1. INTRODUCTION

There are several types of plant disease which includes fungal, bacterial, viral, nematode, and abiotic diseases. Fungal diseases are caused by fungi and can affect different parts of the plant, while bacterial diseases are caused by bacteria and can lead to leaf spots, wilting, and other symptoms. Viral diseases can cause stunted growth, yellowing, and other symptoms, while nematode diseases are caused by worm-like organisms that attack plant roots. Abiotic diseases are caused by non-living factors such as nutrient deficiencies, environmental stress, and chemical exposure.

The consequences of unhealthy crops can be severe, including reduced crop yield, poor crop quality, and economic losses for farmers. In some cases, plant diseases can also spread to neighboring crops and cause widespread damage to entire agricultural systems, potentially leading to food shortages and higher prices for consumers.

Image processing techniques can be applied to various types of images, including photographs, medical images, satellite images, and digital artwork. This project aims to use image processing techniques like adaptive thresholding, Gaussian blur, morphological closing, noise reduction etc. on the dataset before feeding it to the Machine Learning algorithms like SVM, KNN and CNN.

Pre-processing is necessary to remove undesirable distortions and enhance particular attributes that are crucial for the intended purpose, which may vary depending on the application. For software to operate accurately and generate the intended outcomes, pre-processing of an image is indispensable.

Image processing filters can be of two types:

i. **Frequency Domain**

   Frequency domain filters are image processing techniques that work by modifying the frequency spectrum of an image. Examples include high pass filters, low pass filters.

ii. **Spatial Domain**

   Spatial domain filters are image processing filters that operate on the values of pixels in an image directly. Examples include Median filter, Sobel filter, Laplacian Filter.

The Gaussian blur filter used here is a type of Frequency Domain low pass filter. Low-pass meaning it attenuates high frequency components in the image while allowing the low frequency components to pass through.

Morphological closing is a mathematical operation in image processing that is used to fill gaps and smooth out the boundaries of objects in an image. It is a dilation operation followed by an erosion operation. Morphological closing is commonly used to remove small holes and gaps in objects, to connect nearby objects that are almost touching, and to smooth out the boundaries of objects in an image. It can be applied to a variety of image types, including binary images (black and white), grayscale images, and color images.

Thresholding is a commonly used technique in image processing to create binary images from grayscale or color. There are different methods of thresholding, for example adaptive thresholding and Otsu's thresholding. In adaptive thresholding, different threshold values are calculated for different parts of the image based on the local characteristics of the image, such as variations in illumination. Otsu's thresholding is an automatic thresholding technique that determines the optimal threshold value by maximizing the variance between the two classes of pixels (foreground and background).

Machine learning is concerned with creating models that can learn from data and make predictions or take actions based on that learning. Machine learning is categorized into four categories:

i. Supervised Learning
ii. Unsupervised Learning
iii. Semi-Supervised Learning
iv. Reinforced Learning

Support Vector Machine (SVM) is a type of supervised machine learning algorithm. This means that it is an algorithm that requires labeled data to learn a mapping function from input variables (features) to output variables (labels or classes). SVM is a popular choice for supervised classification tasks due to its ability to handle complex data and its ability to work well with high-dimensional feature spaces. Its strength lies in its ability to find the optimal hyperplane that maximizes the margin between different classes of data points.

K-Nearest Neighbors (KNN) is a type of machine learning algorithm that also falls under the category of supervised learning. In the case of KNN, the algorithm predicts the class of a new data point based on the class of its nearest neighbors in the feature space. The value of k (the

number of nearest neighbors to consider) is chosen by the user. It is popular due to its simplicity and effectiveness in certain scenarios since it is non-parametric, meaning it does not make assumptions about the underlying distribution of the data.

Convolutional Neural Networks (CNNs) are a specific type of machine learning algorithm used for image and video analysis, natural language processing, and other applications involving complex input data. CNNs are a type of neural network, a class of machine learning algorithms inspired by the structure and function of the human brain. Neural networks are designed to learn from data, and CNNs are specifically designed to learn from data that has a grid-like topology, such as images.

The project aims to use the aforementioned image processing techniques along with the machine learning algorithms to classify whether a plant is diseased or not. Prior research has concentrated on diagnosing plant illnesses, but its reach has been constrained to particular plant disease categories and picture attributes. A broader group of photos' results have been less accurate as a result of this constraint. We'll create SVM, KNN, and CNN models for training in order to solve this problem, then evaluate the outcomes.

2. LITERATURE SURVEY

Many researchers have been working on improvising plant disease classification and recognition driven machine learning models and have achieved significant results. Still plant disease detection and classification is an open problem because of upcoming many challenges due to weather change and pollution.

Author has proposed a computer vision based algorithm to detect abnormalities in plants, specifically papaya leaves. The algorithm first converts RGB images to grayscale to enable the calculation of shape descriptors and Haralick features. To calculate the histogram, the image is converted to Hue Saturation Value (HSV). The algorithm distinguishes between healthy and diseased leaves using a Random Forest classifier [1]. The accuracy of the model achieved was 70%, and it can be improved by using additional local features. However, the accuracy of the models depended on the number and quality of images used for training. Therefore, further research is needed to improve the performance of these algorithms.

Processes like image capture, de-noising, enhancement, segmentation, feature extraction, and classification has been done to detect grape leaf disease. Particle Swarm Optimization SVM (PSO SVM), Back Propagation Neural Network (BPNN), and Random Forest has been implemented to compare their performance [2]. PSO SVM has the highest accuracy, which was 95.6%, followed by BPNN at 90.1% and Random Forrest at 88.2%. The author concludes PSO SVM algorithm produces better accuracy in classification and detection of grape leaf diseases compared to other algorithms.

A comparative study between a support vector machine and traditional method of neural networks has been done by the author. The study concluded that the SVM model had a far higher accuracy of 89.66% for the five patterns, namely gray levels, inverse difference moments, correlation, standard deviation and contrast, compared to the neural networks 41.38% [3]. The SVM model employs Principal Component Analysis to extract the eigenvalue from the gray information of the images to build a decision surface that classifies the images based on the Inner-Product Kernel. The SVM model proved to be more effective than the traditional neural network method, as demonstrated through extensive experiments. However, the study's limitations, such as the small sample size and the absence of comparative analysis with other machine learning models, should be considered.

A novel approach to plant species classification using convolutional neural networks (CNNs) by the author. The CNN architecture was designed to classify images of sixteen different types of plants. The results showed that the CNN-based approach outperforms the SVM-based classifier, and the accuracy achieved was 96.7%. The proposed method was compared to a traditional support vector machine (SVM) classifier that uses local binary pattern (LBP) and GIST features [4]. The experiments were conducted on experimental data acquired under natural outdoor illumination provided by TARBIL Agro-informatics Research Center of ITU. The authors suggested that future work should focus on building different architectures with various activation functions and experimenting with pre-processing methods to improve classification performance.

