Rice Disease Classification Using Hybrid Deep Learning and Handcrafted Feature Integration

# Abstract

In the field of machine learning, feature extraction is important for the accurate classification of images. Building on the advancements in Convolutional Neural Networks (CNNs), this study addresses the identification of rice leaf diseases by leveraging both deep and handcrafted features to enhance classification performance. Utilizing dataset comprising 5932 on field images of rice leaves affected by bacterial blight, brown spot, tungro, and blast, we checked the efficacy transfer learning and also of combining deep features from multiple CNN models together and with handcrafted features. The performance of the hybrid approaches was benchmarked using multiple shallow machine learning classifiers.

Results indicated that while transfer learning’s performance for ResNet50, VGG16 and VGG19 were good, the results from feature fusion between VGG16 and VGG19 gave the best overall result. The performance of Handcrafted Features specifically LBP with shallow machine learning classifiers was comparable with transfer learning.

The best performance was denoted by ResNet50 using transfer learning approach, feature fusion of VGG16 and VGG19 using SVM classifier, and ------------------------------------------------. Our hybrid model outperformed traditional CNN-SVM models, with the ResNet50-EfficientNetB3-HOG combination achieving superior F1 scores. Additionally, this approach surpassed other traditional image classification methods such as bag-of-features, local binary patterns (LBP) plus SVM, and the Gray Level Co-occurrence Matrix (GLCM) plus SVM. This study underscores the potential of integrating deep and handcrafted features for robust image classification in agricultural applications, offering a promising solution for the early and accurate detection of rice leaves diseases, ultimately aiding in effective crop management and disease control.

# Introduction

Rice is one of main food sources in India, with land under rice cultivation more than even China’s cultivation. Odisha ranks fourth among Indian states in rice production, with the western area, specially the Sambalpur and Bargarh districts which are known as the rice bowl of Odisha, are renowned for rice cultivation. This region cultivates various rice varieties across two farming seasons, Kharif season (which is from July to October), dependent on the monsoon, and Rabi season (which is from October to March), reliant on the Hirakud dam's water supply. Each year, paddy fields suffer from various diseases and pest attacks, posing significant challenges to local farmers. Particularly, young farmers with less experience in agriculture struggle to identify these diseases, leading to ineffective pesticide application. This pressing issue motivated our research on identifying rice diseases prevalent in western Odisha.

In this region, four primary rice diseases are commonly observed: bacterial blight, brown spot, tungro and blast. Traditional methods of identifying diseases, such as visual inspection and laboratory experimentation, are either time-consuming or require expert personnel and chemical reagents, making them impractical for widespread use. While mobile applications like Rice Doctor app and Rice Xpert app have been developed to assist farmers, they often yield inaccurate diagnoses and lack efficiency.

Numerous research studies have focused on automatic rice disease diagnosis using machine learning techniques and image processing techniques. These studies employ various methods, including support vector machines, pattern recognition, computer vision and digital image processing and, not only for rice, but also for other crops like wheat, cucumber, citrus, maize, cotton, and tomato. However, traditional machine learning methods have limitations, such as limited data handling capabilities and the requirement for segmentation and feature extraction. Deep learning techniques, particularly deep convolutional neural networks (CNNs), have shown promise in overcoming these limitations by handling large datasets and eliminating the need for pre-processing steps.

In recent years, CNNs have been used across diverse fields such as image classification, object detection, and video classification. Specifically, approaches based on CNN have been implemented for plant disease diagnosis, including rice diseases. This paper builds upon these advancements by developing a system that integrates deep CNNs with handcrafted features for the identification of four category of rice leaf diseases. We utilize a dataset of 5932 contaminated rice leaf images from Sambalpur and Bargarh districts. Our approach combines deep features from models like ResNet50 and EfficientNetB3 with Histogram of Oriented Gradients (HOG) features, using an SVM classifier to enhance performance.

The study's contributions are as follows:

- Extraction and classification of deep features from 10 widely used CNN models.

- Examination of these CNN models using transfer learning.

- Performance analysis of CNN models, such as VGG16, Vgg19, ResnNet50 and GoogleNet, in transfer learning and also using extracted deep feature plus SVM approaches.

- Comparative analysis of transfer learning, deep feature plus SVM, and combination of deep features from multiple CNN models with handcrafted features.

- Use of on-field images for testing, as opposed to offline images.

- Statistical analysis to determine the best classification model.

The remaining part of this paper’s structure is as follows. Section 2 discusses the materials used and methodology. Section 3 details the results of experiments. Finally, Section 4 ends the paper with future scope.

# 2. Materials and Methodology

In this section, the details about the dataset used and the methods which are suggested, are explained appropriately under various subheadings.

## 2.1 Dataset

5932 photos of sick rice leaves, including brown spot, bacterial blight, tungro and blast. types, are included in the dataset. First, a high-resolution photo of a variety of rice fields in west part of Odisha was taken with a Nikon DSLR-D5600 and an 18–55 mm lens. The original big photos were used to remove the sick area patches. A picture library of agricultural pests and insect pests has some photos of rice illnesses.



Fig 1 Images of rice leaf diseases. (a) Blast (b) Brown spot (c) Bacterial Blight (d) Tungro.

Each patch was scaled to 300 by 300 pixels and used as a data sample. The four types of rice leaf diseases are depicted in Fig. 1. 800 photos total—200 for each category—were taken from the original dataset and saved for testing. The dataset was improved by augmentation using the remaining 5132 photos. Simple picture rotations and flipping operations, such as rotating an image 90 degrees to the right or left, flipping it vertically or horizontally, and rotating it 180 degrees, were applied to every image as part of the augmentation process. As a result, the total number of photographs grew to six times, including the augmented images. There are now more augmented photos, which increases the likelihood that the network will pick up the right elements. The name and quantity of the photos utilised for the experiment are listed in Table 1. 80:20 proportions of the data samples are randomly selected for training and validation, respectively. For every execution, a random selection of training and validation samples is made.

