Acadlore Transactions on AI and Machine Learning
https://www.acadlore.com/journals/ATAIML

Development of a Machine Learning-Driven Web Platform for Automated Identification of Rice Insect Pests

Samuel N. John[1], Nasiru A. Musa[1], Joshua S. Mommoh[2*], Etinosa Noma-Osaghe[3], Ukeme I. Udioko[1], James L. Obetta[4]

[1] Department of Electrical Electronic Engineering, Nigerian Defence Academy, 800281 Kaduna, Nigeria
[2] Department of Software Engineering, Mudiame University Irrua, 310112 Edo, Nigeria
[3] Department of Electrical and Electronics Engineering, Olabisi Onabanjo University, Ogun 120107, Nigeria
[4] Department of Information and Communication Technology, Air Force Institution of Technology, 800282 Kaduna, Nigeria

[*] Correspondence: Joshua S. Mommoh (mommoh.joshua@mudiameuniversity.edu.ng)

Received: 03-26-2025 **Revised:** 05-12-2025 **Accepted:** 05-18-2025

Citation: S. N. John, N. A. Musa, J. S. Mommoh, E. Noma-Osaghe, U. I. Udioko, and J. L. Obetta, "Development of a machine learning-driven web platform for automated identification of rice insect pests," *Acadlore Trans. Mach. Learn.*, vol. 4, no. 3, pp. 137–156, 2025. https://doi.org/10.56578/ataiml040301.

Abstract: An advanced machine learning (ML)-driven web platform was developed and deployed to automate the identification of rice insect pests, addressing limitations associated with traditional pest detection methods and conventional ML algorithms. Historically, pest identification in rice cultivation has relied on expert evaluation of pest species and their associated crop damage, a process that is labor-intensive, time-consuming, and prone to inaccuracies, particularly in the misclassification of pest species. In this study, a subset of the publicly available IP102 benchmark dataset, consisting of 7,736 images across 12 rice pest categories, was curated for model training and evaluation. Two classification models—a Support Vector Machine (SVM) and a deep Convolutional Neural Network (CNN) based on the Inception_ResNetV2 architecture—were implemented and assessed using standard performance metrics. Experimental results demonstrated that the Inception_ResNetV2 model significantly outperformed SVM, achieving an accuracy of 99.97%, a precision of 99.46%, a recall of 99.81%, and an F1-score of 99.53%. Owing to its superior performance, the Inception_ResNetV2 model was integrated into a web-based application designed for real-time pest identification. The deployed system exhibited an average response time of 5.70 seconds, representing a notable improvement in operational efficiency and usability over previous implementations. The results underscore the potential of artificial intelligence in transforming agricultural practices by enabling accurate, scalable, and timely pest diagnostics, thereby enhancing pest management strategies, mitigating crop losses, and supporting global food security initiatives.

Keywords: Automated rice pest identification; Machine learning; Convolutional neural network; Support vector machine; IP102 dataset; Agriculture

1 Introduction

Agricultural productivity receives direct support from insect pest management. Its impacts can be seen on grain production, development in agriculture, as well as on the economy of a farmer. Detecting insect pests within agricultural settings is a process that is rather complex [1]. Rice is one of the crops that is vital regarding food security globally. Rice crops face high vulnerability to diseases that cause major decreases in both yield and quality. The stability of global food supplies faces significant risk. To maintain rice production at peak efficiency, implementing disease prevention and effective pest control strategies is vital [2, 3]. Accurate and timely diagnosis of plant diseases is at the core of effective disease management, as it enables immediate application of efficient pesticide control measures. Crop insect pests cause an annual loss of 40–50% of yields, posing an immediate threat to food security on a global level [4]. Rice crops are particularly vulnerable to destructive insect pests such as the rice stem borer, brown plant hopper, rice gall midge, and rice water weevil, which can significantly reduce yield if not detected and controlled early. Rice cultivation in Nigeria suffers up to 75% yield reduction because of insect pest damage based on statistics [5]. Severe food shortages, along with increased rice importation and higher market prices, result from these

losses, which create economic instability for both farmers and the country. There is a need for a system that can accurately detect and identify insect pests in rice and provide early warning for farmers to take necessary measures to mitigate the damage caused by these pests. Pests and diseases constitute a major problem for agricultural production and a threat to food security.

Manual diagnosis of rice crop diseases is still the most prevalent method at present, relying primarily on visual observation of disease symptoms. The most prevalent method of diagnosing rice crop diseases today is manual inspection by visually observing disease symptoms [6, 7]. This method faces multiple difficulties, which include the limited availability of trained agricultural workers capable of providing quick and accurate diagnoses. The current constraints indicate an immediate requirement for an automated and scalable disease diagnosis method that also manages pest infestations more efficiently.

With the advent of smart agriculture, ML has been integrated in various fields that need to process data holistically. Smart systems have various elements that enable local data fetching, analysis, monitoring, and prediction [8, 9]. The application of ML in agriculture has greatly assisted in identifying optimal methods of farming and crop management, improving crop quality and total agricultural output. Particularly, extensive farming has immensely gained from the capabilities of ML's strong automation and computation. Some of the major uses of ML in agriculture are forecasting agricultural output, detecting and identifying insect pests, and streamlining pest management practices [10–12]. Massive research has proved that systems based on ML can potentially be used to diagnose and solve problems that hamper agricultural efficiency. These advanced technologies support various phases of agricultural production, including inspection, transportation, and storage of farm produce. Based on the insights from ML, agricultural experts can make more efficient and informed decisions regarding pest control and crop management.

Recent advancements in neural networks have obtained promising results in addressing complex issues in agriculture such as feature extraction, image segmentation, image classification, and other vision tasks. CNNs have particularly excelled at classifying images [13–15]. Their ability to decipher high amounts of visual data has seen them become one of the most successful means of disease and pest identification in plants. Through CNNs, researchers and agriculturalists can develop automatic disease identification systems that surpass traditional manual methods in both speed and accuracy. This research seeks to explore two ML algorithms, namely SVM and the Inception_ResNetV2 CNN model, for the detection of rice diseases, helping to mitigate losses associated with rice cultivation. Finally, this study concentrates on the web-based implementation of the disease detection model, guaranteeing that farmers and agricultural specialists have real-time access. Besides, the users can also upload photos of rice crops, obtain real-time illness diagnosis, and see prescribed management practices from an online interface. The present study extends scalability, convenience, and impact to rice disease detection by synergistically integrating cloud computing with ML to contribute to sustainable farming practices.

2 Literature Review

ML and deep learning (DL) techniques have been employed by various researchers to identify various rice insect pest diseases. The techniques, along with evaluation outcomes, are presented in this section.

Doan [7] introduced a novel approach that combines Power Mean SVM (PmSVM) with EfficientNet fine-tuning to classify insect pest images on a wide scale. EfficientNet models were first adjusted and re-trained using fresh image datasets of insect pests. The ML classifier was then constructed using the features that were recovered from the EfficientNet models. During the classification step of the network, PmSVM, a non-linear classifier, replaced the conventional Softmax function. The suggested method's classification accuracy for IP102 large insect pest datasets is 72.31% after fine-tuning EfficientNets and PmSVM. The model considered was unsuitable to handle large datasets; hence, the accuracy was low.

Deng et al. [16] built a smartphone application for automatic rice disease diagnosis. The method was built using DL, trained on a large dataset of 33,026 images of six types of rice diseases, which are leaf blast, false smut, neck blast, sheath blight, bacterial stripe disease, and brown spot. The highlight of this method was an ensemble model, which combined multiple submodels. For evaluating its performance, the ensemble model was evaluated using an independent image set. The result indicated that DenseNet-121, SE-ResNet-50, and ResNeSt-50 were the three best-performing submodels based on parameters such as learning rate, precision, recall, and accuracy in recognizing diseases. Consequently, these three submodels were selected and combined to devise the final ensemble model. The integration of these submodels made it possible to minimize confusion among similar kinds of diseases, thereby reducing the possibility of wrong diagnosis. Used to diagnose the six rice diseases, the ensemble model achieved an overall accuracy of 91%. This performance was considered reasonably high, given the visual similarities among certain rice diseases. However, the model could only identify six rice diseases.

Similarly, Bhartiya et al. [17] proposed a novel approach for rice disease detection using ML techniques. Significant diseases affecting different rice leaves were classified using different machine-learning methods. The rice leaf disease images were taken, from which features of their corresponding diseases were initially extracted. Finally, the features were classified by applying different ML techniques to classify the images as per the methodology explained in the

content. It was found that the quadratic SVM classifier gave an accuracy of around 81.8%. Further, the shape features such as area, roundness, and area-to-lesion ratio were used to differentiate various types of rice diseases. Further, the developed model could not actually determine the effectiveness of the model in real time. Furthermore, Naresh Kumar and Sakthivel [18] proposed a new detection method of rice diseases based on the Fusion Vision Boosted Classifier (FVBC) with Visual Geometry Group (VGG)19 for features and LightGBM for classification. A specially prepared dataset of 2,627 leaf images of rice, divided between training, validation, and testing sets, were used to approximate the model performance. The FVBC model achieved accuracies of 97.78% on the training set, 97.5% on the validation set, and 97.6% on the test set, showing that it had a high performance in detecting rice diseases.

Deng et al. [19] presented a DL-based rice disease and insect pest detection method based on a mobile phone. This paper is a comparative study between two enhanced models in detection of six high-frequency diseases and insect pest detection—the improved You Only Look Once (YOLO)v5 and YOLOv7-tiny. They are both models based on lightweight object detection networks. The former uses the Ghost module for computation reduction while improving the structure of the model, and the latter uses the Convolutional Block Attention Module (CBAM) and SIoU for better learning precision of the model. Initially, the detection accuracy and operational efficiency of the models were evaluated and analyzed. Subsequently, the two proposed methods were deployed on a mobile phone, and an application was designed to further verify their practicality in detecting rice diseases and insect pests. The results indicated that the improved YOLOv5s achieved the highest F1-score of 0.931 and a mean average precision of 0.5.