A framework for detecting and classifying tomato crop diseases using image processing techniques has been developed by the author [5]. The approach used images of plant leaves that exhibit visual symptoms of particular diseases, and extracted useful features from the images for disease classification. The proposed system had achieved a maximum average accuracy of 98.3% during experimentation. The approach only used texture features. The study highlighted the potential of image processing techniques for more effective and efficient detection and classification of crop diseases, which can be of great benefit to farmers. However, more research was needed to explore the use of different features and approaches for better accuracy and reliability.

Convolutional Neural Networks (CNNs) in image recognition, particularly for their reduced computational complexity and improved computing precision has been used by the author. CNNs' ability to tolerate errors enabled for the use of blurry or partial backdrop pictures, improving the accuracy of image recognition [6]. In the study, the propensity to detect seven tea leaf illnesses using two feature extraction approaches and three classifiers was compared. The findings showed that LeafNet produced the best classification accuracy, with an average of 90.16%, followed by the SVM method at 60.62% and the MLP algorithm at 70.77%.

Summary of Literature Survey

Sl.,no.	Author Details	Paper and Publication Details	Findings of the paper	Relevance to the project
1.	Maniyath, S. R., P V, V& Hebbar, R.	Plant Disease Detection Using Machine Learning. International Conference on Design Innovations for 3Cs Compute Communicate Control (ICDI3C), 25 April 2018	• Methods used • Random forest - 70%. • Accuracy can be improved by using additional local features	• Usage of Haralick features. • Pre-processing techniques
2.	Arshiya S. Ansari, Malik Jawarneh, Mahyudin Ritonga, Pragti Jamwal.	Improved Support Vector Machine and Image Processing Enabled Methodology for Detection and Classification of Grape Leaf Disease. Hindawi Journal of Food Quality, 9 July 2022	• SVM - 95.6% • BPNN - 90.1% • Random Forest - 88.2%	• Use of the SVM model with Radial Bias Function.
3.	Xiaowu Sun, Lizhen Liu, Hanshi Wang, Wei Song, & Jingli Lu.	Image classification via support vector machine. 4th International Conference on Computer Science and Network Technology (ICCSNT), 19 December 2015	• SVM - 89% • Neural Network - 41% • Use of Principal Component Analysis for feature extraction.	• SVM kernel • Feature extraction methods
4.	Yalcin, H., & Razavi, S.	Plant classification using convolutional neural networks. 5th International Conference on Agro-Geoinformatics (Agro-Geoinformatics), 18 July 2016	• CNN – 97.4% • SVM – 80% • Varied and large dataset	• CNN model and the parameters used

Sl.,no.	**Author Details**	**Paper and Publication Details**	**Findings of the paper**	**Relevance to the project**
5.	Muhammad Zaka-Ud-Din, Wakeel Ahmad, Sumair Aziz.	Classification of Disease in Tomato Plants' Leaf Using Image Segmentation and SVM. Technical Journal, University of Engineering and Technology (UET) Taxila, Pakistan, 5 August 2018.	• SVM – 98.6% • Extensive use of GLCM	• SVM model • Pre-processing techniques
6.	Jing Chen, Lingwang Gao	Visual Tea Leaf Disease Recognition Using a Convolutional Neural Network Model. Yearly journal, China Agricultural University, Beijing 100193, China.	• CNN – 90.2% • SVM – 60.1%	• Fault tolerance of CNN

Table 2.1: Literature survey for plant disease detection

3. PROBLEM DEFINITION

In recent times, changes in climatic conditions and certain environmental factors have led to the acquisition of various diseases by plants. As plants come in a broad variety of colors and sizes, identifying illnesses and diseases they acquire can be a very taxing task. Additionally, some of these infections only occur in particular plant species under certain specific conditions, thereby making the acquisition of data in those cases extremely difficult.

Therefore, having the ability to recognize a disease without the intervention of a botanist would be an extremely helpful tool for farmers and agriculturalists alike.

The projects objective is to address the difficulties faced in identifying plant diseases by developing a computer vision-based plant disease identification system that utilizes machine learning algorithms and image recognition.

4. SOLUTION STRATERGY

The project aims to develop a plant disease identification system that utilizes machine learning algorithms and image recognition. The proposed system will enable farmers and agriculturalists to accurately identify and diagnose plant diseases by analyzing images of the plants without an expert's intervention.

The algorithm developed would handle a wide range of plant diseases, including those that only occur in certain plant species under specific conditions.

The following process will be followed to develop an algorithm that detects plant disease.

i. Acquisition of plant disease data
ii. Identification and implementation of preprocessing algorithms
iii. Identification of machine learning algorithms for model training and testing

The plant species include Potato, Pepper and Tomato. The images collected are for various plant species and diseases. In total, there are 9 classes, with 1 healthy class and 2 unhealthy classes for each plant, with the only exception being Tomato, which consists of 3 unhealthy classes. For Image pre-processing, the image is first converted from BGR color space to RGB color space. The RGB image is then converted to grayscale.

A Gaussian blur filter is then applied to the grayscale image with a kernel size of 25x25.

Otsu's thresholding technique is applied to the blurred grayscale image. This thresholding technique automatically calculates the optimal threshold value for the image based on its histogram. The resulting image is a binary image with white pixels indicating the object of interest and black pixels indicating the background. A flag inverts the binary image, so the object of interest is now black.

Next a morphological operation to the binary image using the kernel created in step 5. This particular operation is a closing operation, which fills in small holes and gaps in the object of interest and smooths its edges. The resulting image is stored in the variable “closing”.
The images are then converted into a array and stored in a “csv” format.

After performing all the operations mentioned above on the whole dataset, it is then fed to the machine learning algorithms SVM and KNN. CNN takes images as input. Since SVM cannot perform multiclass classification at once, one-versus-rest and radial bias function is used.

