RECOMMENDATION OF PESTICIDES BASED ON PEST CLASSIFICATION USING DEEP LEARNING ALGORITHM

A MINI PROJECT REPORT

Submitted by

AKASH A	(DSUG20104012)
ASHOK KUMAR P	(DSUG20104020)
HEMANTHAN G	(DSUG20104057)
JANAHAR V	(DSUG20104059)

In partial fulfilment for the award of the degree

Of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

DHANALAKSHMI SRINIVASAN ENGINEERING COLLEGE

(AUTONOMOUS)

PERAMBALUR-621212

NOVEMBER 2023

BONAFIDE CERTIFICATE

Certified that this project report **" RECOMMENDATION OF PESTICIDES BASED ON PEST CLASSIFICATION USING DEEP LEARNING ALGORITHM "** is the bonafide work of **" AKASH A (DSUG20104012) , ASHOK KUMAR P (DSUG20104020) , HEMANTHAN G (DSUG20104057) , JANAHAR V (DSUG20104059) "** who carried out the project work under my supervision.

SIGNATUTRE

Mrs. R.ARUNAPRIYA .,M.E

SUPERVISOR

Department Of Computer Science And Engineering

Dhanalakshmi Srinivasan Engineering College(Autonomous)

Perambalur-621212.

SIGNATUTRE

Dr. R.GOPI .,PHD

HEAD OF THE DEPARTMENT

Department Of Computer Science And Engineering

Dhanalakshmi Srinivasan Engineering College(Autonomous)

Perambalur-621212.

Submitted for the final viva voice held on _____

INTERNAL EXAMINER

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

We express our gratitude and thanks to our parents first for giving health and sound mind for completing this project. We give all the glory and thanks to our Almighty GOD for Showering upon, the necessary wisdom and grace for Accomplishing this project. It is our pleasant duty to express deep sense of gratitude to our honourable Chairman **Shri. A SRINIVASAN**, for his kind encouragement. We have unique Pleasure in thanking our principle **Dr. D. SHANMUGA SUNDARAM., PHD** for his support. We express our faithful and sincere gratitude to our Dean(Academics) **K.ANBARASAN.M.E.,PHD.,** for providing as an excellent environment to carry out our course successfully. We are deeply indebted to our beloved Head of the Department, **Dr. R.GOPI., PHD, Department of Computer Science and Engineering**, who modelled us both technically and morally for achieving greater success in life. We express my thanks to our project coordinator and Internal Guide **Mrs. R.ARUNAPRIYA .,ME, Department of Computer Science and Engineering** for her valuable suggestions at every stage of our project to complete the Project work successfully . We render our thanks to all **Staff Members** and **programmers** of **Department of Computer Science and Engineering** for their time

ABSTRACT

Agriculture is an important basic industry worldwide, and pests can cause huge losses to crop production in every country. According to research, nearly half of global crop production will be impacted to varying degrees due to pests every year, which seriously affects the regional economy and people's daily lives. Crop pests are a major agricultural problem worldwide because the severity and extent of their occurrence threaten crop yield. However, traditional pest image segmentation methods are limited, ineffective and time-consuming, which causes difficulty in their promotion and application. Deep learning methods have become the main methods to address the technical challenges related to pest recognition. When pests infect a field, they must be recognized in time; therefore, farmers can deliver timely treatment and avoid the spread of pests. Still, a traditional pest recognition schemes have numerous limitations. Most of the generally used techniques are manual study. Here the farmers or experts manually check the field monthly, daily, and weekly for pest or diseases. We propose an improved deep convolution neural network to better recognize crop pests in a real agricultural environment. The proposed network includes parallel attention mechanism module and residual blocks, and it has significant advantages in terms of accuracy and real-time performance compared with other models. And also extend the framework to predict the fertilizers or pesticides based on identified pest.

TABLE OF CONTENTS

CHAPTER NO	TOPIC	PAGE NO
	ABSTRACT	**IV**
	LIST OF ABBREVIATIONS	**VII**
1	**INTRODUCTION**	**1**
2	**LITERATURE SURVEY**	**2**
3	**SYSTEM ANALYSIS**	**7**
	3.1 EXISTING SYSTEM	**7**
	3.2 PROPOSED SYSTEM	**8**
	3.3 SCOPE OF THE PROJECT	**9**
	3.4 SYSTEM ARCHITECTURE	**10**
4	**REQUIREMENT**	**11**
	4.1 HARDWARE REQUIREMENT	**11**
	4.2 SOFTWARE REQUIREMENT	**11**
5	**MODULE DESCRIPTION & LIBRARY DESCRIPTION**	**12**
	5.1 MODULE DESCRIPTION	**12**
	5.2 INSTALLATION	**15**
	5.3 INSTALLING FLASK MODULE	**15**
6	**BUILDING THE API/GUI**	**16**
	6.1 IMPORT LIBRARIES	**16**
	6.2 CREATE THE BUTTON	**17**

7	**SYSTEM TESTING**	**18**
	7.1 SYSTEM TESTING	**18**
	7.2 TESTING TECHNIQUES	**18**
	7.3 SOFTWARE TESTING STRATEGIES	**19**
8	**SOURCE CODE**	**21**
9	**CONCLUSION AND FUTURE ENHANCEMENT**	**32**
10	**REFERENCE**	
	APPENDIX	

LIST OF ABBREVIATIONS

ABBREVIATION	EXPANSION
CNN	**C**ONVOLUTIONAL **N**EURAL **N**ETWORK
RNN	**R**ECURRENT **N**EURAL **N**ETWORK
ML	**M**ACHINE **L**EARNING
DL	**D**EEP **L**EARNING
SVM	**S**UPPORT **V**ECTOR **M**ACHINE
MLP	**M**ULTI-**L**AYER **P**ERCEPTRON
KNN	**K-N**EAREST **N**EIGHBORS
PCA	**P**RINCIPAL **C**OMPONENT **A**NALYSIS
ADL	**A**CTION **D**ESCRIPTION **L**ANGUAGE
GUI	**G**RAPHICAL **U**SER **I**NTERFACE
API	**A**PPLICATION **P**ROGRAMMING **I**NTERFACE
UAT	**U**SER **A**CCEPTANCE **T**ESTING
OAT	**O**PERATIONAL **A**CCEPTANCE **T**ESTING

1. INTRODUCTION

Pest recognition is an important task in agriculture as it helps farmers detect and identify pest infestations in their crops. Traditional methods of pest recognition rely on manual inspection, which can be time-consuming and costly. Deep learning has emerged as a promising approach to automate pest recognition by leveraging the power of artificial neural networks. In this approach, deep learning algorithms are trained on large datasets of labelled images of healthy and pest-infested crops, enabling them to identify and classify pests with high accuracy. By automating pest recognition, farmers can quickly and accurately detect pest infestations, allowing them to take timely action to protect their crops. In this essay, we will explore the use of deep learning in pest recognition, including the challenges and potential benefits of this approach.

Deep learning has shown promising results in pest recognition, with several studies demonstrating its effectiveness in detecting and identifying various types of pests. One study used a deep convolutional neural network (CNN) to classify images of maize leaves infected with the Fall Armyworm, a common pest that can cause significant damage to crops. The CNN achieved an accuracy of 92.5% in detecting pest-infested leaves, outperforming traditional image processing methods. Another study used a deep learning approach to detect tomato diseases caused by various pests and pathogens, including the Tomato Yellow Leaf Curl Virus, the Tomato Mosaic Virus, and the Bacterial Speck Disease. The study used a combination of a CNN and a recurrent neural network (RNN) to classify images of tomato plants based on the presence or absence of disease symptoms. The deep learning approach achieved an accuracy of 96.3%, outperforming traditional machine learning algorithms. Despite its potential benefits, deep learning in pest recognition also faces several challenges. One of the main challenges is the need for large and diverse datasets to train the deep learning algorithms effectively. Collecting and labelling such datasets can be time-consuming and costly, especially in the case of rare or newly emerging pests. Additionally, the performance of deep learning algorithms can be affected by environmental factors, such as lighting and weather conditions, which can affect the quality of the input data.

