1. INTRODUCTION

Machine learning and IoT add technology to agriculture and lead to advancements such as precise monitoring of environmental conditions, systematic resource management, disease management, etc. The IoT-enabled systems with advanced machine learning models overcome challenges like irrigation optimization, crop health surveillance, and environmental sensing. For example, IoT-enabled devices and wireless sensor networks are extensively used to gather environmental data, and machine learning models analyse the collected environmental data to identify actionable insights. Source identification is important for agriculture because it predicts the conditions in which plant diseases and fungal growth are commonly triggered by moisture on the leaves. Conventional approaches rely on sensor data, but environmental conditions make it difficult to distinguish the source of the wetness: dew, rain, or irrigation. This research puts forward a hybridized model that classifies leaf-wetness sources based on data acquired from leaf-wetness sensors, including moisture and temperature.

A Multilayer Perceptron model was used to predict the diseases related to fungus based on atmospheric and soil sensor data. Their study illustrated the productiveness of the neural network for the environmental data [3]. Motivated by the high accuracy of 98% achieved by their model, our model uses the MLP-LSTM model to detect time-dependent patterns of wetness, which improves the robustness in the detection of the source of leaf wetness. The ensemble methods and deep Learning are used to forecast the susceptibility of floods, exhibiting high accuracy through combining models like deep learning and random forest for sturdy environmental predictions [8]. This approach stimulated the use of LSTM to increase classification accuracy in recognizing leaf wetness sources. The isolated model using Multilayer Perceptron alone achieved 92.86% accuracy with numerical data [21]. By employing Long Short-Term Memory in our proposed model, the accuracy is increased to 99.731%.

The LSTM networks employed to forecast plant diseases through analyzing the temporal relationships in the sensor data [11] motivated the addition of LSTM layers to the multilayer perceptron model to focus on time-dependent changes in the

wetness of the leaf. The integration of LSTM in the Multilayer Perceptron allows the incorporation of patterns that are time-dependent and are important in agriculture. With the growth of IoT in agriculture and smart systems, the necessity of detailed classification for applications in real-time is crucial for strategies of disease management and optimizing irrigation.

Traditional CNN-based plant disease detection lacks environmental context, limiting accuracy. The proposed hybrid deep learning approach integrates MLP-LSTM for wetness source classification and EfficientNetB0 for disease detection, achieving high accuracy. This study provides actionable insights for precision irrigation, enhancing disease forecasting and reducing crop losses [25].

Existing CNN-based plant disease detection faces challenges like dataset reliance and environmental variability. The proposed hybrid deep learning model integrates MLP-LSTM for wetness classification and EfficientNetB0 for disease detection, improving accuracy. This approach enhances real-world applicability by addressing environmental factors and supporting sustainable agriculture [26].

The existing work demonstrates EfficientNetB0 with PCA and Random Forest for high-accuracy fingerprint classification. Similarly, the proposed study applies EfficientNetB0 and MLP-LSTM for plant disease detection, improving accuracy and efficiency. Both emphasize optimal model selection and feature extraction to enhance real-world applicability [27].

The existing work demonstrates EfficientNetB0's success in brain tumour classification with 99% accuracy, enhancing medical diagnostics. Similarly, the proposed work applies EfficientNetB0 and MLP-LSTM for plant disease detection, achieving high accuracy. Both studies highlight deep learning's role in improving classification accuracy and real-time decision-making [28].

2. LITERATURE SURVEY

The advancements in IoT and machine learning for agriculture mainly deal with environmental monitoring, plant health, and disease prediction. The review of methodologies and results focuses on IoT-based systems, machine learning algorithms, and hybrid techniques for precision agriculture. A Social IoT (SIoT) system integrates smartphone images with garden sensors to track plant health and support disease prediction, highlighting IoT's potential in sustainable agriculture through deep learning [1]. An IoT-based Wireless Sensor Network (WSN) combined with fuzzy logic and machine learning (KNN, MLP, and Random Forest) enables effective pest prediction and environmental data collection for precision agriculture [2]. A flexible substrate IoT sensor tracks the duration of leaf wetness. The light and sensitive sensor proves the potential of IoT in real-time tracking of environmental parameters, which focused on the identification of leaf wetness source [3].

An IoT-based smart irrigation system that uses RF energy harvesting for power supports sustainable, real-time irrigation control, which emphasizes the need for accurate environmental sensing [4]. MLP models predict fungal diseases based on soil and atmospheric sensor data with over 98% accuracy, thus demonstrating the effectiveness of neural networks in classifying plant health issues [5]. Genetic algorithms optimize CNN architectures for agricultural image classification tasks and have improved classification accuracy, thus showing the potential of combining optimization techniques with deep learning [6]. A study on climate effects on plant-pathogen interactions emphasizes the need for accurate monitoring of environmental parameters, which is in line with identifying leaf wetness origins for disease control [7]. Ensemble deep learning with Random Forest achieves high accuracy for flood prediction, validating the robustness of combining deep learning and ensemble techniques for agricultural data [8].

CNN-LSTM and CNN-GRU models perform well on sequential data; therefore, hybrid models can be used for complex predictions in agriculture [9]. The ensemble approaches like bagging, stacking, and boosting, improve the prediction robustness, which can be applied to select Random Forest, Bagged SVM, and XGBoost for the proposed model [10]. LSTM networks process time-series agricultural data with high accuracy, making the inclusion of LSTM layers within MLP models appropriate to extract time-related patterns [11]. Hybrid models

outperform traditional models in predicting leaf wetness duration, aiding disease management systems [13]. Nonlinear regression models, such as XGBoost, are well suited for capturing complex relationships between relative humidity and leaf wetness duration [14]. Favorable temperatures combined with prolonged wetness duration increase disease risk, highlighting the importance of precise wetness source detection for disease control strategies [15].

Incorporating temperature data in leaf wetness models optimizes predictions and improves disease prediction accuracy [16]. CNN architectures like DenseNet-121 and ResNet-50 achieve high accuracy in plant disease detection, underscoring deep learning's capability for complex classification tasks. Their study justifies the use of MLP-LSTM in our model for better leaf wetness classification [17]. CNNs for Plant Disease Diagnosis: CNNs effectively diagnose a wide range of plant diseases, validating deep learning's application for precise classification [18]. Ensemble models combining decision trees, random forests, and naive Bayes improve accuracy and stability in environmental forecasting [19]. A voting ensemble achieves 100% accuracy in predictive modeling for improved generalization for robust predictions [20].

The traditional CNN-based plant disease detection with a hybrid deep learning approach that integrates sensor-based classification of leaf wetness sources. The existing work uses CNNs for identifying diseases in tomato and chili plants with an accuracy of 85.03% but lacks environmental context, such as moisture conditions influencing disease spread. In contrast, the proposed research employs an MLP-LSTM model to classify wetness sources (dew, rainfall, irrigation) with 99.71% accuracy and an EfficientNetB0 model for disease detection, achieving 95.16% accuracy. By linking prolonged leaf wetness from overhead irrigation to the spread of Cercospora and wilting disease, the study provides actionable insights for precision irrigation management. The hybrid approach significantly enhances disease forecasting accuracy, offering a more effective decision support system for farmers to optimize water usage and reduce disease prevalence [25].

The existing work highlights the effectiveness of CNNs in detecting diseases across various vegetable crops but also identifies challenges such as reliance on public datasets, environmental variability, and the need for improved generalization across different regions. It underscores the necessity for advanced feature extraction,

integration of new techniques like ensemble learning, and the potential role of IoT in disease detection. Building on these insights, the proposed work advances plant disease detection by incorporating a hybrid deep-learning model that combines numerical sensor-based classification with image-based disease recognition.

Unlike traditional CNN-based methods, this approach integrates an MLP-LSTM model to classify leaf wetness sources with 99.71% accuracy and an EfficientNetB0 model for disease detection with 95.16% accuracy. By linking prolonged leaf wetness from overhead irrigation to the spread of Cercospora and Wilt disease, the proposed framework addresses a key limitation in existing research—environmental variability—while offering actionable insights for precision irrigation management. This study aligns with the identified need for more resilient and adaptive models, improving real-world applicability and contributing to sustainable agricultural practices [26].

The existing work demonstrates the effectiveness of deep learning models, particularly EfficientNetB0 combined with PCA and Random Forest, for high-accuracy classification in biometric fingerprint identification. It highlights how optimizing feature extraction and classification techniques significantly improves accuracy, speed, and efficiency. Similarly, the proposed work leverages EfficientNetB0 for disease classification in chili plants, achieving superior accuracy compared to conventional CNN models.

Just as the existing study refines classification performance through dimensionality reduction, the proposed work enhances disease detection by integrating an MLP-LSTM model for wetness source identification, achieving 99.71% accuracy. Both studies emphasize the importance of selecting optimal models and pre-processing techniques to improve classification accuracy and real-world applicability. By applying deep learning advancements to plant disease management, the proposed work extends the principles of efficient feature extraction and classification, providing actionable insights for optimizing irrigation practices and mitigating disease spread [27].

The existing work demonstrates the effectiveness of EfficientNetB0 in accurately classifying brain tumours using MRI images, achieving 99% accuracy and significantly aiding medical diagnostics. It highlights the advantages of fine-tuning deep learning models to improve classification accuracy and support real-time

decision-making in critical applications. Similarly, the proposed work applies EfficientNetB0 for plant disease classification, achieving 95.16% accuracy, while also integrating an MLP-LSTM model for sensor-based classification of leaf wetness sources with 99.71% accuracy.

Just as early detection of brain tumours enhances patient outcomes, early disease identification in plants can improve agricultural productivity by reducing crop loss. Both studies emphasize the role of advanced deep learning techniques in refining classification accuracy and practical decision-making. By leveraging EfficientNetB0's potential in agriculture, the proposed work extends its proven success in medical imaging to precision irrigation and plant disease management, offering actionable insights for farmers to mitigate disease risks and optimize resource usage [28].

3. SOFTWARE AND HARDWARE SPECIFICATIONS

3.1. Hardware Requirements:

- Leaf Wetness Sensor or Leaf Moisture Sensor(LWS)
- Intel i7 processor, 8GB RAM

3.2. Software Requirements:

- Python (Recommended version: 3.8 or later)
- Jupyter Notebook (Optional, for interactive execution)
- PyTorch (torch, torch.nn, torch.optim)
- Torchvision (torchvision.transforms, torchvision.datasets, torchvision.models)
- TensorFlow & Keras (tensorflow, tensorflow.keras, tensorflow.keras.layers, tensorflow.keras.models, tensorflow.keras.optimizers, tensorflow.keras.callbacks)
- XGBoost (xgboost)
- Scikit-learn (sklearn.metrics, sklearn.preprocessing, sklearn.ensemble, sklearn.model_selection)
- NumPy (numpy)
- Pickle (pickle)
- Matplotlib (matplotlib.pyplot)
- Seaborn (seaborn)
- Time (time)

4. PROBLEM STATEMENT

1. Traditional machine learning models struggle to accurately classify leaf wetness sources due to their isolated nature. This research explores the hybridization of MLP and LSTM to improve prediction accuracy in detecting ambient parameters like leaf wetness sources.

2. Develop an EfficientNetB0-based deep learning model to accurately classify chili plant leaves into three categories—Cercospora, Healthy, and Wilt Disease—while minimizing misclassification errors occurring in traditional Convolutional Neural Network

4.1. Objectives:

1. To evaluate the effectiveness of the MLP-LSTM hybrid model in enhancing accuracy compared to isolated models, demonstrating its applicability in agriculture and other domains requiring machine learning-based classification.

2. To enhance plant disease detection by improving classification accuracy, recall, and specificity for each class, ensuring reliable identification of chili leaf health conditions.

5. DESIGNING

5.1. System Architecture:

The hybridization technique is employed in our study using the Multilayer Perceptron (MLP) and Long Short-Term Memory (LSTM) model. The model architecture is shown in the figure. 1. The architecture of the MLP-LSTM model is visualized in the figure 1. The input features leaf_temperature and leaf_moisture are given as input to the MLP-LSTM model. X1 and X2 represent the input features given as input to the hybridized model MLP-LSTM. The count of hidden layers and the count of neurons in each layer are mentioned in the figure. 1. The dotted lines in the MLP-LSTM architecture represent a dropout layer. Reshape layer emits the shape expected by the LSTM

The patterns or sequential dependencies are captured by the Long Short Term Memory. The activation function used by the hidden layer is the Rectified Linear Unit and the output layer is softmax. The output layer emits the probabilities of three classes as output. The existing work used Multilayer Perceptron for numerical data to classify the source of leaf wetness into three classes they are dew, rainfall, and irrigation [21]. The proposed work added Long Short Term Memory as the final layer because LSTM is good at handling time series.

The data set used in this work is collected over a period of 5 months for every 20 minutes, which tells this data set is time series data. So, LSTM is added to our work to improve the accuracy of the model from 92. 86 percent to 99.731 percent. CNN-GRU is used for image-based datasets, here we are handling numerical data.

MLP Feature Learning:

The MLP with weights W and biases b can be expressed in Equation 1

$$h = f(WX + b)) \qquad (1)$$

Where:

- Xis the input feature vector,
- W and b are learnable parameters,
- f(·) is the activation function (e.g., ReLU, Sigmoid).

LSTM Sequential Processing:

The LSTM processes the output h using Equations 2, 3, 4, and 5:

$$ft = \sigma(Wfht + UfXt + bf) \quad (2)$$

$$it = \sigma(Wiht + UiXt + bi) \quad (3)$$

$$ot = \sigma(Woht + UoXt + bo) \quad (4)$$

$$ct = ft \odot ct - 1 + it \odot \tanh(Wcht + UcXt + bc) \quad (5)$$

The processing of the data takes place at the hidden layers. The final layer which gives the output is called the output layer. Each neuron in the MLP has an activation function except the neurons in the input layer. Long Short-Term Memory is one of the types of Recurrent Neural Network. It is good at handling time series data and avoiding vanishing gradient problems. The dataset used in this study is collected over a period of 5 months that is from May to September. As LSTM is good at handling time series data, adding LSTM to the MLP enhances the performance of the model. L2 regularization and Dropout are two effective techniques to avoid overfitting, and mostly used when the dataset is imbalanced or of limited size. These techniques ensure not only good performance on the training dataset but also on validation and testing datasets that are unseen. Dropout prevents the model from memorizing the patterns while training the model by introducing randomness during the training phase. This nature of the dropout technique helps to overcome overfitting that occurs due to a high number of neurons per layer, and number of layers.

The 'Dropout' regularization technique is implemented by adding dropout layers after each hidden layer to avoid overfitting. It works by deactivating a random fraction of neurons. The output obtained from the second dense layer is given as input to the LSTM model after reshaping. The reshaped dimension is given as input to the LSTM layer. The L2 regularization technique with 0.01 regularization strength is applied in the LSTM layer to avoid overfitting by penalizing the large weights. The output obtained from the LSTM layer is given as input to the output layer containing 3 neurons, and a softmax activation function, used for multiclass classification. The result given by the output layer is the probabilities of 3 classes.

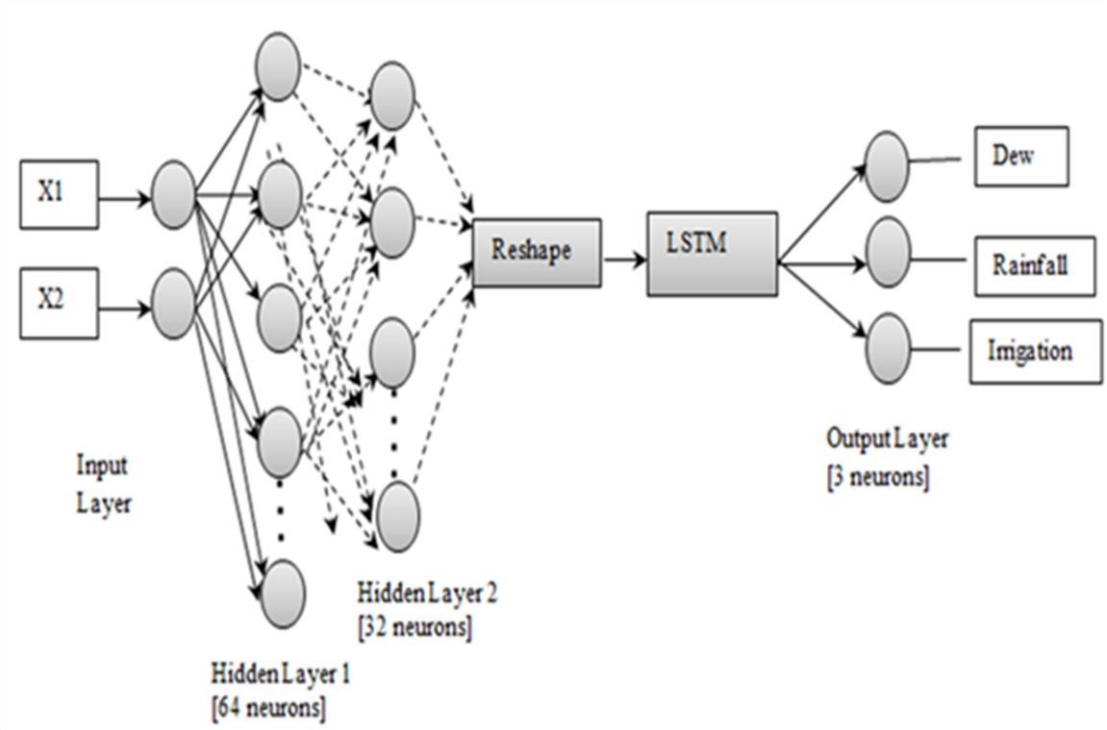

Figure 1. MLP-LSTM Architecture

Figure 2 shows the convolutional neural network (CNN) model designed for multi-class classification with three output classes. It consists of three convolutional layers, each followed by a ReLU activation function and a max-pooling operation. The first convolutional layer takes an input with three channels (RGB images) and applies 32 filters of size 3×3 with padding to maintain the input dimensions. The second convolutional layer expands the feature maps to 64 channels, and the third layer further increases them to 128 channels. Each convolution is followed by a 2×2 max pooling operation, reducing the spatial dimensions by half at each stage.

