Computers and Electronics in Agriculture 169 (2020) 105174

Contents lists available at ScienceDirect

Computers and Electronics in Agriculture

journal homepage: www.elsevier.com/locate/compag

Crop pest recognition in natural scenes using convolutional neural networks

Yanfen Li[a], Hanxiang Wang[a], L. Minh Dang[a], Abolghasem Sadeghi-Niaraki[a,b], Hyeonjoon Moon[a,*]

[a] Department of Computer Science and Engineering, Sejong University, Seoul, Republic of Korea
[b] Geoinformation Tech. Center of Excellence, Faculty of Geodesy& Geomatics Engineering, K.N. Toosi University of Technology, Tehran, Iran

ARTICLE INFO

Keywords:
Pest classification
CNN
Deep learning
GoogLeNet
Crop
Natural scenes

ABSTRACT

Crop diseases and insect pests are major agricultural problems worldwide, because the severity and extent of their occurrence causes significant crop losses. In addition, traditional crop pests recognition methods are limited, ineffective, and time-consuming due to the manual selection of the useful feature sets. This paper introduces a crop pest recognition method that accurately recognizes ten common species of crop pests by applying several deep convolutional neural networks (CNNs). The main contributions of this paper are (1) a manually collected and validated crop pest dataset is described and shared; (2) a fine-tuned GoogLeNet model is proposed to deal with the complicated backgrounds presented by farmland scenes, with pest classification results better than the original model; and (3) the fine-tuned GoogLeNet model obtains an improvement of 6.22% compared to the state-of-the-art method. As a result, the proposed model has the potential to be applied in real-world applications and further motivate research on crop disease identification.

1. Introduction

Crop pests cause significant losses to crops in the world, whether in developing or developed countries. According to recent research, nearly half of the crop yield in the world is lost to pest infestations and crop diseases (Gandhi et al., 2018). As a result, meticulous pest control is a crucial task to reduce losses and improve crop yields. Once pests infect a field, they must be identified in time, so farmers can provide timely treatment and prevent the spread of pests. However, traditional pest identification methods have many drawbacks. Firstly, most of the commonly used methods are manual investigation, in which the experts or farmers manually inspect the field daily, weekly, and monthly for any sign of pests or diseases. Secondly, there are many types of insects and the number of individuals that belongs to the same species is enormous (Lim et al., 2018). Therefore, traditional pest identification methods are time-consuming, error-prone, and tedious.

Previously, many automatic pest recognition systems based on different machine learning (ML) algorithms have been proposed (Nguyen et al., 2019a; Nguyen et al., 2019b). For example, an approach that adopted the k-means clustering algorithm to recognize pests was proposed (Faithpraise et al., 2013). The detection was implemented by manually extracting the features and using the relative filter to identify different types of pests, which is time-consuming when the dataset is huge. In another research, a method was put forward for the sugar beet diseases recognition using Support Vector Machines (SVM) and spectral vegetation (Rumpf et al., 2010). The precision result for the SVM multiclass classification was about 86%. One year later, a framework that used image processing and ML to classify five different plant leaf diseases was proposed (Al-Hiary et al., 2011). Experimental results demonstrated that the proposed method successfully identified the target diseases with accuracy ranging from 83% to 94%. Although traditional ML algorithms were proved to perform well when the number of crop pests species was small, they become inefficient when multiple features need to be extracted manually.

Deep learning is a special kind of ML that uses multilevel neural networks that allow computers to learn and extract deep abstract features automatically. In recent years, several deep learning methods have been applied to classify pests and achieved state-of-the-art results in numerous pest detection applications. A deep learning-based pests and diseases classification framework on the tomato leaves was implemented (Shijie et al., 2017) and obtained an average classification accuracy of 89%. However, this method can only be applied in simple background pest classification, so it is impossible to be integrated into practical applications. In another approach, Generative Adversarial Networks (GAN) was applied to extend the dataset and the extended dataset was fed into a pre-trained CNN model. This model achieved the plant diseases classification accuracy of 92% (Gandhi et al., 2018). Previously, data augmentation technique was also applied to extend the dataset for classifying breast mass disease. After that, the data was trained on GoogLeNet model and obtained a high accuracy of 93.4%

* Corresponding author.
E-mail address: hmoon@sejong.ac.kr (H. Moon).

https://doi.org/10.1016/j.compag.2019.105174
Received 18 July 2019; Received in revised form 20 December 2019; Accepted 23 December 2019

(Lévy and Jain, 2016). A deep learning model was proposed to recognize 13 kinds of diseases (Sladojevic et al., 2016). Manual image preprocessing was adopted to highlight the target area, which was time-consuming. Recently, a method using deep learning architecture for fruit fly recognition was proposed and achieved an accuracy of 95.68% (Leonardo et al., 2018). In another work, ten species of plant pests were used to train the model and the classification accuracy was 93.84% (Dawei et al., 2019). By analyzing previous work, deep learning methods have been proved to improve the performance pests classification significantly (Dang et al., 2018; Dang et al., 2019).

By showing the strengths and weaknesses of previous works, there is an immediate demand to propose an intelligent expert system that can efficiently and automatically identify crop pests images which have background noise. Therefore, in this research, a deep learning-based model that can automatically classify ten types of crop pests in natural scenes is introduced. The following questions will be addressed in this manuscript:

a. Which deep learning models are most suitable for the proposed dataset?
b. Which data preprocessing methods improve the overall performance of the model?
c. What is the comparison result with other work?

This paper is divided into six sections. The dataset collection process is shown in Section 2. After that, five deep learning models and the flowchart of the proposed system are explained in Section 3. Section 4 shows experimental results for the deep learning models on the proposed dataset. Finally, the performance of the deep learning method is discussed in Section 5, and conclusions and future work are described in Section 6.

2. Dataset

2.1. Data preparation

In this paper, we selected 10 common species of crop pest, namely Gryllotalpa, Leafhopper, locust, Oriental fruit fly, Pieris rapae Linnaeus, Snail, Spodoptera litura, Stinkbug, Cydia pomonella, Weevil, as shown in Fig. 1.

There are many studies which considered these crop pests as the research subjects (Cheng et al., 2017; Wang et al., 2017; Leonardo et al., 2018; Xiao et al., 2018; Dawei et al., 2019). Because these crop pests can be found all over the world and their reproductive speed is very fast. It is challenging to treat once they infect the field, and they can cause enormous losses to crop yields. Therefore, these ten pests are significant for research in order to detect them efficiently and implement timely control treatments.

The dataset collection was mainly done by downloading images from popular search engines (Google, Baidu, Yahoo, and Bing) and performing outdoor shooting using the Apple 7 Plus mobile phone. The dataset contains a total of 5629 images; most of the images were downloaded from the Bing search engine, and 650 images were crawled from other websites. Overall, pest type contains more than 400 images, and the snail class has the largest number of images (over 1000). In general, the sizes of these pests are tiny, and it is difficult to find them quickly in natural scenes with the naked eye. After the data collection process, data augmentation method was implemented to generate more images from the original dataset.

