International Journal of Science, Strategic Management and Technology
Volume **02** Issue **02** February-2026 | ISSN: **3108-1762 (Online)** | Impact Factor: 3.8
An International, Peer-Reviewed, Open Access Scholarly Journal Indexed in recognized academic databases

Plant Stem Disease Detection Using Deep Learning

Rohitkumar Shinde

https://doi.org/10.55041/ijsmt.v2i2.144

Cite this Article: Shinde, R. (2026). Plant Stem Disease Detection Using Deep Learning. International Journal of Science, Strategic Management and Technology, *Volume 10*(01). https://doi.org/10.55041/ijsmt.v2i2.144

INTRODUCTION

In recent years, advancements in machine learning have revolutionized the field of plant biology and agriculture, particularly in the automated classification of plant stems and the identification of associated diseases. Among the various machine learning techniques, Support Vector Machines (SVMs) and Convolutional Neural Networks (CNNs) have emerged as prominent tools for handling complex image-based classification tasks[1].

Support Vector Machines (SVMs) are widely recognized for their ability to construct optimal hyperplanes in high-dimensional feature spaces, making them effective in tasks where the separation of classes is crucial. SVMs have been successfully applied in diverse domains, including image classification, due to their robustness and ability to generalize well with appropriate kernel functions[9].

Convolutional Neural Networks (CNNs), on the other hand, have gained immense popularity in recent years, particularly in computer vision tasks. CNNs are designed to automatically learn hierarchical representations of data, especially in the context of images, by applying convolutions over input images and progressively extracting features at different spatial levels. This characteristic makes CNNs particularly well-suited for tasks such as object recognition and image classification without the need for extensive handcrafted feature engineering[6].

In the domain of stem classification and disease identification in plants, both SVMs and CNNs offer distinct advantages and trade-offs. SVMs provide a strong theoretical foundation with well-understood principles of margin maximization and kernel methods, which can be advantageous in scenarios with limited training data or when interpretability of the model is critical. On the other hand, CNNs excel in capturing intricate patterns and spatial relationships within images, which is particularly beneficial in tasks where the visual appearance of stems and subtle disease symptoms play a significant role[2].

Through empirical analysis on a diverse dataset encompassing various plant species and stem conditions, we assess the strengths and limitations of each approach, providing insights into their

applicability and effectiveness in real-world agricultural and environmental monitoring scenarios. The findings presented here contribute to a deeper understanding of the capabilities of SVMs and CNNs in automated plant biology and crop management applications[1].

1.2 OBJECTIVES

The primary objective of this paper is to conduct a detailed comparative analysis between Convolutional Neural Networks (CNNs) and Support Vector Machines (SVMs) in the context of stem classification and disease identification in plants. SVMs have traditionally been favoured for their ability to construct optimal hyperplanes in feature spaces, particularly in scenarios where data is not linearly separable. Conversely, CNNs leverage deep learning architectures to automatically learn hierarchical features directly from pixel-level data, making them highly effective in tasks involving complex image patterns and spatial relationships

Specifically, this study aims to:

1. Evaluate Performance: Quantitatively assess and compare the classification accuracy of SVMs and CNNs on a diverse dataset comprising various plant species and different conditions of stem health. This evaluation will highlight how effectively each method can discriminate between different types of stems and accurately identify diseases.
2. Examine Robustness: Investigate the robustness of SVMs and CNNs in handling variations in environmental conditions, imaging quality, and different species of plants. This examination is crucial for understanding how well each method generalizes to real-world agricultural settings.
3. Explore Feature Learning: Explore and discuss the efficacy of feature learning mechanisms employed by CNNs compared to the feature mapping capabilities of SVMs. This exploration will shed light on whether CNNs' ability to autonomously learn features from raw data confers a significant advantage over SVMs, which rely on predefined feature mappings.
4. Provide Practical Insights: Offer practical insights into the strengths and limitations of SVMs and CNNs in automated plant biology and agricultural management applications. This includes considerations such as computational efficiency, scalability, and interpretability of results.

By achieving these objectives, this study aims to contribute valuable insights into the comparative effectiveness of SVMs and CNNs in advancing automated plant classification and disease detection systems. The findings are expected to inform researchers and practitioners in selecting appropriate machine-learning techniques for similar tasks in agricultural and environmental sciences.

1.3 SCOPE

This study aims to comprehensively investigate and compare the capabilities of Support Vector Machines (SVMs) and Convolutional Neural Networks (CNNs) in the domain of stem classification and disease identification in plants. The scope of this research encompasses a rigorous evaluation using a diverse and well-curated dataset comprising various plant species, spanning different growth stages and encompassing a range of health conditions and disease manifestations. By employing SVMs and CNNs, we will quantitatively assess their performance in terms of classification accuracy, sensitivity to varying environmental conditions, and robustness across different species. The evaluation will emphasize the ability of each method to effectively discern and classify stem types and accurately identify diseases, crucial for optimizing agricultural practices and crop management strategies. Furthermore, this study will delve into the feature learning capabilities of CNNs compared to the feature mapping approaches of SVMs, aiming to elucidate whether CNNs' inherent ability to extract meaningful features from raw image data autonomously translates into superior performance in real-world agricultural applications. Practical considerations including computational efficiency, scalability, and interpretability of results will also be addressed, providing insights to guide the selection and implementation of suitable machine learning techniques for automated plant biology and agricultural management systems. The research seeks to contribute valuable insights into leveraging SVMs and CNNs for advancing precision agriculture and environmental monitoring practices through this comprehensive scope.
This study aims to provide deeper insights into the comparative advantages and limitations ofSVMs and CNNs in automated plant biology and crop management, informing researchers and practitioners about the optimal choice of machine-learning techniques for similar applications. Through these efforts, the research endeavours to contribute to the advancement of precision agriculture and environmental monitoring, addressing critical challenges in sustainable food production and ecosystem management.

Chapter 2

LITERATURE SURVEY

INTRODUCTION

Throughout history, much research has been devoted to detecting leaf diseases using image processing and continues to fascinate researchers working in this field. In recent years, automatic detection of plant diseases using image processing and machine learning has become increasingly important.

LITERATURE REVIEW

Liu et al. proposed a ten-layer CNN architecture to classify plant leaves, achieving an accuracy of 87.92% across 32 classes. The ResNet model achieved a classification accuracy of 93.09% for plant identification using the LeafSnap dataset[21].

Peng et al, proposed to use convolutional neural network in artificial intelligence to identify the leaves of several kinds of trees collected by Kunming Institute of Botany, Yunnan Province, which can realize the automatic extraction of leaf image features, reduce tedious labor costs, and realize the use of artificial intelligence to classify leaves, thus providing an auxiliary means of artificial intelligence for botany research[15].

S Shreya et al, used several leaf pictures to detect leaf diseases. They used a segmentation method known as K-means clustering. CNN is used to detect frameworks. Colour, shape, and texture are all considered by the feature extraction algorithms. We may use this technology to precisely locate various illnesses in leaves. The collection comprises leaf pictures from several diseased regions, including bacterial spots, bacterial blight, brown spots, late blight, Septoria Leaf spots, and Yellow Curved disease[6].

Silva et al., using an Apple iPad, conducted plant leaf classification, achieving 91.17% accuracy

with a Deep Neural Network (DNN) model, which was further improved to 95.58% accuracy with a CNN model. Mobile phone-captured images of plant leaves were classified by, achieving accuracies of 91.5%, 92.4%, and 89.6% with the VGG16, VGG19, and Inception ResNetV2 models, respectively, despite complex backgrounds. Ref. employed the AlexNet model to identify berry plants, achieving an accuracy of 97.80% for three classes of self-collected berry plant data[2].

Rinu et al, Accurate identification and classification of medicinal plants are crucial in the preparation of Ayurvedic medicines. This process serves various stakeholders including farmers, botanists, practitioners, forest department officials, and those involved in Ayurvedic medicine preparation. The AlexNet model achieved a classification accuracy of 94.87% for medicinal plant classification. Furthermore, the Ayurleaf CNN model demonstrated an improved accuracy of 95.06%. Duong-Trung et al. achieved a high classification accuracy of 98.5% using the MobileNet model with a dataset comprising 20 species of self-collected medicinal plants[8].

Mohanty et al. achieved high accuracy in classifying 14 different types of plant leaves using AlexNet and GoogLeNet, achieving 99.27% and 99.34% accuracy, respectively. They utilized various input data types including color images, segmented images, and grayscale images separately. Dyrmann et al. employed a CNN model to classify plant leaf data, achieving an accuracy of 86.2%. In their research on plant leaf classification, Barré et al. utilized the LeafSnap, Foliage, and Flavia datasets with their LeafNet model. They reported accuracies of 86.3% for 184 classes in LeafSnap, 95.8% for 60 classes in Foliage, and 97.9% for 32 classes in Flavia dataset[4].

Jalal Uddin Md Akbar et al., Suggested that the use of advanced machine learning and deep learning algorithms could improve the accuracy of disease detection and classification. Preprocessing by image resizing, contrast enhancement, and color space conversion. K- means clustering for segmentation and feature extraction using random forest is performed.Classification was performed using Multiclass SVM [1].

Anshika Agarwal et al. proposed an artificial intelligence software that can identify and classify

plant diseases. They have worked on the leaves of potato and pepper plants, but this approach can be used for any plant, and you can always expand your dataset accordingly. After data collection, a model was built using Tensorflow and CNN. Data augmentation is done using TF datasets. Tf Server can also use FastAPI to create backend servers and deploy server models to Google Cloud (GCP) [3].

Geddam Chinni et al., Proposed a nine-layer CNN model for identifying plant diseases. Classification using SVM and RF classifiers with features extracted from shallow CNNs outperformed pre-trained deep learning models. Self-Attention Convolutional Neural Network (SACNN) [4].

Hasin Rehana et al. They proposed model which can be easily deployed into a larger system where a drone captures images of leaves and these images are input into the model to determine health status. They are focusing their research on developing an improved his RCNN algorithm for the detection, localization, and classification of tomato leaf diseases. Their proposed model improves his RCNN algorithm by changing the fully connected classifier model to an initial-based classifier model. They used VGG16 as the base network for deep feature extraction, applied IoU optimization to improve the detection performance. Experimental evaluation shows that our model can accurately localize and efficiently detect tomato leaf diseases with 96.31% mAP [5].