2. LITERATURE SURVEY

2.1 TITLE: AUTOMATED PEST DETECTION WITH DNN ON THE EDGE FOR PRECISION AGRICULTURE

AUTHOR: ANDREA ALBANESE, 2021

Due to the constant growth of food production demand, in Europe, agriculture is responsible for more than 10% of greenhouse gas emissions and 44% of water consumption nowadays. Chemical treatments (e.g., pesticides) are being intensively used to improve the market penetration of fruit crops, leading to a remarkable impact on pollinators and the planet's ecosystem. Thus, there is an increasing interest in new techniques to lower the water demand and optimize pesticide treatments to preserve natural resources. Recently, researchers have started investigating smart-trap usage for pest detection as a solution to increase the wealth of orchards while lowering pesticide demand. These traps – installed directly inside orchards – can autonomously detect dangerous parasites and alert the farmer to apply targeted pesticide treatments. Thanks to the implementation of sophisticated at-the-edge machine learning algorithms, the smart trap can detect dangerous parasites without remote cloud infrastructure as generally required for machine learning applications. This paper presents a smart trap for pest detection running a Deep Neural Network on edge. The smart trap enables fast detection of pests in apple orchards by using ML algorithms that improve the overall system efficiency [4], [5]. All the computation is done on-the-node, thus slimming down the amount of data transmission and limiting it to a simple notification of few bytes if threats are detected

2.2 TITLE: AN EFFICIENT APPROACH FOR CROPS PESTS RECOGNITION AND CLASSIFICATION BASED ON NOVEL DEEPPESTNET DEEP LEARNING MODEL

AUTHOR: NAEEM ULLAH

Recently, some research studies have been concentrated on pest classification and recognition. The pest classification and recognition research can be divided into ML, DL, and hybrid based approaches. Hybrid methods include techniques that employ both DL and ML techniques. Many pests' classification research works use DL-based techniques, whereas ML-based techniques are rarely used. Below, we highlighted the most recent and relevant research work in automatic pest identification and classification. The unavailability of a large amount of data for training the DL frameworks is one of the challenges when intending to use DL approaches to pest recognition and classification problems. More crop pest data is difficult and expensive to get, both in terms of time and resources. Data augmentation, or increasing the amount of available data without acquiring new data by applying multiple processes to current data, has been proven advantageous in image classification [30]. The ImageNet classifier challenge winners adopted this method and used it academically to improve training data and reduce over fitting. There are many different types of insects, and there are distinctions between larvae and adults. For example, noctuid pests have only been identified during their most dangerous stage of development (i.e., the larval stage). In future, we are interested in expanding the classification size by including more pests' types to be effectively identified by the proposed DeepPestNet framework. This research could help specialists and farmers identify pests more quickly and effectively, thus reducing economic and crop output losses

2.3 TITLE: PLANT DISEASE DETECTION AND CLASSIFICATION BY DEEP LEARNING—A REVIEW

AUTHOR: LILI LI, SHUJUAN ZHANG

The occurrence of plant diseases has a negative impact on agricultural production. If plant diseases are not discovered in time, food insecurity will increase. Early detection is the basis for effective prevention and control of plant diseases, and they play a vital role in the management and decision making of agricultural production. In recent years, plant disease identification has been a crucial issue. Disease-infected plants usually show obvious marks or lesions on leaves, stems, flowers, or fruits. Generally, each disease or pest condition presents a unique visible pattern that can be used to uniquely diagnose abnormalities. Usually, the leaves of plants are the primary source for identifying plant diseases, and most of the symptoms of diseases may begin to appear on the leaves. In most cases, agricultural and forestry experts are used to identify on-site or farmers identify fruit tree diseases and pests based on experience. This method is not only subjective, but also time-consuming, laborious, and inefficient. Deep learning is a branch of artificial intelligence. In recent years, with the advantages of automatic learning and feature extraction, it has been widely concerned by academic and industrial circles. It has been widely used in image and video processing, voice processing, and natural language processing. At the same time, it has also become a research hotspot in the field of agricultural plant protection, such as plant disease recognition and pest range assessment, etc. The application of deep learning in plant disease recognition can avoid the disadvantages caused by artificial selection of disease spot features, make plant disease feature extraction more objective, and improve the research efficiency and technology transformation speed.. In this paper, we present the current trends and challenges for the detection of plant leaf disease using deep learning and advanced imaging techniques

2.4 TITLE: DEEP CONVOLUTIONAL NEURAL NETWORKS FOR TEA TREE PEST RECOGNITION AND DIAGNOSIS

AUTHOR: JING CHEN, 2021

Usually, tea pests diagnosis is based on appearance characteristics of insects. Due to lack of pests biology and identification, farmers are unable to timely diagnose insect pest types, thus causing serious damage to the production process. Furthermore, the availability of agricultural experts is limited. Therefore, an automatic identification system for tea pests can ensure that agricultural producers accurately assess tea pests and thus determine the most economical and effective control methods to employ. This allows farmers to reduce labor costs and losses while ensuring the quality of tea. With the rapid development of agricultural information technology, computer image processing, and pattern recognition technology, image-based classification has been used to identify crop pests. At present, the general methods of image-based crop pest identification are image acquisition, feature extraction, and classifier selection. Insect pest images can be obtained by photographing insect specimens. Finally, the extracted features are led to the classifier (such as SVM, MLP, and KNN) for classification and recognition. The main disadvantage of the above method is that it requires manual extraction in advance, and the selection of features directly affects the final recognition accuracy. In recent years, with the continuous expansion and improvement of the Internet and hardware performance, deep learning theory has developed rapidly. Due to its superior performance in feature extraction, model generalization, and fitting, it has attracted considerable attention from researchers. CNN is widely used in image recognition. The main characteristic of CNN is that it uses a stacked layer of non-linear processing units to extract the features of input images autonomously, resulting in end-to-end feature extraction and classification

2.5 TITLE: PADDY PEST IDENTIFICATION WITH DEEP CONVOLUTIONAL NEURAL NETWORKS

AUTHOR: CHIRANJEEVI MUPPALA, 2021

Despite good success on insect pest recognition in recent years, most researches were not performed in real field scenarios, and have used a small dataset which can pose reliability issues. Some researches involved in complex pre-processing on pest images which make automation difficult. Instead of identifying pests from the complex background in the field, we can exploit the pest trapping mechanisms to identify pests with lower background complexity and is practicable for automation of pest identification. In this paper, we used pest light trap for collecting pest image samples and implemented paddy pest classification using popular AlexNet deep neural network architecture. NVIDIA DIGITS a web-based application for training deep learning models was used for paddy pest classification. It trains CNN proficiently with high accuracy for image segmentation, object detection, and classification tasks. It simplifies operations like managing datasets, designing and training neural networks. With DIGITS advanced visualization features, the neural network performance can be observed in real-time, from that we can select the best performing model for deployment. Its friendly interactive insect pests are one of the major factors which affect crop yield in agriculture. In this paper classification of yellow stem borer, brown planthopper, leaf folder, and green leafhopper pests in the paddy field were investigated using AlexNet neural network algorithm. Various deep learning frameworks (Caffe, TensorFlow, and Torch) and optimization algorithms in NVIDIA DIGITS platform were exploited for comparing the pest classification accuracy. TensorFlow-stochastic gradient descent model performed better compared to other models and achieved 96.9 % validation accuracy. This approach for classification can help detect invasive pests in the paddy field and can be implemented in real-time.

3. SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

Pest recognition is an important problem in agriculture and forestry as it helps in identifying and controlling pests, thereby preventing crop and tree damage. K-means and Support Vector Machine (SVM) algorithms are commonly used for pest recognition due to their effectiveness in clustering and classification, respectively. The existing system for pest recognition using K-means and SVM algorithms involves several steps, starting with image pre-processing to remove noise and enhance image features. Next, features are extracted using the K-means clustering algorithm to cluster similar pixels and extract relevant features such as color, texture, and shape. The most relevant features are selected using techniques such as Principal Component Analysis (PCA) and correlation-based feature selection. In the feature selection step, Principal Component Analysis (PCA) is used to reduce the dimensionality of the feature space, which improves the performance and speed of the SVM algorithm. Additionally, correlation-based feature selection is applied to select the most relevant features based on their correlation with the class label. For classification, the SVM algorithm is trained using a libelled dataset of pest images. The SVM algorithm is chosen because of its ability to handle high-dimensional feature spaces and classify non-linearly separable data. The SVM algorithm learns to differentiate between different pests based on the selected features and assigns a label to each image. Finally, the SVM algorithm is used for classification based on the selected features using a dataset of labelled images of pests. This system can be used in real-time pest recognition applications to help farmers and foresters identify and control pests.

DISADVANTAGES

- Manual interventions are need
- Irrelevant features are used
- Limited datasets are trained
- Binary classification-based implementation

3.2 PROPOSED SYSTEM

The proposed system for pest recognition using CNN algorithm involves collecting a large dataset of pest images from various sources and pre-processing them to remove noise and standardize their size and orientation. A CNN model is then trained on the pre-processed dataset using supervised learning to extract features from the images and classify them into different pest species. The trained model is validated using a separate dataset of images to evaluate its performance and identify any issues. Once the model is trained and validated, it can be deployed to recognize and classify pests in real-time. The use of CNN algorithm provides an effective and efficient approach to pest recognition, which can be applied to a variety of agricultural and forestry settings. The CNN algorithm is particularly effective for image recognition tasks due to its ability to extract and learn features from images automatically. The CNN model consists of several layers, including convolutional layers, pooling layers, and fully connected layers. The convolutional layers extract features from the images, the pooling layers down sample the features, and the fully connected layers perform classification based on the extracted features. The use of a large dataset of pest images is essential for training the CNN model effectively. The dataset should include a variety of pest species, captured under different lighting and environmental conditions. Data augmentation techniques can also be applied to increase the diversity of the dataset and improve the performance of the CNN model. Once the CNN model is trained and validated, it can be deployed to recognize and classify pests in real-time. The system can be integrated with cameras or drones to capture images of pests and classify them automatically. The system can also provide alerts to farmers or foresters when pests are detected, enabling them to take appropriate measures to control the pests and reduce crop or tree damage.