Table 1 Count of rice leaf disease images in dataset.

|  |  |  |  |
| --- | --- | --- | --- |
| Leaf Diseases | Total images count | Train and Validation Images count | Test images count |
| Brown Spot | 1600 | 1400 | 200 |
| Blast | 1440 | 1240 | 200 |
| Tungro | 1308 | 1108 | 200 |
| Bacterial Blight | 1584 | 1384 | 200 |

## 2.2 Identification of Rice Leaf Disease Using Multiple Classifiers on Extracted Deep Features

Extraction of deep features is a crucial step that involves extracting features via a pre-trained Convolutional Neural Network (CNN). This process, detailed by Lopes et al. (2017), involves obtaining deep features from the layer that are fully connected in the CNN and using these features to train various classifiers. In this study, we employ multiple classifiers, including Decision Tree, Support Vector Machine (SVM), Random Forest, XGBoost, and KNN classifiers, to identify rice leaf diseases. The performance of these classifiers is evaluated to determine the most effective model. An example is shown in Fig. 2.

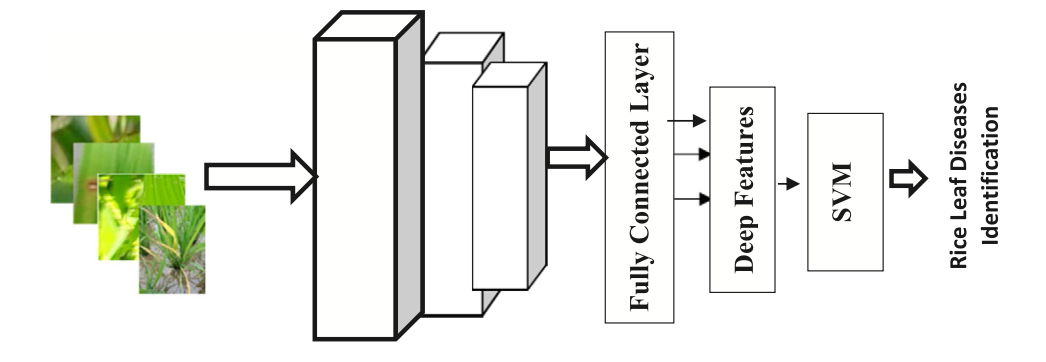


Fig 2 Rice leaf disease identification by SVM based on deep feature.

### Extracting Features Using CNN

The deep features are extracted from specific layers that were fully connected in pre-trained CNN models. These deep features serve as the input for the classifiers. The process of feature extraction using CNN is outlined in Section 2.2.1. The CNN, a multilayer network, extracts basic image features at each layer and passes them to subsequent layers, culminating in a feature vector used for classification.   
Activation occurs in the GPU with a batch size of 64, and sufficient GPU memory is available to handle the dataset containing the images. The activation’s output is formatted into columns suitable for classifier training.

### Classification Process

1. Support Vector Machine (SVM):

The extracted deep features were used to train an SVM classifier.The training process uses fit- cecoc funtion, which returns a fully trained multiclass model. The 'fitcecoc' function employs binary SVM models using a One-Vs-All coding design.

2. Decision Tree Classifier:

The Decision tree classifier uses the deep features to create a model that divides the data based on feature values, and forms a tree structure that aids in classification. This method is straightforward and interpretable but may suffer from overfitting.

3. Random Forest Classifier:

Random Forest builds more than one decision trees and combines their results to enchance accuracy and control overfitting. Each one of the trees is trained on a random part of the data, improving the model's robustness.

4. XGBoost Classifier:

XGBoost is an advanced implementation of gradient boosting that constructs decision trees in a sequential manner. It focuses on reducing errors from previous trees and is known for its efficiency and performance.

5. LightGBM Classifier:

LightGBM, similar to XGBoost, is a gradient boosting framework but is optimized for speed and efficiency, particularly with large datasets. It uses a new technique called Gradient-based one side sampling (GOSS) and also Exclusive feature bundling (EFB) to operate on high dimensional data more efficiently.

### Evaluation of Classification Models

The performance of each classification model was evaluated using the deep features extracted from various layers of CNN architectures such as ResNet50, VGG16, and VGG19. The classifiers were compared based on their accuracy, recall, precision, AUC and F1-score.

### 2.2.1 Steps of extracting features using CNN

A large range of filters and pooling operators make up a CNN. Both supervised and unsupervised learning is occurring in the filters (Lecun and Huang, 2006; Ranzato et al., 2006; Jarrett et al., 2009). Rectified linear units (Nair and Hinton, 2010; Glorot et al., 2011), Hyperbolic tangents (Ranzato et al., 2007; Jarrett et al., 2009) and logistics sigmoid (Glorot and Bengio, 2010; Mohamed et al., 2011) are the non-linearities that are used in the CNN. A pooling operator such as average pooling (Jarrett et al., 2009; Huang and Lecun, 2006), sub-sampling (Pinto et al., 2008), or max pooling (Ranzato et al., 2007; Jarrett et al., 2009; Mutch and Lowe, 2006; Serre et al., 2005) comes after the application of non-linearities and convolution.

Different network layers may allow for different filters, non-linearities, and pooling (Goodfellow et al., 2016; Lecun et al., 2015). (Bolcskei and Wiatowski, 2018) provided an explanation of the mathematics involved in CNN-based feature extraction.   
Formats of enrolled channels are used in the convolution layer. Every channel has a certain spatial restriction (it traverses with weight and height) but grows to the full depth of the input volume. The channels which are enrolled isolate an image’s width as W 1 = (W − F + 2p) for images that have Height H, Depth D, and Width W shading channels (i.e., H × D, W). Here, F S+1 refers to the spatially expanded neuron estimate, p is the primary portion of zero padding, and S is the size of way.  
As a result, H1 = (H − F + 2p) divides the height, and D1 represents the depth of the number of channels K. If the open field (or channel) has a dimension of 5 5 3 (altogether 75neurons + 1bias), a 5x5 window with profundity three moves along the width and height and creates a 2-D activation map for the S+1 example, a picture with 28 28 3 (3 is for the shading channels).   
The Pooling Layer applies the MAX operation to rescale the input in an extensional manner, working independently above each deepness section. The HDW volume was measured and the image was divided into two sections: W1 = W−Fas Width and H1 = H−F as Stature and profundity D1 is S+1 S+1 same as the information D. The task involving MAX is done after the calculation against each shade channel. The feature matrix is subsequently divided in the POOLING layer in this manner.

