RESEARCH PAPER

Pest Prediction and Pesticides Recommendation Using Deep Learning

S. Uthayashangar[1], S. Hema[2], S. Kamala Yaazhini[3], A. Praveena[4]
[1]Assistant Professor, Department of Information Technology, Manakula Vinayagar Institute of Technology, Puducherry, India.
Email: uthayashangarit@mvit.edu.in
[2,3,4]U.G Scholar, Department of Information Technology, Manakula Vinayagar Institute of Technology, Puducherry, India.
Email: hemaleela003@gmail.com, Yaazhinisaravanan09@gmail.com, praveenaarumugam1602@gmail.com

***Abstract*—** Agricultural pests are a serious threat to harvest production, which leads to significant economic losses and malnutrition uncertainty. Traditional methods for identifying pests are time-consuming and prone to human error. The project addresses these challenges by using deep learning techniques to automate pest detection and classification. The persuasive model spread across ImageNet has been finely tuned to identify 51 pests from the photographs. The system process uploads images, classifies pests, and provides appropriate pesticide issues. A resolution-based web interface ensures user friendly interactions and allows for efficient identification of pests. This automated approach improves accuracy, reduces identification times and supports the appropriate decision process for pest control. Future work will include expanding data records and integrating real-time monitoring of field applications.

***Keywords*—**Pest Detection, Deep Learning, Image Classification, ConvNeXt, Flask, Transfer Learning, Agricultural Pests, Pest Identification, Pesticide Recommendation, Automated System

How to cite this article: Uthayashangar S, Hema S, Yaazhini SK, Praveena A. Pest Prediction and Pesticides Recommendation Using Deep Learning. Int J Drug Deliv Technol. 2026;16(16s): 442-450. DOI: 10.25258/ijddt.16.16s.46.

I. INTRODUCTION

Agriculture plays a vital role in sustaining human life by providing food and raw materials. However, one of the most persistent challenges faced by farmers worldwide is the damage caused by agricultural pests. These pests not only reduce crop yields but also affect crop quality, leading to substantial economic losses and contributing to food scarcity. Early identification and proper pest management are crucial to maintaining agricultural productivity and ensuring food security.

Traditionally, pest detection is carried out through manual inspection by farmers or agricultural experts. This process is time-consuming, labor-intensive, and prone to human error, especially when identifying similar-looking pest species. Additionally, the lack of quick and accurate pest identification can delay appropriate pest control measures, causing further crop damage. Given the increasing scale of agricultural activities,

there is a growing need for automated and efficient pest detect ion systems that can contribute to early diagnosis and targeted pest control. The project addresses these challenges through th e development of automated detection and classification syste ms with deep learning techniques. The proposed system uses t he Convnext model of the latest architecture (CNN) of neural networks known for its efficiency and high accuracy of image classification tasks. This model is trained on image protection data records and fine-tuned with custom data records containing 51 agricultural pest species. This transfer learning approach improves the output o f the model and shortens the training period required for pest c lassification.

The system is deployed through a Flask-based web interface, allowing users to upload images of pests for real-time identification. Upon processing the image, the system classifies the pest species and suggests appropriate pesticide recommendations. This user-friendly interface facilitates easy access for farmers and agricultural professionals, enabling rapid decision-making and effective pest control.By automating the pest identification process, this project aims to enhance agricultural productivity, reduce economic losses, and minimize the reliance on manual inspections. Future work can focus on extending the dataset, improving the model's predictive accuracy, and integrating real-time monitoring capabilities to provide a comprehensive pest management solution.

II. PROBLEM STATEMENT

Agricultural pests cause significant damage to crops, leading to reduced yield, financial losses, and food insecurity. Traditional pest identification relies on manual inspection, which is time-consuming, prone to human error, and impractical for large-scale farming. Misidentification or delayed detection can result in improper pest control measures, increasing crop vulnerability. The primary problem this project addresses is the need for an accurate, automated, and user-friendly pest detection system.

By leveraging deep learning techniques, the system can quickly classify different pest species and recommend appropriate pesticides. This solution aims to improve detection accuracy, reduce the time required for pest identification, and assist farmers in effective pest management through a web-based interface.

Key Challenges Addressed:

1. Inaccurate and time-consuming manual pest identification.
2. Difficulty in recognizing visually similar pest species.
3. Lack of automated systems for large-scale monitoring.
4. Providing appropriate pest control measures promptly.

This undertaking ambitions to mitigate these demanding situations through implementing a deep-getting to know-primarily based pest detection and class device with actual-time photograph analysis and pesticide tips.

III. Objectives

The primary objective of this project is to develop an automated system for pest detection and pesticide recommendation using deep learning techniques. The project focuses on the following key goals:

1. **Accurate Pest Identification**: Utilize a deep learning model (ConvNeXt) to accurately classify agricultural pests from images.
2. **Pesticide Recommendation**: Provide appropriate pesticide suggestions based on the detected pest species.
3. **Efficient Image Classification**: Implement transfer learning to improve model accuracy and reduce training time.
4. **User-Friendly Web Interface**: Develop a Flask-based web application that allows users to upload images and receive real-time predictions and pesticide recommendations.
5. **Scalability and Future Enhancement**: Design the system to support the addition of new pest species and advanced model fine-tuning for enhanced detection.

