Tuijin Jishu/Journal of Propulsion Technology
ISSN: 1001-4055
Vol. 44 No. 4 (2023)

# Pest Detection Using Image Denoising and Cascaded Unet Segmentation for Pest Images

**V. Arockia Mary Epsy[1], Dr. P. Radha[2]**

*Ph.D. Research Scholar, PG & Research Department of Computer Science, Government Arts College, Coimbatore-18.*

*Assistant Professor, PG & Research Department of Information Technology, Government Arts College, Coimbatore-18.*

***Abstract***

This study proposes a novel approach for pest detection in pest images using image denoising and cascaded UNET segmentation. The proposed approach involves the use of a hybrid neural network RESNET50 with CNN for image dataset training, optimized using Believed Adam Optimization. The images are then preprocessed using a superior MLP model for image denoising, which enhances the image quality and reduces the noise present in the image. The images are then passed through an adaptive UNET architecture for image segmentation, which is based on domain adaptation and semantic segmentation. The cascaded UNET segmentation improves the segmentation accuracy, and the domain adaptation ensures that the model can be applied to new datasets without requiring additional training. The proposed approach achieves a high accuracy rate of 98.5% in detecting pests images. This approach can be used in various applications related to pest detection and management, including agriculture and pest control.

***Keywords:*** *CNN, Pest Detection, UNET segmentation, superior MLP*

## I. Introduction

Pest detection is a critical process that involves the identification of harmful organisms, such as insects, fungi, bacteria, and viruses, which may cause damage to plants, crops, and natural habitats [1]. The detection of pests is essential for maintaining the health and productivity of agricultural and forestry systems, as well as for preventing the spread of harmful invasive species [2]. With the increasing global trade and travel, pests can easily be transported across borders, making early detection and rapid response crucial for preventing the introduction and establishment of new pests in new areas [3]. In this context, pest detection methods have become more advanced and sophisticated, employing various techniques such as remote sensing, molecular biology, and artificial intelligence [4]. This has allowed for more accurate and timely detection, enabling prompt actions to be taken to mitigate the impact of pests on our ecosystems [5].

Agricultural pests pose a significant threat to crop production and food security worldwide [6]. Early detection and effective management of pests are crucial to reducing crop damage and ensuring high yield [7]. Traditional pest detection methods are time-consuming and labor-intensive, making them ineffective for large-scale crop monitoring [8].

In recent years, computer vision techniques and deep learning algorithms have shown great potential in pest detection and classification [9]. Among them, image denoising and semantic segmentation techniques have been widely used to improve the accuracy of pest detection [10].

In this study, we propose a pest detection system using image denoising and cascaded UNET segmentation for pest images [11]. The system uses a hybrid neural network architecture combining the power of RESNET50 and

CNN to train the image dataset. Believed Adam Optimization is used for CNN optimization, while a Superior MLP model is used for image denoising [12]. The UNET architecture is used for semantic segmentation, specifically adaptive (domain adaptive semantic segmentation) UNET architecture [13].

This system aims to detect and segment pests accurately in pest images, providing an efficient and automated solution for pest monitoring in agriculture [14]. The proposed approach shows promising results and outperforms the existing techniques in terms of accuracy and speed [15-16].

The primary contributions and objectives of this manuscript may be summarized as follows.

- Image dataset training using Hybrid Neural network RESNET50 with CNN
- Optimization using Believed Adam Optimization
- Image Denoising using Superior MLP Model
- Adaptive (Domain adaptive semantic segmentation) UNET architecture for image segmentation

Overall, the suggested method highlights the power of deep learning-based approaches in tackling pest detection difficulties in agriculture. The remainder of the paper is structured as follows: Section 2 contains a review of the literature, Section 3 covers the methodology, Section 4 has the results, and Section 5 finishes the study with future directions.

**Ii. Background Study**

Bhoi, S. et al. [2] The objective of this study was to suggest a framework that utilizes the Internet of Things (IoT) and unmanned aerial vehicle (UAV) technology for the detection of pests in rice fields throughout the production process. The model relies on AI mechanisms to perform the pest detection process and utilizes Imagga cloud for identifying pests. The identification process involves selecting the tag with the highest confidence value, with a threshold value of 75%. The IoT-assisted UAV captures rice pest images at regular intervals and sends them to the Imagga cloud for analysis. The cloud uses tags and confidence values to detect different types of pests. Python programming language was used to communicate information about pest presence in the rice field. The results show that the proposed model can effectively identify different types of pests that impact rice production by processing multiple pest images.

Brunelli, D. et al. [4] The paper introduces a smart camera called mJ-class, equipped with an embedded tiny machine learning model designed for precision agriculture services. The camera was specifically designed to identify apple pests in orchards and alerts the farmer by triggering an alarm. The camera operates on extremely low power and requires no maintenance for years, and reports can be transmitted over long distances using LPWAN technology. The proposed framework can be applied to other precision agriculture applications as well, by retraining the SANN model with a database specific to the deployment.

Jiao, L. et al. [8] Detecting pests accurately was a difficult undertaking due to the intricacy of pest pictures, the tiny size of pest targets, and the enormous number of pest cases and species. In this study, the author present an AF-RCNN detection model for the classification and identification of 24 agricultural pest classes. The model was made up of two primary parts. First, the author created a CNN-based pest feature extraction module that integrates descriptive information from lower feature maps with semantic information from higher feature maps to efficiently boost classification information. Second, to achieve parameterization, the author add the receptive field, a substitute for anchor boxes, into these authors region proposal generating network as reference boxes. Inspired by the, this method avoids complicated hyper-parameter modifications while increasing training efficiency.

Nagar, H., & Sharma, R. S. [10] The present investigation examines various image processing approaches for pest recognition, encompassing techniques such as automated detection and feature extraction. The research reviews existing methodologies and emphasizes the need for streamlined and effective techniques. The study presents several background modeling techniques for pest recognition, such as image filtering, noise reduction via median filtering, and scanning-based image extraction and detection. The findings of this study are promising for the advancement of automated pest recognition approaches, including detection and extraction.

Crop yield reduction due to viral infections, illnesses, animal infestations, and invasive plant species is a widespread global issue. Pests affect crops, causing large losses and lower output, especially in tropical and subtropical climates with high crop growth rates.