After the convolutional and pooling operations, the output is flattened and passed through two fully connected (FC) layers. The first FC layer consists of 256 neurons with a ReLU activation function, followed by a dropout layer (p=0.5) to reduce overfitting. The final fully connected layer has three output neurons, corresponding to the three target classes, and does not use an activation function (assuming softmax is applied externally in classification tasks).

The model is trained using categorical cross-entropy loss (nn.CrossEntropyLoss), suitable for multi-class classification problems. The optimizer used is Adam (Adaptive Moment Estimation) with a learning rate of 0.001, which

efficiently updates model parameters during training. The model is implemented using PyTorch and runs on the specified device (CPU or GPU).

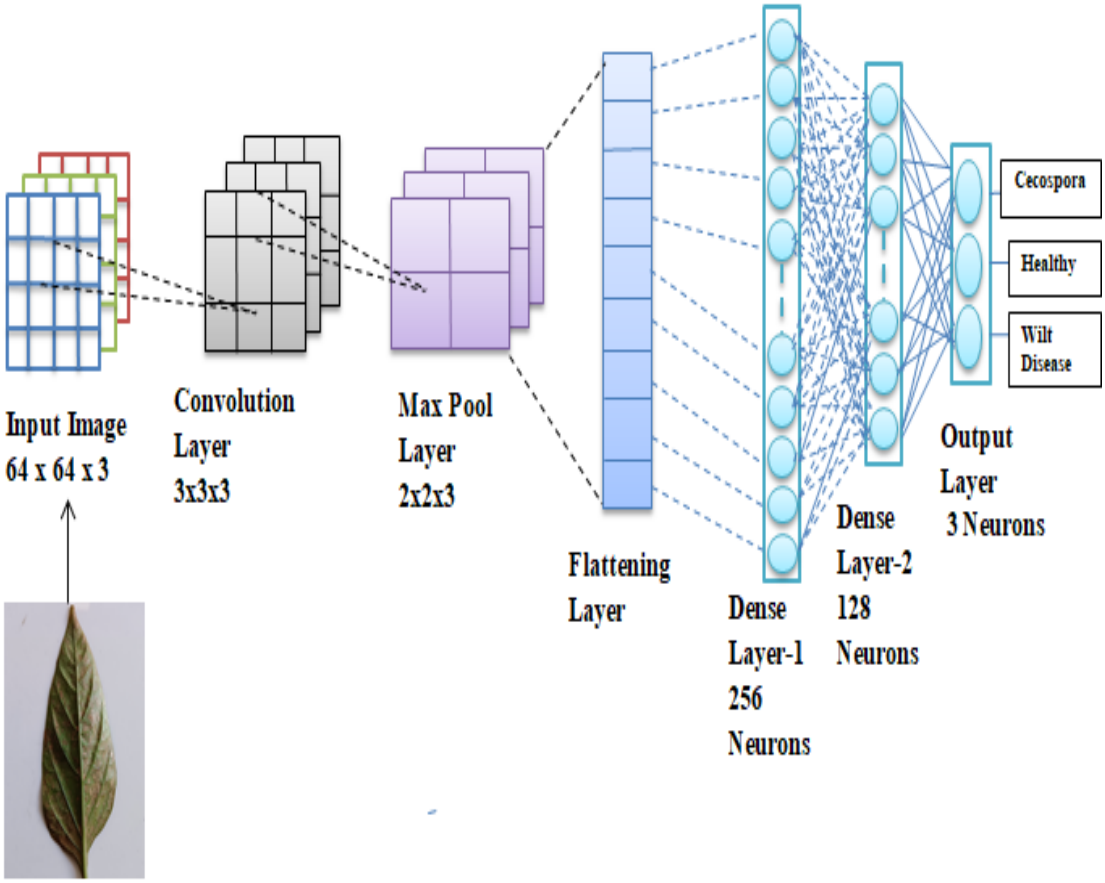

Figure 2. Convolutional Neural Network

Figure 3 shows the EfficientNetB0 model, a deep convolutional neural network that leverages a compound scaling approach to optimize accuracy and efficiency. It is trained on a large dataset (such as ImageNet) and has been fine-tuned for a specific classification task. EfficientNetB0 consists of multiple convolutional layers with mobile inverted bottleneck (MBConv) blocks, which improve computational efficiency while maintaining high performance. These blocks use depthwise separable convolutions and squeeze-and-excitation mechanisms to enhance feature extraction.

In the given implementation, the pre-trained EfficientNetB0 model is modified to adapt to a new classification task. The original classifier, which was designed for ImageNet's 1000 classes, is replaced with a custom fully connected layer structure. The last layer of the classifier is modified to include a linear layer with 256 neurons, followed by a ReLU activation function and a dropout layer (dropout probability of

0.4) to prevent overfitting. The final output layer is a fully connected layer with several neurons equal to the number of classes in the dataset, making it suitable for multi-class classification.

The model is trained using the cross-entropy loss function (nn.CrossEntropyLoss), which is ideal for classification tasks, and the Adam optimizer (learning rate = 0.001) to efficiently update the model's weights. This fine-tuning approach retains the powerful feature extraction capabilities of the pre-trained EfficientNetB0 while adapting it to a specific classification problem. The model is deployed to the specified computing device (CPU or GPU) to leverage hardware acceleration for faster training and inference.

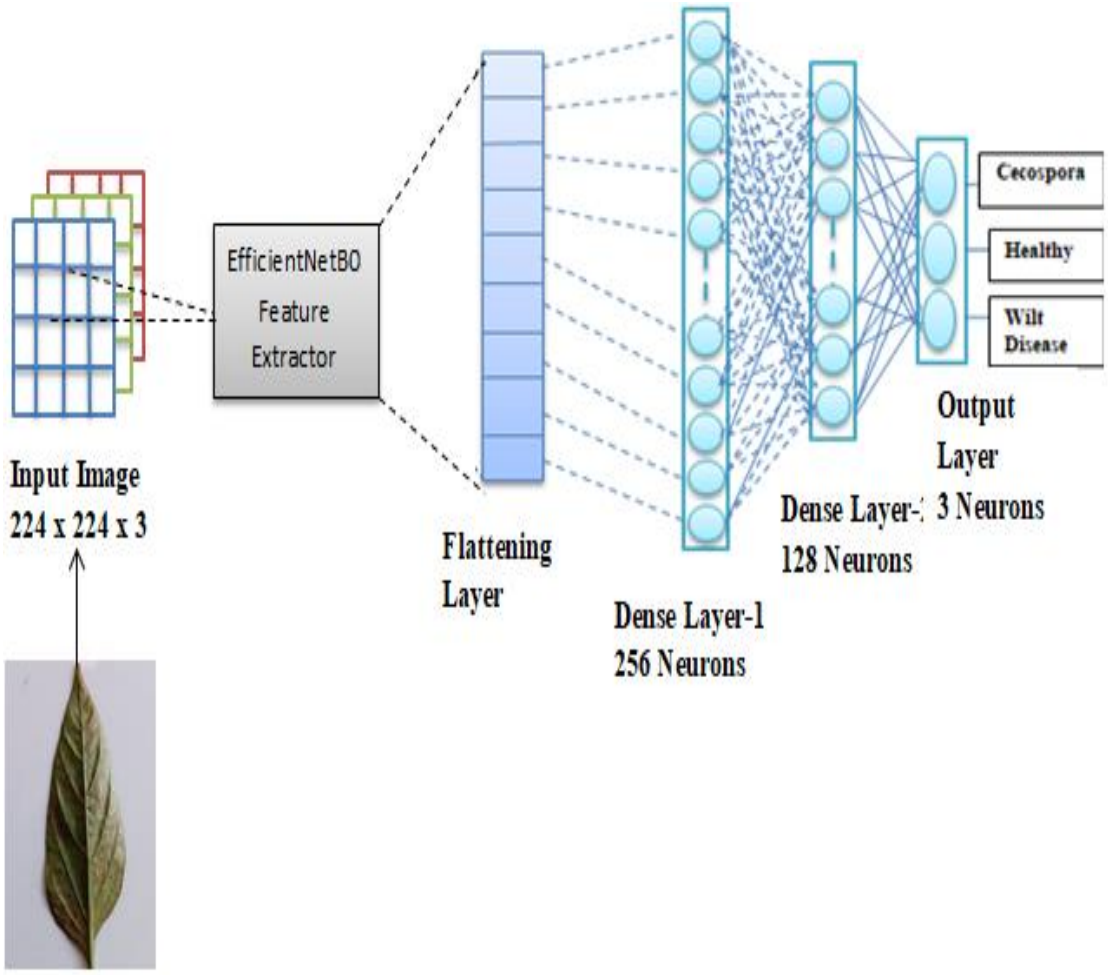

Figure 3. EfficientNetB0 Model Architecture

5.2. Methodology:

Figure 4 shows the sequence of steps involved in the methodology.

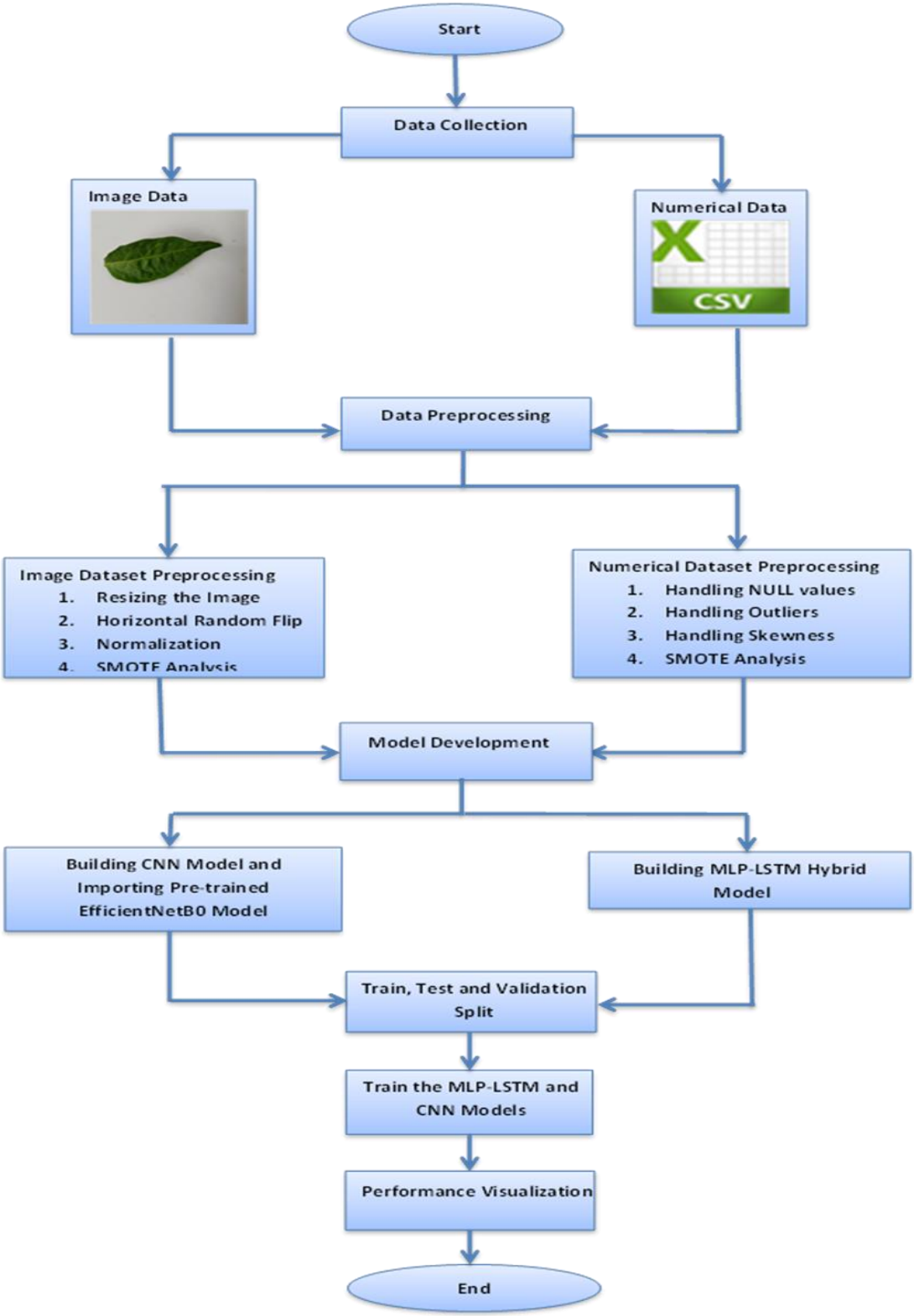

Figure 4. Flowchart of Methodology

5.2.1. Data Collection:

Numerical Data Collection:

The chili crop is chosen for this study to detect leaf wetness sensors using hybridization. The dataset of size 6919 data points is collected through sensors established in the field. The dataset has the column names: Date, Time, Device Name, bat, leaf_moisture, leaf_temperature, rssi, snr, and spreading_factor. The parameters other than leaf moisture and leaf temperature monitor the health status of the leaf wetness sensor, so the parameters that are not useful to detect leaf wetness are removed to avoid unnecessary data for further analysis.

A column with the name ‘class’ is added to the dataset to classify the source of leaf wetness. To classify the leaf wetness source of chili plants using the parameters leaf_moisture and leaf_temperature, the conditions used are, If leaf_temperature < 25 and 60 <= leaf_moisture <= 75, then the source of leaf wetness is ‘dew.’, If 20 <= leaf_temperature <= 30 and leaf_moisture > 80, then the source of leaf wetness is ‘rainfall.’, If 25 <= leaf_temperature <= 35 and 50 <= leaf_moisture <= 70, the source of leaf wetness is ‘irrigation.’ The count of NULL values in each column is found to handle them in the data preprocessing step.

The data collected over a period of 5 months for every 20 minutes duration is time series data, LSTM is good at handling time series data, so it is included in the proposed work to improve the model's performance.

Image Dataset Collection:

The image dataset is created by collecting the images of 618 different leaves of chili plants. The images in the dataset are classified into three different classes 'Healthy', 'Cercospora', and 'Wilt_Disease. 199 images belong to the class 'Cercospora', 209 images belong to the class 'Healthy', and 210 images belong to the class 'Wilt_Disease'. Figure 5 shows the images of chili leaves of the three different classes Cercospora, Healthy, and Wilt Disease.

Cercospora Healthy Wilt Disease

Figure 5. Different Classes of Images in the Dataset

5.2.2. Data Pre-processing:

Numerical Dataset Preprocessing:

Handling NULL Values:

The size of the dataset may decrease after data preprocessing depending on the presence of missing values and unnecessary data. In this study, after data preprocessing, the size of the dataset decreased from 6919 to 3304 data points. Due to the absence of NULL values in the dataset, the step to handle NULL values is skipped. The handling of NULL values in the dataset depends upon the type of data in each column. If the data type is categorical, the NULL values are handled using the statistical measure called MODE to replace the NULL values with the MODE value of the data in that respective column. This is called MODE imputation.

If the data type is numerical, a statistical measure called MEAN or MEDIAN is used to replace the NULL values with the mean or median value of the data in that respective column. This is called MEAN imputation or MEDIAN imputation.