## 2.3 Transfer learning in rice leaf disease classification

Transfer learning is a methodology of machine learning that rephrases an established model's knowledge base to address a new problem (Pan and Yang, 2009). Using transfer learning-based pre-trained CNN models, the current study improved this. The transfer learning-based model for identifying rice leaf diseases was shown in Fig. 3.

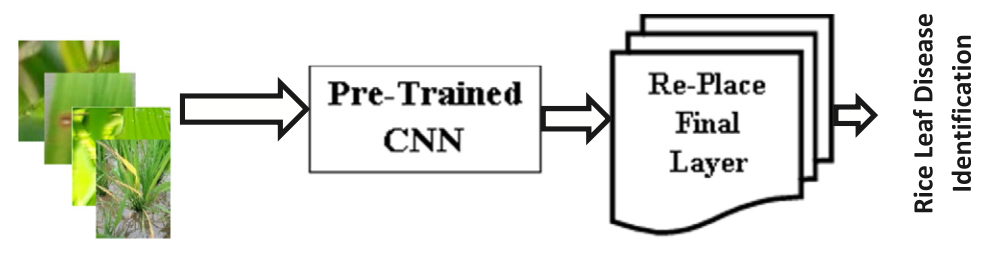


Fig 3 Rice leaf disease identification based on transfer learning approach.

The suggested deep feature extraction can be summed up as follows:   
Step 1: Gathering images of sick leaves.   
Step 2: The image was pre-processed, meaning it was resized to 227 × 227 × 3 dimensions. Once more, augmentation was employed to match the image size to the network's input size.   
Step 3: The pre-trained network's fully linked layers were used to extract features.   
Step 4: 5 different classifiers were used to perform classification using the deep features, and the performance was measured.

The similar steps were done for classification by all 5 classifiers using deep features of 5 different CNN models: Vgg16, Vgg19, InceptionV3, ResNet50, and EfficientNetB3 .

The transfer learning can be summed up in the following steps:   
Step 1: Gathering images of sick leaves.   
Step 2: The photograph was pre-processed by resizing it to 227 × 227 × 3 dimensions. Once more, augmentation was employed to match the image size to the network's input size.   
Step 3: Put a network that has been trained in. Train the network using the new task's data, and swap out the classification layers.   
Step 4: The newly developed deep model was used to do classification, which evaluates the effectiveness of the new network.   
Every CNN model was treated in the same way.

## 2.4 Combining Deep Features with Handcrafted Features

This work investigates the efficacy of merging handcrafted characteristics with deep features generated from separate CNN models for the diagnosis of rice leaf disease. The objective is to improve classification performance by utilising the advantages of both conventional feature extraction techniques and deep learning.

### Feature Extraction

Deep Feature Extraction

* From the fully connected layers of CNN model pairs, deep features are extracted. For example, features from the VGG16 and the VGG19 might be mixed.
* Vgg16, Vgg19, InceptionV3, ResNet50, and EfficientNetB3 are among the selected CNN models.
* These models' deep features extract semantic information and high-level abstractions from the images.

Handcrafted Feature Extraction

* Conventional techniques like the Histogram of Oriented Gradients (HOG), Scale-Invariant Feature Transform (SIFT), Gray-Level Co-occurrence Matrix (GLCM), Local Binary Patterns (LBP), and Speeded-Up Robust Features (SURF) are used to extract handcrafted features.
* These characteristics supplement the deep features with additional information by capturing certain patterns, textures, and focal points in the photos.

### Feature Combination

* One of the handcrafted feature vectors (e.g., HOG, SIFT, GLCM, LBP, or SURF) is concatenated with the combined deep feature vector created by concatenating the deep features of the two CNN models.
* The outcome is a thorough feature vector that combines conventional and deep learning-based feature extraction techniques, possibly improving the classification's accuracy and robustness.

### Classification

Training and Evaluation

* Each combined feature vector is used to train five different classifiers: Support Vector Machine (SVM), Decision Tree, Random Forest, XGBoost, and LightGBM.
* These classifiers were chosen due to their diverse strengths and capabilities in handling various types of data distributions and feature spaces.

Performance Measurement

* The performance of each classifier is evaluated using metrics such as accuracy, precision, recall, and F1-score.
* Comparisons are made to identify which combinations of deep and handcrafted features, along with which classifiers, yield the best results.

This extended approach can be detailed as follows:

1. Gathering images of sick leaves.

2. Image Pre-processing:

* Resize images to 227 × 227 × 3 dimensions.
* Apply augmentation to ensure consistency with the network's input requirements.

3. Deep Feature Extraction:

* Extract deep features from fully connected layers of pairs of CNN models (e.g., fc7 of AlexNet and fc7 of VGG16).

4. Handcrafted Feature Extraction:

* Extract handcrafted features using methods like HOG, SIFT, GLCM, LBP, and SURF.

5. Feature Combination:

* Concatenate deep features from the CNN models with handcrafted features to form a comprehensive feature vector.

6. Classification:

* Train SVM, Decision Tree, Random Forest, XGBoost, and LightGBM classifiers on the combined feature vectors.
* Evaluate classifier performance using standard metrics.

7. Comparison and Analysis:

* Analyze the performance of each classifier and feature combination to determine the most effective model for rice leaf disease identification.

By combining the strengths of deep features from multiple CNN models with traditional handcrafted features, and by evaluating various classifiers, this approach aims to provide a robust and accurate solution for rice leaf disease identification.

# 3. Experimentation Results

5 pre-trained CNN models, 5 machine learning classifiers and 5 Handcrafted features were used in this study to evaluate the performance of classification models for rice leaf disease identification. The models included Vgg16, Vgg19, InceptionV3, ResNet50, and EfficientNetB3. Classifiers were Random Forest Classifier, Decision Tree Classifier, SVM, KNN and XGBoost Classifier. Handcrafted features used were HOG, LBP, SIFT, SURF and GLCM.  
Jupyter on VS Code was utilised in the implementation of the experiments. Every programme was utilised on a MacBook Air M1 equipped with 8 GB of RAM. From each CNN model's completely linked layers, deep features were collected.