2.2. Data augmentation

Data augmentation is a common technique in deep learning to create more data (Cui et al., 2015; Montserrat et al., 2017). Data augmentation includes two main categories, which are offline augmentation and online augmentation. The offline augmentation operates on the dataset directly, which can be applied to relatively small datasets. The main methods are rotation, translation, flipping, and other corresponding changes. For large datasets, online augmentation is a more suitable approach. In this study, the offline augmentation method is applied because the collected dataset is small. Data augmentation has two main advantages. (1) CNN models achieve better generalization ability, (2) The robustness of the model is improved by adding noise data.

As shown in Fig. 2, the main data augmentation methods applied in this study are 90 clockwise degree rotation, mirroring, noise addition, and zooming. By using these image processing techniques, the number of images datasets increased to nearly four times. The total number of images was 5629 originally. After applying the data augmentation, the number of images increased to 14,475.

The new dataset was then divided into a training set and testing set with a 9:1 ratio, as shown in Table 1.

3. Methodology

The flowchart that describes the main processes of the pests classification framework is shown in Fig. 3. After the data collection process, natural background images were preprocessed by two different background removal methods. And then, more images were generated by applying data augmentation techniques mentioned in Section 3.2. Next, the images were fed into five deep learning models, and the most suitable CNN model was selected.

3.1. Image preprocessing

In previous pest identification research, image segmentation algorithms were applied to segment the target object from the complicated background and to reduce the influence of complex background on the

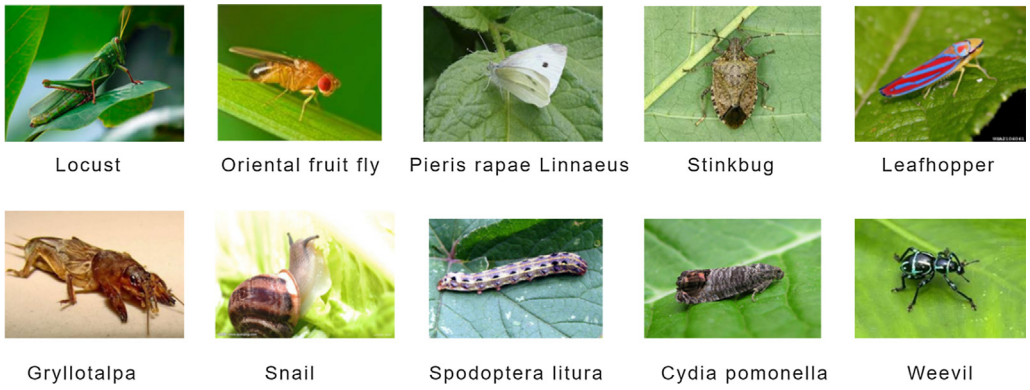

Fig. 1. Sample images for ten common pest classes.

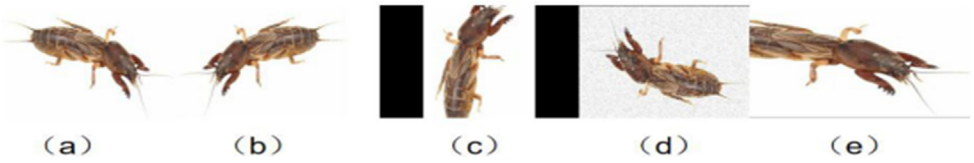

Fig. 2. Example of data augmentation techniques (a) Original image, (b) Mirror image, (c) Rotate 90 degrees and crop, (d) Noise-added image, and (e) Cropped image.

Table 1
Detailed report of the collected dataset before and after applying the augmentation process. The number of train and test images for each class are also shown.

Class	Name	Original	After augmentation	Train	Test
1	Cydia pomonella	415	1165	1049	116
2	Gryllotalpa	505	1243	1119	124
3	Leafhopper	429	1582	1424	158
4	Locust	621	1412	1271	141
5	Oriental fruit fly	461	1545	1391	154
6	Pieris rapae Linnaeus	541	1746	1572	174
7	Snail	1074	1500	1350	150
8	Spodoptera litura	402	1212	1091	121
9	Stinkbug	679	1511	1360	151
10	Weevil	502	1559	1404	155
Total images		5629	14,475	13,031	1444

overall accuracy (Boissard et al., 2008). The CNN models can be easily affected by noises from natural images, so they miss or ignore crucial features, which not only have a certain impact on the overall performance, but also affect the speed of training. Therefore, data preprocessing is an important technique to reduce these problems (Al-Hiary et al., 2011). In this study, two preprocessing methods are implemented to remove the complex background of the image.

3.1.1. *Mixed image processing techniques*

The first preprocessing technique deals with the images that have considerable differences between the color of the pest and the background. The main techniques include thresholding, contour detection, and watershed algorithm.

First of all, an adaptive threshold method is implemented to convert the original image into a binary image. It automatically determines the binary threshold of a pixel according to the distribution of adjacent pixel blocks. After that, morphological operations, including dilation