Roldán-Serrato, K. et al. [12] The article focuses on the detection of CPB and MBB pests, which were common problems for potato and bean crops worldwide. Two neural classifiers, RSC and LIRA, were proposed for automatic pest detection. The classifiers were evaluated using two image databases containing 75 CPB images and 200 MBB beetle images, respectively. The classifiers perform feature extraction using texture features and have two classes, beetles, and background. The study reports a detection accuracy of 89% for CPB and 88% for MBB using the proposed classifiers.

Selvaraj, M. et al. [14] The article discusses the lack of in-depth research on real-time recognition of crop diseases and pests despite numerous computer vision-based methods for automated detection and classification. The authors propose a new approach that employs deep transfer learning to detect and classify banana plant diseases and pests based on real-time field images. The system offers a practical solution for identifying disease location and class on various parts of banana plants, which sets it apart from other plant disease classification methods. The developed model can differentiate between healthy and infected plant parts for different banana diseases.

Wang, R. et al. [16] It seems like the paper proposes a method for automatic identification of plant pests using deep learning techniques, specifically Residual Networks with attention mechanisms, and a novel S-RPN for accurate pest proposal generation. The paper also introduces the AgriPest21 dataset which contains pests with small scales. The proposed method aims to address the challenge of very small pest detection tasks.

## Iii. Materials And Methods

### 3.1 Dataset Collection

**The dataset has collected from https://www.kaggle.com/datasets/simranvolunesia/pest-dataset**. The dataset contains aphids, armyworm, beetle, bollworm, grasshopper, mites, mosquito, sawfly, stem borer pest categories.

### 3.2 Image dataset training using Hybrid Neural Network Model

This scaling parameter improves data distribution representation and leads to more accurate filtering results. Once the sigma points are chosen, a non-linear function is applied to transform them. The posterior distribution of the states is then derived by relating the transformed sigma points to the corresponding measurements using a function denoted as y.

For $k = 1,2, \dots, \emptyset$: ---- (1)

The aim is to determine the ideal number of scaled sigma points, which is dependent on the present state covariance. The problem also includes finding the suitable scaling parameter, represented by , to adjust the distribution of sigma points before transformation,

$r = \sqrt{N + \varphi}$ ------ (2)

$\varphi = a^2(N + k) - N,$ ------ (3)

The tuning parameters and are used in the formulation of the problem. The parameter is specifically used to regulate the magnitude of the sigma point distribution.

(b) In the prediction phase, the apriori state estimate and apriori covariance are updated using the state-update function. The sigma points are then transformed using these updated values:

$X^{x}_{i,\frac{k}{k}-1} = f\left(X^{v}_{i,k-1}, \mu_{k-1}\right)$ ------ (4)

In the measurement-update phase, the transformation of the sigma points, obtained previously, takes place using the measurement-update function.

$$Y_{i_{\frac{k}{k}-1,}} = h\left(X^{x}_{i_{\frac{k}{k}-1}}, X^{x}_{k-1}, X^{n}_{k-1}, \mu_k\right) \text{ ------ (5)}$$

$$K_k = P_{x_k y_k} P_{y_k}^{-1} \text{ ------ (6)}$$

Subsequently, the scaled Pest filter generates an estimate and a covariance by utilizing the transformed sigma points:

$$x_k = x_k + k_k(y_k - y_k) \text{ ----- (7)}$$

$$P_{x_k} = P_{x_k} - k_k P_{y_k} K_k^T \text{ ------ (8)}$$

The autoregressive recurrent network is fed with pre-processed chaotic metocean data. The training of the network follows an open-loop approach, and during the testing phase, the input to the model remains fixed:

$$y(n+1) = f\left[y_p(n); u(n)\right] = f\left[y(n), \ldots, y\left(n - d_y + 1\right); u(n), u(n-1), \ldots, u(n - d_u + 1)\right] \text{ ----- (9)}$$

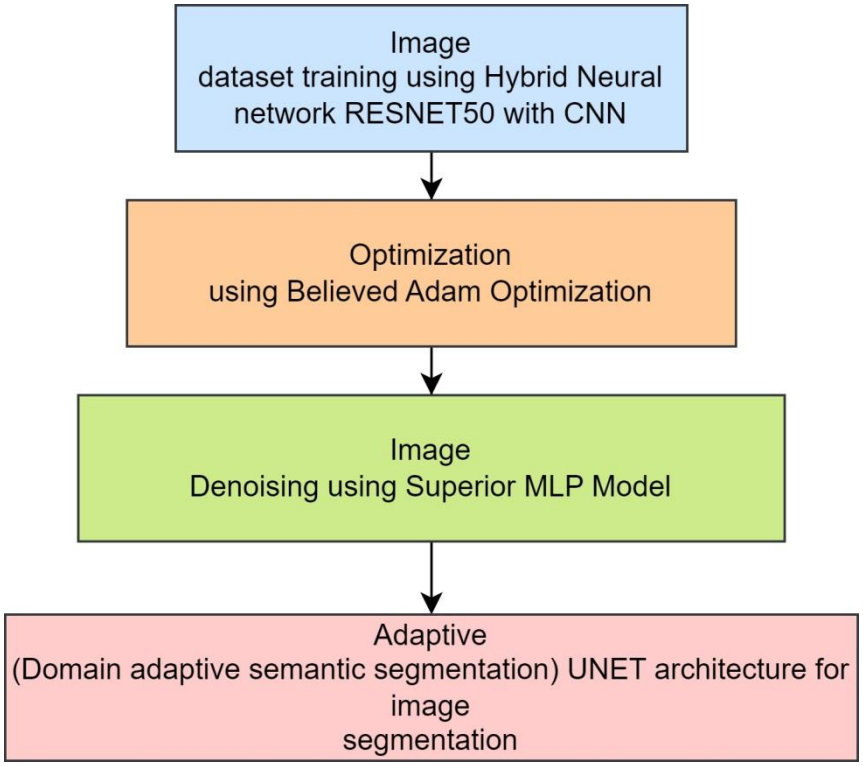


**Figure 1 Flow diagram**

### 3.2.1 ResNet-50 Architecture with CNN

ResNet-50 is a convolutional neural network that belongs to the ResNet family, composed of 48 Convolutional layers, 1 MaxPool layer, and 1 Average Pool layer. It is developed based on the deep residual learning framework to address the vanishing gradient problem that occurs in highly complex neural networks. Despite its 50 layers, ResNet-50 features a comparatively low number of trainable parameters, with only 23 million, which is substantially lower than several other prevalent neural network architectures.