ADVANTAGES

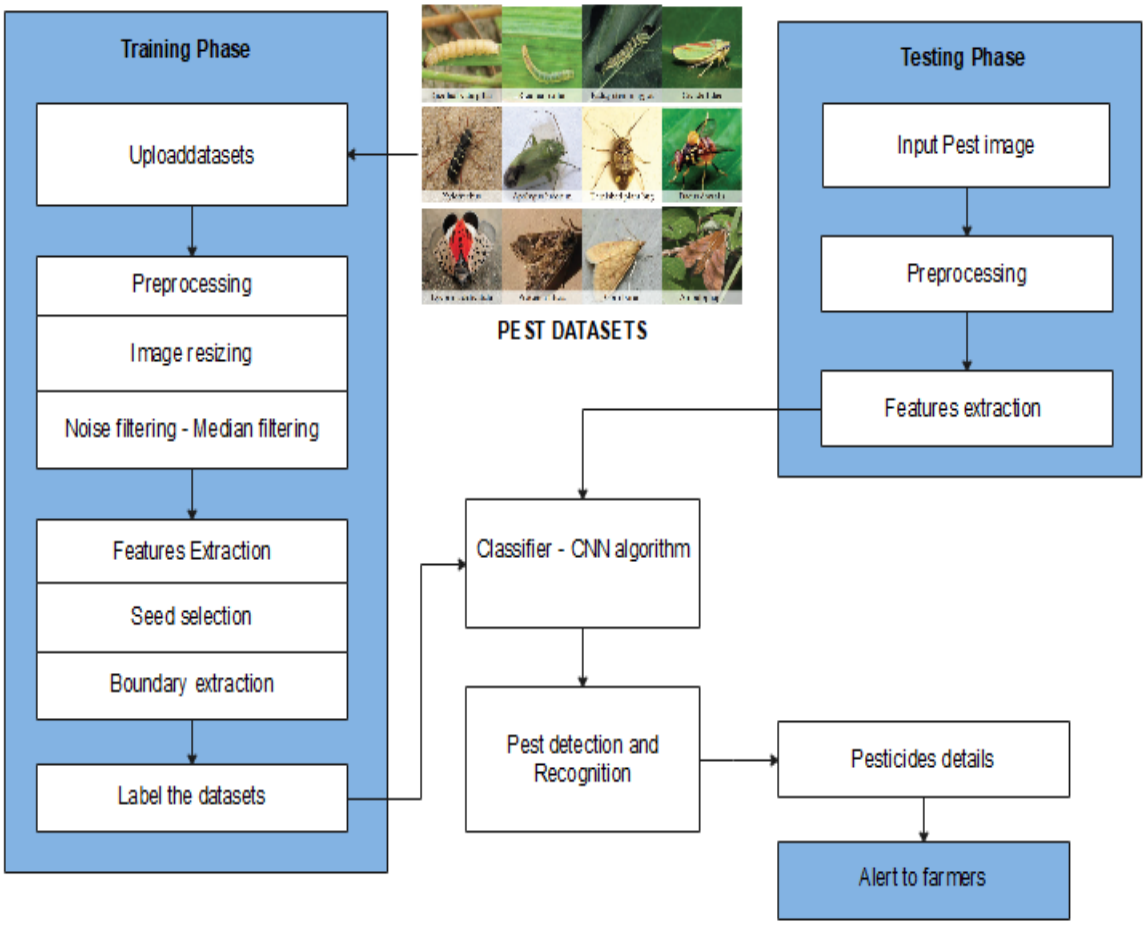
- Automated segmentation
- Relevant features are extracted
- Multiple pests are identified
- Accuracy rate is high

3.3 SCOPE OF THE PROJECT

The research, titled "Recommendation of Pesticides Based on Pest Classification Using Deep Learning Algorithm," has a significant and pertinent scope. Although pests are a persistent challenge to agricultural yields, quality, and sustainability, agriculture remains a vital component of our global economy and food security. Due to the widespread use of broad-spectrum pesticides in traditional pest management techniques, environmental issues and the emergence of pests resistant to pesticides are raised. By utilising deep learning, this initiative seeks to transform pest control. The main goal of this research is to create a deep learning model that is capable of reliably classifying and identifying a wide range of pests in real time, including fungus, insects, and illnesses. This model will be able to quickly and correctly detect and distinguish between different types of pests by utilising Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and other deep learning architectures to analyse photos, environmental data, and other pertinent information. The project's scope will be expanded to include the recommendation of precise and targeted chemical options for pest management after the pests have been categorised. Based on a thorough study of the particular pest species, their life cycles, environmental conditions, and the best and most ecologically friendly insecticides for eradication, suggestions will be made. This strategy lessens the need for broad-spectrum pesticides while simultaneously minimising environmental damage and advancing sustainable farming methods. In addition, the project can involve creating an intuitive platform or application that makes these ideas easily accessible to farmers and other agricultural stakeholders. This will enable them to make knowledgeable decisions about pest management, maximising crop protection and reducing the unfavourable effects of overuse of pesticides. This initiative has far-reaching ramifications, ranging from enhancing food security and agricultural yields to lessening the environmental impact of agriculture. It provides a win-win solution for farmers, consumers, and the environment, and it is in line with the global movement towards precision agriculture and sustainable farming practises. The information and understanding gained from this initiative might change the direction of pest control in the future, making it more effective, long-lasting, and ecologically friendly.

3.4 SYSTEM ARCHITECTURE

A system architecture or systems architecture is the conceptual model that defines the structure, behavior, and more views of a system. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures and behaviors of the system. System architecture can comprise system components, the externally visible properties of those components, the relationships (e.g. the behavior) between them. It can provide a plan from which products can be procured, and systems developed, that will work together to implement the overall system. There have been efforts to formalize languages to describe system architecture; collectively these are called architecture description languages (ADLs).



4. REQUIREMENT

4.1 HARDWARE REQUIREMENT

- Processor : Dual core processor 2.6.0 GHZ
- RAM : 4 GB
- Hard disk : 320 GB
- Compact Disk : 650 Mb
- Keyboard : Standard keyboard
- Monitor : 15 inch color monitor

4.2 SOFTWARE REQUIREMENT

- Operating system : Windows OS
- Front End : Python
- IDLE : Python 2.7 IDLE

5. MODULE DESCRIPTION & LIBRARY DISCRIPTION

5.1 MODULE DESCRIPTION

- Dataset Collection
- Preprocessing
- Features Extraction
- Model Training
- Pest Classification

DATASETS COLLECTION

Collecting a diverse and representative dataset is crucial for the success of the pest recognition system. The dataset should include images of different pest species, captured under different lighting and environmental conditions. Digital image archives can be searched for images of pests. These archives may include images captured by researchers, farmers, or other stakeholders. Dataset acquisition refers to the process of obtaining data for use in various applications, such as machine learning, data analysis, and research. Synthetic datasets can be generated using computer graphics techniques. This approach can provide a large number of images of pests in different poses, lighting, and environmental conditions. In this module, we can input the pest datasets that are collected from KAGGLE web sources. It contains the various pest details as in image format

PREPROCESSING

Image preprocessing refers to the process of preparing digital images for further analysis or processing, such as computer vision or image recognition tasks. Image preprocessing steps can include:

- Noise reduction: Removing any unwanted noise or artifacts from the image.
- Image normalization: Adjusting the image brightness and contrast to ensure consistent image quality.
- Image resizing: Changing the size of the image to a specified dimension.
- Image cropping: Removing any unnecessary portions of the image.

Median filtering is a pre-processing technique that can be used to remove noise from images. The technique involves replacing the value of each pixel in an image with the median value of

its neighbouring pixels. Median filtering is particularly effective for removing salt-and-pepper noise, which appears as white or black pixels scattered randomly throughout the image. In the pest recognition system, median filtering can be applied as a pre-processing step to improve the quality of the images in the dataset. The noisy images can be filtered using a median filter to remove the salt-and-pepper noise and produce cleaner images for further processing.

FEATURES EXTRACTION

Image feature extraction using Convolutional Neural Networks (CNNs) is a technique used in computer vision to extract meaningful and informative features from digital images. In this process, a pre-trained CNN is used to extract a set of features from the input image. These features are learned and optimized by the CNN during its training process, and can be used for various computer vision tasks, such as image classification, object detection, and segmentation. In this module, we can extract the features such as shape, color and texture features values. CNN algorithm is particularly effective for feature extraction due to its ability to automatically learn and extract features from images. The CNN model consists of several layers, including convolutional layers, pooling layers, and fully connected layers. The convolutional layers extract features from the images, the pooling layers down sample the features, and the fully connected layers perform classification based on the extracted features. CNN algorithm has become the state-of-the-art approach for feature extraction from images due to its ability to learn and extract hierarchical features from images. The convolutional layers in the CNN model are designed to learn and extract features that are relevant for classification, while the fully connected layers are used to map the extracted features to the output classes. One of the main advantages of using CNN algorithm for feature extraction is that it does not require manual feature engineering. Instead, the CNN model automatically learns and extracts features from the images during training, allowing for more accurate and efficient feature extraction.