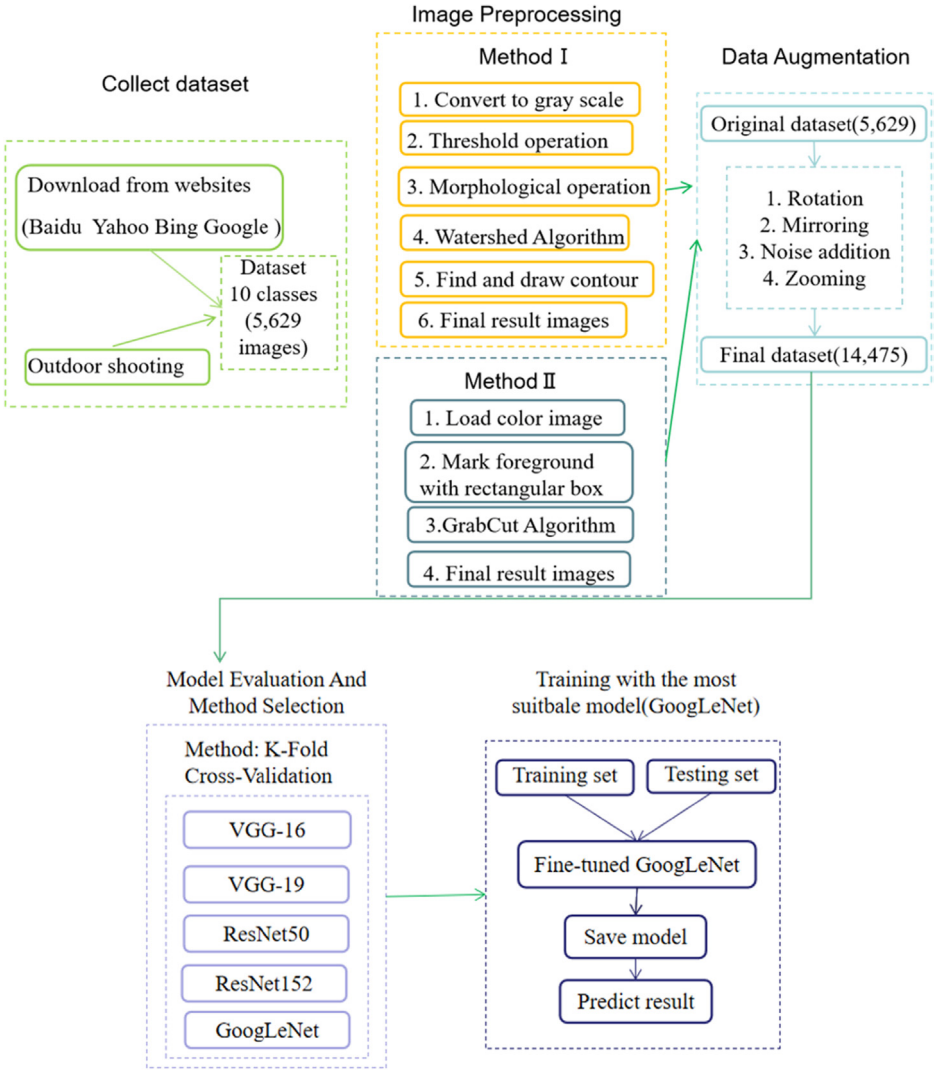

Fig. 3. Overall process of the proposed model from (1) data collection to (2) image processing, and (3) the training process.

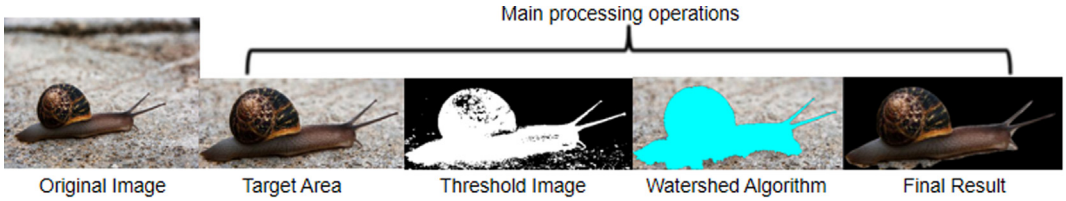

Fig. 4. Input image and main process of watershed algorithm and the final result after applying the background removal algorithm.

and erosion, are applied to remove part of the noise and connect parts of the disconnected pixels of the pest body. A foreground and a background image are extracted after the morphological operations, and then the watershed algorithm is used to localize the pest (Seal et al., 2015). Finally, a contour detection method is applied to obtain the contour of the target. The illustration of the mentioned procedures is shown in Fig. 4.

3.1.2. GrabCut algorithm

The GrabCut algorithm is implemented to remove the background when the foreground and background are similar (Boykov and Jolly, 2005). The foreground and the background of need to be determined before applying the grabcut algorithm. After loading an image, the pest is surrounded by a rectangular box, and everything outside the box is considered as the background. Initially, the algorithm marks the background according to the provided data, and a Gaussian mixture model (GMM) is applied to simulate the foreground and background. Based on the input, GMM model learns and creates new pixel distributions.

For the unspecified pixels in the rectangular box, which can be either foreground or background. They are classified based on their relevance to the pixels of known classifications similar to the clustering operation. Five Gaussian models are corresponding to foreground and background. The equation for calculating the Gaussian mixture probability is:

$$D(x) = \sum_{i=1}^{K} \pi i g i(x; \mu, \sum i), \sum_{i=1}^{K} \pi i = 1,\ 0 \leqslant \pi i \leqslant 1 \tag{1}$$

$$g(x; \mu, \sum) = \frac{1}{\sqrt{(2x)^d |\sum|}} \exp\left[-\frac{1}{2}(x-\mu)^T \sum^{-1} (x-\mu)\right] \tag{2}$$

In Eq. (1), $x$is a BGR three-channel vector; Krepresents the number of Gaussian components corresponding to each pixel ($K = 5$); $\pi_i$represents the weight of each Gaussian component. Formulas (2) are the concrete expression of g_i, it refers to the probability model formula of the i-th Gaussian model, which contains two parameters, mean value (μ_b, μ_g, μ_r) and covariance matrix. So three parameters, which include π_i, mean value(μ), and covariance need to be initialized (Rother et al., 2011).

After classifying the pixels, a pair of pixel distribution map is extracted in which the nodes are the pixels. In addition to the pixels as nodes, there are two other nodes (source node and sink node). All foreground pixels are linked to the source node, and all background pixels are connected to the sink node. Then the minimum cut algorithm is used to segment the newly obtained image (Boykov and Jolly, 2005). The ideal image can not be obtained after a single GrabCut operation, as shown in Fig. 5(b). According to the practical situation, the manual markers are used to make the computer know which areas are needed, so after many iterations, the desired image can be obtained, as shown in Fig. 5(c).

3.2. CNN models

CNN model contains convolution layers, pooling layers and fully connected layers. Image features can be extracted through convolutional operations. Then, the pooling layers are applied to reduce the volume of data processing and retain useful features. The fully connected layer is responsible for reconstructing previously neglected local features into a complete image through the weight matrix.

In this study, five different CNN models were investigated, including VGGNet (VGG-16 and VGG-19), ResNet (ResNet50 and ResNet152) and GoogleNet (Inception-V3). These networks have achieved state-of-the-art performance in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) (Simonyan and Zisserman, 2014; Szegedy et al., 2016). Moreover, many studies have applied these popular deep learning models on the pests detection (Cheng et al., 2017; Leonardo et al., 2018). The following parts give a brief introduction for the five CNN models.

3.2.1. VGG-16 and VGG-19

VGGNet ranked second in the ILSVRC Competition in 2014, and it outperformed GoogLeNet in many transfer learning tasks. It consists of three fully connected layers and five convolution layers. Compared with the other deep learning networks, VGGNet has a concise structure, a small pooling size, and a wider feature map, which makes the architecture deeper, wider, and at the same time decrease the computational time. In this study, the two most famous models, VGG-16 and VGG-19, were used, which contain 13 and 16 convolutional layers, respectively. Fig. 6 demonstrates the differnces between the VGG-16 and VGG-19 models.