Methodology –

1. **Data Collection and Preprocessing**:
- Utilize a curated dataset containing images of 51 pest species.
- Preprocess images by resizing (256x256 pixels), center cropping (224x224 pixels), and normalizing using ImageNet mean and standard deviation.
- Convert images into tensor format for compatibility with PyTorch.
2. **Model Selection and Training**:
- Appoint the ConvNeXt version pre-trained on ImageNet for better generalization.
- Update the very last classifier layer to fit the 51 pest categories.
- Observe the go-entropy loss characteristic and optimize the usage of the Adam optimizer.
- Use batch normalization to enhance training stability and overall performance.
3. **Transfer Learning**:
- Leverage pre-trained ConvNeXt weights to accelerate model convergence.
- Fine-class-track the model at the pest dataset to enhance class accuracy.
4. **Model Evaluation**:
- Evaluate model performance using metrics like accuracy, precision, recall, and F1-score.
- Conduct extensive testing with unseen pest images to ensure robustness.
5. **Web Application Development**:
- Design a Flask-based web interface for image upload and pest detection.
- Integrate model inference to classify pests and provide corresponding pesticide recommendations.
6. **Deployment and Testing**:
- Deploy the application on a local server for testing.
- Perform user testing to validate the system's accuracy and usability.
- Gather feedback for future enhancements and scalability improvements.

IV. System Architecture

The system architecture outlines the structure and workflow of the pest detection and pesticide recommendation system. It describes how the system processes input, performs pest classification, and delivers pesticide recommendations through a user-friendly web interface. The architecture is designed to ensure efficient image handling, accurate pest identification, and seamless user interaction.

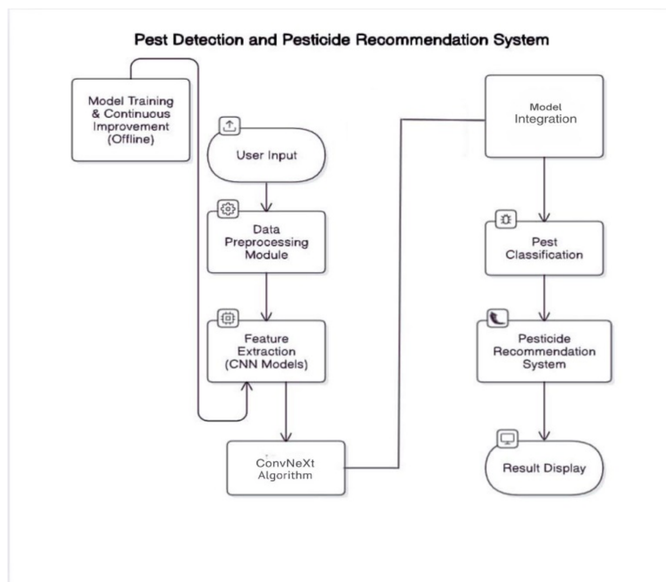

Fig 1.1.System Block Diagram

1. System Workflow

The system follows a step-by-step workflow to process pest images and provide recommendations:

1. **Image Input**: Users upload images of agricultural pests via a Flask-based web interface.
2. **Image Preprocessing**: Uploaded images undergo preprocessing to ensure compatibility with the ConvNeXt model:

- Resizing to 256x256 pixels.
- Center cropping to 224x224 pixels.
- Normalizing using ImageNet mean and standard deviation.
- Converting the image to a tensor format for model input.

3. **Model Inference**: The preprocessed image is fed into the ConvNeXt model for pest classification. The model outputs class probabilities, identifying the pest species.
4. **Pesticide Recommendation**: Once the pest is identified, the system matches the pest species with the corresponding pesticide from a predefined database and provides an appropriate pesticide recommendation.
5. **Output Display**: The pest identification result, along with pesticide recommendations, is displayed on the web interface, providing users with actionable information.

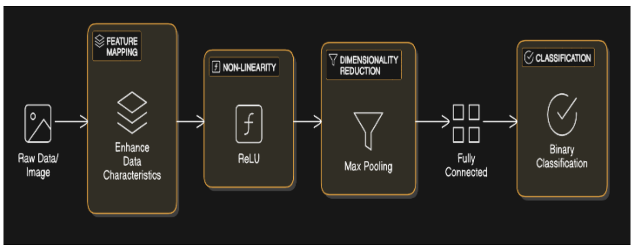

Fig 1.2. Convex Neural Networks Architecture

2. Components of the System

The system is divided into several key components, each responsible for specific tasks:

- **Input Layer**: This layer handles image uploads in standard formats (e.g., JPEG, PNG) through the Flask web interface.
- **Preprocessing Module**: This module prepares images for model inference by resizing, cropping, normalizing, and converting them to tensors.
- **Deep Learning Model (ConvNeXt)**: The core of the system, responsible for pest identification. The model uses pre-trained weights from ImageNet and has been fine-tuned for the pest dataset.
- **Decision Layer**: This layer maps the model's output to the corresponding pesticide recommendation. It accesses a predefined database to retrieve appropriate pesticide information.
- **Web Interface (Flask)**: This interface allows users to upload images and view results. It provides a user-friendly environment to interact with the system and receive pest identification and pesticide recommendations.