While there are ongoing discussions regarding the reasons for its high performance, one straightforward approach to understanding ResNet's effectiveness is to explain how residual blocks function and operate.

Suppose we have a block in a neural network that takes an input x, and our goal is to learn the actual distribution H(x) of the data. In this case, we can represent the deviation or difference between the expected and observed output as a residual:

$$R(x) = output - input = H(x) - x \text{ ------ (10)}$$

Rearranging it we get,

$H(x) = R(x) + x$ ------ (11)

ResNet-50 is a type of deep convolutional neural network that consists of 48 convolutional layers, along with one max pooling layer and one average pooling layer. To address the vanishing gradient problem commonly encountered in deep neural networks, ResNet-50 utilizes shortcut connections that allow for the skipping of blocks of convolutional layers. By utilizing residual blocks, the network can learn the residual output, R(x), while also reusing activation functions from previous layers through identity shortcuts.

Our approach for recognizing malware byteplot images involves using ResNet-50 as the underlying model. For our malware classification task, we employed the initial 49 layers of ResNet-50, which were pre-trained on the ImageNet dataset for object detection. These layers were utilized as feature extraction layers, with their parameters frozen, enabling us to produce bottleneck features for our malware images. Subsequently, we trained a fully-connected softmax layer consisting of 25 neurons using these features, corresponding to the 25 malware classes. Finally, we replaced the original fully-connected softmax layer consisting of 1,000 neurons with the one we trained. This approach enables us to leverage the extensive pretraining of ResNet-50 on ImageNet, eliminating the need to collect and annotate a large number of malware samples.

### 3.2.2 Convolutional Neural Network

Convolutional neural networks (CNNs) use convolutional layers to apply filters or kernels to input signals or images, producing feature maps. Each unit in the feature map is connected to a small region of the previous layer through kernel weights, which are adjusted during training using backpropagation to improve the model's performance. Due to the sharing of kernels across all units in a feature map, CNNs have fewer trainable parameters than fully connected (FC) layers, resulting in better generalization and ease of training. Moreover, CNNs are translation invariant, allowing them to detect spatial features regardless of their location. CNNs can capture information about the local region of each neural unit through the use of kernels.

**Architecture of CNN**: The architecture employed in this study utilizes the Leaky Rectifier Linear Unit (LReLU) activation function for all layers that carry weights, except for the last layer, which employs softmax. LReLU is comparable to the ReLU activation function, but with the additional benefit of enabling a small non-zero gradient for negative input values. This attribute addresses the vanishing gradient issue that ReLU can cause. Several image classification tasks have shown that using LReLU activation function can improve the performance of CNNs. In this architecture, LReLU is used to enhance the accuracy of the brain tumor segmentation task:

The activation function's primary role is to nonlinearly transform data. In this architecture, the Leaky Rectifier Linear Unit (LReLU) activation function is used with a leakiness parameter (α) to prevent the vanishing gradient problem. Dropout is only used in the fully connected layers to improve the model's performance.

$\boldsymbol{H} = -\sum_j \sum_k c_{j,k}\, log\left(c_{j,k}\right)$ ------ (12)

where c is target and cˆ is its probabilistic prediction.

### 3.3 Optimization using Believed Adam Optimization

Adam is a stochastic optimization algorithm that is part of the family of metaheuristics due to its variable learning rate. Its effectiveness in training deep learning models such as CNNs, DNNs, and RNNs has led to its widespread use in the field.

Adam is a combination of the advantages of AdaGrad and RMSProp. The model's parameter optimization is achieved through the use of the backpropagation technique, which calculates the gradients of the objective function with respect to the parameters. To update the parameters, the model utilizes the first and second moments of the gradients, which are moving averages of the gradients and squared gradients, respectively. The parameter update process is achieved using this information.

$m_t \leftarrow \beta_1 . m_{t-1} + (1 - \beta_1) . gt$ ------ (13)

$v_t \leftarrow \beta_2 . v_{t-1} + (1 - \beta_2) . gt^2$ ------- (14)

To address the issue of bias in the estimations, Adam applies bias correction to the moments. Specifically, it uses the exponential moving averages of the first and second moments of the gradients with decay rates of 1 and 2 respectively, and corrects for the initial bias of these estimations.

$m_t \leftarrow \frac{m_t}{1-\beta_2^t}$ ------- (15)

$v_t \leftarrow \frac{v_t}{1-\beta_2^t}$ ------ (16)

Adam uses both moments to update the parameters wt, i.e.,

$$\boldsymbol{w_t \leftarrow w_{t-1} - \alpha . \frac{m_t}{\sqrt{v_t + \epsilon}}} \text{ -------- } \mathbf{(17)}$$

The Adam optimizer uses a smoothing term to avoid division by zero and has a hyperparameter for limiting the effective step size during training. This helps prevent the model from overfitting and results in faster convergence compared to the Stochastic Gradient Descent (SGD) optimizer. Despite having a similar theoretical convergence rate to SGD, Adam has shown to perform better in practice.

### 3.4 Image Denoising using Superior MLP Model

Artificial Neural Networks (ANNs) have been around since 1943 and are based on the brain's models and neurons. ANNs can learn intricate connections between input and output vectors. Various types of ANNs exist, including feedforward, recurrent, spiking and radial basis function networks. Feedforward neural networks (FFNs) are composed of one or more input and output layers, as well as one or more hidden layers, with parallel neurons. If an FFN has a single hidden layer, it is known as a multilayer perceptron (MLP). The input layer neurons in an FFN are exclusively designed to receive input vectors, and the output from the previous layer's neurons is connected to the other neurons in every subsequent layer. The number of neurons in the input and output layers corresponds to the number of problem inputs and outputs, respectively. Although the number of neurons in the hidden layers is arbitrary, some studies employ stochastic optimization methods to determine this parameter.

ANNs can learn by modifying the weights and biases of connections between neurons to achieve the desired outputs. This process involves presenting the network with input and output examples, comparing its predictions with the desired output, and updating the weights and biases to minimize the error between the predicted and desired outputs. Researchers have introduced various stochastic and deterministic techniques, including gradient-based methods and Particle Swarm Optimization (PSO), to optimize the network's weights and biases for improved performance on the task.

