Complex & Intelligent Systems
https://doi.org/10.1007/s40747-022-00847-x

ORIGINAL ARTICLE

Custom CornerNet: a drone-based improved deep learning technique for large-scale multiclass pest localization and classification

Waleed Albattah[1] · Momina Masood[2] · Ali Javed[2] · Marriam Nawaz[2] · Saleh Albahli[1]

Received: 30 September 2021 / Accepted: 5 August 2022

Abstract

Insect pests are among the most critical factors affecting crops and result in a severe reduction in food yield. At the same time, early and accurate identification of insect pests can assist farmers in taking timely preventative steps to reduce financial losses and improve food quality. However, the manual inspection process is a daunting and time-consuming task due to visual similarity between various insect species. Moreover, sometimes it is difficult to find an experienced professional for the consultation. To deal with the problems of manual inspection, we have presented an automated framework for the identification and categorization of insect pests using deep learning. We proposed a lightweight drone-based approach, namely a custom CornerNet approach with DenseNet-100 as a base network. The introduced framework comprises three phases. The region of interest is initially acquired by developing sample annotations later used for model training. A custom CornerNet is proposed in the next phase by employing the DenseNet-100 for deep keypoints computation. The one-stage detector CornerNet identifies and categorizes several insect pests in the final step. The DenseNet network improves the capacity of feature representation by connecting the feature maps from all of its preceding layers and assists the CornerNet model in detecting insect pests as paired vital points. We assessed the performance of the proposed model on the standard IP102 benchmark dataset for pest recognition which is challenging in terms of pest size, color, orientation, category, chrominance, and lighting variations. Both qualitative and quantitative experimental results showed the effectiveness of our approach for identifying target insects in the field with improved accuracy and recall rates.

Keywords Classification · CornerNet · DenseNet · Deep learning · Pest recognition

Introduction

The agriculture sector is the most significant part of the economy, and the majority population of the entire world depends on it. Meanwhile, the recent growth in the world's population rate necessitates an increase in crop quantity to meet the food requirements of people all over the world [1]. However, several challenging factors like the climate conditions and crop pests hinder the farmers in taking care of crops and improving the yield. Conventional crop pest identification is performed by manually arranging pest traps to analyze and assess the category of pests in crops. However, such methods are unreliable and often suffer from high error rates.

Furthermore, we often experience delays in the pest detection process due to the limited availability of agronomists. Moreover, because of the lack of technical information about the various pest types and high visual similarity between different insect species, it is difficult to accurately specify the related insecticide types, resulting in an extensive and blind application of pesticides [2]. The delay in pest recognition can prevent the farmers from taking timely measurements, which causes massive damage to both the quality and quantity of crops. However, identifying the type and amount of crop pests is a tedious and time-consuming activity. At the same time, early recognition and usage of timely spraying pesticides on the plants can improve the yield and enhance the economy. Recent progress in machine learning (ML) and computer vision (CV) has urged the researcher present computer-aided approaches to simplify this time-consuming

✉ Saleh Albahli
Salbahli@qu.edu.sa

[1] Department of Information Technology, College of Computer, Qassim University, Buraydah, Saudi Arabia

[2] Department of Computer Science, University of Engineering and Technology-Taxila, Taxila 47050, Pakistan

Published online: 25 August 2022

task and build effective automated insect identification systems.

Initially, conventional ML-based approaches used local descriptors like local binary pattern (LBP) [3], regional ternary pattern (LTP) [4], SIFT [5], SURF [6], etc., with classifiers such as SVM [7], K-means [8], etc. Such approaches are heavily explored for pest detection and classification [9–14]. Although hand-crafted keypoint computation techniques are easier to use and require less training data, they are slow and require the skill of experienced human specialists. Moreover, the recent progression in image acquisition has introduced such challenging datasets. These conventional approaches employ ML as ready-to-use solutions are not promising for real-world pest detection, and they face a severe reduction in classification performance. This might be because of several reasons, such as ineffective hand-crafted keypoints computation.

Moreover, the exact pest can appear with different motions and positions in various images, which causes to generate the varying keypoints vectors for the same insects. Primarily, the research community is working to improve the detection performance for specific categories of insects by proposing new keypoints solutions. Such work lacks focus on introducing novel frameworks for multi-category pest recognition missions that need to acquire insect localization and classification information to assist in pest monitoring [15, 16].

Recently, we observed the robustness of Deep learning (DL)-based frameworks, i.e., convolution neural networks (CNNs) [17], recurrent neural networks (RNNs) [18], deep belief networks [19] in a variety of areas covering the agriculture sector as well. DL is a robust approach for image analysis and object recognition with superior effectiveness in classifying various categories of pests [20]. Transfer learning is an essential task in DL, in which pre-trained frameworks are modified to perform a new job. Deep transfer learning (DTL) proposes a new network for processing digital samples and performing predictive analytics, with better generalization power and the ability for pest recognition. DL-based approaches employ CNNs that can automatically detect the discriminative keypoints from input data without the assistance of human specialists. Because of the considerable evolution of hardware, DL frameworks are extensively being used to handle complicated problems in a reasonable amount of time. In agriculture, DL-based algorithms have shown to be entirely accurate and have been effectively adapted to perform various tasks [21].

As progress in DL methods [22, 23] has exhibited promising results in the area of object identification, so, the extensive research work concerns proposing more complicated object localization frameworks for better detection accuracy, i.e., Super-FAN [22] and unsupervised multi-stage keypoints learning [23], etc. Moreover, several CNN-based approaches namely GoogLeNet [24], AlexNet (AN) [25], VGG [26], ResNet (RN) [27], R-CNN [28], Fast R-CNN [29], Faster R-CNN [30], and YOLO [31] are also evaluated for pest detection and classification. Even though the aforementioned DL-based object identification frameworks have demonstrated robust performance in general object identification systems, their applications for pest detection are still limited. Pest recognition has its characteristics and differs from existing object identification and classification tasks [21]. The insect pests are small target and they are usually surrounded by complex environment in the real-field images; thus, the identification network can be easily deceived by the background while computing keypoints. In addition, because of the varying angles and distance at which they are captured in the field, there is a considerable difference in pest size and postures which makes accurate recognition more challenging. Moreover, distinct insect pest species often have a high degree of resemblance in appearance, and the same species may exist in many states such as egg, larva, pupa, and adult, indicating a large intra and inter-class variances. Furthermore, the existence of poor lighting and harsh environments complicates the automated identification process. Therefore, a low-complexity automated framework for precise pest recognition in the field and performance improvement both in terms of classification robustness and computational complexity is still required. In this work, we presented a cost-effective DL-based model for pest recognition and categorization using drone. The presented framework is based on custom CornerNet model with the DenseNet-100 serving as a backbone for deep feature extraction from the input samples. Our results show that the proposed technique is capable of effectively localizing and classifying the multiple pest species in the presence of high variation in shape, size, color, position, and variability across and within classes. The main contributions of our work are as follows:

- We proposed a low-complexity AI-based framework for drone systems using a custom CornerNet model by employing DenseNet-100 for automated pest recognition in the field, improving the accuracy of classifying the various pests.
- Introduced computationally efficient approach for precise insect pest detection as CornerNet is a one-stage object detection framework.
- Improved classification accuracy of insect pests due to the ability of the DenseNet to compute deep keypoints and CornerNet model to deal with the over-fitted training data.
- A rigorous quantitative and qualitative comparison of the presented technique was performed using a publically available challenging benchmark dataset, namely IP102, to exhibit the efficacy of our method.

The rest of the paper is structured as follows: “Related work” reviews related work for insect pest recognition, while

"Proposed method" provides a detailed description of the proposed framework. In "Experimental details and results", we provide the details of experiments performed and a discussion on the results. Lastly, "Conclusion" concludes our research.

Related work

Recently, pest localization and classification have attained the consideration of the research community due to immense development in the area of computer vision. Numerous standard datasets are available for this purpose. However, these datasets consist of the minimum number of samples compared to the normal data compulsory for the latest DL-based frameworks. This section conducted a thorough examination of previous work for the automated identification and categorization of pests from crops.

Nanni et al. [32] presented a method to merge CNNs with saliency approaches to recognize and classify the pest from crops automatically. Initially, the saliency method was applied for data augmentation, and then five different CNN models, namely AN, DenseNet201, ShuffleNet (SN), GoogLeNet (GL), and MobileNetv2 (MN), were trained to classify the insects. This approach [32] improves the pest classification accuracy. However, the performance degrades for identifying pest species with significant intra-class differences. In [33], the authors presented a novel CNN framework and compared it with existing DL models, i.e., AN, RN, GL, and VGG. Transfer learning and data augmentation were employed to prevent the network from overfitting and improve classification accuracy.

Similarly, Li et al. [34] proposed an approach for automated recognition and categorization of crop insects. Initially, the adaptive threshold (AT) algorithm was applied over the input sample to convert it to a binary image, on which morphological operations together with the watershed algorithm were used to acquire the region of interest (ROI). Then, the GrabCut technique was utilized to remove the background, and several DL models, namely VGG, GL, and RN, were applied to classify the pest from the input samples. However, these methods [33, 34] exhibit better insect classification accuracy at the expense of a longer computing time. Wang et al. [35] introduced a DL framework for mobile devices, namely DeepPest, to automatically detect and categorize insects. The method [35] employed contextual information as prior information for its training and worked well for the localization of small size insects. However, the approach [35] is inadequate for different mobile devices due to processing limitations. Jiao et al. [36] introduced an anchor-free region CNN (AF-RCNN) for automated localization and categorization of various classes of crop insects. Initially, a keypoints fusion unit was proposed to compute the representative set of features, particularly for small-sized pests. Next, an anchor-free region proposal network (AFRPN) was introduced to calculate practical object proposals based on pest positions by employing fusion feature maps. Lastly, AF-RCNN was trained to identify 24 classes of insects by integrating the AFRPN with Fast R-CNN into a single framework. This method [36] works well for the localization of small insects. However, the performance relies on extensive hyper-parameter choices selected during the training process.