5. DESIGN

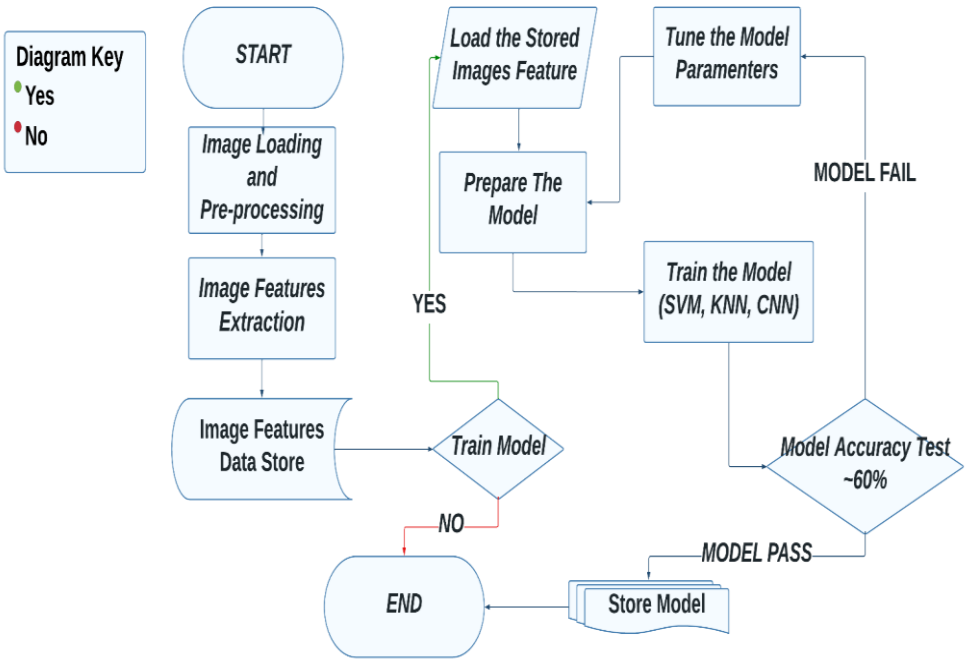

Figure 5.1 Image feature extraction and model preparation

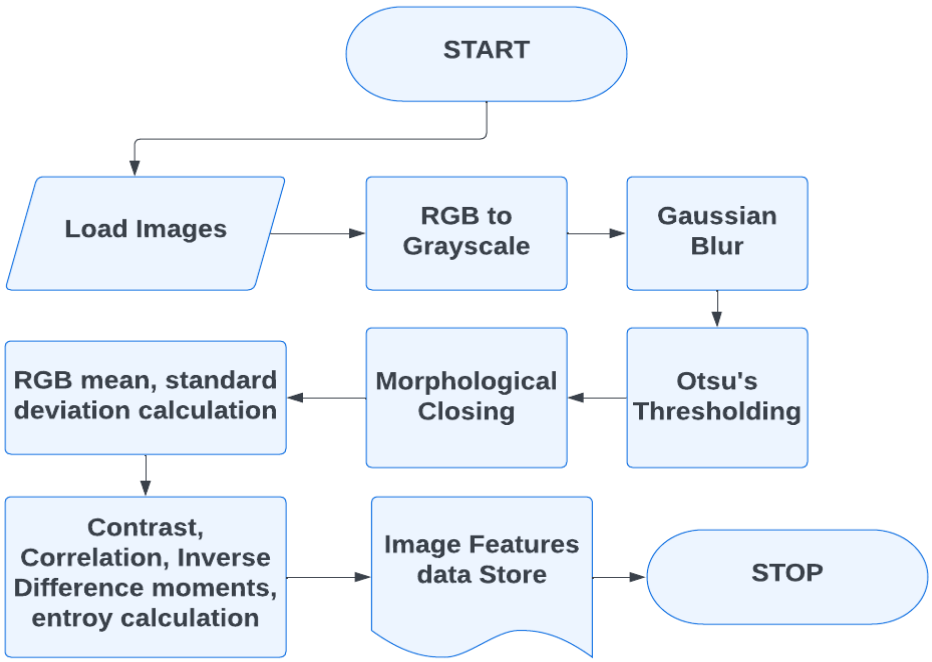

Figure 5.2 Image Pre-processing Steps

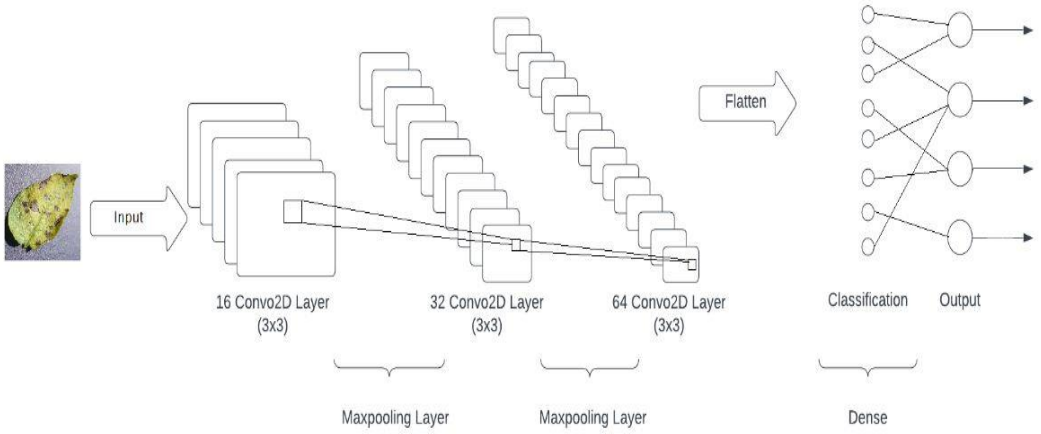

Figure 5.3 Architecture of CNN

6. METHODOLOGY

6.1 Image Acquisition

The images were collected for various plants species and diseases. The images were placed in jpg format. The source images were taken from the Kaggle plant village dataset, Potato leaf disease dataset contributed by Samarnjit Ghose, Tomato Leaf dataset by Muhummadiputra.

6.2 Image Feature Extraction

A total of 10 features are extracted, including color and textures. First, the input image's red, green, and blue colour channels' means and standard deviations are determined. Next, a grayscale conversion and a Gaussian blur filter are used to lessen noise in the image. The Haralick technique is then used to extract texture features, which uses the (Gray Level Co-occurrence matrix) GLCM to compute contrast, correlation, inverse difference moments, and entropy. The relative frequencies of a pair of grey levels that are present at a specific distance and angle are represented by the GLCM. The values of the texture features are then used to form a vector, which is appended to a data frame. The features can be used for further image analysis or machine learning tasks. The mathematical equations of the features used are given below (1)-(5):

$$G = \begin{bmatrix} P(1,1) & \cdots & P(1,Dg) \\ \vdots & \ddots & \vdots \\ P(Dg,1) & \cdots & P(Dg,Dg) \end{bmatrix}$$ (1) [Research Gate, 24/03/2023]

Contrast = $\sum_{n-0}^{Dg-1} x^2\{\sum_{i=1}^{Dg}\sum_{j=1}^{Dg} P(i,j)\},\ |i=j|=n$ (2) [Research Gate, 24/03/2023]

where Dg is numbers of gray levels that can be represented by a matrix G having dimension Dg as shown in Equation (1) with any pixel point (i,j) and P(i,j) represents the probability of presence of pixel pairs at certain distance d at angle θ in GLCM image.

Correlation = $\frac{\sum_{i=1}^{Dg}\sum_{j=1}^{Dg}(i,j)P(i,j)-\mu_x\mu_y}{\sigma_x\sigma_y}$ (3) [Source: SciElo, 24/03/2023]

where μx μy are means and σx σy are standard deviations of Px and Py the partial derivative function.

