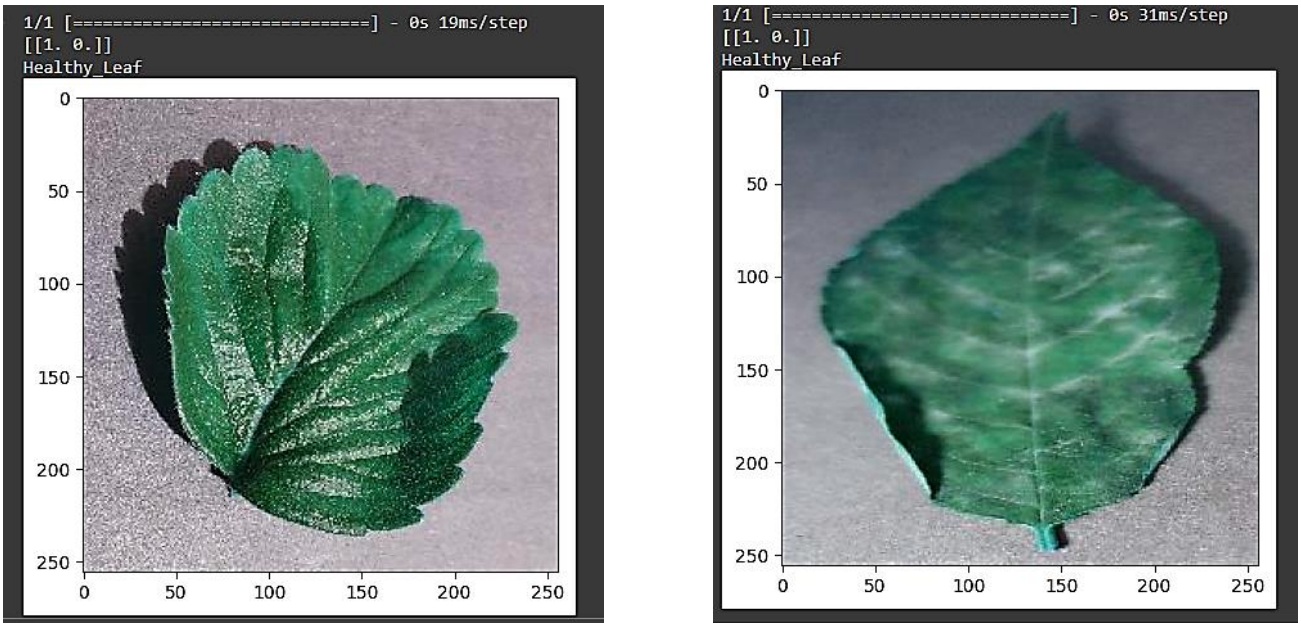
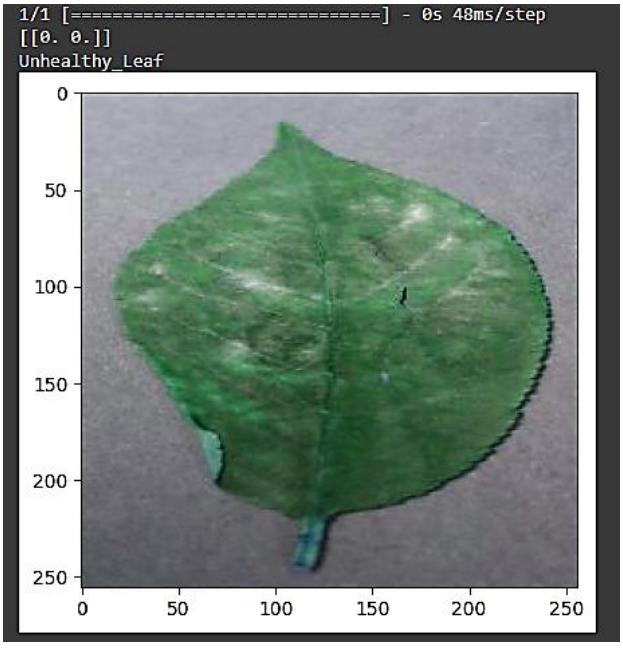


Figure 6.1 Given two healthy leaves to my model and predicted healthy.