3.2.2. ResNet50 and ResNet152

ResNet won first place in the image classification task of the ImageNet competition in 2015. Its main contribution is to solve the degradation problem and the vanishing gradient problem of previous models by introducing building block and Bottleneck structures (He et al., 2016). The residual network uses a network structure that is eight times larger than VGGNet, but it is simpler than VGGNet. In this study, Resnet50 and ResNet152 were investigated, and they contain five blocks of convolution layers with the input size of 224 × 224. Fig. 7 shows the structure of the two models, and the text box at the bottom right corner explains that ResNet152 has 34 more building blocks in the third and fourth convolution blocks than ResNet50. Moreover, each block contains three convolution layers, so the ResNet152 model has 102 more convolution layers than the ResNet50 model.

3.2.3. GoogLeNet

GoogLeNet is a new structure of deep learning proposed in 2014. In recent years, GoogLeNet has been proved to perform well in many practical classification tasks. In this paper, Inception-V3 is used as the implementation of GoogLeNet model. A significant improvement of Inception-v3 is the factorization. The convolution of 7 × 7 is decomposed into two one-dimensional convolutions (1 × 7, 7 × 1), and the convolution of 3 × 3 is decomposed into (1 × 3, 3 × 1) to increase the network depth. Inception-V3 consists of 5 convolution layers, 3 inception modules in block1, 5 inception modules in block2, and 2 inception modules in block3 (Szegedy et al., 2016). In the first five convolution layers, the kernel size is 3 × 3, the first convolution layer has a stride size of 2, and the other convolution layers have a stride size of 1. In addition, the default input size of the data is 299 × 299.

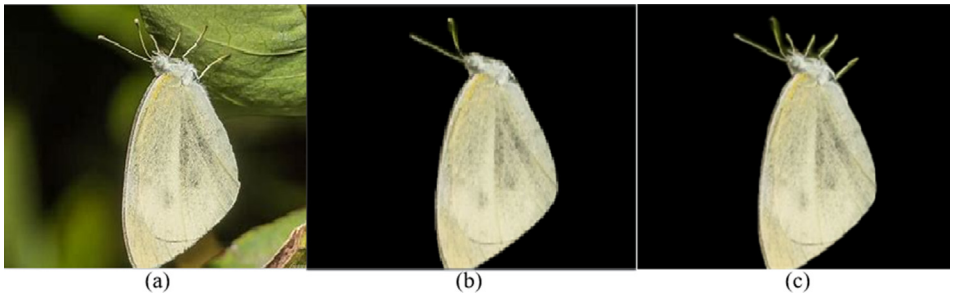

Fig. 5. After applying a single GrabCut operation, the entire background is removed from the raw image(a). However, the pest legs are missing after the above process in (b), so we use the mask of the pest to ensure its body is retained after the Grabcut process, (c) is the final image.

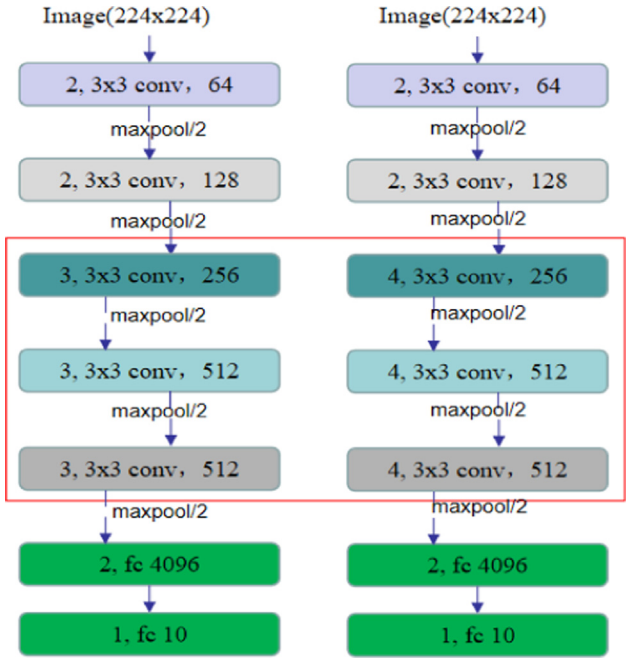

Fig. 6. Overall architecture of VGG model, which shows the difference between VGG-16 (Left) and VGG-19 (Right).

4. Experimental result

All experiments were implemented on a Linux machine pre-installed with Ubuntu 14.04. It has four Titan X 12 GB GPUs, an Intel® Core i7-5930K processor, and 64 GB of DDR4 RAM. Firstly, the feature extraction process is explained in Section 4.1. After that, Section 4.2 shows how the CNN models are trained and tested on the proposed dataset.

4.1. Features extraction

In this section, Keras and OpenCV were used to visualize middle layer feature maps of the CNN model to show how the CNN model extract the abstract features (Zeiler and Fergus, 2014). Fig. 8 explains the details of the feature extraction process on several images.

The first column is the original crop pest image, while the second column is the feature maps extracted by the first convolution layer (there are many feature maps, only 9 of which are randomly selected). Finally, the third column shows fused images of the feature maps. Some background features marked with red circles are not necessary for the feature extract, and they can affect the overall performance.

In order to prove the influence of complex background on the classification performance, an experiment that uses different input data is implemented. In this experiment, two pests with similar morphology (Locust and Gryllotalpa) are selected as the classification subjects. 1000 complex background images and 1000 simple background images are trained on the GoogleNet model. Table 2 shows the details of the input data and the performance of the deep learning model. The experimental results show that the classification accuracy on simple background images is 5.9% higher than complex background images.

4.2. Comparison of the performance of five models

In this section, the classification performance of five deep learning models is examined. These models were configured to use the same optimizer (SGD), the classifier (softmax) and learning rate (0.0001), and then 5-fold cross-validation method was implemented reduce the over-fitting and under-fitting problems (Bergmeir et al., 2018). In the 5-fold cross-validation approach, the training dataset is divided into five subsets (each subset contains 2895 images). In each fold, one subset is used as the testing set, and the remaining subsets are used as the training sets.

Fig. 9 shows the classification results of the five models. Overall, they achieve the accuracy of over 90%, and GoogLeNet obtains the highest accuracy in every fold. At fold 3, GoogLeNet gets the highest classification accuracy of 94.61%, whereas ResNet152 and VGG-19 achieve a slightly lower accuracy of 93.02% and 93.05%, respectively. As a result, GoogLeNet is the most suitable model for the collected dataset.

