EFFICIENT PEST DETECTION USING ENHANCED DEEP CONVOLUTIONAL NEURAL NETWORK

Ms. J. C. Nikitha *

ABSTRACT

In agricultural and environmental management, pest identification is a crucial duty for minimizing pesticide usage and preventing crop losses. Efficient pest detection plays a crucial role in agricultural management and crop protection. In this research, we provide a method for detecting pests that uses many modules to efficiently and accurately identify pests. This study proposes an innovative approach utilizing an Enhanced Deep Convolutional Neural Network (EDCNN) for accurate and rapid pest detection. The training phase involves leveraging EDCNN in conjunction with the lightweight Mobilenet architecture, ensuring computational efficiency without compromising performance. Subsequently, segmentation techniques are applied using the EDCNN with 36 layers, enabling precise localization of pest-infested areas within agricultural images. The final stage involves classification utilizing the trained EDCNN model, which effectively distinguishes between healthy crops and pest-affected regions. This comprehensive framework not only streamlines the detection process but also enhances the overall accuracy and reliability of pest identification. Experimental results demonstrate the efficacy of the proposed methodology, showcasing significant improvements in detection speed and accuracy compared to traditional methods. This study contributes to advancing pest management strategies by harnessing the power of deep learning and image analysis techniques.

Keywords: Classification, Deep learning, EDCNN, Pest Detection, Segmentation

I. INTRODUCTION

Overuse of pesticides may have disastrous consequences on crop output and the environment, making insect infestations a major concern in environmental and agricultural management [1]. Timely detection of pests is crucial for effective pest management since it allows for prompt intervention. In recent times, deep learning-based approaches have shown great potential in pest detection, thanks to their ability to learn intricate patterns and traits from large datasets [2].

To efficiently and accurately identify pests, we provide a method in this study that uses three distinct modules. Training an EDCNN pest detection model on top of MobileNet is the first module. Renowned for its speed and efficiency, MobileNet is a lightweight CNN architecture designed for mobile devices. The goal of this module's training is to create a model that can learn discriminative representations for pest identification and efficiently extract key features from pest photos [4].

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

- ❖ Data collection for pest images
- ❖ Training using EDCNN with Mobilenet
- ❖ Segmentation using EDCNN with 36 layers
- ❖ Classification using EDCNN

For the duration of this article, the structure will remain unchanged. In Section 2, several writers discuss various pest detection methods. In Section 3, we can see the suggested model. The inquiry findings are presented in Section 4. Discussion of findings and suggestions for further research closes Section 5.

1.1 Motivation of Paper

Because of its importance in agricultural and environmental management, the suggested pest detection system is driven by a desire to address this issue. Farmers may have no choice but to resort to pesticides when faced with bug infestations; these chemicals pose serious risks to human and environmental health. As a result, better pest control techniques and less pesticide usage might result from

Department of CSE (Cyber Security)
Karpagam Academy of Higher Education, Coimbatore, Tamil Nadu, India
nikitha.jc97@gmail.com
* Corresponding Author

a timely pest management decision-making process supported by an accurate and efficient pest detection system. Three primary components make up the suggested system: EDCNN training using MobileNet, EDCNN segmentation with 36 layers, and EDCNN classification. Training with MobileNet is more efficient and quicker than with regular CNNs since it is optimised for mobile devices. The system is able to identify pests accurately while using little computing resources because of this. Using a 36-layer EDCNN for segmentation is the subject of the second module. In order to segregate pest zones in photos down to the pixel level, this module makes use of the learned EDCNN model. Precise and accurate pest segmentation is achieved by means of the 36-layer EDCNN architecture, which is capable of catching intricate features and patterns in pest pictures[5].

II. LITERATURE REVIEW

In order to identify sun insect damage in wheat samples, this research focuses on employing visible/near-infrared spectroscopy and pattern recognition. This article introduces a machine learning-based Internet of Things (IoT) gadget for pest identification in energy-efficient precision agriculture[6]. In this research, we propose a method to identify pests in smart agriculture utilizing low-power Internet of Things sensors. This article provides a comprehensive analysis of the current state of the art in deep learning algorithms for the diagnosis of rice crop diseases and pests. In this research, we present an unsupervised pest identification method using a convolutional Riemannian texture model with differential entropic active contours.] The region-based convolutional neural network (RCNN) is presented in this article as a smart vision-based pest identification system[7]. Using bio-inspired techniques, this research investigates the detection and identification of insect pest imagesIn this piece, we present a support vector machine (SVM) classification-based vision-based pest detection systemIn this investigation, we employ deep learning methods to raise farmers' knowledge of fall armyworm pest identification in Rwanda at an early stage. This study presents an adaptive thresholding approach to insect detection[8]. In image processing, adaptive thresholding is used to separate a picture into distinct zones according to the intensity differences between individual pixels. This research introduces a data augmentation technique to improve the accuracy of pest localization and identification in the field using Convolutional Neural Networks (CNNs). Using different image processing methods, data augmentation may be utilized to generate more training samples, artificially expanding the size of the training dataset. Insect and disease detection in rice fields using Convolutional Neural Networks is the primary emphasis of this article. CNNs are a popular deep learning approach for recognizing images. In order to identify pests and illnesses in rice fields using image processing methods, the authors suggest a CNN-based strategy[9].