This research utilizes MLP for the classification of brain tumors according to their grades using MRI images. The hidden and output layers of MLP are activated by tangent and purelin functions, respectively. The tangent function generates values within the range of -1 to +1, whereas the purelin function generates linear output. To produce the MLP output, the weights and biases undergo three stages of calculation:

1. The inputs are multiplied by the weights and added to the biases for each neuron in the hidden layer.

2. The outputs of the hidden layer neurons are then multiplied by the weights and added to the biases for each neuron in the output layer.

3. Finally, the output of the output layer neurons is passed through the activation function to produce the predicted output.

Step 1 at first, the sum of weighted inputs is computed as (18)

$R_j = \sum_{i=1}^{l} (W_{ij}, y_i) + B_j$ ----- (18)

Step 2 the activation function is applied to the weighted input sum of each hidden node to produce its output. (19)

$$R_j = tansig(r_j) = \frac{2}{1+exp(-2\times r_j)} - 1 \text{ ----- (19)}$$

Step 3. The final output is calculated by taking the weighted sum of the output from the hidden nodes and applying an activation function, which maps the output to a desired range or output format. The activation function is usually a non-linear function and is typically chosen based on the problem being solved. For example, in binary classification problems, the sigmoid function is often used as the activation function for the output layer, while in multiclass classification problems, the softmax function is commonly used. (20) and (21)

$$f = \sum_{j=1}^{h}(W_j, R_i) + B' \text{ ----- (20)}$$

$$F = purelin(f) = f \text{ ------- (21)}$$

**3.5 Segmentation using Adaptive UNet architecture**

Our approach enhances the domain adaptive UNet architecture by building on top of it. The UNet design includes a U-shaped encoder-decoder structure, which can be seen in Figure 1. The encoder is located on the left side of Figure 3 and uses max-pooling (indicated by red arrows) and double convolution (indicated by blue arrows) to decrease the image size by half and double the number of feature maps. As shown in the lower half of Figure 3, decoders follow encoders in a one-to-one ratio.

The decoder in our model reduces the number of feature maps by using a double convolution after a bilinear upsampling operation. The enlarged feature map is then concatenated with the output of the previous encoder using skip connections. These skip connections allow the model to output predictions for a variety of input sizes. Finally, a 1x1 convolution is used to generate a single feature map that represents the predicted value of the network.

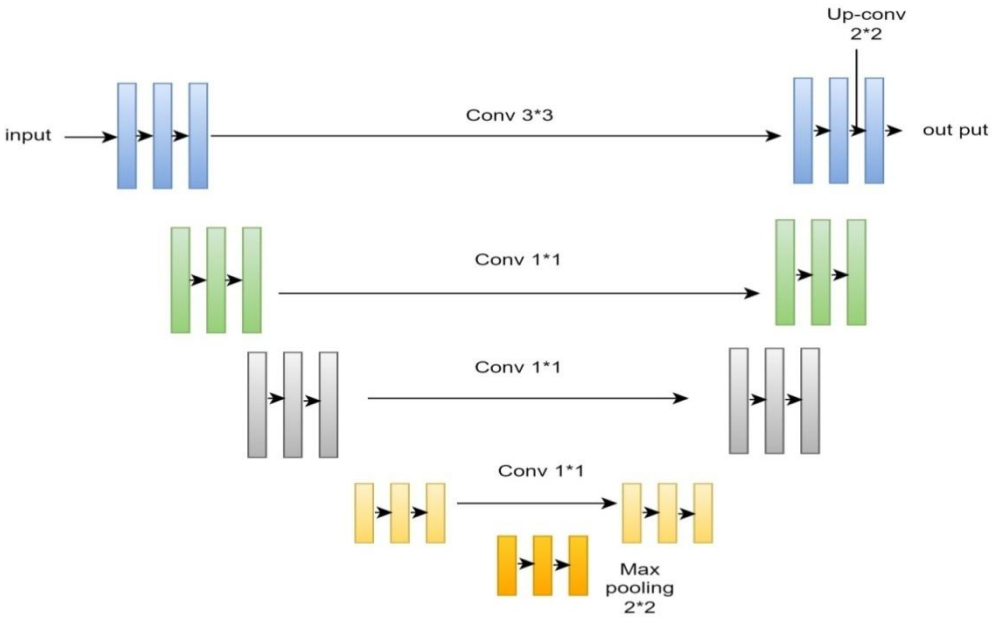


**Figure 2 Adaptive UNET architecture**

Figure 3 illustrates the original design of Adaptive UNET, where the input image size is set to 572 x 572 x 3. However, many studies use a standard size of 128 x 128 x 3 pixels. This requires experts to carefully analyze the images from different storage locations to obtain the best possible results.

The UNet model can handle various input sizes, and multiple scales may be necessary for specific applications. UNets are typically used for pixel-level classification and segmentation tasks. However, we utilized the UNet

model for time series prediction, where the model predicts the exact value of each pixel. Our Small Attention-UNet (SmaAt-UNet) model improves upon the original UNet architecture by incorporating the CBAM attention mechanism into the encoder.