Table 3 shows the computational complexity of these models. ResNet152 is the most complicated model, because it contains 58 million

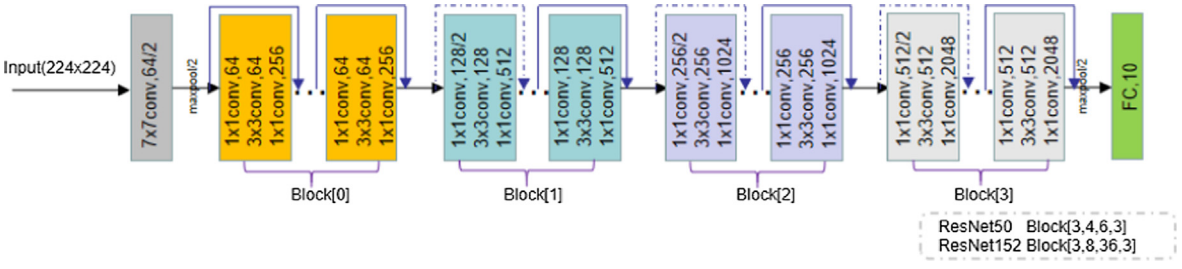

Fig. 7. Overall architecture of ResNet model, which shows the difference between ResNet50 and ResNet152 at Block[3].

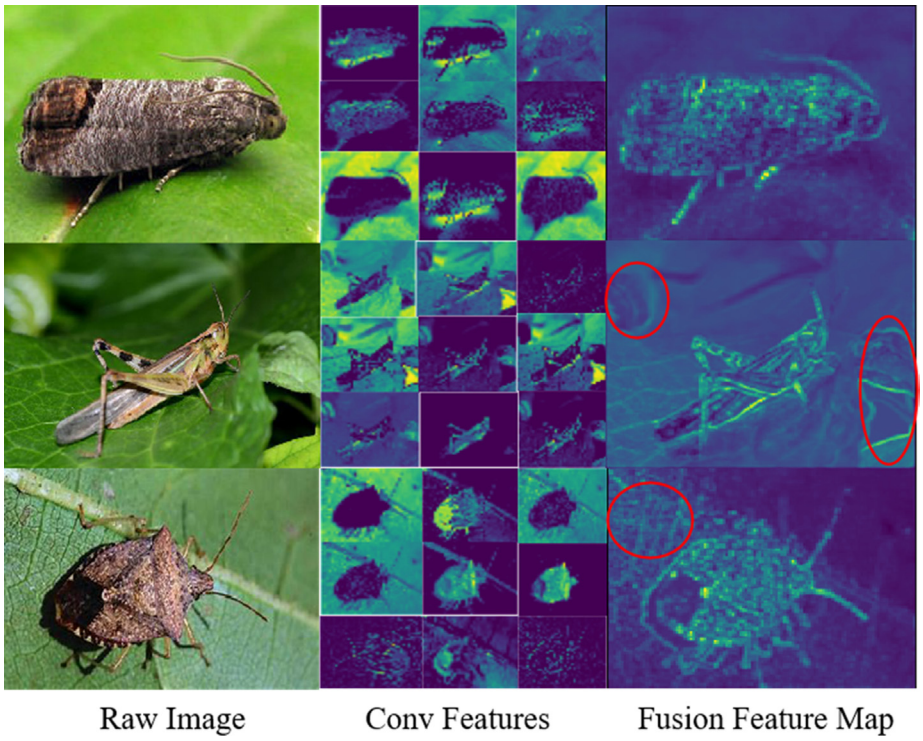

Fig. 8. CNN feature visualization of Inception-V3 (First column: Input images; Second column: Convolutional Features from the first Convolutional layer; Third column: Fusion Feature Map from Maxpooling layer); unimportant background information is marked with a red circle. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Table 2
Information of the input data and the performance of the deep learning model.

Input data (Locust and Gryllotalpa)	Accuracy (%)
Complex background	93
Simple background	**98.9**

Table 3
Computational complexity of the five models.

Model	Number of parameters	Training time(s)
GoogleNet	23 million	4346
ResNet50	25 million	1239
ResNet152	58 million	5520
VGG-19	20 million	1089
VGG-16	15 million	1027

parameters in total. Followed by ResNet50, which has 25 million parameters. The most simple network is VGG-16, which has 15 million parameters. The table also shows the training time of VGG-16, VGG-19, ResNet50, ResNet152, and GoogleNet on the collected dataset. The training time required for VGG-16, VGG-19, and ResNet50 are comparatively similar, among which VGG-16 takes about 17 min because it has fewer layers and fewer parameters than other architectures. ResNet152 and GoogLeNet require more computational time, which takes about 92 min and 72 min, respectively.

4.3. Fine-tuning of GoogleNet model

Fine-tuning refers to the process of receiving weights directly from others' trained networks, and new dataset is used to train the model. In this paper, the GoogleNet model was fined-tuned on the pre-trained ImageNet model, because it helps the network converge faster (Yosinski et al., 2014).

Some adjustments were also made on the model's fully connected

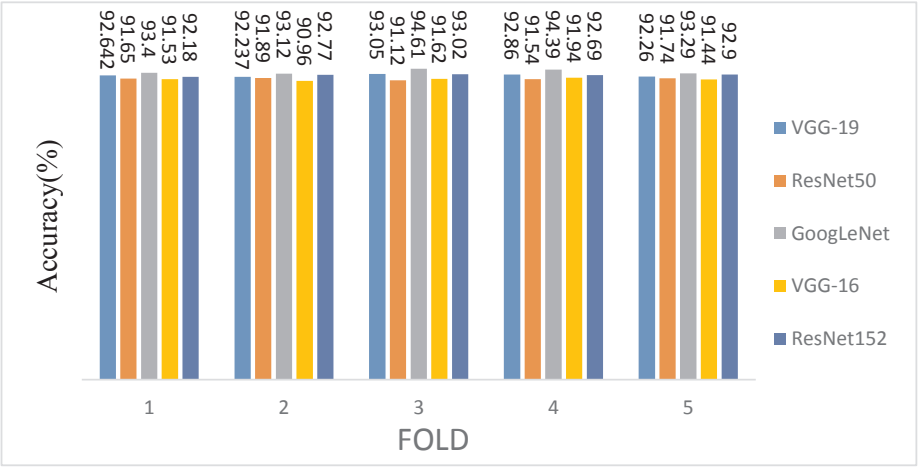

Fig. 9. Comparison in terms of accuracy for five CNN models through 5-fold cross-validation on the collected dataset. GoogLeNet has the best performance.