Handling Outliers:

The data points that do not lie within the range of the upper and lower boundary of the dataset in each column are called outliers. The performance of a model is affected by the outliers in a dataset. So, outliers in each column are handled using the interquartile range (IQR). In Figure 6, the black spots in the form of small circles are outliers present in the leaf_moisture parameter. These data points are clipped within the range of upper bound and lower bound values of the leaf_moisture

column. The IQR is calculated as the difference between Q1 and Q3. The lower and upper bounds are defined as 1.5 times the IQR below Q1 and above Q3 respectively.

All data points that have values outside of this range are clipped to the nearest boundary with the `clip` function, preventing extreme values from impacting the performance of the model. A box plot is created after the removal of outliers to visually exhibit the impact of this procedure. The outliers present in the leaf_moisture column are represented in the form of circles on one another as in the boxplot shown in figure 6. Figure 7 shows the boxplot of the leaf_moisture parameter after removing outliers, that is, after handling outliers. The outliers are handled using the formulae mentioned for calculating the quartile Range (IQR). The distribution of each column is visualized to know if data is distributed normally over the mean of the data of each column.

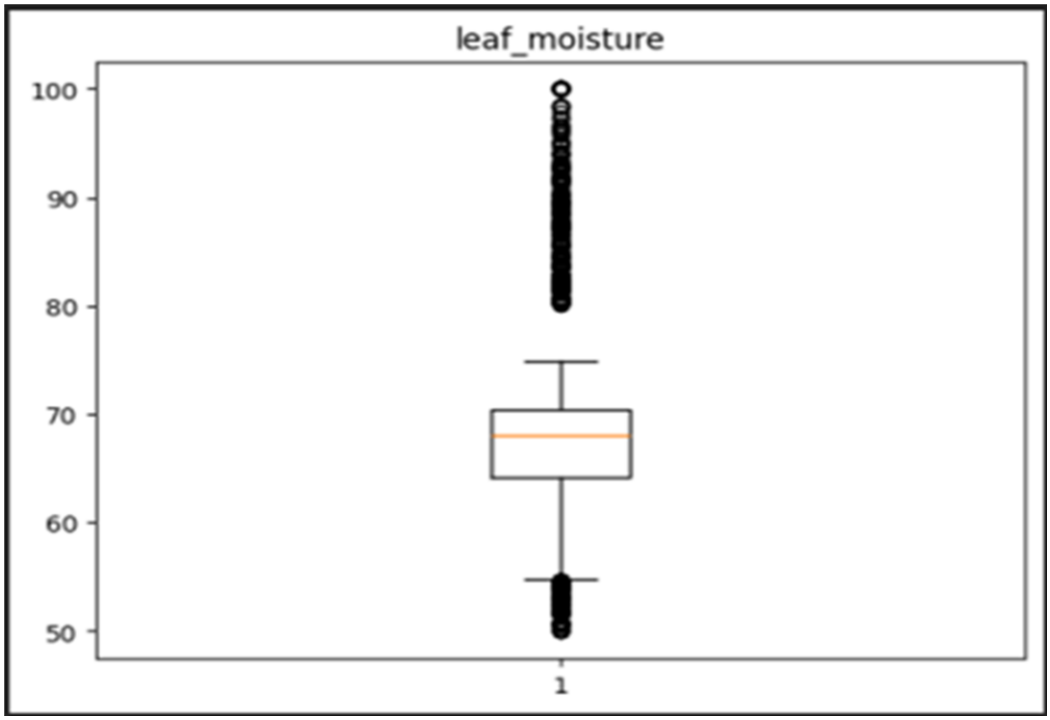

Figure 6. Leaf_Moisture Before Handling Outliers

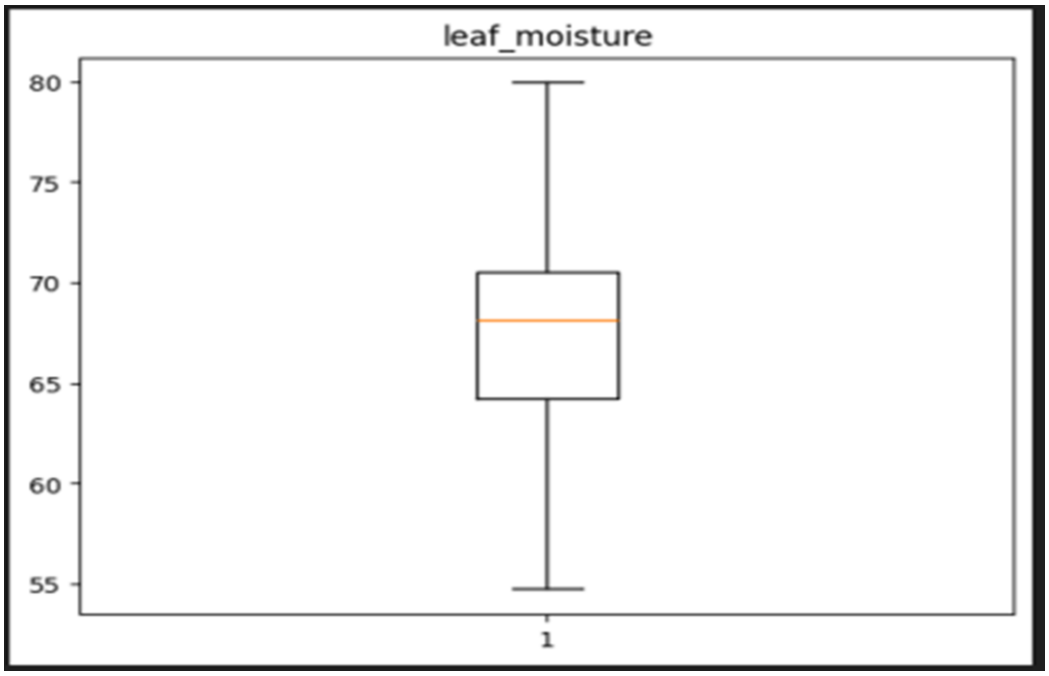

Figure 7. Leaf_Moisture After Handling Outliers.

Handling Skewness:

First, continuous numerical columns are identified to analyze their skewness using histograms and skewness values. If the data in any column of the dataset is not normally distributed, then the data of that column needs to be handled to make the data normally distributed; this is possible through the skewness metric. The PowerTransformer (Yeo-Johnson method) is applied to reduce skewness and standardizing the data such that the transformed features take on a more Gaussian-like distribution. The transformation is applied only to those columns with an absolute skewness greater than 1, as these columns are found to be significantly skewed.

This approach improves the performance of the model as it reduces the bias that comes with non-normally distributed data. The distributions are then re-plotted after transformation to visualize the skewness correction effect. Figure 8 shows the positive skewness in the leaf_moisture parameter. The skewness values in the range of -0.5 and 0.5 show the distribution is normal or symmetric. Here the skewness is 1.469, which shows data in that column is highly skewed. Boxcox transformation is applied to handle the skewness. Boxcox is applied for strictly positive skewed data.

Figure 9 is obtained after applying boxcox transformation on the leaf_moisture parameter to get the normal distribution.

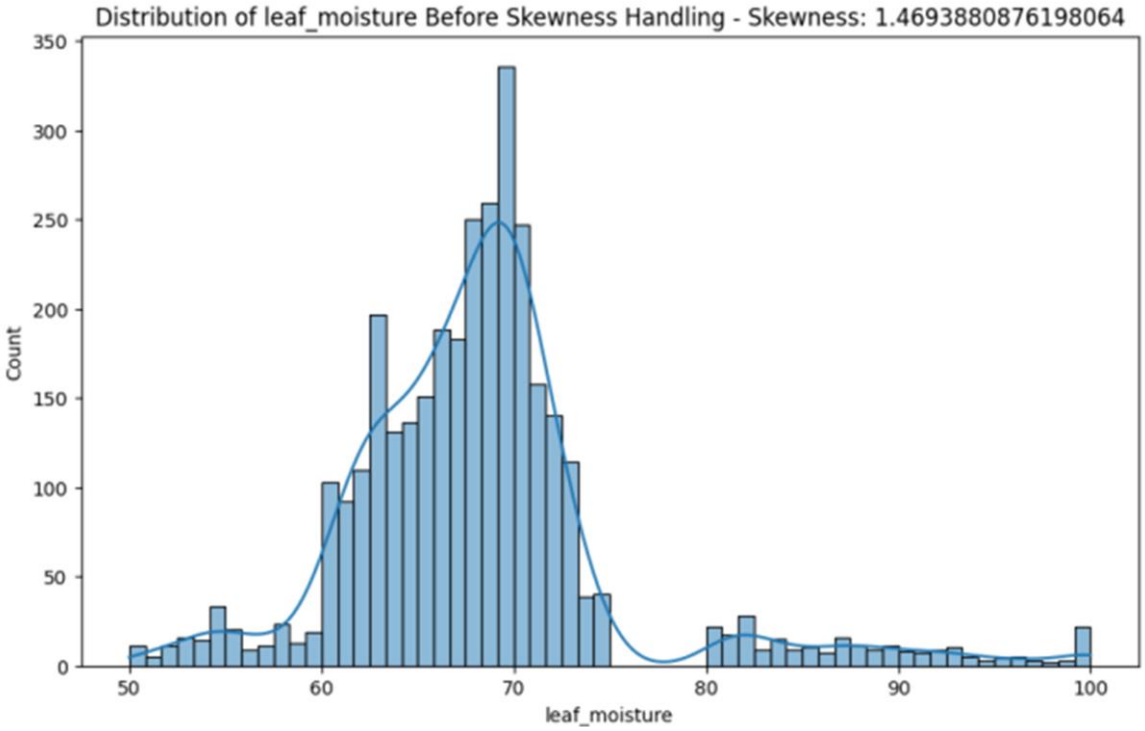

Figure 8. Distribution of Leaf_moisture column Before Handling Skewness

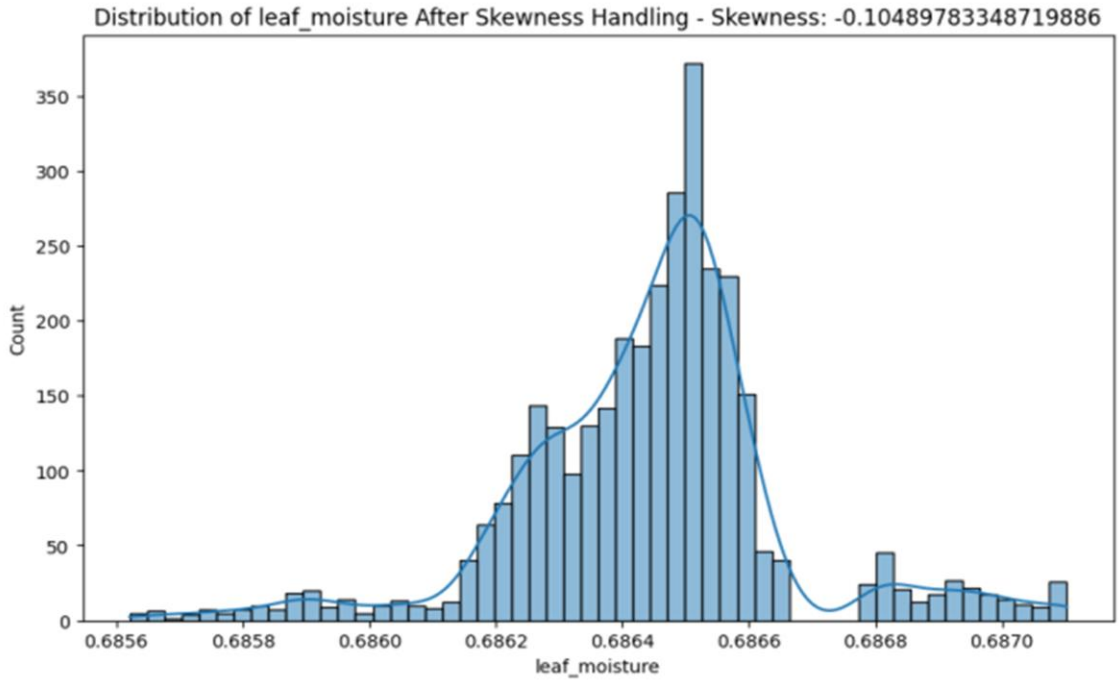

Figure 9. Distribution of Leaf_Moisture Column After Handling Skewness

This plot has a skewness value of -0.104, which lies in the range of normal distribution, which is -0.5 to 0.5.is used, it is represented as SMOTE. An imbalance in the dataset is caused when the data points belonging to the respective class labels are

not nearly equal. To convert the imbalanced dataset to a balanced one, SMOTE analysis is performed by generating synthetic samples. These synthetic samples can be generated using three ways,

- Standard SMOTE: generates synthetic samples to make some data points of majority and minority classes equal.
- Minority SMOTE: generates synthetic samples only to balance underrepresented classes.
- Custom/Dictionary-based SMOTE: generates synthetic samples based on a user-specified sample size of each class.

To avoid overfitting in a model, Standard SMOTE is not recommended. In this study, custom-based SMOTE is used to balance the dataset to avoid overfitting of machine learning models. The distribution of data points to each class before SMOTE analysis is represented in figure 10, 1888 data points belong to the class dew, 1171 data points belong to the class irrigation, and 245 data points belong to the class rainfall before SMOTE analysis. The distribution of data points to each class after SMOTE analysis is shown in figure 11, 1307 data points belong to class rainfall, 1267 data points belong to class irrigation, and 1146 data points belong to class dew.

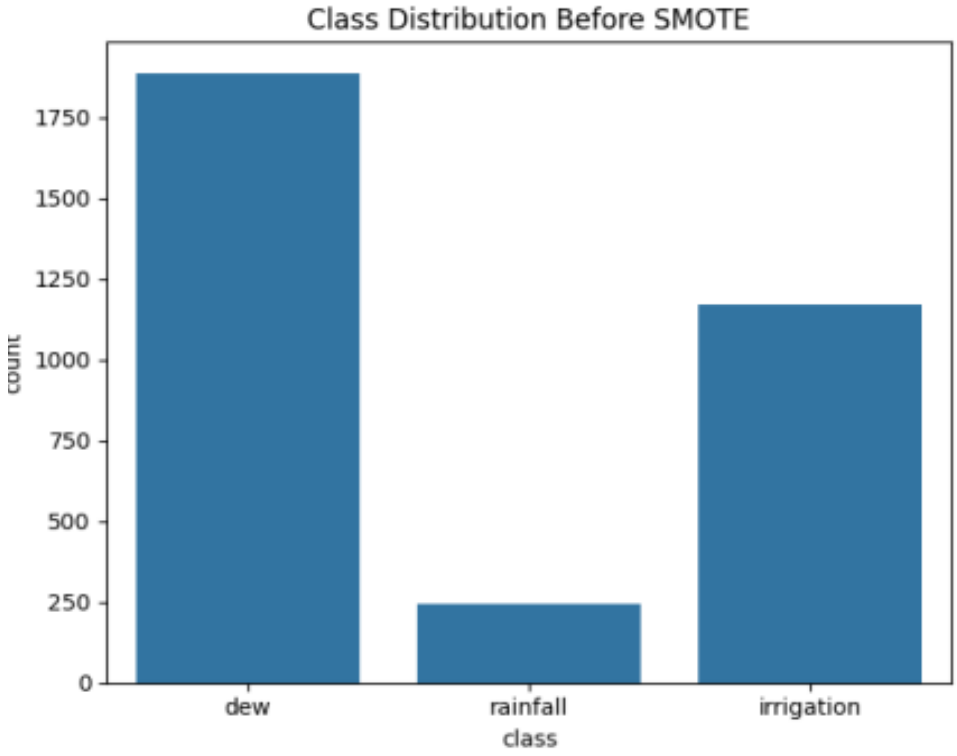

Figure 10. Class Distribution Before Smote Analysis

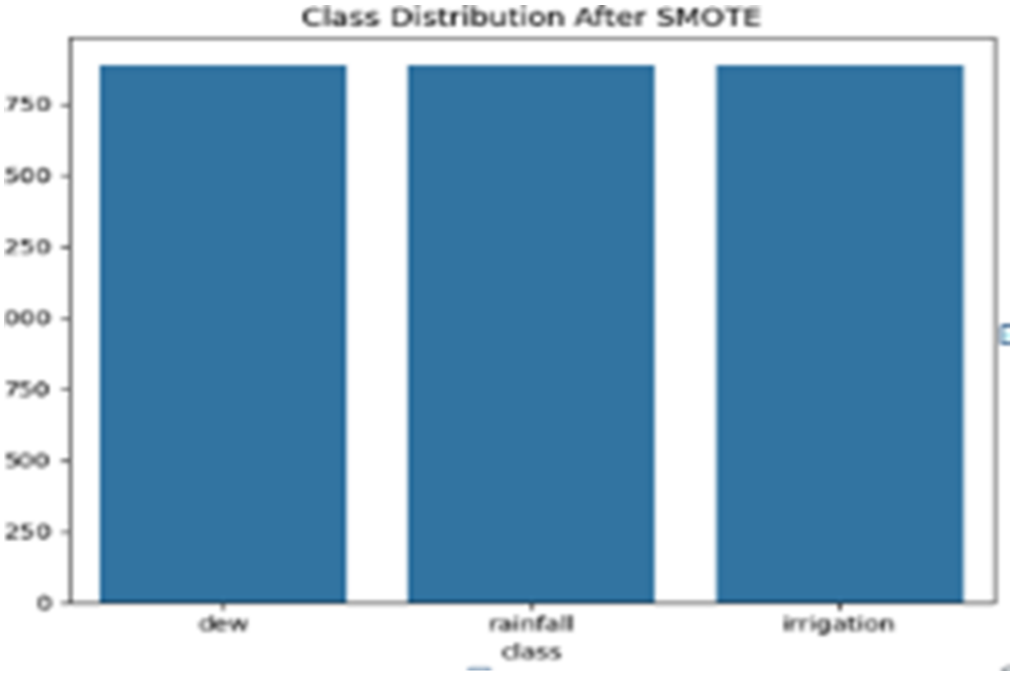

Figure 11. Class Distribution After Smote Analysis

Image Dataset Preprocessing:

To artificially expand the dataset and introduce variations that improve model robustness, data augmentation techniques were applied exclusively to the training images. Augmentation helps to prevent overfitting by exposing the model to different transformations of the same image, ensuring better generalization to unseen data. Unlike the training set, no data augmentation was applied to the validation and test datasets to maintain consistency and ensure that the model is evaluated on unaltered images.