MODEL TRAINING

Model training using Convolutional Neural Networks (CNNs) is a process in machine learning where a CNN is learned from a dataset for image classification, object detection, or other computer vision tasks. Here's a general overview of the process:

- Data preparation: Prepare the image dataset for training, including splitting the data into training and validation sets, data augmentation, and normalizing the data.
- Define the model architecture: Choose a suitable CNN architecture and design a custom architecture based on the specific requirements of the task.

- Compile the model: Define the optimizer, loss function, and metrics to be used during training.
- Train the model: Train the model on the training data by feeding the images into the network and updating the weights using the defined optimizer, loss function, and metrics. The training process involves multiple iterations, also known as epochs, over the training data until the desired level of accuracy is achieved.

PEST CLASSIFICATION

In this module classify the pest using CNN framework and it includes the steps as

- Validate the model: Evaluate the model's performance on the validation data to avoid over fitting and to ensure that the model is generalizing well to new data.
- Fine-tune the model: Based on the validation performance, adjust the model architecture, hyper parameters, or training procedure to improve the model's performance. Repeat this process until the desired accuracy is achieved.
- Test the model: Finally, evaluate the model's performance on a separate test set to obtain an unbiased estimate of its performance on unlabeled data.
- Finally provide the pesticides to recognized pest

5.2 INSTALLATION

Tkinter is a built-in library for creating graphical user interfaces (GUIs) in Python. It's included with most Python installations, so you typically don't need to install it separately. If Tkinter is already installed on your system. You can do this by running the following command in your Python environment:

import tkinter

After installing or verifying Tkinter, you can use the same import command as in step 1 to ensure that it's available and functioning properly.

5.3 INSTALLING FLASK MODULE

To install a Python module or package, you can use the pip package manager, which is the standard tool for managing Python packages. Open a terminal or command prompt on your system. To install a specific Python module or package, use the pip install command followed by the name of the package. For example, to install a package called example-package, run:

pip install tkinter

pip install keras

6. BUILDING THE API/GU

Tkinter is a standard Python library for creating GUI applications. It is included with most Python installations. To ensure you have it installed, you can run the following command: Create a Python script (e.g., weapon_detection_gui.py) and add the following code: Run your script using a Python interpreter, and the GUI application will open, allowing you to open images for weapon detection.

6.1 IMPORT LIBRARIES

To create a weapon detection framework, you often need to import various libraries or modules to extend the functionality of your code. Below are some libraries and modules that you might want to import when working with Flask for detecting weapon

```
from tkinter import *
from tkinter import filedialog
from keras.models import Sequential
from keras.layers import Convolution2D
from keras.layers import MaxPooling2D
from keras.layers import Flatten
from keras.layers import Dense
from keras.models import model_from_json
```

6.2 CREATE THE BUTTON

Create an HTML template file (e.g., index.html) in the templates directory of your Flask project. In your HTML template, you can create a button using the <button> HTML element.

```
<!DOCTYPE html>
<html>
<head>
<title>Flask Button Example</title>
</head>
<body>
<h1>Click the Button</h1>
<button type="button">Click Me</button>
</body>
</html>
```

7. SYSTEM TESTING

7.1 SYSTEM TESTING

Software testing is the last phase of the software development cycle. Testing is very important for the success of a system. System testing makes a logical assumption that if all parts of the system are correct, then the goal has been achieved. The testing should be done at the end of all development steps. Even though the final testing and verification are inevitable for better life and functionality of the software. The major phases in testing are design of test plan, setting up test case and test candidate and test procedure, testing and correction. This is a cycle process and the software will circulate through all the steps till it attains the required quality.

7.2 TESTING TECHNIQUES

Unit Testing:

The first test in the development process is the unit test. The source code is normally divided into modules, which in turn are divided into smaller units called units. These units have specific behavior. The test done on these units of code is called unit test. Unit test depends upon the language on which the project is developed. Unit tests ensure that each unique path of the project performs accurately to the documented specifications and contains clearly defined inputs and expected results.

System Testing

System testing is defined as testing of a complete and fully integrated software product. This testing falls in black-box testing wherein knowledge of the inner design of the code is not a pre-requisite and is done by the testing team. It is the final test to verify that the product to be delivered meets the specifications mentioned in the requirement document. It should investigate both functional and non-functional requirements.

Validation Testing

The process of evaluating software during the development process or at the end of the development process to determine whether it satisfies specified business requirements. Validation Testing ensures that the product actually meets the client's needs. It can also be defined as to demonstrate that the product fulfils its intended use when deployed on appropriate environment.

Acceptance Testing

This is a type of testing done by users, customers, or other authorised entities to determine application/software needs and business processes. Acceptance testing is the most important phase of testing as this decides whether the client approves the application/software or not. It may involve functionality, usability, performance, and U.I of the application. It is also known as user acceptance testing (UAT), operational acceptance testing (OAT), and end-user testing.

7.3 SOFTWARE TESTING STRATEGIES

In order to make sure that the programme satisfies its intended criteria and performs as expected, software testing strategies are detailed blueprints that direct the testing procedure throughout a software development project. Various testing approaches are used, depending on the particulars of the project:

Waterfall Testing Strategy: The requirements, design, implementation, testing, and deployment phases of the waterfall model are sequentially approached, with each stage being finished before going on to the next. With a focus on validation against predetermined requirements, this strategy's testing is thorough and well-documented.

V-Model Testing Strategy: The V-Model, which is a development and testing phase relationship model, is an extension of the Waterfall model. To guarantee that requirements are verified early in the process, there is a testing phase that corresponds with each development phase.

Iterative Testing Strategy: This approach, which incorporates incremental testing with gradual additions of modules or components, is perfect for iterative development. Early flaw detection and adaptable modifications in response to feedback are made possible by it.

Exploratory Testing Strategy: Without preset test cases, testers use their experience and instincts to investigate the product, which is a helpful method for identifying bugs that scripted tests, could miss.

Automated Testing Strategy: This approach is especially helpful for regression testing, repetitive jobs and large-scale projects since it emphasises the use of automated testing tools and scripts to carry out test cases.

The nature of the project, the development process, the available funds, and the amount of time all play a role in selecting a testing approach. Frequently, a blend of these tactics might be utilised to guarantee thorough testing coverage and produce software that meets high standards.