Ayan et al. [20] proposed the use of a genetic algorithm-based weighted ensemble of deep CNNs for crop pest classification. Using suitable transfer learning and fine-tuning techniques, seven distinct pre-trained CNN models (VGG-16, VGG-19, ResNet-50, Inception-V3, Xception, MobileNet, and SqueezeNet) were adjusted and re-trained on the 40-class D0 dataset that is made available to the public during the study. Later, to improve classification performance, the top three CNN models—Inception-V3, Xception, and MobileNet—were ensembled using the sum of maximum probabilities technique. This new model was called SMPEnsemble. Following that, weighted voting was used to ensemble these models. The genetic algorithm, known as GAEnsemble, was used to establish the weights. The algorithm considers the predictive stability and success rate of three CNN models. With 98.81% classification accuracy for the D0 dataset, GAEnsemble achieved the top performance. The procedure was repeated using two other datasets—the SMALL dataset with 10 classes and the IP102 dataset with 102 classes—in order to improve the robustness of the ensembled model without affecting the first best-performing CNN models on D0. For the SMALL dataset and IP102, the accuracy numbers for GAEnsemble were 95.15% and 67.13%, respectively. However, the version of the IP102 dataset used is outdated as it does not contain the images of the recent insect pest.

In similar research, Pattnaik and Parvathy [21] proposed the use of ML-based approaches for tomato pest classification. In the paper, automatic tomato pest identification and categorization utilizing ML-based image processing techniques was reported. The study took into consideration the textural properties of pest photos that were recovered using feature extraction methods such as local binary pattern (LBP), histogram of oriented gradient (HOG), gray level co-occurrence matrix (GLCM), and sped up robust features (SURF). For the classification process, three common classification techniques were applied: decision tree (DT), k-nearest neighbor (KNN), and SVM. To show which classifier produces the best accuracy with which feature, a thorough analysis of the three classifiers was conducted. The experiment's findings demonstrated that the greatest value of 81.02% is achieved by the SVM classifier's precision when employing the feature extracted by the LBP approach. However, no part of the work specified the dataset used to train the algorithms, rendering the presented results questionable.

In the work of Kasinathan et al. [22], modern ML techniques were proposed for insect classification and detection in field crops. The Wang and Xie dataset's nine and twenty-four insect classes were the subjects of classification experiments utilizing shape features and ML techniques like CNN, ANN, SVM, KNN, and Naive Bayes (NB). Also the research paper presents the insect pest detection algorithm that consists of foreground extraction and contour identification to detect the insects for Wang, Xie, Deng, and IP102 datasets in a highly complex background. Using the CNN model, the insects in the nine and twenty-four classes had the highest classification rates—91.5% and 90%, respectively. The data augmentation technique used increased the computational complexity of the model by increasing the training time.

Adi et al. [23] adopted the use of CNN for the identification of rice plant disease. The dataset of the study consisted of 1,600 images of rice leaves, divided into 4 classes: 400 images of rice affected by leaf blight disease, 400 images of rice affected by brown spot disease, 400 images of rice affected by leaf smut disease, and 400 images of healthy rice. In each class, the images were further divided into 380 for training and 20 for testing. During the rice image training process using the Python programming language, the accuracy achieved was 83%. The results were then saved as a model file and imported as training data into the program in Android Studio for use in an application. Testing the application with 20 rice leaf images from each class resulted in an actual accuracy of 94%. However, the size of the dataset class used was small, and the model was only capable of detecting a few diseases.

Singh et al. [24] introduced a custom CNN architecture designed to detect and classify common diseases in rice plants while reducing the network's parameter count. The proposed model was trained on a dataset comprising four

common rice plant diseases. Additionally, the research incorporated 1,400 on-field images of healthy rice leaves to facilitate the identification of disease-free plants. Independent experiments were conducted both with and without the inclusion of the healthy leaf dataset. The performance of the proposed model was assessed using stochastic gradient descent with momentum (SGDM) and adaptive moment estimation (Adam) optimization techniques by employing multiple performance metrics. Experimental findings indicated that, for the classification of four rice plant diseases, the model achieved a maximum test accuracy of 99.66% with SGDM and 99.83% with Adam in the 7th epoch. Furthermore, when the healthy leaf dataset was included, the model utilizing the Adam optimizer outperformed the SGDM-based model, achieving maximum test accuracies of 99.66% and 97.61% in the 7th epoch, respectively. The study achieved high accuracy; however, the model was not deployed in real time, which may affect the performance in field conditions, which are essential for practical agricultural applications.

Nayem et al. [25] used a custom CNN to detect ten different classes of rice pests, including bacterial leaf blight, bacterial leaf streak, bacterial panicle blight, blast, brown spot, dead heart, downy mildew, healthy, hispa, and tungro. The proposed model was trained on a dataset of 10,400 images and implemented with the Keras framework and a TensorFlow backend. The model achieved a validation accuracy of 88.18% using only 0.57 million parameters, demonstrating its effectiveness and efficiency in identifying rice pests.

From the review conducted, extensive research has been done by quite a number of scholars with respect to the detection and identification of insect pests and has achieved positive results. However, previous researches have been faced with challenges such as the use of small-sized datasets, which affects the model's performance on unseen examples; the use of outdated datasets that do not contain images of recent insect pests; the use of datasets from unknown sources; models that perform poorly on large datasets due to shallow depth; and the use of data augmentation techniques that increase the training time of the model. In addition, most research was not deployed in real time. As such, this research seeks to adopt and deploy a model that can perform well on a large dataset that contains recent images of insect pests and improved classes of insect pest species to solve the problem of generalization to unseen examples and prevent the use of data augmentation to decrease the training time of the model.

3 Methodology

The methodology used to develop a ML-driven web platform for automated insect pest identification is explained in detail in this section. The section starts with a conceptual framework, as illustrated in Figure 1, which lists the steps used, i.e., data collection from the maize leaf image, image preprocessing, fine-tuning of the model, training and evaluation of the model, and finally, model deployment.

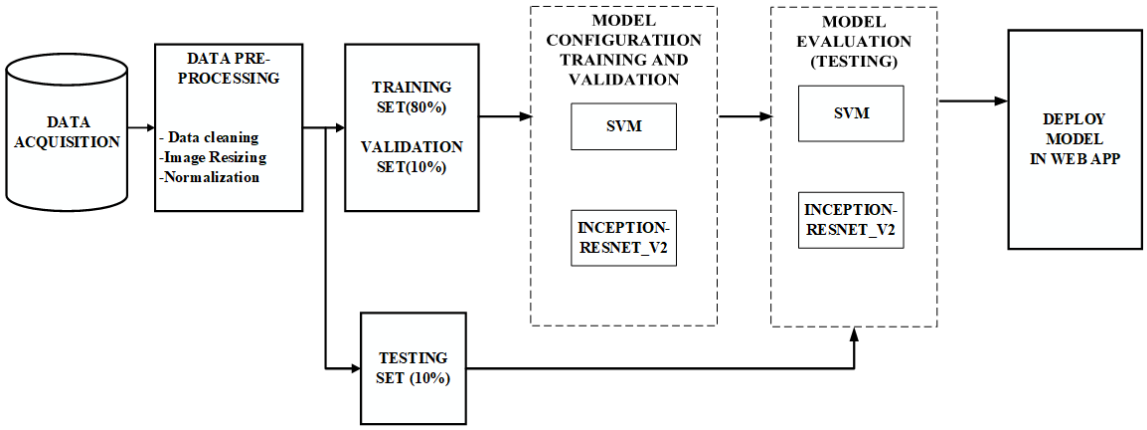

Figure 1. Conceptual framework for model development and deployment

3.1 Data Acquisition

The necessity for efficient rice pest identification has encouraged numerous researchers to create benchmark datasets appropriate for ML models. The IP102 dataset is a set of images specially designed for insect pest classification tasks. It comprises 102 classes of insect pests and 75,000 images that are widely encountered in agricultural environments [26]. Each category is a varied species of insect pest, and the dataset provides a lot of images for a single category in varied poses, angles, and lighting conditions. For this work, considering the species of the insect pest affecting rice, 12 classes of rice pest disease were selected with about 7,736 images, which occupy a space of 371 MB of memory from the IP102 dataset. The IP102 benchmark dataset contains rice images with unique characteristics like uniform backgrounds with relative intensity, complex back-grounds with occluded images,

complex backgrounds with multiple leaves in an image, and varying image quality (low quality, high-fidelity, and multiple backgrounds) for the rice pest dataset. This necessitates the need for effective preprocessing. Therefore, the obtained dataset was preprocessed using a uniform image size of 224 by 224 for compatibility, lower memory and resource demands. The IP102 dataset consists of rice pest images categorized into 12 classes, as presented in Table 1.

Table 1. Classification of rice pest diseases

S/N	Rice Rest	Quantity of Images
1	Rice leaf roller	605
2	Rice leaf caterpillar	475
3	Paddy stem maggot	325
4	Asiatic rice borer	745
5	Yellow rice borer	455
6	Rice gall midge	791
7	Brown planthopper	290
8	Rice stem fly	1,110
9	Rice water weevil	1,194
10	Rice leaf hopper	686
11	Rice shell pest	480
12	Thrips	580
	Total	7,736

From Table 1, it is evident that the distribution of images across the different rice pest classes is imbalanced. Such data imbalance can negatively affect model performance, especially by causing the model to become biased toward the classes with more samples. To mitigate this issue and ensure equal representation of each class during training, class weights were computed and assigned accordingly. The weights were calculated using Eq. (1).

$$\text{Class Weight} = \frac{\text{Total Number of Samples}}{\text{Number of Classes} \times \text{Number of Samples in Class}} \tag{1}$$

Figure 2. Sample images from the IP102 dataset

This formula ensures that minority classes are assigned higher weights, thereby penalizing their misclassification more heavily during model training. By incorporating these weights into the categorical cross-entropy loss function, the model gives more importance to underrepresented classes, effectively reducing bias and improving classification performance across all categories. Many researchers have attributed the existing model's computationally expensive nature to larger dataset sizes and high-resolution images. The IP02 rice pest dataset images have varying sizes; this shows that the images in the dataset contain high-resolution images, which demand more computational and memory resources if adopted as they are. Therefore, it is important to ensure that the images are resized into uniform sizes to ensure consistency and compatibility across the dataset, preventing errors during model training and inference. Therefore, the images were resized to 224 by 224. Furthermore, to reduce the storage size (50%), improve efficient querying, faster data access (random access and optimized I/O), and improve data loading/saving across multi-platform, the Python programming language was used to convert the rice pest dataset into hierarchical data (.h) format. Thus, the new improved dataset, an HDF5 dataset, has .h as an extension (RicePest.h). Figure 2 depicts sample images loaded and saved from the IP102 dataset.