Inverse Difference Moments = $\sum_{i=1}^{Dg}\sum_{j=1}^{Dg}\frac{1}{1+(1-j)^2}P(i,j)$ (4) [Source: IJSTE, 24/03/2023]

Entropy = - $\sum_{i=1}^{Dg}\sum_{j=1}^{Dg} P(i,j)\log[P(i,j)]$ (5) [Source: SciElo, 24/03/2023]

6.3 Model Training and Testing

There exist several techniques for solving Multi-class classification problems using Support Vector Machine (SVM), such as One Against-One (OAO), One-Against-All (OAA), Binary Tree (BT), and Directed Acyclic Graph (DAG) classifiers. The primary goal of SVM is to construct an optimal hyperplane that acts as a decision surface using input samples to maximize the margin between two sides. To perform multi-class classification with SVM, the one-versus-rest method and Gaussian radial basis function will be utilized. The RBF kernel is a positive parameter used to regulate the radius and is given by Equation (6) . Since SVM cannot perform multi-class classification simultaneously, it uses the one-versus-rest method to perform binary operations on each dataset before making the final multi-class classification.

$$K(x_i, x_j) = \exp\left(\frac{-||x_i - x_j||^2}{2\sigma^2}\right)$$ (6) [Source: DataFlair, 24/03/2023]

where k is the kernel function, $x_i = (x_{i1}, x_{i2}, \ldots, x_{iN})$ corresponds to the attribute set for the i^{th} sample in each sample tuple represented by (x_i, x_j) in N training data of a binary classification.

K Nearest Neighbor (KNN) is a versatile technique that can be used for both classification and regression tasks. It is a non-parametric algorithm that is widely used for pattern recognition, where it selects the k nearest neighbors for classification or regression purposes.

KNN is a classification method that determines the class of a given data point by looking at the K nearest neighbors and selecting the most frequent class based on the similarity of those neighbors, which is computed using distance metrics. Some of the common distance functions are Euclidean, Manhattan, Minkowski and Hamming distance.

The ball tree algorithm is a data structure used by the KNN model to perform efficient nearest neighbor searches. It works by recursively partitioning the data space into a set of nested hyperspheres (or balls), where each node in the tree represents a single ball, and the children of each node represent two smaller balls that partition the parent ball.

The ball tree algorithm is preferred over other algorithms (e.g., brute force, KD-tree) when the data has a high number of dimensions, as it can be faster and more memory-efficient than other methods in these cases.

The Euclidean distance is a common choice for distance metric in KNN models, especially for numerical data. It measures the straight-line distance between two points in Euclidean space. Consider two points in two-dimensional space, (x1, y1) and (x2, y2). The Euclidean distance between these points is:

$$d(p,q) = \sqrt{(q1-p1)^2 + (q2-p2)^2} \quad (7)$$ [Source: Wikipedia, 23/4/2023]

This distance measure is used to find the nearest neighbors of a test point in the training set, based on their feature values. The KNN algorithm then uses the class labels of these nearest neighbors to make a prediction for the test point.

In this study, the recommended deep learning technique was decided to be the convolutional neural network (CNN). With its multi-layered structure, CNN's ability to recognize and categorize objects with little to no pre-processing enables it to analyze visual pictures and quickly extract key properties. The key layers used are Rectified Linear Unit (ReLU) and an output layer with Softmax function.

ReLU formula:

$$f(x) = \max(0,x) \quad (8)$$ [Source: Deepchecks, 23/4/2023]

Softmax formula:

$$\sigma(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{k} e^{z_j}} \quad (9)$$ [Source: DeepAi.org, 23/4/2023]

In summary, this model architecture consists of several convolutional and pooling layers that extract features from the input image, followed by a fully connected layer that processes the extracted features and outputs a probability distribution over the nine possible classes.

Along with accuracy, macro average and weighted avg metrics are also used to assess the performance of the SVM and the KNN model. The macro average is calculated by taking

the average of the precision, recall, and F1 score for each class in the dataset. This means that each class is given equal weight in the calculation of the macro average. The macro average provides a way to evaluate the overall performance of the model on the dataset, regardless of class imbalance.

The weighted average, on the other hand, takes into account the number of samples in each class when calculating the average precision, recall, and F1 score. This means that the performance of the model on each class is weighted by the number of samples in that class. The weighted average is a useful measure when the dataset is imbalanced, as it provides a more accurate representation of the overall performance of the model.

7. IMPLEMENTATION

The dataset obtained from Kaggle was pre-processed and split into test and train subfolders. SVM, KNN, and CNN algorithms were implemented and evaluated for image classification, with the highest-performing model being selected as the final model.

7.1 Dataset Used for Training Models

Sl. No	Name Of Dataset	Type of Data	Size on Disk	No of images	Labelled/ unlabeled	Source of dataset [Link]
1	Plant Village	.jpg	910.0 MB	15897	Labelled	www.kaggle.com/datasets/soumiknafiul/plantvillage-dataset-labeled
2	Potato Leaf Disease	.jpg	26.28 MB	1500	Labelled	www.kaggle.com/datasets/muhammadardiputra/potato-leaf-disease-dataset
3	Tomato Leaf Disease	.jpg	186.2 MB	11000	Labelled	www.kaggle.com/datasets/kaustubhb999/tomatoleaf

Table 7.1: Dataset Description

The details of the dataset have been shown in Table 7.1, along with the various sources from which they were acquired. After acquisition, they were divided into train and test folders before moving ahead with pre-processing. It is shown in Figure 7.1