8. SOURCE CODE

GUI.PY

```
from tkinter import *
import os
from tkinter import filedialog
import cv2
from tkinter import messagebox
def file_sucess():
global file_success_screen
file_success_screen = Toplevel(training_screen)
file_success_screen.title("File Upload Success")
file_success_screen.geometry("150x100")

Label(file_success_screen, text="File Upload Success").pack()
Button(file_success_screen, text="'ok'", font=(
'Palatino Linotype', 15), height="2", width="30").pack()
global ttype
def training():
global training_screen
global clicked

training_screen = Toplevel(main_screen)
training_screen.title("Training")
# login_screen.geometry("400x300")
training_screen.geometry("600x450+650+150")
training_screen.minsize(120, 1)
training_screen.maxsize(1604, 881)
training_screen.resizable(1, 1)
training_screen.configure()
# login_screen.title("New Toplevel")
```

```
Label(training_screen, text="'Upload Image '",
foreground="#000000", width="300", height="2", font=("Palatino Linotype", 16)).pack()
Label(training_screen, text="").pack()
options = [
'aphids', 'armyworm', 'beetle', 'bollworm', 'grasshopper', 'mites', 'mosquito', 'sawfly',
'stem_borer'
]
# datatype of menu text
clicked = StringVar()

# initial menu text
clicked.set("Normal")

# Create Dropdown menu
drop = OptionMenu(training_screen, clicked, *options)
drop.config(width="30")

drop.pack()

ttype = clicked.get()

Button(training_screen, text="'Upload Image'", font=(
'Palatino Linotype', 15), height="2", width="30", command=imgtraining).pack()
def imgtraining():
name1 = clicked.get()

print(name1)
import_file_path = filedialog.askopenfilename()
import os
s = import_file_path
os.path.split(s)
```

```
os.path.split(s)[1]
splname = os.path.split(s)[1]

image = cv2.imread(import_file_path)
# filename = 'Test.jpg'
filename = 'Data/' + name1 + '/' + splname

cv2.imwrite(filename, image)
print("After saving image:")
image = cv2.resize(image, (780, 540))

gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

cv2.imshow('Original image', image)
cv2.imshow('Gray image', gray)
# import_file_path = filedialog.askopenfilename()
print(import_file_path)
fnm = os.path.basename(import_file_path)
print(os.path.basename(import_file_path))
from PIL import Image, ImageOps

im = Image.open(import_file_path)

im_invert = ImageOps.invert(im)
im_invert.save('lena_invert.jpg', quality=95)
im = Image.open(import_file_path).convert('RGB')

im_invert = ImageOps.invert(im)
im_invert.save('tt.png')
image2 = cv2.imread('tt.png')
```

```
image2 = cv2.resize(image2, (780, 540))
cv2.imshow("Invert", image2)
img = image

gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
cv2.imshow('Original image', img)

dst = cv2.medianBlur(img, 7)
cv2.imshow("Nosie Removal", dst)
def fulltraining():
import model as mm

def testing():
global testing_screen
testing_screen = Toplevel(main_screen)
testing_screen.title("Testing")
# login_screen.geometry("400x300")
testing_screen.geometry("600x450+650+150")
testing_screen.minsize(120, 1)
testing_screen.maxsize(1604, 881)
testing_screen.resizable(1, 1)
testing_screen.configure()
# login_screen.title("New Toplevel")
Label(testing_screen, text='"Upload Image"', width="300", height="2", font=("Palatino
Linotype", 16)).pack()
Label(testing_screen, text="").pack()
Label(testing_screen, text="").pack()
Label(testing_screen, text="").pack()
Button(testing_screen, text='"Upload Image"', font=(
'Palatino Linotype', 15), height="2", width="30", command=imgtest).pack()
global affect
```

```
def imgtest():
import_file_path = filedialog.askopenfilename()

image = cv2.imread(import_file_path)
print(import_file_path)
filename = 'Output/Out/Test.jpg'
cv2.imwrite(filename, image)
print("After saving image:")
# result()
# import_file_path = filedialog.askopenfilename()
print(import_file_path)
fnm = os.path.basename(import_file_path)
print(os.path.basename(import_file_path))

# file_sucess()

print("\n**********\nImage : " + fnm + "\n**********")
img = cv2.imread(import_file_path)
if img is None:
print('no data')

img1 = cv2.imread(import_file_path)
print(img.shape)
img = cv2.resize(img, ((int)(img.shape[1] / 5), (int)(img.shape[0] / 5)))
original = img.copy()
neworiginal = img.copy()
img1 = cv2.resize(img1, (960, 540))
cv2.imshow('original', img1)
gray = cv2.cvtColor(img1, cv2.COLOR_BGR2GRAY)
img1S = cv2.resize(img1, (960, 540))
```

```
cv2.imshow('Original image', img1S)
grayS = cv2.resize(gray, (960, 540))
cv2.imshow('Gray image', grayS)
dst = cv2.fastNlMeansDenoisingColored(img1, None, 10, 10, 7, 21)
dst = cv2.resize(dst, (960, 540))
cv2.imshow("Nosie Removal", dst)
result()
def result():
import warnings
warnings.filterwarnings('ignore')
import tensorflow as tf
classifierLoad = tf.keras.models.load_model('model.h5')
import numpy as np
from keras.preprocessing import image
test_image = image.load_img('./Output/Out/Test.jpg', target_size=(200, 200))
# test_image = image.img_to_array(test_image)
test_image = np.expand_dims(test_image, axis=0)
result = classifierLoad.predict(test_image)

out = ''
Remedy = ''
if result[0][0] == 1:
out = "aphids"
Remedy = 'A few tablespoons of liquid dish or insecticidal soap diluted in a pint of water'

elifresult[0][1] == 1:
out = "armyworm"
Remedy = 'If infestations are large, you can use insecticides containing active ingredients such as spinosad, bifenthrin, cyfluthrin, and cypermethrin'
```

elifresult[0][2] == 1:

out = "beetle"

Remedy = 'The best chemical treatment for Armyworms is Bifen LP and Reclaim IT'

elifresult[0][3] == 1:

out = "bollworm"

Remedy = 'Spraying any one of the following insecticides: Phosalone 35%EC 2000 ml/ha'

elifresult[0][4] == 1:

out = "grasshopper"

Remedy = ' you can use a PUMP SPRAYER to spray. Add .25 oz of Maxxthor per gallon of water and use this mixture to cover up to 1,000 sq/ft'

elifresult[0][5] == 1:

out = "mites"

Remedy = 'The cold presses neem oil spray is more effective against chemical resistant bed bugs and dust mites'

elifresult[0][6] == 1:

out = "mosquito"

Remedy = 'Larvicides are chemicals designed to be applied directly to water to control mosquito larvae'

elifresult[0][7] == 1:

out = "pest"

Remedy = 'HIT Crawling Insect Killer – Cockroach Killer Spray (400ml) | Instant Kill | Deep-Reach Nozzle | Fresh Fragrance.'

elifresult[0][8] == 1:

out = "sawfly"

Remedy = 'emamectin benzoate proved as the best with maximum reduction in sawfly larval population followed by indoxacarb, spinosad, fipronil, cartap hydrochloride, lambda cyhalothrin, Carbosulfan 25 EC and Quinalphos as the mean larval population was found to be 1.83, 3.00, 5.33, 4.50, 7.00, 8.17, 8.33 and 12.83, after 15 days'

elifresult[0][9] == 1:

out = "stem_borer"

```
Remedy = ' The castor seedlings attract female moths of Spodoptera for egg laying. Leaves having egg masses and tiny caterpillars are clipped and destroyed'

messagebox.showinfo("Result", "Classification Result : " + str(out))

messagebox.showinfo("Remedy", 'Remedy' + Remedy)

def main_account_screen():

global main_screen

main_screen = Tk()

width = 600

height = 500

screen_width = main_screen.winfo_screenwidth()

screen_height = main_screen.winfo_screenheight()

x = (screen_width / 2) - (width / 2)

y = (screen_height / 2) - (height / 2)

main_screen.geometry("%dx%d+%d+%d" % (width, height, x, y))

main_screen.resizable(0, 0)

# main_screen.geometry("300x250")

main_screen.configure()

main_screen.title("Pest classification ")


Label(text="Pest Prediction ", width="300", height="5", font=("Palatino Linotype", 16)).pack()


Button(text="UploadImage", font=(

'Palatino Linotype', 15), height="2", width="20", command=training, highlightcolor="black").pack(side=TOP)

Label(text="").pack()

Button(text="Training", font=(

'Palatino Linotype', 15), height="2", width="20", command=fulltraining, highlightcolor="black").pack(side=TOP)


Label(text="").pack()

Button(text="Testing", font=(
```

'Palatino Linotype', 15), height="2", width="20", command=testing).pack(side=TOP)

Label(text="").pack()

main_screen.mainloop()

main_account_screen()

MODEL.PY

```
from keras.models import Sequential
from keras.layers import Convolution2D
from keras.layers import MaxPooling2D
from keras.layers import Flatten
from keras.layers import Dense
from keras.models import model_from_json
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings('ignore')
batch_size = 32
from tensorflow.keras.preprocessing.image import ImageDataGenerator
# All images will be rescaled by 1./255
train_datagen = ImageDataGenerator(rescale=1/255)
# Flow training images in batches of 128 using train_datagen generator
train_generator = train_datagen.flow_from_directory(
'Data',  # This is the source directory for training images
target_size=(200, 200),  # All images will be resized to 200 x 200
batch_size=batch_size,
# Specify the classes explicitly
classes =
['aphids','armyworm','beetle','bollworm','grasshopper','mites','mosquito','sawfly','stem_borer'],
# Since we use categorical_crossentropy loss, we need categorical labels
class_mode='categorical')
import tensorflow as tf
model = tf.keras.models.Sequential([
# Note the input shape is the desired size of the image 200x 200 with 3 bytes color
```

```
# The first convolution
tf.keras.layers.Conv2D(16, (3,3), activation='relu', input_shape=(200, 200, 3)),
tf.keras.layers.MaxPooling2D(2, 2),
# The second convolution
tf.keras.layers.Conv2D(32, (3,3), activation='relu'),
tf.keras.layers.MaxPooling2D(2,2),
# The third convolution
tf.keras.layers.Conv2D(64, (3,3), activation='relu'),
tf.keras.layers.MaxPooling2D(2,2),
# The fourth convolution
tf.keras.layers.Conv2D(64, (3,3), activation='relu'),
tf.keras.layers.MaxPooling2D(2,2),
# The fifth convolution
tf.keras.layers.Conv2D(64, (3,3), activation='relu'),
tf.keras.layers.MaxPooling2D(2,2),
# Flatten the results to feed into a dense layer
tf.keras.layers.Flatten(),
# 128 neuron in the fully-connected layer
tf.keras.layers.Dense(128, activation='relu'),
# 5 output neurons for 5 classes with the softmax activation
tf.keras.layers.Dense(9, activation='softmax')
])
model.summary()
from tensorflow.keras.optimizers import RMSprop
early = tf.keras.callbacks.EarlyStopping(monitor='val_loss',patience=5)
model.compile(loss='categorical_crossentropy',
optimizer=RMSprop(lr=0.001),
metrics=['accuracy'])
total_sample=train_generator.n
n_epochs = 45
```

```
history = model.fit_generator(
train_generator,
steps_per_epoch=int(total_sample/batch_size),
epochs=n_epochs,
verbose=1)
model.save('model.h5')
acc = history.history['accuracy']
loss = history.history['loss']
epochs = range(1, len(acc) + 1)
# Train and validation accuracy
plt.plot(epochs, acc, 'b', label='Training accurarcy')
plt.title('Training  accurarcy')
plt.legend()
plt.figure()
# Train and validation loss
plt.plot(epochs, loss, 'b', label='Training loss')
plt.title('Training  loss')
plt.legend()
plt.show()
```

9. CONCLUSION & FUTURE ENHANCEMENT

In conclusion, the use of CNN algorithm for pest recognition has shown promising results in accurately identifying and classifying pests from images. The CNN model is able to learn and extract relevant features from images without the need for manual feature engineering, making it a powerful tool for image classification tasks. Pre-processing techniques such as median filtering can be used to improve the quality of the images before passing them through the CNN model. During the training process, monitoring the training loss and training accuracy is important to ensure that the model is learning effectively from the training data and not over fitting. While CNN algorithm has shown to be effective for pest recognition, it is important to note that the accuracy of the model depends on the quality and diversity of the training data. Collecting and labeling a large and diverse dataset of pest images is crucial for training a robust and accurate pest recognition system. Overall, the use of CNN algorithm for pest recognition has the potential to revolutionize the way pests are identified and managed in agriculture, leading to more efficient and sustainable pest control practices.