3.2 Data Preprocessing

Data preprocessing was conducted to enhance the quality of the images and the performance of the model. The normalization of the pixel values ranged from 0 to 1, determined by the pixel value divided by 255. This step helped reduce variance at training and allowed faster convergence at optimization. The Gaussian filter was applied to remove image noise without destroying crucial visual characteristics so as not to confuse the model with disease-specific features. Background noise removal algorithms were also employed to crop the images and delineate areas of interest, particularly the rice leaves without eliminating the background noise. This allowed the model to focus on the most important features required for accurate disease classification, improving its predictability. Preprocessing was not just resizing but was a series of techniques to improve dataset quality and readability. Normalization gave a standard pixel value range and also prevented the model from being overwhelmed by larger values during training. This ensured learning stability and feature weighting balance. Preprocessing also addressed any potential data imperfections lastly. Non-functional or anomalous images were detected and removed to ensure the quality of the training dataset. Such quality control was performed manually and using the properties of automated functions, i.e., checksum verification, to detect and remove anomalies from the data. Elimination of irrelevant photos allowed the model to learn from pertinent data only for the classification problem, thus improving overall accuracy and reliability. For this research, I_{orginal} denotes the original image with dimensions $H_{\text{original}} \times W_{\text{original}}$. After resizing to the target size, $H_{\text{target}} \times W_{\text{target}}$, the resized image (I_{resized}) is denoted by Eq. (2).

$$I_{\text{resized}} = \text{resize}\left(I_{\text{original,}}\left(H_{\text{target}} \times W_{\text{target}}\right)\right) \tag{2}$$

Let $x_{\min}, y_{\min}, x_{\max}$, and $y_{\max}$ denote the coordinates of the bounding box defining the region of interest. The cropped image (I_{cropped}) can be obtained by extracting the region of interest from the original image, as presented in Eq. (3).

$$I_{\text{cropped}} = I_{\text{original}}\left[y_{\min} : y_{\max}, x_{\min} : x_{\max}\right] \tag{3}$$

Preprocessing may also entail trimming photos to eliminate extraneous background or unimportant elements. It is noteworthy that by restricting the area of interest to these locations exclusively, this study may reduce the noise level resulting from superfluous photos of other regions and improve the model's performance when comparing the model and the real result for the position of the rice leaves. Normalization is a crucial preprocessing step for data. To reduce the run time of optimization approaches during model training, it is a usual practice to standardize the pixel values at multiples of [0,1] or [-1,1]. This entails normalizing each pixel value by the maximum possible pixel value, which for 8-bit pictures NAIVE is 255, and then putting all of the pixel values inside that range. Naturally, there is also the choice to divide the mean pixel value by the standard deviation, which can yield values that are roughly equal to zero and have a unit standard deviation. In this research, $I_{\text{normalized}}$ denotes the normalized image. Eq. (4) shows the normalization of the range if it is between 0 and 1.

$$I_{\text{normalized}} = \frac{I_{\text{original}}}{255} \tag{4}$$

Standardization was conducted by dividing by the standard deviation after subtracting the mean, as presented in Eq. (5).

$$I_{\text{normalized}} = \frac{I_{\text{original}} - \mu}{\sigma} \tag{5}$$

where, σ is the standard deviation of the dataset's pixel values, and μ is the mean pixel value.

3.3 Dataset Splitting

The dataset needs to be split in order to employ any ML technique: one for testing, one for model validation checks, and one for model training. This is called data partitioning and needs to be completed before applying any ML techniques. The dataset for this study was split into three parts: testing (10%), validation (10%), and training (80%). This split is based on accepted practices and literature. The majority of the dataset was allotted to the training set, which is in charge of discovering the patterns, connections, and characteristics of the dataset.

3.4 Development of the Inception_ResNetV2 and SVM Model for Effective Pest Classification

For the purpose of enhancing the robustness, accuracy, and computational expense of rice pest identification operations, this research designed an improved Inception_ResNetV2 and SVM model. Inception_ResNetV2 is a variant deep CNN architecture and is highly trending for its factorized convolutions in feature extraction optimization and reduced computational complexity. SVM, however, is a strong machine-learning algorithm that is well known for its better classification performance in high-dimensional feature spaces. Hence, the improved model took the best of both approaches.

3.4.1 Inception_ResNetV2 CNN

The strength of an Inception ResNet-v2 network is its ability to combine a convolutional and a pooling layer, which eliminates the requirement to choose the appropriate filter size and its dimensions from a wide range of options for a convolutional layer or between a conv and a pooling layer. As a result, they make excellent selections for images of various sizes. In this research, to train the proposed Inception ResNet-v2 model to learn representative features from the insect pest dataset, the standard architecture was adopted. Figure 3 shows the Inception_ResNetV2 architecture.

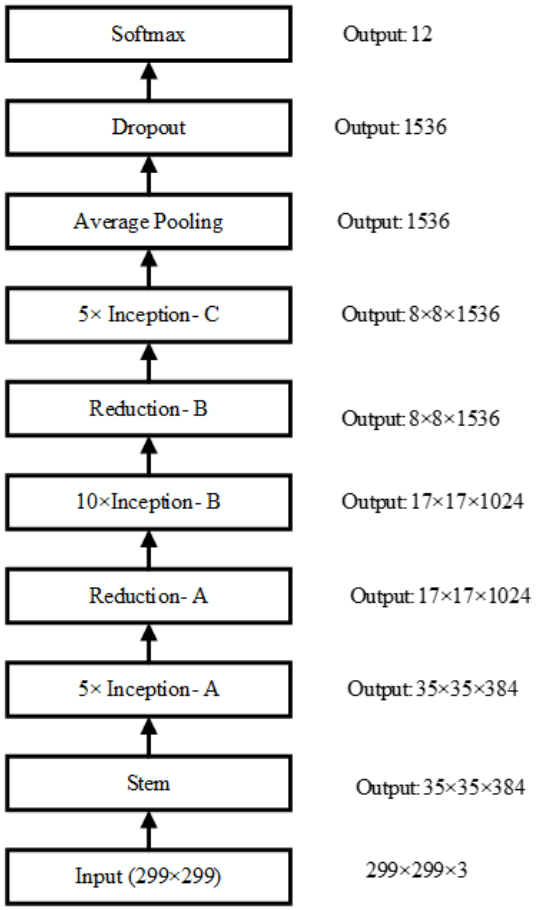

Figure 3. Inception_ResNetV2 architecture

In this research, Inception_ResNetV2 extracts deep hierarchical features from the input rice pest images which can be mathematically expressed by Eq. (6).

$$F = \text{Inception\_RestNetV2}(X, W, b) \tag{6}$$

where, X is the input pest image, W is the weight of the CNN layers, b represents the biases of the CNN layers, and $F \in R^d$ is the extracted feature vector of dimension d. The Inception_ResNetV2 algorithm is presented as below.

Algorithm 1 Inception_ResNetV2

Step 1: Load Dataset
    The input images of pests from the IP02 dataset were collected.
    Then the images were preprocessed (resized and normalized).
Step 2: Feature Extraction Using Inception_ResNetV2
    The Inception_ResNeXt model (trained from scratch) was imported.
    The last fully-connected (FC) layer was cut off.
    Input the images X_i one-by-one for feature extraction using Eq. (6).
    Save the feature vectors F_i extracted for all training images.
Step 3: Train the Model
    Train with the extracted features F_i and respective labels y_i.
    Hyper-parameter tuning for cross-validation.
Step 4: Model Evaluation
    Test the hybrid model on the validation/test dataset.
    Measure performance in terms of accuracy, precision, recall, and F1 score.
Step 5: Deployment
    Deploy a production-level model for real-time pest classification in agricultural environments.

This Inception_ResNetV2 model addressed the challenges of class imbalance, misclassification due to background noise, and variations in pest morphology.

3.4.2 SVM

SVM is the technique that allows effectively separating various classes of data points with the best hyperplane in an associated high-dimensional feature space. The data were projected into this feature space using kernel functions, where SVM maximized the smallest margin obtained by the closest support vectors to different classes. A regularization parameter (C) was used to create the trade-off such that maximizing margins and keeping classification errors to a minimum on the training data were optimized. The SVM architecture is shown in Figure 4.

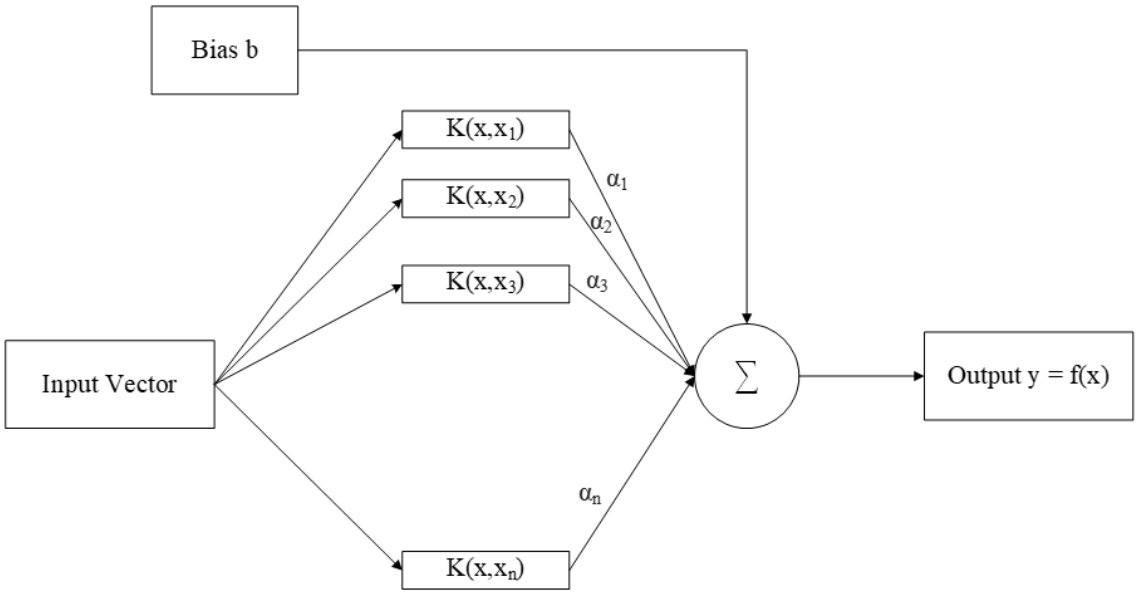

Figure 4. Architecture of SVM [27]

Since SVM relies on determining the optimal decision boundary based on the distribution of the input data in the feature space rather than layered structures, it is especially useful for classification tasks with intricate, nonlinear decision borders.