Resizing:

Since different deep learning architectures have varying input size requirements, all images were resized to match the respective model's input dimensions. For the traditional CNN model, images were resized to (64×64) pixels, while for the pre-trained EfficientNetB0 model, images were resized to (224×224) pixels to align with its expected input dimensions. Resizing ensures uniformity across the dataset, reducing computational complexity while preserving essential spatial features. As in the training set, all validation and test images were resized to either (64×64) pixels for CNN or (224×224) pixels for EfficientNetB0. This ensures that the evaluation images match the model's input requirements without introducing distortions.

Random Horizontal Flip:

To introduce additional variability in the dataset, a random horizontal flip was applied to training images with a probability of 0.5. This augmentation prevents the model from learning biases related to a specific orientation of the objects within images. By flipping images horizontally, the model becomes invariant to such transformations, improving its ability to generalize to real-world variations in leaf positioning.

Normalization:

Normalization is a crucial step in stabilizing neural network training. The pixel values of the images were normalized using a mean of [0.5, 0.5, 0.5] and a standard deviation of [0.5, 0.5, 0.5] for each color channel (Red, Green, and Blue). This transformation scales pixel intensities from the range [0, 255] to [-1, 1], ensuring that input values are centered around zero. Normalization reduces internal covariate shifts during training, leading to faster convergence and improved model stability. The same normalization strategy applied to the training set was used for the validation and test images. Normalizing the test data ensures that the model receives inputs with the same statistical distribution as those seen during training, preventing discrepancies between training and inference phases.

Through the application of these preprocessing measures, the data was standardized in order to increase model training stability, improve generalization, and maximize classification accuracy. Data augmentation, which was specifically applied on training images, was important to enhance robustness against leaf orientation and illumination variability, thus optimizing the overall performance of both the CNN and EfficientNetB0 models.

Synthetic Minority Oversampling Technique Analysis(SMOTE):

The image dataset used in our model is balanced, it has the following number of images belonging to each class, 199 images belong to the class ‘Cercospora’, 209 images belong to the class ‘Healthy’, and 210 images belong to the class ‘Wilt_Disease’

5. 2. 3. Model Development

5.2.3.1. MLP-LSTM Model Development:

The MLP-LSTM model is designed for multi-class classification, integrating Multi-Layer Perceptron (MLP) and Long Short-Term Memory (LSTM) networks to capture both spatial and sequential dependencies in the dataset. The input features first pass through fully connected layers with ReLU activation and L2 regularization ($\lambda = 0.01$) to prevent overfitting. A dropout layer (rate = 0.3) enhances generalization, followed by another dense layer with 32 neurons, which refines feature extraction. The output is reshaped into a (1, 32) sequence, preparing it for the LSTM layer, which consists of 16 units with L2 regularization to reduce overfitting. The final dense output layer has 3 neurons with a softmax activation function, enabling multi-class classification.

The model is compiled with the Adam optimizer, categorical cross-entropy loss, and accuracy as the evaluation metric. It is trained for 200 epochs with a batch size of 32, using early stopping to avoid overfitting. Performance evaluation includes accuracy, classification reports, and confusion matrices, while training history is visualized through accuracy and loss plots. The trained model is saved for future use, and its structure is verified by reloading and reviewing its summary. By combining MLP for feature extraction and LSTM for sequence modeling, this hybrid architecture is well-suited for structured or time-dependent classification tasks.

5.2.3.2. Convolutional Neural Network Model Development:

The Convolutional Neural Network (CNN) model for chili leaf disease classification is designed to categorize leaf images into three classes: Healthy, Cercospora, and Wilt Disease. The model follows a hierarchical feature extraction approach, utilizing multiple layers to analyze and learn patterns from input images.

The feature extraction process begins with three convolutional layers, each equipped with 3×3 filters and the ReLU activation function. The first convolutional layer captures low-level features such as edges and textures, while the second layer extracts mid-level features like contours and shapes. The third convolutional layer focuses on identifying high-level disease-specific features, including leaf spots and structural abnormalities. Each convolutional layer is followed by a 2×2 max-pooling layer,

which helps in reducing spatial dimensions while preserving the most important features for classification.

Once the feature maps are extracted, a Flatten layer is applied to convert the multidimensional feature maps into a 1D feature vector, preparing the data for classification. This vector is then processed by a fully connected layer with 256 neurons and ReLU activation, which helps in learning complex patterns within the extracted features. To prevent overfitting, a dropout layer with a 50% probability is included, ensuring that the model generalizes well to unseen data.

Finally, the output layer consists of 3 neurons, each representing one of the target classes (Healthy, Cercospora, Wilt Disease). The softmax activation function is used to assign probabilities to each class, enabling multi-class classification. This structured CNN architecture ensures that the model effectively learns hierarchical representations of chili leaf images, optimizing classification performance by extracting and analyzing relevant features at multiple levels.

5.2.3.3. EfficientNetB0 Model Develpoment:

The EfficientNetB0-based model for chili leaf disease classification follows a structured architecture, beginning with an input layer that processes images of size 224×224×3 (height, width, and three color channels). This input is passed through EfficientNetB0's feature extractor, a pre-trained convolutional network, which efficiently extracts meaningful features from the images. The feature extraction process starts with a Conv Stem, which consists of a 3×3 convolution, batch normalization, and SiLU activation to enhance learning efficiency.

The backbone of the model includes seven Mobile Inverted Bottleneck Convolution (MBConv) blocks, utilizing depthwise separable convolutions to reduce computational complexity while maintaining high performance. Finally, a global average pooling (GAP) layer aggregates the extracted features, reducing dimensionality and preparing them for classification. To adapt EfficientNetB0 for the specific task, the default classifier is modified with fully connected layers. The extracted feature maps are first flattened into a 1D vector, followed by a dense layer with 256 neurons and ReLU activation to learn complex patterns.

A dropout layer with a 40% dropout probability is incorporated to prevent overfitting. The final output layer applies a softmax activation function, where the

number of neurons corresponds to the number of classes (Healthy, Cercospora, Wilt Disease), generating probabilistic predictions for classification.

5.2.4. Train, Test, and Validation Split:

5.2.4.1. Numerical Dataset Split:

Split the dataset in the ratio 80:10:10, resultant training, validation, and testing datasets were saved in the form of pickle files, which were used by the developed model for learning, validation, and evaluation. The pickle files thus obtained were given as input to the MLP-LSTM (Multilayer Perceptron-Long Short Term) for training, validating, and evaluating the model.

5.2.4.2. Image Dataset Split:

The image dataset utilized for training and evaluating our model is balanced, ensuring that each class is well-represented to prevent bias in classification. The dataset comprises a total of 618 images, evenly distributed across the three categories. Specifically, there are 199 images labeled as 'Cercospora', 209 images classified as 'Healthy', and 210 images belonging to the 'Wilt Disease' category. This balanced distribution helps the model learn the distinguishing features of each class effectively, reducing the likelihood of class imbalance issues and improving overall classification accuracy.

5.2.5. Train the MLP-LSTM and CNN Models:

5.2.5.1 Training the MLP-LSTM Model:

The MLP-LSTM model is trained on a balanced dataset, where input features (X_train_scaled) and target labels (y_train_encoded) undergo preprocessing before training. The categorical labels (y_train, y_val, y_test) are encoded using LabelEncoder and then transformed into one-hot encoded vectors to support multi-class classification. This ensures the model effectively learns from structured data while maintaining class balance, enabling accurate classification of leaf wetness sources. The model architecture consists of an MLP component followed by an LSTM layer, allowing it to capture both feature-based and sequential dependencies in the data.

It begins with a dense layer (64 neurons, ReLU activation, L2 regularization), followed by a dropout layer (30% dropout rate) to enhance generalization. Another

dense layer with 32 neurons refines feature extraction before reshaping the output into a (1, 32) sequence format for the LSTM layer (16 units). The final dense output layer with softmax activation classifies samples into one of three categories: dew, rainfall, or irrigation—identifying the primary source of leaf wetness. The model is compiled using the Adam optimizer and categorical cross-entropy loss, making it suitable for multi-class classification.

The training process evaluates the model's performance using accuracy scores, classification reports, and confusion matrices, while training and validation accuracy/loss plots help visualize learning trends. Once trained, the model is saved for future use and can be reloaded using load_model(), eliminating the need for retraining. By integrating MLP for feature extraction and LSTM for sequential learning, the model effectively captures spatial and temporal dependencies, improving the accuracy of leaf wetness source classification.

5.2.5.2. Training the CNN Model:

The CNN model is trained using a structured pipeline that includes dataset preprocessing, model architecture definition, training with early stopping, and evaluation using performance metrics. The training dataset consists of chilli leaf images, categorized into three classes: Healthy, Cercospora, and Wilt Disease. During data preprocessing, images are resized to 64×64 pixels to ensure uniform input size. Data augmentation techniques, such as random horizontal flips, are applied to the training set to improve generalization. The images are then converted to PyTorch tensors and normalized to a range of -1 to 1 for efficient training. The dataset is divided into training, validation, and test sets, loaded using PyTorch DataLoaders for efficient batch-wise processing.

The CNN architecture consists of three convolutional layers with 32, 64, and 128 filters, each followed by ReLU activation and max-pooling (2×2) to extract hierarchical spatial features while reducing overfitting. The final feature maps are flattened and passed through a dense layer with 256 neurons, followed by a dropout layer (50%) to enhance generalization. The final softmax output layer generates probability scores for the three classes. The model is trained using the Cross-Entropy Loss function and Adam optimizer (learning rate = 0.001) for 50 epochs with a batch size of 32. Early stopping (patience = 5) is implemented to prevent overfitting, ensuring the best-performing model is saved and reloaded before evaluation.

Post-training, the model is evaluated on the test dataset, where it predicts class probabilities using softmax activation, and the accuracy is assessed by comparing predictions with actual labels. A confusion matrix helps analyze misclassifications, while a classification report provides precision, recall, and F1 scores for each class. Additional performance metrics, such as specificity, Precision-Recall curves, and ROC curves with AUC scores, are computed to visualize model effectiveness. The CNN model's structured approach ensures high accuracy in classifying chili leaf health status while minimizing overfitting through regularization techniques and performance monitoring.

5.2.6. Performance Visualization

5.2.6.1. Classification Report of Hybridized MLP-LSTM Model:

From Table 1, the precision values of each class are close to 1.00 showing that instances of each class are correctly predicted. The recall value of 0.98 for rainfall shows the existence of slight variation in prediction, whereas the other classes dew and irrigation is 1.00. The F1-score value of 0.99 or 1.00 shows an excellent balance between precision and recall. The support value shows that each class has 131 samples, resembling a balanced dataset.

Table 1. Classification Report of Hybridized MLP-LSTM Model

Class	Precision	Recall	F1-score	Support
Rainfall (0)	1.00	0.98	0.99	131
Dew (1)	0.98	1.00	0.99	131
Irrigation (2)	1.00	1.00	1.00	131

5.2.6.2. Sensitivity Values of MLP-LSTM Model:

From table 2, the sensitivity values of the classification model show that it is accurate in identifying each class. The sensitivity of both the Dew and Irrigation classes is 0.9924, which means that the model is accurate in identifying 99.24% of the actual instances of these classes. The Rainfall class has a perfect sensitivity of 1.0000, which means that the model does not have any false negative detections of

Rainfall. These results indicate that the model is very efficient in differentiating between the three categories.

Table 2. Sensitivity Values of MLP-LSTM Model

Class	Sensitivity
Dew	0.9924
Irrigation	0.9924
Rainfall	1.0000

5.2.6.3. Z-score Values of Normalized Test Dataset of MLP-LSTM Model:

From table 3, the Z-score normalized test dataset has two features: Leaf Moisture and Leaf Temperature. The data normalization guarantees that the data has been standardized to have a mean of zero and a standard deviation of one. The values in the table show the variation in the data, for instance, Index 3 has a positive Z-scores (Leaf Moisture: 1.165810, Leaf Temperature: 1.092326) which means that the data points are above the mean, while others, for example, Index 1 (Leaf Moisture: 0.717994, Leaf Temperature: -0.587481) have mixed deviations from the mean. This standardized dataset is beneficial for the model as it allows for easier comparison of data that has been measured on different scales.

Table 3. Z-Score Normalized Test of MLP-LSTM Model Dataset

Index	Leaf_moisture	Leaf_temperature
0	-0.732224	2.419514
1	0.717994	-0.587481
2	-0.574054	1.146497
3	1.165810	1.092326
4	1.715619	-0.745857

5.2.6.4. Classification Report Analysis of CNN Model:

Table 4 shows the classification report of CNN model, it provides precision, recall, and F1 scores for each class. Cercospora: Achieves high precision (0.95), recall (0.95), and F1-score (0.95), indicating that the model classifies this class almost perfectly. Healthy: Shows precision of 0.54 and recall of 0.67, meaning that when the model predicts Healthy, it is correct only 54% of the time and captures 67% of actual Healthy cases. Wilt Disease: Has low precision (0.50) and recall (0.38), meaning the model struggles to correctly identify Wilt Disease instances, often confusing them with Healthy cases. Overall, the weighted average accuracy is 66%, with the model performing best on Cercospora but showing weaknesses in distinguishing between Healthy and Wilt Disease.

Table 4. Classification Report of CNN Model

Class	**Precision**	**Recall**	**F1-score**	**Support**
Cercospora	0.95	0.95	0.95	20
Healthy	0.54	0.67	0.60	21
Wilt Disease	0.50	0.38	0.43	21

5.2.6.5. Specificity Analysis of CNN Model:

Table 5 shows the specificity of the CNN modle, it measures how well the model correctly identifies negative cases for each class. Cercospora Specificity (0.95): The model accurately avoids falsely predicting Cercospora when it is a different class. Healthy Specificity (0.5385): The model does not perform well in avoiding false positives for the Healthy class. Wilt Disease Specificity (0.50): The lowest specificity, meaning many non-Wilt Disease instances are mistakenly classified as Wilt Disease. The lower specificity for Healthy and Wilt Disease suggests that the model frequently misclassifies these classes.

Table 5. Specificity of CNN Model

Class	Speciificity
Cercospora	0.9500
Healthy	0.5385
Wilt Disease	0.5000

5.2.6.6. Classification Report Analysis of EfficientNetB0 Model:

Table 6 shows the classification report of EfficientNetB0 model, it provides key performance metrics: precision, recall, F1-score, and support for each class. For Cercospora, the model achieves 1.00 precision, meaning all instances predicted as Cercospora are indeed Cercospora (no false positives). The recall is 0.95, indicating that 95% of actual Cercospora samples were correctly classified, with a small number misclassified as Wilt Disease. The F1-score (harmonic mean of precision and recall) is 0.97, reflecting strong overall performance. For Healthy, the precision is 0.94, meaning that 94% of samples predicted as Healthy are Healthy.