Rodríguez et al. [37] proposed a framework for pest identification. Initially, the RGB sample was transformed into a quaternion matrix, on which Quaternion Gaussian Low-Pass Filter was applied to remove the noise. The processed sample was then subjected to Sangwine's method to obtain the two colored keypoints maps and horizontal and vertical directions. Both maps were converted to the HSV domain to differentiate the monotone horizontal and vertical edge maps. In the following steps, the obtained maps were combined, on which binarization together with the morphological approach was applied to achieve the ROIs. This method [37] is robust to pest detection under chrominance and size variations. However, the generalization performance of this work can be further improved. Nam et al. [38] suggested a DL-based approach to locate and categorize crop insects. The Single-shot detector (SSD) framework was used to calculate the in-depth features from input samples and classify the pests into respective classes. The approach [38] resulted in higher accuracy than previously developed methods. However, it was unable to detect small insects. CNN-based techniques need diverse training samples to show better accuracy. However, datasets lack this aspect. Li et al. [39] proposed a data augmentation-based approach to deal with such challenges. Data augmentation was applied during the training step by rotating input samples to different angles together with a cropping operation. This step assisted in obtaining diverse multi-scale samples that could be utilized to train a multi-scale insect identification framework. Various CNN models were trained to demonstrate the effectiveness of the proposed strategy. This technique [39] detects insects despite significant variations in position. However, it is computationally costly. A two-stage CNN framework was proposed in [40] to locate and categorize crop pests. Initially, the Global activated Feature Pyramid Network (GaFPN) was applied to compute the representative set of features from the input images. The calculated feature vector was passed to the Local activated Region Proposal Network (LaRPN) to identify and classify the pests in the next phase. The method [40] shows better pest classification performance. However, it is prone to overfitting issues, thus resulting in poor performance on unseen data. Another framework for pest detection was proposed in [41]. Initially, the input image was converted to greyscale. In the next step, the processed sample was compared to a reference image to identify the changes saved as the feature vector. In the next step, the Density-Based Spatial Clustering (DBSCAN) was applied over the calculated key points to cluster the pests from samples. This approach [41] can effectively identify the insects from the noisy samples; however, this approach is computationally complex.

Nieuwenhuizen et al. [42] presented an approach to locate and classify the insects from the input sample. In the first step, annotations were developed from the input images, which were employed for transfer learning. In the next step, the annotated images were passed to a DL framework, namely Faster R-CNN, to localize and classify the insects. In the last step, insects were counted manually. The approach [42] improves the insect classification accuracy. However, few results were reported. In [43], the authors conducted a comparative analysis of various CNN-based frameworks, namely VGG, ResNet-50, ResNet-101, AN, and Inception-Net, together with SVM, KNN, and ELM classifiers. The mentioned CNN models were employed to compute the in-depth features, which were later used to train the classifiers to classify the pest from crops. It is concluded in [43] that in-depth features with SVM and ELM classifiers exhibit better classification accuracy. Liu et al. [44] proposed a DL-based model, namely PestNet, for classifying crop pests. In the first step, a module, namely Channel-Spatial Attention (CSA), was introduced to be replaced with CNN for key-points computation. Then, Region Proposal Network (RPN) was employed for calculating region proposals to locate the positions of insects based on extracted keypoints maps from the samples. Finally, a Position-Sensitive Score Map (PSSM) was applied to show the located pest together with their computed class. This approach [44] works well for multi-class classification of crop pests, however, at the expense of increased computational complexity. Another automated pest detection framework was introduced in [45]. After performing image preprocessing, EM and KMM were applied to obtain the ROIs. In the following steps, the GLCM matrix was employed to compute the image features from computed ROIs. The obtained feature vector was used to train the SVM to classify the insects. The method [45] is robust to pest detection. However, it requires a substantial amount of time for data preparation and training. Rustia et al. [46] proposed an approach to localize and classify crop insects. After performing preprocessing, YOLO-V3 has been employed to calculate the deep features and classify the pests from the input samples. This technique [46] shows better insect detection accuracy. However, it is unable to locate pests under intense chrominance and light variations. Another CNN model, namely AN, was utilized in [47] to identify and categorize the insect from images. The technique [47] exhibits better recognition accuracy. However, performance decreases when multiple pest species are present. Xia et al. [48] presented a technique to identify and classify insects. Initially, a data augmentation step was applied to improve the diversity of data. Then the samples were used to train the VGG-19 framework to localize and categorize the pests. This approach [48] works well for insect classification, however, performance is evaluated on limited insect species data. In a very recent work [73], the Custom CenterNet framework with DenseNet-77 method was presented to automate the plant diseases detection and categorization efficiently. The model outperformed the latest plant disease approaches and were able to efficiently locate and classify 38 types of crop diseases from the PlantVillage dataset.

Table 1 presents the analysis of existing techniques employed for pest detection and classification along with their limitations. From Table 1, it can be seen that although the research community has presented extensive work in the field of automated pest categorization, however, there is still a need for performance improvement.

Proposed method

In this section, we have discussed the framework presented for the automated identification and categorization of several crop pests in the field. The aim of this work is to propose a technique that is computationally efficient and capable of automatically extracting reliable image features without the need for any manual inspection. The proposed approach follows two phases: in the first step, a set of samples comprising images from a standard dataset, namely IPI02, is used to prepare the annotations, which are later used for the model training. In the test phase, suspected samples are passed to the trained framework to evaluate the model's performance. More specifically, we have proposed an improved CornerNet model [49] by employing the DenseNet-100 as its base network. Initially, the deep features from the input images are calculated by using the DesneNet-100 framework, which is later used by CornerNet to locate the presence of pests on plant leaves and determine the corresponding category. In the last step, performance is evaluated by employing several standard metrics used in the field of object detection. Figure 1 shows the structural description of the introduced pest detection and classification methodology.

Annotations

For practical DL-based model training, it is required to specify the RoI precisely. To accomplish this, we have utilized the LabelImg tool [50] to develop the annotations. A few annotated images are shown in Fig. 2. The annotations specify the information regarding the position and class of each pest, which is saved in an XML file. After this, the final training file is generated from this XML file to train the model.

Table 1 Comparative analysis of existing pest detection techniques

Reference	Technique	Limitation
[32]	A combination of CNN framework with CNNs with saliency approaches was introduced to classify the crop pests	Its accuracy degraded for cases with huge intra-class variations
[33]	A DL-based framework was employed for pest classification	This approach is computationally inefficient
[34]	GrabCut technique along with several DL frameworks named VGG, GL, and RN, was used to categorize the pest from the input samples	This approach has considerable computational complexity
[35]	A DL-based model DeepPest for mobile devices, was introduced to locate and classify the pests	It is inadequate for different mobile devices due to processing limitations
[36]	A CNN-based method, namely AF-RCNN, was presented for the automated classification of crop pests	The classification accuracy degrades for the samples with intense light changes
[37]	After preprocessing step, the binarization approach, together with the morphological operation, was applied to achieve the ROIs	The generalization performance requires further improvement
[38]	A DL-based approach named SSD was utilized to calculate the deep key points and categorize the pest from the input	It shows low performance for the localization of small-sized pests
[39]	After performing the data augmentation step, several CNN frameworks were evaluated to show the robustness of this method	This approach is computationally expensive
[40]	The CNN framework, namely GaFPN, together with LaRPN, was employed to identify and classify the pests	It exhibits poor detection accuracy for real-world scenarios
[41]	The computed image features along with the DBSCAN clustering technique were used to locate the pests	This method is unable to deal with the insect's rotational variations
[42]	A DL-based approach Faster-RCNN was employed to locate and categorize the pests	This approach only reported the initial results
[44]	PestNet, a DL-based methodology for categorizing crop pests, was introduced	This approach has considerable computational requirements
[45]	The GLCM matrix and the SVM classifier were utilized for pest categorization	This framework does not generalize well to multiple insect presence on images
[46]	A DL-based approach, namely YOLO-V3, was applied to detect and classify crop pests	This method is unable to locate pests under intense chrominance and light variations
[47]	Another CNN model, namely AN, utilized pest classification	This method is incapable of dealing with the complexities of real-world scenarios
[48]	A CNN framework named VGG-19 was employed to classify the pests into various categories	It suffers from overfitting due to limited training data

CornerNet model

The CornerNet [49] is a one-step detector that locates the RoIs using keypoint estimation. It predicts corners, i.e., the Top-Left (TL) and Bottom-Right (BR), to compute bounding boxes (bbox) that are more accurate and efficient as compared to other anchor-based techniques [29, 51]. The overall architecture of the CornerNet model comprises two major components; the backbone network and the prediction head (Fig. 1). Initially, the model uses a backbone feature extraction network to compute a set of related keypoint maps that