a) Feature extraction

In this research, after the dataset of rice insect pest images was retrieved globally and preprocessed by resizing images, it was converted to grayscale or Red, Green, and Blue (RGB) format, and pixel values were normalized. The

relevant features were extracted from these preprocessed images. Features, including color histograms, texture, shape, and descriptors, serve as the input to the SVM classifier.

b) Training the SVM model

To train the SVM model, the extracted features and their corresponding class labels were used. The SVM algorithm attempted, during training, to identify the hyperplane that maximizes the margin between classes and most correctly separated the feature vectors of the various classes. The data points closest to the decision boundary, or support vectors, have a very important role to play in deciding this hyperplane. The regularization parameter (C) and the type of kernel function affect the performance and the generalization ability of SVM. SVM was used to continuously re-tune the hyperplane to minimize the classification errors on the training set while maintaining a maximum margin.

c) Classification of test images

After training, the SVM model was able to classify new images of rice insect pests. This entailed extracting the same feature set in the test images as what had been done to train on them. These features were then passed to the SVM model, which evaluated them for distance calculations from the decision boundary. SVM categorized the test images into one of the pre-defined classes based on the distances. The comparison for the predicted test image class label starts from analyzing the minimum distance from the hyperplane or the highest class confidence score. The insect pest photo classification outcome is simply an outworking of applying this procedure for every test image and using the patterns learned from the training set. The SVM model has a special property that can distinguish well among different species of rice pests. This is given by Eq. (7).

$$f(X) = \text{sign}\left(\sum_{i=1}^{n} \alpha_i y_i K\left(F_i, F\right) + b\right) \tag{7}$$

where, α_i are the langrage multipliers, $y_i \in \{-1, +1\}$ are the class labels, $K\left(F_i, F\right)$ is the kernel function, e.g., Radial Basis Function (RBF) or linear kernel, and b is the bias term.

SVM optimizes the hinge loss function to maximize the margin between classes, as shown in Eq. (8).

$$\mathrm{L} = \sum_{i=1}^{n} \max\left(0, 1 - y_i f\left(X_i\right)\right) + \frac{\lambda}{2}\|w\|^2 \tag{8}$$

where, the first term ensures correct classification, and the second term is the regularization term with hyperparameter λ to avoid overfitting.

3.5 Model Configuration and Training

The hyperparameters in Table 2 represent the various configurations used to train the InceptionResNetV2 model and the SVM model for this research on automated identification of insect pests. Batch sizes of 32 were used for both models, a standard choice that balances memory efficiency and model convergence. 50 epochs were trained to provide enough time for the models to learn from the data and to avoid underfitting. To avoid overfitting, early stopping with patience 5 was used, which means training would be halted if validation loss does not improve for five consecutive epochs. Optimization-wise, InceptionResNetV2 utilizes the Adam optimizer, which is known for its adaptive learning rate as well as speed of convergence at a learning rate of 0.0001.

Table 2. Detailed hyperparameters configured in the Inception ResNetV2 and SVM models

Parameters	Inception_ResNetV2 Value	SVM Value
Image size	299×299	64×64
Batch size	32	32
Epoch	50	50
Patience	5	5
Learning rate	0.0001	0.0001
Optimizer	Adam	SGD
Loss	Categorical cross-entropy	Squared hinge
SVM kernel	NA	Linear
SVM probability	NA	True
Training split	0.8	0.8
Validation split	0.1	0.1
Test split	0.1	0.1

The SVM model, in contrast, uses Stochastic Gradient Descent (SGD), a commonly used optimization technique, for large-scale ML problems. Categorical cross-entropy was used as the loss function for InceptionResNetV2, suitable

for multi-class classification, whereas squared hinge loss was employed for SVM, which is designed for margin-based classification. In terms of input size, InceptionResNetV2 accepts images of size 299×299, providing more intricate feature extraction at the cost of increased computational requirements. The SVM model, however, accepts images of size 64×64, where computational efficiency and processing speed are accorded greater priority, and thus the model is comparatively more viable for web-based real-time systems. The train, validation, and test splits were kept at 80:10:10 to ensure a well-balanced dataset for the adequate training, validation, and evaluation purposes of the models. The SVM classifier uses a linear kernel, which works well with high-dimensional data. In addition, probability estimates were set to true to allow confidence-based predictions.

3.6 Model Evaluation

Accuracy, precision, recall, and F1-score were utilized in this research as performance metrics for the classification of rice insect pests into their classes. Accuracy is the general percentage correct in classifying, yet it can be misleading if the dataset isn't sufficiently balanced like in the case where one species of an insect is a lot larger than the rest. Precision measures the proportion of true positives, i.e., pest species correctly identified to all positive predictions, which is critical in preventing false positives, e.g., incorrectly classifying a harmless insect as a pest. Recall verifies the model's ability to predict actual rice disease, which is critical in pest control, where misidentification of harmful species could lead to significant agricultural loss. The F1-score provides a balance between precision and recall and is especially useful in the case of imbalanced datasets. Eqs. (9)-(12) depict the mathematical form of accuracy, precision, recall, and F1-score [28, 29].

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{9}$$

$$\text{Precision} = \frac{TP}{TP + FP} \tag{10}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{11}$$

$$F1 - \text{score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{12}$$

3.7 Model Deployment

Since the expected end users of this system are farmers who may have little or no programming skills, it becomes necessary to develop a friendly user interface that can ease the classification of any captured rice pest images for effective and early pest control. This study used a user-friendly methodology to create a web-based application that uses ML to classify rice insect pests in real time. The web interface, which was built using HTML/CSS/JavaScript on the frontend and the Streamlit framework on the backend, was designed to allow researchers to upload and categorize images of rice insect pests into groups related to various rice diseases, such as brown plant hopper, Asiatic rice borer, rice leaf roller, and rice leaf caterpillar. The real-time display of the classification results was made possible by the application's dynamic and responsive frontend. The following is an outline of the phases:

a) Configuring the web interface

Flask serves as the web application's first backend framework, handling user requests and controlling ML model interaction. To build a responsive and tidy layout, HTML and CSS were used to create the frontend. The layout included a middle file upload button to allow users to upload their rice insect pest images and a space to show the classification output.

b) Uploading and displaying rice insect images

The users can upload a rice insect pest from their personal workstation by clicking the upload image button. The users can see a file dialog box where they may search for and choose the image they wish to categorize. Following the selection of an image, the Pillow library was used for preprocessing and scaling. The online interface displayed the image so that the user could examine it thoroughly before classifying it.

c) Displaying results of classification

The model output showed the predicted class of rice disease and was presented on the web interface, allowing users to easily identify what disease the model predicted.

3.8 System Specification

The simulation was conducted on a Zinox computer that has an Intel Core i7-6700 processor clocked at 3.40 GHz, 16 GB of random access memory (RAM), a dedicated 64 MB graphics card, and 64-bit Windows 10 Professional. The models were developed and configured in the Spyder Integrated Development Environment (IDE) using libraries, including TensorFlow, Keras, matplotlib, numpy, pandas, and tkinter. Because of the absence of a dedicated Graphic Processing Unit (GPU) for ML processing, training models was extremely slow with the configurations in Table 2. The length of training time was primarily due to the size of the images (299 × 299 for Inception_ResNetV2) and the complex model architectures that require vast amounts of processing power. Performance was also restricted by the use of 16 GB of RAM, especially while backpropagating and splitting the dataset into training, validation, and testing. Therefore, every training experiment took hours to complete.

4 Result and Discussion

This section discusses the obtained results from the deployment of the developed ML-driven web platform for automated insect detection. This entails the results obtained from a comparative analysis of SVM and Inception_ResNetV2 models. Accuracy, precision, recall, F1-score, loss, latency inference time, energy consumption, computational time (non-GPU system), bandwidth, and efficiency as evaluation metrics were equally discussed. Equally, the results obtained from the comparative analysis were validated with the most recent related research.

4.1 Globally Retrieved Dataset of Insect Pest and Preprocessed Dataset

This subsection discusses the retrieved benchmark insect pest dataset as well as the communication requirement in the preprocessing of the data using bits as performance metrics.

The benchmark insect pest dataset called the IP102 Dataset has been majorly adopted by several researchers owing to its large image sizes totaling 75,000 images of 102 pest species, making it an essential resource for research in agricultural monitoring and pest management. Although this dataset's main goal is to construct a viable and solid model that accurately labels rice pests, it is essential to evaluate the dataset's data transmission, preprocessing, and performance from the communications aspects of the dataset, which adds consideration about effective and reliable use of the dataset with data integrity. The accessing and downloading of the dataset, as described in Section 3.1, allude to both understanding and encoding techniques, bandwidth limitations, data transmission. This is summarized in Table 3.

Table 3. Downloading the IP102 benchmark dataset

S/N	Metrics	Methods
1	Comprehension and encoding techniques	Lossy compression (JPEG), lossless compression (PNG), LZMA (7-Zip)
2	Bandwidth usage	Caching and resume support and concurrent downloads
3	Data transmission protocols	HTTPS protocols, Content Delivery Networks (CDNs), and Transport Layer Security (TLS)

a) Comprehension techniques

The IP102 dataset relies on the inherent structure of image files to convey information. The images in the dataset are in JPEG or PNG. Therefore, lossy and lossless compression were used, respectively, while ensuring high fidelity. To ease downloading and storage, the dataset was compressed into ZIP and TAR archive formats and facilitated downloading and storage on the computer with specifications captured in Section 3.8, employing algorithms like the Lempel–Ziv–Markov chain algorithm (LZMA) for efficient size reduction. LZMA is a lossless data compression technique that was created by Igor Pavlov. It is used in the 7z (7-Zip) compression format and is well known for its ability to achieve superior compression ratios compared to algorithms like DEFLATE.

b) Bandwidth usage

Bandwidth usage is a critical aspect when downloading large datasets like IP102. The IP102 pest dataset contains images of various pests affecting various crops, while this research focused on the pest peculiar to rice disease alone; it is paramount to download the entire dataset before extraction of the rice pest images. The IP102 dataset, with a total size of 7.8 GB, consumed network resources amidst the usage of effective compression like LZMA, reduced the size of the dataset and minimized bandwidth consumption during downloads. In addition, Kaggle supports multiple simultaneous downloads, which can optimize time but may increase overall bandwidth demands for large datasets. The caching and download resumption mechanisms inherent in Kaggle enabled the usage of download managers in downloading the dataset, thereby reducing redundant data transfers in case of network interruptions.

c) Data transmission protocols

Data transmission on Kaggle was secured through the adoption of the Hypertext Transfer Protocol Secure (HTTPS), which provides a robust method for safeguarding data in transit against unauthorized alterations, efficiently transferring either sensitive or non-sensitive data. This can mostly help speed distribution of the dataset via CDNs around the world, thus reducing latency and allowing faster data downloads for all users, regardless of location. In addition, Kaggle employs the Secure Socket Layer or Transport Layer Security to encrypt data in transit to protect against interception. Kaggle also provides Application Programming Interfaces (APIs) that allow users to download the datasets programmatically. APIs make it easier and more efficient for users to access the datasets using HTTP methods (GET and POST), enabling easier scripting and automation of processes.