However, recall is 0.81, showing that only 81% of actual Healthy samples were correctly identified—some were confused with Wilt Disease. The F1-score of 0.87 balances this trade-off, indicating the model performs well but has room for improvement in identifying all Healthy samples. For Wilt Disease, the model achieves 0.80 precision, meaning 80% of predicted Wilt Disease samples were correctly classified, while some were mistakenly classified as Healthy. The recall is 0.95, showing that 95% of actual Wilt Disease samples were correctly detected.

This suggests that the model effectively detects Wilt Disease cases but sometimes incorrectly labels healthy leaves as diseased. The F1-score is 0.87, indicating strong overall performance despite some classification errors. The overall accuracy of the model is 90%, meaning that 90% of all predictions were correct. The macro average (average of all precision, recall, and F1-scores) is 0.91, while the weighted average, which accounts for class imbalances, is 0.90. These values confirm that the model is highly effective.

Table 6. Classification Report of EfficientNetB0 Model

Class	Precision	Recall	F1-score	Support
Cercospora	1.00	0.95	0.97	20
Healthy	0.94	0.81	0.87	21
Wilt Disease	0.80	0.95	0.87	21

5.2.6.7. Specificity Analysis for EfficientNetB0 Model:

Table 7 shows the specificity of EfficientNetB0 model, it measures how well the model correctly identifies negative cases for each class. It is calculated as TN / (TN + FP). For Cercospora, specificity is 1.000, meaning that the model never mistakenly classifies non-Cercospora samples as Cercospora. This indicates a perfect ability to avoid false positives for this class. For Healthy, the specificity is 0.944, meaning that 94.4% of non-Healthy samples were correctly classified as non-Healthy. Some misclassifications occur, particularly with Wilt Disease, which slightly reduces specificity. For Wilt Disease, specificity is 0.800, meaning 80% of non-Wilt Disease samples were correctly classified as either Healthy or Cercospora.

This suggests that Wilt Disease is sometimes confused with Healthy samples, reducing its specificity. The EfficientNetB0 model demonstrates strong classification performance with an overall accuracy of 90%. The Cercospora class is the best-classified category, with perfect precision and specificity. The Healthy class shows high precision but lower recall, indicating some misclassification with Wilt Disease. The Wilt Disease class has high recall but slightly lower precision and specificity, suggesting that it is occasionally misclassified as Healthy. These insights can help in fine-tuning the model to further improve classification accuracy, particularly for Healthy and Wilt Disease samples.

Table 7. Specificity of EfficientNetB0 Model

Class	Speciificity
Cercospora	1.000
Healthy	0.944
Wilt Disease	0.500

6. IMPLEMENTATION

6.1. MLP-LSTM Model Implementation:

The implementation of the MLP-LSTM model begins with loading and preprocessing the dataset. The dataset is stored in pickle files and includes features (X_train, X_val, X_test) and corresponding labels (y_train, y_val, y_test). The labels are then encoded using LabelEncoder, which transforms categorical labels into numerical values. To facilitate multi-class classification, the encoded labels are converted into a one-hot encoded format using TensorFlow's to_categorical function. This ensures that the model outputs probabilities for each class. The model itself is built as a hybrid architecture combining Multi-Layer Perceptron (MLP) and Long Short-Term Memory (LSTM) layers.

The input layer takes the feature set and passes it through two fully connected (dense) layers with ReLU activation functions and L2 regularization to prevent overfitting. Dropout layers are included to enhance generalization. The feature representation is then reshaped into a 3D format to be compatible with the LSTM layer, which processes sequential dependencies in the data. The final output layer applies a softmax activation function, predicting the probability of each class. The model is compiled using the Adam optimizer and trained with categorical cross-entropy loss for multi-class classification.

Training is performed over 200 epochs with validation on a separate dataset, and performance is evaluated using accuracy and classification metrics. Post-training, the model's performance is analyzed through visualization techniques such as accuracy/loss curves and confusion matrices, which help assess how well the model differentiates between the three classes. Finally, the trained model is saved using TensorFlow's save function and reloaded for verification. The reloaded model is checked using its summary, ensuring that its architecture remains intact. This approach integrates both deep learning (MLP-LSTM) and traditional classification performance evaluation, providing a robust solution for classifying the given dataset.

6.2. Implementation of CNN Model:

The Convolutional Neural Network (CNN) model implemented in this code is designed to classify chili plant leaf images into three categories: healthy, Cercospora, and wilt disease. The dataset is loaded using PyTorch's ImageFolder and undergoes

preprocessing with data augmentation techniques such as resizing, normalization, and random horizontal flipping to improve generalization. The CNN architecture consists of three convolutional layers, each followed by ReLU activation and max-pooling layers, which extract meaningful features from the input images.

After feature extraction, the output is flattened and passed through fully connected layers, including a dropout layer to prevent overfitting. The final output layer applies softmax activation to classify the images into one of the three categories. The model is trained using the Adam optimizer and cross-entropy loss, ensuring effective weight updates during backpropagation. The training process follows a structured approach where the model alternates between training and validation phases for multiple epochs. During each epoch, the loss and accuracy are monitored, and an early stopping mechanism is implemented to prevent overfitting by stopping training when the validation loss does not improve for a specified number of iterations.

The trained model is then evaluated on the test set using various performance metrics, including accuracy, confusion matrix, classification report, specificity, precision-recall curves, and ROC-AUC scores. These evaluations provide insights into the model's effectiveness in distinguishing between healthy and diseased leaves. The final results show that the pre-trained EfficientNetB0 model significantly outperforms the traditional CNN, demonstrating the importance of using deeper architectures for image classification tasks.

6.3. Implementation of EfficientNetB0 Model:

The EfficientNetB0 model is utilized for classifying chili plant leaf images into three categories: healthy, Cercospora, and wilt_disease. The dataset is structured into training, validation, and test sets, and image augmentation techniques such as resizing, normalization, and horizontal flipping are applied to enhance model generalization. The EfficientNetB0 model, pre-trained on the ImageNet dataset, is modified by replacing its classifier head with a custom fully connected layer that maps the extracted features to three output classes. The modified classifier consists of a 256-unit fully connected layer with ReLU activation, dropout for regularization, and a final classification layer using softmax activation.

The model is trained using the Adam optimizer with a learning rate of 0.001, and categorical cross-entropy is employed as the loss function. A key component of this implementation is early stopping, which prevents overfitting by monitoring validation loss and halting training if no improvement is observed for five consecutive epochs. After training, the best-performing model (based on validation loss) is saved and used for testing. The test performance is evaluated using accuracy, confusion matrix, and classification report metrics. Additionally, precision-recall and ROC curves are plotted for each class to visualize model performance in handling imbalanced data.

The training computation time per epoch is also analyzed to assess computational efficiency. By leveraging EfficientNetB0's strong feature extraction capabilities and fine-tuning its classifier, the model achieves high accuracy, making it a robust solution for chili leaf disease detection.

7. EXPERIMENTAL RESULTS

7.1. Experimental Results of MLP-LSTM Model:

Figure 12 shows the training accuracy and validation accuracy versus epochs. Regularization techniques such as L2, Dropout, and Early stopping are employed during the training of the model to prevent overfitting by constraining the model's capacity. This led to a slightly higher accuracy of the validation dataset when compared to the training dataset accuracy. Due to this reason, the plot in figure 12 shows the training accuracy curve below the validation accuracy curve. The non-declining behavior of validation accuracy and inclining behavior of training accuracy after a few numbers of epochs show the absence of overfitting in the model and show good capability of generalization.

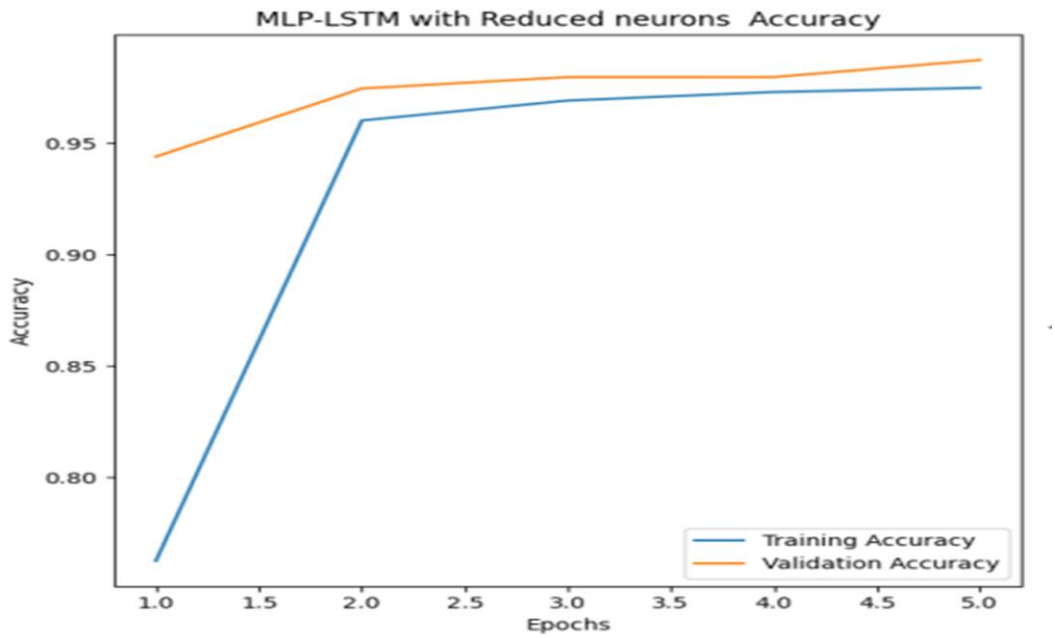

Figure 12. The Training Accuracy and Validation Accuracy Versus Epochs

Figure 13 shows that training loss is greater than validation loss, which shows a well-regularized model that is not overfitting. The constant lowering of validation loss without increasing at any epoch shows good generalization. The higher training loss compared to validation loss exhibits the model is memorizing training data or is not over-optimized. The constraints introduced by the regularization techniques such as L2 regularization and Dropout in the training process penalize the few parameter updates, which results in increasing training loss to prevent the model from being overfitted.

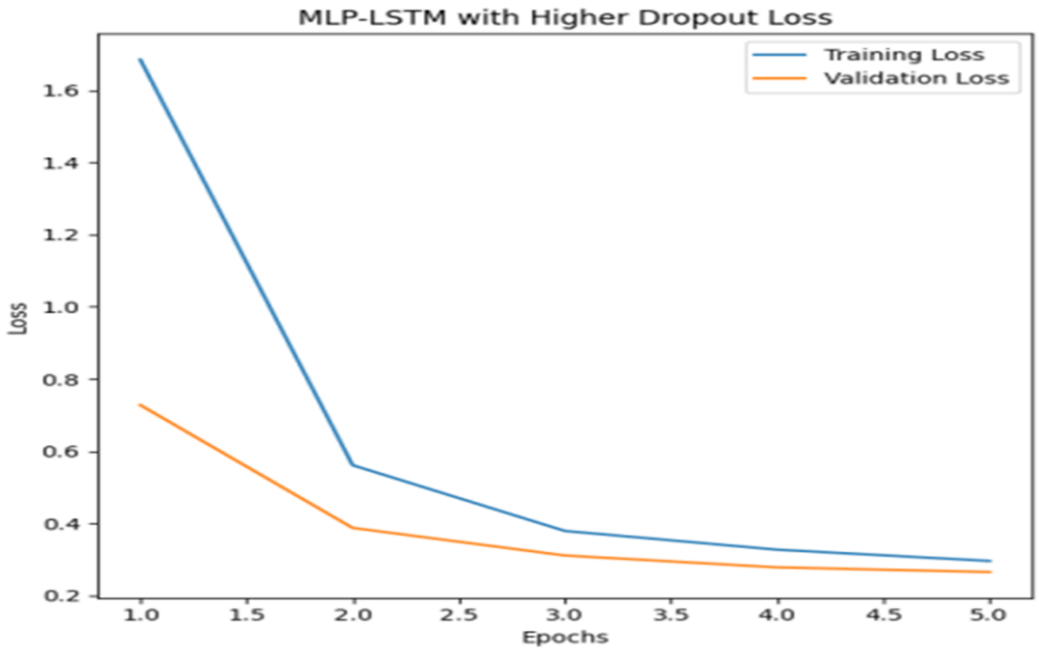

Figure 13. The Training Loss and Validation Loss versus Epochs

Figure 14 shows the confusion matrix of the MLP-LSTM model. The elements present in the diagonal indicate the samples that were correctly predicted. 129 samples were accurately classified as dew, similarly 131 samples as irrigation, and 131 samples as rainfall. 2 of the samples were misclassified as irrigation for dew. The diagonal elements of the confusion matrix showed the model did not favour any individual class. This shows the model is not overfitted.

Figure 15 shows the AUC-ROC curve for MLP-LSTM was nearly perfect with an AUC of 1.00 for all three classes. The true positive rate of the model was almost 1.0 for all false positive rates, with no misclassification. Both the macro-average and micro-average AUC values are also 1.00, which reinforces that the model does a great job of predicting the class. Thus, the model is perfect at differentiating classes without any kind of trade-off in performance