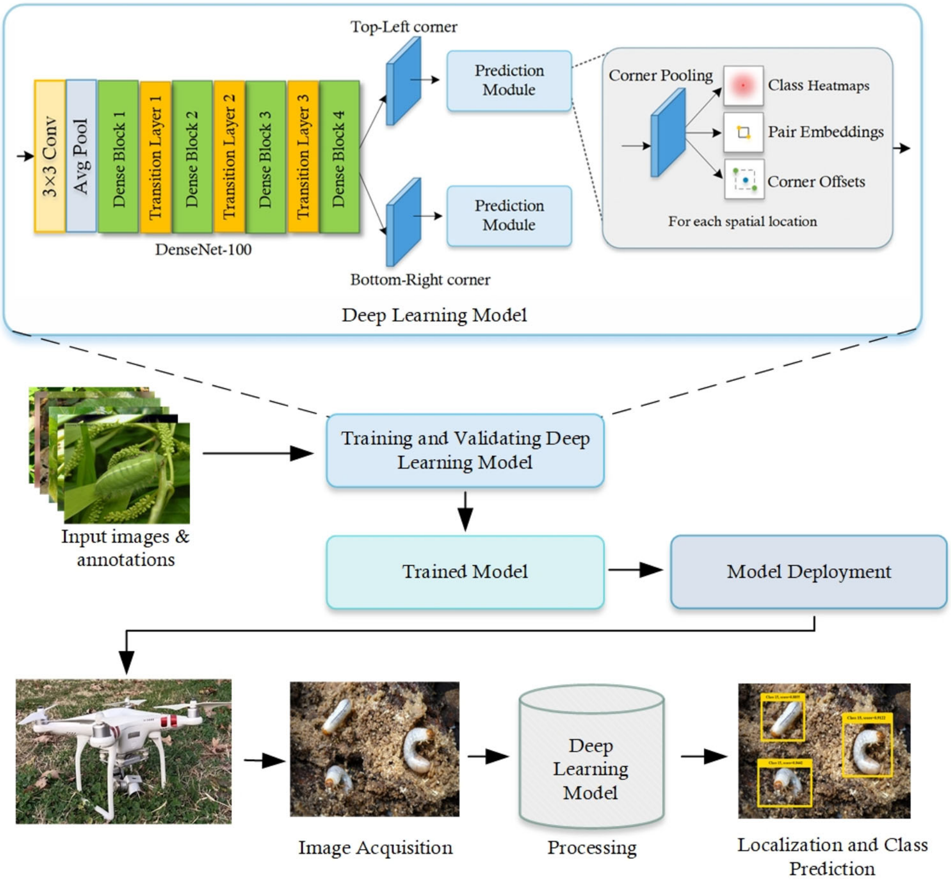

Fig. 1 Visual representation of the introduced approach for pest recognition

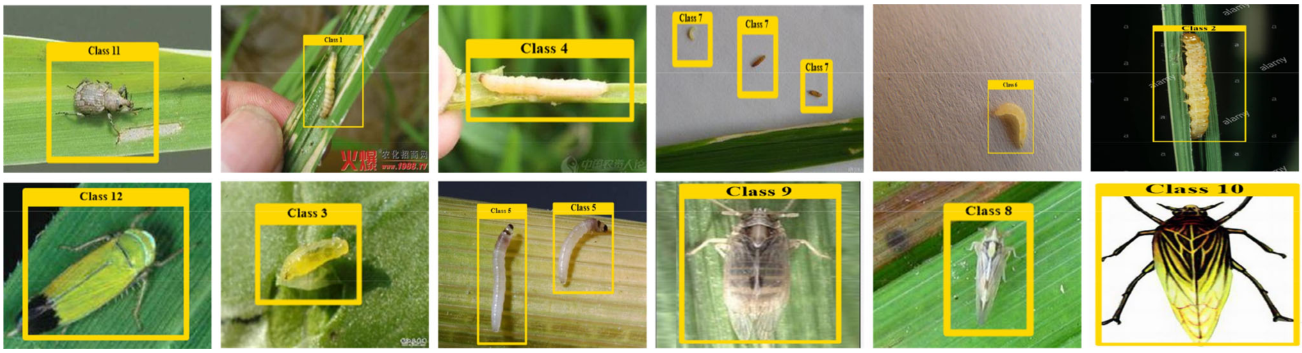

Fig. 2 Sample annotated images of the IP102 dataset

are used to predict heatmaps (HMs), embeddings, offset, and class (*C*). The HMs provide the probability to determine if a particular position is a TL/BR corner belonging to a specific class. At the same time, the main objective of the embeddings is to distinguish the corner pairs and offsets for adjusting the position. The highest scored TL, and BR points are used to determine the exact location of the bbox, and class is determined by utilizing the embedding distances on more relevant feature pairs. The CornerNet model tends to outperform the existing object detection frameworks [28–31]. However, the recognition of insect pests has its unique properties, i.e., small size and a similar visual appearance to their surroundings that differentiates it from the existing object recognition and classification methods. In this work, we have customized the CornerNet model for the detection and classification of multiple pest species. We improved the backbone network of the CornerNet model to increase model effectiveness and achieve more accurate results for pest identification. The improved backbone computes high-level discriminative information that improves the pest localization accuracy and overall classification performance. Moreover, the improved architecture is lightweight and computationally efficient as compared to the original CornerNet model.

The motivation for employing the CornerNet model for the identification of pests is its ability to effectively identify objects by using keypoint estimation as compared to previous models [29, 51–54]. The model uses precise features and locates the object employing features; thus, it removes the requirement of utilizing extensive anchor boxes for different target dimensions as compared to other one-stage object detection approaches like SSD [52] and YOLO (v2, v3) [53]. While in comparison to two-stage approaches like R-CNN [54], Fast R-CNN [29], and Faster R-CNN [51]), the proposed approach is computationally efficient as these methods use two steps to perform the object detection and classification task. Thus, the proposed DenseNet-100-based CornerNet approach better tackles the problems of existing techniques by proposing a more robust framework that computes more reliable image features and minimizes the estimation cost as well due to its one-stage detection.

Custom CornerNet model

A backbone network extracts visual features which provide a semantic and robust representation of an image. The pest is a smaller target. Therefore more precise and discriminative features are required to distinguish them from the complicated surroundings, such as varying acquisition angles, brightness, luminosity conditions, and blurring. The traditional CornerNet model was presented with the Hourglass104 feature extractor [49]. The limitation of the Hourglass network is that it is computationally expensive, i.e., involves extensive network parameters and space requirements which unavoidably slows the detection process and reduces the overall efficiency of the model. Moreover, the accuracy of the feature extractor impacts the detection accuracy [55]. We have customized the backbone network for localization and classification of pests to improve framework robustness and achieve better performance. We adopt DenseNet-100 [56] as the backbone network for improved feature extraction and to reduce computational complexity.

DenseNet-100 feature extractor

The DenseNet-100 contains four densely connected blocks with 100 layers and is shallower than Hourglass104. The basic architecture of employed DenseNet-100 is presented in Fig. 1. The DenseNet-100 framework contains fewer parameters (7.08 M) than the Hourglass104 network (187 M), giving it a computational benefit. In DenseNets, all layers are directly connected to one another, and the keypoint maps from earlier layers are passed to the subsequent layers. The DenseNet architecture promotes feature reuse and enhances the information flow throughout the network, which makes it appropriate to tackle complex transformations efficiently for pest localization [56]. The structural details of DenseNet-100 are elaborated in Table 2.

Table 2 Structure of DenseNet-100

Layer_name	Filter_size	Stride
CnL	3 × 3 con	2
Pooling	2 × 2 avg pool	2
DnB (1)	$\begin{pmatrix}1\times1\text{ con}\\3\times3\text{ con}\end{pmatrix}\times 12$	1
TrL (1)		
ConL	1 × 1 con	1
Pooling layer	2 × 2 avg pool	2
DnB (2)	$\begin{pmatrix}1\times1\text{ con}\\3\times3\text{ con}\end{pmatrix}\times 12$	1
TrL (2)		
ConL	1 × 1 conv	1
Pooling layer	2 × 2 avg pool	2
DnB (3)	$\begin{pmatrix}1\times1\text{ con}\\3\times3\text{ con}\end{pmatrix}\times 12$	1
TrL (4)		
ConL	1 × 1 conv	1
Pooling layer	2 × 2 avg pool	2
DnB (4)	$\begin{pmatrix}1\times1\text{ con}\\3\times3\text{ con}\end{pmatrix}\times 12$	1
Prediction layer	7 × 7 avg pool FC layer	

The DenseNet comprises several Convolutional Layer (ConL), Dense Block (DnB), and Transition Layer (TrL). Figure 3 presents the DnB structure, which is a main component of the DenseNet framework. In Fig. 3, z_0 represents the input layer with f_0 feature maps. Moreover, $H_n(.)$ is a compound method comprising three successive operations: a 3 × 3 ConL filter, Batch Normalization (BtN), and ReLU. Every $H_n(.)$ operation generates f keypoint maps that are passed to z_n subsequent layers. As every layer takes all previous layer keypoints maps data as input, this produces $f \times (n-1) + f_0$ feature maps at the nth layer of DnB, which causes the dimension of the keypoint map to increase dramatically. Therefore, the TnL layers are introduced among the DnB to minimize the keypoints map dimension. The TnL contains a BtN and 1 × 1 ConL along with the average pooling layer, as demonstrated in Fig. 3.

Prediction module

The feature extraction network is followed by two distinct output branches, which represent the TL corner and the BR

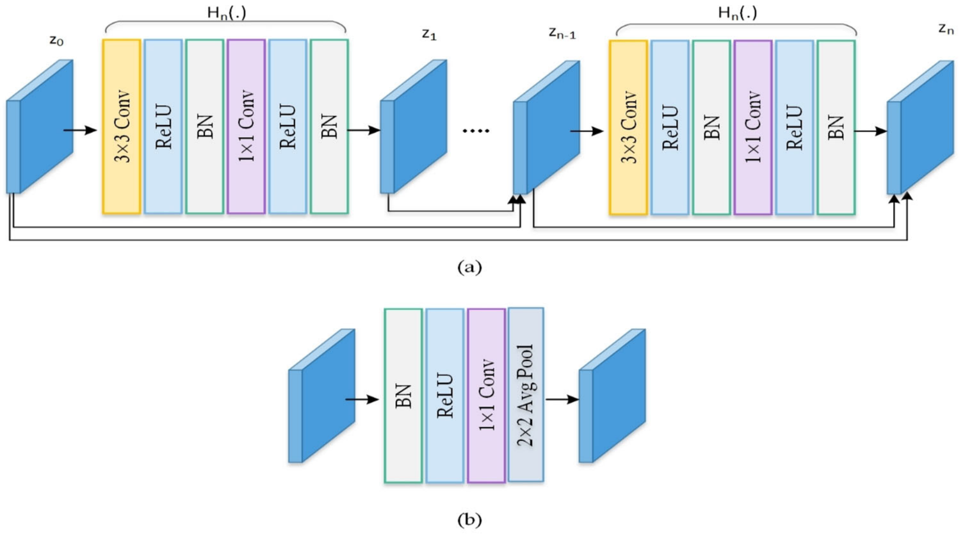

Fig. 3 The architecture of **a** dense block and **b** transition block

corner prediction branch. Each branch module consists of a corner pooling layer placed on the top of the backbone to pool features and generate three outputs: HMs, embeddings, and offsets. The prediction module is a modified residual block comprised of two 3 × 3 ConL and one 1 × 1 residual network that is followed by a corner pooling layer. The corner pooling layer helps the network to localize the corners better. The pooled features are passed to a 3 × 3 ConL-BN layer, and reverse projection is added. This modified residual block is then followed by a 3 × 3 ConL, which generates HMs, embeddings, and offsets. The HMs are used to estimate the location of corner points. The offsets are employed to correct the corner location because a quantization error occurs when mapping from keypoints in the input image to the feature map is performed. There may exist multiple pests in an image. The embeddings are used to determine whether the corner point is a group, i.e., the TL and BR corner belong to the same or different pest.