4.2 Evaluation of the Models' Performance

The evaluation of Inception_ResNetV2 and SVM for effective rice insect pest classification was assessed using their computational efficiency, predictive accuracy, and communication effectiveness in real-time systems. The classification performance of these models was quantitatively assessed using standard performance metrics (accuracy, precision, recall, and F1-score), which collectively offer insights into the models' reliability in a practical deployment setting. This subsection discusses the results using communication metrics and performance metrics.

4.2.1 Communication performance metrics

From the communication system's perspective, the performance of the Inception_ResNetV2 and SVM models was analyzed using energy consumption, bandwidth efficiency, and latency, which are important when deploying these models in edge computing or Internet of Things (IoT)-based smart agriculture systems. The comparative performance of Inception_ResNetV2 and SVM using the communication performance metric is presented in Table 4.

Table 4. Comparative analysis of Inception_ResNetV2 and SVM using communication performance metrics

S/N	Performance Metrics	Inception_ResNetV2	SVM
1	Latency (inference time)	70 ms per image	54 ms per image
2	Energy consumption	$\sim$ 3.2 W per inference	$\sim$ 1.4 W per inference
3	Computational time (non-GPU)	1 hr 15 minutes	0 hr 49 minutes
4	Bandwidth efficiency	Higher	High

From Table 4, the Inception_ResNetV2 model exhibits a higher computational cost owing to its deep architecture, and this increases energy consumption. In addition, the convolutional layers in Inception_ResNetV2 require more data transmission bandwidth but can be optimized using compressed feature maps instead of raw images. SVM, on the other hand, operates on lower-dimensional handcrafted features and is more bandwidth-efficient with lower computational time. For low-power devices like IOT edge devices, SVM may be preferable due to its lower consumption and faster inference time. However, communication parameters alone cannot be used to ascertain the best model among the two. Therefore, the models were further subjected to statistical evaluation.

4.2.2 Performance metrics in terms of accuracy and loss

To evaluate how well the model is learning over time, the training and validation accuracy and loss graphs of both models were generated. The training accuracy measures how well the model performs on the training dataset, whereas the validation accuracy measures how well the model generalizes to unseen validation data. These graphs plot accuracy and loss values against epochs (iterations over the entire dataset) to visualize the model's performance, as seen in Table 5 and Figure 5 and Figure 6.

Table 5. Training and validation accuracy and loss comparative analysis of SVM and Inception_ResNetV2

Metrics	SVM		Inception_ResNetV2	
	Training (%)	Validation (%)	Training (%)	Validation (%)
Accuracy	99.64	99.97	99.97	86.47
Loss	0.0237	0.0098	0.0098	0.8360

As stated in Table 2, the epoch was set at 50. But during the model training phase, it automatically terminated at Epoch 10 and Epoch 16, as shown in Figure 5 and Figure 6, respectively. The reason hinged on the early stop mechanism introduced into the model. This helps the model to overcome overfitting, early convergence, and computational efficiency; hence, the model is prevented from memorizing noise and unnecessary details in the dataset and terminates when no further improvements are observed, leading to an effective reduction in the number of epochs. It is seen that the training and validation loss decreases and terminates at a low-value rate in both models. While the training loss terminates at 0.0237, the validation loss, on the other hand, terminates at 0.8360 in the SVM

model, and the training and validation loss of Inception_ResNetV2 terminate at 0.0098 and 0.7250, respectively. Inception_ResNetV2 achieves a training accuracy of 99.97 and a validation accuracy of 88.25, while the SVM achieves a training accuracy of 99.64 and a validation accuracy of 86.47. A balance of bias and variance can be seen as the gap between the training and validation performance indicates a well-generalized model.

To ascertain the robustness and effectiveness of the models, the accuracy, precision, recall, and F1-score were used to evaluate the performances of the models. To effectively evaluate the percentage improvement of the models, Eq. (13) was used.

$$\%\text{Improvement} = \frac{\text{ModelAP-ModelBP}}{\text{ModelAP}} \times 100 \tag{13}$$

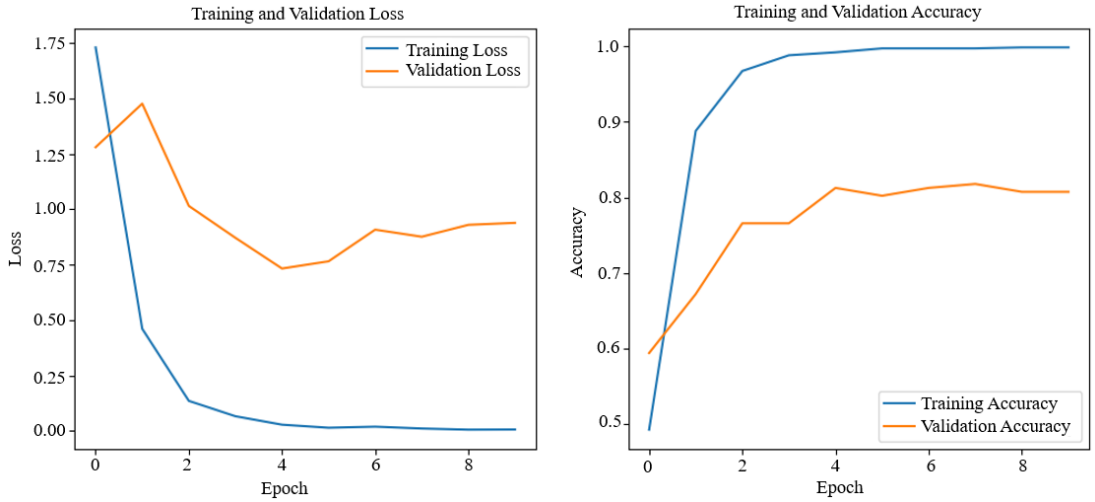

Figure 5. Training and validation loss and accuracy of SVM

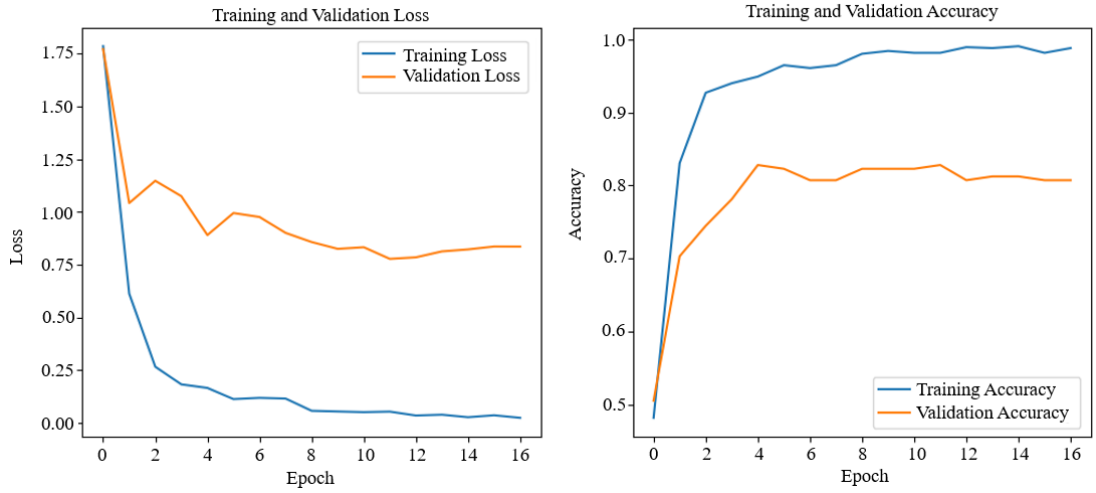

Figure 6. Training and validation loss and accuracy of Inception_ResNetV2

The results obtained from the simulations of the two models are presented in Table 6.

Table 6. Comparative results between Inception_ResNetV2 and SVM

Models	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Inception_ResNetV2	99.97	99.46	99.81	99.53
SVM	99.64	99.34	99.40	99.37
%Improvement	0.33	0.12	0.41	0.16

From Table 6, it is seen that Inception_ResNetV2 outperforms SVM with an improvement of 0.33%, 0.12%, 0.41%, and 0.16% in accuracy, precision, recall, and F1-score, respectively. This shows that Inception_ResNetV2 demonstrates its capability to learn complex hierarchical patterns in pest images and ensure that fewer pest cases go undetected, which is critical in real-time pest monitoring and decision support systems. In addition, the result demonstrates a balanced trade-off between precision and recall. For an effective real-time and high-accuracy pest classification system, Inception_ResNetV2 is the superior choice due to its higher recall and precision. However, in low-resource settings, like IoT edge devices, SVM may be preferable due to its lower power consumption and faster inference time. Hence, the choice of the best algorithm hinges on the trade-off and the expectation of the system. Inception_ResNetV2 was adopted in this research owing to its higher accuracy and detection rate, which are critical for the agricultural sector to ease farmers in early detection of diseases for effective measures. Inception_ResNetV2's performance was further evaluated using the confusion matrix to compare the predicted pest classes with the actual ones. The result obtained is presented in Figure 7.