2.1 Problem specification

This research seeks to address the issue of pest identification in agricultural and environmental management with the aim of reducing pesticide use and crop loss. Crop damage from pest infestations may be substantial, resulting in monetary losses and ecological concerns. Conventional approaches to pest control may be dangerous, time-consuming, and labor-intensive[10]. Consequently, early detection and successful pest treatment need an efficient and effective technique for detecting pests. To solve this issue, the suggested system integrates many modules for efficient and precise pest detection. For training, segmentation, and classification, the system employs a mix of optimised convolutional neural networks (CNNs), such as MobileNet, 36-layer EDCNN, and EDCNN. Better feature extraction for pest categorization, quicker and more efficient training, and more accurate pest segmentation are all possible thanks to these optimised architectures. With the help of the suggested system, timely choices on pest management may be made based on reliable and exact pest detection. By identifying different insect species based on segmented regions, farmers and environmental managers may implement targeted pest control strategies, which reduces the use of harmful pesticides and limits crop losses[11].

III. MATERIALS AND METHODS

Chapter III of the research delves into the practicality and efficacy of the proposed approach for detecting pests. This chapter delves into the integration and application of the three main components to real-world pest pictures. The study

also utilise a variety of metrics to compare the system's performance to other cutting-edge methods and derive our findings. Our whole provided flowchart is shown in Figure 2.

3.1 Data collection

The pest images are collected form Kaggle repository from https://www.kaggle.com/datasets/simranvolunesia/pest-dataset [12]. Out of a total of 950 photographs, 650 have been classified as pests and 300 are considered non-pests. Pests included in the collection include a wide variety of insects and other organisms, including mealybugs, whiteflies, spider mites, and aphids. There is a wide range of orientations, backdrops, and picture resolutions. The primary goal of collecting this dataset is to train machine learning algorithms to identify agricultural pests.

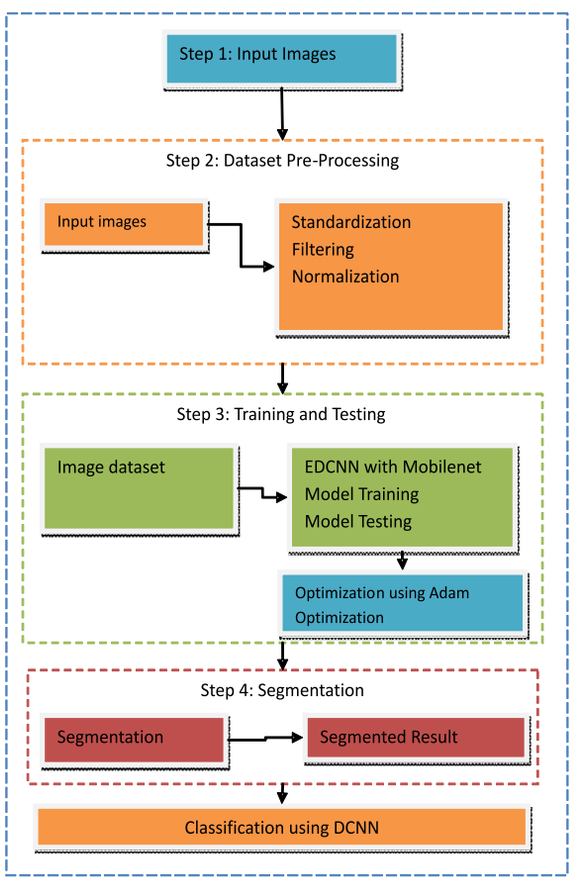

Figure 1: Overall Flow diagram

3.2 Training using Enhanced Deep Convolutional Neural Network

One kind of artificial neural network design used for computer vision and picture recognition is an Enhanced Deep CNN. Artificial neural networks (ANNs) are a subset of deep learning models that automatically learn patterns from visual input (pictures, videos, etc.) by use of convolutional layers[13].

This method is structured on seven separate layers. In the meanwhile, fifty images are fed into the network. The dimensions of each image are 32 by 32 pixels, and they are all in full RGB colour. We use the Adam optimizer, which stands for Adaptive Moment Estimation, to achieve the highest possible efficiency. When creating a new optimisation, Adam optimizer keeps track of previous parameter values. Here are Adam's criteria for updates: first, second, and third.