Detection

To obtain the final bbox from the corner predictions, non-maximal suppression (NMS) on the corner HM via 3 × 3 max-pooling layer is applied. The top 100 TL corners and BR corners over all classes are extracted from the HMs. The predicted offsets are used to adjust the corner locations. The TL corner and BR corners per class are paired with the most similar embedding, and the pairs with an $L1$ distance greater than 0.5 are eliminated. For the obtained candidate bbox, we applied soft-NMS to remove strongly overlapping bbox. The average scores of the TL and BR corners are used as the detection scores.

Loss function

CornerNet is an end-to-end learning methodology that uses multi-task loss to improve its performance and precisely locate pests. The training loss function L is the summation of four different losses, defined as:

$$L = L_{\text{det}} + \alpha L_{\text{pull}} + \beta L_{\text{push}} + \gamma L_{\text{off}}, \tag{1}$$

where L_{det} is the detection loss responsible for corner detection and is a variant of a focal loss, L_{pull} is the grouping loss which is responsible for grouping corners of the same bbox, L_{push} is corner separation loss responsible for separating corners of the different bbox, L_{off} is the smooth, and $L1$ loss is responsible for the offset correction. The parameters α, β, and γ represent the weights for the pull, push and offset loss and are set as $\alpha = \beta = 0.1$ and $\gamma = 1$. The L_{det} is defined as:

$$L_{\text{det}} = \frac{-1}{M} \sum_{i=1}^{C} \sum_{x=1}^{H} \sum_{y=1}^{W} \begin{cases} (1-T)^{\varphi} \log(T) & \text{if}(G) = 1 \\ (1-G)^{\omega} (T)^{\varphi} \log(1-T) & \text{otherwise} \end{cases}. \tag{2}$$

Here, M is the number of pests in an image. C, H, and W denote the number of channels, width, and height, respectively, from the input. T and G represent T_{ixy} and G_{ixy}, where T_{ixy} is the predicted score at the position (x, y) for pest of class i in the input image, and G_{ixy} is the respective ground-truth value. The φ and ω are the hyperparameters that control the contribution of each point and are set as 2 and 4, respectively.

During downsampling, the output size is decreased compared to the original input image. Therefore, the location (a, b) of a pest in the input image is mapped to the location $\left(\frac{a}{n}, \frac{b}{n}\right)$ in the HMs, where n is the factor to which downsampling is performed. During remapping of locations from the HM to the original size input image, it results in a precision loss that affects the quality of IoU for smaller bbox. To resolve this problem, the position offsets are calculated to adjust the corner locations and are given by:

$$O_k = \left(\frac{a^k}{n} - \left\lfloor \frac{a^k}{n} \right\rfloor, \frac{b^k}{n} - \left\lfloor \frac{b^k}{n} \right\rfloor \right), \tag{3}$$

where O_k denotes computed offset, a_k and b_k are the coordinators of a and b for corner k. For training purposes, to compute L_{off}, the smooth $L1$ function is used to adjust the corner locations slightly and is defined as:

$$L_{\text{off}} = \frac{1}{M} \sum_{k=1}^{M} \text{Smooth } L1 \text{ Loss}(O_k,\ O_k'). \tag{4}$$

An input image may contain multiple pests; thus, multiple BR and TL corners are computed in a single image. For each detected corner, the network predicts an embedding vector used to decide whether a pair of TL and BR corner belongs to the same pest. We apply the "pull and "push" losses to train the network and are defined as:

$$L_{\text{pull}} = \frac{1}{M} \sum_{i=1}^{M} [(e_{li} - e_i)^2 + (e_r - e_i)^2], \tag{5}$$

$$L_{\text{push}} = \frac{1}{M(M-1)} \sum_{i=1}^{M} \sum_{\substack{j=1 \\ j \neq i}}^{M} \max[0,\ \Delta - |e_i - e_j|], \tag{6}$$

where e_{l_i} represents the TL corner, e_{r_i} is the BR corner for pest i, and e_i is the average value of e_{l_i} and e_{r_i}. The maximum distance for two corners belonging to different pests is set as 1; that is, $\Delta = 1$ used in all our experiments.

Experimental details and results

This section describes the implementation details and the experiments carried out to evaluate the performance of the suggested model. To comprehensively demonstrate the efficacy of the custom CornerNet model, we have evaluated pest recognition and classification and compared it with other models.

Dataset

In this work, we have utilized the IP102 insect pest recognition dataset [57] to evaluate the performance of the proposed model. This dataset contains 75,222 images covering 102 common insect pest classes. The IP102 dataset is organized hierarchically, with two super-classes: field crops (FC) and economic crops (EC), further divided into sub-classes depending on particular crop types damaged by pest insects. The FC contains five sub-classes, i.e., Rice, Corn, Wheat, Beet, and Alfalfa, whereas the EC contains three sub-classes, i.e., Citrus, Vitis, and Mango. All these sub-classes are further categorized and contain 102 pest classes that define pest insects associated with the specific crop. The further details of the classes and the number of samples in each class are given in [57]. It is worth noting that the images in the IP102 dataset are diverse, i.e., insects of very different ages, colors, sizes, and shapes. In addition, the variations in luminosity, zoom level, and angle further make the dataset very challenging to deal with the complexities of real-life scenes. Figure 4 presents some sample images of pests from various species from the IP102 dataset. It can be observed from the Fig. 4 that the samples in the dataset are challenging, having intricacies of various environmental factors such as varying lighting conditions or insects hidden in the background.

Implementation details

The overall implementation of the proposed framework is achieved in TensorFlow using the Keras library. Table 3 presents the detail of the final training parameters for the Custom CornerNet model. In our study, we have tuned the model's hyperparameters by varying epochs, batch size, and learning rate to obtain the final optimized model. The model learning rates of 0.01, 0.001, and 0.0001 with Stochastic Gradient Descent (SGD) training optimizer were utilized during the experiment. The mini-batch size and epoch were set at 15, 25, 35, 45, and 16, 32, 64, respectively. To prevent the model overfitting, we set the dropout value to 0.3. The size of the input images was fixed at 224 × 224, and the data were randomly divided into training, validation, and test sets. We used 60% of the data for training, 10% for validation, and the remaining 30% for testing.

Evaluation parameters

For evaluating the performance of the proposed technique, we have used different quantitative metrics such as precision (P), recall (R), accuracy (Acc), Intersection over Union (IoU), and mean average precision (mAP). These metrics are computed as follows:

$$P = \frac{\text{TP}}{(\text{TP} + \text{FP})}, \tag{7}$$

$$R = \frac{\text{TP}}{(\text{TP} + \text{FN})}, \tag{8}$$

$$\text{Acc} = \frac{(\text{TP} + \text{TN})}{(\text{TP} + \text{TN} + \text{FP} + \text{FN})}, \tag{9}$$

$$\text{IoU} = \frac{\text{TP}}{(\text{FN} + \text{FP} + \text{TP})} \times 2, \tag{10}$$

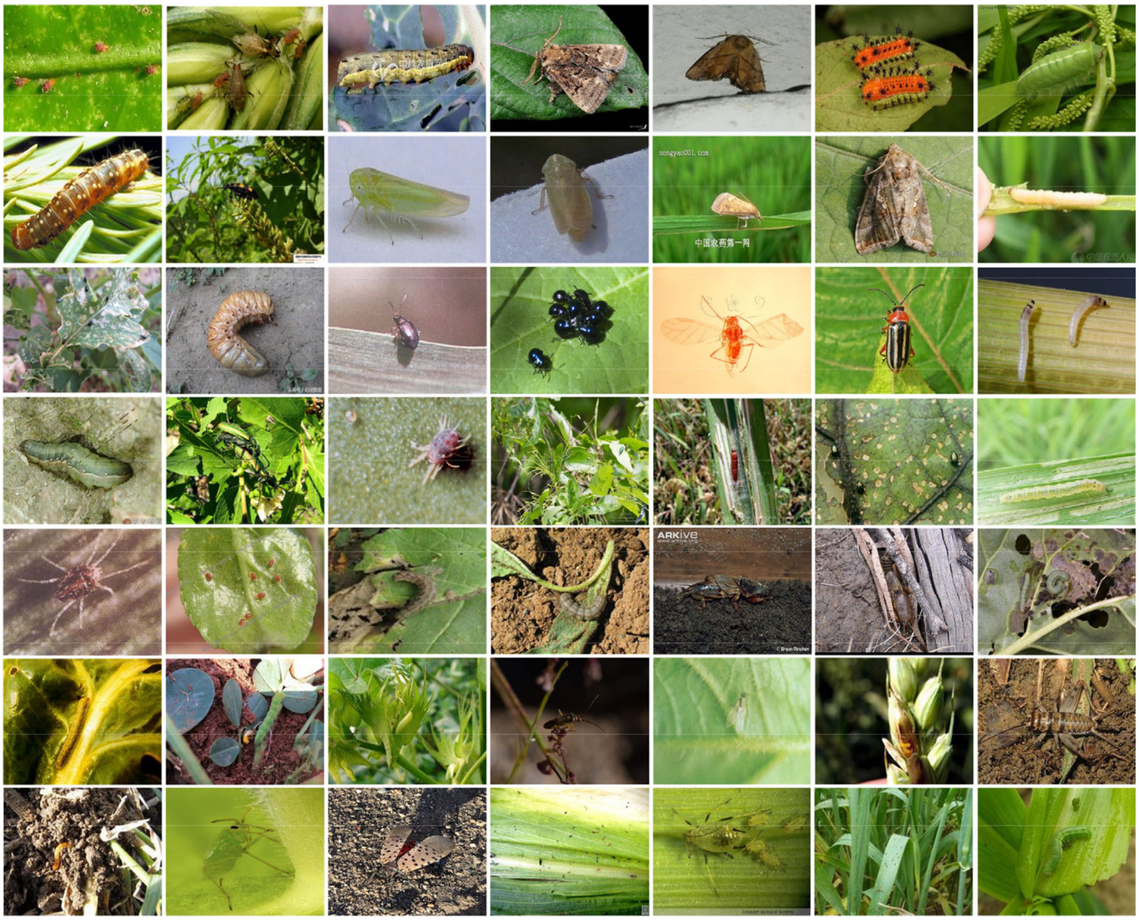

Fig. 4 Sample images from IP102 Dataset

Table 3 Training parameters for the proposed model

Framework parameters	Value
Epochs	35
Learning rate	0.001
Batch size	64
Confidence score threshold	0.5
Unmatched threshold	0.5

