CDNN Model for Insect Classification Based on Deep Neural Network Approach

Hiep Xuan Huynh[1], Duy Bao Lam[2], Tu Van Ho[1(✉)], Diem Thi Le[1], and Ly Minh Le[1]

[1] Can Tho University, Can Tho, Vietnam
{hxhiep, hvtu, ltdiem, leminhly}@ctu.edu.vn
[2] Mekong University, Long Hồ, Vietnam
lambaoduy84@gmail.com

Abstract. The Mekong Delta has made great progress in rice production over the past ten years. Intensive cultivation with multi-cropping brings many benefits to farmers as well as the food export industry. However, this is also an opportunity for raising epidemic outbreak, Brown Plant-hoppers can directly damage by sucking the rice's vitality, and they can cause the wilting and complete drying of rice plants, a noncontagious disease known as "Hopper-burn". In this article, we propose the CDNN model for insect classification based on Neural Network and Deep Learning approach. First, insect images are collected and extracted features based on Dense Scale-Invariant Feature Transform. Then, Bag of Features is used for image representation as feature vectors. Lastly, these feature vectors are trained and classified using CDNN model based on Deep Neural Network. The approach is demonstrated with experiments, and measured by a large amount of Brown Plant-hoppers and Ladybugs samples.

Keywords: Bag of Features · Brown Plant-hoppers · Classification · Deep neural network · Dense SIFT · Insect · Ladybugs

1 Introduction

Rice cultivation plays a very important role for farmers in Vietnam. There are many insect pests attack rice tree [23], they would destroy the rice crop. Especially, Brown Plant-hopper (BPH), a small insect pest causes extensive crop damages. It has high reproductive capacities. Besides Hopper-burn, BPH also causes serious diseases in rice crop, such as "Rice yellow dwarf disease". In the other hand, predators (beneficial insect such as ladybug, ladybird, and spider) kill and feed on several to many individual insect pest during their lifetimes, they are bio-control groups in agriculture. Without these predators, insect pests would grow and destroy crops quickly [8].

Classification of living insects is on the agenda for several reasons. First, due to climate change, it is important to understand how insects distribute or response. Second, the significant development of insects leads to unbalancing some surrounding conditions. Nevertheless, it is necessary to have agricultural specialists to identify insects. In case of lacking domain experts, a requests for insect recognition and classification to be carried out more efficiently have become pressing. In response, image-

P. C. Vinh and A. Rakib (Eds.): ICCASA 2019/ICTCC 2019, LNICST 298, pp. 127–142, 2019.
https://doi.org/10.1007/978-3-030-34365-1_10

based technology is used to improve the wide range of applications especially in agriculture, ecology and environmental science [5]. It generally can be utilized in prevention of plant disease and insect pests, plant quarantine and as an essential part of eco-informatics research. Insect classification has to be taken into a serious measure because insect presents an especially severe threat and it can cause many negative effects on agriculture in a short period of time.

This paper proposes a novel approach by developing a CDNN model of classifying insects in images based on Neural Network [11, 20] and Deep Learning [3, 16, 35]. Image features are extracted by Dense Scale-Invariant Feature Transform (Dense SIFT) [6] and represented as feature vectors by Bag of Features (BoF) [19, 26, 28, 29]. The research contributes to building a sampling BPH light trap surveillance network in the Mekong Delta, Vietnam [2], helping reduce crop damage caused by insect pests.

The rest of this article is presented as follows. Section 2 depicts some previous work relating to insect image classification. Insect images representation based on the BoF model is presented in the next section. System of insect classification is proposed in Sect. 4. Section 5 illustrates some results of the classification method. The last section is our conclusion and future plans.

2 Related Work

There have been many research of insect detection or classification in image data. Zhu and Zhang [21] introduced a method to classify insects by using color histogram and Gray Level Co-occurrence Matrix (GLCM) of wing images. First, the image of lepidopteran insect is preprocessed to get the ROI (Region of Interest); then the color image is converted from RGB (Red-Green-Blue) to HSV (Hue-Saturation-Value) space, and the 1D color histograms of ROI are generated from hue and saturation distributions. Afterward, the color image is converted to grayscale one, rotated and transformed to a standard position, and their GLCM features are extracted. Matching is first undergone by computing the correlation of the histograms vectors between testing and template images. Then, their GLCM features are further matched when the correlation is higher than certain threshold.

According to Hassan *et al.* [27], several methods used in machine vision learning in detecting and classifying insects based on their features, colors, and shape. Each method has the advantages and disadvantages in detecting insects. Among the method used, color histogram seems to be the best approach in classifying and recognizing species of insects. In the method, each acquired image is divided into several same size squares and each of the square of images has its own histogram. Even though the detected insects is not in the same position in the trained image, the system still can identified which type of insect based on the color histogram.