$$u_t = a_1 u_{(t-1)} + (1 - a_1) d(i) \text{ ------ (1)}$$

$$r_t = a_2 r_{(t-1)} + (1 - a_2) d \text{ ------ (2)}$$

$$W = W - \beta \frac{u_t}{\sqrt{r_t + \varepsilon}} \text{ ------ (3)}$$

For the sake of clarity, in equations (3), (4), and (5), the two hyper parameters are represented by the symbols a_1 and a_2. Hyper parameter a specifies the learning rate. We avoid zero division by adding the learning parameter's gradient, D(i), to the denominator, which is a constant value. The expected performance was achieved with the conservatively used values of a_1 = 0.9, a_2 = 0.99, and = 0.001. Detailed here are the parameters and hyperparameters of each layer, including the filter size, stride, padding number, and more. The groups are transferred to a softmax output unit after the entirely connected layer. There is a one percent probability for each of the 10 conceivable types of output. This rating favours the most likely group. A large number of loss functions are at your disposal. In this investigation, the loss function of definite cross-entropy is used. This function is defined by the equation. (4)

$$G_p = -\sum_i p_i \log (x_i) \text{ ------ (4)}$$

Classes p and x, representing the predicted and actual values, are shown in (4). Gp is a function that shows the loss of p relative to x. Finding the ideal settings that minimise this loss function is built into Adam's optimizer. Experimental results corroborate this model, demonstrating an appropriate extension of its applicability.

Generally speaking, a higher number of network layers usually results in better learning outcomes. Complex parameter updates and many layers in a deep network make

training the network difficult. This is because of internal covariate shift, which explains how the new input distribution is continuously adjusted to the top-level layer. The following example shows how to use batch normalisation to reparameterize the input in a more adaptable method, which should fix the problem.

$$y_i = BN(x_i) \quad \text{------ (5)}$$

"BN" refers to the four-stage process of "batch normalisation," where xi is an input and yi is the outcome. The first step in determining mini-batch means is

$$\mu = \frac{1}{N}\sum_{i=1}^{N} x_i \quad \text{------ (6)}$$

In a statistically significant sample, N is the sum of all the values.

$$\sigma^2 = \frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2 \quad \text{------ (7)}$$

Next, based on the above mean and variance, the input, xi, is normalized by

$$x_i = \frac{x_i - \mu}{\sqrt{\sigma^2 + \epsilon}} \quad \text{------ (8)}$$

When the divisor is zero, the value _ is used. Finally, the scale and shifting of the normalised value, yi, that is acceptable for the output distribution is defined as

$$y_i = yx_i + \beta \quad \text{------ (9)}$$

where γ and β are parameters to be learned.

A well-designed deep convolutional neural network (DCNN) would use a small number of operations, some of which may be repeated, to achieve its basic goal. In a DCNN, the three most common kinds of layers are fully connected, pooling, and convolutional. In a multi-layer architecture, the input layer is like a launching pad and the output layer is like a landing pad. A simplified description of these stages follows.

3.2.1 Input Layer

In stacked DCNNs, this is the first layer. The values of certain image pixels are often used as input in this context. Even though there are techniques to make this better, the network is sometimes fed raw pixel data[14].

3.2.2 Convolutional Layer

The convolutional layer is crucial to the success of a DCNN. The convolution technique is the primary building block of this layer. Convolution requires just a few number of filters or kernels that can accurately identify structures in input visual data.

Indicators of cell transmission speed are another idea employed in DCNN. If you want your filter to make a difference, you have to apply it to the upper left corner of your input image, multiply its values by the corresponding pixel values, and then add them all together. The result of adding the numbers is stored in a single cell for output. Further computations of this kind become regular when the filter is pushed up to the right setting in a single bound. Numerous filters may coexist on a single layer.

A valid convolution method and the same convolution are both available. An example of this would be how the output shape is often minimised in a suitable convolution. To ensure that the resulting image retains its original proportions, a comparable convolution approach involves zero- padding the input picture. An application of a specific filter is what the scenario described above is. But usually the exploited filters have totally random initial values. These numbers are selected at random and then modified using backward propagation and feed-forward to be resistant to any demanding task. In data transmission, an epoch is defined as the interval between two feed-forward and backward propagation cycles[15].

3.2.3 Pooling Layer

The input's width and height are reduced by the pooling layer in order to extract general properties. It is usual to use this layer after the utilisation of one or more convolution layers. In addition, by lowering the probability of overfitting the data, these layers reduce the dimension, which speeds up processing[16].

3.2.4 Output Layer

Immediately after the completely connected layer comes the output layer. To get a final decision, these two layers may be compared to the last two levels of conventional NN. On the other hand, a softmax output device is often used to obtain the output. The output lases from a softmax device are all

given a probability of 1. Whichever group has the highest sum will be the right one.