$$\text{mAP} = \sum_{i=1}^{T} \frac{\text{AP}(t_i)}{T}, \tag{11}$$

$$F1\_\text{score} = \frac{2 \times P \times R}{(P + R)}. \tag{12}$$

TP, TN, FP, and FN denote the true-positive, true-negative, false-positive, and false-negative cases. If the insect in the picture is correctly classified, it is considered TP; otherwise, it is considered FN. The insect not present in the image is classified as TN if the classification is wrong; otherwise, it is classified as FP. The mAP computation is shown in Eq. (11), where AP is the average precision of each class, t and T represent the test image and the total number of test images, respectively.

Insect pest localization results

The precise localization of pests is important for designing an effective automated pest recognition method. Therefore, we designed an experiment to assess the localization effectiveness of the proposed framework. For analysis, we utilized all the test images from the IP102 database and presented a few visual results, as shown in Fig. 5. From the reported results, we can see that the proposed approach is capable of locating pests of varying sizes, shapes, and colors. Additionally, our technique can effectively detect pests even in complex background, illumination, orientation changes, and varying acquisition angle. The localization ability of the proposed framework by employing keypoint estimation allows it to identify and discriminate pests of various categories effectively and precisely. We have computed the mAP and IOU to quantitatively measure the localization performance. These metrics show how well the proposed model performs

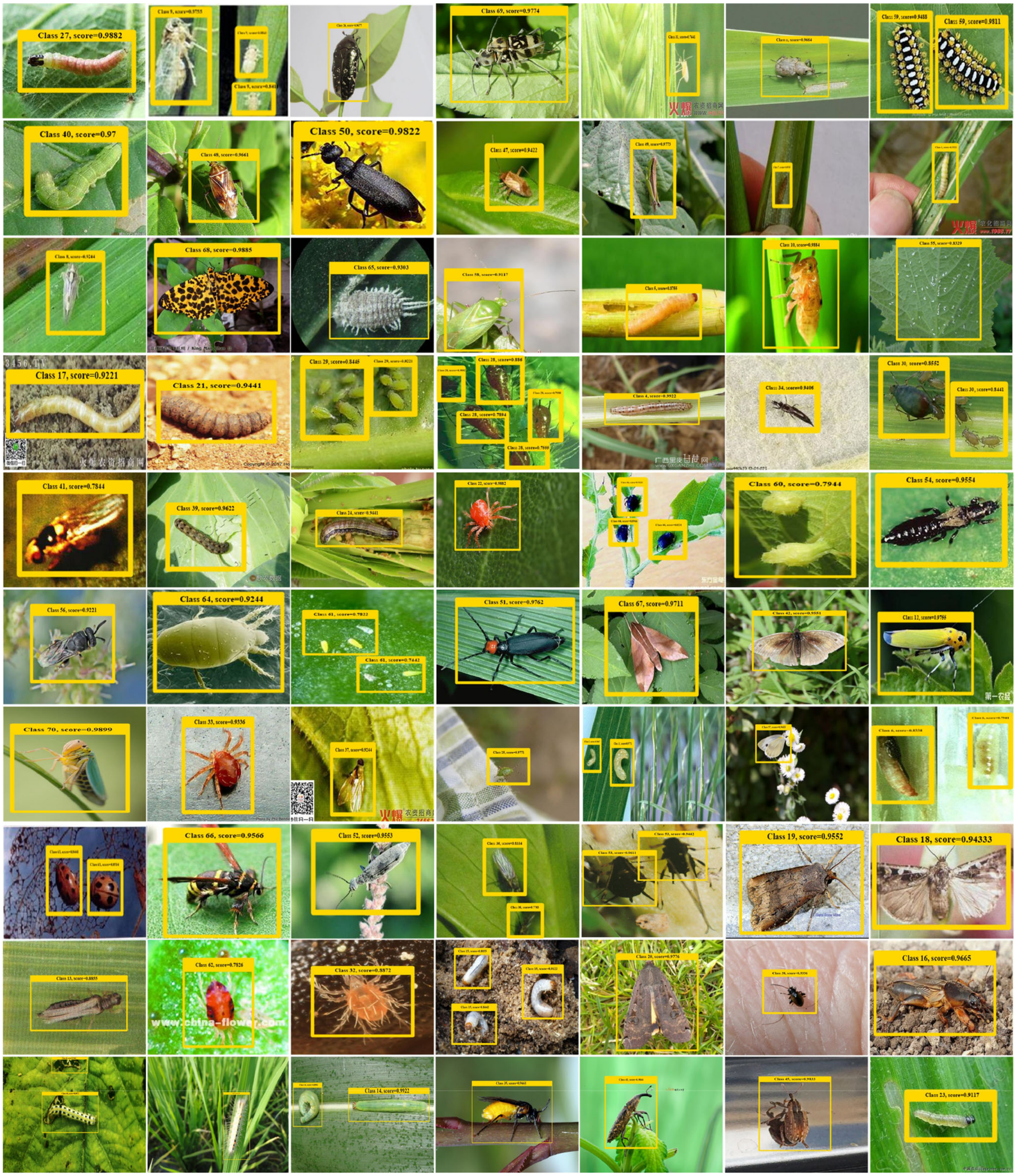

Fig. 5 Sample detection results of insect pests using the proposed model

Table 4 Class-wise crop-based insect classification performance of the proposed method

Crop	Precision (%)	Recall (%)	F1 score (%)
Rice	31.5	34.47	32.9
Corn	55.1	54.4	54.7
Beet	54.87	42.23	47.7
Wheat	57.86	51.55	54.5
Alfalfa	59.37	53.61	56.3
Mango	79.94	75.82	77.8
Citrus	74.87	70.28	72.5
Vitis	80.25	77.31	78.7

in localization and recognition for several pest categories. For localization, the IOU threshold is set to 0.5, which means that the overlap score between the predicted region and ground truth is less than this value is considered background, otherwise considered a pest. The proposed framework achieved the mAP and mean IOU values of 0.578 and 0.621, respectively. We can infer from these results that the presented technique can effectively detect and precisely localize the pests even in a diverse background.

Insect pest classification results

The accurate categorization of various pests is important to demonstrate the robustness of a model. A crop cultivation area is suspected of having multiple types of insects depending on the crop category in the real world. Therefore, we have performed an experiment to measure the efficacy of the proposed technique for classifying insect pests based on eight hierarchical crop categories. The trained CornerNet model is applied to all the test images from the IP102 dataset to accomplish this task. Table 4 shows the crop-based pest categorization performance of the proposed method in terms of recall, precision, and F1 score. It can be observed from the stated results that the presented framework has acquired the precision, recall, and F1 score of 61.72%, 57.46%, and 59.39%, respectively, for all the crops-specific insect classes. The reason for robust pest classification performance is the correctness of the employed keypoints computation technique that represents each pest class in a discriminative and reliable manner. As a result, our custom CornerNet performs well in crop-wise pest identification, demonstrating the effectiveness of the introduced method.

We have also reported the accuracies of eight crop-wise pest classes in a boxplot in Fig. 6. The boxplot indicates the distribution of classification accuracy over different classes. According to Fig. 6, our method attained the average accuracy values of 0.484, 0.707, 0.593, 0.695, 0.497, 0.899, 0.773, and 0.851 for eight crop classes, i.e., rice, corn, beet, wheat, alfalfa, mango, citrus, and vitis, respectively. More specifically, we achieved an average classification accuracy of 0.6874 with a low error rate on all classes that exhibit the efficacy of the proposed method. It can be observed from Fig. 6 that our method has achieved somewhat promising results for crops like mango, vitis, and citrus. However, the proposed framework achieves low classification accuracy on some classes such as rice and alfalfa due to visual similarities with the background and high intra-class variances. In Fig. 7, we have provided some example images from the IP102 dataset having a similar appearance. The sample in the same column represents pests from different species, but their visual characteristics are similar.

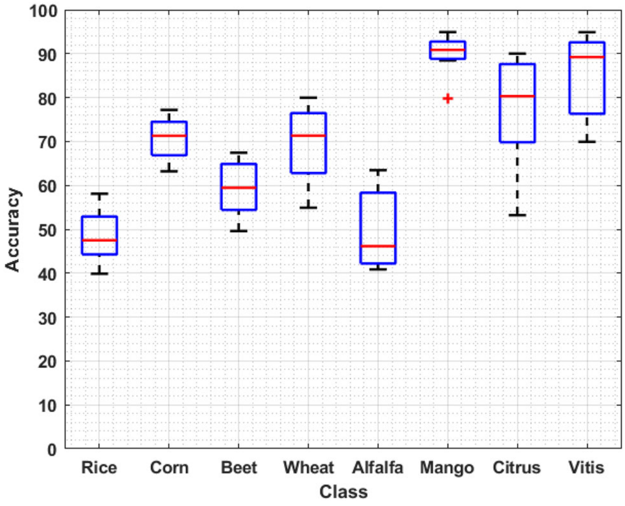

Fig. 6 Accuracy of the proposed method over crop-wise pest classes

In addition, Fig. 8 shows the normalized confusion matrix plot of the presented technique that describes the summarized crop level pest classification results in terms of predicted and actual class. To further demonstrate the recognition performance of the proposed model for each of the 102 pest species, we have presented the obtained accuracy values in Fig. 9. These results validate the robust performance of the proposed approach over crop-wise pest categorizes and 102 pest species.