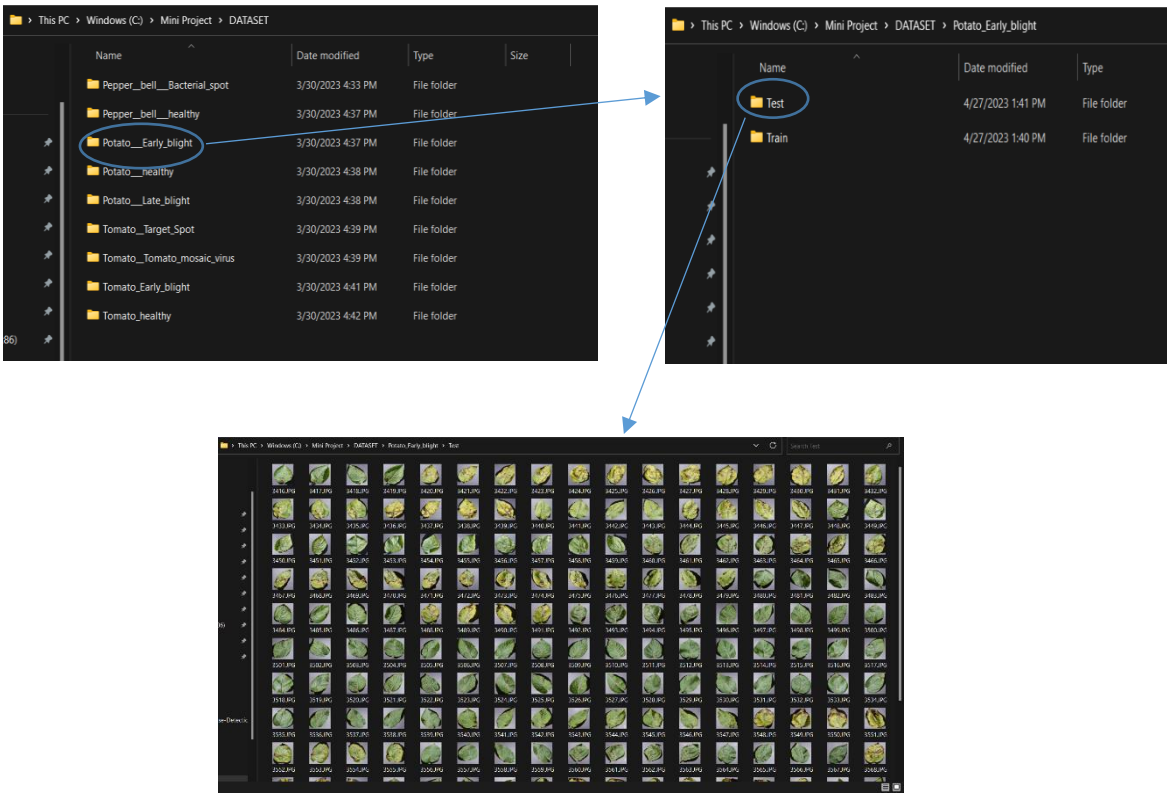

Figure 7.1 Dataset Organization into sub folders

Before training and testing the model, the dataset was structured into directories. The main directory was named "DATASET" and it comprised nine subdirectories, as shown in Table 7.2.

Organization of dataset for plant disease detection:

Sl. No	Categories of sub folders in root folders Train and Test	Images count in respective sub folders of root folder (Train)	Images count in respective sub folders of root folder. (Test)
1.	Potato_Early_blight	800	200
2.	Potato_Late_blight	800	200
3.	Potato_Healthy	152	40
4.	Pepper_bell_bacterial_spot	752	188
5.	Pepper_bell_healthy	1180	296
6.	Tomato_Target_Spot	1123	281
7.	Tomato_Mosaic_Virus	298	84
8.	Tomato_Early_Blight	800	200
9.	Tomato_Healthy	800	200

Table no 7.2: Dataset Organization

7.2 Pre-processing

The dataset used in this study was sourced from multiple sources as specified earlier. After importing the dataset into the Jupyter notebook, it was partitioned into two parts: one for training and the other for testing, with a ratio of 80:20 respectively. The 80% portion was designated for training purposes, while the remaining 20% was reserved for testing.

Before the training process commenced, the data in the training partition was subjected to a normalization process. Normalization refers to the process of rescaling the features of a dataset to a standard scale. This ensures that all features have an equal contribution during the training process. The normalization process that was employed in this study involved transforming the features into z-scores. This means that the features were scaled in such a way that their mean was equal to zero, and their standard deviation was equal to one.

Subsequently, the normalized training dataset was subjected to several preprocessing algorithms to extract various features. These algorithms involved the extraction of color features such as the red, green, and blue values of the images, and the calculation of their standard deviation. Texture features such as contrast, correlation, inverse difference moments, and entropy were also extracted. These features were computed to help the model differentiate between the different classes in the dataset.

After feature extraction, the resulting feature values were stored in CSV format for easy access and manipulation.

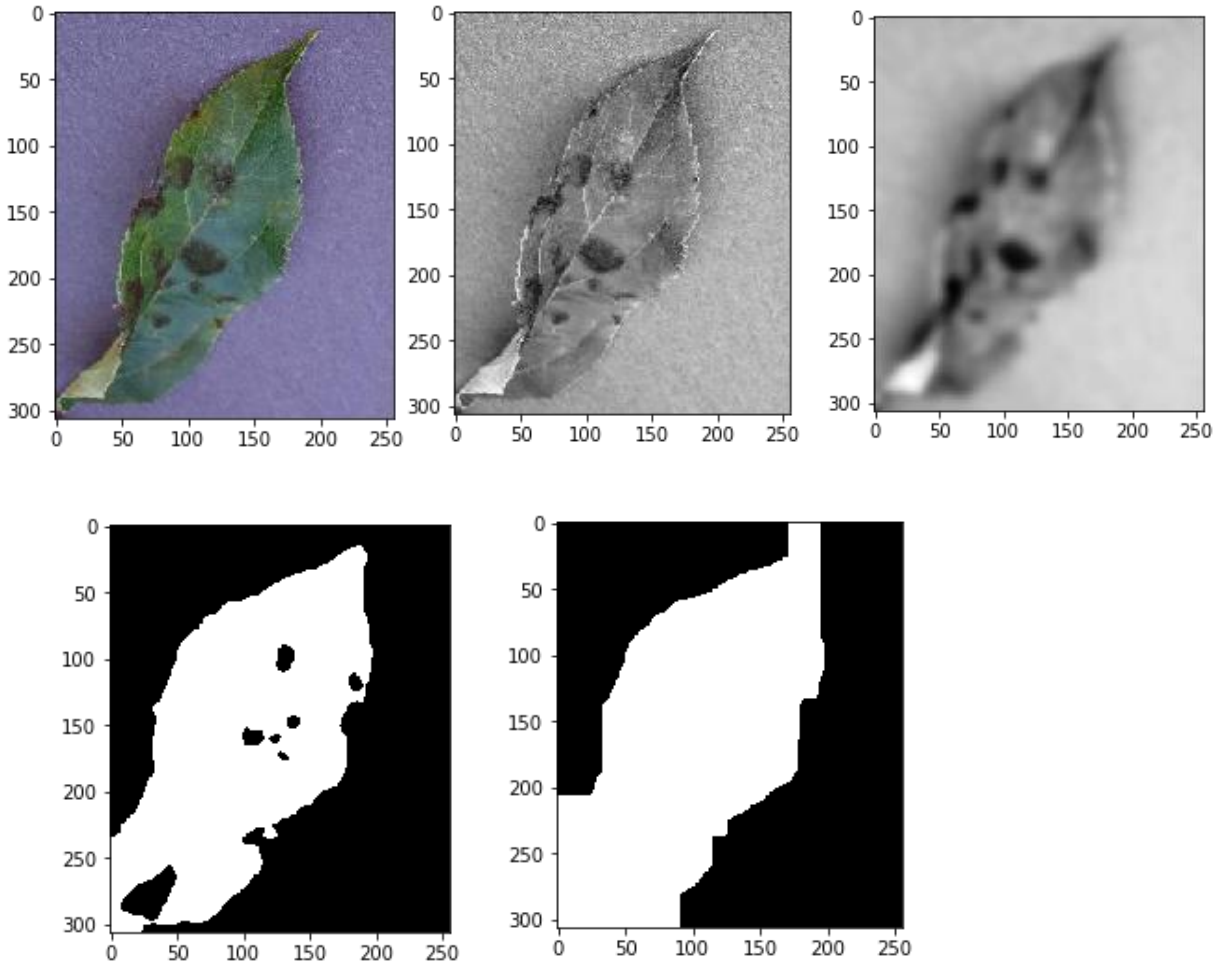

Figure 7.2: Input-Output on applying preprocessing algorithm

7.3 Model Training & Testing

SVM Model

Due to its ability to transform non-linearly separable data into a high-dimensional space where it is linearly separable, the radial basis function (RBF) kernel is a popular option in SVM. Based on two data points' separation from a reference point or centre point, this kernel function determines how similar the two points are. The RBF kernel has been demonstrated to be effective in a number of classification tasks, and it is particularly helpful when there are no previous assumptions about the distribution of the data.