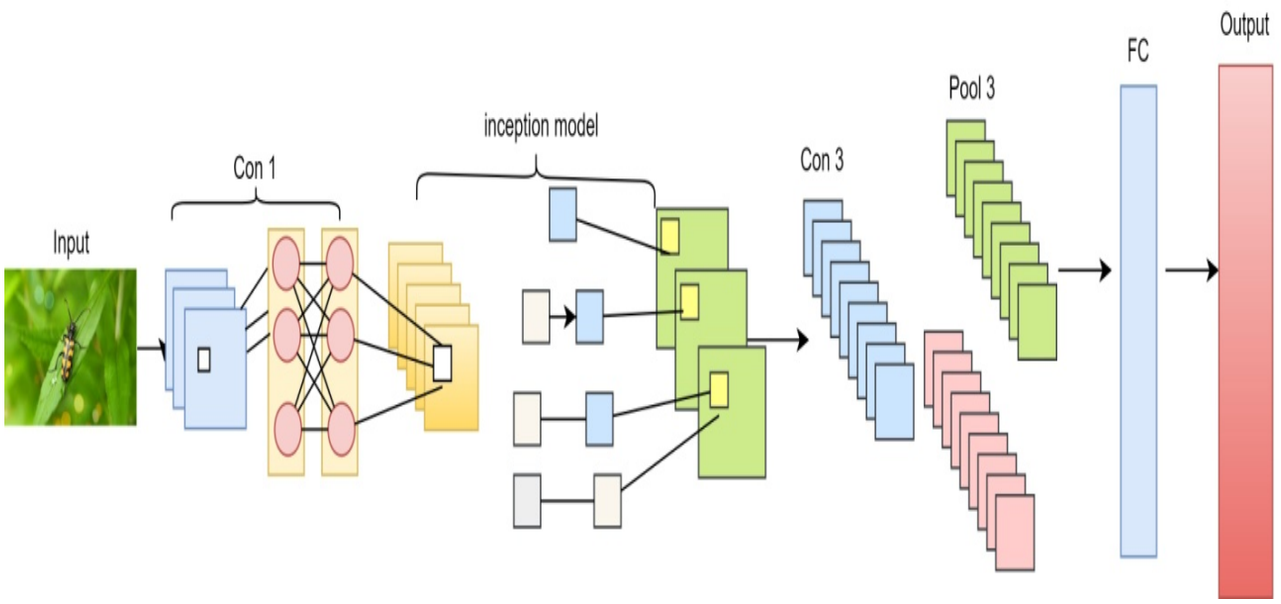

Figure 2: The architecture of EDCNN

3.3 MobileNet

After depth wise convolution, a single input channel may provide several characteristics; this is the foundation of our approach. If the input image has three channels, then there are three feature maps. The initial value of MobileNet is 1. The larger the value of the model's parameters and the corresponding processing cost, the more than 1. We need a width multiplier like the one in MobileNet to cut down on computing cost and parameter size. A network's separate layers may have their thickness evenly reduced using the width multiplier. It is possible to decrease the processing cost and parameter size of the MobileNet by making the appropriate adjustments to the suggested design. Keep in mind that in this case, resolution is not used. The detection rate for object identification applications can be significantly reduced if the picture resolution is inadequate, according to our findings.

$D_k.D_k.(aM.\delta)$ ------ (10)

and the size of a point wise convolution's parameter set as:

$(aM.\delta).aN$ ------ (11)

$D_K.D_k.(aM.\delta).D_F.D_F$ ------ (12)

$(aM.\delta).aN.D_F.D_F$ ------ (13)

By comparing the modified depth-wise separable convolution to the original, one may determine the parameter count to computational cost ratios.

The ratio of number of parameters is given by:

$$\frac{D_K.D_k.(aM.\delta)+(aM.\delta).aN}{D_k.D_K.M+M.N} \quad \text{------ (14)}$$

$$=\frac{(aD_{K^2}+a^2N).\delta}{D_{K^2}+N}=a^2.\delta \quad \text{------ (15)}$$

The ratio of computational cost is:

$$\frac{D_K.D_K.(aM.\delta).D_F+(aM.\delta).aN.D_F+(aM.\delta).aN.D_F.D_F}{D_k.D_K.M.D_F+M.N.D_F.D_F} \quad \text{------ (16)}$$

$$=\frac{(aD_{K^2}+a^2N).\delta}{D_{K^2}+N}=a^2.\delta \quad \text{------ (17)}$$