Evaluation of DenseNet-100 model

For the image recognition task, deep features are effective. We analyzed to evaluate the feature learning ability of the employed DenseNet-100 model compared to other deep feature extraction models for pest identification and classification task. For this reason, the detection performance of the proposed Custom CornerNet is compared with different base models, i.e., Alexnet [58], GoogleNet [59], VGGNet [60], ResNet-50 [61], ResNet-101 [61], Inception V4 [62], HourGlass104 [63], EfficientNet [64], and DenseNet-121 [56].

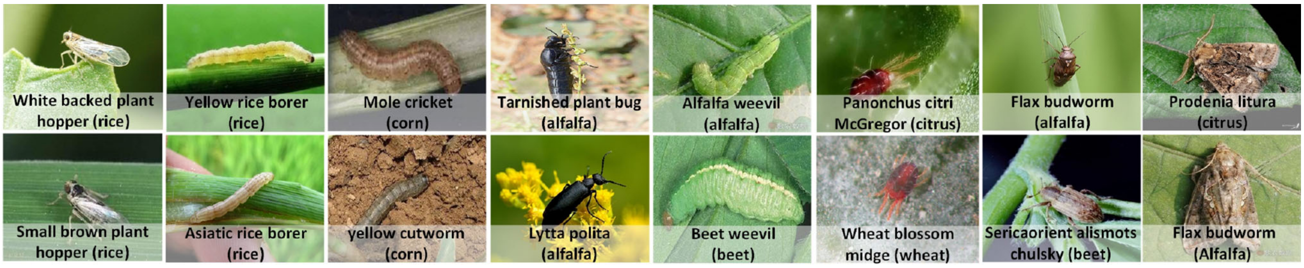
White backed plant hopper (rice)
Yellow rice borer (rice)
Mole cricket (corn)
Tarnished plant bug (alfalfa)
Alfalfa weevil (alfalfa)
Panonchus citri McGregor (citrus)
Flax budworm (alfalfa)
Prodenia litura (citrus)
Small brown plant hopper (rice)
Asiatic rice borer (rice)
yellow cutworm (corn)
Lytta polita (alfalfa)
Beet weevil (beet)
Wheat blossom midge (wheat)
Sericaorient alismots chulsky (beet)
Flax budworm (Alfalfa)

Fig. 7 Sample images of pests species having similar visual features (the label shows the pest-wise class and corresponding crop subclass)

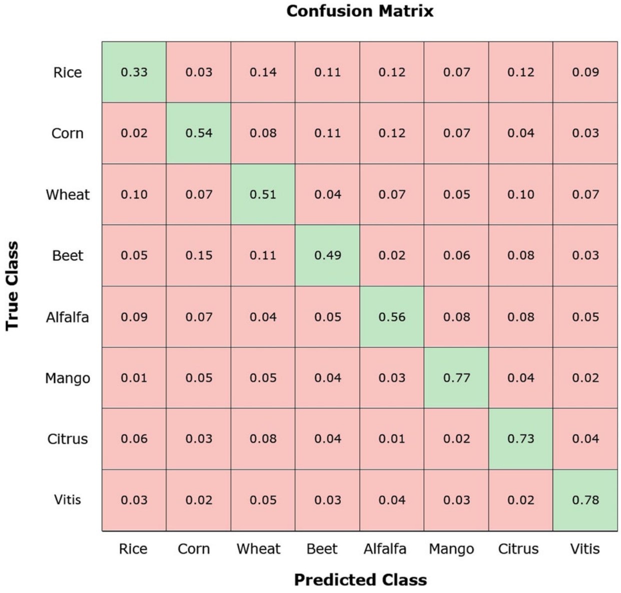

Fig. 8 Confusion matrix of the proposed method over crop-wise pest classes

Table 5 Performance comparison of the proposed approach with other feature extraction models

Method	Parameters (million)	Accuracy ± STD (%)
AlexNet	62.3	41.8 ± 1.51
GoogleNet	7.8	43.5 ± 1.01
VGGNet	138	48.2 ± 2.09
ResNet-50	23.72	57.39 ± 1.35
ResNet-101	42.63	53.18 ± 1.05
Inception V4	41.2	47.8 ± 1.64
HourGlass104	187	54.63 ± 1.44
EfficientNet	19.4	60.2 ± 1.02
DenseNet-121	8.06	54.71 ± 1.45
DenseNet-100	7.08	68.74 ± 0.82

We adopted transfer learning to achieve more accurate generalizing power on the unseen data. All these base networks were pre-trained on ImageNet [65] and then fine-tuned the last layer of the network on the IP102 database. For this experiment, the networks were trained for 30 epochs, and the mini-batch sizes were set to 16 and 64, respectively. In addition, the learning rate was set at 0.001 with the SGD algorithm and a momentum value of 0.9. We have analyzed the acquired classification results of these models over the IP102 database and their computational complexity in terms of network parameters.

The comparative analysis of our approach with other feature extraction models is given in Table 5. The classification accuracies and standard deviation (STD) are presented. The STD shows the consistency of the model's classification output. The higher value of STD shows the inconsistent

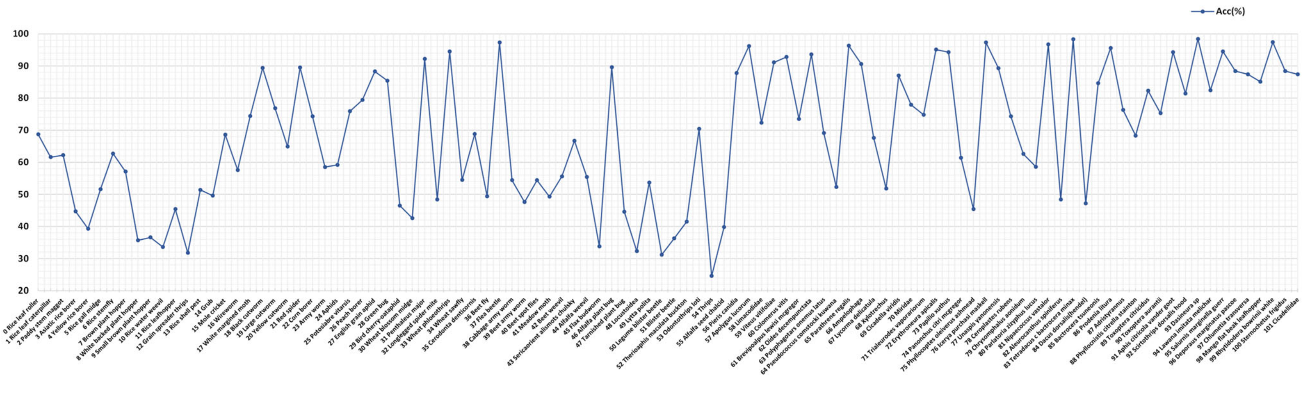

Fig. 9 Accuracy of the proposed method over 102 pest classes

model's behavior for pest recognition and classification results. According to the results, it can be seen that the custom CornerNet with DenseNet-100 as the backbone network performs better as compared to other models. This is due to effective deep feature computation using the DenseNet model, which provides a more accurate and diverse feature representation of different insect pest species. Table 5 shows the base frameworks, i.e., AlexNet, VGGNet, ResNet, Inception v4, and HourGlass, yield low performance for pest recognition. This could be due to their inability to learn fine-level characteristics to distinguish multiple pest species in a complex background, thus resulting in a high misclassification rate. AlexNet attains the lowest accuracy of 41.8% for predicting pests for all 102 categories. The primary reason for the poor performance of the model is that the network is too simple to learn the complexities, i.e., the shape and texture of input pest data.

In comparison, the deeper networks, i.e., ResNet-101, HourGlass, and DenseNet-121, are capable of learning more descriptive and fine-grained differences between many similar insect species. However, their performance is still low for identifying multiple pest classes. This might indicate that due to many network parameters, these models are more prone to overfit on pest classes in the IP102 dataset with fewer training samples. In comparison, the custom CornerNet with DenseNet-100 reached the best performance (68.74% of accuracy) in classifying the various pest species. The EfficientNet model attains the second highest accuracy (60.2%). However, it is computationally more complex. The DenseNet-100, on the other hand, has just 7.08 million parameters, which is fewer than any of the other employed DL models.

The better pest classification performance of our approach is its improved network architecture, which allows the optimal reuse of model parameters. We have used their original implementations for the base models, which are quite complex in structure and unable to extract reliable features. Our approach overcomes the shortcomings of comparative models by incorporating an efficient framework for discriminative keypoints calculation by reusing features from the original layers in each subsequent layer. As a result, it accurately handles complex transformations, resulting in improved performance. From this analysis, we can say that the proposed custom CornerNet with DenseNet-100 backbone performs better than other feature extraction models in terms of accuracy and efficiency.

Performance comparison with ML-based classifiers

To evaluate the performance of the proposed method against the ML-based classifiers with deep features, we performed an experiment to exhibit the classification performance analysis with other ML-based classifiers to demonstrate the efficacy of the proposed technique further. We used the IP102 dataset for this experiment and divided it into 60%, 10%, and 30% for training, validation, and testing sets, respectively. The detailed experimental settings are described in Sect. 4.2. We utilized the deep features extracted from the three highest performing feature extraction models in Table 5. The deep features from ResNet-50 [61], EfficientNet [64], and DenseNet-100 [56] are used to train the ML classifiers, i.e., SVM and KNN, and the classification results with standard deviation are shown in Table 6. From Table 6, it can be observed that using the DenseNet-100-based deep features with SVM and KNN classifiers achieved better results among other combinations. However, our Custom Corner Net model still obtained the best results. More specifically, DenseNet-100 with SVM and KNN as back-end classifiers achieved 52.5% and 50.4%, respectively. Whereas, the proposed Custom CornerNet model achieved an accuracy of 68.74%. This illustrates that the proposed model provides more accurate feature representation of the pests and better deals with overfitted training data than ML-based classifiers.