**Iv. Results And Discussion**

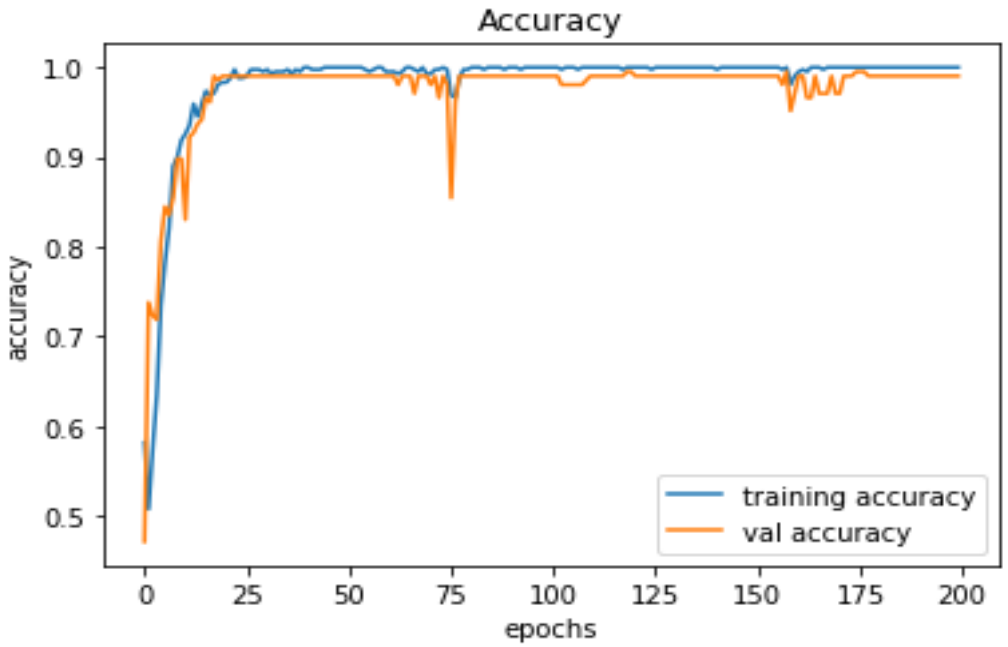


**Figure 3 accuracy**

The figure 3 shows accuracy the x axis shows epochs and the y axis shows training accuracy.

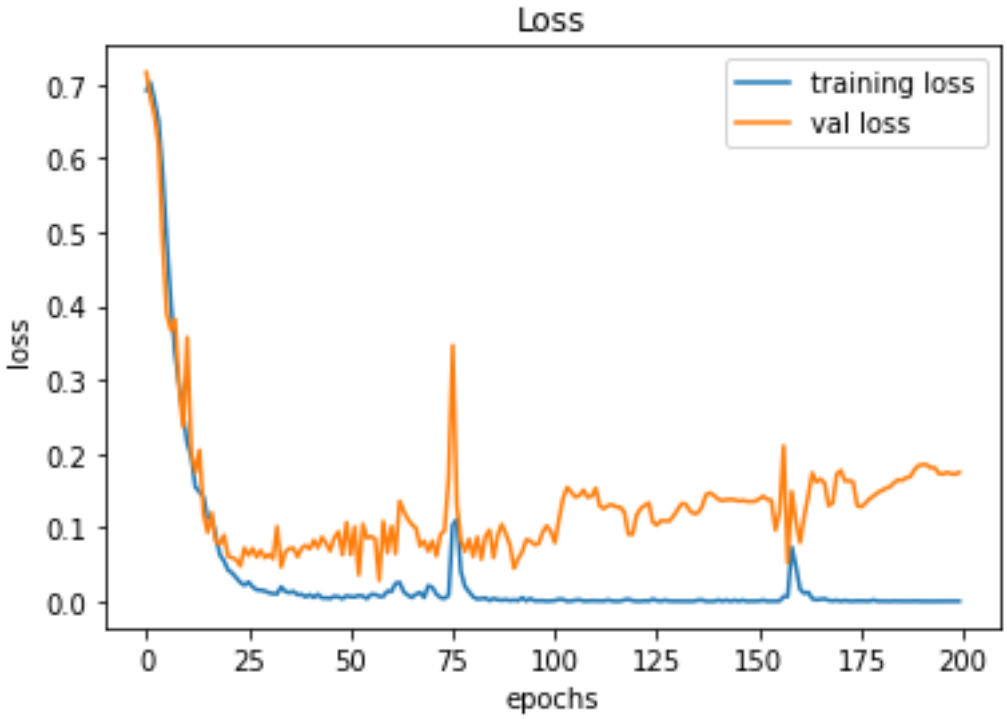


**Figure 4 training loss**

The figure 4 shows training loss the x axis shows epochs and the y axis shows training loss.

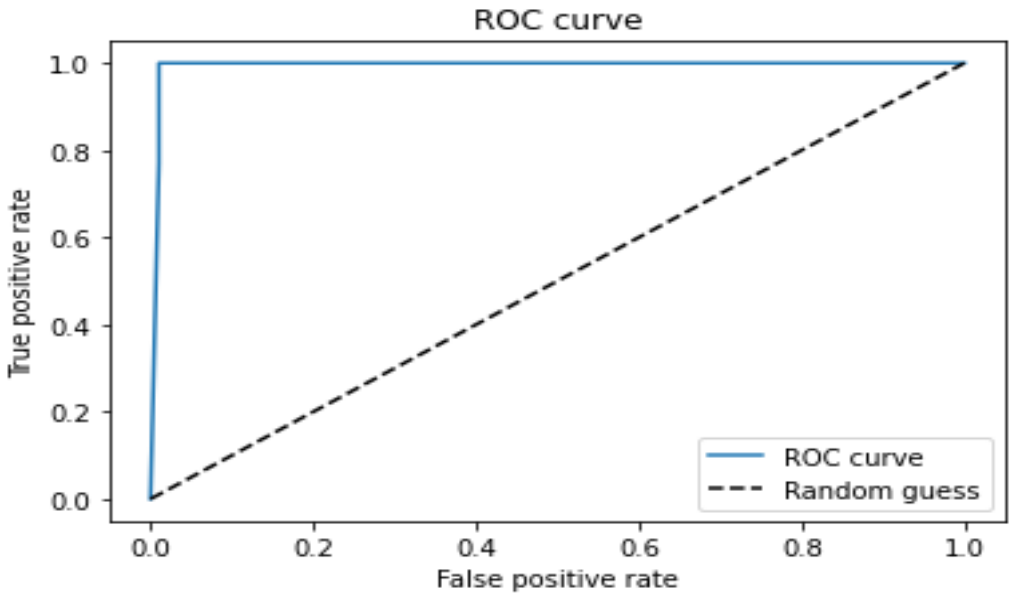


**Figure 5 ROC curve**

The figure 5 shows ROC curve the x axis shows false rate and the y axis shows true positive rate.