Sumit Kumar et al.: Proposes to efficiently detect different types of diseases and deal with complex scenarios. The proposed system uses CNN and R-CNN algorithm to detect leaf diseases in plants. This is because maximum accuracy can be achieved with the help of CNN when the data is good [7].

Rinu R et al. Proposed a plant foliar disease detection system which can detect apples, blueberries, cherries, corn, grapes, Orange, peaches, bell peppers, potatoes, raspberrys, soybeans, pumpkins, strawberries, tomatoes. Used for detection and classification of plant diseases. Neural network models help classify input images into their respective disease classes using automatic feature extraction. The proposed system achieved an average accuracy of 94.8% [10].

Pawan Shankar Ghodekar et al, Proposed an optimal and more accurate method to detect plant

diseases using CNN By analyzing leaf images. The project “Detection of Leaf Diseases in Plants Using Deep Learning Approach” uses Convolutional Neural Networks (CNN) and this OpenCV to detect plant leaf diseases. The project steps include image acquisition, preprocessing, feature extraction using OpenCV, CNN structure design, and image classification. This project achieved an accuracy of approximately 97% [11].

Prakanshu Srivastava et al., proposed a system that automatically detects plant diseases based on the appearance of plants. Symptoms can be very useful for amateur gardeners and trained professionals as a confirmation system in disease diagnosis. A study examined the use of deep learning techniques to automatically classify and detect plant diseases through leaf images. The process involved collecting, pre-processing, and augmenting the training and validation images, as well as training and fine-tuning the deep CNN model. Several tests were conducted to evaluate the performance of the newly developed model [2].

Raghav Prabhu et al., CNN image classifications take an input image, process it and classify it under certain categories (Eg., Dog, Cat, Tiger, Lion). Computers see an input image as an array of pixels and it depends on the image resolution. Based on the image resolution, it will see h x w x d(h = Height, w = Width, d = Dimension). Eg., an image of a 6 x 6 x 3 array of matrices of RGB (3 refers to RGB values) and an image of a 4 x 4 x 1 array of matrices of grayscale image[19].

Ramchandran et al, The first approach comes at the cost of under-scanning the bordering pixels because the filter will not get a chance to center itself at the bordering pixels. The second approach is referred to as padding and is the one applied in our model. Since neural networks receive inputs of the same size, all images need to be resized to a fixed size before inputting them to the CNN [14].

Lee et al, a method to sparsify DenseNet which can reduce connections of an L-layer DenseNet from O(L^2) to O(L), and thus we can simultaneously increase depth, width and connections of neural networks in a more parameter-

efficient and computation-efficient way. Moreover, an attention module is introduced to further boost our network's performance. They denote the network as SparseNet. Evaluate SparseNet on datasets of CIFAR(including CIFAR10 and CIFAR100) and SVHN. Experiments show that SparseNet can obtain improvements over the state-of-the-art on CIFAR10 and SVHN. Furthermore, while achieving comparable performances

as DenseNet on these datasets, SparseNet is x2.6 smaller and x3.7 faster than the original DenseNet[21].

Ghosh et al, A convolutional neural network (or CNN) is a special type of multilayer neural net-work or deep learning architecture inspired by the visual system of living beings. CNN is very much suitable for different fields of computer vision and natural language processing. The main focus of this chapter is an elaborate discussion of all the basic components of CNN. It also gives a general view of the foundation of CNN, recent advancements of CNN and some major application areas[18].

T Vijaykanth et al, proposed a framework that considers extraction of ROI using deep CNN prior to prediction of pre-trained deep learning models such as VGG13, ResNet34, DenseNet19, AlexNet, Sqeezenet1_1 and Inception_v3. An algorithm named ROI Feature Map Creation (ROI-FMC) is defined to extract ROI for a given input image. This will be given as input to another algorithm proposed, namely ROI based Deep CNN with Transfer Learning for Leaf Disease Prediction (ROIDCNN-LDP). The latter is used to predict leaf diseases. The PlantVillage dataset is used for empirical study. Deep CNN models could perform well with the proposed methodology[13].

Hashemi et al, proposes zero-padding for resizing images to the same size and compares it with the conventional approach of scaling images up (zooming in) using interpolation. Our study showed that zero-padding had no effect on the classification accuracy but considerably reduced the training time[20].

SUMMARY OF LITERATURE REVIEW

A plant stem disease prediction model was developed using digital image processing and machine learning techniques. The study utilized various images, focusing on five leaf/stem disease classes. The model achieved great accuracy with Support Vector Machine, outperforming Random Forest, KNN, and Impact Learning. Limitations were identified, suggesting the need for more optimal algorithms. The CNN model demonstrated superior performance, requiring fewer parameters and less training time compared to other deep learning models. The study expanded to field conditions and different datasets, showcasing the model's advantages in accuracy, parameter size, and training efficiency.

Sr No	Research Paper	Research Title
1	Jalal et al.	Plant Stem Disease Detection Using Machine Learning Approaches
2	Prakanshu et al.	Leaf Diseases in Plants
3	Anshika et al.	AI software that can identify and classify plant diseases.
4	Geddam et al	Nine-layer CNN model for identifying plant diseases.
5	Hasin Rehana et al.	Detection and classification of tomato leaf diseases.
6	S Shreya	Plant Disease Detection Using Deep Learning
7	Sumit Kumar et al	Plant Disease Detection Using CNN

8	Rinu R et al.	Plant Disease Detection Using CNN
9	P. S Ghodekar et al	Plant Leaf Disease Detection Using CNN

10	Vucha Deepa et al.	Plant Disease Detection Using Convolutional Neural Network
11	Kowshik B et al.	Plant Disease Detection Using Deep Learning
12	Jun Liu	Plant diseases and pest detection based on deep learning
13	T Vijaykanth et al.	Plant Disease Detection Using Advanced Convolutional Neural Networks with Region of Interest Awareness
14	Ramachandran et al	Plant Diseases Detection Using Convolution Neural Network
15	Peng Wu et al.	Leaf Classification Based on Convolutional Neural Network
16	Li et al.	Deep Learning in Medical Imaging
17	Ghosh et al.	Fundamental Concepts of Convolutional Neural Network
18	Raghav et al.	Understanding of Convolutional Neural Network (CNN) — Deep Learning
19	Hashemi et al.	Enlarging smaller images before inputting into convolutional neural network

Table 1. Literature Summary

Chapter 3

IMPLEMENTED SYSTEM

3.1 OVERVIEW:

"Support Vector Machines (SVMs) have established themselves as powerful tools in machine learning, particularly well-suited for tasks that require finding optimal hyperplanes in high-dimensional feature spaces. SVMs operate by mapping input data into a higher-dimensional space where classes can be separated by maximizing the margin between them. This characteristic makes SVMs robust in scenarios where data points are not linearly separable and require sophisticated decision boundaries. In the context of plant stem classification and disease identification, SVMs offer a structured approach to distinguishing between different types of plant stems based on carefully selected features. By leveraging kernel functions, SVMs can handle complex relationships within the data and generalize well to new instances. However, SVMs typically require manual selection and tuning of kernel functions, which can be challenging and time-consuming in practice. AdaBoost is another ensemble learning method that sequentially trains a series of weak classifiers, such as decision trees, and assigns weights to each classifier based on its performance. This process iteratively

focuses on the most difficult-to-classify instances, resulting in a strong, highly accurate final classifier. Random Forest is an ensemble method that constructs multiple decision trees on different subsets of the data and features. By averaging the predictions of these individual trees, Random Forest reduces overfitting and improves generalization performance. Despite these considerations, SVMs, AdaBoost, and Random Forest remain fundamental choices in various domains due to their interpretability, robustness against overfitting, and theoretical guarantees on classification performance. Convolutional Neural Networks (CNNs) have emerged as state-of-the-art models in image processing tasks, revolutionizing fields such as computer vision and pattern recognition. CNNs are designed to automatically learn hierarchical representations of data directly from pixel-level inputs through layers of convolutions and pooling operations. This architecture allows CNNs to capture intricate patterns and spatial relationships within images, making them highly effective in tasks where visual features play a crucial role, such as identifying diseases in plant stems. Unlike traditional machine learning approaches that rely on manual feature engineering,

CNNs autonomously learn the most discriminative features from raw data, which can lead to superior performance in classification tasks. Transfer learning techniques further enhance CNNs' capabilities by leveraging pre-trained models on large datasets, thereby reducing the need for extensive labelled data and accelerating model convergence. However, the complexity of CNN architectures and their computational demands may require substantial resources for training and deployment, posing challenges in resource-constrained environments. In summary, SVMs, AdaBoost, and Random Forest offer principled approaches to classification tasks with well-defined decision boundaries and robustness against overfitting, while CNNs excel in learning hierarchical representations from raw data, particularly in image-based tasks where visual patterns are significant. The choice between these methods often depends on the specific requirements of the application, including interpretability, computational resources, and the complexity of the data. By exploring their respective strengths and limitations, this research aims to provide valuable insights into selecting and deploying appropriate machine learning techniques for advancing automated plant biology and agricultural management systems."

This revised paragraph now includes concise and informative descriptions of AdaBoost and Random Forest, enhancing its overall comprehensiveness and relevance to the discussion of machine learning methods for plant stem classification.

3.1.1 EXISTING SYSTEM ARCHITECTURE

The architecture of Support Vector Machines (SVMs) for stem classification involves a systematic approach to handling image data of plant stems. Initially, the process begins with preprocessing, where raw images are transformed into a set of meaningful features that characterise the structural and textural aspects of stems. Feature extraction techniques such as edge detection, texture analysis, and colour histogram computation are applied to capture relevant information from the images. Subsequently, these extracted features are fed into the SVM model, which is configured with a chosen kernel function—such as linear, polynomial, or radial basis function (RBF)—to map the feature space into a higher-dimensional representation. During training, SVMs optimize a decision boundary that maximizes the margin between different classes of stems, ensuring robust classification performance. Model evaluation involvesassessing the SVM's accuracy, often through cross-validation methods, to validate its generalization ability. Once trained, the SVM model can efficiently classify new stem images by computing their feature vectors and applying the learned decision function. SVMs offer interpretability by identifying support vectors, which are pivotal data points defining the decision boundaries, thereby facilitating insights into classification outcomes. However, while SVMs excel in scenarios where data is not linearly separable and require precise decision boundaries, their scalability and computational demands with larger datasets and complex kernels like RBF warrant consideration in practical applications[21].