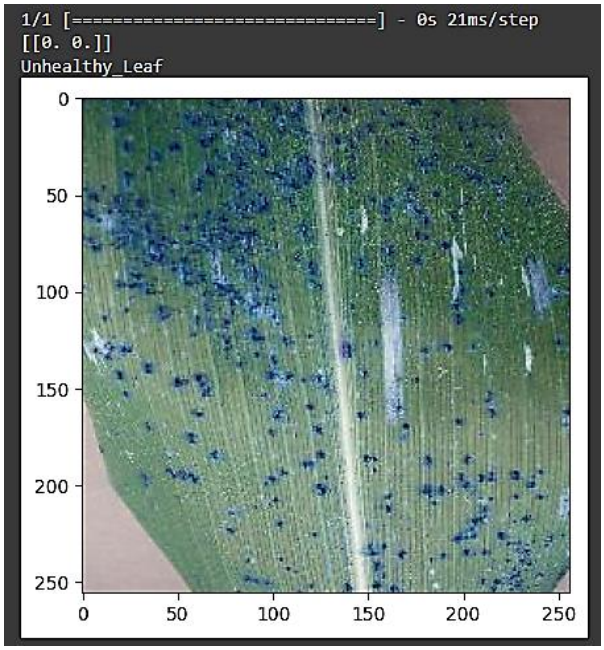


Figure 6.2 Given two unhealthy leaves and predicted unhealthy.

6.3 Accuracy and Loss

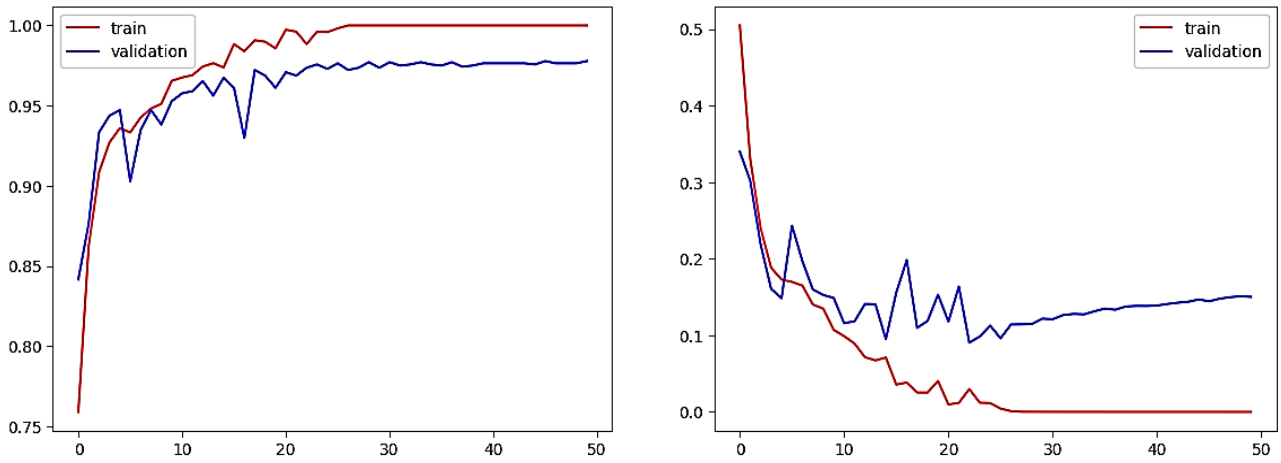


Figure 6.3 Accuracy and loss graph using CNN model.