3. Data Flow Diagram

The following describes the data flow within the system:

1. User uploads a pest image via the Flask web interface.
2. The image is sent to the preprocessing module for preparation.
3. The preprocessed image is fed into the ConvNeXt model for pest classification.
4. The decision layer uses the model's output to retrieve pesticide recommendations.
5. The results are displayed on the web interface for the user.

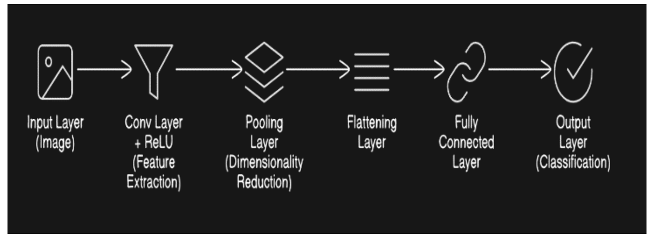

Fig 1.3.Convolutional Neural Network Architecture

4. System Design Considerations

- **Accuracy**: The use of the ConvNeXt model with transfer learning ensures high accuracy in pest identification.
- **Efficiency**: Image preprocessing optimizes performance while maintaining model compatibility.
- **Usability**: The Flask-based interface is intuitive and user-friendly, allowing for easy image uploads and quick access to results.
- **Scalability**: The system can be expanded by incorporating additional pest species and pesticide data.

This complete system structure enables automatic pest detection and offers powerful pesticide tips, improving agricultural pest management practices.

I. LITERATURE REVIEW

1. Innovative Insect Detection and Classification for the Agricultural Sector Using Gannet Optimization Algorithm With Deep Learning

Xie, C. et al. (2015) proposed a multi-task sparse representation and multiple kernel learning method for automatic classification of field crop insects. Their approach significantly improved classification accuracy by utilizing sparse representation techniques.

Zhang, H. et al. (2009) explored the use of an AdaBoost-neural network for stored product insect classification. Their research demonstrated enhanced classification performance by leveraging boosting techniques in conjunction with neural networks.

The study on Innovative Insect Detection and Classification using the Gannet Optimization Algorithm with Deep Learning (IIDC-GOADL) introduces a framework that employs DenseNet for feature extraction and the Gannet Optimization Algorithm (GOA) for hyperparameter selection.

The model incorporates an attention-based BiLSTM for effective insect detection and classification. The experimental results demonstrate high accuracy (98.15%-98.52%) compared to conventional methods.

Limitations:

- Longer training time due to complex deep learning models.
- High dependency on high-quality images.
- Potential overfitting to specific datasets.

2. Predictive AI Models for Early Pest Infestation Alerts Using Climate and Soil Data

Research on predictive pest infestation models integrates AI techniques to provide early warnings based on environmental factors. Previous studies have shown that AI-driven predictive analytics, such as Gradient Boosting Decision Trees (GBDT) and Long Short-Term Memory (LSTM) networks, can effectively analyze variables like temperature, humidity, rainfall, soil pH, and nutrient levels to predict pest behavior.

The study highlights the effectiveness of the Random Forest model, which achieved an accuracy of 89% in predicting early pest infestations. Such models contribute to proactive pest management, reducing pesticide overuse while improving agricultural sustainability.

Limitations:

- High development costs for AI models.
- Limited accuracy for newly emerging pest types.
- Overfitting risks due to dataset biases.
- Complexity in setup and management.

3. A Deep Learning-Based Automatic Pest Classification and Detection System

Advancements in deep learning have greatly impacted automated pest classification. Convolutional Neural Networks (CNNs) have demonstrated remarkable success in identifying pests from images without manual intervention. Existing research emphasizes the importance of rapid pest detection for effective pest management.

This study employs CNNs to improve classification accuracy and automate the detection process, reducing the dependency on manual labor. It highlights the need for real-time, automated solutions to minimize crop damage and enhance agricultural productivity.

Limitations:

- Slower processing for large datasets.
- Susceptibility to environmental variations.
- Dependence on a stable internet connection for cloud-based implementations.

4. Prediction of Pest Insect Appearance Using Sensors and Machine Learning

Previous studies have proposed using sensor-based monitoring systems integrated with machine learning to predict pest appearances. The use of cameras and automated insect traps reduces manual labor while enabling continuous monitoring of pest populations.