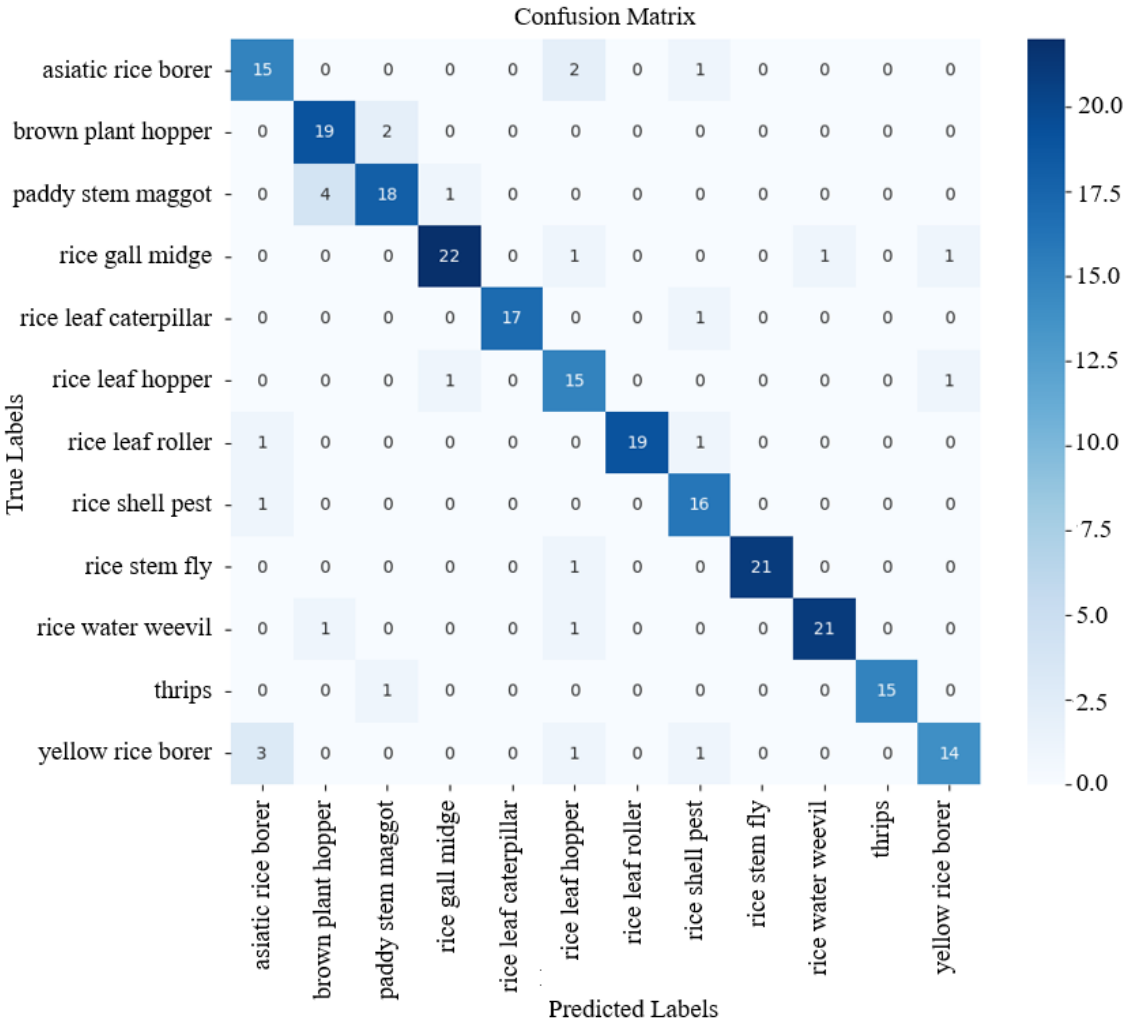

Figure 7. Inception_ResNetV2 confusion matrix

From Figure 7, the confusion matrix reveals key misclassifications among two rice pests like paddy stem maggot and rice leaf caterpillar, and such errors can lead to incorrect pest control measures, impacting crop yield and increasing costs. Enhancing classification accuracy is crucial for effective and targeted pest management.

As for both models, the ultimate goal is to apply them in practical applications like autonomous pest monitoring systems. Real-time communication in this scenario is extremely critical, as the image data needs to be sent in real time from sensors (drones, cameras, etc.) to the model on a processing unit (e.g., server or edge device). This is an instance of a communication system in which the "message" (in this example, the identification of the pest) is communicated from the sensors to the model, and the model responds with a classification or a suggestion back to the user or control system.

4.3 Deployment of an Inception_ResNetV2 Model on a Web Platform Using Streamlit

The Inception_ResNetV2 model was adopted due to its superior performance and was deployed in Python IDLE using Streamlit, serving as a bridge between ML, web technologies, and real-time communication systems. This allowed a high-performance DL model with an interactive and accessible web application for real-time insect pest classification, making it usable for agricultural stakeholders. To deploy the server online, the command "streamlit run <filename.py>" was used, where <filename.py> refers to the default model. The developed model was named Inception_ResNetV2PestClassifier.py; thus, the Streamlit run Inception_ResNetV2PestClassifier.py was used to start the server, as seen in Figure 8.

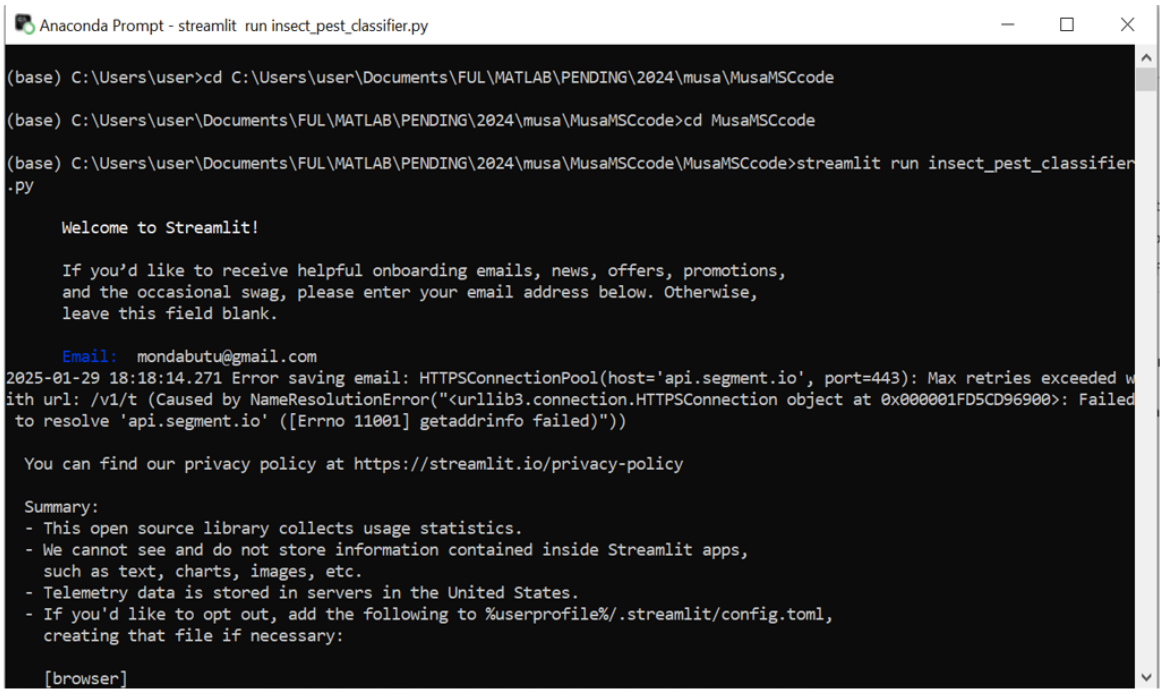

Figure 8. Starting Streamlit

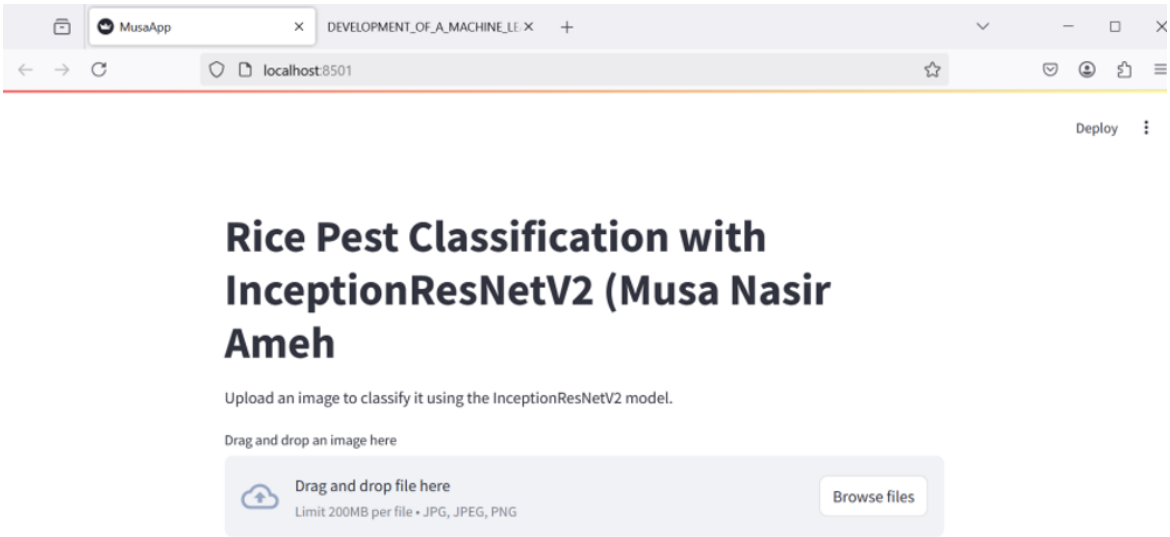

Figure 9. Developed user-friendly interface

The developed web application can be accessed either locally or remotely through any installed browser on the host and client computers. The host system allows access through the URL http://localhost:8501, while the connected client system uses the network URL http://192.168.44.167:8501, as shown in Figure 8. The user interface of the developed model can be accessed through any web browser by entering the server address. Upon establishing a successful connection, users are directed to an intuitive and user-friendly interface, as illustrated in Figure 9.

The user-friendly interface allows farmers, researchers, and agronomists to upload images of insect pests as well as real-time pest image acquisition using live camera input support, as presented in Figure 9. The browse files button allows the user to navigate to the directory where the pest image is, while the drag-and-drop option allows the user to drag the image directly into the file field. The loaded image is visualized on the browser, as shown in Figure 10.