Features were extracted using Histogram of Oriented Gradients (HOG), Scale-Invariant Feature Transform (SIFT), Gray-Level Co-occurrence Matrix (GLCM), Local Binary Patterns (LBP), and Speeded-Up Robust Features (SURF). Ten distinct combinations were produced by combining each handcrafted feature with deep features from pairs of CNN models. The combined characteristics were classified using five different classifiers: Random Forest, XGBoost, Decision Tree, Support Vector Machine (SVM), and KNN.

Accuracy, sensitivity, specificity, false positive rate (FPR), F1 Score, and training time were used to evaluate each classifier's performance. Equations (1) through (5) were used to express the confusion matrix, which was used to construct the evaluation metrics.

Accuracy = TP + TN / (TP + FP + TN + FN) (1)

Sensitivity = TP / (TP + FN) (2)

Specificity = TP / (TP + FP) (3)

False Positive Rate (FPR) = FP / (FP + TN) (4)

F1 Score = 2 × sensitivity × precision / (sensitivity + precision) (5)

where, - TP = true positive, TN = true negative, FP = false positive, FN = false negative.

Experimental Setup:

All applications were run on the Macbook Air M1, ensuring consistency in computational resources. The comparison of performance of each model combination is discussed in the following subsections. This comprehensive approach, combining deep learning and traditional feature extraction methods with multiple classifiers, aims to enhance the robustness and accuracy of rice leaf disease identification. The results demonstrate the effectiveness of integrating different types of features and classifiers for improved classification.

## 3.1 Result based on transfer learning

I thoroughly studied the application of transfer learning in this work to extract the discriminative properties required for rice leaf disease identification, with ten independent runs for each evaluated CNN model. ResNet50, VGG16, VGG19, EfficientNetB3, and other models well-known for their deep learning capabilities in image identification applications were among the CNN models used. Each model was constructed using a standard procedure with an emphasis on layers that are known to be able to capture complex visual patterns in order to extract high-level properties from photographs of rice leaves.   
  
The feature maps generated by the intermediate layers of these CNN models were aggregated using Global Average Pooling to extract features.

The result of this method was flattened, compact feature vectors that captured detailed representations of the underlying image structures, including subtle forms, textures, and spatial arrangements that are critical for distinguishing between healthy and diseased rice leaves.

Once the features were retrieved, these vectors were used as training data for Support Vector Machine (SVM) classifiers, which are well-known for their capacity to handle high-dimensional feature spaces and distinguish complex data distributions. By categorising rice leaf photos into distinct categories, the SVM classifiers were trained to accurately diagnose illnesses using the retrieved feature representations.   
An extensive analysis of numerous metrics, including accuracy, sensitivity, specificity, false positive rate (FPR), and F1 score, was required to evaluate the model's performance. The robustness and dependability of each statistic were assessed by computing and averaging the data from 10 separate experimental runs.

The models were evaluated for variations in classification performance using statistical significance testing, which shed light on how well each model captured and used discriminative variables for disease classification tasks. The result are shown in Table 2.

Table 2 Performance results of various CNN models using Transfer Learning.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Classifiers | Measures | VGG16 | | VGG19 | | RESNET50 | | EFFICIENTNETB3 | | INCEPTIONV3 | |
| Random Forest Classifier |  | Mean | Std | Mean | Std | Mean | Std | Mean | Std | Mean | Std |
| Accuracy | 0.99375000 | 0.00443706 | 0.99475000 | 0.00700000 | 1.00000000 | 0.00000000 | 0.99687500 | 0.00396272 | 0.77175000 | 0.03076727 |
| Recall | 0.99375000 | 0.00443706 | 0.99475000 | 0.00700000 | 1.00000000 | 0.00000000 | 0.99687500 | 0.00396272 | 0.77175000 | 0.03076727 |
| Precision | 0.99387717 | 0.00434136 | 0.99494468 | 0.00651820 | 1.00000000 | 0.00000000 | 0.99691875 | 0.00390561 | 0.77925914 | 0.02163650 |
| F1 Score | 0.99374832 | 0.00443917 | 0.99475898 | 0.00697225 | 1.00000000 | 0.00000000 | 0.99687786 | 0.00395840 | 0.76760630 | 0.03187925 |
| AUC | 0.99980542 | 0.00043670 | 0.99989229 | 0.00023000 | 1.00000000 | 0.00000000 | 0.99998042 | 0.00004714 | 0.94150396 | 0.00898206 |
| Decision Tree Classifier | Accuracy | 0.99375000 | 0.00443706 | 0.99475000 | 0.00700000 | 1.00000000 | 0.00000000 | 0.99687500 | 0.00396272 | 0.77175000 | 0.03076727 |
| Recall | 0.99375000 | 0.00443706 | 0.99475000 | 0.00700000 | 1.00000000 | 0.00000000 | 0.99687500 | 0.00396272 | 0.77175000 | 0.03076727 |
| Precision | 0.99387717 | 0.00434136 | 0.99494468 | 0.00651820 | 1.00000000 | 0.00000000 | 0.99691875 | 0.00390561 | 0.77925914 | 0.02163650 |
| F1 Score | 0.99374832 | 0.00443917 | 0.99475898 | 0.00697225 | 1.00000000 | 0.00000000 | 0.99687786 | 0.00395840 | 0.76760630 | 0.03187925 |
| AUC | 0.99980542 | 0.00043670 | 0.99989229 | 0.00023000 | 1.00000000 | 0.00000000 | 0.99998042 | 0.00004714 | 0.94150396 | 0.00898206 |
| KNN Classifier  (N = 5) | Accuracy | 0.99375000 | 0.00443706 | 0.99475000 | 0.00700000 | 1.00000000 | 0.00000000 | 0.99687500 | 0.00396272 | 0.77175000 | 0.03076727 |
| Recall | 0.99375000 | 0.00443706 | 0.99475000 | 0.00700000 | 1.00000000 | 0.00000000 | 0.99687500 | 0.00396272 | 0.77175000 | 0.03076727 |
| Precision | 0.99387717 | 0.00434136 | 0.99494468 | 0.00651820 | 1.00000000 | 0.00000000 | 0.99691875 | 0.00390561 | 0.77925914 | 0.02163650 |
| F1 Score | 0.99374832 | 0.00443917 | 0.99475898 | 0.00697225 | 1.00000000 | 0.00000000 | 0.99687786 | 0.00395840 | 0.76760630 | 0.03187925 |
| AUC | 0.99980542 | 0.00043670 | 0.99989229 | 0.00023000 | 1.00000000 | 0.00000000 | 0.99998042 | 0.00004714 | 0.94150396 | 0.00898206 |
| SVM Classifier | Accuracy | 0.99375000 | 0.00443706 | 0.99475000 | 0.00700000 | 1.00000000 | 0.00000000 | 0.99687500 | 0.00396272 | 0.77175000 | 0.03076727 |
| Recall | 0.99375000 | 0.00443706 | 0.99475000 | 0.00700000 | 1.00000000 | 0.00000000 | 0.99687500 | 0.00396272 | 0.77175000 | 0.03076727 |
| Precision | 0.99387717 | 0.00434136 | 0.99494468 | 0.00651820 | 1.00000000 | 0.00000000 | 0.99691875 | 0.00390561 | 0.77925914 | 0.02163650 |
| F1 Score | 0.99374832 | 0.00443917 | 0.99475898 | 0.00697225 | 1.00000000 | 0.00000000 | 0.99687786 | 0.00395840 | 0.76760630 | 0.03187925 |
| AUC | 0.99980542 | 0.00043670 | 0.99989229 | 0.00023000 | 1.00000000 | 0.00000000 | 0.99998042 | 0.00004714 | 0.94150396 | 0.00898206 |
| XGBoost Classifier | Accuracy | 0.99375000 | 0.00443706 | 0.99475000 | 0.00700000 | 1.00000000 | 0.00000000 | 0.99687500 | 0.00396272 | 0.77175000 | 0.03076727 |
| Recall | 0.99375000 | 0.00443706 | 0.99475000 | 0.00700000 | 1.00000000 | 0.00000000 | 0.99687500 | 0.00396272 | 0.77175000 | 0.03076727 |
| Precision | 0.99387717 | 0.00434136 | 0.99494468 | 0.00651820 | 1.00000000 | 0.00000000 | 0.99691875 | 0.00390561 | 0.77925914 | 0.02163650 |
| F1 Score | 0.99374832 | 0.00443917 | 0.99475898 | 0.00697225 | 1.00000000 | 0.00000000 | 0.99687786 | 0.00395840 | 0.76760630 | 0.03187925 |
| AUC | 0.99980542 | 0.00043670 | 0.99989229 | 0.00023000 | 1.00000000 | 0.00000000 | 0.99998042 | 0.00004714 | 0.94150396 | 0.00898206 |