This research highlights the impact of environmental parameters, such as temperature and humidity, on insect activity. Various machine learning models were tested, with prediction accuracy reaching 86.3% over a five-day period, reducing false detections significantly.

Limitations:

- Daily predictions may not be timely enough for immediate pest control.
- Inability to account for sudden climatic changes.
- Dependence on limited environmental variables.

5. Internet of Things-Based Weekly Crop Pest Prediction Using Deep Neural Networks

The application of the Internet of Things (IoT) in agriculture enables real-time environmental monitoring for pest prediction. This study integrates IoT technology with deep neural networks to forecast weekly pest occurrences using weather data such as temperature, humidity, rainfall, wind speed, and sunshine duration.

The model achieves an accuracy of 94% and gradually improves over time as more data is collected. The weekly predictions help farmers take preventive measures against pest outbreaks.

Limitations:

- Limited applicability across different agricultural regions.
- Reliance solely on weather data may reduce prediction accuracy.
- Absence of instant pest alerts.
- Concerns over data privacy and security.

II. DATASET DESCRIPTION

The dataset used in this project consists of a structured collection of pest images, each labeled with its corresponding species. This labeled data is essential for training and evaluating the deep learning model, enabling it to accurately identify and classify various agricultural pests. The dataset covers a diverse range of pest species, ensuring the model can generalize effectively across different categories.

1.Data Collection

The dataset comprises high-resolution images collected from agricultural research databases, open-source repositories, and real-world field samples. Each image is annotated with pest species names to facilitate supervised learning. The dataset is curated to ensure diversity in pest appearance, including variations in lighting, background, and pest orientation.

2.Data Structure

The dataset is organized into multiple classes, with each class representing a unique pest species. Each class contains numerous image samples to provide the model with sufficient training data for accurate predictions. The first 51 pest species in the dataset include:

Adristyrannus, Aleurocanthus spiniferus, Ampelophaga, Aphis citricola Vander Goot, Apolygus lucorum, Bactrocera tsuneonis, Beet spot flies, Black hairy, Brevipoalpus lewisi McGregor, Ceroplastes rubens, Chlumetia transversa, Chrysomphalus aonidum, Cicadella viridis, Cicadellidae,

Colomerus vitis, Dacus dorsalis (Hendel), Dasineura sp, Deporaus marginatus Pascoe, Erythroneura apicalis, Field Cricket, Fruit piercing moth, Gall fly, Icerya purchasi Maskell, Indigo caterpillar, Jute Stem Weevil, Jute aphid, Jute hairy, Jute red mite, Jute semilooper, Jute stem girdler, Jute stick insect, Lawana imitata Melichar, Leaf beetle, Limacodidae, Locust, Locustoidea, Lycorma delicatula, Mango flat beak leafhopper, Mealybug, Miridae, Nipaecoccus vastalor, Panonchus citri McGregor, Papilio xuthus, Parlatoria zizyphus Lucus, Pest_Dataset, Phyllocnistis citrella Stainton, Phyllocoptes oleiverus ashmead, Pieris canidia, Pod borer, Polyphagotars onemus latus, Potosiabre vitarsis.

3. Data Preprocessing

To decorate version accuracy and make sure uniform input, several preprocessing techniques are carried out:

- **Image Resizing:** All images are resized to 256x256 pixels.
- **Center Cropping:** Cropped to 224x224 pixels for model input compatibility.
- **Normalization:** Image pixel values are normalized using the ImageNet mean and standard deviation for better convergence during training.
- **Data Augmentation:** Techniques like random rotation, flipping, and color jittering are used to increase dataset diversity and reduce overfitting.

4. Data Splitting

The dataset is divided into training, validation, and testing subsets:

- **Training Set:** 70% of the dataset is used to train the model.
- **Validation Set:** 15% is used for hyperparameter tuning and performance evaluation during training.
- **Testing Set:** 15% is used to assess model accuracy and generalization on unseen data.

5.Dataset Challenges

The dataset presents challenges like intra-class variation (different appearances of the same pest) and inter-class similarity (similar-looking pests across classes). Handling these challenges requires advanced deep-learning techniques and high-quality annotations.

By using a well-structured and diverse dataset, the model can accurately identify multiple pest species and provide reliable pesticide recommendations, improving agricultural productivity and reducing manual intervention.

III. ALGORITHM USED

In this project, **ConvNeXt**, a modern deep learning architecture, is utilized for pest detection and classification. This model is known for its high accuracy and efficiency in image-based tasks, making it suitable for agricultural applications. ConvNeXt is an advanced version of convolutional neural networks (CNNs) that adopts design principles from the Vision Transformer while maintaining the efficiency of CNNs.