To improve the classification accuracy, Xie *et al.* [12] develop an insect recognition system using advanced multiple task sparse representation and multiple-kernel learning (MKL) techniques. As different features of insect images contribute differently to the

classification of insect species, the multiple-task sparse representation technique can combine multiple features of insect species to enhance the recognition performance.

In [5], Lu *et al.* proposed a hybrid approach called discriminative local soft coding (DLSoft) which combines local and discriminative coding strategies together. This method used neighbor codewords for getting a local soft coding and class-specific codebooks (sets of codewords) for a discriminative representation. On obtaining the vector representation of image via spatial pyramid pooling of patches, a linear SVM classifier is used for classifying images into species.

Shapes and sizes can be used to detect BPHs in images by using morphology operations [7, 17]. The experimental results show that the proposed approach is suitable for detecting and counting BPHs in images.

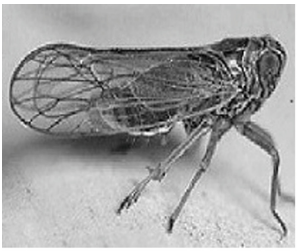
3 Insect Images Representation

3.1 Characteristics of Insect Images

Two interested insect species (see Fig. 1) are BPH [23] and Ladybugs [8], they have an average size of about 4-10 mm, and their characteristics can be identified by morphological. For example, BPH has a yellowish brown body and their head overhangs towards the front, their wings are transparent and the front wings have a black spot on the back side.

Fig. 1. RGB insect image: adult BPH [23] (a) and Ladybug [8] (b)

Insect images were taken by 1280 × 720 pixels resolution. Before extracting feature, insect images were converted to grayscale and stored in grayscale image matrix, each cell of matrix has value between 0 and 255 (see Fig. 2).



BPH grayscale image

71	20	24	67	8	33	5	...
2	4	255	0	170	7	9	...
57	0	23	255	0	123	75	...
0	0.8	10	0.6	20	38	40	...
12	0	46	57	6	8	9	...
0	12	0.4	0.11	0.2	0.8	79	...
9	3	0.6	34	67	89	23	...
...	...	...	...	...	...	...	...

BPH grayscale image matrix

Fig. 2. BPH grayscale image stored in a matrix

3.2 Bag of Features Model

Insect image representation is the key step in classification, its performance directly affects the insect classification results. Bag of Features (BoF) [19, 26, 28, 29] approach can be motivated by an analogy to learning methods using the Bag-of-Words (BoW) [36] representation for text categorization. BoF methods have been applied to image classification, object detection, image retrieval, and even visual localization for robots. BoF approaches are characterized by the use of an orderless collection of image features. Due to its simplicity and performance, the BoF approach has become well-established in the field.

BoF model is designed for representation insect image features as feature vectors. The main idea is to reduce storage space and minimize computation. This model includes 3 main functions as described in Fig. 3: (1) extracting insect image features based on Dense SIFT algorithm, (2) vector quantization using the variant of the K-means algorithm, and (3) constructing bag of features by applying the Spatial Pyramid Matching framework.

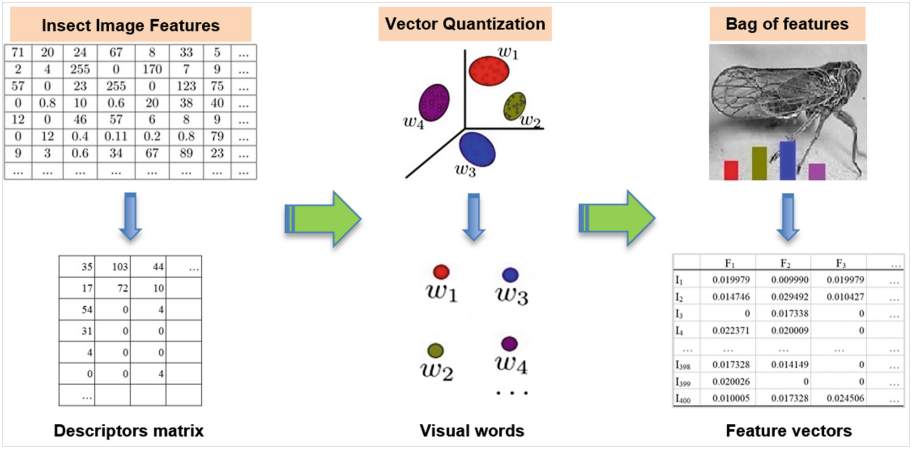


Fig. 3. BoF model for insect images representation

Extracting Insect Image Features

Scale-Invariant Feature Transform (SIFT) [13] provides a set of features of objects in an image. These features are invariant with the change of scale, rotation, view, noise or light intensity in an acceptable level. This method archives high efficiency in image recognition [14]. Four major stages of computation used to generate the set of image features are involved in the following order:

- Scale-space extrema detection: the first stage of computation searches over all scales and image locations. It is implemented efficiently by using a difference-of-Gaussian function to identify potential interest points that are invariant to scale and orientation.
- Keypoint localization: at each candidate location, a detailed model is fit to determine location and scale. Keypoints are selected based on measures of their stability.
- Orientation assignment: one or more orientations are assigned to each keypoint location based on local image gradient directions. All future operations are performed on image data that has been transformed relative to the assigned orientation, scale, and location for each feature, thereby providing invariance to these transformations.
- Keypoint descriptor: the local image gradients are measured at the selected scale in the region around each keypoint. These are transformed into a representation that allows for significant levels of local shape distortion and change in illumination.

Dense SIFT [1, 6, 31, 33] is derived from the SIFT [13, 14], the most significant difference between them is that Dense SIFT assumes all the significant points are evenly distributed. Therefore, the selection of keywords in all areas of the image is dense and standardized.

Dense SIFT operation starts with segmenting a grayscale image into small segments, each of these segment is further divided into smaller segments. For each of these segments, which represent the neighborhoods around the feature point (center of the segment), the image gradients were calculated. A smoothed weighted histogram of eight orientation bins (corresponding to eight directions) is created based on the sum of gradient value. Consequently, a descriptor of a local region (keypoint) is formed by calculating the gradient magnitude and orientation around the keypoint.

These descriptors are set in a Gaussian window. They are accumulated into orientation histograms in 4×4 sub regions of which the length of each arrow depicts the number of orientation bins inside a region. A best results are achieved with a 4×4 array of histograms with 8 orientation bins in each, the local descriptor results in a 128-dimensionl vector. After extracting, each insect image feature is represented by a descriptors matrix.

Vector Quantization

Vector quantization [4] is a classical quantization technique from signal processing that allows the modeling of probability density functions by the distribution of prototype vectors. It works by dividing a large set of points (vectors) into groups having

approximately the same number of points closest to them. Each group is represented by its centroid point, as in K-means and some other clustering algorithms.

A variant of the K-means algorithm [9] is used for grouping descriptor vectors in descriptors matrix to the set of clusters. K-means algorithm is a popular and unsupervised learning algorithm. The goal of the K-means clustering algorithm is to minimize the sum of squared Euclidean distances [11] between each point and the nearest cluster center. In this paper, Dense SIFT feature of each insect image has average feature descriptors number about few ten thousand descriptors. Therefore, in this section we initialize number of clusters with value 1000. Then, clustering process is done by a variant of the K-means algorithm [9], this algorithm applies a technique based acceleration the triangular inequality. Besides, we use k-dimensional tree algorithm [18] for enhancing performance of vector quantization. The results of this phase is the number of descriptor vectors in each cluster, and each cluster is a word in visual words.

Constructing Bag of Features
Visual words is used for constructing feature vectors which represent insect images. The number of descriptor vectors in each cluster are calculated and built a spatial histogram of visual words. Visual words are associated into a spatial histogram by applying the Spatial Pyramid Matching framework [28]. This method combines the technique of generating visual words into the Pyramid matching plan. For each level of space, the pyramid apply a method for matching a series of grids. At each level, the number of histogram matches is counted in each grid and a weighted sum is collected for all resolutions. The result is a histogram for each image taking into account the relative location of image features. It should be pointed out that level of a spatial pyramid L = 0 is equivalent to a standard BoF implementation. After this process, a feature vector which represents an insect image Dense SIFT feature, is obtained.

4 Insect Classification

The system handles insect classification including 3 main phases with functions described in Fig. 4:

- Data phase: real insects are caught and taken their pictures. Some preprocessing tasks are done in order to remove unnecessary points in image.
- BoF representation phase: BoF model is used for image feature extraction and representation as feature vectors.
- Classification phase: these feature vectors are trained by CDNN model and applied for insect images classification.