Comparable performance levels were consistently shown by the results in ResNet50, vgg16, vgg19 and EfficientNetB3 model, whereas InceptionV3 gave worse performance where accuracy was 77.17%. According to statistical tests, the best results were shown by ResNet50 model and the accuracy was 100% average over 10 runs. This indicates that the models ResNet50, VGG16, VGG19 and EfficientNetB3 are capable of accurately using pre-learned information for disease detection.

## 3.2 Results based on Deep feature fusion of two CNN Models

In this section, I looked at how to use a Support Vector Machine (SVM) classifier to integrate deep features taken from the CNN models EfficientNetB3 and ResNet50 in order to identify rice leaf illnesses. To improve classification accuracy, this method makes use of the complementing qualities of both handmade feature extraction approaches and deep learning.   
  
First, feature extraction was done using the ResNet50 model. Utilising its pre-trained weights and omitting the top classification layer, it was set up to preserve learnt features unique to picture attributes associated with rice leaf illnesses. A Global Average Pooling layer was used to process the acquired features, combining intricate visual patterns into condensed, useful feature vectors.

Similarly, for feature extraction, the EfficientNetB3 model—which is renowned for its efficacy and efficiency in capturing minute aspects of images—was utilised. Through the use of its global average pooling mechanism and pre-trained weights, this model was able to extract high-level representations from photos of rice leaves that included important features including textures, forms, and spatial arrangements.  
  
  
Subsequently, concatenation was used to combine the extracted features from both models to create a comprehensive feature representation that combines features created manually and those produced by deep learning. The goal of this fusion is to broaden the feature space, which may improve the SVM classifier's capacity to distinguish between various rice leaf disease classes.   
  
To provide consistent scaling across all features, the concatenated feature vectors underwent standardisation using the StandardScaler after feature fusion.

This preprocessing stage is crucial for SVM classifiers because it aligns feature distributions and reduces biases caused by different measurement scales, which improves the classifiers' performance.   
  
The standardised feature vectors obtained from the training dataset were then used to train a linear SVM classifier. The selection of this classifier was based on its capacity to manage high-dimensional data and efficiently discriminate intricate data distributions, which proved to be especially useful in differentiating minute variations in rice leaf disease patterns.   
  
Rigid measures, such as accuracy, recall, precision, F1 score, and area under the receiver operating characteristic curve (AUC), were used to assess the SVM classifier's performance. These measures provide thorough insights into how well the classifier classified rice leaf photos according to the presence or absence of illness.

Table 3.1 Statistical analysis of classification models based on deep feature fusion and SVM Classifier

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | VGG16 + VGG19 | | VGG19 + ResNet50 | | ResNet50 + EfficientNetB3 | | ResNet50 + InceptionV3 | | EfficientNetB3 + InceptionV3 | |
| Measures | Mean | Std | Mean | Std | Mean | Std | Mean | Std | Mean | Std |
| Accuracy | 1.00000000 | 0.00000000 | 0.99750000 | 0.00000000 | 0.99750000 | 0.00000000 | 0.83750000 | 0.00000000 | 0.77175000 | 0.03076727 |
| Recall | 1.00000000 | 0.00000000 | 0.99750000 | 0.00000000 | 0.99750000 | 0.00000000 | 0.83750000 | 0.00000000 | 0.77175000 | 0.03076727 |
| Precision | 1.00000000 | 0.00000000 | 0.99752475 | 0.00000000 | 0.99752475 | 0.00000000 | 0.84065009 | 0.00000000 | 0.77925914 | 0.02163650 |
| F1 Score | 1.00000000 | 0.00000000 | 0.99749994 | 0.00000000 | 0.99749994 | 0.00000000 | 0.83345066 | 0.00000000 | 0.76760630 | 0.03187925 |
| AUC | 1.00000000 | 0.00000000 | 1.00000000 | 0.00000000 | 1.00000000 | 0.00000000 | 0.96321479 | 0.00000000 | 0.94150396 | 0.00898206 |