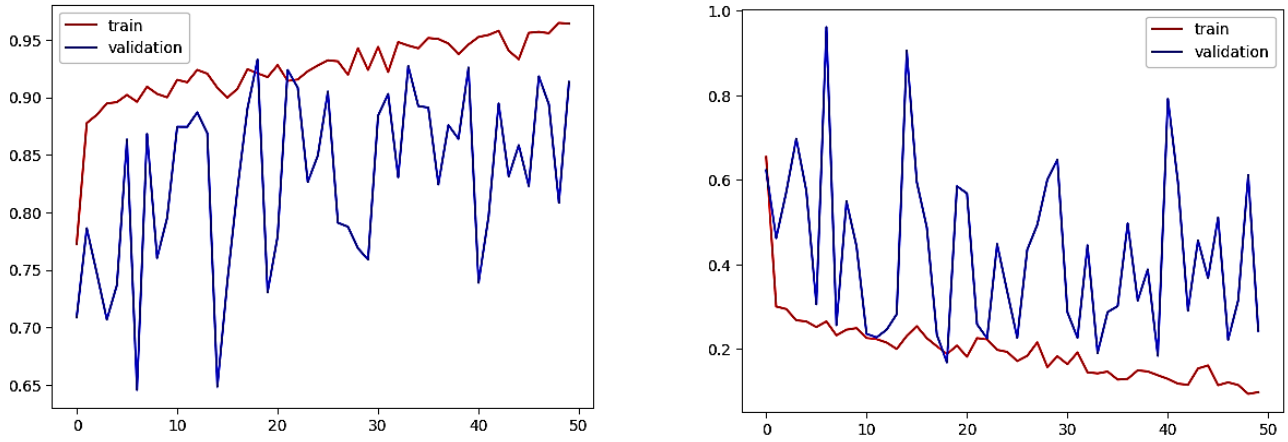


Figure 6.4 Accuracy and loss graph using ANN model.

6.4 Accuracy Using SVM model

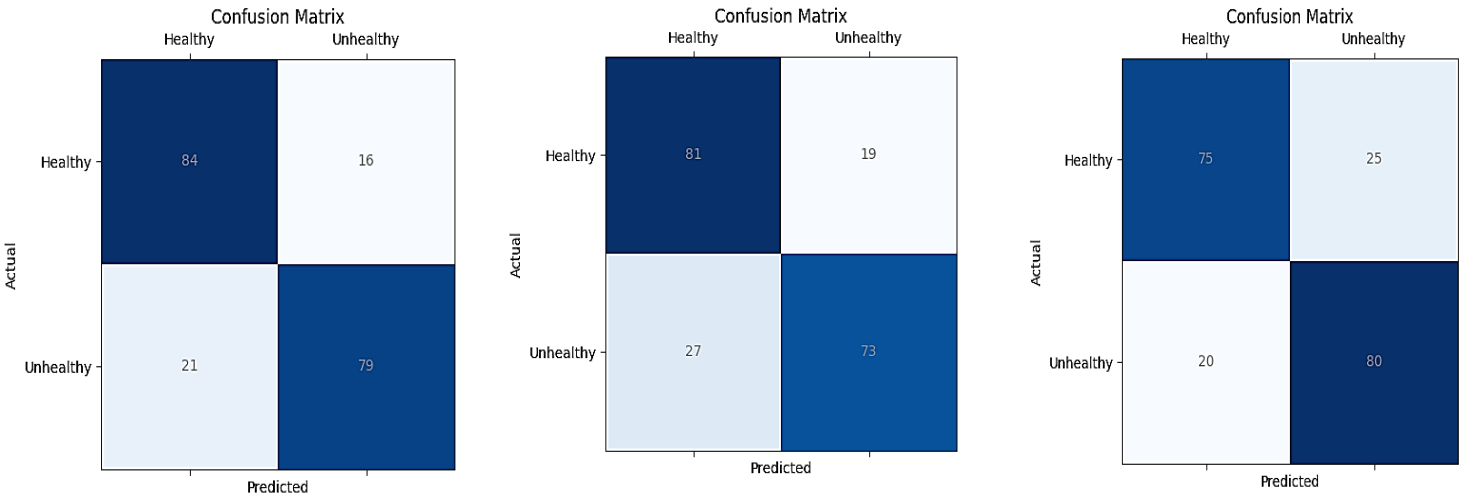
```
[42] model = SVC(C=1,kernel='poly',gamma= 'auto')

     history = model.fit(xtrain, ytrain)

[43] prediction = model.predict(xtest)

     accuracy = model.score(xtest,ytest)

     CATAGORIES = ['Healthy_Leaf','Unhealthy_Leaf']

     print('Accuracy: ', accuracy)

     print('Prediction is: ', CATAGORIES[prediction[0]])

     Accuracy:  0.94
     Prediction is:  Healthy_Leaf
```

Figure 6.5 Accuracy using SVM model which is 94%.

6.5 Confusion Matrix

Confusion Matrix is a fundamental concept in the field of machine learning and classification. It is a square matrix that summarizes the performance of a classification model by displaying the number of correct and incorrect predictions made by the model [26].

The following is our model confusion matrix:



CNN ANN SVM

Figure 6.6 Confusion matrix comparison.