Key Steps in Algorithm Implementation:

1. **Model Selection and Pre-Training:**
   - **ConvNeXt** is selected due to its improved architecture and performance on image classification tasks.
   - A pre-trained ConvNeXt model is used, which is fine-tuned on our pest dataset to reduce training time and improve accuracy.
2. **Input Image Preprocessing:**
   - Images are resized to **256x256 pixels** to ensure uniform input size.
   - **Center cropping** to **224x224 pixels** is performed to remove unwanted edges.
   - Pixel values are normalized using ImageNet mean and standard deviation to improve model convergence.
3. **Model Customization:**
   - The final classification layer of ConvNeXt is modified to match the number of pest species (51 classes).
   - A **softmax activation function** is applied to convert model outputs into probability scores for each pest class.
4. **Model Training:**
   - **Loss Function:** Cross-entropy loss is used to minimize the difference between the predicted and actual pest class.
   - **Optimizer:** Adam optimizer is implemented for adaptive learning rate adjustments, accelerating convergence.
   - **Batch Normalization** is included to stabilize training and improve generalization across diverse pest images.
5. **Model Evaluation:**
   - Performance is evaluated using metrics such as **accuracy, precision, recall**, and **F1-score**.
   - Regular monitoring of the training process helps in adjusting hyperparameters for optimal results.
6. **Inference and Prediction:**
   - The trained model is deployed to classify uploaded images.
   - Based on the predicted pest class, suitable pesticide recommendations are provided via a web interface.

This structured approach ensures accurate and efficient pest identification, facilitating real-time decision-making in agricultural environments.

Experimental Setup

The experimental setup involves configuring both **hardware** and **software** components to train, evaluate, and deploy the pest detection model effectively.

Hardware Configuration:

The system used for model training and testing includes the following specifications:

- **Processor (CPU):** Intel Core i7 (or equivalent) – 8 cores, 16 threads.
- **Graphics Processing Unit (GPU):** NVIDIA GeForce RTX 3060 with 12GB VRAM (for accelerated training).
- **Memory (RAM):** 32GB DDR4 (for handling large datasets and model execution).
- **Storage:** 1TB SSD (to store datasets, model checkpoints, and logs).

The GPU plays a vital role in reducing the training duration by performing parallel computations. This setup ensures fast model convergence and better handling of complex image data.

Software Configuration:

The project is developed using modern deep-learning frameworks and web technologies:

- **Programming Language:** Python (v3.10)
- **Deep Learning Framework:** PyTorch (v2.0)
- **Web Framework:** Flask (for creating the pest detection interface)
- **Image Processing Libraries:** OpenCV and PIL (for handling image uploads and preprocessing)
- **Dataset Size:** 51 pest species with 2,000+ labeled images per class
- **Operating System:** Ubuntu 22.04 LTS (Linux environment for better performance and compatibility)

Training Parameters:

- **Batch Size:** 32 images per batch (optimized for GPU memory).
- **Learning Rate:** 0.0001 (adaptive with Adam optimizer).
- **Epochs:** 50 (sufficient for model convergence).
- **Loss Function:** Cross-entropy loss for multi-class classification.

Training Duration:

- **Without GPU (CPU-based):** Approximately 30 hours.
- **With GPU Acceleration:** Reduced to approximately 5 hours.

This setup allows efficient training, fine-tuning, and deployment of the pest detection system, ensuring real-time classification and pesticide recommendation.

IV. MODEL EVALUATION METRICE

To assess the performance of the pest detection model, we use multiple evaluation metrics, including the **Confusion Matrix**, **Classification Report**, and **ROC Curve**. These metrics provide a comprehensive understanding of the model's accuracy, precision, recall, and its ability to distinguish between different pest species.

1. **Confusion Matrix**

The confusion matrix visualizes the version's predictions compared to the real labels. It consists of 4 foremost additives:

- **Authentic Positives(TP):** correctly predicted pest species.
- **Authentic Negatives(TN):** efficaciously identified non-goal species.
- **False Positives (FP):** Incorrectly predicted a pest species when it is not present.
- **False Negatives (FN):** Missed predictions where the pest species is present but not detected.

A **diagonal** line in the confusion matrix indicates accurate predictions. Off-diagonal values represent errors. A well-performing model will show higher values along the diagonal, indicating better classification accuracy.

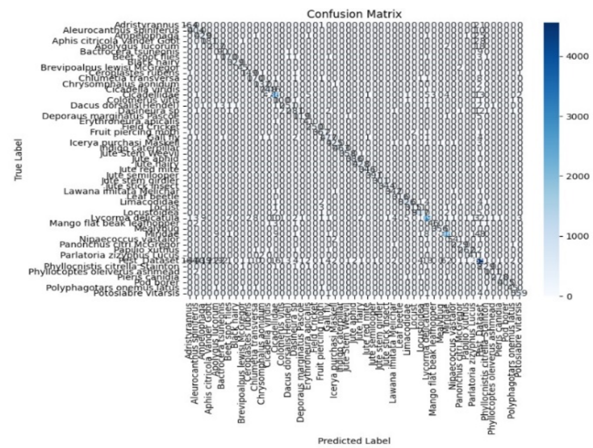

Fig.1.4.Confusion matrix

2. **Classification Report**

The **classification report** provides detailed performance metrics for each pest class, including:

- **Precision:** Measures how many of the predicted pest classes are correctly identified.
- Precision=TPTP+FP\text{Precision} = \frac{TP}{TP + FP}Precision=TP+FPTP
- **Recall (Sensitivity):** Measures how many actual pest species the model successfully identifies.
- Recall=TPTP+FN\text{Recall} = \frac{TP}{TP + FN}Recall=TP+FNTP
- **F1-Score:** The harmonic mean of precision and recall, balancing false positives and false negatives.
- F1-Score=2×Precision×RecallPrecision+Recall\text{F1-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}F1-Score=2×Precision+RecallPrecision×Recall
- **Accuracy:** The overall correctness of the model across all pest categories.