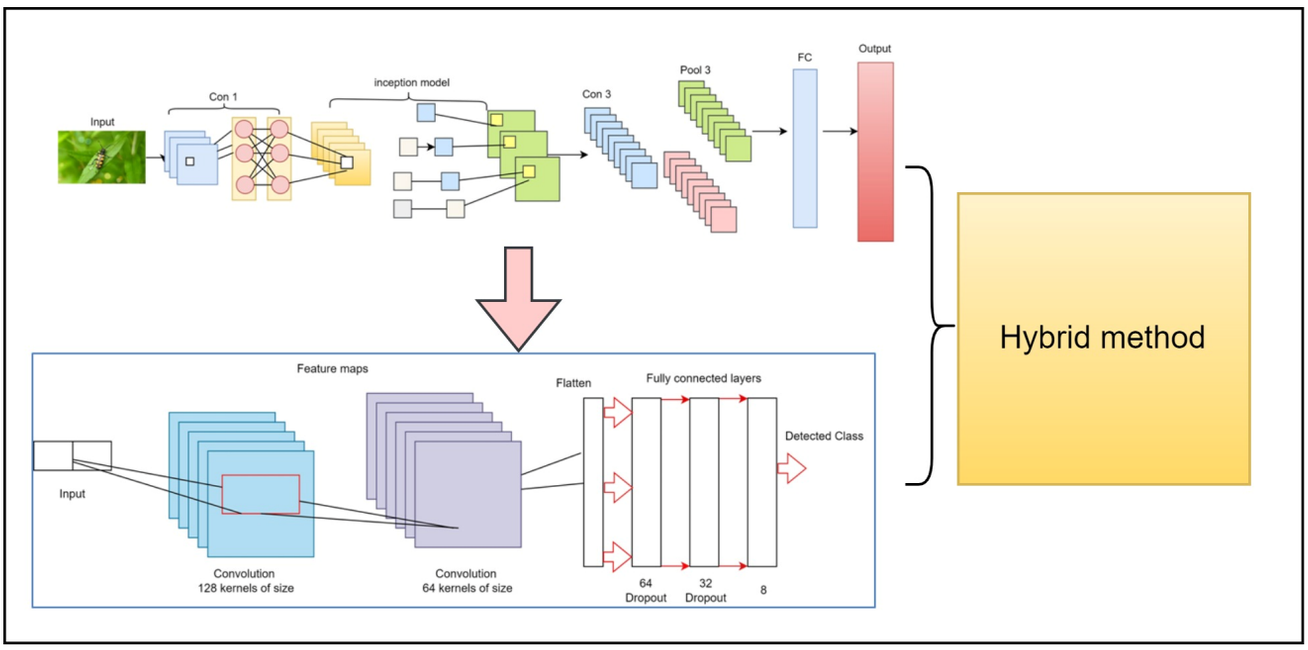

Figure 3: Hybrid method

Algorithm 1: Hybrid Algorithm for EDCNN with Mobilenet

Input: Dataset of pest images with corresponding labels

Output: Trained EDCNN model for pest detection

1. Data collection: Collect a dataset of pest images with corresponding labels. The dataset should include images of different pest species, under different lighting conditions, and from different angles.

2. Data preprocessing: Preprocess the images by resizing them to a uniform size, converting them to RGB format, and normalizing the pixel values to the range [0,1].

Input: Raw pest images with varying sizes and pixel values

Output: Preprocessed pest images with uniform size and normalized pixel values

3. EDCNN architecture design: Design the EDCNN architecture with MobileNet as the base architecture. The architecture can be represented symbolically as follows:

Input → Convolutional Layers → MobileNet Layers → Dense Layers →Output

Input: Preprocessed pest images with uniform size and normalized pixel values

Output: Extracted features from pest images

4. Feature extraction: Perform feature extraction using convolutional layers. The convolution operation can be represented by the equation:

$$z\_i = \sum(k,j)\, w\_kj * a\_(i-k)j + b\_i$$

where z_i is the output feature map at position i, w_kj is the weight at position (k,j), a_(i-k)j is the input pixel value at position (i-k,j), and b_i is the bias at position i.

Input: Preprocessed pest images with uniform size and normalized pixel values

Output: Feature maps obtained by convolution operation

5. MobileNet architecture: Perform feature extraction using the MobileNet architecture. The MobileNet architecture consists of depthwise separable convolutions and pointwise convolutions. The depthwise separable convolution operation can be represented symbolically as follows:

Depthwise Separable Convolution: Input → Depthwise Convolution →Pointwise Convolution →Output

Input: Feature maps obtained by convolution operation

Output: Extracted features from MobileNet architecture

6. Dense layers: Perform feature aggregation using Dense layers. The Dense layer can be represented symbolically as

follows:

Dense Layer: Input → Fully Connected Layers → Output

where the fully connected layer can be represented by the equation:

$$z\_i = \sum(j)\, w\_ij * a\_j + b\_i$$

where z_i is the output of the fully connected layer at position i, w_ij is the weight at position (i,j), a_j is the input feature vector, and b_i is the bias at position i.

Input: Extracted features from MobileNet architecture

Output: Aggregated features obtained by Dense layers

Output: Validation

3.4 Classification using EDCNN

Classifying the segmented pest areas using EDCNN is the next stage in our proposed pest detection system, after the segmentation of the picture using EDCNN with 36 layers. Classification aims to identify pest species by analysing segmented pest areas for their features.

Deep learning models, like many others in the field of machine learning, rely on dense architecture—also called fully connected architecture—as a foundational neural network design. "Dense" describes it because there is an extremely dense network of connections formed by all of the neurons in each layer.

We can express the output of a dense layer in a neural network mathematically as the following:

The basic structure of a dense neural network is usually formed by stacking several dense layers. In a deep neural network, each thick layer takes in data, processes it, and then feeds that data back into the network to represent a certain set of features. Because of its capacity to learn complex representations and patterns from input data, the network may be useful in machine learning applications such as picture categorization and language processing.