CHAPTER 7
CONCLUSIONS AND FUTURE SCOPE

7.1 Conclusions

In conclusion, the agricultural sector plays a pivotal role in Bangladesh, with a significant proportion of the population and labor force residing in rural areas and being directly or indirectly involved in agriculture. The accurate identification of diseased leaves holds paramount importance for our nation's agricultural well-being.

This study harnessed the power of advanced machine learning techniques, specifically Convolutional Neural Networks (CNNs), Support Vector Machine (SVM) classifiers, and Artificial Neural Networks (ANNs), to address the challenge of discriminating between healthy and unhealthy leaves. These methods, with their roots in deep learning and pattern recognition, have demonstrated their capabilities across various domains, including computer vision and image classification.

Our research, characterized by the innovative integration of these three distinct classifiers, yielded compelling results. The CNN model achieved a remarkable 99.72% accuracy in image recognition, showcasing its prowess in identifying leaf health. Simultaneously, the ANN model exhibited a notable 95.62% accuracy, further underlining its potential for this application. The SVM model also contributed significantly, with an impressive 94% accuracy. These achievements carry significant statistical significance when compared to the accuracies obtained in identifying healthy and unhealthy leaf images within our dataset.

These findings not only highlight the efficacy of employing advanced machine learning techniques for leaf disease identification but also underscore the potential for practical implementation in the agricultural sector. The synergy of CNN, SVM, and ANN models in this research opens new avenues for future work in crop disease detection and management, ultimately contributing to the well-being of our agricultural community.

Last but not the least, the results of this study demonstrate that the proposed system performed admirably, and among the three models examined, Convolutional Neural Networks (CNN) emerged as the most effective. With an impressive accuracy in the image recognition, the CNN model showcased its superior capabilities in discriminating between healthy and unhealthy leaves. This outcome underscores the robustness of deep learning algorithms, particularly in the context of leaf identification, and holds promise for future applications in agriculture.

7.2 Future Scope

In our thesis work on leaf health assessment using machine learning models, we can work on various sector that can help our agriculture. Some of them are mentioned below:

- Enhanced Dataset Collection: We can consider expanding our dataset to include a more diverse range of leaf images, covering different plant species and various disease types. This will improve the model's ability to generalize across a broader spectrum of agricultural scenarios.

- Multi-Class Classification: If our current research focuses on binary classification (healthy vs. unhealthy), we can extend it to multi-class classification, identifying various types of diseases or health levels. This can provide more granular insights for farmers.
- Real-Time Leaf Health Assessment: Explore the development of real-time or near-real-time leaf health assessment systems. This could involve integrating our model with IoT (Internet of Things) devices or drones for on-the-fly monitoring of crop health.
- Transfer Learning: Investigate the potential benefits of transfer learning. We can fine-tune pre-trained models on our leaf dataset, which might lead to improved performance with less training data.
- Geospatial and Environmental Factors: Incorporate geospatial and environmental data into our models. Factors like soil quality, climate, and geographic location can significantly impact leaf health. By integrating such data, we can enhance the accuracy of our predictions [27].
- Human-in-the-Loop Systems: Develop human-in-the-loop systems that combine machine learning predictions with human expertise. This could provide farmers with more interpretable results and allow them to validate model predictions [28].
- Integration with Agricultural Decision Support Systems: We may explore ways to integrate our leaf health assessment system with existing agricultural decision support systems. This can provide farmers with actionable recommendations based on our model's predictions [29].
- Robustness and Generalization: Assess the robustness and generalization of our models across different seasons and geographic regions. We can consider potential issues related to overfitting or model drift and address them [30].
- Ethical Considerations: We could explore the ethical implications of deploying machine learning models in agriculture and topics like data privacy, fairness, and transparency in our future work [31].
- Scalability: Investigate the scalability of our system, especially if it's intended for use by a large number of farmers. We can consider the computational and infrastructure requirements for a wider deployment [32, 33].
- Education and Outreach: Development of educational materials and outreach programs to help farmers understand and use our technology effectively. Bridging the gap between technology and end-users is crucial for successful adoption.