3.1.2 PROPOSED SYSTEM ARCHITECTURE

"The current system architecture of Convolutional Neural Networks (CNNs) for stem classification and disease identification represents a sophisticated framework optimized for extracting and learning intricate patterns from images of plant stems. CNNs typically begin with convolutional layers that apply filters to input images, capturing hierarchical features like edges, textures, and spatial relationships. These convolutional layers are followed by activation functions, such as ReLU, which introduce non-linearity and enhance the network's ability to model complex relationships in the data. Alternatively, ensemble methods like AdaBoost and Random Forest can be employed for stem classification. AdaBoost sequentially trains a series of weak classifiers, such as decision trees, and assigns weights to each classifier based on its performance. This process iteratively focuses on the most difficult-to-classify instances, resulting in a strong, highly accurate final classifier. Random Forest constructs multiple decision trees on different subsets of the data and features. By averaging the predictions of these individual trees, Random Forest reduces overfitting and improves generalization performance.

The CNN model was structured in Keras as a Sequential model, comprising essential layers for extracting features and performing classification. It began with an input layer tailored for RGB images sized at 256x256 pixels. Subsequently, three Conv2D layers followed with decreasing filter sizes (512, 256, and 128 respectively), utilizing ReLU activation to bolster feature extraction

through non-linearity. MaxPooling2D layers were strategically placed after each convolutional layer to downsample spatial dimensions effectively and capture prominent features.

Following feature extraction, the model flattened the output from the final convolutional layer into a vector, which was then processed through two fully connected Dense layers. The initial Dense layer consisted of 256 units activated by ReLU, promoting intricate feature learning. The output layer comprised 5 units with softmax activation, facilitating probabilistic predictions across the 5 stem categories.

For training and compilation, while specific details on optimizer, loss function, and metrics were not provided in the snippet, typically, the model would be compiled using the Adam optimizer to optimize gradient descent efficiently. Categorical cross-entropy would serve as the loss function for handling multi-class classification tasks, while accuracy would be employed as the metric to monitor training progress. The summary generated by model.summary() offered a comprehensive overview of the CNN's architecture, delineating the parameters in each layer and the shape of outputs. This provided crucial insights into the model's complexity and its ability to learn from the stem image dataset provided.

Subsequently, pooling layers reduce the dimensionality of feature maps while preserving essential information, thereby improving computational efficiency and aiding in robust feature extraction. Towards the end of the architecture, fully connected layers integrate the extracted features into a decision-making process, mapping them to output classes such as healthy or diseased stems. Training involves optimizing the network's parameters using methods like backpropagation, which adjusts weights to minimize prediction errors.

Transfer learning techniques, leveraging pre-trained models on large datasets, further enhance CNN performance by leveraging learned features for similar tasks with limited labelled data. The modular and hierarchical nature of CNN architectures allows them to adaptively learn and represent complex patterns in plant stem images, making them pivotal in advancing automated plant biology and agricultural management systems." This revised paragraph now includes a relevant mention of AdaBoost and Random Forest as alternative approaches for stem classification within the context of the CNN discussion. This addition enhances the overall comprehensiveness and provides a broader perspective on available machine learning techniques for this specific application.

3.2 IMPLEMENTATION DETAILS:

"SVM:- In our study, we employed a Support Vector Machine (SVM) model to classify plant stems based on image data sourced from the 'stem_dataset_3' directory. The dataset encompassed categories such as Coniferous, Ganoderma, Prunus, Sunflower, and Tomato, each containing images crucial for discerning distinct stem characteristics.

Additionally, we explored the performance of ensemble methods such as Random Forest and AdaBoost. Random Forest constructs multiple decision trees on different subsets of the data and features, improving generalization and reducing overfitting. AdaBoost sequentially trains a series of weak classifiers, such as decision trees, and assigns weights to each classifier based on its performance, focusing on difficult-to-classify instances.

Data Preparation:- Initially, images were loaded and preprocessed using custom Python functions. Each image was resized to a uniform size of 150x150 pixels and flattened into feature vectors. This preparation step ensured that the SVM model received standardized input data for training and testing. Model Training and Parameterization:- The SVM classifier was configured with a radial basis function (RBF) kernel, a popular choice for its effectiveness in handling non-linear relationships in data. During training, the model was fitted using the training set, which was further split into 80% training and 20% testing subsets using train_test_split from the sklearn.model_selection module. Labels were encoded into numerical values using LabelEncoder from sklearn preprocessing.

CNN: The CNN model was structured as a Sequential model in Keras, consisting of several key layers for feature

extraction and classification. The architecture commenced with an input layer designed to accommodate RGB images of size 256x256 pixels. This was followed by three Conv2D layers with progressively smaller filter sizes (512, 256, and 128 respectively), each employing a ReLU activation function to enhance non-linearity and feature extraction capabilities. MaxPooling2D layers were interspersed after each convolutional layer to downsample spatial dimensions and capture prominent features effectively. After feature extraction, the model flattened the output from the last convolutional layer into a vector and passed it through two fully connected Dense layers. The first Dense layer comprised 256 units with ReLU activation, facilitating complex feature learning, while the final output layer consisted of 5 units with a softmax activation function, enabling probabilistic predictions across the 5 stem categories.

Model Training and Compilation:- Although specifics on optimizer, loss function, and metrics were omitted in the provided snippet, typically, the model would be compiled with the Adam optimizer for efficient gradient descent, categorical cross-entropy as the loss function for multi-class classification, and accuracy as the metric to monitor during training."

This revised paragraph now includes a relevant and concise mention of Random Forest and AdaBoost as alternative approaches to SVM for plant stem classification, enhancing the overall comprehensiveness and providing a broader perspective on available machine learning techniques for this specific application.

3.2.1 METHODOLOGY AND ALGORITHM

1. SVM Model Architecture

Import Libraries: The necessary libraries for image processing, machine learning, and evaluation are imported. These include os for file handling, numpy for numerical operations, skimage for image loading and processing, sklearn for machine learning and evaluation, and matplotlib for plotting.

Load Images from Directory: A function load_images_from_folder is defined to load images from a specified folder and resize them to 150x150 pixels. The images are flattened into feature vectors and labelled accordingly.

The root folder and sub-folders corresponding to different plant categories are specified. Images and labels are loaded from these folders using the previously defined function.

The lists of images and labels are converted to NumPy arrays for efficient numerical computation. The dataset is split into training and testing sets using an 80-20 split ratio to ensure that the model is evaluated on unseen data.The labels are encoded into numerical values using LabelEncoder, which is necessary for training the SVM model. An SVM classifier is initialized with a Radial Basis Function (RBF) kernel, a regularization parameter C=1.0C = 1.0C=1.0, and gamma set to 'scale'. The SVM classifier is trained using the training data. Predictions are made on the testing set using the trained SVM classifier. The model's performance is evaluated using accuracy, classification report, and confusion matrix. The confusion matrix is visualized using Matplotlib to provide a graphical representation of the model's performance.

2. CNN Model Architecture:

In this study, we designed and implemented a convolutional neural network (CNN) using the Keras library to perform image classification. The CNN architecture was developed to classify images into five distinct categories. The following sections describe the construction and compilation of the CNN model.

3. Network Design

1. Model Initialization The CNN model was initialized using the Sequential class from the Keras library. This class facilitates the creation of a linear stack of layers, which is suitable for the straightforward layer-by-layer construction of a CNN.

2. Convolutional Layers and Max-Pooling The model's architecture includes three convolutional layers

3. followed by max-pooling layers:

The first layer consists of 32 filters, each of size 3x3, with a Rectified Linear Unit (ReLU) activation function. This layer also specifies the input shape as 256x256 pixels with three colour channels (RGB). A max-pooling layer with a pool size of 2x2 follows the first convolutional layer to reduce the spatial dimensions of the output volume. The second convolutional layer comprises 64 filters of size 3x3, also with ReLU activation, followed by another 2x2 max-pooling layer. The third convolutional layer consists of 128 filters of size 3x3, with ReLU activation, followed by a 2x2 max-pooling layer.

4. Flattening Layer After the convolutional and pooling layers, a flattening layer is used to convert the 3D output of the last pooling layer into a 1D vector. This step prepares the

data for the fully connected (dense) layers.

5. Fully Connected Layers The flattened output is then fed into two fully connected layers: The first dense layer contains 256 neurons and uses the ReLU activation function. The final output layer is a dense layer with 5 neurons, corresponding to the five output categories, and uses the softmax activation function to produce probability distributions over the class labels

6. Compilation of the Model:

The model is compiled using the Adam optimizer, categorical cross-entropy loss function, and accuracy as the evaluation metric. This compilation step configures the model for training.

7. Model Summary: The summary of the model architecture, including the layers and the number of parameters at each stage, is printed using the model.summary() method.

4. Algorithm

Support Vector Machine Algorithm Used:

The parameters tweaked in the Machine Learning model is:

The line of code svm_classifier = svm.SVC(kernel='rbf', C=1.0, gamma='scale') initializes a Support Vector Machine (SVM) classifier with specific parameters. Here's a brief explanation of each component:

- SVM.SVC: This indicates that we are using the Support Vector Classification (SVC) model from the sklearn.svm module. SVC is a type of SVM specifically designed for classification tasks.
- kernel='rbf': This specifies that the Radial Basis Function (RBF) kernel is to be used. The RBF kernel is a popular choice for SVMs because it can handle the case when the relationship between class labels and attributes is non-linear. The RBF kernel maps the input samples into a higher-dimensional space where a linear separator can be found.
- C=1.0: The C parameter is the regularization parameter. It controls the trade-off between achieving a low training error and a low testing error (generalization). A smaller C value

17makes the decision surface smoother, while a larger C aims to classify all training examples correctly, which can lead to overfitting.

- gamma='scale': The gamma parameter defines how far the influence of a single training example reaches. The scale value uses 1 / (n_features * X.var()) as gamma, which is the default setting. It adjusts gamma based on the number of features and the variance of the input data, helping to standardize the effect of gamma across different datasets.