Table 6 Performance comparison of the proposed approach with ML-based classifiers

Methods	Accuracy ± STD (%)	
	SVM	KNN
ResNet-50	49.5 ± 1.69	49.4 ± 2.28
EfficientNet	51.9 ± 1.07	51.7 ± 1.86
DenseNet-100	52.5 ± 0.97	50.4 ± 1.19
Proposed (Custom CornerNet)	68.74± 0.82	

Performance comparison with other object detection techniques

We have compared the performance of the proposed model with other state-of-the-art object detection methods. An accurate pest localization is important because a noisy background can mislead the classifier when the target pest is not apparent. The existence of several pests can further complicate the detection process. The correct localization can further improve the classification accuracy by ignoring irrelevant background information. To evaluate this, we have considered different two-stage detectors, i.e., Fast R-CNN [29], Faster R-CNN [51] and one-stage object detection models, i.e., SSD [52], YOLOv3 [53], RefineDet [66] and CornerNet [49] which have demonstrated robust performance on the COCO dataset [67]. We have assessed the performance of these models over the IP102 dataset to analyze their pest localization ability under different challenging conditions such as the presence of complex background,

Table 7 Performance comparison of the proposed approach with other object detection methods

Method	Backbone	mAP	Test time (s/img)
Two-stage detectors			
Fast R-CNN	VGG-16	46.12	2
Fast R-CNN	DenseNet-100	47.34	1.73
Faster R-CNN	VGG-16	47.87	0.45
Faster R-CNN	DenseNet-100	48.12	0.41
One-stage detectors			
Refinedet	VGG-16	49.01	0.30
Refinedet	DenseNet-100	50.25	0.28
SSD	VGG-16	47.21	0.36
SSD	DenseNet-100	49.17	0.33
YOLOv3	DarkNet-53	50.64	0.29
YOLOv3	DenseNet-100	50.95	0.27
Proposed (Custom CornerNet)	DenseNet-100	57.23	0.23

noise, luminosity, and variation in color, size, and shape. We have computed the mAP measure to conduct the performance analysis, a standard metric used in object recognition tasks. Furthermore, we have computed test times of all models to assess their computational complexity. Table 7 shows the performance comparison of mAP and inference time of different object detection approaches with varying backbones for pest detection.

Results reported in Table 7 show the superiority in performance of the proposed model for pest identification compared to the other. It can be seen from Table 7 that different object detection models show better performance with the powerful backbone, i.e., DenseNet, for the recognition of pests. The two-stage object detectors, Fast R-CNN and Faster R-CNN, show degraded performance. They are computationally expensive, as these approaches use anchor boxes to identify the potential region of interest and then perform classification and regression to find the corresponding bbox. In comparison, the one-stage networks RefineDet, SSD, and YOLOv3 directly determine the position and category of the object and show better performance. However, as the original implementations of these approaches are evaluated in this work, they cannot perform well in recognizing and locating the pests under intense light variations. Figure 10 presents the visual results of one-stage detection models on the test sample.

Moreover, regarding the computation speed, the one-stage detectors are shown to be fast as compared to two-stage detection models. Our model efficiently overcomes the limitation of these methods using a custom CornerNet model with DenseNet-100 as the backbone network. The reason for improved performance is that the DenseNet backbone enables the CornerNet to learn more representative features, which assist in better pest localization and classification into different categories. Furthermore, the CornerNet model provides a computational benefit over other models due to its one-stage detection nature and takes only 0.23 s to process a sample.

Performance comparison with existing approaches

In this section, we present the comparison of the classification performance of our approach with results obtained by previous works [32, 68–72] over the same dataset, i.e., IP102 [57]. Table 8 compares pest insect classification results with existing approaches in terms of average accuracy.

In [68], the authors employed transfer learning to train the deep-learning models (i.e., VGG-19, inceptionNetV3, and ResNet-50) for the classification of pest species and achieved the highest overall average accuracy 57.08% using inceptionNetV3. However, manually cropping and data augmentation techniques were applied before training the model. Ayan et al. [69] employed CCNs (Inception-V3, Xception, and MobileNet) with ensemble methodology, namely GAEnsemble, to improve the classification performance. Similarly, in [32], the authors combined CNN and the saliency method to create an ensemble of classifiers and used the fusion-sum method at the output layer. However, these methods [32, 69] achieved an accuracy of 61.93% and 67.13%, respectively, at the expense of slow computing speed because of

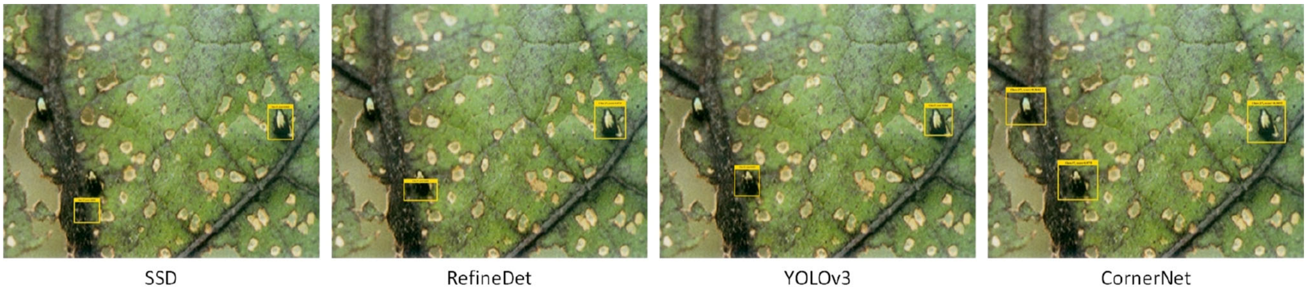

Fig. 10 Sample visual result of SSD, RefineDet, YOLOv3, and the proposed CornerNet model

Table 8 Performance comparison of the proposed method with existing techniques

Reference	Method	Accuracy (%)
Reza et al. [68]	InceptionNetV3	57.08
Ayan et al. [69]	GAEnsemble	67.13
Zhou et al. [70]	EquisiteNet	52.32
Ren et al. [71]	FR-ResNet-50	55.24
Liu et al. [72]	DFF-ResNet-82	55.43
Nanni et al. [32]	FusionSum-Densenet201	61.93
Proposed	CornerNet-DenseNet-100	68.74

ensemble weights calculation. Zhou et al. [70] used the EquisiteNet model comprising double fusion with squeeze-and-excitation and max-feature expansion blocks. The model achieved an accuracy of 52.32%. However, the obtained accuracy is much lower for practical use in the real world. These methods [71, 72] used the modified Resnet block by incorporating feature reuse and feature fusion mechanism for efficient feature computation and obtained an accuracy of 55.24% and 55.43%. However, the ResNet-based architecture is computationally more expensive as compared to DenseNet. These results clearly show that the proposed CornerNet model with DenseNet-100 outperforms the other studies by achieving an average accuracy of 68.74%. In particular, the reason for improved performance is that the DenseNet effectively computes the feature maps by connecting the output from preceding layers as input to all the subsequent layers. The computed features are used by the CornerNet architecture for localization and classification of the pests. Thus, strongly enhances the performance of the proposed model for the task of pest recognition and classification over the challenging dataset IP102. Moreover, our approach is computationally efficient and robust enough to identify insects more precisely in comparison to existing approaches. As a result, we can conclude that our technique has a lot of potential for classifying target pests in the field using drones.

Conclusion

In our work, we have presented a low-cost DL-based framework for the automated recognition and categorization of crop pests in the field using drones. The presented method is based on a custom CornerNet model that employs DenseNet architecture as a backbone network for feature extraction. More precisely, we employed the DenseNet-100 network to extract a discriminative set of keypoints from the input samples. The custom CornerNet model is then trained to recognize various types of pests. We evaluated our approach on the IP102 dataset, a large-scale challenging pest recognition benchmark database comprising in-field captured images. Through extensive experimentation, we have shown the efficacy of our approach for real-world pest monitoring applications. The reported results showed that our method could accurately localize and classify pests of various categories in the presence of complex background and variations in pest shape, color, size, orientation, and luminosity. In the future, we intend to develop a more effective feature fusion approach to improve the performance of our method for fine-grained pest categorization.