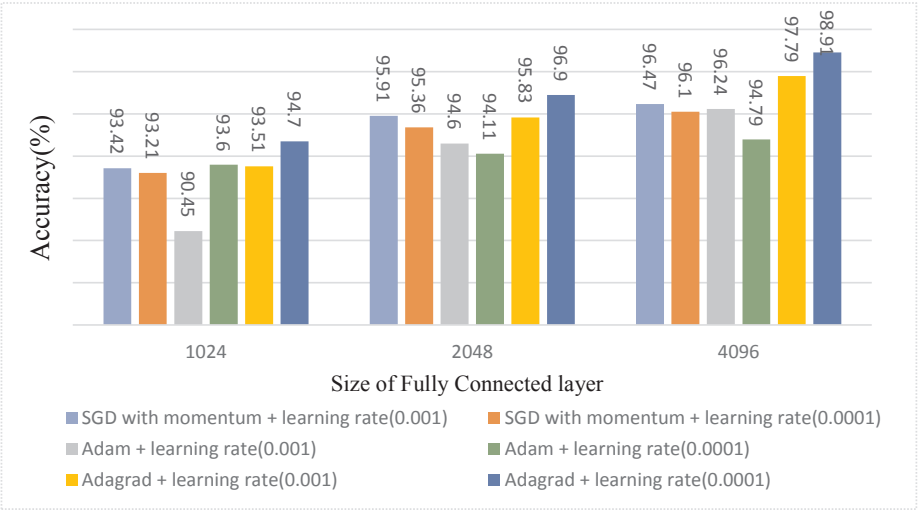

Fig. 10. Comparison of model performance on different parameter sets, including optimizers, momentum and learning rate.

layer and the optimizers because they have a significant impact on the model performance. The collected dataset contains 10 kinds of insects, so the output layer must be changed from 1000 (the pre-trained model) to 10. Moreover, the first 17 layers of the pre-trained model were frozen because these layers have been well trained on the ImageNet dataset. Finally, the parameters of GoogLeNet model were modified to improve the classification accuracy and reduce the computational complexity. Fig. 10 shows the model performance on different sets of parameters, the highest accuracy of 98% is achieved when the fully connected layer size is 4096 and the optimizer is Adagrad (Hadgu et al., 2015) with the optimal learning rate of 0.0001, which is 8% better than the Adam optimizer with the learning rate of 0.001. The size of the fully connected layer has a great impact on the training results because it connects all the neuron and gives the final decision.

Fig. 11 shows that the fine-tuned GoogLeNet achieves higher accuracy and more robust compare to the original model. As the epoch increases, the accuracy of both models keeps increasing and then stabilizes after the 15th epoch. The original model achieves an accuracy of about 93%, whereas the fine-tuned model gets the accuracy of 98.91%.

4.4. Crop pest classification results

Table 4 shows the confusion matrix of the GoogLeNet model on the testing set. The indexes of 10 types of pests are represented as follow: 1. Cydia pomonella, 2. Gryllotalpa, 3. Leafhopper, 4. Locust, 5. Oriental fruit fly, 6. Pieris rapae Linnaeus, 7. Snail, 8. Spodoptera litura, 9. Stinkbug, 10. Weevil.

The result suggests that the model correctly recognizes ten species with an average accuracy of 98.91%. The table shows that the error rate of class 3 (leafhopper) is the highest at 2.84% because the model misclassified it as locus, oriental fruit fly, and snail. Those species have similar shape and color with the background environment. Furthermore, the model achieves 100% classification accuracy on three

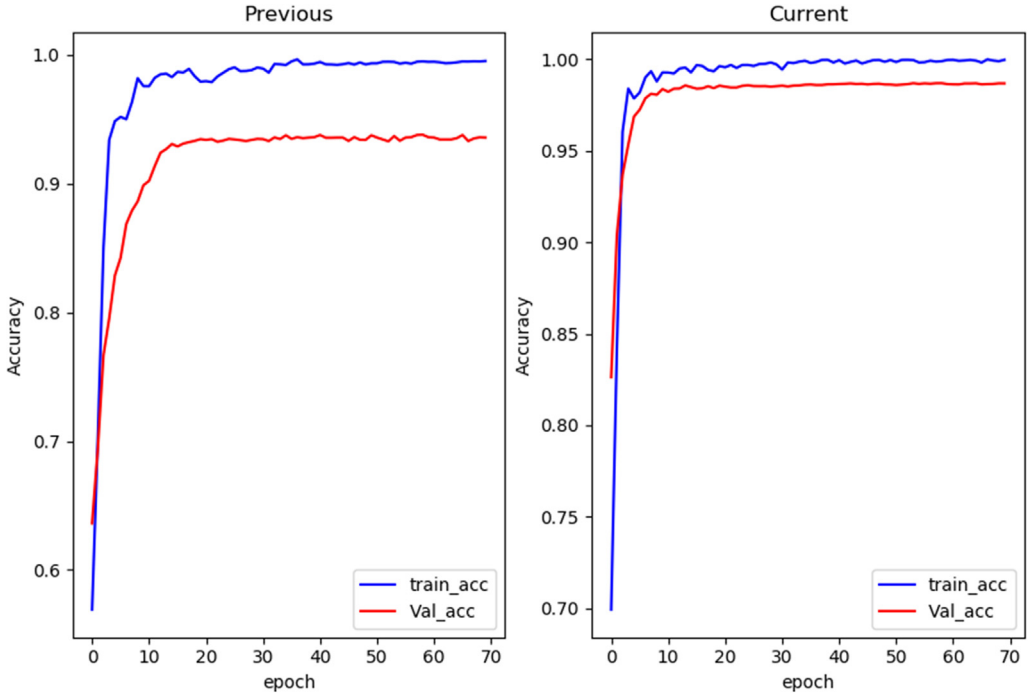

Fig. 11. Performance of the original GoogLeNet and the fine-tuned GoogLeNet.

Table 4
Confusion matrix for the crop pests classification with 10 classes.

Class	1	2	3	4	5	6	7	8	9	10	Accuracy(%)
1	**100**	0	0	0	0	0	0	0	0	0	100
2	0	**110**	0	0	0	0	0	0	0	0	100
3	0	0	**101**	1	1	0	1	0	0	0	97.16
4	0	0	1	**113**	0	0	0	0	1	0	98.26
5	0	0	0	0	**117**	0	0	1	0	0	98.34
6	0	0	0	0	0	**136**	0	0	0	0	100
7	0	0	0	0	0	0	**118**	2	0	0	99.33
8	0	0	0	1	0	0	0	**101**	0	0	99.02
9	0	1	1	0	0	0	0	0	**126**	0	98.43
10	0	0	0	0	0	0	0	0	2	**132**	98.51
Average accuracy											**98.91**