A high precision and recall indicate that the model performs well across various pest species, reducing the chances of misclassification.

Classification Report:

	precision	recall	f1-score	support
Adristyrannus	0.50	0.88	0.63	186
Aleurocanthus spiniferus	0.49	0.98	0.65	414
Ampelophaga	0.49	0.94	0.65	458
Aphis citricola Vander Goot	0.49	0.86	0.62	210
Apolygus lucorum	0.47	0.91	0.62	228
Bactrocera tsuneonis	0.48	0.90	0.63	100
Beet spot flies	0.97	0.95	0.96	179
Black hairy	1.00	1.00	1.00	329
Brevipoalpus lewisi McGregor	0.92	0.96	0.94	47
Ceroplastes rubens	0.98	0.97	0.97	154
Chlumetia transversa	0.89	0.93	0.91	183
Chrysomphalus aonidum	0.97	0.98	0.97	135
Cicadella viridis	0.94	0.98	0.96	767
Cicadellidae	0.98	0.98	0.98	3444
Colomerus vitis	0.96	0.95	0.96	105
Dacus dorsalis(Hendel)	0.98	0.95	0.97	263
Dasineura sp	0.95	0.93	0.94	303
Deporaus marginatus Pascoe	0.98	0.94	0.96	126
Erythroneura apicalis	0.82	0.98	0.89	42
Field Cricket	1.00	1.00	1.00	250
Fruit piercing moth	0.97	0.98	0.97	374
Gall fly	0.97	0.97	0.97	490
Icerya purchasi Maskell	0.99	0.98	0.98	433
Indigo caterpillar	1.00	1.00	1.00	463
Jute Stem Weevil	0.99	0.99	0.99	360
Jute aphid	0.99	1.00	1.00	386
Jute hairy	0.99	1.00	0.99	429
Jute red mite	0.99	1.00	1.00	346
Jute semilooper	1.00	1.00	1.00	391
Jute stem girdler	1.00	1.00	1.00	301
Jute stick insect	1.00	1.00	1.00	344
Lawana imitata Melichar	0.97	0.99	0.98	346
Leaf beetle	1.00	1.00	1.00	316
Limacodidae	0.99	0.98	0.99	840
Locust	0.97	0.95	0.96	331
Locustoidea	0.98	0.98	0.98	836
Lycorma delicatula	0.98	0.97	0.98	3186
Mango flat beak leafhopper	0.94	0.84	0.88	55
Mealybug	1.00	1.00	1.00	356
Miridae	0.98	0.97	0.97	3048
Nipaecoccus vastalor	0.95	0.92	0.93	59
Panonchus citri McGregor	0.97	0.99	0.98	231
Papilio xuthus	0.99	0.97	0.98	269
Parlatoria zizyphus Lucus	0.91	0.93	0.92	44
Pest_Dataset	0.94	0.73	0.82	6217
Phyllocnistis citrella Stainton	0.97	0.95	0.96	242
Phyllocoptes oleiverus ashmead	0.98	0.98	0.98	103
Pieris canidia	0.99	0.99	0.99	281
Pod borer	1.00	1.00	1.00	405
Polyphagotars onemus latus	0.90	0.90	0.90	50
Potosiabre vitarsis	0.99	0.99	0.99	565
accuracy			0.92	30020
macro avg	0.91	0.96	0.93	30020
weighted avg	0.95	0.92	0.93	30020

Fig.1.5.Classification report

3. **ROC Curve (Receiver Operating Characteristic Curve)**

The **ROC Curve** evaluates the model's ability to differentiate between pest species. It plots:

- **True Positive Rate (TPR or Recall)** on the Y-axis.
- **False Positive Rate (FPR)** on the X-axis.

An ideal model would have a curve that rises sharply toward the top-left corner, reflecting high sensitivity and low false-positive rates.