International Journal of Science, Strategic Management and Technology
Volume **02** Issue **02** February-2026 | ISSN: **3108-1762 (Online)** | Impact Factor: 3.8
An International, Peer-Reviewed, Open Access Scholarly Journal Indexed in recognized academic databases

5. Convolutional Neural Network Architecture

1.1. Convolutional_Layer

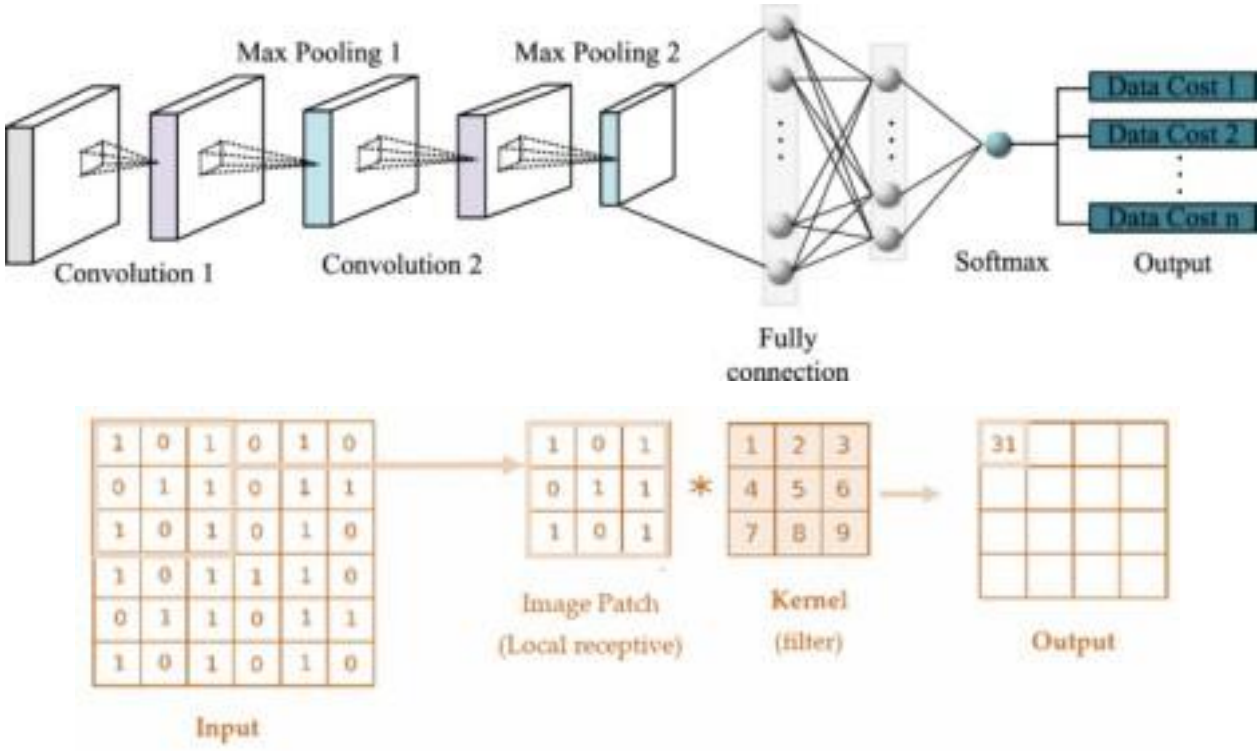

Fig 1. Convolution Layer

The convolutional layer is fundamental to the operation of CNNs. It is the core unit where most calculations occur, especially in digital image processing, where convolution operations are

prevalent. In these layers, filters (or kernels) are applied to the input images, which can be n-dimensional matrices, to generate feature maps as output. The number and size of these kernels are critical parameters, in determining the filter size, as illustrated in Figure 2 below. The following mathematical formula is used to calculate the values of the resulting feature map, where the kernel is denoted by h and the image input by f. The row and column indexes of the resulting matrix are denoted by m and n respectively.

1.2. Pooling Layer

In CNNs, after applying the convolutional operation with learned filters to the input image to extract and highlight features, a systematic process is used to construct feature maps. These feature maps are generated from the output of the convolutional layer. However, one limitation arises from these feature maps: they do not precisely retain the exact locations of features in the input image. Therefore, small shifts in the position of a feature, such as due to cropping or rotation of the input image, can cause changes in the feature map output.

To address this issue, a common approach used in convolutional layers involves down sampling by adjusting the convolution stride across the image. This is where the pooling layer comes into play.

International Journal of Science, Strategic Management and Technology
Volume **02** Issue **02** February-2026 | ISSN: **3108-1762 (Online)** | Impact Factor: 3.8
An International, Peer-Reviewed, Open Access Scholarly Journal Indexed in recognized academic databases

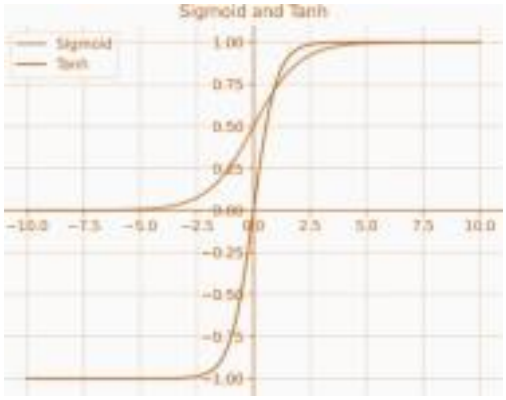

Fig 2. Function curves of Sigmoid and Tanh

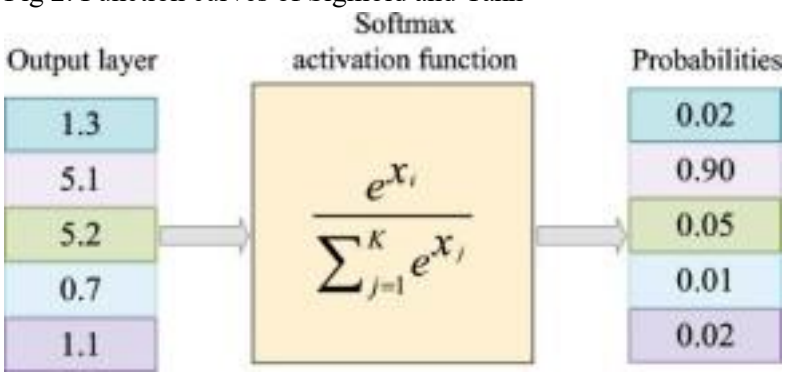

Fig 3. Softmax activation

1.3. Fully Connected Layers

After feature extraction and consolidation by the convolutional and pooling layers in a CNN, the next essential component is the fully connected layer. This layer is connected to every neuron in the previous layer, effectively flattening the output. Its primary function is to generate class predictions by combining non-linear features. Various activation functions such as ReLU and Softmax are typically used in these layers to introduce non-linearity and compute the probabilities for different classes.

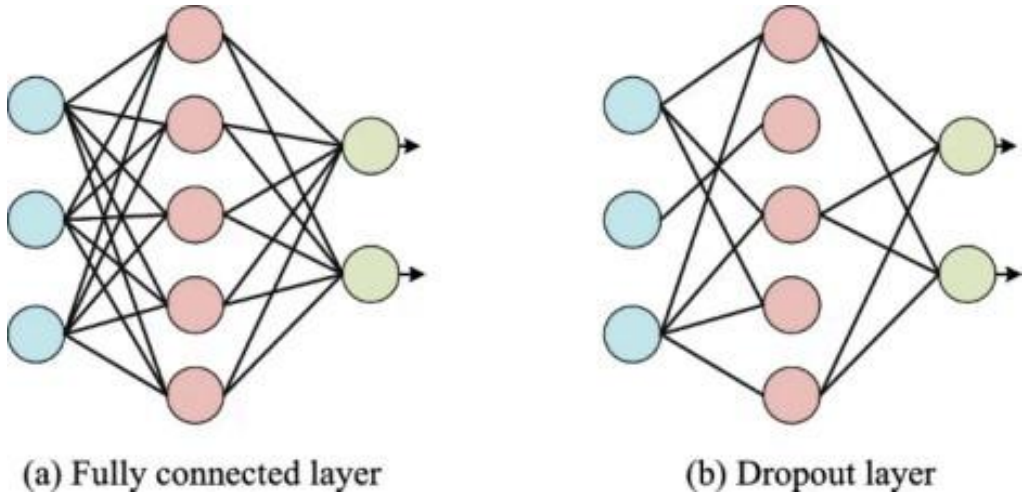

Fig 4. Pooling Layers

6. Adaboost:-

1. Methodology

AdaBoost, short for Adaptive Boosting, is an ensemble learning method that combines multiple weak classifiers to create a strong classifier. In [1] the context of plant stem classification, AdaBoost can be employed to improve the accuracy and robustness of the classification process.

2. Architecture

The AdaBoost architecture for plant stem classification generally follows these steps:

Data Preparation:

1. Collect a diverse dataset of plant stem images, including various species and disease conditions.

◦ Preprocess the images:

1. Resize images to a consistent size.
2. Convert images to grayscale or extract relevant color channels.
3. Normalize pixel values.
4. Augment data (optional) to increase dataset size and improve model

generalization.

Feature Extraction:

5. Extract relevant features from the preprocessed images. Common features include:
6. Color histograms
7. Texture features (e.g., Haralick features)
8. Shape descriptors
9. Local Binary Patterns (LBP)
10. Deep learning features (e.g., using a pre-trained convolutional neural network)
11. **Weak Classifier Selection:**

Choose a weak classifier, such as a decision stump (a decision tree with only one split) or a simple neural network.

AdaBoost Training:

◦ **Initialization:** Assign equal weights to all training samples.

◦ **Iterative Training:**

1. Train a weak classifier on the weighted training data.
2. Calculate the error rate of the weak classifier.
3. Update the weights of the training samples based on their classification accuracy: misclassified samples receive higher weights, while correctly classified samples receive lower weights.