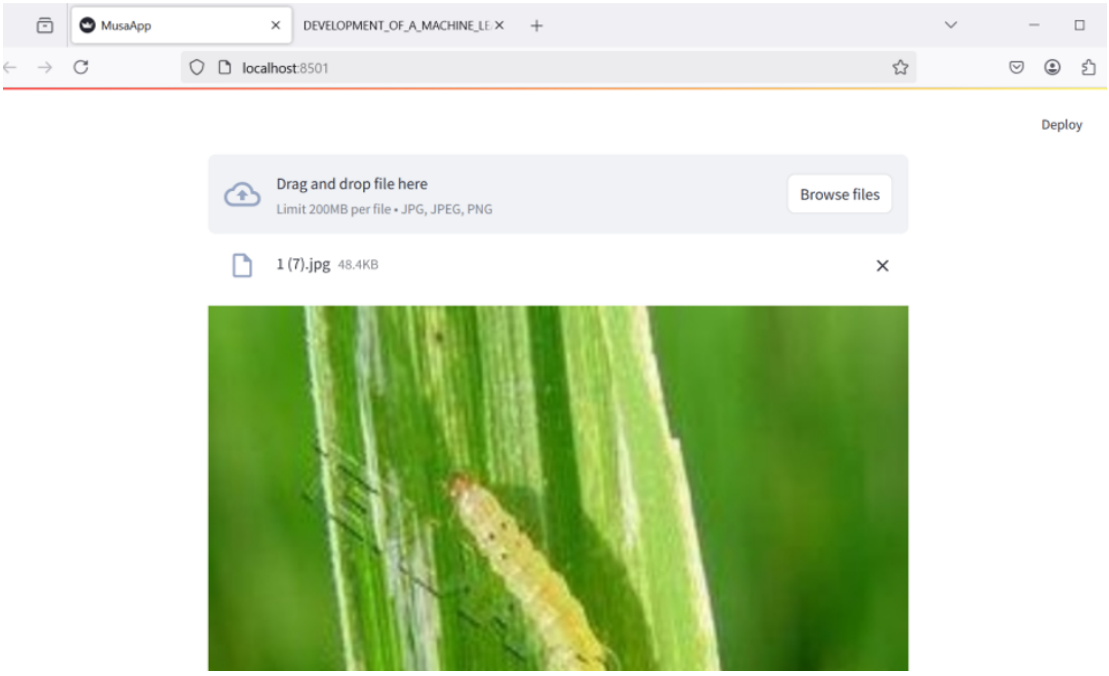

Figure 10. Verifying a pest image

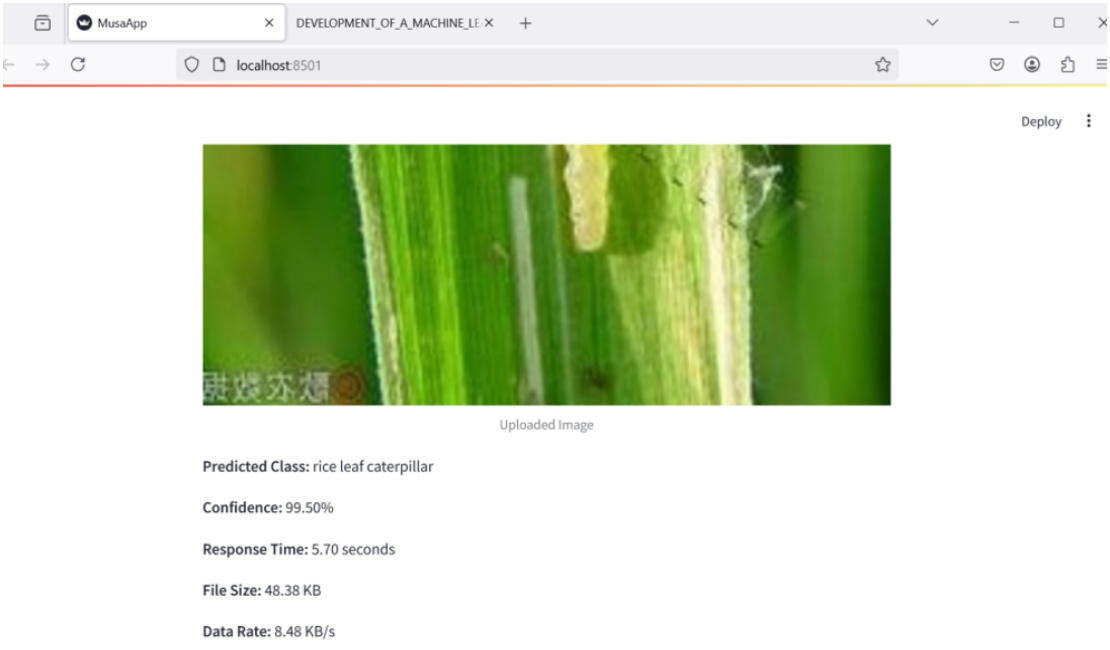

Figure 11. Pest prediction

In Figure 10, the pest image is successfully uploaded and processed through the user interface. Once the image is

loaded, the system activates the pre-trained model, saved in the .pkl file format, to analyze and predict the pest in the uploaded image. This cooperative coordination of the user interface and model allows effective and accurate detection and identification of the pests in order to enable farmers to get real-time and meaningful information for effective pest management. The prediction output is transmitted from the model to the user interface, as indicated in Figure 11.

As illustrated in Figure 11, the visualization dashboard presents the predicted pest as a rice leaf caterpillar, with a confidence level of 99.50%, a response time of 5.70 seconds, a data rate of 8.48 KB/s, and a file size of 48.38 KB. At the sight of the specific pest, the farmers are able to take appropriate and immediate actions. The rice leaf caterpillar feeds on rice leaves, which may lead to decreased photosynthesis and yield if uncontrolled. This means, with a confidence level of 99.50%, that the model is very sure of the fact that the pest in question is the rice leaf caterpillar. This represents the model's confidence in making the prediction. Thus, the high confidence ensures that the farmers believe the prediction and follow it accordingly without doubting the result. The particular information provides farmers with valuable knowledge regarding the specific pest affecting their crops and enables them to know the magnitude of damage that it can cause to their farms. By providing accurate and timely predictions, the dashboard allows farmers to make informed decisions and adopt the most effective pest control measures unique to the outlined pest, hence minimizing crop loss and optimizing agricultural output. Further analysis was performed to predict the remaining twelve classes. The findings obtained are presented in Table 7.

Table 7. Developed system performance

S/N	Pest Name	Data Rate (KB/s)	File Size (KB)	Response Time (seconds)	Confidence (%)
1	Rice leaf roller	7.21	43.25	5.57	99.89
2	Rice leaf caterpiller	8.48	48.38	5.70	99.50
3	Paddy stem maggot	9.56	51.27	6.23	99.72
4	Asiatic rice borer	5.10	40.72	5.21	99.45
5	Yellow rice borer	4.98	37.88	5.04	99.23
6	Rice gall midge	9.34	67.34	9.45	99.67
7	Brown planthopper	10.07	80.04	98.83	98.12
8	Rice stem fly	09.13	70.30	99.02	89.45
9	Rice water weevil	06.59	74.41	98.67	99.49
10	Rice leaf hopper	06.8	65.80	98.89	99.12
11	Rice shell pest	08.60	55.07	97.46	98.67
12	Thrips	07.44	53.37	96.23	98.50

From the evaluation of the developed system, the proposed system has a higher confidence level in the effective classification of the pest images into their respective classes at a shorter data rate and response time.

4.4 Comparative Analysis with Existing Models and Discussion

This section provides the comparative analysis of the deployed model with the closest related work in precision, recall, accuracy, F1-score, and deployment mode.

Table 8. Comparative analysis of the developed model with existing models

Reference	Classifier	Dataset Size	Accuracy	Precision	Recall	F1-Score	Deployment
[18]	FVBC	2627	97.60%	97.85%	97.70%	97.78%	No
[23]	AlexNet	1600	94%	94%	94%	94%	Mobile application
Developed Model	Inception_ResNetV2	7736	99.97%	99.46%	99.81%	99.53%	Web application

Table 8 presents a comparison of the proposed model with other models, showing significant enhancement in most of the evaluation metrics and deployment factors. The proposed model with the Inception_ResNetV2 classifier was trained on 7,736 images and achieved a remarkable accuracy of 99.97%. This is a very big improvement compared to AlexNet, which has 94% accuracy, and the FVBC, which has 97.60% accuracy. In addition, the model outperforms FVBC with +1.61% precision, +2.11% recall, and +1.75% F1-score. It also outperforms AlexNet with +5.46% precision, +5.81% recall, and +5.53% F1-score. These improvements suggest that the model can reduce false positives and negatives, thus ensuring the reliability of pest management. The improved precision, recall, and F1-score have direct real-world applications for pest management. Therefore, high precision reduces false positives, enabling crops not to be misclassified as infected, thus saving farmers time and resources. High recall ensures most pest infestations are detected, precluding late-stage damage from concealed pests. The enhanced F1-score, balancing precision with recall, ensures enhanced and more reliable pest detection regardless of the conditions. Moreover, the deployment

of the developed model as a web application offers farmers increased accessibility and ease of use over previous models not deployed or limited to mobile applications. This online deployment enables real-time monitoring of pests in multiple agricultural setups. The larger training set of 7,736 samples also makes the model more robust and generalizable, thus enabling it to handle changing environmental conditions and pest appearance more efficiently.

5 Conclusion

This research presents an improved ML-driven web platform for automated rice pest identification. It is necessary to resolve the inherent issue of the traditional approach in rice pest detection as well as the limitations of the classical ML algorithms and DL models in the effective detection of rice pests. No doubt that the traditional approach of rice pest detection and diagnosis entails the evaluation of pests and their effect on the farmland rate by an expert for effective pest control and removal. This approach is time-consuming, and an inaccurate understanding of some pest types can lead to poor control strategies. To alleviate this problem, modern technologies such as cloud computing, IOT, and artificial intelligence are integrated into agricultural practices to enhance crop monitoring for precise personality management, intelligent production control, quantitative decision-making, and effective crop harvesting. This research aims to improve the classification of insect pests in order to improve the performance of the training outcome. The collection of rice pest datasets and image preprocessing were first discussed. The rice pest images were globally retrieved from a benchmark insect pest dataset known as IP102. The IP102 dataset contains 102 classes of different agricultural insect pests with over 77,000 images. For the purpose of this study, 12 classes of the insect pest with about 7,736, which were peculiar to rice, were extracted from the IP102 dataset and were preprocessed to remove blur, corrupt, and incomplete images and resize images to suit the architectures of the models considered. Two ML models were considered for this research, namely, SVM and inception_ResNetv2, which were configured from scratch with a learning rate of 0.0001 and trained with 50 epochs to accurately classify the rice insect pest. The performance of the aforementioned ML models was documented. Judged by using the accuracy, precision, recall, and F1-score, inception_ResNetv2 was at the top of the performance metrics as it achieved an accuracy of 99.97%, a precision of 99.46%, a recall of 99.81%, and an F1-score of 99.53%. Finally, based on performance, inception_ResNetv2 was deployed into a web application. The outcome of this research highlights the potential of machines in the field of agriculture, particularly in reducing the reliance on traditional methods and mitigating the loss associated with rice farming, thereby improving pest detection, leading to higher crop yields, and strengthening food security.