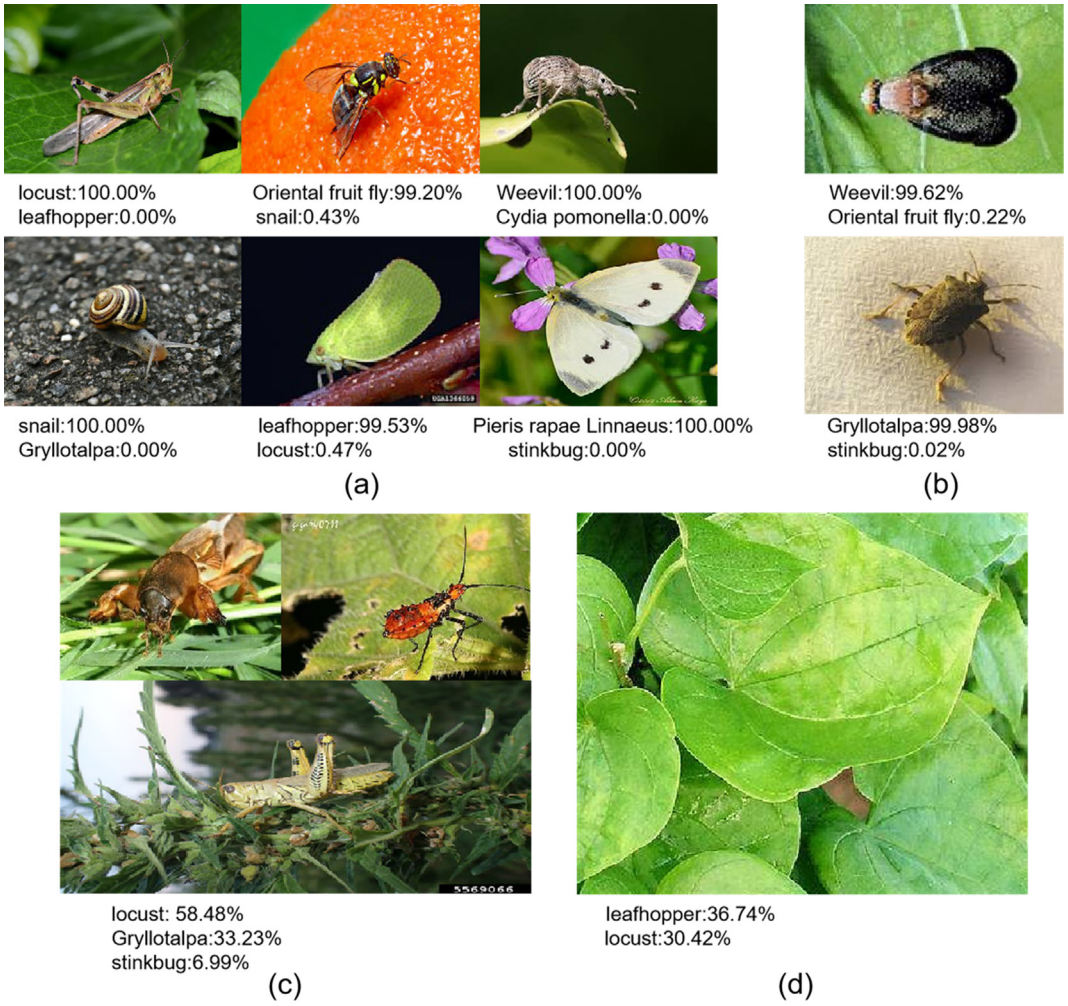

Fig. 12. Classification result for different cases, (a) six correctly classified images; (b) two misclassified images; (c) image that has multiple pests; (d) image without pests.

species of pests (Cydia pomonella, Gryllotalpa, Pieris rapae linnaeus), while the other six species of pests have the accuracy between 98.26% and 99.33%.

Fig. 12(a) shows the correct classification results for 6 randomly selected images using the fine-tuned GoogLeNet model. Fig. 12(b) demonstrates two misclassified results, where the Oriental fruit fly is mistakenly classified as snail because the color and shape of the wings are different from the typical Oriental fruit fly. In addition, Stinkbug is classified as Gryllotalpa because the pest shadow is recognized as part of the pest. Furthermore, Fig. 12(c) and 12(d) show two special cases of the pest classification, Fig. 12(c) has multiple pests and Fig. 12(d) contains no pests. The experimental results show that when there are more than one pests in the image, the model gives the corresponding probability of each class by using the Top-K approach. On the other hand, the model also provides a class names when there are no pests in the image, but the probability for each class is low.

4.5. Comparison with other work

In this section, the fine-tuned model is compared with the state-of-the-art CNN model proposed by (Xie et al., 2015; Cheng et al., 2017). In that research, the authors used the dataset that contains 400 images in

Table 5
Parameters setting for GoogLeNet and ResNet101.

Model	Learning rate	Decay	Optimizer	Class/Number of images	Training time (s)	Accuracy (%)
GoogLeNet	0.0001	0	Adagrad	10/550	3765	96.67
ResNet101	0.0001	0.0005	SGD with momentum (0.9)	10/550	4938	90.45

the training set and 150 images in the testing set. The model used in (Cheng et al., 2017) was the fine-tuned ResNet101. The momentum parameter was set to 0.9, and the basic learning rate was 0.0001. The fine-tuned ResNet101 achieved a high accuracy of 98.67%.

In this paper, the fine-tuned GoogLeNet model is compared with the fine-tuned ResNet101 model. The experimental result in Table 5 shows that the fine-tuned GoogLeNet model performs better than the fine-tuned ResNet101 in both accuracy and training time.

5. Discussion

Based on the experimental results, three questions proposed in the introduction are answered in this section. The first question was about the best model used in this research. In this research, five popular CNN models (VGG-16, VGG-19, ResNet50, ResNet152 and GoogLeNet) were used. Among them, GoogLeNet was selected as the target model because this model showed the highest performance. The second question asked about which data preprocessing methods are used to improve the overall performance of the model. As shown in Section 4.1, two background removal methods were applied before the training process. After that, data augmentation was used to generate more training data. As for the last question, the optimized model was compared to another work, and the experiment results proved that the GoogLeNet model achieved the classification accuracy of 6.22% higher than the ResNet101 model. The experimental result proved that the GoogLeNet model was effective and robust for the identification of crop pests, and can significantly reduce processing times and labour costs if it is integrated into the practical applications. Although GoogLeNet achieves better accuracy than other models, certain limitations can be seen, such as it requires higher computing capacity and more training time. Moreover, the complex architecture of Inception-V3 makes it difficult to adjust the layer structure according to the dataset.

6. Conclusion

In this research, deep learning-based pests detection framework was proposed to classify ten types of crop pests using a manually collected dataset. A total of 5629 images was downloaded from different websites and manually validated. In the data preparation phase, data augmentation was used to expand the dataset. In addition, GrabCut and watershed algorithms were implemented to remove the complicated background. In the training phase, VGG-16, VGG-19, ResNet50, ResNet152, and GoogLeNet were investigated. The experimenta show that the GoogLeNet model outperformed other models in terms of accuracy, model complexity, and robustness. The fine-tuned model achieved 5.91% higher accuracy than the original model.

In the future, more concentration should be put on the development of crop pest identification systems on mobile devices, because mobile devices are widely accessible to farmers. Besides RGB images, infrared images can be employed to monitor the pests, and the infrared filter can be easily equipped on the UAV.