Pseudocode for SVM Model:

```
SVM_Define_Hyperparameters{
C=100.0, cache_size=200, class_weight=None,
coef0=0.0, decision_function_shape=ovr,
degree=3, gamma=0.01, kernel=rbf,max_iter=-1,
probability=False, random_state=None, shrinking=True,
tol=0.001, verbose=False
}
```

The SVM model has several hyperparameters that can be adjusted to optimize the model's performance. The regularization parameter, one of these hyperparameters, regulates the trade-off between attaining a low training error and a low testing error.

A smaller value of this parameter creates a more generalized model with a wider margin, while a larger value creates a narrower margin, potentially overfitting the training data.

Another important hyperparameter is the type of kernel function used by the model. There are several kernel functions to choose from, such as linear, radial basis function, polynomial, and sigmoid. T The input data must be mapped to a higher-dimensional space via the kernel function in order to support nonlinear classification. Additional hyperparameters that may be changed to enhance the model's performance include the degree of the polynomial kernel function and the kernel coefficient for the radial basis function, polynomial, and sigmoid kernels.

Finally, there are several additional hyperparameters that can be adjusted to fine-tune the model's performance, such as the maximum number of iterations to run the solver, the tolerance for stopping criteria, whether to use the shrinking heuristic, and whether to calculate class probabilities in addition to predicting class labels. Properly tuning these hyperparameters can lead to a more accurate and robust SVM model.

Model classification report:

Class labels	precision	recall	f1-score	Support
1	0.84	0.80	0.82	183
2	0.90	0.93	0.91	303
3	0.94	0.98	0.96	201
4	0.68	0.52	0.59	25
5	0.88	0.83	0.85	219
6	0.88	0.91	0.90	292
7	0.97	0.91	0.94	68
8	0.79	0.79	0.79	182
9	0.99	0.99	0.99	196
Accuracy			0.89	

Table 7.3 Classification Report for SVM

Table 7.3 summarizes the performance of an SVM model with 9 classes, as measured by precision, recall, and F1 scores. The scores for all classes fall within the range of 0.70 to 0.80, indicating that the model is performing consistently well across all classes. These results suggest that the RBF kernel is functioning as expected and that the model is effectively able to classify instances within each of the eight classes.

Confusion Matrix for SVM:

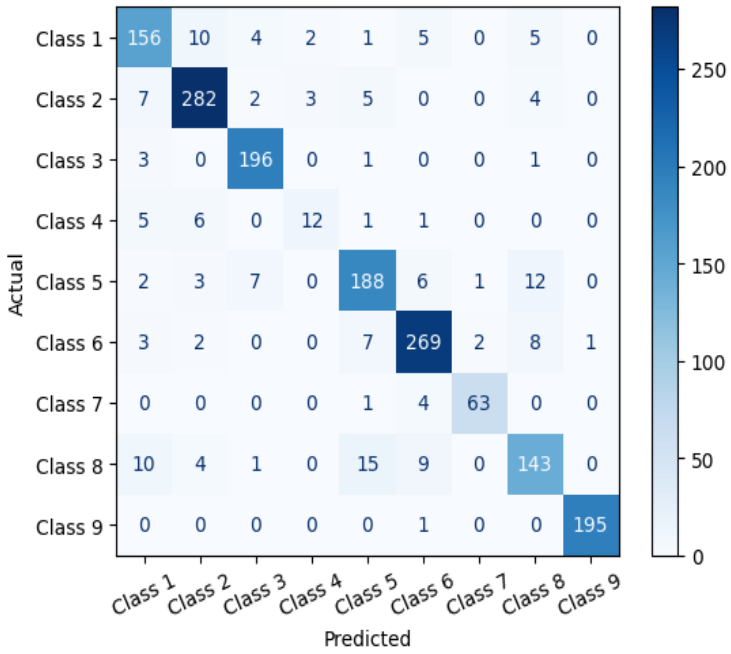

Figure 7.3 Confusion Matrix for SVM

The confusion matrix in Figure 7.3 indicates that the SVM model is performing reasonably well in accurately classifying instances within each of the nine classes. The majority of instances in each class are correctly classified, as indicated by the high values along the diagonal of the matrix. However, there are some instances that are misclassified, as indicated by the off-diagonal values in the matrix.

KNN Model:

The Ball Tree algorithm is used in the KNN model because it is faster for high-dimensional data. The Euclidean distance is used as the distance metric for calculating the distance between points in the feature space.

Pseudocode for KNN Model:

```
KNN_Define_Hyperparameters {
n_neighbors=11,
weights= 'distance',
algorithm='ball_tree',
leaf_size=30, p=2
}
```

A non-parametric technique for classification and regression is the K-Nearest Neighbours (KNN) algorithm. In this particular implementation, the n_neighbors parameter is used to define the number of neighbors to consider during the majority voting process. A larger value of n_neighbors will result in a smoother decision boundary but may also lead to overfitting. Here, n_neighbors is set to 11.

The weights parameter specifies the weight function used in the prediction. In this case, it is set 'distance', which means that closer neighbors will have a higher weight in the prediction than farther ones. This approach is useful when the distance between the samples varies widely.

The algorithm parameter specifies the algorithm used to compute the nearest neighbors. Here, it is set to 'ball_tree' which is faster for high-dimensional data. The leaf_size parameter affects the speed of the construction and query of the BallTree or KDTree algorithm. Finally, the p parameter is the power parameter for the Minkowski metric. When p=2, this is equivalent to using the standard Euclidean distance.

Model classification report for KNN:

Class labels	precision	recall	f1-score	support
1	0.75	0.65	0.70	290
2	0.82	0.81	0.81	451
3	0.87	0.91	0.89	298
4	0.72	0.27	0.39	49
5	0.75	0.77	0.76	318
6	0.75	0.88	0.81	450
7	0.91	0.73	0.81	96
8	0.72	0.71	0.72	286
9	0.95	0.96	0.96	266
Accuracy			0.80	

Table 7.4 Classification report for KNN

Table 7.4 provides a summary of the precision, recall, and F1 scores of an KNN model with 9 classes. These results suggest that the ball tree algorithm, which was utilized in this study, is a suitable approach for addressing this particular problem. Overall, the KNN model appears to be performing well in accurately classifying instances within each of the nine classes.