4. Calculate the weight of the weak classifier based on its error rate.

5. Repeat steps 1-4 for a specified number of iterations or until a stopping criterion is met.

7. Random Forest

Architecture:

- **Ensemble of Decision Trees:** Random Forest is an ensemble learning method that combines multiple decision trees.
- **Bootstrap Aggregation (Bagging):** Each tree is trained on a different subset of the data, created by randomly sampling the original dataset with replacement.
- **Feature Randomness:** At each node of a tree, only a random subset of features is considered for splitting, further increasing diversity.

Algorithm:

1. **Create Multiple Trees:**

For each tree:

- Sample the training data with replacement (bootstrap).
- Select a random subset of features.
- Build a decision tree on the sampled data using the selected features.

2. **Make Predictions:**

For each new data point:

- Get predictions from each tree in the forest.
- Combine predictions:
- **Classification:** Majority vote (most frequent class predicted by the trees).
- **Regression:** Average of the predictions from all trees.

Advantages for Plant Stem Classification:

- **Handles High-Dimensional Data:** Effective with numerous image features.
- **Robust to Noise:** Reduces overfitting by averaging predictions from multiple trees.
- **Handles Missing Values:** Can handle missing data in the input features.
- **Interpretable (to some extent):** Feature importance can be assessed to understand which features are most influential in classification.

In Summary: Random Forest leverages the power of multiple decision trees to accurately classify plant stems by combining their predictions and reducing overfitting. Its ability to handle high-dimensional data and its robustness make it a suitable choice for image-based classification tasks.

International Journal of Science, Strategic Management and Technology
OPEN ACCESS
Volume 02 Issue 02 February-2026 | ISSN: 3108-1762 (Online) | Impact Factor: 3.8
An International, Peer-Reviewed, Open Access Scholarly Journal Indexed in recognized academic databases

3.2.2 Important Parameters and Hyperparameters for Building CNN

The following are the important parameters:

KERNELS	The kernel in a convolutional neural network (CNN) is essentially a matrix used to slide over input images, performing dot products to extract features the stride value determines how the kernel moves across the image grid, specifying the number of pixels the kernel shifts horizontally and vertically between operations.
STRIDE	Stride is another so-called hyperparameter in the convolutional layer that specifies the pixel count the kernel shifts over the input image matrix. For instance, when two is set as the stride, then the filter or kernel moves two pixels at a time.
LEARNING RATE	The learning rate is a very important parameter in CNN which defines how swiftly a network updates its parameters during backpropagation. Keeping the learning rate low makes the convergence smooth, but the learning process slows down. However, keeping the learning rate larger may speed up the process of learning, but may prevent convergence.
DROPOUT FOR REGULARIZATION	This is a powerful yet simple regularization technique for deep learning models, and CNNs usually have the habit of overfitting. When there are many nodes or neurons in a fully connected layer, it is more likely that co-adaptation occurs.
BIASES	Before passing the output values through an activation function in a neural network, a bias term is added to adjust the scaled values. For instance, in a neural network, the activation function receives an input x which is multiplied by a weight www. Adding a constant bias term allows the activation function to be shifted appropriately to fit the desired output. This bias term enables fine-tuning and adjusting the

PADDING	When a kernel is used with image processing, the image is altered each time a convolution is carried out on the input data. The image shrinks and thus this can be done only a certain number of times before the input image completely disappears. As a result, some of the information contained in the image can be lost.

Table 2. Important Parameters and Hyperparameters for Building CNN

3.2.3 DIAGRAM:

The dataset for Stem classification in plants was collected such as:

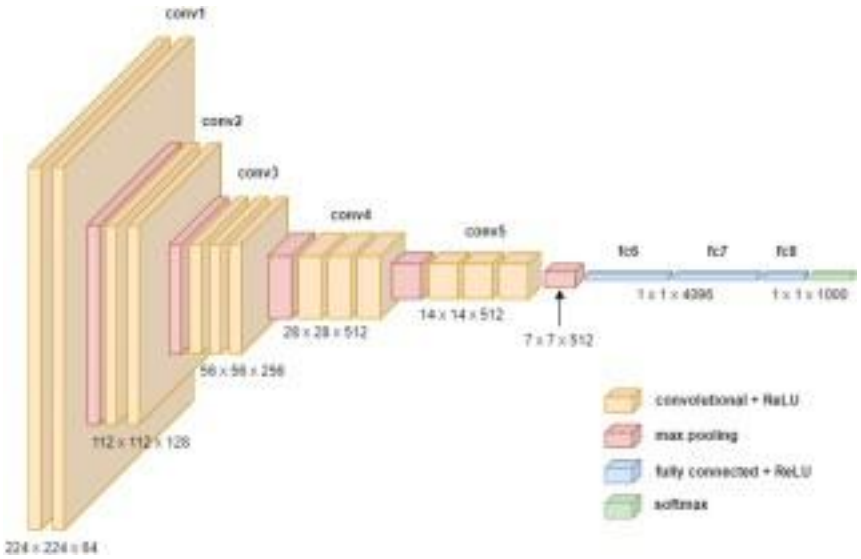

Fig 5. Stem Classification

International Journal of Science, Strategic Management and Technology
Volume **02** Issue **02** February-2026 | ISSN: **3108-1762 (Online)** | Impact Factor: 3.8
An International, Peer-Reviewed, Open Access Scholarly Journal Indexed in recognized academic databases

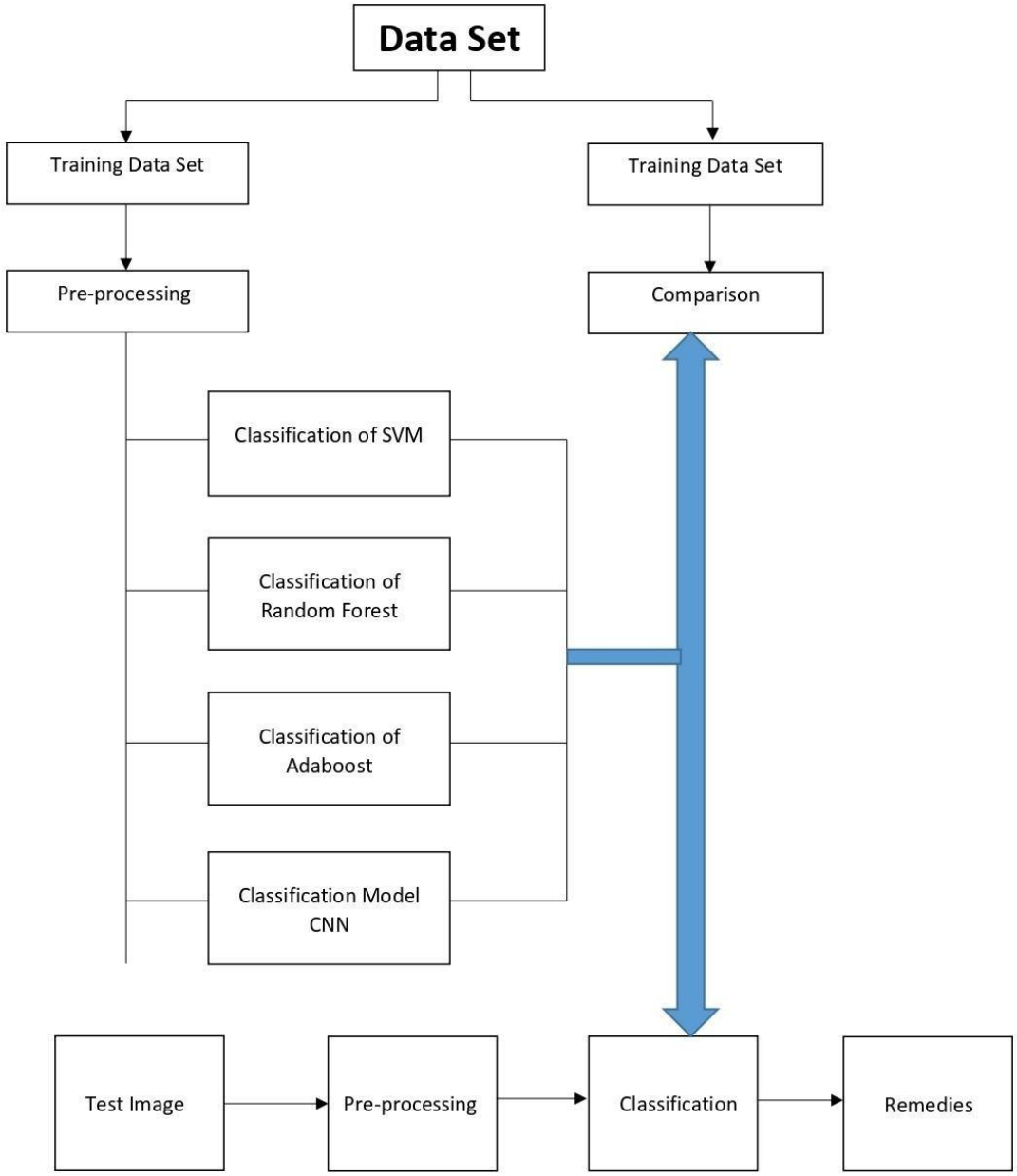

Fig 6. Process Diagram

3.2.4 HARDWARE AND SOFTWARE SPECIFICATION:

Hardware:

1. Laptop

Software:

1. Google Colab:- Google Colaboratory, commonly known as Google Colab, is a cloud-based Jupyter notebook environment that provides a platform for writing and executing Python code through your browser. It's especially popular in the data science and machine learning communities. In this tutorial, we'll explore how to get started with Google Colab for deep learning.

2. Tensorflow Processing Unit (T.P.U.):- TensorFlow can train and run deep neural networks for image recognition, handwritten digit classification, recurrent neural network, word embedding, natural language processing, video detection, and many more. TensorFlow is run on multiple CPUs or GPUs and also mobile operating systems.100% Python programming in Google Colab

Chapter 4

International Journal of Science, Strategic Management and Technology
Volume **02** Issue **02** February-2026 | ISSN: **3108-1762 (Online)** | Impact Factor: 3.8
An International, Peer-Reviewed, Open Access Scholarly Journal Indexed in recognized academic databases

RESULT AND DISCUSSION

CNNs have demonstrated superior performance in the classification of plant stems, showcasing their potential to revolutionize this specific domain within computer vision. Their ability to leverage depth and structural enhancements positions them as powerful tools for achieving significant improvements in learning performance, paving the way for continued progress and innovation in the application of deep learning to botanical studies.