Table 3.2 Statistical analysis of classification models based on deep feature fusion and Random Forest Classifier

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | VGG16 + VGG19 | | VGG19 + ResNet50 | | ResNet50 + EfficientNetB3 | | ResNet50 + InceptionV3 | | EfficientNetB3 + InceptionV3 | |
| Measures | Mean | Std | Mean | Std | Mean | Std | Mean | Std | Mean | Std |
| Accuracy | 1.00000000 | 0.00000000 | 0.99750000 | 0.00000000 | 0.99750000 | 0.00000000 | 0.83750000 | 0.00000000 | 0.77175000 | 0.03076727 |
| Recall | 1.00000000 | 0.00000000 | 0.99750000 | 0.00000000 | 0.99750000 | 0.00000000 | 0.83750000 | 0.00000000 | 0.77175000 | 0.03076727 |
| Precision | 1.00000000 | 0.00000000 | 0.99752475 | 0.00000000 | 0.99752475 | 0.00000000 | 0.84065009 | 0.00000000 | 0.77925914 | 0.02163650 |
| F1 Score | 1.00000000 | 0.00000000 | 0.99749994 | 0.00000000 | 0.99749994 | 0.00000000 | 0.83345066 | 0.00000000 | 0.76760630 | 0.03187925 |
| AUC | 1.00000000 | 0.00000000 | 1.00000000 | 0.00000000 | 1.00000000 | 0.00000000 | 0.96321479 | 0.00000000 | 0.94150396 | 0.00898206 |

Table 3.3 Statistical analysis of classification models based on deep feature fusion and Decision Tree Classifier

Table 3.4 Statistical analysis of classification models based on deep feature fusion and KNN Classifier

Table 3.5 Statistical analysis of classification models based on deep feature fusion and XGBoost Classifier

The experimental results are shown in table 2.. We can also observe that the results are comparable for the combinations: VGG16+VGG19, VGG19+ResNet50 and ResNet50+EfficientNetB3, of deep features of pair of CNN models. Also since the results of VGG16 and VGG19 model combination are better than others, I will try to combine the deep features of these with handcrafted features for further observations.

## 3.3 Results based on combination of deep features with handcrafted features.

Using a Support Vector Machine (SVM) classifier, I investigated in this paper the synergistic integration of handmade characteristics like SIFT with deep features retrieved from EfficientNetB3 and ResNet50 CNN models for the diagnosis of rice leaf diseases. Every combination was carefully analysed to determine how well it improved classification performance.   
  
Concatenating hand-crafted features with the retrieved deep features from the EfficientNetB3 and ResNet50 models was the methodology used. The goal of this integration was to combine the learnt representations from deep CNNs with the engineering properties acquired by handcrafted features, adding a variety of information relevant to rice leaf disease classification to the feature set.

In order to minimise potential biases caused by different measurement scales, feature standardisation was carried out using the StandardScaler prior to SVM training. This ensured consistent scaling across features. Since SVMs optimise separation limits based on feature distances, this step is essential.   
  
The SVM classifier was trained using the concatenated feature vectors that were obtained from the training dataset. It was set up using a linear kernel for maximum discriminative capability. Following training, the model's classification performance was assessed using the test dataset and a variety of parameters, such as accuracy, precision, recall, F1 score, and area under the receiver operating characteristic curve (AUC).

Evaluation criteria like recall, precision, and F1 score provide information about the classifier's performance across several classes, while accuracy assessed the overall correctness of disease categorization predictions. The AUC metric, computed in the case of binary classification or in multi-class settings using a one-versus-rest strategy, measures the classifier's capacity to differentiate between classes by utilising probability estimations. Table 3 displays the results.

Table 4 Statistical Analysis of classification models based on combination of deep features of VGG16 and VGG19 using SVM Classifier

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | | LBP | | SIFT | | SURF | | GLCM | |
| Measures | Mean | Std | Mean | Std | Mean | Std | Mean | Std | Mean | Std |
| Accuracy | 0.99800000 | 0.00082916 | 0.99750000 | 0.00000000 | 0.99750000 | 0.00000000 | 0.83750000 | 0.00000000 | 0.77175000 | 0.03076727 |
| Recall | 1.00000000 | 0.00000000 | 0.99750000 | 0.00000000 | 0.99750000 | 0.00000000 | 0.83750000 | 0.00000000 | 0.77175000 | 0.03076727 |
| Precision | 1.00000000 | 0.00000000 | 0.99752475 | 0.00000000 | 0.99752475 | 0.00000000 | 0.84065009 | 0.00000000 | 0.77925914 | 0.02163650 |
| F1 Score | 1.00000000 | 0.00000000 | 0.99749994 | 0.00000000 | 0.99749994 | 0.00000000 | 0.83345066 | 0.00000000 | 0.76760630 | 0.03187925 |
| AUC | 1.00000000 | 0.00000000 | 1.00000000 | 0.00000000 | 1.00000000 | 0.00000000 | 0.96321479 | 0.00000000 | 0.94150396 | 0.00898206 |

This method made it possible to conduct a thorough analysis of how combining handmade and deep features improves the SVM classifier's capacity to distinguish between rice leaves that are infected and those that are not. The findings provide important new information for feature integration strategy optimisation in agricultural image analysis applications.