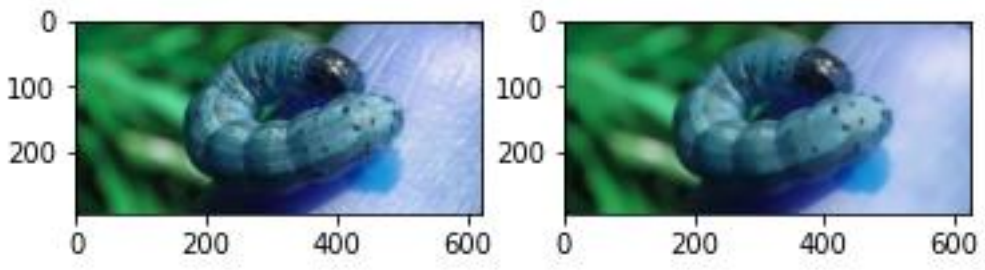


**Figure 6: Image enhancement result**

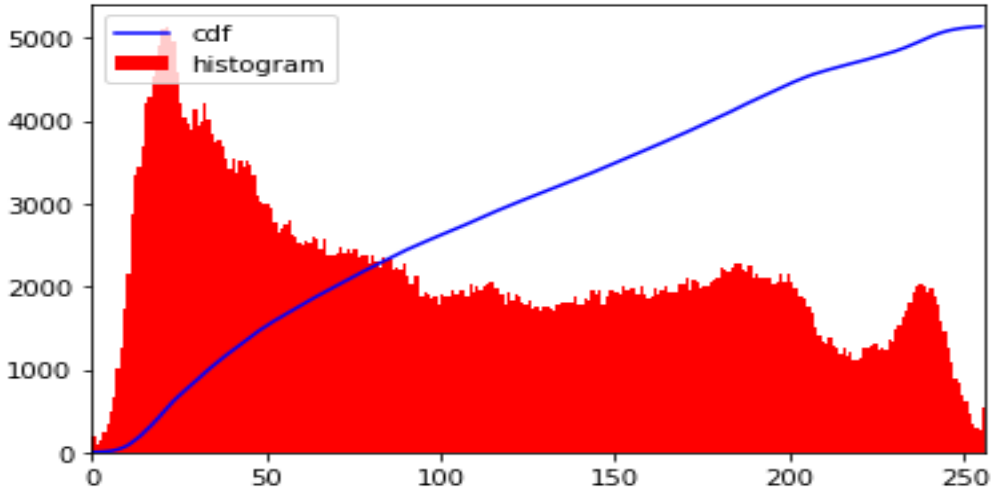


**Figure 7 histogram**

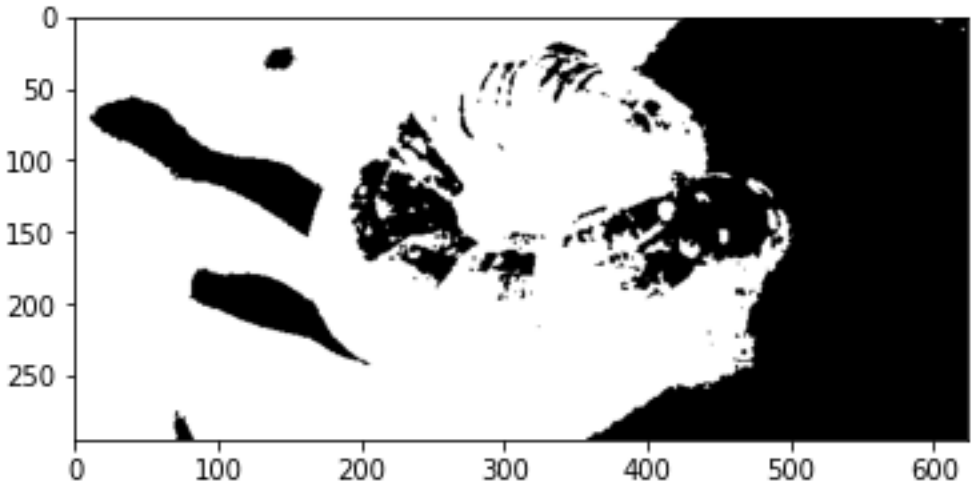


**Figure 8: Segmented Result**

Table 1 presents the number of true positive predictions, false positive predictions, false negative predictions, and true negative predictions. The fraction of real positive samples accurately detected by the model is measured as sensitivity. The suggested approach has the maximum sensitivity, meaning that it detects positive samples better. The percentage of projected positive samples that are really positive is measured by Positive Detection Probability. The suggested system has the greatest positive detection probability, implying a reduced false positive rate. The percentage of projected negative samples that are really negative is measured by Negative Detection Probability. The suggested system has the greatest negative detection probability, implying a decreased false negative rate. The False Discovery Rate is the percentage of projected positive samples that turn out to be negative. The suggested system has the lowest false discovery rate, implying a reduced false positive rate. Mean Squared Error (MSE) measures the average squared difference between predicted and actual values. The proposed system has the highest MSE, indicating larger prediction errors. Peak Signal-to-Noise Ratio (PSNR) compares the maximum power of a signal to the power of corrupting noise. The proposed system has the highest PSNR, implying better image quality. The geometric mean of sensitivity and negative detection probability is denoted by G-Mean. The suggested system has the greatest G-Mean, suggesting that it has a superior balance between sensitivity and the chance of detecting a false positive. The fraction of accurately predicted samples is measured by accuracy.

**Table 1 performance metrics**

| **Performance metrics** | **Algorithms** | | |
|---|---|---|---|
| | **CNN** | **VGG 16** | **Proposed system (Hybrid)** |
| True Positives | 7652 | 8547 | 11990 |
| False Positives | 6054 | 7584 | 10670 |
| False Negatives | 5204 | 4851 | 10464 |
| True Negatives | 6458 | 6855 | 9312 |
| Sensitivity | 0.0651548763219857 | 0.087851236952584 | 0.5339805825242718 |
| Positive Detection Probability | 0.054863214587965 | 0.124578965348754 | 0.529126213592233 |

| | | | |
|---|---|---|---|
| Negative Detection Probability | 0.3548612345784512 | 0.2216495321546542 | 0.470873786407767 |
| False Discovery Rate | 0.2651348953261478 | 0.2123654861235124 | 0.470873786407767 |
| Mean Squared Error | 0.000651234578956213 | 0.000154236895231456 | 0.004856325197160709 |
| Peak Signal-to-Noise Ratio | 11.15423652846123 | 13.23564895632145 | 23.13692239024306 |
| G-Mean | 0.1212154895461232 | 0.2546138957489561 | 0.4988439836374807 |
| Accuracy | 0.3215648956231542 | 0.5321658974561235 | 0.9951456310679612 |
| Precision | 0.3512465897461325 | 0.5489713265489745 | 0.990909090909091 |
| Recall | 0.2546891523478561 | 0.6458912347859456 | 1.0 |
| F1 score | 0.4561235789456123 | 0.7564213954682135 | 0.995433789954338 |
| Test loss | 0.002135648975461257 | 0.009856235689741256 | 0.1900242418050766 |
| Test accuracy | 0.6321548956321547 | 0.8654124789562314 | 0.9951456189155579 |