While CNN algorithm has shown promising results in pest recognition, scaling up the system to handle larger and more complex datasets could improve the accuracy and effectiveness of the system. This could involve exploring different architectures and optimization techniques to improve the model's performance. In addition to images, other data sources such as audio recordings or sensor data could be used to improve the accuracy of the pest recognition system. For example, audio recordings could be used to identify pests based on their distinct sounds, while sensor data could be used to detect environmental factors that are conducive to pest infestations.

10.REFERENCE

[1] Albanese, Andrea, Matteo Nardello, and Davide Brunelli. "Automated pest detection with DNN on the edge for precision agriculture." IEEE Journal on Emerging and Selected Topics in Circuits and Systems 11.3 (2021): 458-467.

[2] Ullah, Naeem, et al. "An Efficient Approach for Crops Pests Recognition and Classification Based on Novel DeepPestNet Deep Learning Model." IEEE Access 10 (2022): 73019-73032.

[3] Li, Lili, Shujuan Zhang, and Bin Wang. "Plant disease detection and classification by deep learning—a review." IEEE Access 9 (2021): 56683-56698.

[4] Chen, Jing, Qi Liu, and Lingwang Gao. "Deep convolutional neural networks for tea tree pest recognition and diagnosis." Symmetry 13.11 (2021): 2140.

[5] MUPPALA, Chiranjeevi, and Velmathi GURUVIAH. "Paddy Pest Identification with Deep Convolutional Neural Networks." Engineering in Agriculture, Environment and Food 14.2 (2021): 54-60.

[6] Xin, Mingyuan, and Yong Wang. "Image recognition of crop diseases and insect pests based on deep learning." Wireless Communications and Mobile Computing 2021 (2021).

[7] Gupta, Yash Munnalal, and SOMJIT HOMCHAN. "Insect detection using a machine learning model." Nusantara Bioscience 13.1 (2021).

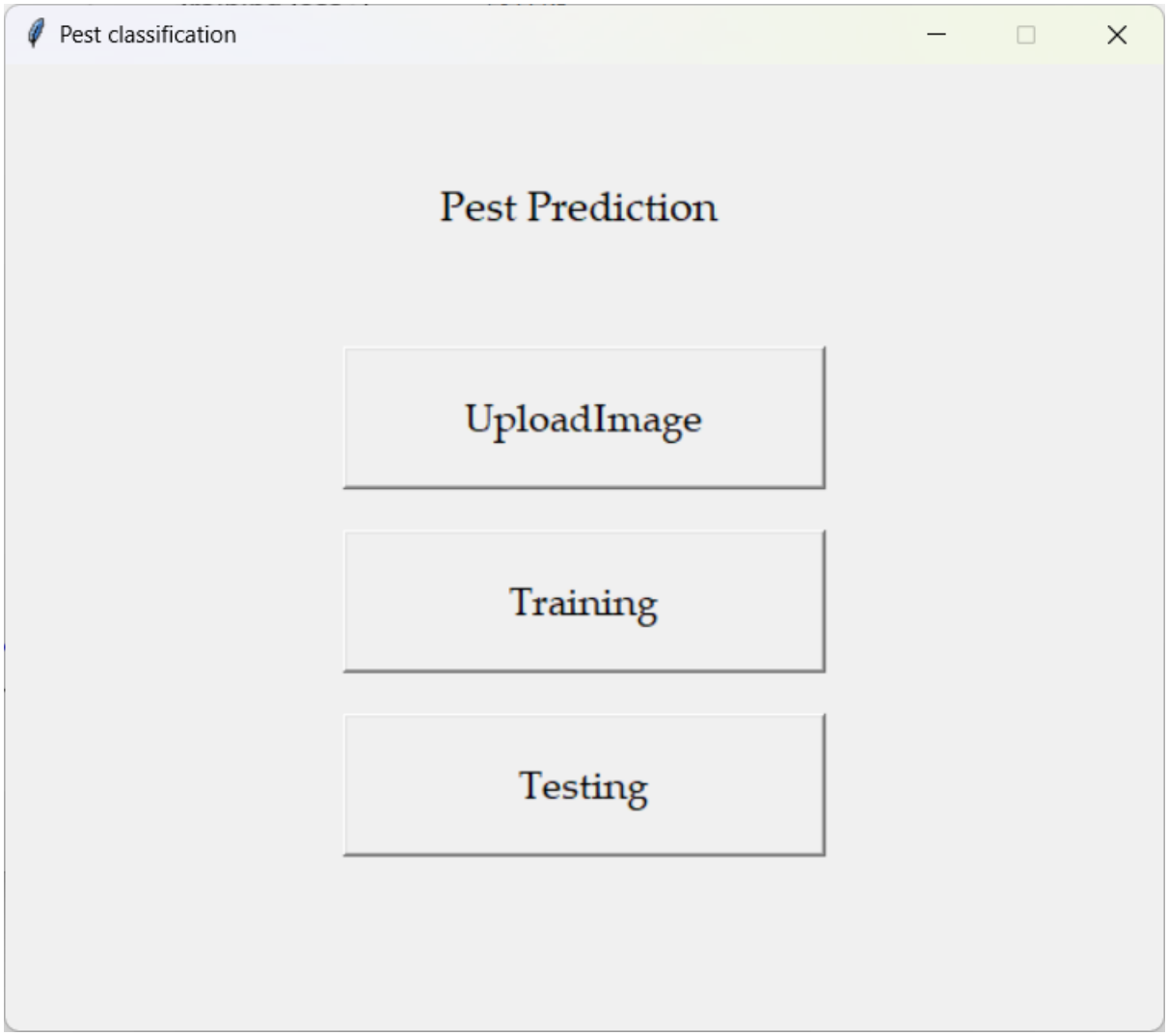
[8] Yang, Jiachen, et al. "Efficient data-driven crop pest identification based on edge distance-entropy for sustainable agriculture." Sustainability 14.13 (2022): 7825.