CRediT authorship contribution statement

Yanfen Li: Data curation, Methodology, Visualization, Writing - original draft. **Hanxiang Wang:** Data curation, Methodology, Writing - original draft. **L. Minh Dang:** Formal analysis, Investigation, Writing - review & editing. **Abolghasem Sadeghi-Niaraki:** Conceptualization. **Hyeonjoon Moon:** Conceptualization, Funding acquisition, Validation.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (2019-0-00136, Development of AI-Convergence Technologies for Smart City Industry Productivity Innovation) and by Korea Institute of Planning and Evaluation for Technology in Food, Agriculture, Forestry and Fisheries(IPET) through Agri-Bio Industry Technology Development Program, funded by Ministry of Agriculture, Food and Rural Affairs(MAFRA) (316033-04-2-338 SB030).

Appendix A. Supplementary material

Supplementary data to this article can be found online at https://doi.org/10.1016/j.compag.2019.105174.

References

Al-Hiary, H., Bani-Ahmad, S., Reyalat, M., Braik, M., ALRahamneh, Z., 2011. Fast and accurate detection and classification of plant diseases. Int. J. Comput. Appl. 17, 31–38.
Bergmeir, C., Hyndman, R.J., Koo, B., 2018. A note on the validity of cross-validation for evaluating autoregressive time series prediction. Comput. Stat. Data Anal. 120, 70–83.
Boissard, P., Martin, V., Moisan, S., 2008. A cognitive vision approach to early pest detection in greenhouse crops. Comput. Electron. Agric. 62, 81–93.
Boykov, Y., Jolly, M.-P. (2005). Graph cuts for binary segmentation of n-dimensional images from object and background seeds. Google Patents.
Cheng, X., Zhang, Y., Chen, Y., Wu, Y., Yue, Y., 2017. Pest identification via deep residual learning in complex background. Comput. Electron. Agric. 141, 351–356.
Cui, X., Goel, V., Kingsbury, B., 2015. Data augmentation for deep neural network acoustic modeling. IEEE/ACM Trans. Audio Speech Language Process. (TASLP) 23, 1469–1477.
Dang, L.M., Hassan, S.I., Suhyeon, I., kumar Sangaiah, A., Mehmood, I., Rho, S., Seo, S., Moon, H. (2018). UAV based wilt detection system via convolutional neural networks. Sustain. Comput.: Informat. Syst.
Dang, L.M., Piran, M., Han, D., Min, K., Moon, H., 2019. A survey on internet of things and cloud computing for healthcare. Electronics 8, 768.
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, 4524–4531.
Faithpraise, F., Birch, P., Young, R., Obu, J., Faithpraise, B., Chatwin, C., 2013. Automatic plant pest detection and recognition using k-means clustering algorithm and correspondence filters. Int. J. Adv. Biotechnol. Res. 4, 189–199.
Gandhi, R., Nimbalkar, S., Yelamanchili, N., Ponkshe, S. (2018). Plant disease detection using CNNs and GANs as an augmentative approach. In: 2018 IEEE International Conference on Innovative Research and Development (ICIRD). IEEE, pp. 1–5.
Hadgu, A.T., Nigam, A., Diaz-Aviles, E. (2015). Large-scale learning with AdaGrad on Spark. In: 2015 IEEE International Conference on Big Data (Big Data). IEEE, pp. 2828–2830.
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.
Leonardo, M.M., Carvalho, T.J., Rezende, E., Zucchi, R., Faria, F.A. (2018). Deep feature-based classifiers for fruit fly identification (diptera: Tephritidae). In: 2018 31st SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI). IEEE, pp. 41–47.
Lévy, D., Jain, A. (2016). Breast mass classification from mammograms using deep convolutional neural networks. arXiv preprint arXiv:1612.00542.
Lim, S., Kim, S., Park, S., Kim, D. (2018). Development of Application for Forest Insect

Classification using CNN. In: 2018 15th International Conference on Control, Automation, Robotics and Vision (ICARCV). IEEE, pp. 1128–1131.
Montserrat, D.M., Lin, Q., Allebach, J., Delp, E.J., 2017. Training object detection and recognition CNN models using data augmentation. Electronic Imaging 2017, 27–36.
Nguyen, T.N., Lee, S., Nguyen-Xuan, H., Lee, J., 2019a. A novel analysis-prediction approach for geometrically nonlinear problems using group method of data handling. Comput. Methods Appl. Mech. Eng. 354, 506–526.
Nguyen, T.N., Thai, C.H., Luu, A.-T., Nguyen-Xuan, H., Lee, J., 2019b. NURBS-based postbuckling analysis of functionally graded carbon nanotube-reinforced composite shells. Comput. Methods Appl. Mech. Eng. 347, 983–1003.
Rother, C., Kolmogorov, V., Boykov, Y., Blake, A., 2011. Interactive foreground extraction using graph cut. Advances in Markov Random Fields for Vision and Image Processing.
Rumpf, T., Mahlein, A.-K., Steiner, U., Oerke, E.-C., Dehne, H.-W., Plümer, L., 2010. Early detection and classification of plant diseases with support vector machines based on hyperspectral reflectance. Comput. Electron. Agric. 74, 91–99.
Seal, A., Das, A., Sen, P., 2015. Watershed: an image segmentation approach. Int. J. Comput. Sci. Informat. Technol. (IJCSIT) 6, 2295–2297.
Shijie, J., Peiyi, J., Siping, H. (2017). Automatic detection of tomato diseases and pests based on leaf images. In: 2017 Chinese Automation Congress (CAC). IEEE, pp. 2537–2510.
Simonyan, K., Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.
Sladojevic, S., Arsenovic, M., Anderla, A., Culibrk, D., Stefanovic, D., 2016. Deep neural networks based recognition of plant diseases by leaf image classification. Comput. Intell. Neurosci.
Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z., 2016. Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 2818–2826.
Wang, R., Zhang, J., Dong, W., Yu, J., Xie, C.J., Li, R., Chen, T., Chen, H., 2017. A crop pests image classification algorithm based on deep convolutional neural network. Telkomnika 15.
Xiao, D., Feng, J., Lin, T., Pang, C., Ye, Y., 2018. Classification and recognition scheme for vegetable pests based on the BOF-SVM model. Int. J. Agric. Biol. Eng. 11, 190–196.
Xie, C., Zhang, J., Li, R., Li, J., Hong, P., Xia, J., Chen, P., 2015. Automatic classification for field crop insects via multiple-task sparse representation and multiple-kernel learning. Comput. Electron. Agric. 119, 123–132.
Yosinski, J., Clune, J., Bengio, Y., Lipson, H. (2014). How transferable are features in deep neural networks? In: Advances in Neural Information Processing Systems, pp. 3320–3328.
Zeiler, M.D., Fergus, R. (2014). Visualizing and understanding convolutional networks. In: European Conference on Computer Vision, Springer, pp. 818–833.