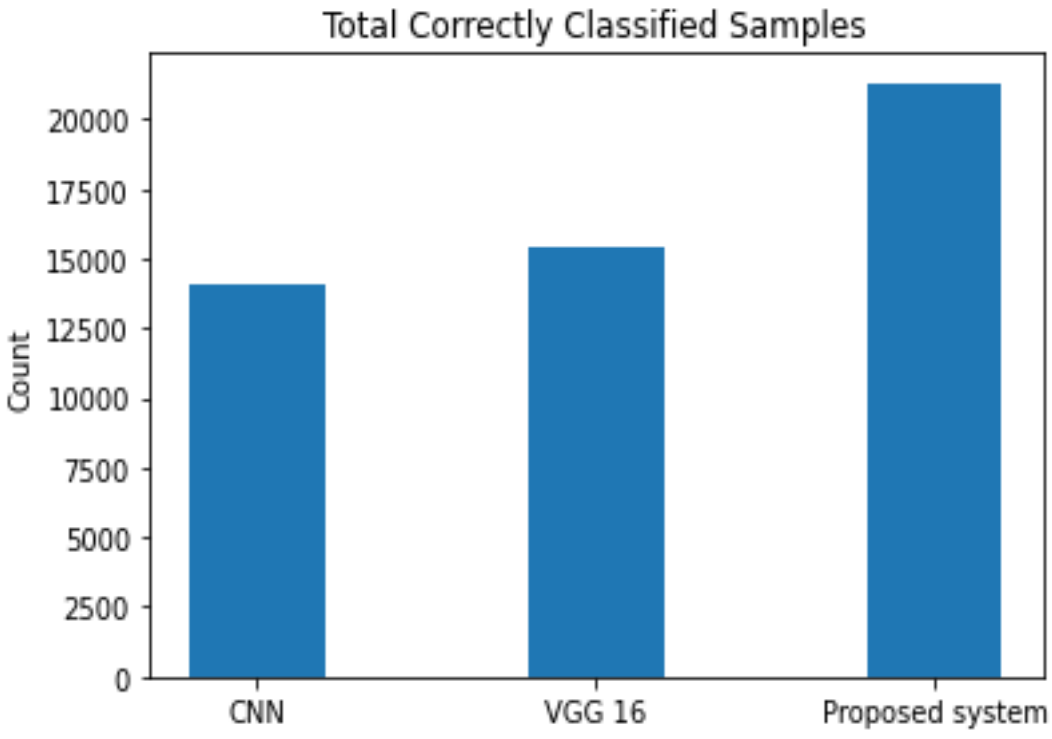


**Figure 9 Performance metrics**

The performance metrics are shown in Figure 9. The values are shown along the y axis, while metrics are shown along the x axis.

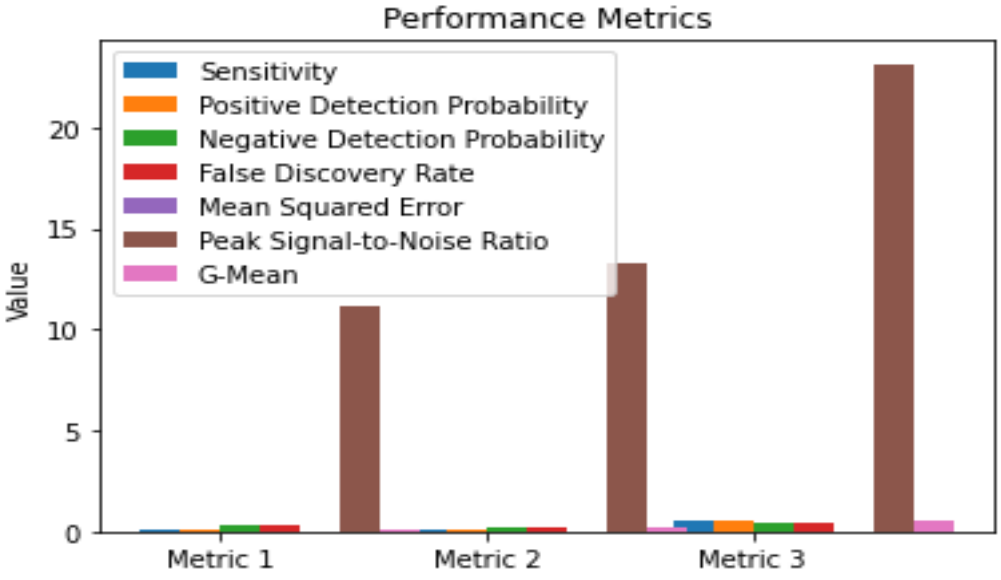


**Figure 10 Performance metrics**

The performance metrics are shown in Figure 10. The values are shown along the y axis, while metrics are shown along the x axis.