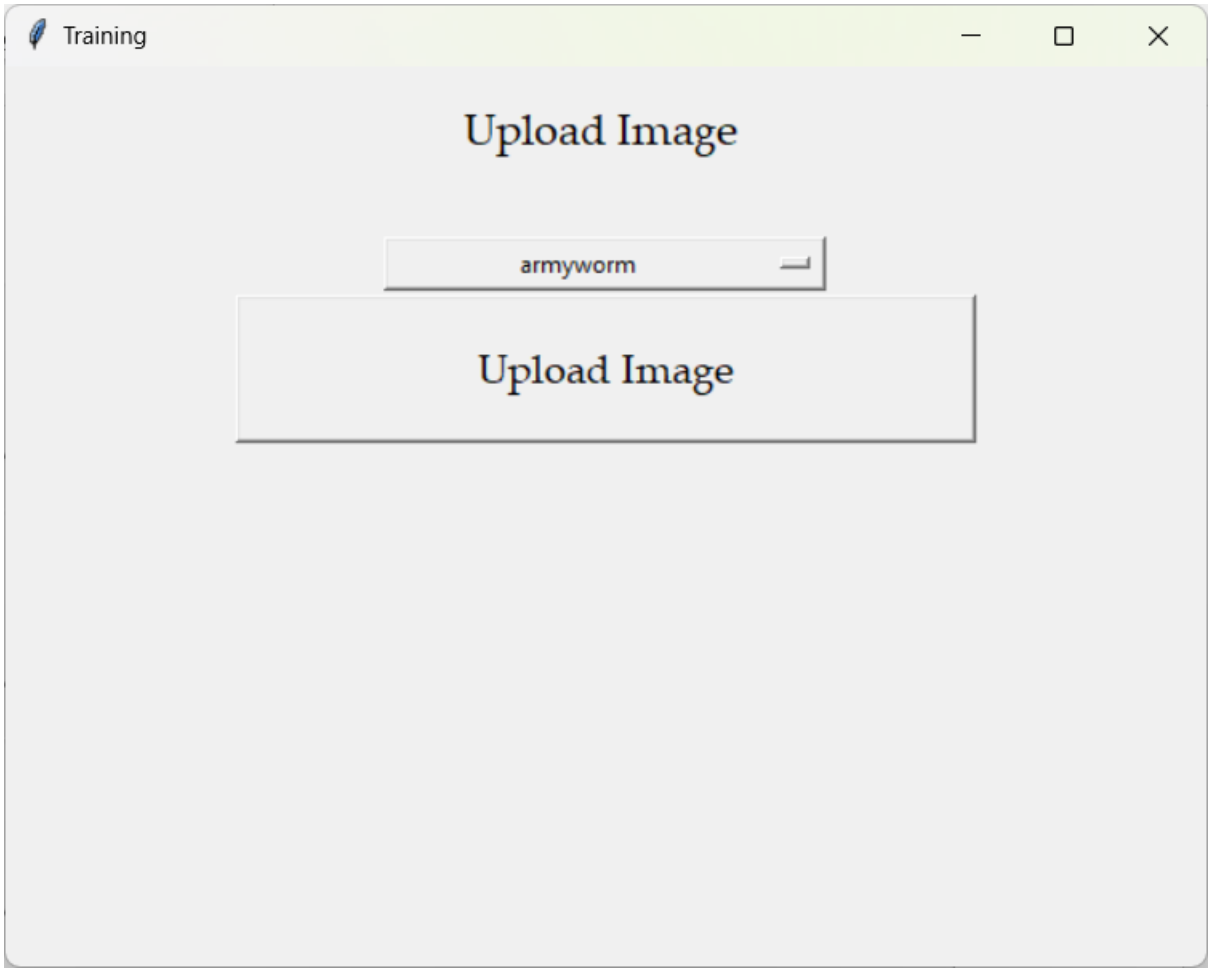
[9] Pattnaik, Gayatri, and Kodimala Parvathy. "Machine learning-based approaches for tomato pest classification." TELKOMNIKA (Telecommunication Computing Electronics and Control) 20.2 (2022): 321-328.

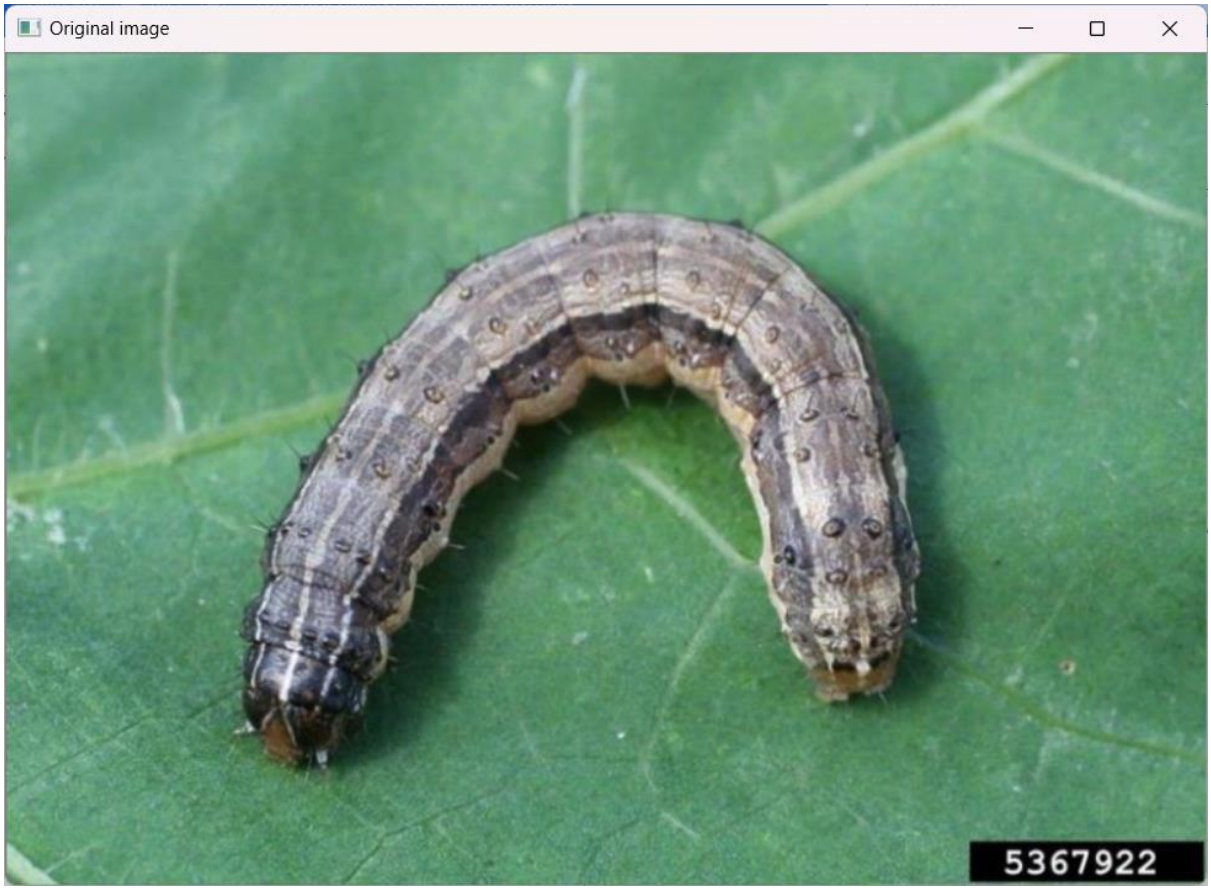
[10] Patel, Pruthvi P., and Dineshkumar B. Vaghela. "Crop diseases and pests detection using convolutional neural network." 2019 IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT). IEEE, 2019.