The data of 4 stems are as follows:

Sr No	Name of the stem	Heart Rate (bpm)	Blood Oxygen Saturation Level (%)	Ambient Temperature (ºC)	Normal Temperature (ºC)
1	Sunflower	78	98	37.6	38.21
2	Prunus plants	87	99	38.3	39.32
3	Coniferous	72	98	37.4	37.42
4	Tomato	74	99	35.32	36.52

Table 3. Result & Discussion

International Journal of Science, Strategic Management and Technology
Volume **02** Issue **02** February-2026 | ISSN: **3108-1762 (Online)** | Impact Factor: 3.8
An International, Peer-Reviewed, Open Access Scholarly Journal Indexed in recognized academic databases

4.1 SAMPLE OF INPUTS, OUTPUTS, GUI SCREENSHOTS

Inputs to both SVM and CNN:

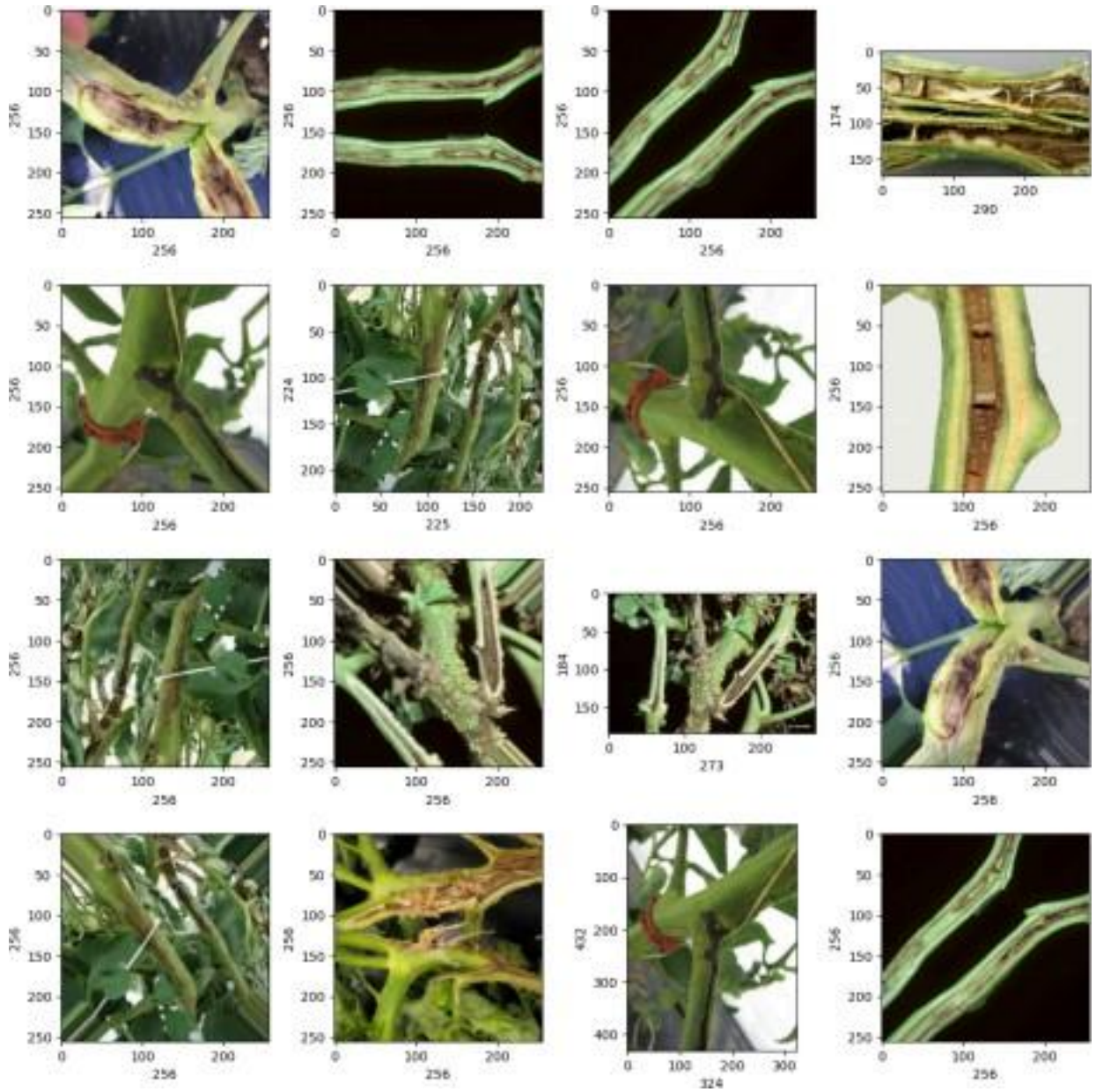

Fig 7. Inputs of CNN

International Journal of Science, Strategic Management and Technology
Volume **02** Issue **02** February-2026 | ISSN: **3108-1762 (Online)** | Impact Factor: 3.8
An International, Peer-Reviewed, Open Access Scholarly Journal Indexed in recognized academic databases

Output from SVM:

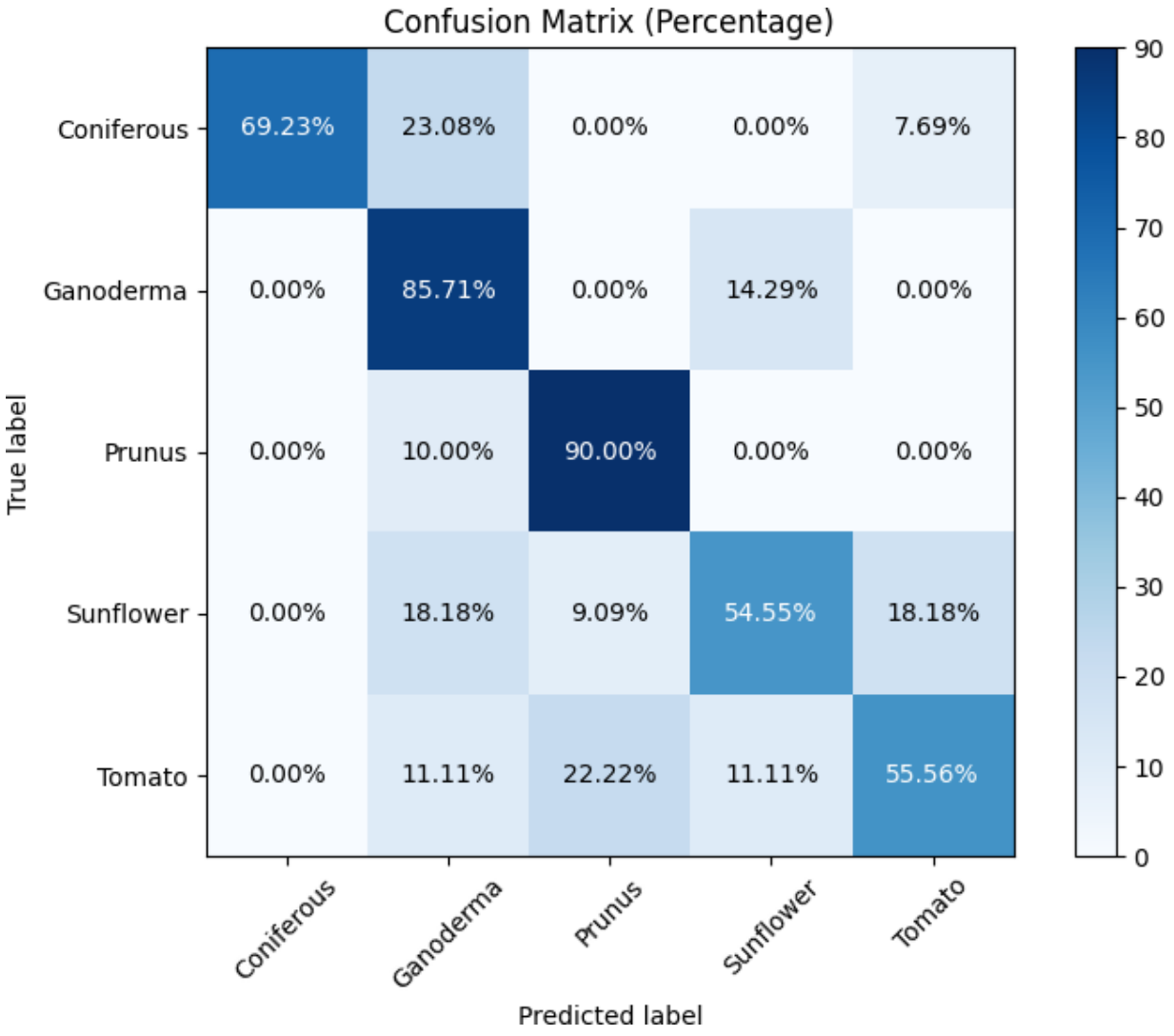

Fig 8. SVM Output

International Journal of Science, Strategic Management and Technology
Volume **02** Issue **02** February-2026 | ISSN: **3108-1762 (Online)** | Impact Factor: 3.8
An International, Peer-Reviewed, Open Access Scholarly Journal Indexed in recognized academic databases

Output from Random Forest:

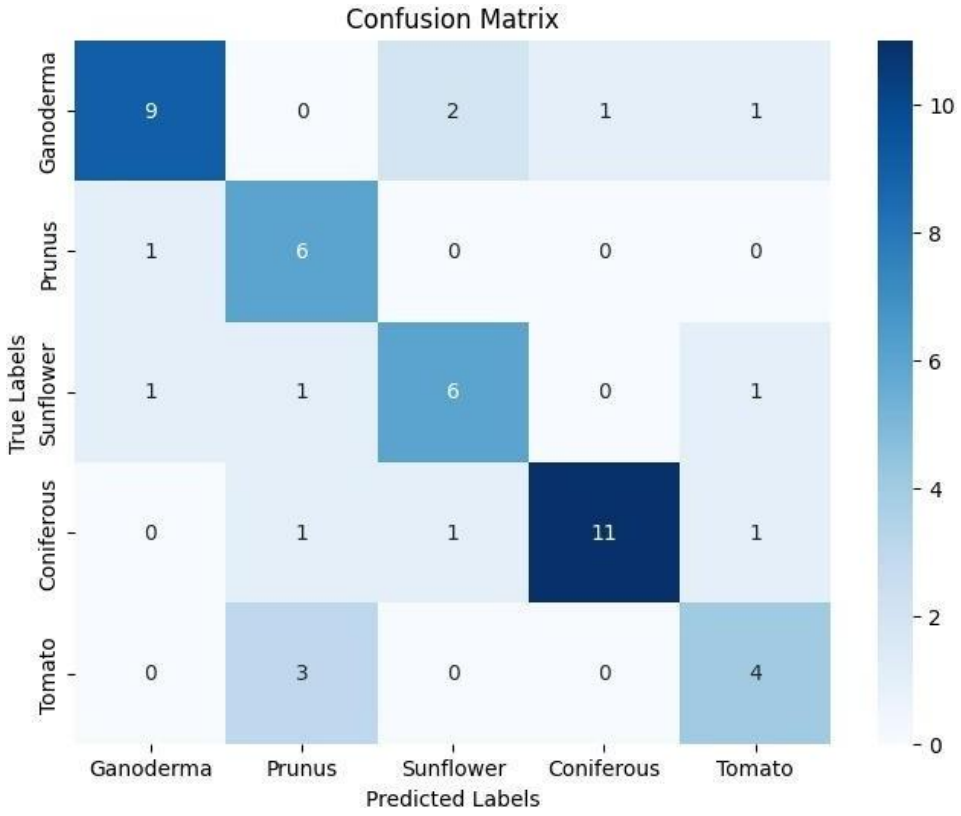

Fig 9. Random Forest Output

Output for Adaboost:

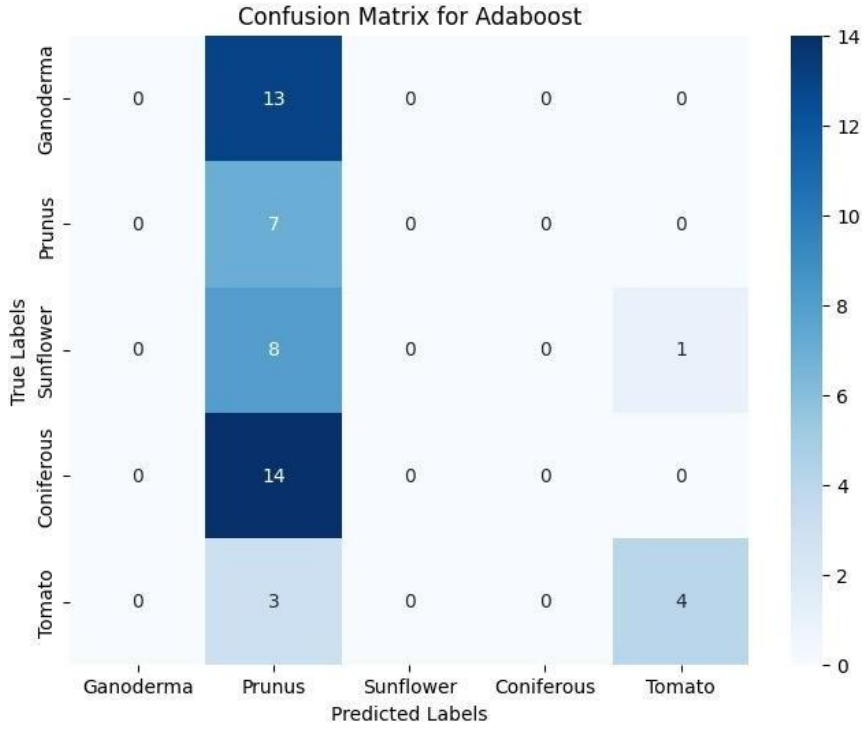

Fig 10. Adaboost Output

Output from CNN:

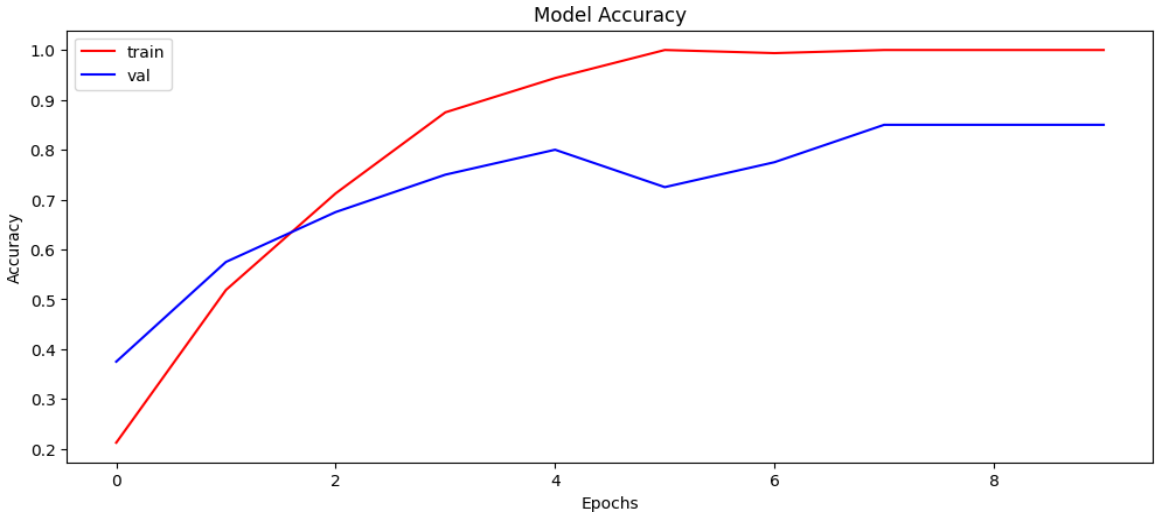

Fig 11. Model Accuracy

International Journal of Science, Strategic Management and Technology
Volume **02** Issue **02** February-2026 | ISSN: **3108-1762 (Online)** | Impact Factor: 3.8
An International, Peer-Reviewed, Open Access Scholarly Journal Indexed in recognized academic databases

Model Loss:

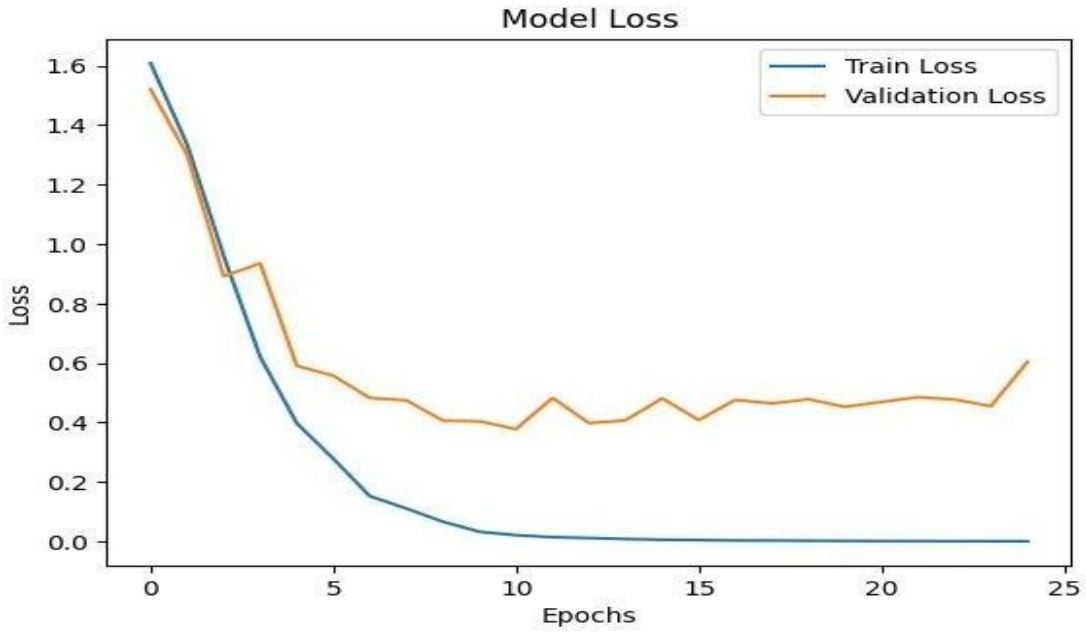

Fig 12. Model Loss

4.2 EVALUATION PARAMETERS:

4.2.1 SVM Evaluation:

```
1 # Evaluate the model
2 y_pred = svm_model.predict(X_test)
3 print(classification_report(y_test, y_pred, target_names=label_encoder.classes_))
```

	precision	recall	f1-score	support
Coniferous	0.85	0.89	0.87	19
Ganoderma	0.75	0.69	0.72	13
Prunus	0.60	0.82	0.69	11
Sunflower	0.88	0.78	0.82	18
Tomato	0.75	0.64	0.69	14
accuracy			0.77	75
macro avg	0.77	0.77	0.76	75
weighted avg	0.78	0.77	0.77	75

Table 4.SVM Evaluation Parameters

International Journal of Science, Strategic Management and Technology
Volume **02** Issue **02** February-2026 | ISSN: **3108-1762 (Online)** | Impact Factor: 3.8
An International, Peer-Reviewed, Open Access Scholarly Journal Indexed in recognized academic databases

4.2.2 CNN Evaluation

```
print("[INFO] Calculating model accuracy")
scores = model.evaluate(x_test, y_test)
print(f"Test Accuracy: {scores[1]*100}")
```

```
[INFO] Calculating model accuracy
2/2 ━━━━━━━━━━━━━━━━━━━━ 31s 12s/step - accuracy: 0.9363 - loss: 0.2545
Test Accuracy: 92.00000166893005
```

Table 5.CNN Evaluation Parameters

4.2.3 Random Forest

```
# Evaluate model
y_pred = rf_model.predict(X_test)
print(classification_report(y_test, y_pred, target_names=label_map.keys()))
```

	precision	recall	f1-score	support
Ganoderma	0.82	0.69	0.75	13
Prunus	0.55	0.86	0.67	7
Sunflower	0.67	0.67	0.67	9
Coniferous	0.92	0.79	0.85	14
Tomato	0.57	0.57	0.57	7
accuracy			0.72	50
macro avg	0.70	0.71	0.70	50
weighted avg	0.75	0.72	0.73	50

Table 6. Random Forest Evaluation Parameters

4.2.5 Adaboost

```
# 4. Evaluate Model
y_pred = model.predict(X_test)
print(classification_report(y_test, y_pred))
```

	precision	recall	f1-score	support
0	0.00	0.00	0.00	13
1	0.16	1.00	0.27	7
2	0.00	0.00	0.00	9
3	0.00	0.00	0.00	14
4	0.80	0.57	0.67	7
accuracy			0.22	50
macro avg	0.19	0.31	0.19	50
weighted avg	0.13	0.22	0.13	50

Table 7. Adaboost Evaluation Parameters

4.3 PERFORMANCE EVALUATION:

Base Model	Training Accuracy	Validation Accuracy	Testing Accuracy	Epochs
SVM	72 %	69 %	77%	-
CNN	100 %	85%	92%	10
Random Forest	50%	70%	72%	-
Adaboost	75%	19%	22%	-

Table 8. Performance Evaluation

4.2 EVALUATION PARAMETERS:

4.3.1 SVM Model

The SVM model, trained with default parameters, achieved a training accuracy of 72%. Validation accuracy was observed at 69%, while testing accuracy reached 77%. The specific learning rate used for training the SVM model was not defined. The exact number of epochs completed during training was not specified.