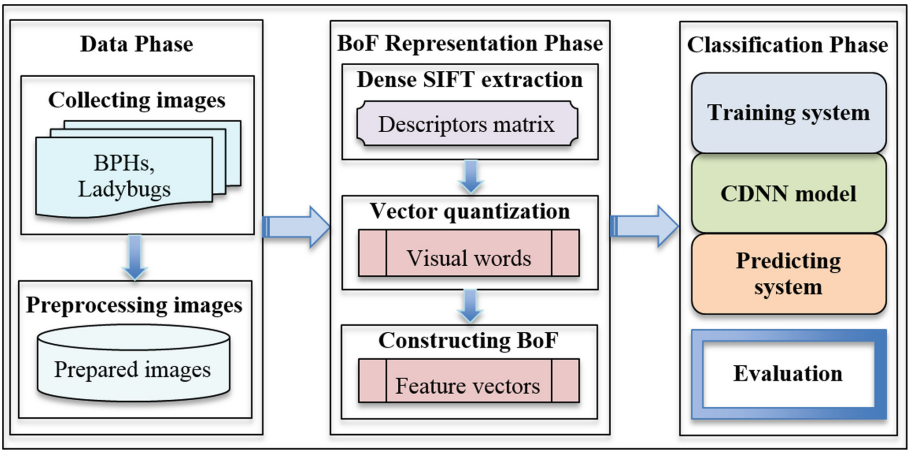


Fig. 4. System of insect classification

4.1 Data Phase

Insect image samples (see Fig. 5) are collected from adult BPH images and ladybug images by using a camera with [1280 × 720] resolution in different views. After preprocessing, information relating to these insects is kept, others are removed.

Fig. 5. BPH image samples

4.2 BoF Representation Phase

Bag of features (BoF) model is designed for representation phase. The number of keypoints is used to form feature vectors. Therefore, feature vector F = {F_1, F_2,…, F_{4000}} represents for each insect image. The steps for BoF representation are as follows:

Algorithm 1: BoF Representation

Input: Insect image
Output: Feature vector

```
Step 1: Extracting insect image features
        Reading insect image
        Converting insect image to grayscale
        Standardizing image size
        Calculating keypoints (KEYPTS) and (descriptors) DESCRS
            [KEYPTS, DESCRS] = DSift.calculate(img)
Step 2: Vector quantization
        Clustering descriptor vectors (WORDS)
            WORDS = Elkan_Kmeans(DESCRS, numWords)
        Building a k-dimensional tree (KDTREE)
            KDTREE = build_kdtree (WORDS)
Step 3: Constructing bag of features
        Computing a spatial histogram of visual words
            HIST = compute_Histogram(KEYPTS, WORDS)
        Reducing a spatial histogram to single
            HIST = single(HIST / sum(HIST))
```

4.3 Classification Phase

Classification phase as described in Fig. 4 includes 4 main functions: (1) CDNN model, (2) training system, (3) predicting system (testing system), and (4) evaluation. In this section, CDNN model and training system are described. Predicting system and evaluation will be presented in the next part.

CDNN Model

Deep neural networks [10, 16] are distinguished from the more commonplace single-hidden-layer neural networks by their depth. The traditional neural network has at most 3 layers: input, hidden, output while a deep neural networks has more than 1 hidden layers. The principle of deep neural networks is that nodes in a layer is trained by specific features based on outputs of the previous layer. The more layers the network has, the more complex features can recognize because it is able to combine features in previous layers.

CDNN model which is designed for insect images classification consists of many layers of interconnected neurons, it is a straight forward neural network as described in Fig. 6. In this model, each layer has a specific role and responsibility.

- Input layer: plays an input role to match inputs relevant to feature vectors of insect images. The number of nodes in the input layer is fixed with number of insect image feature vectors. For example, feature vectors $F = \{F_1, F_2, \ldots, F_{4000}\}$ represents for insect images, the number of nodes in the input layer is $n = 4000$ nodes.
- Hidden layers: is the middle layers, the number of hidden layer and the number of nodes in each hidden layer is designed for the purpose of increasing the accuracy of

the model. CDNN model must has at least 2 hidden layers. In this paper, the experiment is designed in two scenarios: using 2 hidden layers and 3 hidden layers.

- Output layer: is a linear classification which is relevant to the output space. The number of nodes in the output layer is class numbers which needs for classification, this model has two classes (PBHs and Ladybugs).

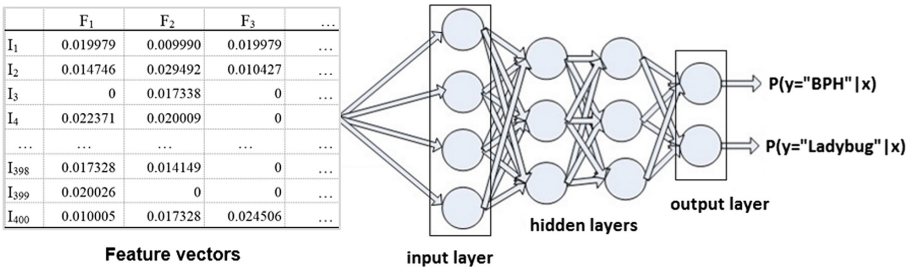


Fig. 6. CDNN model with 2 hidden layers

Training System

The training protocol of CDNN model applies parallel distributed and multi-threaded in H2O Deep Learning [3]. Loss function based on Mean squared error loss [3]:

$$L(W, B \mid j) = \frac{\sum_{i=1}^{n} (t_j - o_j)}{n}$$

Where:

- W is the collection $\{w_i\}_{1:N-1}$, where W_i denotes the weight matrix connecting layers i and i + 1 for a network of N layers.
- B is the collection $\{b_i\}_{1:N-1}$, where b_i denotes the column vector of biases for layer i + 1.
- j is training sample.
- n is number of training samples.
- t_j is value the predicted output (target output).
- o_j is output value of the network (actual output).