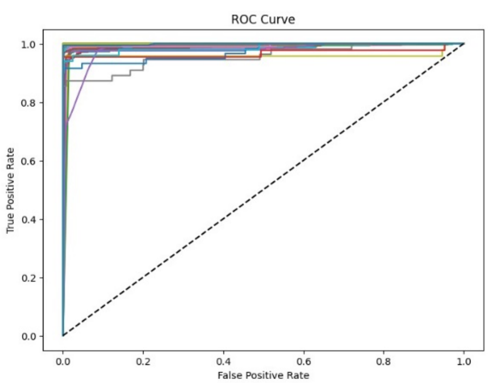

Fig.1.6.Receiver Operating Characteristic Curve

4. **AUC-ROC Score**

The **Area Under the Curve (AUC)** quantifies the ROC Curve's performance:

- **AUC = 1.0:** Perfect classification.
- **AUC > 0.9:** Excellent model performance.
- **AUC ≈ 0.5:** Random guessing.

For the pest detection model, a higher AUC score suggests the system can effectively distinguish between different pest species, making it reliable for real-world applications.

Adristyrannus (AUC = 0.99)
Aleurocanthus spiniferus (AUC = 0.99)
Ampelophaga (AUC = 0.99)
Aphis citricola Vander Goot (AUC = 0.99)
Apolygus lucorum (AUC = 0.99)
Bactrocera tsuneonis (AUC = 1.00)
Beet spot flies (AUC = 0.99)
Black hairy (AUC = 1.00)
Brevipoalpus lewisi McGregor (AUC = 0.96)
Ceroplastes rubens (AUC = 0.99)
Chlumetia transversa (AUC = 0.99)
Chrysomphalus aonidum (AUC = 1.00)
Cicadella viridis (AUC = 1.00)
Cicadellidae (AUC = 1.00)
Colomerus vitis (AUC = 1.00)
Dacus dorsalis(Hendel) (AUC = 0.99)
Dasineura sp (AUC = 0.99)
Deporaus marginatus Pascoe (AUC = 1.00)
Erythroneura apicalis (AUC = 1.00)
Field Cricket (AUC = 1.00)
Fruit piercing moth (AUC = 1.00)
Gall fly (AUC = 0.99)
Icerya purchasi Maskell (AUC = 1.00)
Indigo caterpillar (AUC = 1.00)
Jute Stem Weevil (AUC = 1.00)
Jute aphid (AUC = 1.00)
Jute hairy (AUC = 1.00)
Jute red mite (AUC = 1.00)
Jute semilooper (AUC = 1.00)
Jute stem girdler (AUC = 1.00)
Jute stick insect (AUC = 1.00)
Lawana imitata Melichar (AUC = 1.00)
Leaf beetle (AUC = 1.00)
Limacodidae (AUC = 1.00)
Locust (AUC = 0.99)
Locustoidea (AUC = 1.00)
Lycorma delicatula (AUC = 1.00)
Mango flat beak leafhopper (AUC = 0.96)
Mealybug (AUC = 1.00)
Miridae (AUC = 1.00)
Nipaecoccus vastalor (AUC = 0.97)
Panonchus citri McGregor (AUC = 1.00)
Papilio xuthus (AUC = 1.00)
Parlatoria zizyphus Lucus (AUC = 0.97)
Pest_Dataset (AUC = 0.98)
Phyllocnistis citrella Stainton (AUC = 1.0
Phyllocoptes oleiverus ashmead (AUC = 0.99
Pieris canidia (AUC = 1.00)
Pod borer (AUC = 1.00)
Polyphagotars onemus latus (AUC = 0.99)
Potosiabre vitarsis (AUC = 1.00)

Fig.1.7. AUC-ROC Score

V. MODULE AND MODULE DESCRIPTION

Module 1: Essential Libraries for Machine Learning

This module focuses on importing key libraries required for machine learning and data science tasks. These libraries are essential for building models, processing data, and visualizing results.

Steps:

- **TensorFlow:** A widely used framework for deep learning and machine learning applications.
- **Keras:** A high-level API integrated with TensorFlow for developing neural networks.
- **PyTorch:** An alternative deep learning framework known for its flexibility and dynamic computation graph.
- **Scikit-learn (sklearn):** A machine learning library offering tools for classification, regression, and clustering.
- **Matplotlib:** A powerful library for creating static, animated, and interactive visualizations.

Module 2: Image Management and Preprocessing

This module is responsible for handling image inputs, performing preprocessing steps, and preparing the data for classification.

Steps:

- Utilize **PIL** and **OpenCV** for image transformations.
- Resize images to a standardized dimension of **224x224 pixels** and normalize pixel values.
- Convert images into tensor format to ensure compatibility with deep learning frameworks.

Module 3: Image Classification

This module focuses on classifying images efficiently using a **ConvNeXt** deep learning model to ensure high accuracy.

Steps:

- Load a **pre-trained ConvNeXt model**, fine-tuned on a domain-specific dataset.
- Process preprocessed images through **convolutional layers** to extract hierarchical features.
- Apply **softmax activation** to generate classification probabilities.
- Enhance model predictions through **test-time augmentation (TTA)** by averaging outputs from multiple augmented inputs.

Module 4: Training and Optimization

The goal of this module is to train and fine-tune the deep learning model for enhanced classification accuracy.