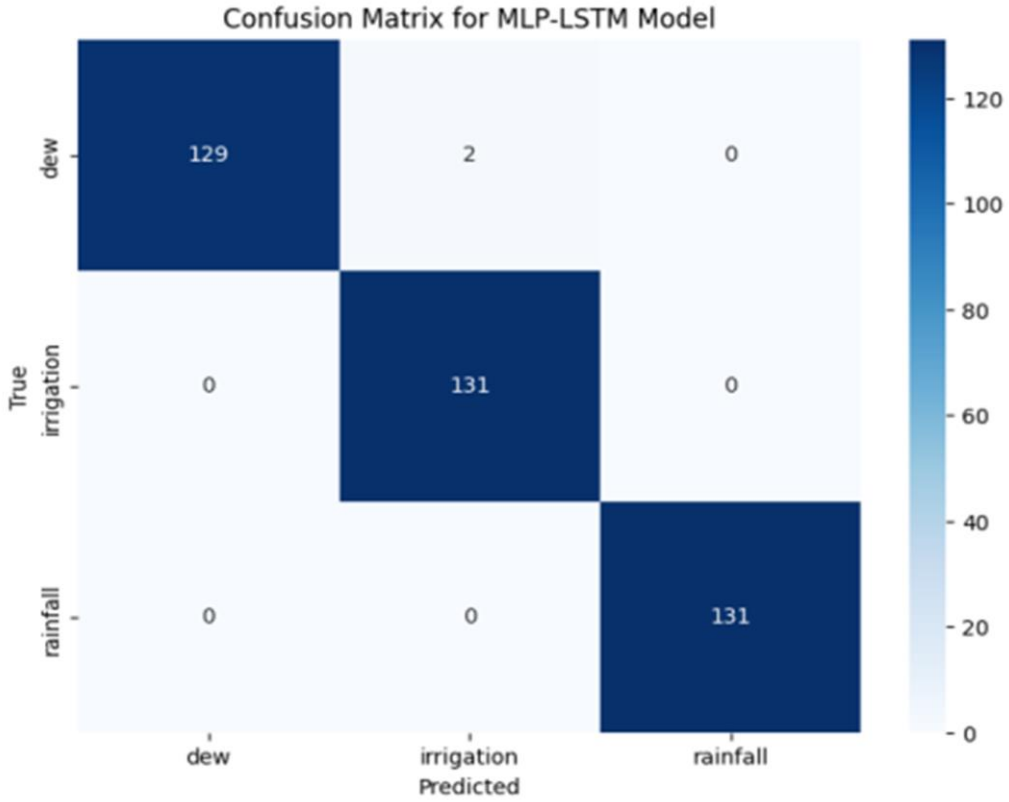

Figure 14.Confusion Matrix of MLP-LSTM Model

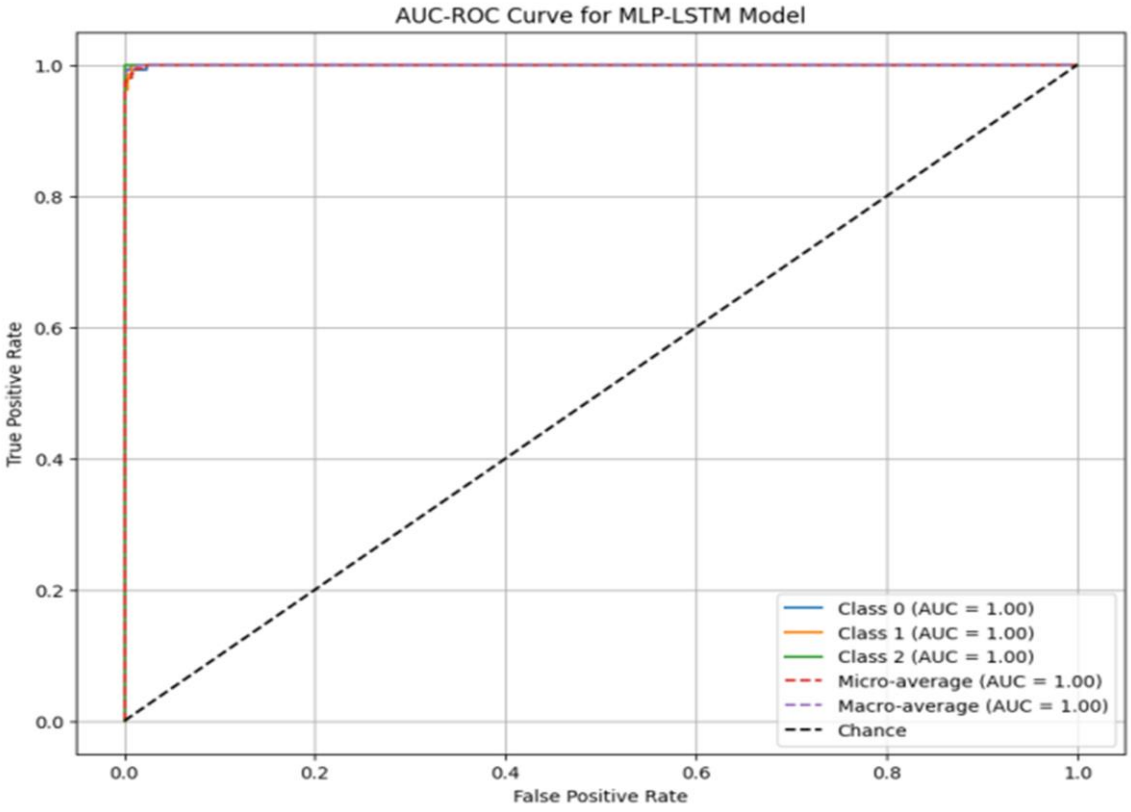

Figure. 15. AUC-ROC Curve of MLP-LSTM Model

7.2. Experimental Results of CNN Model:

Figure 16 shows the Precision-Recall Curve of the Convolutional Neural Network Model. The Precision-Recall (PR) curve is an essential evaluation metric for classification models, particularly in imbalanced datasets, where the number of instances belonging to each class varies significantly. In the case of the CNN classification model for detecting Cercospora, Healthy, and Wilt disease classes in chili plant leaves, the PR curve helps assess the trade-off between precision and recall at different classification thresholds. For the Cercospora class, precision starts relatively low (~0.32) but gradually increases as recall remains at 1.0 for a long stretch.

This suggests that the model correctly identifies all Cercospora cases but does so with low confidence initially. However, as recall starts to decrease, precision improves significantly, eventually reaching 1.0, indicating that at a higher threshold, the model only predicts Cercospora when it is almost certain, though it may miss some positive cases. This trend is characteristic of a classifier that is initially over-predicting Cercospora but refines its decisions at higher thresholds. For the Healthy class, the precision-recall curve follows a similar trend but maintains a more stable increase in precision as recall drops. Initially, recall is perfect (1.0), but precision is around 0.33-0.35, meaning many non-healthy samples are being misclassified as healthy.

However, precision improves as the model becomes more confident, eventually reaching 1.0 when recall is nearly zero. This indicates that at high thresholds, the model identifies healthy leaves correctly but at the expense of missing many actual healthy cases. For Wilt Disease, the precision-recall curve shows that precision fluctuates more compared to the other two classes. Initially, precision is around 0.34, with recall at 1.0. Unlike the other two classes, Wilt Disease exhibits more variation in precision, which suggests that the model sometimes confuses wilt disease with Cercospora or healthy samples, especially at intermediate thresholds.

Overall, Cercospora and Healthy classes show a steady increase in precision as recall decreases, indicating that as the model becomes more confident, its predictions improve. Wilt Disease, on the other hand, shows more variation in precision, suggesting a need for better feature extraction or threshold tuning to improve classification confidence.

The Average Precision (AP) scores indicate how well the model balances precision and recall for each class: Cercospora (AP = 1.00): The model achieves a perfect AP score, meaning it consistently identifies Cercospora with high confidence. Healthy (AP = 0.78): A reasonably good score, suggesting that while there are misclassifications, the model still captures a good proportion of Healthy instances.

Wilt Disease (AP = 0.52): The lowest AP score, indicating difficulty in precisely and consistently identifying Wilt Disease cases. The PR curve highlights that Cercospora is well classified, but Wilt Disease suffers from lower recall and precision.

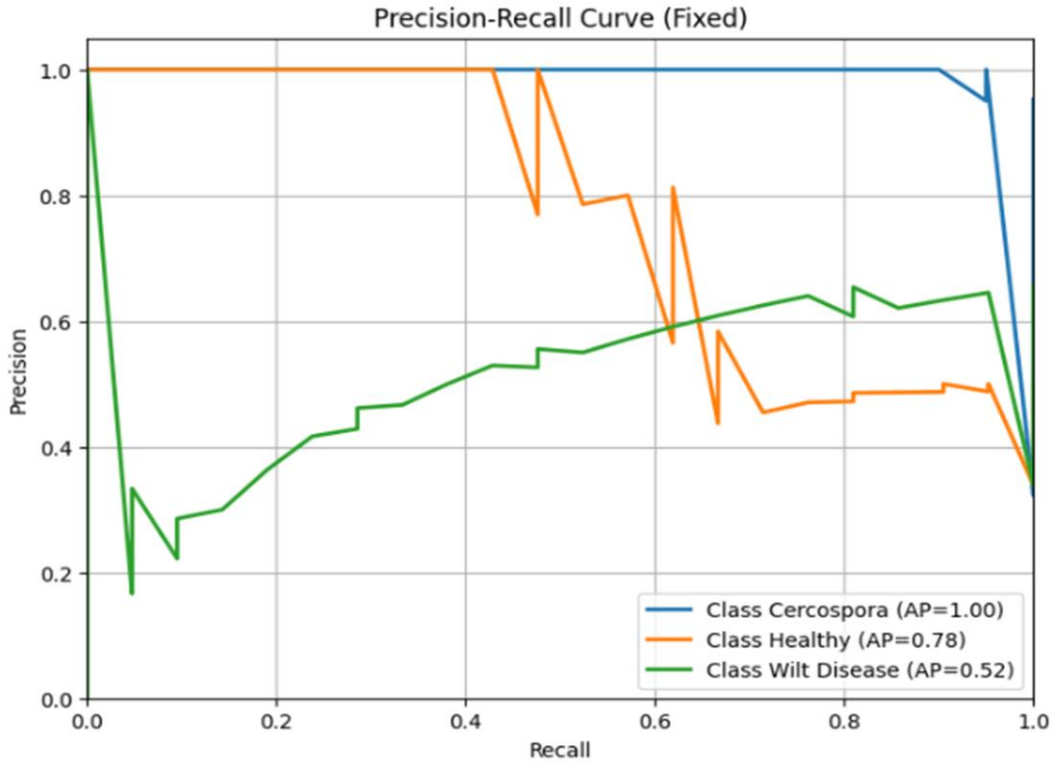

Figure 16. Precision-Recall (PR) Curve for CNN Model

Figure 17 Shows the AUC-ROC Curve of the EfficientNetB0Model. The Receiver Operating Characteristic (ROC) curve visually represents a classification model's performance by plotting the True Positive Rate (TPR) against the False Positive Rate (FPR). The Area Under the Curve (AUC) quantifies the model's ability to distinguish between classes, with higher AUC values indicating better performance. A CNN model has been evaluated for classifying chili plant leaf images into Cercospora, healthy, and wilt disease. The ROC curve for each class helps assess how well the model differentiates between categories.

One of the classes exhibits an AUC equal to 1.00, indicating near-perfect classification with minimal errors. Another class shows the AUC= 0.82, suggesting that while the model performs well, there is some level of misclassification. Certain points on the ROC curve reveal that increasing FPR does not significantly improve TPR, highlighting classification challenges. The third class has the AUC= 0.80, reflecting reduced performance. The ROC curve suggests that at some thresholds, the model struggles to distinguish between this category and others. This could be due to overlapping features, class imbalance, or dataset limitations.

The ROC-AUC values confirm that Cercospora is well classified, while Healthy and Wilt Disease have overlapping features that make them harder to distinguish. The CNN model performs exceptionally well in classifying Cercospora but struggles to differentiate between Healthy and Wilt Disease. The confusion matrix and specificity indicate high misclassification rates between these two classes. The PR and ROC-AUC scores further confirm that Wilt Disease is the most challenging class to classify correctly. Future improvements could involve data augmentation, better feature extraction, or fine-tuning the CNN architecture to enhance classification performance for Healthy and Wilt Disease classes.

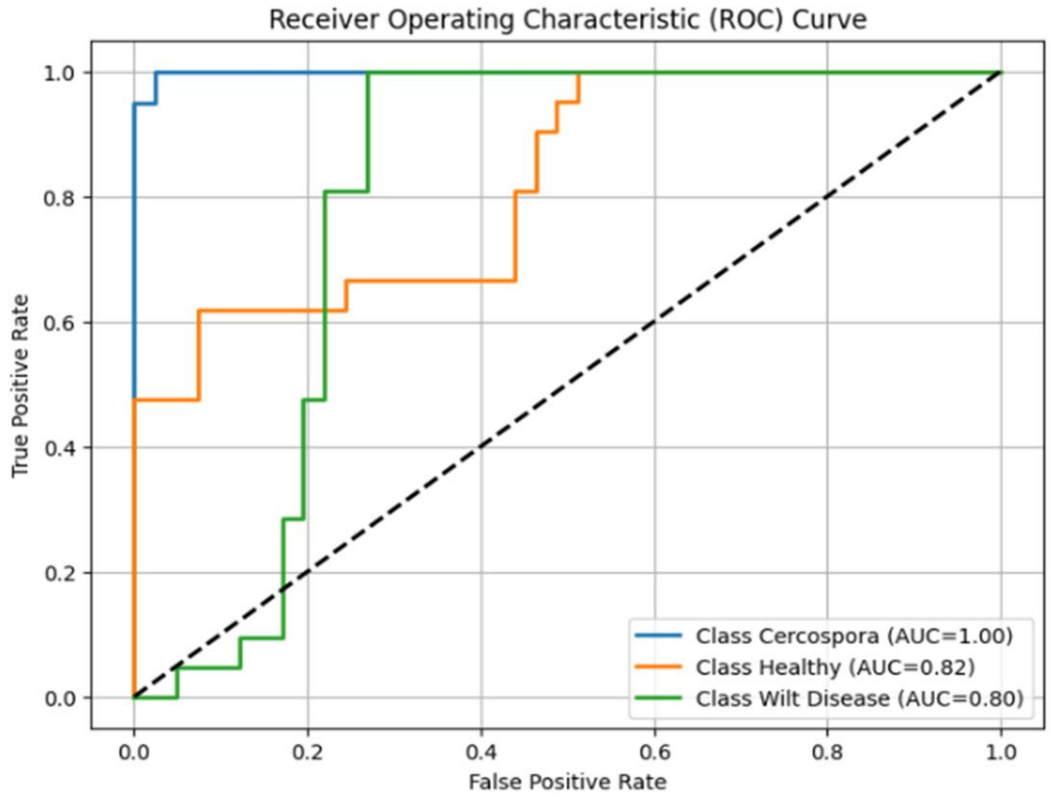

Figure 17. AUC-ROC Curve of CNN Model

Figure 18 shows the confusion matrix of CNN Model, it provides an overview of how the CNN model performed in classifying the three classes: Cercospora, Healthy, and Wilt Disease. It shows the number of correct and incorrect predictions for each class. Cercospora: The model correctly classified 19 out of 20 instances, with only 1 misclassification (classified as Wilt Disease). Healthy: The model correctly classified 14 out of 21 instances but misclassified 7 instances as Wilt Disease. Wilt Disease: The model correctly classified 8 out of 21 instances but 12 were misclassified as Healthy and 1 as Cercospora.

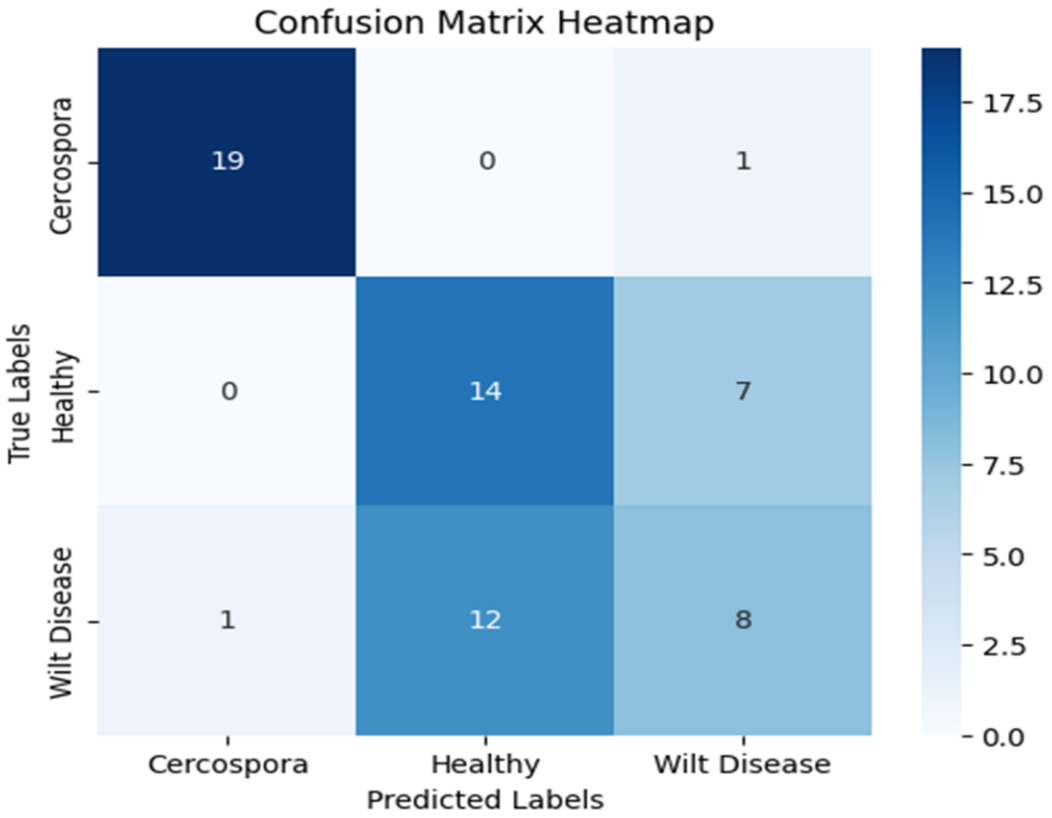

Figure 18. Confusion Matrix of CNN Model

7.3. Experimental Results of EfficientNetB0 Model

Figure 19 shows the Precision-Recall curve of the EfficientNetB0 Model. For the Cercospora class, the precision starts at approximately 0.32 and gradually increases, reaching a perfect score of 1.0 when the recall decreases to 0.25. This indicates that the model can achieve perfect precision, but only when it predicts a small number of samples with high confidence. The high recall at lower precision values suggests that the model is effective in detecting Cercospora but might misclassify some instances due to overlapping features with other classes. The healthy class exhibits a similar trend, with precision starting at 0.33 and gradually increasing.

The recall remains at 1.0 for most of the curve, meaning that the model correctly identifies almost all healthy samples. However, as precision increases beyond 0.86, the recall starts to drop. This drop suggests that when the model becomes stricter in classification, it starts missing some healthy samples, leading to a lower recall. For the wilt disease class, the model initially maintains a balance between precision and recall, but a significant drop in recall is observed at higher precision values. The recall drops below 0.50 when the precision reaches approximately 0.90, which suggests that while the model is highly confident in its positive predictions, it sacrifices recall, leading to a reduced number of correctly identified wilt disease cases.