The process of minimizing the loss function $L(W, B \mid j)$ is a parallelized version of stochastic gradient descent (SGD). The gradient $\nabla L(W, B \mid j)$ updated with back-propagation algorithm [34]. The SGD method is fast and optimal memory but not easily parallelizable without becoming slow, Hogwild update method [15] is used for resolving this problem.

Bias units are included in each non-output layer of the network. The weights linking neurons and biases with other neurons fully determine the output of the entire network. Learning occurs when these weights are adapted to minimize the error on the labeled training data.

Let the constant α is the learning rate to control the step sizes during gradient descent. Avg_n represents the final averaging of these local parameters across all nodes to obtain the global model parameters and complete training. The following steps outline the training protocol of CDNN model:

Algorithm 2: The training protocol of CDNN model

Step 1: Initialing global parameters: Weights (W) and Biases (B)
Step 2: Distributing training data T to all nodes
Step 3: Repeat until convergence criterion reached:
    3.1. For nodes $n \in T_n$, do in parallel:
        Obtain copy of the global model parameters W_n, B_n
        Select active subset $T_{na} \subset T_n$
        Partition T_{na} into T_{nac} by cores n_c
        For cores n_c on node n, do in parallel
            Get training sample $i \in T_{nac}$
            Update all weights $w_{jk} \in T_n$, biases $b_{jk} \in B_n$
            $$w_{jk} = w_{jk} - \propto \frac{\partial L(W,B|j)}{\partial w_{jk}}$$
            $$b_{jk} = b_{jk} - \propto \frac{\partial L(W,B|j)}{\partial b_{jk}}$$
    3.2. Updating global parameters
        $W, B = Avg_n W_n, Avg_n B_n$
    3.3. Optionally score the model on train/validation scoring sets

In all of the CDNN model nodes, input data is distributed and training on all nodes, weight and bias are calculated in parallel on each node until weights and biases (W, B) obtained by averaging.

5 Experimental Results and Discussion

In all experiments, the learning rate is $\alpha = 0.005$ to control the step sizes during gradient descent. Weights and biases are randomly initialized. Accuracy and error rate are calculated by confusion matrix [30].

5.1 Data Used

Experiments were performed on BPH images and Ladybug images. All experimental data are describe in Table 1.

Table 1. Experimental data.

Insect name	Training sets	Testing sets	Total
BPH	200	100	300
Ladybug	200	100	300

All insect images in the training dataset and testing dataset were extracted Dense SIFT features and applied BoF model for feature vectors. Training feature vectors and testing feature vectors are described in Tables 2 and 3. The test labels are not assigned.

Table 2. Training feature vectors.

	F_1	F_2	F_3	…	F_{3999}	F_{4000}	Label
I_1	0.019979	0.009990	0.019979	…	0.098386	0.101874	BPH
I_2	0.014746	0.029492	0.010427	…	0.261088	0	BPH
I_3	0	0.017338	0	…	0.478497	0.484118	BPH
I_4	0.022371	0.020009	0		0.255067	0.086062	BPH
…	…	…	…	…	…	…	…
I_{398}	0.017328	0.014149	0	…	0	0	LDBUG
I_{399}	0.020026	0	0	…	0.440688	0.106440	LDBUG
I_{400}	0.010005	0.017328	0.024506	…	0	0.159446	LDBUG

Table 3. Testing feature vectors

	F_1	F_2	F_3	…	F_{3999}	F_{4000}	Label
I_1	0.086084	0.026476	0.026476	…	0	0	
I_2	0	0.088023	0.191645	…	0	0	
I_3	0.011362	0.039360	0.093003	…	0.017302	0.028255	
I_4	0.415988	0.009997	0.009997		0	0.295634	
…	…	…	…	…	…	…	
I_{198}	0.185016	0	0	…	0.102334	0.053099	
I_{199}	0.407395	0	0	…	0.145924	0.387371	
I_{200}	0.010005	0	0.015688	…	0.048343	0.016114	

5.2 Tool Used

The experimental tool is installed in Matlab [22] and VLFeat 0.9.20 [33]. BOF model is used to represent insect image as feature vectors. These feature vectors are imported into R tools [25, 32] with H2O Deep Learning package [3] for classification. Experiments are operated on the computer with the configurations: Intel Core i7-4710HQ, CPU 2.50 GHz, Memory 16 GB RAM, Ubuntu 16.04 LTS operating system.