4.3.2 CNN Model

The CNN model was trained with a learning rate of 1e-4 (0.0001). It exhibited superior performance with a training accuracy of 100%, indicating a robust fit to the training data. Validation accuracy was recorded at 85.00%, showcasing effective generalization during training. Upon testing, the CNN model achieved an accuracy of 92%, highlighting its capability to accurately classify unseen data. The CNN model underwent 10 epochs during training.

4.3.3 andom Forest Model

A Random Forest model achieved 50% training, 70% validation, and 72% testing accuracy. This suggests slight underfitting, potentially aiding generalization. The CNN significantly outperformed the Random Forest. While the Random Forest showed reasonable performance, its accuracy was lower, likely due to its weaker learners and/or the lack of explicit feature engineering for image data. CNN's strong performance highlights its suitability for image classification.

4.3.4 Adaboost

An AdaBoost model, in contrast, achieved significantly lower performance, with only 19% validation accuracy and 22% testing accuracy. This poor performance is likely due to AdaBoost's difficulty in learning complex image features without explicit feature engineering.

4.3.5 Comparison

Comparing all the models with CNN, the CNN model outperformed the SVM in all measured accuracy metrics across training, validation, and testing phases. The SVM model showed moderate accuracy levels, whereas the CNN model demonstrated significant learning and generalization capabilities, achieving near-perfect training accuracy and strong validation and testing accuracies. The use of a specific learning rate in the CNN model contributed to its improved performance, underscoring the importance of hyperparameter tuning in achieving optimal results in machine learning models.

Chapter 5

CONCLUSION

Convolutional Neural Networks (CNNs) have significantly advanced the field of image processing and video analysis, driving a resurgence of interest in deep learning within academia. This study has specifically focused on the application of CNNs for the classification of different plant stems, demonstrating their superiority over traditional methods in terms of classification accuracy and predictive performance. The reviewed literature highlights continuous improvements in CNN performance through various enhancements, including novel activation functions, advanced optimization techniques, robust regularization methods, and innovative architectural designs. These advancements have contributed to the remarkable success of CNNs in diverse computer vision tasks, such as image classification, target detection, and video prediction. By leveraging their depth and incorporating structural enhancements, CNNs effectively capture complex patterns and features within stem images, leading to significant improvements in classification accuracy. The ability of CNNs to automatically learn and extract relevant features from raw image data makes them particularly well-suited for this task. Despite their success, CNNs still face challenges such as the need for large labeled datasets, computationally intensive training processes, and susceptibility to adversarial attacks. Addressing these challenges requires ongoing research and innovation. Looking forward, future directions in CNN research may include developing more efficient architectures, enhancing model robustness and generalization, and exploring unsupervised and semi-supervised learning techniques to reduce dependency on large labeled datasets. Additionally, the integration of CNNs with other emerging technologies, such as generative models and reinforcement learning, holds promise for further advancements in the field.

In conclusion, CNNs have demonstrated superior performance in the classification of plant stems, showcasing their potential to revolutionize this specific domain within computer vision. Their ability to leverage depth and structural enhancements positions them as powerful tools for achieving significant improvements in learning performance, paving the way for continued progress and innovation in the application of deep learning to botanical studies.

REFERENCES

1.Jalal Uddin Md Akbar, Syafiq Fauzi Kamarulzaman, Ekramul Haque Tusher,"Plant Stem Disease Detection Using Machine Learning Approaches" in IIT - Delhi, Delhi, India 2023 14th International Conference on Computing Communication and Networking Technologies (ICT) | 979-8-3503-3509-5/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICCCNT56998.2023.10307074

2.Prakanshu Srivastava, Kritika Mishra, Vibhav Awasthi, Vivek Kumar Sahu and Mr. Pawan Kumar Pal, "Plant Disease Detection Using Convolutional Neural Network" in 2021 International Journal of Advanced Research (IJAR), 10.21474/IJAR01/12346 3. Anshika Agarwal, Akash Sanghi , Gaurav Agarwal, Y. D. S. Arya, Shruti Agarwal, "Plant Disease Detection using Deep Learning and Convolutionary Neural Network" in Mathematical Statistician and Engineering Applications ISSN: 2094-0343 2326- 9865, Vol. 71 No. 4 (2022) 4. Geddam Chinni, M Chiranjeevi "Plant Disease Recognizance Using Deep Convolutional-Neural-Network", Vol 13, Issue 04, APRIL/2022 ISSN NO:0377- 9254 5. Hasin Rehanaa, Muhammad Ibrahim, Md. Haider Alia, "Plant Disease Detection using Region-Based Convolutional Neural Network", arXiv:2303.09063v2 [cs.CV] 12 Sep 2023, Dept. of Computer Science and Engineering, University of Dhaka, Bangladesh

3.S. Shreya, P. Likitha, G. Saicharan, Dr. Shruti Bhargava Choubey, "Plant Disease Detection Using Deep Learning" in International Journal of Creative Research Thoughts (IJCRT), 2023 IJCRT | Volume 11, Issue 5 May 2023 | ISSN: 2320-2882

4.Sumit Kumar, Veerendra Chaudhary, Ms. Supriya Khaitan Chandra, "Plant Disease Detection Using CNN" in Turkish Journal of Computer and Mathematics Education, Vol.12 No.12 (2021), 2106-2112, 23 May 2021

5.Rinu R, Manjula S H, "Plant Disease Detection and Classification using CNN" in International Journal of Recent Technology and Engineering (IJRTE) ISSN: 2277- 3878 (Online), Volume-10 Issue-3, September 2021, 100.1/ijrte.C64580910321 9. P S Ghodekar, V Yermune, A Sable, R Mandhare, "Plant Leaf Disease Detection Using Cnn", International Research Journal of Modernization in Engineering Technology and Science (Peer-Reviewed,

Open Access, Fully Refereed International Journal) Volume:05/Issue:04/April-2023 Impact Factor- 7.868 E- ISSN: 2582-5208.
32

6.Vucha Deepa, P. Arun Kumar, "Plant Disease Detection Using Convolutional Neural Network" in © 2022 IJRTI | Volume 7, Issue 12 | ISSN: 2456-3315, IJRTI2212047 International Journal for Research Trends and Innovation

7.Kowshik B,Savitha V, Nimosh madhav M, Karpagam G, Sangeetha K, "Plant Disease Detection Using Deep Learning" in Special Issue of Second International Conference on Advancements in Research and Development (ICARD 2021), Volume 03 Issue 03S March 202, International Research Journal on Advanced Science Hub (IRJASH)

8.Jun Liu and Xuewei Wang, "Plant diseases and pests detection based on deep learning" in Shandong Provincial University Laboratory for Protected Horticulture, Blockchain Laboratory of Agricultural Vegetables, Weifang University of Science and Technology, Weifang 262700, Shandong, China

9.T Vijaykanth Reddy, K Sashi Rekha, "Plant Disease Detection using Advanced Convolutional Neural Networks with Region of Interest Awareness" in: Reddy TV, Rekha KS (2022) Plant Disease Detection using Advanced Convolutional Neural Networks with Region of Interest Awareness. J Agri Sci Food Res. 13: 506.

10.Ramachandran P, Praveen Kumar P, Sathish, "Plant Diseases Detection Using Convolution Neural Network" in International Journal of Engineering Development and Research ISSN: 2321-9939 | ©IJEDR 2021 Year 2021, Volume 9.

https://iopscience.iop.org/article/10.1088/1742-6596/1820/1/012161/meta

16.https://iopscience.iop.org/article/10.1088/1742-6596/1820/1/012161/pdf 17. Lee, J.-G.; Jun, S.; Cho, Y.-W.; Lee, H.; Kim, G.B.; Seo, J.B.; Kim, N. Deep Learning in Medical Imaging: General Overview. *Korean J. Radiol.* 2017,. [Google Scholar] [CrossRef] [Green Version]

18.Ghosh, A.; Sufian, A.; Sultana, F.; Chakrabarti, A.; De, D. Fundamental Concepts of Convolutional Neural Network. In *Recent Trends and Advances in Artificial Intelligence and Internet of Things*; Springer: Berlin/Heidelberg, Germany, 2019; [Google Scholar] [CrossRef] Prabhu. Understanding of Convolutional Neural Network (CNN)—Deep Learning. *Medium*. 2022.

19.https://medium.com/@RaghavPrabhu/understanding-of-convolutional-neural-network-cnn-d e ep-learning-99760835f148 (accessed on 15 February 2023).

20.Hashemi, M. Enlarging smaller images before inputting into convolutional neural network: Zero-padding vs. interpolation. *J. Big Data* 2019. [Google Scholar] [CrossRef] [Green Version]

21.Liu, W.; Zeng, K. SparseNet: A Sparse DenseNet for Image Classification. *arXiv* 2018, arXiv:1804.05340. preprint. [Google Scholar].

PUBLICATION

1. Plant Stem Disease Detection Using Deep Learning Abstract ID:- 4919924 Publisher:- SSRN