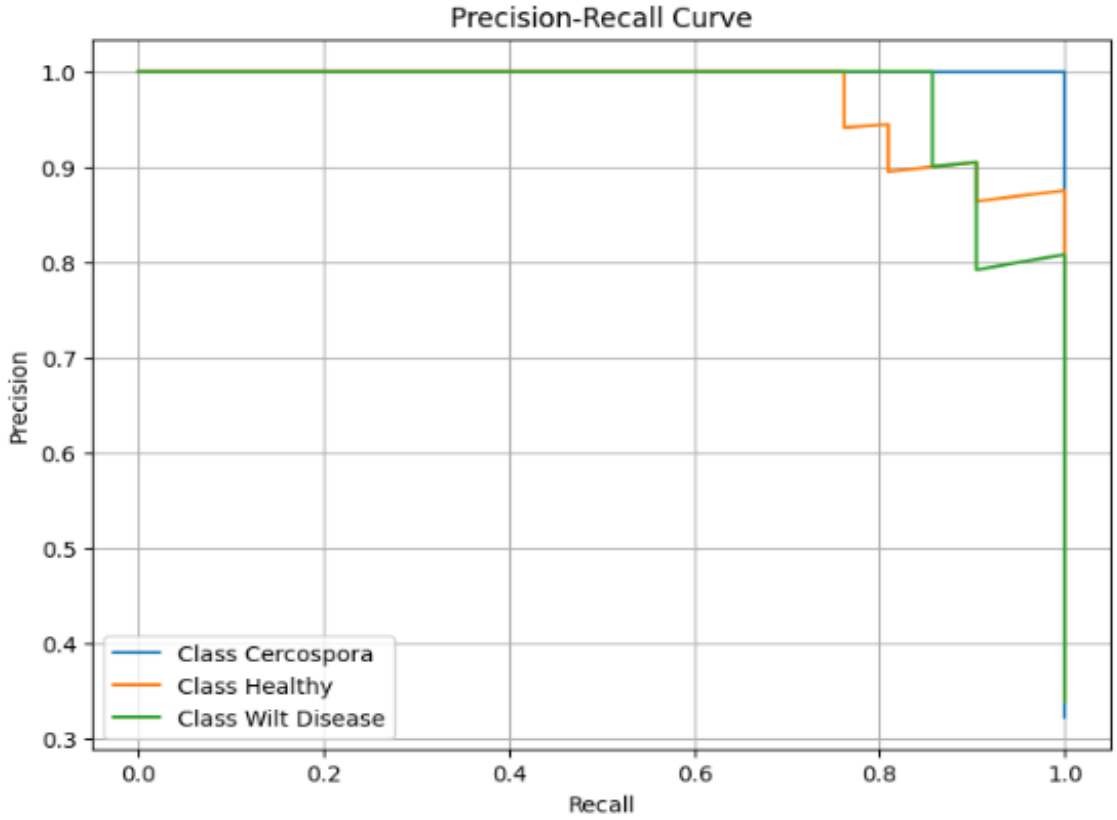

Figure 19. Precision-Recall Curve of EfficientNetB0 Model

Overall, the EfficientNetB0 model exhibits strong classification capabilities, particularly when recall is prioritized over precision. However, as precision increases, recall tends to decline, indicating that the model becomes more selective in its predictions. This behavior is common in deep learning models and can be further optimized by adjusting decision thresholds, incorporating additional data augmentation techniques, or leveraging class-specific weighting to balance precision and recall effectively

Figure 20 shows the AUC-ROC curve of the EfficientNetB0 Model. For the Cercospora class, the AUC is 1.00, indicating perfect discrimination. The false positive rate (FPR) remains at 0 until a threshold where it jumps to 1, while the true positive rate (TPR) moves from 0 to 1. This implies that the model can effectively separate Cercospora cases from the other two classes with high confidence. The Healthy class has an AUC of 0.99, which is nearly perfect. The FPR starts at 0, and increases slightly at intermediate thresholds (0.0244 to 0.0732), while the TPR rapidly rises from 0.0476 to 1.0. This suggests that while the model misclassifies a few healthy leaves as diseased at some thresholds, it still distinguishes healthy leaves exceptionally well.

For the Wilt Disease class, the AUC is also 0.99, showing a strong classification ability. The FPR rises slightly at a threshold (0.0488 to 0.1219), and the TPR increases steadily from 0.0476 to 1.0. This means the model sometimes confuses wilt disease with other classes at specific threshold points but remains highly accurate overall. The EfficientNetB0 model exhibits high precision-recall and near-perfect AUC-ROC scores, indicating a strong ability to classify Cercospora, Healthy, and Wilt Disease leaves accurately.

The AUC values of 1.00 (Cercospora) and 0.99 (Healthy, Wilt Disease) suggest that the model performs exceptionally well in distinguishing between classes. While minor misclassifications occur, they are minimal, making this model highly reliable for chili plant leaf disease detection.

Figure 21 shows the confusion matrix of EfficientNetB0 Model. For the Cercospora class, the confusion matrix shows that 19 samples were correctly classified as Cercospora (true positives), while 1 sample was incorrectly classified as Wilt Disease (false negative). This means the model performs well in identifying Cercospora cases, with minimal misclassification. For the Healthy class, 17 samples were correctly classified (true positives), while 4 samples were misclassified as Wilt Disease (false negatives). No healthy samples were wrongly predicted as Cercospora. This indicates that the model sometimes confuses healthy leaves with wilt disease, possibly due to visual similarities in symptoms.

For the Wilt Disease class, 20 samples were correctly classified (true positives), while 1 sample was mistakenly classified as Healthy (false negative). No Wilt Disease samples were misclassified as Cercospora. This suggests

that the model effectively identifies Wilt Disease, though there is a small error in distinguishing it from healthy leaves,

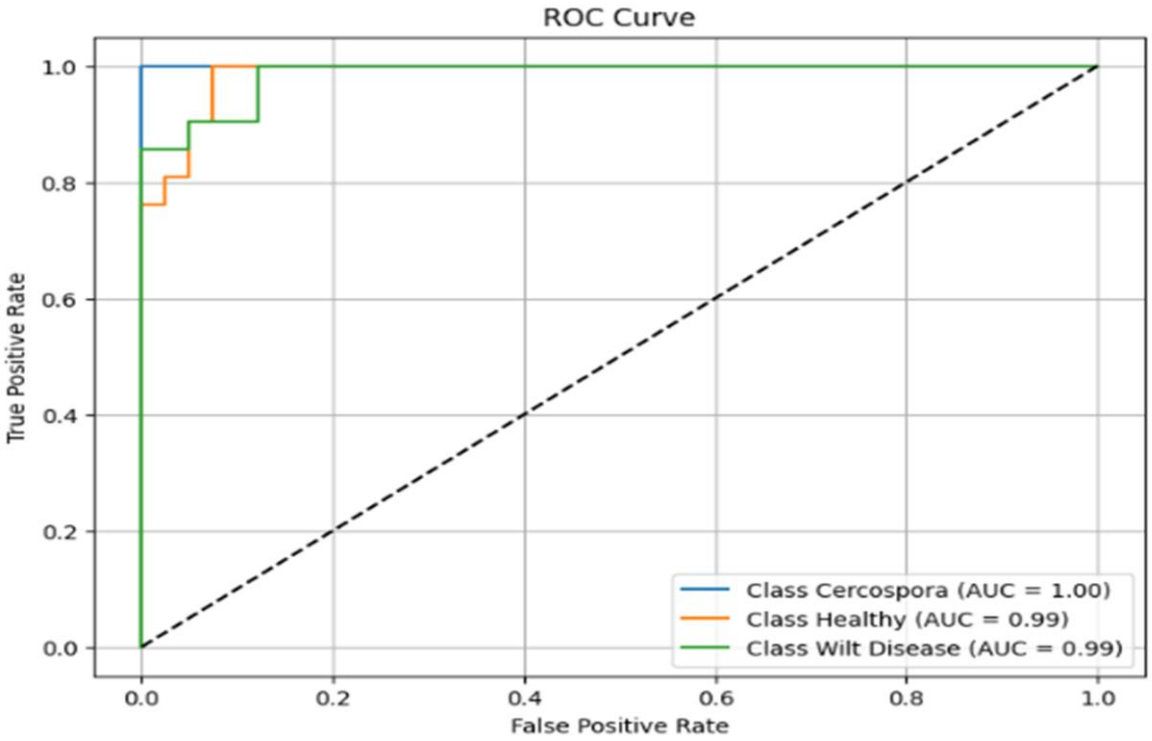

Figure 20. AUC-ROC Curve of EfficientNetB0 Model

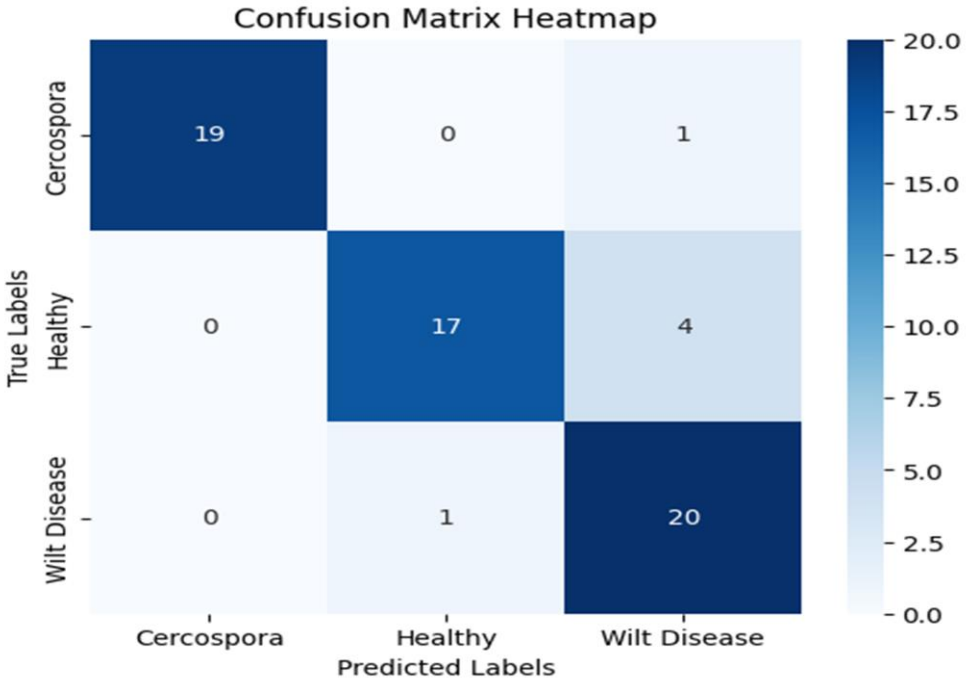

Figure 21. Confusion Matrix of EfficientNetB0 Model

Table 8 shows the report of the three models implemented in our research work and their corresponding evaluation metrics. Figure 26 visualizes the comparision of three models compared in the table 8. The MLP-LSTM Hybrid Model demonstrated the highest performance, achieving an impressive 99.731% accuracy and a 0.99 sensitivity, indicating its strong ability to correctly identify positive cases. The EfficientNetB0 Model follows with a solid 90.32% accuracy, 0.90 sensitivity, and 91.55% specificity, showing a well-balanced performance in both identifying positive and negative cases. In contrast, the Convolutional Neural Network (CNN) Model exhibited the lowest performance, with an accuracy of 66.13%, sensitivity of 0.66, and specificity of 66.51%, suggesting challenges in distinguishing between different classes. Overall, while the MLP-LSTM Hybrid Model appears to be the most effective, the EfficientNetB0 Model maintains a strong balance between sensitivity and specificity, whereas the CNN Model shows room for improvement.

Figure 22 shows the training accuracy of the CNN model has a consistent upward trend, starting at 39.68% and reaching a peak of 83.00%, indicating that the model is learning effectively over epochs. However, there are slight fluctuations, such as at epochs 6 and 9, where the accuracy momentarily dips before recovering. The validation accuracy follows a similar trend, starting at 48.39% and improving to 74.19%, though it experiences more instability compared to training accuracy. Notably, the validation accuracy fluctuates around epochs 5 and 9, which might suggest minor overfitting or challenges in generalizing to unseen data.

Figure 23 shows the training loss of the CNN model decreases from 1.0809 to 0.3963, reflecting a successful reduction in errors as the model learns. However, there are fluctuations around epochs 6 and 9, where the loss momentarily increases, suggesting some instability in training. The validation loss also decreases from 1.0433 to 0.6120, but it does not follow a consistently downward trend. Instead, there are noticeable fluctuations, particularly around epochs 7 and 10, where it temporarily increases. This variability in validation loss suggests that while the model improves, it may struggle with generalization and could benefit from techniques like regularization or early stopping.

Figure 24 shows the training accuracy of the EfficientnetB0 model starts at 75.10% and steadily improves to 98.58%, demonstrating strong learning progress. The model quickly reaches above 90% accuracy within a few epochs and maintains

high performance with minimal fluctuations. However, the validation accuracy, while following a generally positive trend, exhibits more variability. It begins at 77.42%, peaks at 88.71%, and fluctuates around 79% to 88% in later epochs. These fluctuations suggest that the model is learning well but may be experiencing some instability in generalization, possibly due to overfitting.

Figure 25 shows the training loss shows a clear downward trend, starting at 0.6009 and reducing to 0.0441, indicating that the model is effectively minimizing errors. However, the validation loss is highly inconsistent, starting at 0.7127, dropping sharply to 0.3396, but then spiking to 1.2147 before fluctuating throughout training. These fluctuations suggest that the model may struggle with generalization, as lower training loss does not always correspond to a stable validation loss. The inconsistency in validation loss, despite high training accuracy, suggests potential overfitting, and techniques like dropout or early stopping could be useful to improve stability.

Table 8. Report of the MLP-LSTM, CNN and EfficientNetB0 Models

Model Name	**Accuracy**	**Sensitivity**	**Specificity**
MLP-LSTM Hybrid Model	99.731%	0.99	----
Convolutional Neural Network Model	66.13%	0.66	66.51%
EfficientNetB0 Model	90.32%	0.90	91.55%