5.3 Scenario 1: Insect Classification in CDNN Model with 2 Hidden Layers

In this scenario, we evaluate CDNN model with 2 hidden layers. A number of neurons are customized in specific cases.

Case 1. Each hidden layer has 10 neurons (10, 10). Figure 7 illustrates number of BPH and ladybugs after classifying with the accuracy 91.5%.

```
> summary(predictions, exact_quantiles=TRUE)
 predict     BPH                  LDBUG
 BPH  :117   Min.    :5.392e-08   Min.    :1.625e-07
 LDBUG: 83   1st Qu.:1.313e-06   1st Qu.:2.133e-06
             Median :9.999e-01    Median :1.370e-04
             Mean    :5.432e-01   Mean    :4.568e-01
             3rd Qu.:1.000e+00    3rd Qu.:1.000e+00
             Max.    :1.000e+00   Max.    :1.000e+00
```

Fig. 7. Summary of classification in case 1:2 hidden layers (10, 10).

Case 2. A number of neurons in each hidden layer are adjusted, each hidden layer has 20 neurons (20, 20). Figure 8 illustrates number of BPHs and ladybugs after classifying with the accuracy 93%.

```
> summary(predictions, exact_quantiles=TRUE)
 predict     BPH                  LDBUG
 BPH  :114   Min.    :1.381e-08   Min.    :3.845e-08
 LDBUG: 86   1st Qu.:5.690e-07   1st Qu.:7.520e-07
             Median :9.991e-01    Median :8.938e-04
             Mean    :5.205e-01   Mean    :4.795e-01
             3rd Qu.:1.000e+00    3rd Qu.:1.000e+00
             Max.    :1.000e+00   Max.    :1.000e+00
```

Fig. 8. Summary of classification in case 2:2 hidden layers (20, 20).

Continuing experiments by adjusting the number nodes in 2 hidden layers of CDNN model. Table 4 shows the results of execution time and Mean squared errors (MSE) in different neural networks, predictable result in 2 classes (BPHs and Ladybugs) based on insect feature vectors, as well as the percentage of classification accuracy and the error rate.

Table 4. Summary of classification in Scenario 1: CDNN model with 2 hidden layers

Id	Number of nodes	Execution time (sec)	Mean squared errors (MSE)	Result PBHs; Ladybugs	Accuracy (%)	Error (%)
1	10, 10	3.437	1.304558e−09	117; 83	91.5	8.5
2	20, 20	3.568	1.111752e−10	114; 86	93.0	7.0
3	40, 40	4.631	2.224379e−11	114; 86	93.0	7.0
4	80, 80	6.224	7.733223e−12	111; 89	94.5	5.5
5	100, 100	10.276	2.35158e−12	109; 91	95.5	4.5

The result in Table 4 concludes that, in the Deep neural network with 2 hidden layers, the more the number of nodes in a layer is, the more the accuracy increases (but the error decreases).

5.4 Scenario 2: Insect Classification in CDNN Model with 3 Hidden Layers

In this scenario, we evaluate CDNN model with 3 hidden layers. A number of nodes in hidden layers are also adjusted in specific cases. Table 5 shows the summary of classification in Deep neural network with 3 hidden layers.

Table 5. Summary of classification in Scenario 2: CDNN Model with 3 Hidden Layers

Id	Number of nodes	Execution time (sec)	Mean squared errors (MSE)	Result PBHs; Ladybugs	Accuracy (%)	Error (%)
1	50, 50, 50	17.554	0.04597361	114; 86	93.0	7.0
2	100, 50, 50	22.214	0.03883661	106; 94	97.0	3.0
3	100, 100, 50	28.162	0.02325019	106; 94	97.0	3.0
4	100 ,100, 100	28.312	0.02747581	107; 93	96.5	3.5
5	200, 100, 100	58.002	0.03568920	109; 91	95.5	4.5

The result in Table 5 shows that when adjustment the number of nodes in the Deep neural network with 3 hidden layers, the rate of accuracy classification increases. The best result (accuracy rate 97% with smallest MSE) achieves with the number of nodes (100, 100, 50). However, when the number of nodes in 3 hidden layers adjusts to (200, 100, 100), the accuracy classification rate decreases (95.5%). This problem is called overfitting [24], a large number of nodes in hidden layers affects classification results.

5.5 Discussion

In comparing the accuracy, Fig. 9 illustrates the accuracy prediction (percentage) of the classification in different networks. In Deep neural network with 2 hidden layers, the accuracy increases corresponding to the increment of number of nodes. Similarly, in 3 hidden layers network, there is a rise of accuracy when the number of nodes increases. However, when increasing the number of classes to (200,100,100), the accuracy prediction rate tends to decrease. There is a distinction between the accuracy of 2 hidden layers and 3 hidden layers network. Obviously, accuracies of 3 hidden layers network are better than those of 2 hidden layers ones since they require one more layer to train and classify feature vectors. In short, the accuracy of a network is ratio with the number of hidden layers as well as the number of nodes in a hidden layer (with the appropriate number of nodes).