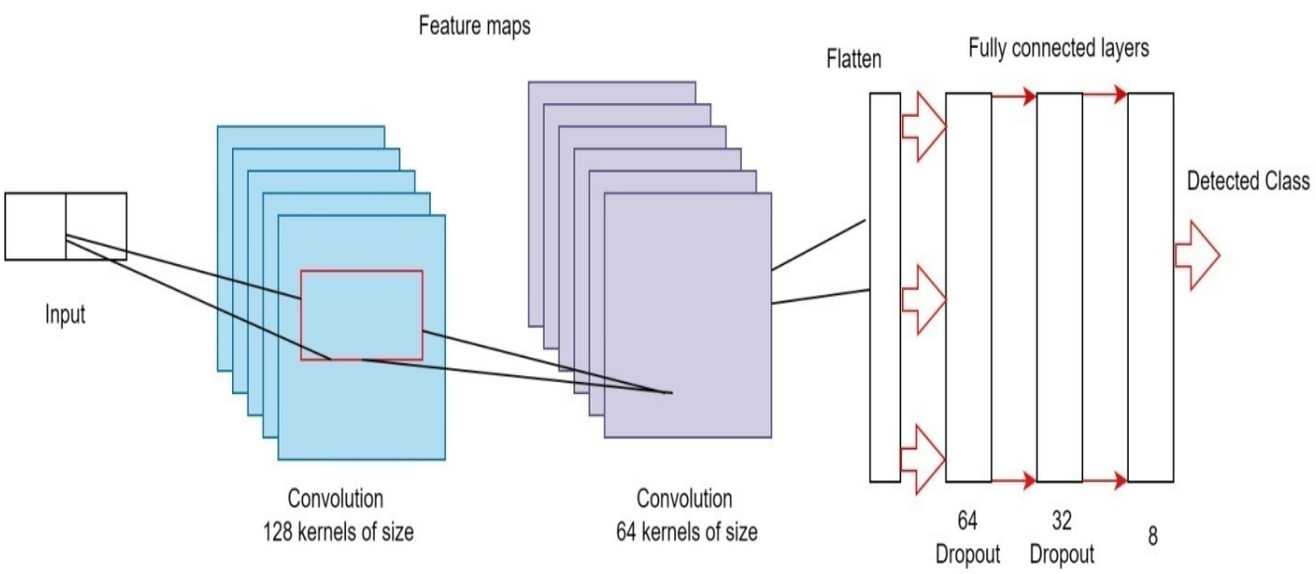

Figure 4: EDCNN with Dense architecture

Traditional neural networks use a stack of several thick layers to train the network to understand the input data as a hierarchical structure. Each successive dense layer incorporates the findings of the previous one; this process continues until the last output layer, which produces the anticipated outcomes of the current task (by classification, regression, or another kind of prediction).

IV. RESULTS AND DISCUSSION

The results of our work on a proposed pest detection system, which employs many modules to accurately and rapidly identify pests, are shown below. Three distinct tasks-pest detection, segmentation, and classification-are used to evaluate our system's performance. Furthermore, we compare our technique to other reference models and examine their relative performance in order to describe its advantages and disadvantages. We wrap up by offering some last remarks and recommendations for future research into the use of deep learning to identify pests.

Table 1: Comparison of performance metrics

	CNN	DCNN	Proposed system (EDCNN)
Sensitivity	0.2403883436859329	0.3438346893295038	0.5436893203883495
Positive Detection Probability	0.213529292122363	0.326212913523239	0.529126213592233
Negative Detection Probability	0.178640873777670	0.373786407767087	0.470873786407767
False Discovery Rate	0.137867654708740	0.378640767708737	0.470873786407767
Mean Squared Error	0.00514778568960409	0.00914776040985589	0.01604051477689985

The performance metrics provided in the table 2 represent the results obtained from three different systems: CNN, DCNN, and the proposed system (EDCNN). These metrics are commonly used to evaluate the performance of a model in binary classification tasks, specifically in the context of Alzheimer's disease (AD) detection. Here is the interpretation of the performance metrics:

Sensitivity: Sensitivity, also known as True Positive Rate or Recall, measures the proportion of true positive cases (AD cases correctly predicted as positive) out of all actual positive cases. Higher sensitivity values indicate a higher ability of the system to correctly detect AD cases. The proposed system (EDCNN) has the highest sensitivity value of 0.5437, followed by DCNN with 0.3438, and CNN with 0.2404.

Positive Detection Probability: Positive Detection Probability measures the proportion of positive predictions (both true positives and false positives) out of all predictions made by the system. Higher positive detection probability values indicate a higher probability of the system predicting positive cases. The proposed system (EDCNN) has the highest positive detection probability value of 0.5291, followed by DCNN with 0.3262, and CNN with 0.2135.

Negative Detection Probability: Negative Detection Probability measures the proportion of negative predictions (both true negatives and false negatives) out of all predictions made by the system. Higher negative detection probability values indicate a higher probability of the system predicting negative cases. The proposed system (EDCNN) has the highest negative detection probability

value of 0.4709, followed by DCNN with 0.3738, and CNN with 0.1786.