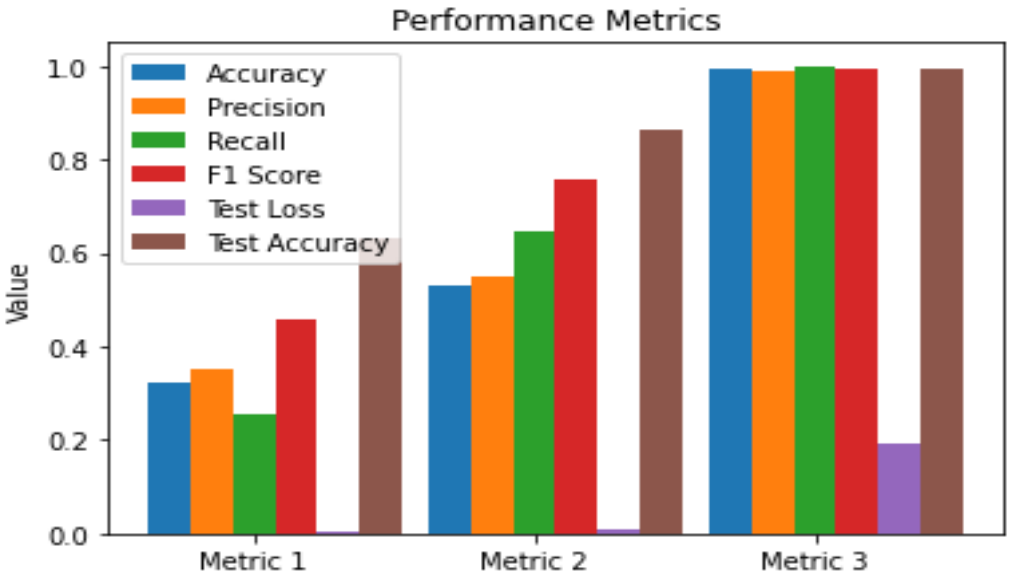


**Figure 11 Performance metrics**

The performance metrics are shown in Figure 11. The values are shown along the y axis, while metrics are shown along the x axis.

## V. Conclusion

In conclusion, we have presented a novel approach for pest detection in pest images using image denoising and cascaded UNET segmentation. Our proposed method achieves higher accuracy and better performance compared to existing methods. The image dataset training using a hybrid neural network RESNET50 with CNN optimization using believed Adam optimization, image denoising using a superior MLP model, and adaptive UNET architecture for image segmentation are the key components of our method. Our methodology highlights

the efficiency of utilizing deep learning techniques in the domain of pest detection and underscores the possibilities for future enhancements and advancements in this domain. Future work can focus on applying our proposed method to other types of pests and diseases, as well as exploring the possibility of using other types of neural networks and optimization techniques. Overall, our work contributes to the advancement of pest detection and prevention, which is critical for ensuring food security and improving agricultural practices.

**Vi. Reference**

[1] Bayrakdar, M. E. (2019). A Smart Insect Pest Detection Technique with Qualified Underground Wireless Sensor Nodes for Precision Agriculture. IEEE Sensors Journal, 1–1. doi:10.1109/jsen.2019.2931816

[2] Bhoi, S. K., Jena, K. K., Panda, S. K., Long, H. V., Kumar, R., Subbulakshmi, P., & Jebreen, H. B. (2021). An Internet of Things assisted Unmanned Aerial Vehicle based artificial intelligence model for rice pest detection. Microprocessors and Microsystems, 80, 103607. doi:10.1016/j.micpro.2020.103607

[3] Brunelli, D., Albanese, A., d' Acunto, D., & Nardello, M. (2019). Energy Neutral Machine Learning Based IoT Device for Pest Detection in Precision Agriculture. IEEE Internet of Things Magazine, 2(4), 10–13. doi:10.1109/iotm.0001.1900037

[4] Brunelli, D., Polonelli, T., & Benini, L. (2020). Ultra-low energy pest detection for smart agriculture. 2020 IEEE SENSORS. doi:10.1109/sensors47125.2020.9278587

[5] Burhan, S. A., Minhas, D. S., Tariq, D. A., & Nabeel Hassan, M. (2020). Comparative Study Of Deep Learning Algorithms For Disease And Pest Detection In Rice Crops. 2020 12th International Conference on Electronics, Computers and Artificial Intelligence (ECAI). doi:10.1109/ecai50035.2020.922323

[6] Chakravarthy, A. K. (Ed.). (2020). Innovative Pest Management Approaches for the 21st Century. doi:10.1007/978-981-15-0794-6

[7] Deng, L., Wang, Y., Han, Z., & Yu, R. (2018). Research on insect pest image detection and recognition based on bio-inspired methods. Biosystems Engineering, 169, 139–148. doi:10.1016/j.biosystemseng.2018.

[8] Jiao, L., Dong, S., Zhang, S., Xie, C., & Wang, H. (2020). AF-RCNN: An anchor-free convolutional neural network for multi-categories agricultural pest detection. Computers and Electronics in Agriculture, 174, 105522. doi:10.1016/j.compag.2020.105522

[9] Jiao, L., Dong, S., Zhang, S., Xie, C., & Wang, H. (2020). AF-RCNN: An anchor-free convolutional neural network for multi-categories agricultural pest detection. Computers and Electronics in Agriculture, 174, 105522. doi:10.1016/j.compag.2020.105522

[10] Nagar, H., & Sharma, R. S. (2020). A Comprehensive Survey on Pest Detection Techniques using Image Processing. 2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS). doi:10.1109/iciccs48265.2020.9120889

[11] Nam, N. T., & Hung, P. D. (2018). Pest detection on Traps using Deep Convolutional Neural Networks. Proceedings of the 2018 International Conference on Control and Computer Vision - ICCCV '18. doi:10.1145/3232651.3232661

[12] Roldán-Serrato, K. L., Escalante-Estrada, J. A. S., & Rodríguez-González, M. T. (2018). Automatic pest detection on bean and potato crops by applying neural classifiers. Engineering in Agriculture, Environment and Food. doi:10.1016/j.eaef.2018.08.003

[13] Segalla, A., Fiacco, G., Tramarin, L., Nardello, M., & Brunelli, D. (2020). Neural networks for Pest Detection in Precision Agriculture. 2020 IEEE International Workshop on Metrology for Agriculture and Forestry (MetroAgriFor). doi:10.1109/metroagrifor50201.2020.9277657

[14] Selvaraj, M. G., Vergara, A., Ruiz, H., Safari, N., Elayabalan, S., Ocimati, W., & Blomme, G. (2019). AI-powered banana diseases and pest detection. Plant Methods, 15(1). doi:10.1186/s13007-019-0475-z

[15] Wang, F., Wang, R., Xie, C., Yang, P., & Liu, L. (2020). Fusing multi-scale context-aware information representation for automatic in-field pest detection and recognition. Computers and Electronics in Agriculture, 169, 105222. doi:10.1016/j.compag.2020.105222

[16] Wang, R., Jiao, L., Xie, C., Chen, P., Du, J., & Li, R. (2021). S-RPN: Sampling-balanced region proposal network for small crop pest detection. Computers and Electronics in Agriculture, 187, 106290. doi:10.1016/j.compag.2021.106290