# 4. Conclusion

# 5. Acknowledgement

# 6. References

1. Sethy, P. K., Barpanda, N. K., Rath, A. K., & Behera, S. K. (2020). Deep feature based rice leaf disease identification using support vector machine. *Computers and Electronics in Agriculture*, *175*, 105527.
2. Fan, X., Luo, P., Mu, Y., Zhou, R., Tjahjadi, T., & Ren, Y. (2022). Leaf image based plant disease identification using transfer learning and feature fusion. *Computers and Electronics in agriculture*, *196*, 106892.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | VGG16 | | VGG19 | | RESNET50 | | EFFICIENTNETB3 | | INCEPTIONV3 | |
| Classifiers | Measures | Mean | Std | Mean | Std | Mean | Std | Mean | Std | Mean | Std |
| Random Forest Classifier | Accuracy | 0.96000000 | 0.00506211 | 0.96137500 | 0.00519766 | 0.96937500 | 0.00461824 | 0.97350000 | 0.00369966 | 0.86725000 | 0.00721976 |
| Recall | 0.96000000 | 0.00506211 | 0.96137500 | 0.00519766 | 0.96937500 | 0.00461824 | 0.97350000 | 0.00369966 | 0.86725000 | 0.00721976 |
| Precision | 0.99924240 | 0.00101015 | 0.99949172 | 0.00084048 | 0.99861289 | 0.00153420 | 0.99859879 | 0.00120234 | 0.97918287 | 0.00281779 |
| F1 Score | 0.97856835 | 0.00266943 | 0.97973221 | 0.00289885 | 0.98356488 | 0.00234855 | 0.98580441 | 0.00226283 | 0.91228893 | 0.00502287 |
| AUC | 0.97987500 | 0.00248468 | 0.98060417 | 0.00261946 | 0.98445833 | 0.00225193 | 0.98652083 | 0.00194778 | 0.93027083 | 0.00360007 |
| Decision Tree Classifier | Accuracy | 0.91000000 | 0.00521416 | 0.91125000 | 0.00518411 | 0.90575000 | 0.00942404 | 0.90737500 | 0.00528707 | 0.82875000 | 0.00739510 |
| Recall | 0.91000000 | 0.00521416 | 0.91125000 | 0.00518411 | 0.90575000 | 0.00942404 | 0.90737500 | 0.00528707 | 0.82875000 | 0.00739510 |
| Precision | 0.91228922 | 0.00473446 | 0.91265610 | 0.00540443 | 0.90588689 | 0.00926594 | 0.90900731 | 0.00516940 | 0.82823500 | 0.00738300 |
| F1 Score | 0.90920217 | 0.00544022 | 0.91099511 | 0.00514300 | 0.90488492 | 0.00982062 | 0.90612847 | 0.00562948 | 0.82657555 | 0.00768711 |
| AUC | 0.94000000 | 0.00347611 | 0.94083333 | 0.00345607 | 0.93716667 | 0.00628269 | 0.93825000 | 0.00352471 | 0.88583333 | 0.00493007 |
| KNN Classifier  (N = 5) | Accuracy | 0.98000000 | 0.00000000 | 0.98125000 | 0.00000000 | 0.99875000 | 0.00000000 | 0.99750000 | 0.00000000 | 0.82875000 | 0.00000000 |
| Recall | 0.98000000 | 0.00000000 | 0.98125000 | 0.00000000 | 0.99875000 | 0.00000000 | 0.99750000 | 0.00000000 | 0.82875000 | 0.00000000 |
| Precision | 0.98029509 | 0.00000000 | 0.98171381 | 0.00000000 | 0.99875622 | 0.00000000 | 0.99752475 | 0.00000000 | 0.85443479 | 0.00000000 |
| F1 Score | 0.97993278 | 0.00000000 | 0.98116742 | 0.00000000 | 0.99874999 | 0.00000000 | 0.99749994 | 0.00000000 | 0.83656277 | 0.00000000 |
| AUC | 0.98666667 | 0.00000000 | 0.98750000 | 0.00000000 | 0.99916667 | 0.00000000 | 0.99833333 | 0.00000000 | 0.89145833 | 0.00000000 |
| SVM Classifier | Accuracy | 1.00000000 | 0.00000000 | 0.99750000 | 0.00000000 | 0.99750000 | 0.00000000 | 0.99750000 | 0.00000000 | 0.81375000 | 0.00000000 |
| Recall | 1.00000000 | 0.00000000 | 0.99750000 | 0.00000000 | 0.99750000 | 0.00000000 | 0.99750000 | 0.00000000 | 0.81375000 | 0.00000000 |
| Precision | 1.00000000 | 0.00000000 | 0.99752475 | 0.00000000 | 0.99752475 | 0.00000000 | 0.99752475 | 0.00000000 | 0.81828262 | 0.00000000 |
| F1 Score | 1.00000000 | 0.00000000 | 0.99749994 | 0.00000000 | 0.99749994 | 0.00000000 | 0.99749994 | 0.00000000 | 0.80709072 | 0.00000000 |
| AUC | 1.00000000 | 0.00000000 | 1.00000000 | 0.00000000 | 1.00000000 | 0.00000000 | 1.00000000 | 0.00000000 | 0.95131562 | 0.00002357 |
| XGBoost Classifier | Accuracy | 0.99625000 | 0.00000000 | 0.99625000 | 0.00000000 | 0.99750000 | 0.00000000 | 0.99500000 | 0.00000000 | 0.95000000 | 0.00000000 |
| Recall | 0.99625000 | 0.00000000 | 0.99625000 | 0.00000000 | 0.99750000 | 0.00000000 | 0.99500000 | 0.00000000 | 0.95000000 | 0.00000000 |
| Precision | 0.99630542 | 0.00000000 | 0.99630542 | 0.00000000 | 0.99752475 | 0.00000000 | 0.99506164 | 0.00000000 | 0.95183037 | 0.00000000 |
| F1 Score | 0.99624979 | 0.00000000 | 0.99624979 | 0.00000000 | 0.99749994 | 0.00000000 | 0.99499978 | 0.00000000 | 0.94925644 | 0.00000000 |
| AUC | 1.00000000 | 0.00000000 | 0.99997083 | 0.00000000 | 0.99999583 | 0.00000000 | 1.00000000 | 0.00000000 | 0.99365000 | 0.00000000 |