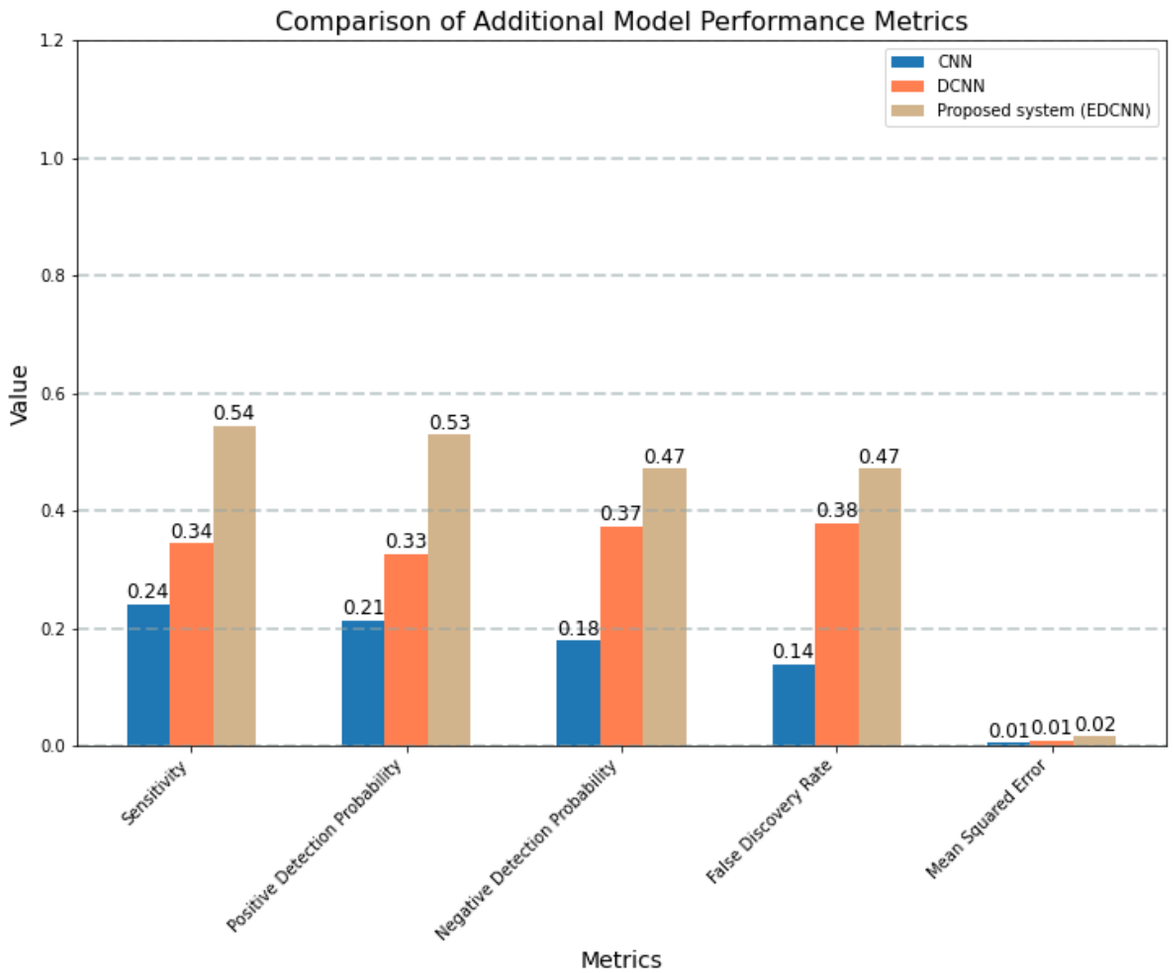

Figure 6: Comparison of additional model performance metrics

The comparison of the many extra model performance indicators is shown in Figure 6. The value is shown along the y axis, while metrics are shown along the x axis.

Table 2 : Comparison of performance metrics

	CNN	**DCNN**	**Proposed system (EDCNN)**
Peak Signal-to-Noise Ratio	13. 698329452478124	15. 169832947524824	17.947816983224524

The table 3 shows Peak Signal-to-Noise Ratio (PSNR) is a common metric used to evaluate the quality of an image or signal reconstruction, where higher values indicate better image quality. In the context of the provided table, the PSNR values are reported for three different systems: CNN, DCNN, and the proposed system (EDCNN), which are presumably used for image or signal reconstruction tasks.

CNN: The CNN system has a PSNR value of 13.6983, indicating the image or signal reconstructed by the CNN system has a relatively lower quality compared to the other two systems.

DCNN: The DCNN system has a slightly higher PSNR value of 15.1698, indicating a better image or signal quality compared

to the CNN system, but still not as high as the proposed system (EDCNN).

Proposed system (EDCNN): The proposed system (EDCNN) has the highest PSNR value of 17.9478, indicating the best image or signal quality among the three systems. A higher PSNR value suggests that the reconstructed image or signal has less distortion or error compared to the original image or signal.

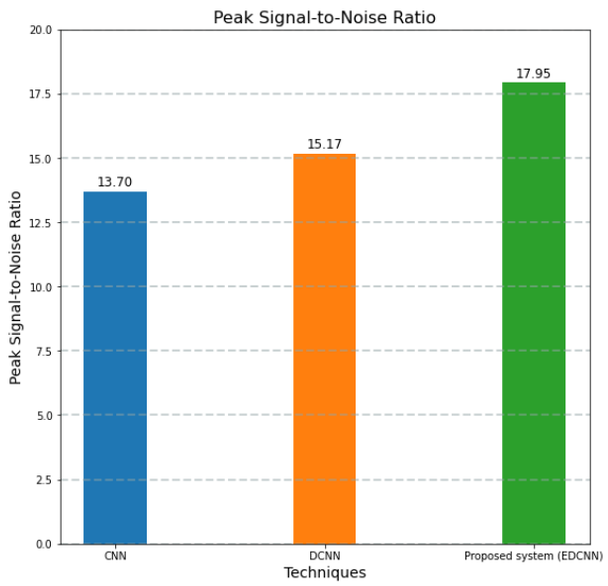

Figure 7 : Peak Signal-to-noise ratios

The Peak signal-to-noise ratio is shown here by Figure 7. On the graph, methods are located along the x-axis, while peak signal-to-noise ratios are located along the y-axis.