Throughout the research, a number of limitations were found. The primary constraint is that deployment is currently limited to a web platform, which may be inaccessible to users in rural or remote regions with unstable internet connectivity. For future research, a hybrid deployment strategy is recommended—incorporating both web and mobile applications. A mobile app with offline functionality would enhance accessibility, allowing farmers to use the pest identification system without requiring a constant internet connection. This hybrid system could further integrate with other agricultural tools already present on farmers' devices, although challenges such as model compression, cross-platform optimization, and offline capability must be addressed through further development. Another limitation is the extended training time for the Inception_ResNetV2 model, primarily due to the lack of a dedicated GPU suitable for DL applications. Future work should utilize systems with high-performance GPUs and apply optimization methods such as mixed-precision training to reduce computational demands and enhance training efficiency. To improve the suitability of the model for real-time applications, future work should also focus on optimizing the Inception_ResNetV2 architecture. Techniques such as model pruning, quantization, and knowledge distillation can significantly reduce energy consumption and inference latency while preserving model performance. Additionally, the scalability and long-term maintenance of the web platform were not addressed in this study. Future research should explore strategies for scaling the platform to accommodate increasing user demand and maintaining model accuracy over time. This includes implementing mechanisms for periodic model retraining with newly collected data, and continuously updating the platform to integrate emerging technologies and ensure its relevance and effectiveness in real-world farming environments.

Data Availability

The image dataset supporting our research results are deposited in Kaggle, which does not issue DOIs. The data can be accessed at https://www.kaggle.com/datasets/hungt1/ip102-dataset.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] W. Li, T. F. Zhu, X. Y. Li, J. Z. Dong, and J. Liu, "Recommending advanced deep learning models for efficient insect pest detection," *Agriculture*, vol. 12, no. 7, p. 1065, 2022. https://doi.org/10.3390/agriculture12071065

[2] N. Ullah, J. A. Khan, L. A. Alharbi, A. Raza, W. Khan, and I. Ahmad, "An efficient approach for crops pests recognition and classification based on novel DeepPestNet deep learning model," *IEEE Access*, vol. 10, pp. 73 019–73 032, 2022. https://doi.org/10.1109/ACCESS.2022.3189676

[3] N. A. Mohidem, N. Hashim, R. Shamsudin, and H. Che Man, "Rice for food security: Revisiting its production, diversity, rice milling process and nutrient content," *Agriculture*, vol. 12, no. 6, p. 741, 2022. https://doi.org/10.3390/agriculture12060741

[4] M. A. Ali, I. M. Abdellah, and M. R. Eletmany, "Towards sustainable management of insect pests: Protecting food security through ecological intensification," *Int. J. Chem. Biochem. Sci.*, vol. 24, no. 4, pp. 386–394, 2023.

[5] Y. Bashir, Z. Sani, A. Ashafa, A. Mijinyawa, S. Sufiyanu, and U. Abdulazeez, "Morphological, biochemical and molecular characterization of the causal agent of bacterial panicle blight disease of rice (BPB) in selected rice production zones of Zamfara State," *Int. J. Sci. Glob. Sustain.*, vol. 10, no. 2, pp. 138–145, 2024. https://doi.org/10.57233/ijsgs.v10i2.656

[6] K. Li, X. Li, B. K. Liu, C. X. Ge, Y. H. Zhang, and L. Chen, "Diagnosis and application of rice diseases based on deep learning," *PeerJ Comput. Sci.*, vol. 9, p. e1384, 2023. https://doi.org/10.7717/peerj-cs.1384

[7] T. N. Doan, "Large-scale insect pest image classification," *J. Adv. Inf. Technol.*, vol. 14, no. 2, pp. 328–341, 2023. https://doi.org/10.12720/jait.14.2.328-341

[8] J. S. Mommoh, J. L. Obetta, S. N. John, K. Okokpujie, O. N. Omoruyi, and A. A. Awelewa, "Detection of fruit ripeness and defectiveness using convolutional neural networks," *Inf. Dyn. Appl.*, vol. 3, no. 3, pp. 184–199, 2024. https://doi.org/10.56578/ida030304

[9] S. Mandal, A. Yadav, F. A. Panme, K. M. Devi, and S. M. S. Kumar, "Adaption of smart applications in agriculture to enhance production," *Smart Agric. Technol.*, vol. 7, p. 100431, 2024. https://doi.org/10.1016/j.atech.2024.100431

[10] J. Botero-Valencia, V. García-Pineda, A. Valencia-Arias, J. Valencia, E. Reyes-Vera, M. Mejia-Herrera, and R. Hernández-García, "Machine learning in sustainable agriculture: Systematic review and research perspectives," *Agriculture*, vol. 15, no. 4, p. 377, 2025. https://doi.org/10.3390/agriculture15040377

[11] A. A. Mana, A. Allouhi, A. Hamrani, S. Rehman, I. el Jamaoui, and K. Jayachandran, "Sustainable AI-based production agriculture: Exploring AI applications and implications in agricultural practices," *Smart Agric. Technol.*, vol. 7, p. 100416, 2024. https://doi.org/10.1016/j.atech.2024.100416

[12] M. K. Sri, K. Saikrishna, and V. V. Kumar, "Classification of ripening of banana fruit using convolutional neural networks," in *Proceedings of the 4th International Conference: Innovative Advancement in Engineering & Technology (IAET)*, Rajasthan, India, 2020, pp. 1–5. https://doi.org/10.2139/ssrn.3558355

[13] S. Castillo-Girones, S. Munera, M. Martínez-Sober, J. Blasco, S. Cubero, and J. Gómez-Sanchis, "Artificial neural networks in agriculture, the core of artificial intelligence: What, when, and why," *Comput. Electron. Agric.*, vol. 230, p. 109938, 2025. https://doi.org/10.1016/j.compag.2025.109938

[14] M. El Sakka, J. Mothe, and M. Ivanovici, "Images and CNN applications in smart agriculture," *Eur. J. Remote Sens.*, vol. 57, no. 1, 2024. https://doi.org/10.1080/22797254.2024.2352386

[15] M. Albahar, "Survey on deep learning and its impact on agriculture: Challenges and opportunities," *Agriculture*, vol. 13, no. 3, p. 540, 2023. https://doi.org/10.3390/agriculture13030540

[16] R. L. Deng, M. Tao, H. Xing, X. L. Yang, C. Liu, K. F. Liao, and L. Qi, "Automatic diagnosis of rice diseases using deep learning," *Front. Plant Sci.*, vol. 12, 2021. https://doi.org/10.3389/fpls.2021.701038

[17] V. P. Bhartiya, R. R. Janghel, and Y. K. Rathore, "Rice leaf disease prediction using machine learning," in *2022 Second International Conference on Power, Control and Computing Technologies (ICPC2T)*, Raipur, India, 2022, pp. 1–5. https://doi.org/10.1109/ICPC2T53885.2022.9776692

[18] B. Naresh Kumar and S. Sakthivel, "Rice leaf disease classification using a fusion vision approach," *Sci. Rep.*, vol. 15, no. 8692, 2025. https://doi.org/10.1038/s41598-025-87800-3

[19] J. Deng, C. Yang, K. Huang, L. Lei, J. Ye, W. Zeng, J. Zhang, Y. Lan, and Y. Zhang, "Deep-learning-based rice disease and insect pest detection on a mobile phone," *Agronomy*, vol. 13, no. 8, p. 2139, 2023. https://doi.org/10.3390/agronomy13082139

[20] E. Ayan, H. Erbay, and F. Varçın, "Crop pest classification with a genetic algorithm-based weighted ensemble of deep convolutional neural networks," *Comput. Electron. Agric.*, vol. 179, p. 105809, 2020. https://doi.org/10.1016/j.compag.2020.105809

[21] G. Pattnaik and K. Parvathy, "Machine learning-based approaches for tomato pest classification," *TELKOMNIKA Telecommun. Comput. Electron. Control*, vol. 20, no. 2, pp. 321–328, 2022. https://doi.org/10.12928/TELKOMNIKA.v20i2.19740

[22] T. Kasinathan, D. Singaraju, and S. R. Uyyala, "Insect classification and detection in field crops using modern machine learning techniques," *Inf. Process. Agric.*, vol. 8, no. 3, pp. 446–457, 2021. https://doi.org/10.1016/j.inpa.2020.09.006

[23] K. Adi, C. E. Widodo, A. P. Widodo, A. Setiadi, and A. Setiawan, "Identification of rice plant diseases using convolutional neural network method," *Math. Model. Eng. Probl.*, vol. 11, no. 5, pp. 1279–1285, 2024. https://doi.org/10.18280/mmep.110517

[24] S. P. Singh, K. Pritamdas, K. J. Devi, and S. D. Devi, "Custom convolutional neural network for detection and classification of rice plant diseases," *Procedia Comput. Sci.*, vol. 218, pp. 2026–2040, 2023. https://doi.org/10.1016/j.procs.2023.01.179

[25] Z. H. Nayem, I. Jahan, A. A. Rakib, and S. Mia, "Detection and identification of rice pests using memory efficient convolutional neural network," in *2023 International Conference on Computer, Electrical & Communication Engineering (ICCECE)*, Kolkata, India, 2023, pp. 1–6. https://doi.org/10.1109/ICCECE51049.2023.10084936

[26] Kaggle, "IP102 dataset." https://www.kaggle.com/datasets/hungt1/ip102-dataset

[27] X. C. Wang, J. Zhao, Q. Q. Li, N. R. Fang, P. C. Wang, L. T. Ding, and S. Q. Li, "A hybrid model for prediction in asphalt pavement performance based on support vector machine and grey relation analysis," *J. Adv. Transp.*, vol. 2020, pp. 1–14, 2020. https://doi.org/10.1155/2020/7534970

[28] K. Okokpujie, I. P. Okokpujie, O. I. Ayomikun, A. M. Orimogunje, and A. T. Ogundipe, "Development of a web and mobile applications-based cassava disease classification interface using convolutional neural network," *Math. Model. Eng. Probl.*, vol. 10, no. 1, pp. 119–128, 2023. https://doi.org/10.18280/mmep.100113

[29] K. Okokpujie, S. John, C. Ndujiuba, and E. Noma-Osaghae, "Development of an adaptive trait-aging invariant face recognition system using convolutional neural networks," in *Information Science and Applications, Lecture Notes in Electrical Engineering*. Springer, Singapore, 2020. https://doi.org/10.1007/978-981-15-1465-4_41