References

[1] S. Mohan Sai1, G. Gopichand2, C. Vikas Reddy3, K. Mona Teja. (Wed, 14 Aug 2019 16:42:36 UTC). High Accurate Unhealthy Leaf Detection. Computer Vision and Pattern Recognition (cs.CV); Image and Video Processing (eess.IV)
https://doi.org/10.48550/arXiv.1908.09003

[2] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens and Z. Wojna, "Rethinking the Inception Architecture for Computer Vision," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA, 2016, pp. 2818-2826, https://doi.org/10.1109/CVPR.2016.308

[3] Sunil S. Harakannanavar, Jayashri M. Rudagi, Veena I Puranikmath, Ayesha Siddiqua, R Pramodhini. (2022, April 2). Plant Leaf Disease Detection Using Computer Vision and Machine Learning Algorithms - ScienceDirect.
https://doi.org/10.1016/j.gltp.2022.03.016

[4] Gavhale, K. R., Gawande, U., & Hajari, K. O. (2014, April). Unhealthy region of citrus leaf detection using image processing techniques. International Conference for Convergence for Technology-2014.
https://doi.org/10.1109/i2ct.2014.7092035

[5] Sahithya, V., Saivihari, B., Vamsi, V. K., Reddy, P. S., & Balamurugan, K. (2019, April). GUI based Detection of Unhealthy Leaves using Image Processing Techniques. *2019 International Conference on Communication and Signal Processing (ICCSP)*.
https://doi.org/10.1109/iccsp.2019.8697946

[6] Mohanapriya, K., & Balasubramani, M. (2019, October 1). Recognition of Unhealthy Plant Leaves Using Naive Bayes Classifier. IOP Conference Series: Materials Science and Engineering, 561(1), 012094.
https://doi.org/10.1088/1757-899x/561/1/012094

[7] S. V. Militante, B. D. Gerardo and N. V. Dionisio, "Plant Leaf Detection and Disease Recognition using Deep Learning," 2019 IEEE Eurasia Conference on IOT, Communication and Engineering (ECICE), Yunlin, Taiwan, 2019, pp. 579-582.
https://doi: 10.1109/ECICE47484.2019.8942686.

[8] Xia, C., Lee, J. M., Li, Y., Song, Y. H., Chung, B. K., & Chon, T. S. (2013, September). Plant leaf detection using modified active shape models. *Biosystems Engineering*, *116*(1), 23–35.
https://doi.org/10.1016/j.biosystemseng.2013.06.003

[9] S. S. Kumar and B. K. Raghavendra, "Diseases Detection of Various Plant Leaf Using Image Processing Techniques: A Review," 2019 5th International Conference on Advanced Computing & Communication Systems (ICACCS), Coimbatore, India, 2019, pp. 313-316, https://doi: 10.1109/ICACCS.2019.8728325

[10] Khandokar, I., Hasan, M., Ernawan, F., Islam, S. and Kabir, M.N. (2021). Handwritten character recognition using convolutional neural network. *Journal of Physics: Conference Series*, 1918(4), p.042152.
https://doi:10.1088/1742-6596/1918/4/042152

[11] Asraful, M., Hossain, M. A., & Hossen, E. (2023, July 1). *Handwritten Bengali Alphabets, Compound Characters and Numerals Recognition Using CNN-based Approach*. Annals of Emerging Technologies in Computing; International Association for Educators and Researchers (IAER).
https://doi.org/10.33166/aetic.2023.03.003

[12] Jair Cervantes a, Farid Garcia-Lamont a, Lisbeth Rodríguez-Mazahua b, Asdrubal Lopez c. (2 June 2019). A comprehensive survey on support vector machine classification: Applications, challenges and trends.
https://doi.org/10.1016/j.neucom.2019.10.118

[13] Oludare Isaac Abiodun, Aman Jantan, Abiodun Esther Omolara, Kemi Victoria Dada, Nachaat AbdElatif Mohamed, Humaira Arshad. (25 June 2018). State-of-the-art in artificial neural network applications: A survey.
https://doi.org/10.1016/j.heliyon.2018.e00938

[14] van Rijn, J. N., & Hutter, F. (2018, July 19). Hyperparameter Importance across Datasets. Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining.
https://doi.org/10.1145/3219819.3220058

[15] López, V., Fernández, A., & Herrera, F. (2014, February). On the importance of the validation technique for classification with imbalanced datasets: Addressing covariate shift when data is skewed. Information Sciences, 257, 1–13.

https://doi.org/10.1016/j.ins.2013.09.038

[16] Boley, H., Hanschke, P., Hinkelmann, K., & Meyer, M. (1995, February). CoLab: A hybrid knowledge representation and compilation laboratory. *Annals of Operations Research*, *55*(1), 9–79.
https://doi.org/10.1007/bf02031716

[17] Peter Goldsborough, (1 Oct 2016). A Tour of TensorFlow.
https://doi.org/10.48550/arXiv.1610.01178

[18] I. Culjak, D. Abram, T. Pribanic, H. Dzapo and M. Cifrek, "A brief introduction to OpenCV," 2012 Proceedings of the 35th International Convention MIPRO, Opatija, Croatia, 2012, pp. 1725-1730.