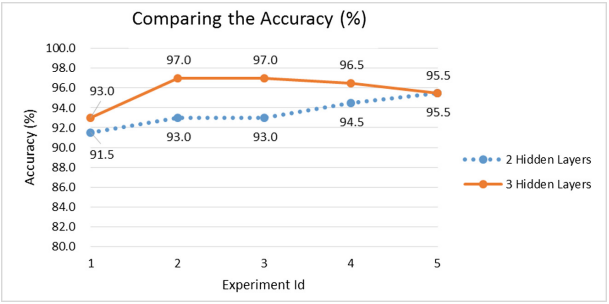


Fig. 9. Comparing the accuracy (%)

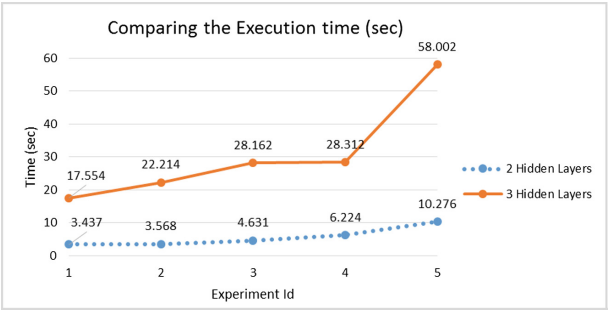


Fig. 10. Comparing the execution time (sec)

In comparing the execution time, Fig. 10 demonstrates convincingly that when the number of nodes in the hidden layer increases, the execution time increases as well. Further more, in all cases of experiment, execution times in 3 hidden layers network are longer than those in 2 hidden layers network.

6 Conclusion

We have advocated proposed the CDNN model to classify insect images using Bag of features and Deep Neural Networks approach. Insect image features are extracted and saved to descriptors matrix based on Dense SIFT. Vector quantization is done with a variant of the K-means algorithm, each cluster is a vocabulary of visual words. A spatial histogram is built with spatial pyramid matching, feature vector is the result at the end of this process. These feature vectors become the input of CDNN model which is applied for classifying insect image. The accuracy of the classification process can be increased by adjusting the number of nodes in a layer as well as the number of hidden layers in a network. Experiments show that the model is suitable for classification in static insect images. The best result (accuracy rate 97% with smallest MSE) achieves in case of using CDNN model 3 hidden layers with the number of nodes (100, 100, 50).

We believe that there would be tremendous benefit to an insect identification application and hope that this work will provide a starting point for further work on such a technology. It is intended that the proposed method will serve as a corner stone for research into real-time monitoring and tracking insects or other living organisms with the participation of experts in the field of information technology and agriculture.

References

1. Chavez, A.J.: Image classification with dense sift sampling an exploration of optimal parameters. Kansas State University, Manhattan (2012)
2. Tran, A.C., Tran, N.C., Huynh, H.X.: An approach to detecting brown plant hopper based on morphological operations. In: Vinh, P.C., Barolli, L. (eds.) ICTCC 2016. LNICST, vol. 168, pp. 52–61. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46909-6_6
3. Candel, A., LeDell, E., Parmar, V., Arora, A.: Deep Learning with H_2O, 5th edn. H2O.ai (2016)
4. Gersho, A., Gray, R.M.: Vector Quantization and Signal Compression. The Springer International Series in Engineering and Computer Science. Springer, Berlin (1992). https://doi.org/10.1007/978-1-4615-3626-0
5. Lu, A., Hou, X., Liu, C.-L., Chen, X.: Insect species recognition using discriminative local soft coding. In: Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012), pp. 1221–1224. IEEE, Tsukuba (2012)
6. Vedaldi, A., Fulkerson, B.: VLFeat: an open and portable library of computer vision algorithms. In: Proceedings of the 18th ACM International Conference on Multimedia, pp. 1469–1472. ACM, New York and Firenze (2010)
7. Lam, B.H., Van Tran, H., Huynh, H.X., Pottier, B.: Synchoronous networks for insects surveillance. In: Proceedings of the Sixth International Symposium on Information and Communication Technology, pp. 163–170, Hue City (2015)
8. Shepard, B.M., Barrion, A.T., Litsinger, J.A.: Friends of the rice farmer: helpful insects, spiders, and pathogens. International Rice Research Institute (1987)
9. Elkan, C.: Using the triangle inequality to accelerate k-means. In: Proceedings of the Twentieth International Conference on Machine Learning (ICML-2003), pp. 147–153. AAAI Press, Washington, DC (2003)
10. Szegedy, C., Toshev, A., Erhan, D.: Deep neural networks for object detection. In: Neural Information Processing Systems Conference (2013)
11. Christopher, M.: Bishop: Pattern Recognition and Machine Learning. Springer, New York (2006)
12. Xie, C., et al.: Automatic classification for field crop insects via multiple-task sparse representation and multiple-kernel learning. Comput. Electron. Agric. **119**, 123–132 (2015)
13. David, G.: Lowe: distinctive image features from scale-invariant keypoints. Int. J. Comput. Vis. **60**, 91–110 (2004)
14. Lowe, D.G.: object recognition from local scale-invariant features. In: Proceedings of the 7th IEEE International Conference on Computer Vision, pp. 1150–1157. IEEE (1999)
15. Niu, F., Recht, B., Re, C., Wright, S.J.: Hogwild: a lock-free approach to parallelizing stochastic gradient descent. In: Advances in Neural Information Processing Systems, pp. 693–701 (2011)
16. Larochelle, H., Bengio, Y., Louradour, J., Lamblin, P.: Exploring strategies for training deep neural networks. J. Mach. Learn. Res. **10**, 1–40 (2009)