APPENDIX

SCREENSHOT

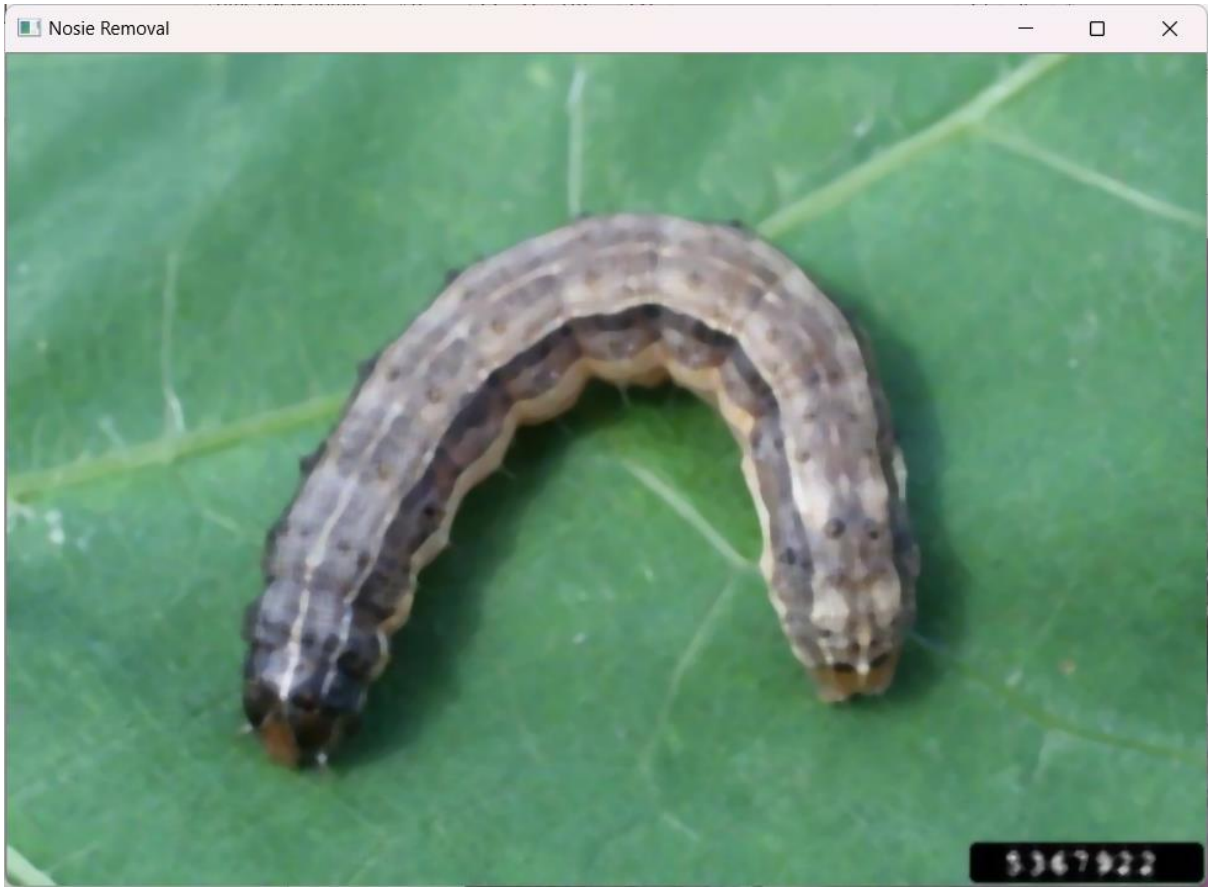



Training
Upload Image
armyworm
Upload Image

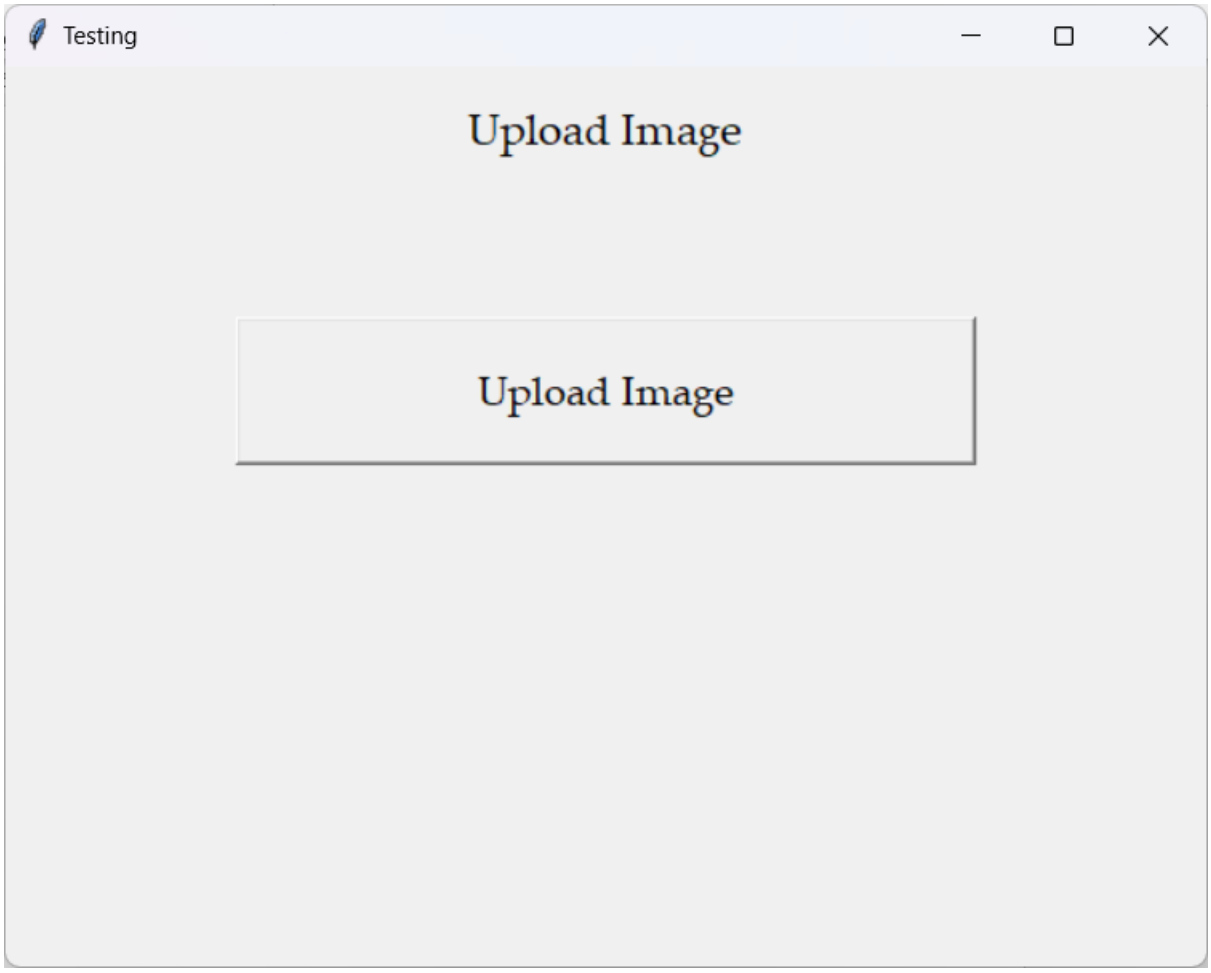

Original image
5367922

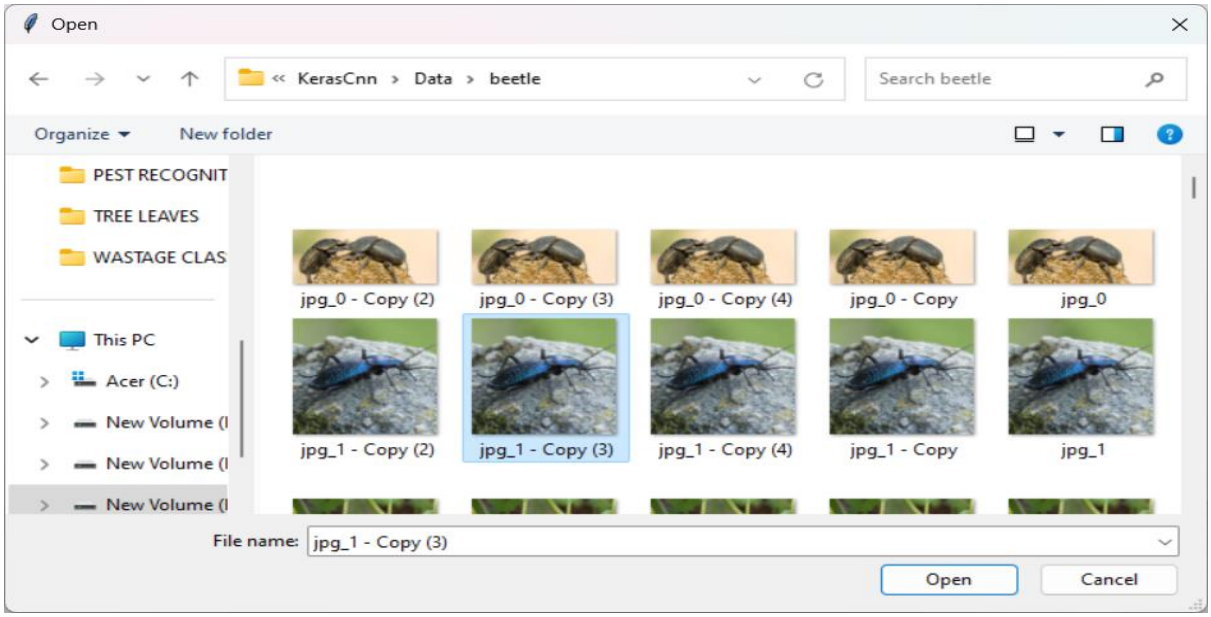

Gray image
5367922


Invert
5367922


Nosie Removal
5367922

TESTING PHASE

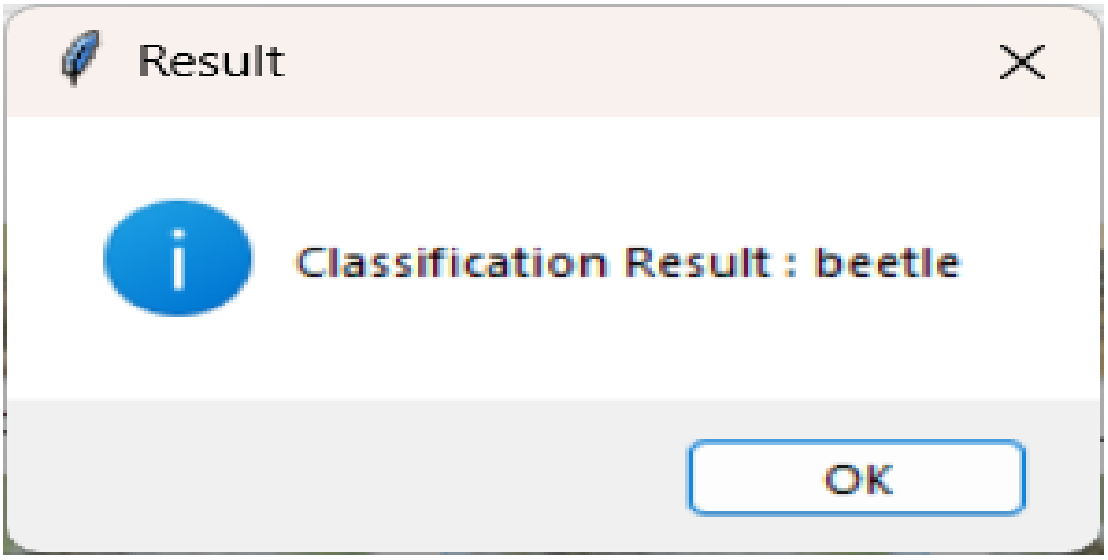





original


Gray image


Nosie Removal


Result
Classification Result : beetle
OK