Confusion Matrix for KNN:

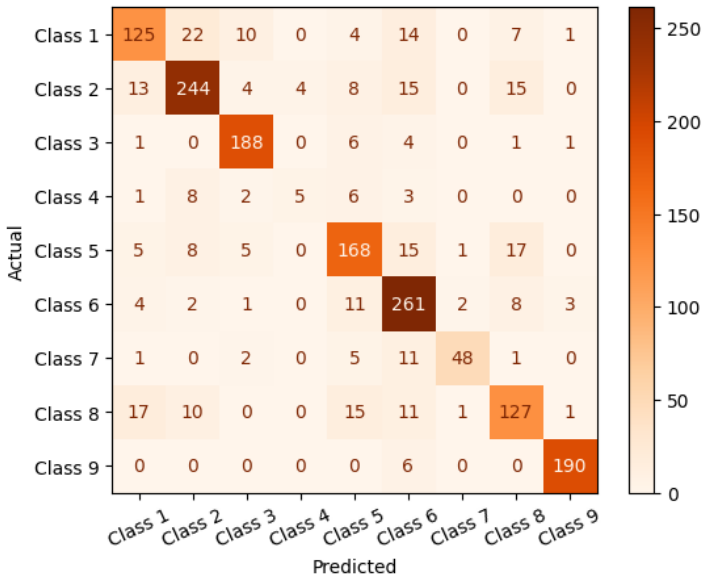

Figure 7.4 Confusion Matrix KNN

Figure 7.4 suggests that although the KNN model is performing reasonably well in accurately classifying instances within each of the nine classes, the confusion matrix suggests that it is slightly less accurate than the SVM model discussed earlier. While there are high values along

the diagonal of the matrix, indicating that the majority of instances in each class are correctly classified, there are more instances that are misclassified compared to the SVM model.

CNN model:

ReLU layers are used in this model because they introduce non-linearity, which is necessary for the model to learn complex relationships between the input data and output labels. Specifically, ReLU (Rectified Linear Unit) is a simple and efficient activation function that has been shown to work well in deep learning models. It sets all negative values to zero and leaves positive values unchanged, resulting in a sparser and more efficient representation of the input data.

Pseudocode for CNN model:

```
Sequential_CNN{
        Layer 1:
                Convolution (activation = 'relu')
                MaxPooling
        Layer 2:
                Convolution (activation = 'relu')
                MaxPooling
        Layer 3:
                Convolution (activation = 'relu')
                MaxPooling
        Layer 4:
                Flatten ()
        Layer 5:
                Dense (activation = 'relu')
                Dense (activation = 'softmax')
}
```

Here,

Rescaling layer: Normalizes the pixel values of the input images to the range of [0, 1] to improve the convergence of the model during training. Convolutional layer with 16 filters, a kernel size of 3x3, and ReLU activation: Applies a set of learnable filters to the input image to extract relevant features, while the ReLU activation function ensures non-linearity in the model.

Max pooling layer: Reduces the spatial dimensions of the input while preserving the most important features, making the model more robust to variations in the input. Two more convolutional and max pooling layers: The model uses multiple layers to extract higher-level features from the input image, leading to a more accurate classification.

Flatten layer: Reshapes the output of the previous layers into a 1D vector, which can be fed into a fully connected layer. Dense layer with 128 units and ReLU activation: A fully connected layer that performs a linear transformation on the flattened input, followed by a non-linear activation function.

Output layer with softmax activation: A dense layer with 9 units (one for each class) that outputs the probability of the input image belonging to each class. The softmax activation ensures that the output probabilities sum up to one.

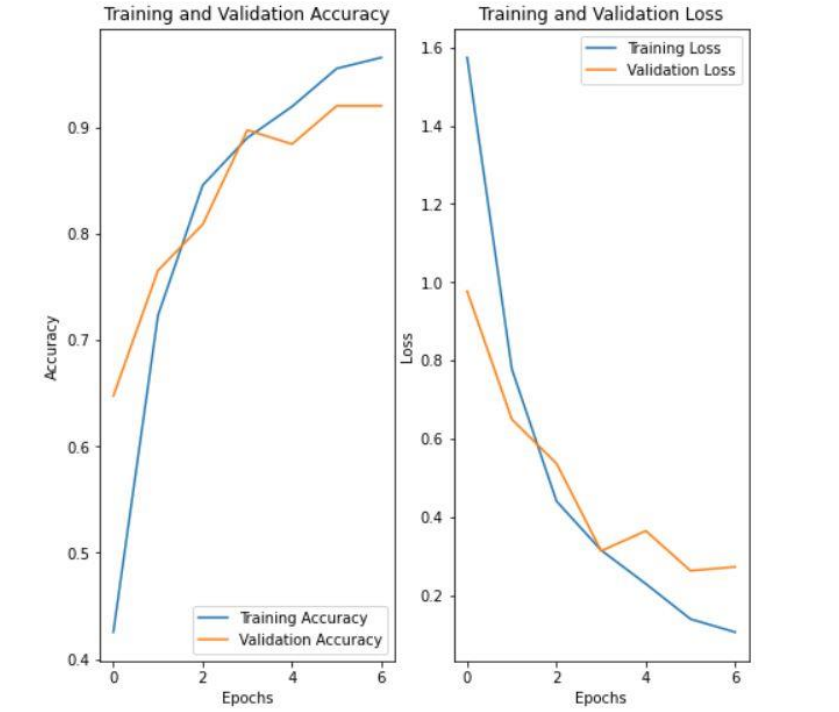

Figure 7.5 Training & Validation Accuracy/Loss

Figure 7.5 shows the training and validation accuracy and loss curves for the CNN model. It suggests that the model is performing well, with high accuracy and low loss values on both the training and validation sets. The curves indicate that the model is not overfitting to the training data, and is able to generalize well to new data, resulting in accurate classification of instances in the nine classes.

Confusion Matrix for CNN:

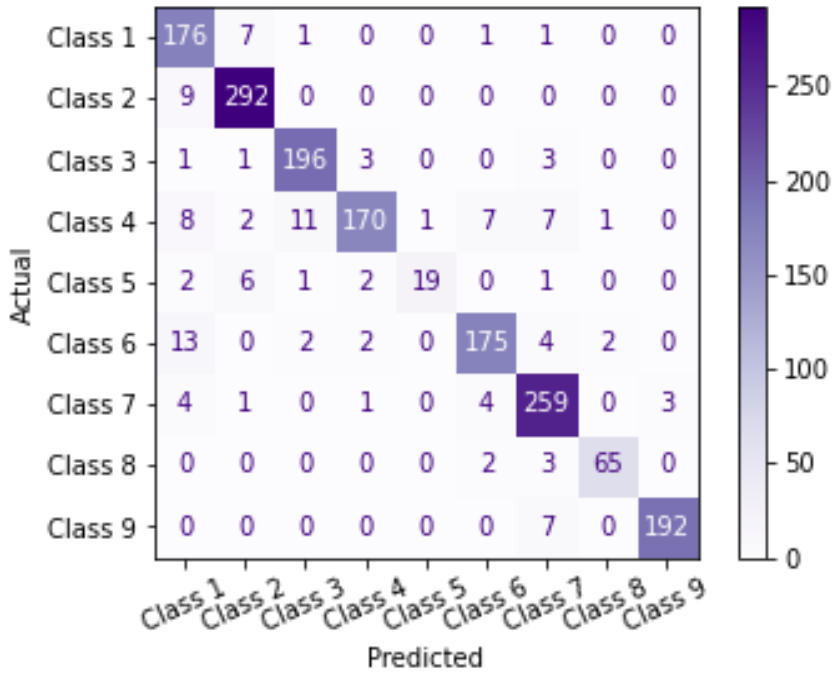

Figure 7.6 Confusion Matrix for CNN

Figure 7.6 indicates that the CNN model appears to be performing the best among all the models, as indicated by the high values along the diagonal of the confusion matrix, which indicate correctly classified instances. The model is able to accurately classify the majority of instances in each class, resulting in relatively low off-diagonal values in the matrix, indicating that misclassifications are infrequent.

8. RESULTS & DISCUSSION

Model	Accuracy Achieved
Support Vector Machine	89%
K-Nearest Neighbor	80%
Convolutional Neural Network	96.75%

Table 8.1 Accuracy of implemented models

After conducting experiments with SVM, KNN, and CNN models for plant disease detection, it was found that the CNN model outperformed both the SVM model and the KNN model by large margins in terms of accuracy and precision.

In addition to the training and validation results, the performance of the CNN model was further assessed by passing single images of different classes through the model and evaluating its classification accuracy.

This additional assessment serves as an important validation of the model's ability to accurately classify images of different classes, and indicates that the model is capable of effectively generalizing to new, unseen data. The successful classification of these images further supports the performance of the model as suggested by the training and validation accuracy and loss curves, and provides additional evidence of its ability to accurately classify images in the eight classes.

Overall, the combination of the training and validation results, along with the successful classification of single images from different classes, provides strong evidence of the effectiveness and accuracy of the CNN model in accurately classifying instances in the nine classes.

9. CONCLUSION

Based on the experimental results, it can be concluded that the CNN model is best suited for plant disease detection tasks. This suggests that the convolutional layers of a CNN model can learn complex features from images that are useful for distinguishing between different classes of plant diseases. Additionally, the use of other techniques such as data augmentation and normalization during the training process can further enhance the performance of the CNN model.

While the CNN model demonstrated the highest accuracy and best performance among the three models, the SVM and KNN models also exhibited strong classification abilities, and may be suitable for different use cases or scenarios. The comparison between the models provides useful insights into the relative strengths and weaknesses of different classification algorithms, and highlights the importance of selecting the most appropriate model for a particular problem domain.

Overall, the CNN model provides a powerful and effective tool for plant disease detection that can be used in various applications, including precision agriculture and crop management.

10. LIMITATIONS AND FUTURE SCOPE

One of the primary limitations encountered was the restricted availability of high-quality datasets. As a result, the researchers had to rely on Kaggle as the only dependable source for data. Thus, the researchers had to carefully consider the limitations of the dataset and its potential impact on the study's outcomes.

The study determined that the Convolutional Neural Network (CNN) was the most effective algorithm for the classification of plant diseases. However, in order to gain a more comprehensive understanding of the subject matter, it would be beneficial to compare the performance of other algorithms such as Random Forest.

In order to improve the accessibility and usefulness of the model, it may be beneficial to explore the possibility of deploying it on a cloud platform such as TensorFlow Lite or ONNX. This would enable farmers and agriculturalists to more readily access and utilize the model, ultimately providing them with more effective tools for the detection and management of plant diseases.

11. GANTT CHART

ACTIVITY	TIME FRAME			
	January 2023	February 2023	March 2023	April 2023
Literature Survey				
Problem Definition				
Identification and Preprocessing of Dataset				
Development of Model				
Testing and Validation				
Documentation				

Proposed Activity Achieved Activity Ongoing Activity

Figure 11.1: Gantt chart for Study on Plant Disease Detection Using SVM, KNN & CNN

12. REFERENCES

[1] Maniyath, S. R., P V, V., M, N., R, P., N, P. B., N, S., & Hebbar, R. (2018). Plant Disease Detection Using Machine Learning. 2018 International Conference on Design Innovations for 3Cs Compute Communicate Control (ICDI3C).

[2] Arshiya S. Ansari, Malik Jawarneh, Mahyudin Ritonga, Pragti Jamwal. Improved Support Vector Machine and Image Processing Enabled Methodology for Detection and Classification of Grape Leaf Disease. Hindawi Journal of Food Quality.

[3] Yalcin, H., & Razavi, S. (2016). Plant classification using convolutional neural networks. 2016 Fifth International Conference on Agro-Geoinformatics (Agro-Geoinformatics).

[4] Muhammad Zaka-Ud-Din, Wakeel Ahmad, Sumair Aziz. Classification of Disease in Tomato Plants' Leaf Using Image Segmentation and SVM. Technical Journal, University of Engineering and Technology (UET) Taxila, Pakistan.

[5] Muthukrishnan Ramprasath, M.Vijay Anand, Shanmugasundaram Hariharan. Image Classification using Convolutional Neural Networks. International Journal of Pure and Applied Mathematics. Volume 119 No. 17 2018, 1307-1319.

[6] Jing Chen, Lingwang Gao. Visual Tea Leaf Disease Recognition Using a Convolutional Neural Network Model. Yearly journal, College of Plant Protection, China Agricultural University, Beijing 100193, China.

Equations Referred

1. https://www.researchgate.net/publication/342149372 [24/3/2023]
2. https://www.researchgate.net/publication/342149372 [24/3/2023]
3. https://www.scielo.br/j/aabc/a/VCt3WLtyDNCVbg9S3nXzSnt/?lang=en [24/3/2023]
4. https://www.scielo.br/j/aabc/a/VCt3WLtyDNCVbg9S3nXzSnt/?lang=en [24/3/2023]
5. http://www.ijste.org/articles/IJSTEV2I11015.pdf [24/3/2023]
6. https://data-flair.training/blogs/svm-kernel-functions [24/3/2023]
7. https://en.wikipedia.org/wiki/Euclidean_distance [23/4/2023]
8. https://deepchecks.com/glossary/rectified-linear-unit-relu/ [23/4/2023]
9. https://deepai.org/machine-learning-glossary-and-terms/softmax-layer [23/4/2023]

Websites Referred

1. https://www.kaggle.com/datasets/soumiknafiul/plantvillage-dataset-labeled [24/3/2023]
2. https://cropwatch.unl.edu/soybean-management/plant-disease [24/3/2023]
3. https://www.bulkrenameutility.co.uk/ [24/3/2023]