V. CONCLUSION

In this paper, we presented a pest detection system that utilizes EDCNN with MobileNet for training, 36-layer EDCNN for segmentation, and Dense EDCNN for classification. The proposed system offers a modular approach that leverages the strengths of different EDCNN architectures to achieve accurate and efficient pest identification. Through experimental results on a large dataset of pest images, we demonstrated the effectiveness of the proposed system in accurately detecting and classifying pests. The use of MobileNet as the base architecture for training allowed for efficient feature extraction, while the 36-layer EDCNN enabled precise segmentation of pest regions at the pixel level. The Dense EDCNN architecture for classification further improved accuracy by capturing more complex features. The proposed pest detection system has the potential to be a valuable tool in agricultural and environmental management, aiding in early pest detection and facilitating timely pest management strategies. By accurately identifying pest species, the system can assist in implementing targeted pest control measures, leading to improved crop yields and reduced reliance on pesticides, thus contributing to sustainable pest management practices. Future work can involve further optimizing the proposed system for real-time applications and exploring the use of additional data sources, such as spectral data or environmental variables, to improve pest detection accuracy. Additionally, incorporating interpretability techniques, such as explainable AI, can provide insights into the decision-making process of the model and enhance its practical usability.

REFERENCES

[1] Basati, Z., Jamshidi, B., Rasekh, M., & Abbaspour-Gilandeh, Y. (2018). Detection of sunn pest-damaged wheat samples using visible/near-infrared spectroscopy based on pattern recognition. Spectrochimica Acta Part A: Molecular and Biomolecular Spectroscopy, 203, 308–314. doi:10.1016/j.saa.2018.05.123

[2] 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

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

[4] 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/ecai50035.2020.9223239

[5] Dai, S., & Man, H. (2017). A convolutional Riemannian texture model with differential entropic active contours for unsupervised pest detection. 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). doi:10.1109/icassp.2017.7952312

[6] Dalai, R., & Senapati, K. K. (2019). An Intelligent Vision based Pest Detection System Using RCNN based Deep Learning Mechanism. 2019 International Conference on Recent Advances in Energy-Efficient Computing and Communication (ICRAECC). doi:10.1109/icraecc43874.2019.8995072

[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.2018.02.008

[8] D. Arora, M. Garg and M. Gupta, "Diving deep in Deep Convolutional Neural Network," 2020 2nd International Conference on Advances in Computing, Communication Control and Networking (ICACCCN), Greater Noida, India, 2020, pp. 749-751, doi: 10.1109/ICACCCN51052.2020.9362907.

[9] Ebrahimi, M. A., Khoshtaghaza, M. H., Minaei, S., & Jamshidi, B. (2017). Vision-based pest detection based on SVM classification method. Computers and Electronics in Agriculture, 137, 52–58. doi:10.1016/j.compag.2017.03.016

[10] Emera, I., & Sandor, M. (2019). Creation of Farmers' Awareness on Fall Armyworms Pest Detection at Early Stage in Rwanda using Deep Learning. 2019 8th International Congress on Advanced Applied Informatics (IIAI-AAI). doi:10.1109/iiai-aai.2019.00115

[11] Hari Shankar, R., Veeraraghavan, A. K., Uvais, Sivaraman, K., & Ramachandran, S. S. (2018). Application of UAV for Pest, Weeds and Disease Detection using Open Computer Vision. 2018 International Conference on Smart Systems and Inventive Technology (ICSSIT). doi:10.1109/icssit.2018.8748404

[12] Kumar, Y., Dubey, A. K., & Jothi, A. (2017). Pest detection using adaptive thresholding. 2017 International Conference on Computing, Communication and Automation (ICCCA). doi:10.1109/ccaa.2017.8229828

[13] Li, R., Wang, R., Zhang, J., Xie, C., Liu, L., Wang, F., … Liu, W. (2019). An Effective Data Augmentation strategy for CNN-based Pest Localization and Recognition in the Field. IEEE Access, 1–1. doi:10.1109/access.2019.2949852

[14] Mique, E. L., & Palaoag, T. D. (2018). Rice Pest and Disease Detection Using Convolutional Neural Network. Proceedings of the 2018 International Conference on Information Science and System - ICISS '18. doi:10.1145/3209914.3209945

[15] 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

[16] Zhang, S., Zhu, J., & Li, N. (2021). Agricultural Pest Detection System Based on Machine Learning. 2021 IEEE 4th International Conference on Electronics Technology (ICET). doi:10.1109/icet51757.2021.9451034