References

1. Bruinsma J (2009) The resource outlook to 2050: by how much do land, water and crop yields need to increase by 2050. In: Expert meeting on how to feed the world in, vol 2050, pp 24–26
2. Neethirajan S (2020) The role of sensors, big data and machine learning in modern animal farming. Sens Biosens Res 29:100367
3. Heikkilä M, Pietikäinen M, Schmid C (2009) Description of interest regions with local binary patterns. Pattern Recogn 42(3):425–436
4. Liao W-H (2010) Region description using extended local ternary patterns. In: 2010 20th international conference on pattern recognition. IEEE, pp 1003–1006
5. Ng PC, Henikoff S (2003) SIFT: predicting amino acid changes that affect protein function. Nucleic Acids Res 31(13):3812–3814
6. Bay H, Tuytelaars T, Van Gool L (2006) Surf: speeded up robust features. In: European conference on computer vision. Springer, pp 404–417
7. Joachims T (1998) Making large-scale SVM learning practical. Technical report
8. Krishna K, Murty MN (1999) Genetic K-means algorithm. IEEE Trans Syst Man Cybern Part B 29(3):433–439
9. Aziz S, Bashir M, Mughal O, Khan MU, Khan A (2019) Image patten classification for plant disease identification using local tri-directional features. In: 2019 IEEE 10th annual information technology, electronics and mobile communication conference (IEMCON), 2019. IEEE, pp 0973–0978
10. Aurangzeb K, Akmal F, Khan MA, Sharif M, Javed MY (2020) Advanced machine learning algorithm based system for crops leaf diseases recognition. In: 2020 6th conference on data science and machine learning applications (CDMA). IEEE, pp 146–151
11. Phung VH, Rhee EJ (2019) A high-accuracy model average ensemble of convolutional neural networks for classification of cloud image patches on small datasets. Appl Sci 9(21):4500

12. Batool A, Hyder SB, Rahim A, Waheed N, Asghar MA (2020) Classification and identification of tomato leaf disease using deep neural network. In: 2020 international conference on engineering and emerging technologies (ICEET). IEEE, pp 1–6
13. Tetila EC et al (2020) Detection and classification of soybean pests using deep learning with UAV images. Comput Electron Agric 179:105836
14. Iniyan S, Jebakumar R, Mangalraj P, Mohit M, Nanda A (2020) Plant disease identification and detection using support vector machines and artificial neural networks. In: Artificial intelligence and evolutionary computations in engineering systems. Springer, pp 15–27
15. Nilsback M-E, Zisserman A (2008) Automated flower classification over a large number of classes. In: 2008 Sixth Indian conference on computer vision, graphics & image processing. IEEE, pp 722–729
16. Solis-Sánchez LO et al (2011) Scale invariant feature approach for insect monitoring. Comput Electron Agric 75(1):92–99
17. Roska T, Chua LO (1993) The CNN universal machine: an analogic array computer. IEEE Trans Circuits Syst II Anal Digit Signal Process 40(3):163–173
18. Medsker LR, Jain L (2001) Recurrent neural networks. Des Appl 5:64–67
19. Hinton GE (2009) Deep belief networks. Scholarpedia 4(5):5947
20. Kamilaris A, Prenafeta-Boldú FX (2018) Deep learning in agriculture: a survey. Comput Electron Agric 147:70–90
21. Liu J, Wang X (2021) Plant diseases and pests detection based on deep learning: a review. Plant Methods 17(1):1–18
22. Bulat A, Tzimiropoulos G (2018) Super-fan: Integrated facial landmark localization and super-resolution of real-world low resolution faces in arbitrary poses with gans. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 109–117
23. Zhou P, Ni B, Geng C, Hu J, Xu Y (2018) Scale-transferrable object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp 528–537
24. Sermanet P, Frome A, Real E (2014) Attention for fine-grained categorization. arXiv preprint
25. Kataoka H, Iwata K, Satoh Y (2015) Feature evaluation of deep convolutional neural networks for object recognition and detection. arXiv preprint 07627
26. Wang L, Guo S, Huang W, Qiao Y (2015) Places205-vggnet models for scene recognition. arXiv preprint 01667
27. Targ S, Almeida D, Lyman K (2016) Resnet in resnet: Generalizing residual architectures. arXiv preprint 08029
28. Raj A, Namboodiri VP, Tuytelaars T (2015) Subspace alignment based domain adaptation for rcnn detector. arXiv preprint 05578
29. Girshick R (2015) Fast r-cnn. In: Proceedings of the IEEE international conference on computer vision. pp 1440–1448
30. Zhao X, Li W, Zhang Y, Gulliver TA, Chang S, Feng Z (2016) A faster RCNN-based pedestrian detection system. In: 2016 IEEE 84th Vehicular Technology Conference (VTC-Fall). IEEE, pp 1–5
31. Redmon J, Divvala S, Girshick R, Farhadi A (2016) You only look once: unified, real-time object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 779–788
32. Nanni L, Maguolo G, Pancino F (2020) Insect pest image detection and recognition based on bio-inspired methods. Eco Inform 57:101089
33. Thenmozhi K, Reddy US (2019) Crop pest classification based on deep convolutional neural network and transfer learning. Comput Electron Agric 164:104906
34. Li Y, Wang H, Dang LM, Sadeghi-Niaraki A, Moon H (2020) Crop pest recognition in natural scenes using convolutional neural networks. Comput Electron Agric 169:105174
35. 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. Comput Electron Agric 169:105222
36. 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. Comput Electron Agric 174:105522
37. Rodríguez LAR, Castañeda-Miranda CL, Lució MM, Solís-Sánchez LO, Castañeda-Miranda R (2020) Quarternion color image processing as an alternative to classical grayscale conversion approaches for pest detection using yellow sticky traps. Math Comput Simul 182:646–660
38. Nam NT, Hung PD (2018) Pest detection on traps using deep convolutional neural networks. In: Proceedings of the 2018 international conference on control and computer vision, pp 33–38
39. Li R et al (2019) An effective data augmentation strategy for CNN-based pest localization and recognition in the field. IEEE Access 7:160274–160283
40. Liu L et al (2020) Deep learning based automatic multiclass wild pest monitoring approach using hybrid global and local activated features. IEEE Trans Ind Inform 17:7589–7598
41. Miranda JL, Gerardo BD, Tanguilig BT III (2014) Pest detection and extraction using image processing techniques. Int J Comput Commun Eng 3(3):189
42. Ard N, Hemming HSJ (2018) Detection and classification of insects on stick-traps in a tomato crop using faster R-cnn. In: Proceedings of the Netherlands conference on computer vision, pp 1–4
43. Türkoğlu M, Hanbay D (2019) Plant disease and pest detection using deep learning-based features. Turk J Electr Eng Comput Sci 27(3):1636–1651
44. Liu L et al (2019) PestNet: An end-to-end deep learning approach for large-scale multiclass pest detection and classification. IEEE Access 7:45301–45312
45. Chodey MD, Tamkeen H (2019) Crop pest detection and classification by K-means and EM clustering. Methodology 6:09
46. Rustia DJA et al (2020) Automatic greenhouse insect pest detection and recognition based on a cascaded deep learning classification method. J Appl Entomol 145:206–222
47. Dawei W, Limiao D, Jiangong N, Jiyue G, Hongfei Z, Zhongzhi H (2019) Recognition pest by image-based transfer learning. J Sci Food Agric 99(10):4524–4531
48. Xia D, Chen P, Wang B, Zhang J, Xie C (2018) Insect detection and classification based on an improved convolutional neural network. Sensors 18(12):4169
49. Law H, Deng J (2019) CornerNet: detecting objects as paired keypoints. Int J Comput Vis 128:642–656
50. Lin T (2020) Labelimg. https://github.com/tzutalin/labelImg/blob/master/README. Accessed 08 Apr 2020
51. Ren S, He K, Girshick R, Sun J (2016) Faster R-CNN: towards real-time object detection with region proposal networks. IEEE Trans Pattern Anal Mach Intell 39(6):1137–1149
52. Liu W et al (2016) Ssd: single shot multibox detector. In: European conference on computer vision. Springer, pp 21–37
53. Redmon J, Farhadi A (2018) Yolov3: an incremental improvement. arXiv preprint.02767
54. Girshick R, Donahue J, Darrell T, Malik J (2015) Region-based convolutional networks for accurate object detection and segmentation. IEEE Trans Pattern Anal Mach Intell 38(1):142–158
55. Zhao Z-Q, Zheng P, Xu S-T, Wu X (2019) Object detection with deep learning: a review. IEEE Trans Neural Netw Learn Syst 30(11):3212–3232
56. Huang G, Liu Z, Van Der Maaten L, Weinberger KQ (2017) Densely connected convolutional networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 4700–4708
57. Wu X, Zhan C, Lai Y-K, Cheng M-M, Yang J (2019) Ip102: a large-scale benchmark dataset for insect pest recognition. In: Proceedings

of the IEEE/CVF conference on computer vision and pattern recognition, pp 8787–8796

58. Krizhevsky A, Sutskever I, Hinton G (2012) Imagenet classification with deep convolutional neural networks. Adv Neural Inf Process Syst 25:1097–1105
59. Szegedy C et al (2015) Going deeper with convolutions. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 1–9
60. Simonyan K, Zisserman A (2014) Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556
61. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 770–778
62. Alom MZ, Hasan M, Yakopcic C, Taha TM, Asari VK (2020) Improved inception-residual convolutional neural network for object recognition. Neural Comput Appl 32(1):279–293
63. Newell A, Yang K, Deng J (2016) Stacked hourglass networks for human pose estimation. In: European conference on computer vision. Springer, pp 483–499
64. Atila Ü, Uçar M, Akyol K, Uçar E (2021) Plant leaf disease classification using EfficientNet deep learning model. Eco Inform 61:101182
65. Deng J, Dong W, Socher R, Li L-J, Li K, Fei-Fei L (2009) Imagenet: a large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. IEEE, pp 248–255
66. Zhang S, Wen L, Bian X, Lei Z, Li SZ (2018) Single-shot refinement neural network for object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp 4203–4212
67. Lin T-Y et al (2014) Microsoft coco: common objects in context. In: European conference on computer vision. Springer, pp 740–755
68. Reza MT, Mehedi N, Tasneem NA, Alam MA (2019) Identification of crop consuming insect pest from visual imagery using transfer learning and data augmentation on deep neural network. In: 2019 22nd international conference on computer and information technology (ICCIT). IEEE, pp 1–6
69. Ayan E, Erbay H, Varçın F (2020) Crop pest classification with a genetic algorithm-based weighted ensemble of deep convolutional neural networks. Comput Electron Agric 179:105809
70. Zhou S-Y, Su C-Y (2020) Efficient convolutional neural network for pest recognition-ExquisiteNet. In: 2020 IEEE Eurasia conference on IOT, communication and engineering (ECICE). IEEE, pp 216–219
71. Ren F, Liu W, Wu G (2019) Feature reuse residual networks for insect pest recognition. IEEE Access 7:122758–122768
72. Liu W, Wu G, Ren F, Kang X (2020) DFF-ResNet: an insect pest recognition model based on residual networks. Big Data Min Anal 3(4):300–310
73. Albattah W, Nawaz M, Javed A, Masood M, Albahli S (2022) A novel deep learning method for detection and classification of plant diseases. Complex Intell Syst 8(1):507–524

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.