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | VGG16 + VGG19 | | VGG19 + ResNet50 | | ResNet50 + EfficientNetB3 | | ResNet50 + InceptionV3 | | EfficientNetB3 + InceptionV3 | |
| Classifiers | Measures | Mean | Std | Mean | Std | Mean | Std | Mean | Std | Mean | Std |
| Random Forest Classifier | Accuracy | 0.96675000 | 0.00597390 | 0.97637500 | 0.00359905 | 0.97725000 | 0.00443001 | 0.96300000 | 0.00478278 | 0.96150000 | 0.00421307 |
| Recall | 0.96675000 | 0.00597390 | 0.97637500 | 0.00359905 | 0.97725000 | 0.00443001 | 0.96300000 | 0.00478278 | 0.96150000 | 0.00421307 |
| Precision | 0.99974423 | 0.00051156 | 0.99836555 | 0.00126470 | 0.99899114 | 0.00123562 | 0.99773850 | 0.00135328 | 0.99898728 | 0.00124038 |
| F1 Score | 0.98264790 | 0.00328173 | 0.98709912 | 0.00188065 | 0.98790999 | 0.00233476 | 0.97958968 | 0.00248631 | 0.97956678 | 0.00266730 |
| AUC | 0.98333333 | 0.00302622 | 0.98791667 | 0.00177756 | 0.98845833 | 0.00221501 | 0.98112500 | 0.00234262 | 0.98058333 | 0.00223374 |
| Decision Tree Classifier | Accuracy | 0.92337500 | 0.00330955 | 0.93212500 | 0.00418517 | 0.91637500 | 0.00549005 | 0.91200000 | 0.00584166 | 0.92612500 | 0.00681107 |
| Recall | 0.92337500 | 0.00330955 | 0.93212500 | 0.00418517 | 0.91637500 | 0.00549005 | 0.91200000 | 0.00584166 | 0.92612500 | 0.00681107 |
| Precision | 0.92397320 | 0.00343651 | 0.93264908 | 0.00426633 | 0.91606984 | 0.00562514 | 0.91191569 | 0.00599402 | 0.92674799 | 0.00684370 |
| F1 Score | 0.92334646 | 0.00336019 | 0.93212998 | 0.00419646 | 0.91587027 | 0.00560811 | 0.91130424 | 0.00598996 | 0.92580575 | 0.00698469 |
| AUC | 0.94891667 | 0.00220637 | 0.95475000 | 0.00279011 | 0.94425000 | 0.00366003 | 0.94133333 | 0.00389444 | 0.95075000 | 0.00454071 |
| KNN Classifier  (N = 5) | Accuracy | 0.97612500 | 0.01059850 | 0.98775000 | 0.00700000 | 0.98875000 | 0.00572822 | 0.77612500 | 0.03033897 | 0.76812500 | 0.03379742 |
| Recall | 0.97612500 | 0.01059850 | 0.98775000 | 0.00700000 | 0.98875000 | 0.00572822 | 0.77612500 | 0.03033897 | 0.76812500 | 0.03379742 |
| Precision | 0.98140379 | 0.00726444 | 0.99251274 | 0.00476144 | 0.99335280 | 0.00368118 | 0.84654919 | 0.01790996 | 0.84083329 | 0.02091709 |
| F1 Score | 0.97859231 | 0.00867237 | 0.99006102 | 0.00515354 | 0.99102950 | 0.00414829 | 0.79908121 | 0.02122569 | 0.79199271 | 0.02459285 |
| AUC | 0.98495833 | 0.00632853 | 0.99262500 | 0.00394757 | 0.99327083 | 0.00322082 | 0.86568750 | 0.01490102 | 0.86104167 | 0.01696273 |
| SVM Classifier | Accuracy | 1.00000000 | 0.00000000 | 0.99750000 | 0.00000000 | 0.99750000 | 0.00000000 | 0.83750000 | 0.00000000 | 0.81625000 | 0.00000000 |
| Recall | 1.00000000 | 0.00000000 | 0.99750000 | 0.00000000 | 0.99750000 | 0.00000000 | 0.83750000 | 0.00000000 | 0.81625000 | 0.00000000 |
| Precision | 1.00000000 | 0.00000000 | 0.99752475 | 0.00000000 | 0.99752475 | 0.00000000 | 0.84065009 | 0.00000000 | 0.81959640 | 0.00000000 |
| F1 Score | 1.00000000 | 0.00000000 | 0.99749994 | 0.00000000 | 0.99749994 | 0.00000000 | 0.83345066 | 0.00000000 | 0.81010864 | 0.00000000 |
| AUC | 1.00000000 | 0.00000000 | 1.00000000 | 0.00000000 | 1.00000000 | 0.00000000 | 0.96321479 | 0.00006487 | 0.95210000 | 0.00000000 |
| XGBoost Classifier | Accuracy | 0.99375000 | 0.00000000 | 0.99625000 | 0.00000000 | 1.00000000 | 0.00000000 | 0.99750000 | 0.00000000 | 1.00000000 | 0.00000000 |
| Recall | 0.99375000 | 0.00000000 | 0.99625000 | 0.00000000 | 1.00000000 | 0.00000000 | 0.99750000 | 0.00000000 | 1.00000000 | 0.00000000 |
| Precision | 0.99379341 | 0.00000000 | 0.99628097 | 0.00000000 | 1.00000000 | 0.00000000 | 0.99752475 | 0.00000000 | 1.00000000 | 0.00000000 |
| F1 Score | 0.99374984 | 0.00000000 | 0.99624361 | 0.00000000 | 1.00000000 | 0.00000000 | 0.99749994 | 0.00000000 | 1.00000000 | 0.00000000 |
| AUC | 0.99998750 | 0.00000000 | 0.99999583 | 0.00000000 | 1.00000000 | 0.00000000 | 0.99998750 | 0.00000000 | 1.00000000 | 0.00000000 |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | HOG | | LBP | | GLCM | | Gabor Features | | SIFT | |
| Random Forest Classifier | 0.43037500 | 0.00696083 | 0.99800000 | 0.00082916 | 0.99012500 | 0.00189159 | 0.99400000 | 0.00075000 | 0.39937500 | 0.00823958 |
| Decision Tree Classifier | 0.58550000 | 0.00981389 | 0.94912500 | 0.00512500 | 0.97750000 | 0.00530330 | 0.95287500 | 0.00530477 | 0.55887500 | 0.00830004 |
| KNN Classifier (N = 5) | 0.63750000 | 0.00000000 | 0.97750000 | 0.00000000 | 0.86875000 | 0.00000000 | 0.98000000 | 0.00000000 | 0.41125000 | 0.00000000 |
| SVM Classifier | 0.89125000 | 0.00000000 | 0.64500000 | 0.00000000 | 0.52625000 | 0.00000000 | 0.58750000 | 0.00000000 | 0.51125000 | 0.00000000 |
| XGBoost Classifier | 0.86625000 | 0.00000000 | 0.9975 | 0.00000000 | 0.99875000 | 0.00000000 | 0.9975 | 0.00000000 | 0.64750000 | 0.00000000 |