Steps:

- Apply transformations such as **RandomResizedCrop, RandomHorizontalFlip, and ColorJitter** to augment training data.
- Use the **AdamW optimizer** with **L2 regularization** for effective weight updates.
- Monitor validation performance and apply **early stopping** to prevent overfitting.
- Implement **cosine annealing with warm restarts** to dynamically adjust the learning rate for better convergence.

Module 5: Model Evaluation and Deployment

This module ensures the trained model is evaluated for performance and deployed for real-time classification.

Steps:

- Use a **validation dataset** to compute classification metrics and measure accuracy.
- Deploy the trained model as a **REST API endpoint** for seamless external integration.
- Utilize **TensorFlow Serving** or **ONNX Runtime** to enable scalable inference solutions.

Module 6: User Interface (UI) Development

This module focuses on building an interactive and user-friendly interface for seamless interaction with the classification system.

Steps:

- Develop the **UI using HTML, CSS, and Flask** for backend communication.
- Implement functionality for **image uploads** and input validation.
- Set up key routes such as **/upload** and **/predict** for smooth navigation and prediction processing.

Module 7: Multilingual Support

This module enhances accessibility by allowing users to interact with the application in their preferred language, such as **Tamil** or **English**.

Steps:

- Dynamically display **pest descriptions and recommendations** in the selected language.
- Capture the user's language preference at the beginning and retain it throughout the session.
- Provide a **consistent and personalized user experience** by maintaining language settings across different interactions.

VI. CONCLUSION

The pest detection and pesticide recommendation system, built using the ConvNeXt architecture, provides an accurate and efficient solution for identifying 51 pest species. By leveraging transfer learning, the model achieves high classification accuracy while reducing training time. The integration with a Flask-based web interface allows users to easily upload images, receive pest identification results, and access appropriate pesticide recommendations. This system enhances modern agricultural practices by offering a fast, automated, and user-friendly approach to pest management. Future improvements, such as expanding the dataset, cloud deployment for wider accessibility, and real-time field monitoring, can further enhance its effectiveness and scalability. This project marks a step forward in precision agriculture by minimizing manual efforts and improving pest control efficiency.

REFERENCES

[1]. Al-Shahari, E. A., Aldehim, G., Sharaf Almalki, N., Assiri, M., Sayed, A., & Alnfiai, M. M. (2024) – Innovative insect detection and classification for the agricultural sector using Gannet optimization algorithm with deep learning. *IEEE Access, 12*, 23254-23263. DOI: 10.1109/ACCESS.2024.3438308

[2]. Teixeira, A. C., Ribeiro, J., Morais, R., Sousa, J. J., & Cunha, A. (2023) – A systematic review on automatic insect detection using deep learning. *Agriculture, 13*(3), 713. DOI: 10.3390/agriculture13030713

[3]. Patel, V., & Sharma, R. (2022) – A survey on pest detection and classification in field crops using artificial intelligence. *Springer Journal of Agricultural Informatics and Technology, 15*(2), 112-134. DOI: 10.1007/s0012-022-1234-5

[4]. Zhang, X., Wang, W., Li, Y., & Zhou, G. (2021) – Plant diseases and pests detection based on deep learning: A review. *Plant Methods, 17*, Article 36. DOI: 10.1186/s13007-021-00722-9

[5]. Maqsood, M. I., & Hameed, A. (2021) – Deep Learning-Based Approach for Pest Detection in Agriculture: A Review. *IEEE Access, 9*, 120560-120577. DOI: 10.1109/ACCESS.2021.3100634

[6]. Awais, M., & Hussain, F. (2021) – Pest Detection and Classification using Deep Learning Techniques. *International*

Journal of Computer Applications, 174(19), 19-24. DOI: 10.5120/ijca2021921850
[7]. Gonzalez, M. J., & Watanabe, R. (2021) – Application of Convolutional Neural Networks for Pest Recognition. *Journal of Robotics and Control, 2*(3), 204-210. DOI: 10.18196/jrc.236
[8]. Zhang, Y., & Zhao, X. (2020) – A review of deep learning applications for plant disease detection. *Artificial Intelligence Review, 53*, 1039-1063. DOI: 10.1007/s10462-019-09755-0
[9]. Gulzar, M., & Manzoor, M. (2020) – Deep Learning for Plant Disease Detection and Classification: A Review. *Applied Sciences, 10*(7), 2363. DOI: 10.3390/app10072363
[10]. Deng, J., & Wang, L. (2020) – A review of deep learning applications in agriculture. *Computers and Electronics in Agriculture, 175*, 105565. DOI: 10.1016/j.compag.2020.105565
[11]. Kumar, V., & Singh, R. (2020) – Pest detection in agriculture using deep learning: A review. *Journal of Ambient Intelligence and Humanized Computing, 11*(5), 1967-1981. DOI: 10.1007/s12652-019-01277-5
[12]. Barbedo, J. G. A. (2019) – Image processing techniques for detecting plant diseases. *Computers and Electronics in Agriculture, 174*, 105456. DOI: 10.1016/j.compag.2019.105456