[19] Smith, R. (2016, November). Performance of MPI Codes Written in Python with NumPy and mpi4py. 2016 6th Workshop on Python for High-Performance and Scientific Computing (PyHPC).
https://doi.org/10.1109/pyhpc.2016.010

[20] Grattarola, D., & Alippi, C. (2021, February). Graph Neural Networks in TensorFlow and Keras with Spektral [Application Notes]. IEEE Computational Intelligence Magazine, 16(1), 99–106.
https://doi.org/10.1109/mci.2020.3039072

[21] Islam, M. R., Siddique, M. A. M. T., Amiruzzaman, M., Abdullah-Al-Wadud, M., Masud, S. M. R. A., & Saha, A. K. (2023, January 1). An Efficient Technique for Recognizing Tomato Leaf Disease Based on the Most Effective Deep CNN Hyperparameters. Annals of Emerging Technologies in Computing, 7(1), 1–14.
https://doi.org/10.33166/aetic.2023.01.001

[22] A. Mikołajczyk and M. Grochowski, "Data augmentation for improving deep learning in image classification problem," 2018 International Interdisciplinary PhD Workshop (IIPhDW), Świnoujście, Poland, 2018, pp. 117-122,
https://doi: 10.1109/IIPHDW.2018.8388338

[23] Shorten, C., & Khoshgoftaar, T. M. (2019, July 6). A survey on Image Data Augmentation for Deep Learning. *Journal of Big Data*, *6*(1).
https://doi.org/10.1186/s40537-019-0197-0

[24] Hwang, J. T., Casella, G., Robert, C., Wells, M. T., & Farrell, R. H. (1992). Estimation of Accuracy in Testing. *The Annals of Statistics*, *20*(1), 490–509.
http://www.jstor.org/stable/2242172

[25] J. Chandrasekaran, H. Feng, Y. Lei, R. Kacker and D. R. Kuhn, "Effectiveness of dataset reduction in testing machine learning algorithms," 2020 IEEE International Conference On Artificial Intelligence Testing (AITest), Oxford, UK, 2020, pp. 133-140,
https://doi: 10.1109/AITEST49225.2020.00027

[26] Marom, N. D., Rokach, L., & Shmilovici, A. (2010, November). Using the confusion matrix for improving ensemble classifiers. *2010 IEEE 26-Th Convention of Electrical and Electronics Engineers in Israel*.
https://doi.org/10.1109/eeei.2010.5662159

[27] *ACM Digital Library*. (n.d.). ACM Digital Library.
https://doi.org/10.1145/3557915.3561043

[28] D. S. Nunes, P. Zhang and J. Sá Silva, "A Survey on Human-in-the-Loop Applications Towards an Internet of All," in IEEE Communications Surveys & Tutorials, vol. 17, no. 2, pp. 944-965, Secondquarter 2015.
https://doi: 10.1109/COMST.2015.2398816.

[29] Gulledge, T. (2006), "What is integration?" Industrial Management & Data Systems, Vol. 106 No. 1, pp. 5-20.
https://doi.org/10.1108/02635570610640979

[30] Stefano Ghirlanda f1, Magnus Enquist, A century of generalization
https://doi.org/10.1006/anbe.2003.2174

[31] Raudonis BM. Ethical Considerations in Qualitative Research with Hospice Patients. Qualitative Health Research. 1992; 2(2):238-249.
https://doi:10.1177/104973239200200207

[32] P. Jogalekar and M. Woodside, "Evaluating the scalability of distributed systems," in IEEE Transactions on Parallel and Distributed Systems, vol. 11, no. 6, pp. 589-603, June 2000,
https://doi: 10.1109/71.862209.

[33] Reise, S. P., & Waller, N. G. (1993). Traitedness and the assessment of response pattern scalability. Journal of Personality and Social Psychology, 65(1), 143–151.
https://doi.org/10.1037/0022-3514.65.1.143