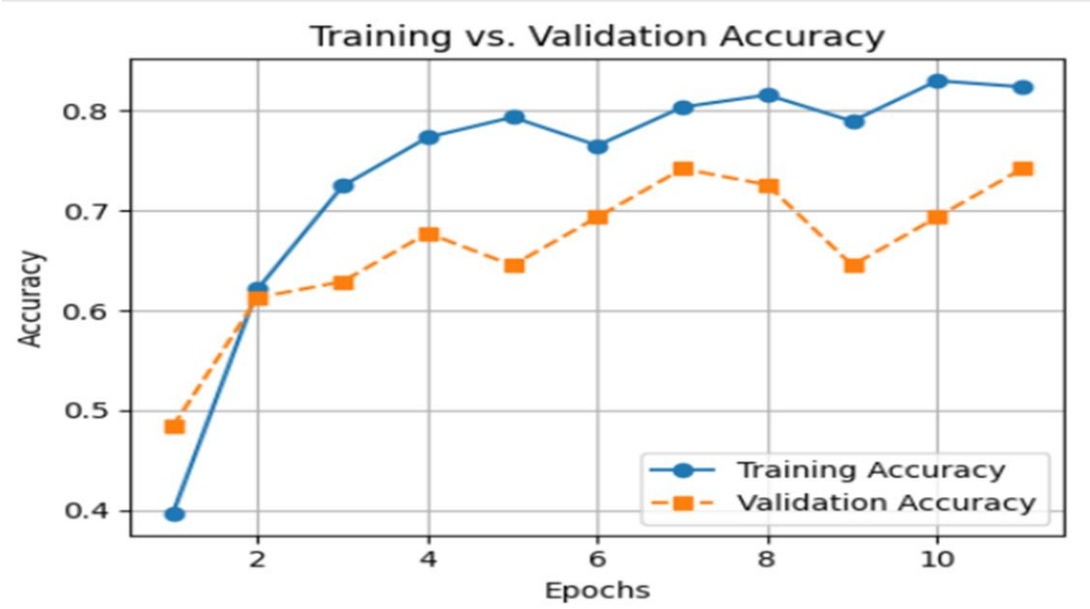

Figure 22. Training vs Validation Accuracy of CNN Model

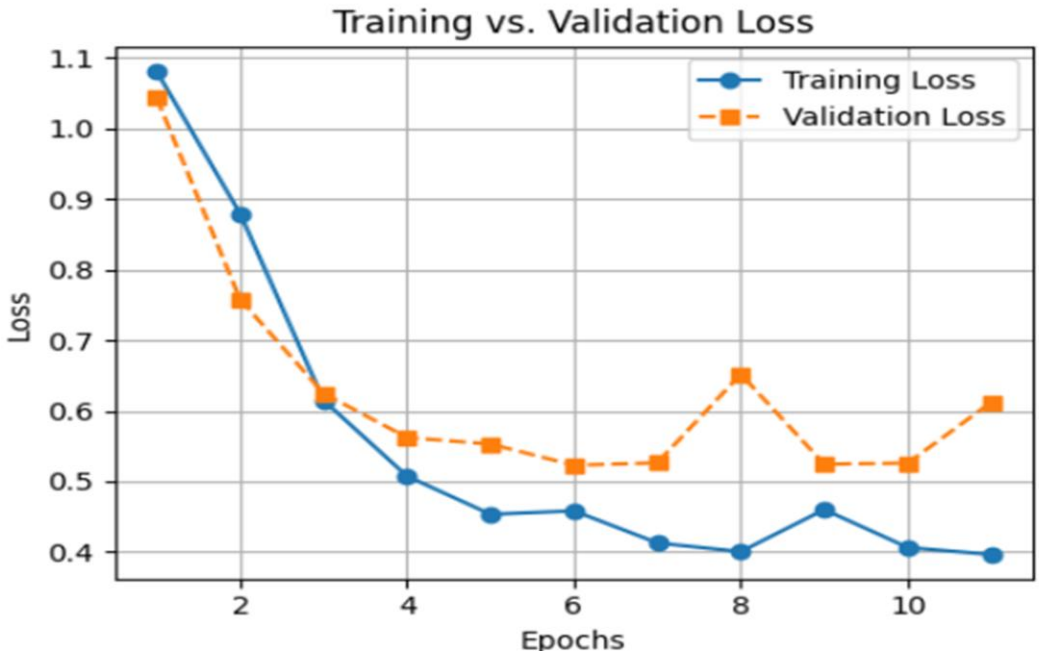

Figure 23. Training vs Validation Loss of CNN Model

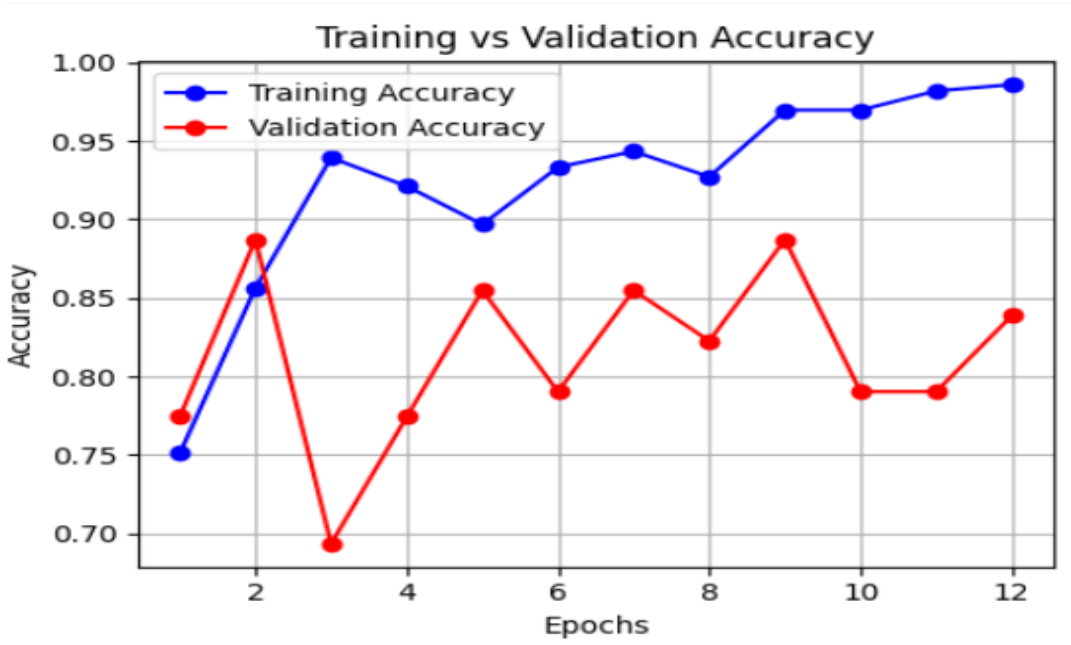

Figure 24. Training vs Validation Accuracy of EfficientNetB0 Model

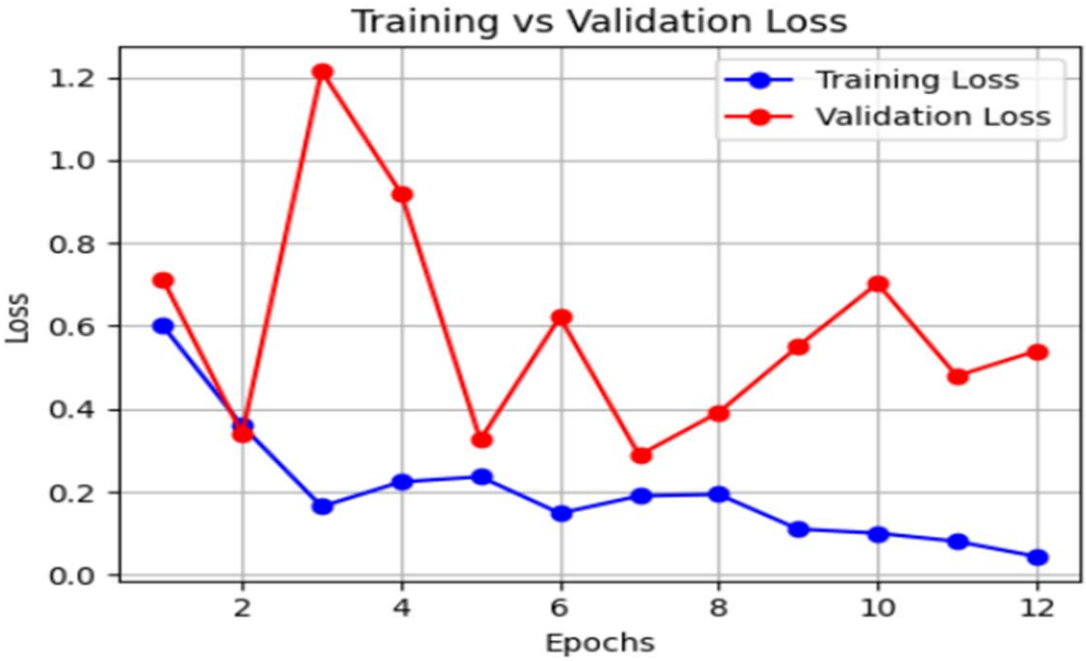

Figure 25. Training vs Validation Loss of EfficientNetB0 Model

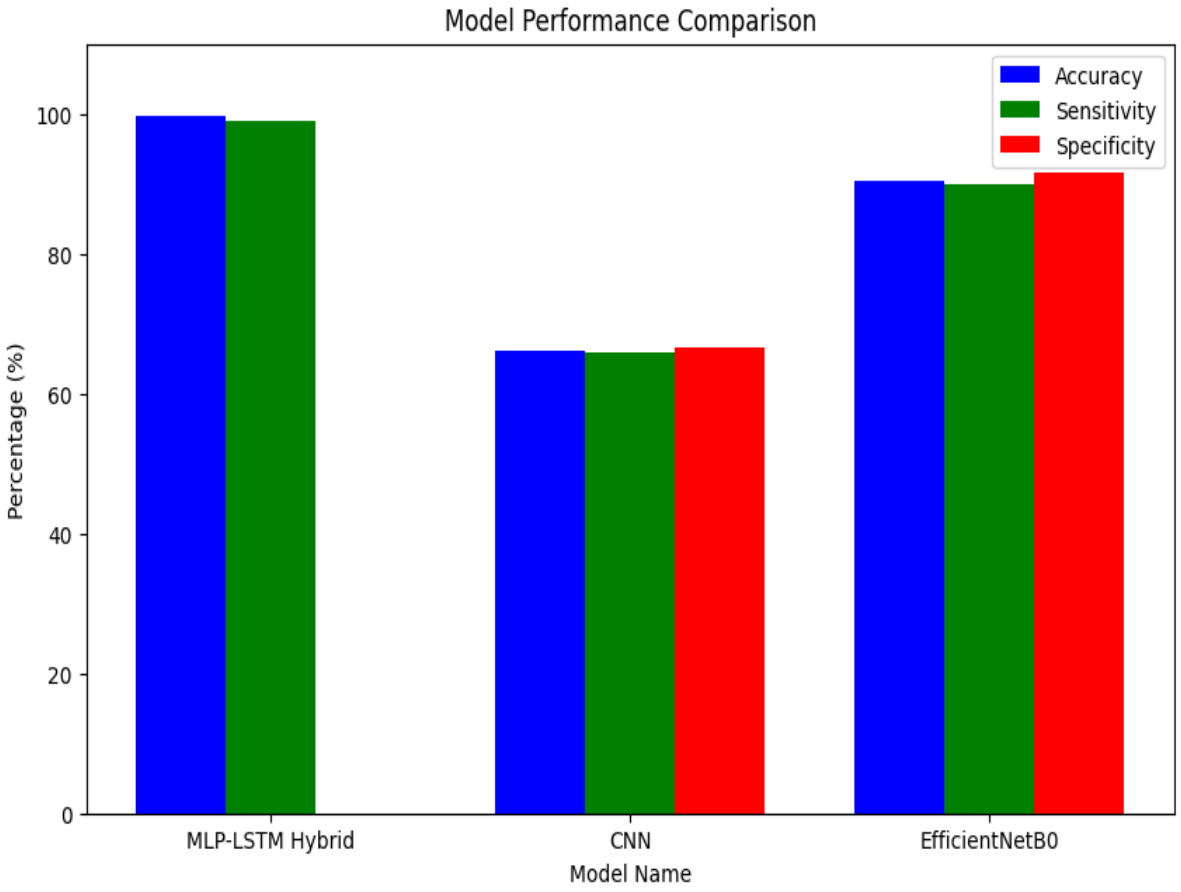

Figure 26. Model Performance Comparison

8. CONCLUSION

This study presents the dual capability of numerical source wetness sensor classification and image-based disease detection in assessing source impacts on leaf wettability in chili peppers. The MLP-LSTM model achieved a perfect classification of wetness sources—dew, rainfall, and irrigation—at an astounding accuracy rate of 99.71%. For the detection of Cercospora and wilt disease, the EfficientNetB0 model outperformed the conventional CNN which achieved an accuracy of 66.6 percent, by an extraordinary margin at an accuracy rate of 95.16%. This paper correlates classified wetness sources with diseases to show that overhead irrigation is a major contributor to Cercospora and wilt disease due to prolonged exposure of the leaves to moisture. These results stress precision irrigation strategies aimed at reducing outbreaks of diseases late in crops as they improve health. Proposed frameworks provide farmers with efficient decision support systems for early disease forecasting coupled with optimum irrigation management for enhanced agricultural productivity.

9. FUTURE WORK

Future research focuses on enhancing the model's robustness by expanding the dataset to include a more diverse range of chili plant leaf images and sensor readings across different varieties and environmental conditions. Additionally, developing an IoT-based real-time monitoring system can enable continuous data collection and processing of leaf wetness and temperature, facilitating automated disease prediction and proactive intervention. Furthermore, deploying and validating the proposed approach in real agricultural settings will help assess its practical effectiveness, refine prediction accuracy, and optimize recommendations for precision irrigation management, ultimately supporting sustainable and data-driven farming practices.

REFERENCES

1. G. Delnevo, et al., "A deep learning and social IoT approach for plant disease prediction toward a sustainable agriculture," IEEE Internet of Things Journal 9, 7243–7250 (2021).

2. R. P. Sharma, et al., "IoT-enabled IEEE 802.15.4 WSN monitoring infrastructure-driven fuzzy-logic-based crop pest prediction," IEEE Internet of Things Journal 9, 3037–3045 (2021).

3. K. S. Patle, et al., "IoT enabled, leaf wetness sensor on the flexible substrates for in-situ plant disease management," IEEE Sensors Journal 21, 19481–19491 (2021).

4. A.D. Boursianis, et al., "Smart irrigation system for precision agriculture—The ARETHOU5A IoT platform," IEEE Sensors Journal 21, 17539–17547 (2020).

5. M. Kumar, A. Kumar, and V. S. Palaparthy, "Soil sensors-based prediction system for plant diseases using exploratory data analysis and machine learning," IEEE Sensors Journal 21, 17455–17468 (2020).

6. Y. Sun, et al., "Automatically designing CNN architectures using the genetic algorithm for image classification," IEEE Transactions on Cybernetics 50, 3840–3854 (2020).

7. C. Velásquez, C. D. M. Castroverde, and S. Y. He, "Plant–pathogen warfare under changing climate conditions," Current Biology 28, R619–R634 (2018).

8. Y. Li and H. Hong, "Modelling flood susceptibility based on deep learning coupling with ensemble learning models," Journal of Environmental Management 325, 116450 (2023).

9. V. B. Semwal, A. Gupta, and P. Lalwani, "An optimized hybrid deep learning model using ensemble learning approach for human walking activities recognition," The Journal of Supercomputing 77, 12256–12279 (2021).

10. N. Rane, S. P. Choudhary, and J. Rane, "Ensemble deep learning and machine learning: applications, opportunities, challenges, and future directions," Studies in Medical and Health Sciences 1, 18–41 (2024).

11. K. S. Patle, et al., "Field evaluation of smart sensor system for plant disease prediction using LSTM network," IEEE Sensors Journal 22, 3715–3725 (2021).

12. M. Solís and V. Rojas-Herrera, "Approaches for the prediction of leaf wetness duration with machine learning," Biomimetics 6, 29 (2021).

13. B. Gama, et al., "Evaluation of a multi-model approach to estimate leaf wetness duration: an essential input for disease alert systems," Theoretical and Applied Climatology 149, 83–99 (2022).

14. K. Alsafadi, et al., "Prediction of daily leaf wetness duration using multi-step machine learning," Computers and Electronics in Agriculture 224, 109131 (2024).

15. L. L. Marcuzzo and D. Füchter, "Influence of temperature and daily leaf wetness duration on the severity of bacterial leaf blight of garlic," Summa Phytopathologica 47, 180–182 (2021).

16. S. Zito, et al., "Optimization of a leaf wetness duration model," Agricultural and Forest Meteorology 291, 108087 (2020).

17. J. Eunice, et al., "Deep learning-based leaf disease detection in crops using images for agricultural applications," Agronomy 12, 2395 (2022).

18. M. V. Mallikarjuna Nandi, "Plant leaf disease detection using convolutional neural network," International Journal of Science and Research (IJSR) 13, 1545–1548 (2024).

19. Y. Resti, et al., "Ensemble of Naive Bayes, Decision Tree, and Random Forest to predict air quality," IAES International Journal of Artificial Intelligence 13, 3039–3051 (2024).

20. N. Buslim, "Ensemble learning techniques to improve the accuracy of predictive model performance in the scholarship selection process," Journal of Applied Data Sciences 4, 264–275 (2023).

21. R. Saini, P. Garg, N. K. Chaudhary, M. V. Joshi, V. S. Palaparthy, and A. Kumar, "Identifying the source of water on plant using the leaf wetness sensor and via deep learning-based ensemble method," IEEE Sensors Journal 24, 7009–7017 (2024).

22. Theerthagiri, P., Ruby, A. U., Chandran, J., Sardar, T. H., & Shafeeq B¬¬M, A. “Deep SqueezeNet learning model for diagnosis and prediction of maize leaf diseases”. Journal of Big Data, 11, 1-16, (2024).

23. Husna Tabassum, and T. Prasannavenkatesan, “Review of image processing and artificial intelligence methodologies for apple leaf disease diagnosis”, IAES International Journal of Artificial Intelligence,13, 2459-2471, (2024).

24. Husna T, and Prasannavenkatesan, “Performance Analysis of AI-based Learning Models on Leaf Disease Prediction”, 4th International Conference on Circuits, Control, Communication and Computing, IEEE, 21st to 23rd December 2022, Bengaluru, India.

25. C. H. R., et al., "Tomato and Chilli Plant Disease Detection using CNN," International Journal of Engineering Research & Technology (IJERT), Special Issue - 2022, ICEI - 2022 Conference Proceedings, ISSN: 2278-0181.

26. Mahmood ur Rehman, Muhammad, et al. "Leveraging Convolutional Neural Networks for Disease Detection in Vegetables: A Comprehensive Review." Agronomy 14.10 (2024): 2231.

27. Alhijaj, Jenan A., and Raidah S. Khudeyer. "Integration of efficientnetb0 and machine learning for fingerprint classification." Informatica 47.5 (2023).

28. Iqbal, Amna, Muhammad Arfan Jaffar, and Rashid Jahangir. "Enhancing Brain Tumour Multi-Classification Using Efficient-Net B0-Based Intelligent Diagnosis for Internet of Medical Things (IoMT) Applications." Information 15.8 (2024): 489.