17. Lim, J., Cho, J., Nam, T., Kim, S.: Development of a classification algorithm for butterflies and ladybugs. In: TENCON 2006 - 2006 IEEE Region 10 Conference, pp. 51–63. IEEE, Hong Kong (2006)
18. Bentley, J.L.: Multidimensional binary search trees used for associative searching. Commun. ACM **18**, 509–517 (1975)
19. Sivic, J., Zisserman, A.: Video google: a textretrieval approach to object matching in videos. In: Proceedings Ninth IEEE International Conference on Computer Vision, pp. 1470–1477. IEEE, Nice (2003)
20. Du, K.-L., Swamy, M.N.S.: Neural Networks and Statistical Learning. Springer, London (2014). https://doi.org/10.1007/978-1-4471-5571-3
21. Zhu, L.Q., Zhang, Z.: Auto-classification of insect images based on color histogram and GLCM. In: Seventh International Conference on Fuzzy Systems and Knowledge Discovery, Yantai, Shandong, China, pp. 2589–2593 (2010)
22. MathWorks Homepage. http://www.mathworks.com. Accessed 25 June 2018
23. Pathak, M.D., Khan, Z.R.: Insects pests of rice. International Rice Research Institute (1994)
24. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. **15**, 1929–1958 (2014)
25. RStudio Homepage. https://www.rstudio.com. Accessed 25 June 2018
26. López-Sastre, R.J., Renes-Olalla, J., Gil-Jiménez, P., Maldonado-Bascón, S.: Visual word aggregation. In: Vitrià, J., Sanches, J.M., Hernández, M. (eds.) IbPRIA 2011. LNCS, vol. 6669, pp. 676–683. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-21257-4_84
27. Hassan, S.N.A., Rahman, N.S.A., Htike, Z., Win, S.L.: Advanced in automatic insect classification. Electr. Electron. Eng.: Int. J. **3**(2), 51–63 (2014)
28. Lazebnik, S., Schmid, C., Pomce, J.: Beyond bag of features: spatial pyramid matching for recognizing natural scene categories. In: 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 2169–2178. IEEE, New York (2006)
29. O'Hara, S., Draper, B.A.: Introduction to the bag of features paradigm for image classification and retrieval. arXiv:1101.3354v1 (2011)
30. Visa, S., Ramsay, B., Ralescu, A., Knaap, E.V.D.: Confusion matrix-based feature selection. In: Proceedings of The 22nd Midwest Artificial Intelligence and Cognitive Science Conference 2011, Cincinnati, Ohio, USA, pp. 120–127 (2011)
31. Vo, T., Tran, D., Ma, W.: Tensor decomposition of dense SIFT descriptors in object recognition. In: European Symposium on Artificial Neural Networks, Bruges, Belgium, vol. 1, pp. 319–324 (2014)
32. The R Project for Statistical Computing Homepage. https://www.r-project.org. Accessed 25 June 2018
33. VLFeat Homepage. http://www.vlfeat.org. Accessed 25 June 2018
34. LeCun, Y.: A theoretical framework for back-propagation. In: Proceeding of the 1988 Connectionist Model Summer School, pp. 21–28. Morgan Kaufmann, Pittsburg (1988)
35. Bengio, Y.: Deep learning of representations: looking forward. In: Dediu, A.-H., Martín-Vide, C., Mitkov, R., Truthe, B. (eds.) SLSP 2013. LNCS (LNAI), vol. 7978, pp. 1–37. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-39593-2_1
36. Harris, Z.S.: Distributional structure. Word **10**, 146–162 (1954)