Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2019) 27: 1636 – 1651
© TÜBİTAK
doi:10.3906/elk-1809-181

Plant disease and pest detection using deep learning-based features

Muammer TÜRKOĞLU[1,*], **Davut HANBAY**[2]
[1]Department of Computer Engineering, Faculty of Engineering and Architecture, Bingöl University, Bingöl, Turkey
[2]Department of Computer Engineering, Faculty of Engineering, İnönü University, Malatya, Turkey

Received: 25.09.2018 • **Accepted/Published Online:** 25.02.2019 • **Final Version:** 15.05.2019

Abstract: The timely and accurate diagnosis of plant diseases plays an important role in preventing the loss of productivity and loss or reduced quantity of agricultural products. In order to solve such problems, methods based on machine learning can be used. In recent years, deep learning, which is especially widely used in image processing, offers many new applications related to precision agriculture. In this study, we evaluated the performance results using different approaches of nine powerful architectures of deep neural networks for plant disease detection. Transfer learning and deep feature extraction methods are used, which adapt these deep learning models to the problem at hand. The utilized pretrained deep models are considered in the presented work for feature extraction and for further fine-tuning. The obtained features using deep feature extraction are then classified by support vector machine (SVM), extreme learning machine (ELM), and K-nearest neighbor (KNN) methods. The experiments are carried out using data consisting of real disease and pest images from Turkey. The accuracy, sensitivity, specificity, and F1-score are all calculated for performance evaluation. The evaluation results show that deep feature extraction and SVM/ELM classification produced better results than transfer learning. In addition, the fc6 layers of the AlexNet, VGG16, and VGG19 models produced better accuracy scores when compared to the other layers.

Key words: Plant disease and pest detection, convolutional neural networks, deep learning architectures, feature extraction, classifier methods

1. Introduction

In plants, disease indications usually occur on leaves, fruit, buds, and young branches. This situation causes fruit to be wasted (to drop) or be damaged. In addition, these diseases lead to the formation of new infections and the spread of the disease for reasons such as seasonal conditions. For this reason, it is very important to determine the disease in advance and to take the necessary precautions before it spreads to other trees. As a result, the fight against diseases and pests in plants is the single most important issue in agriculture [1–3].

There are a variety of diseases that affect plants, which can each cause economic, social, and ecological loss. In this context, a timely and accurate diagnosis of plant diseases plays an important role in preventing the loss of productivity and quantity of agricultural products. Detection of plant diseases is usually performed manually. Such processes are conducted by experts such as botanists and agricultural engineers, first by visual inspection and later in a laboratory environment. These traditional methods are often time-consuming and complex processes [1, 3]. For these reasons, it has become important to automatically identify diseases based on image processing and machine learning. Automatic plant disease diagnosis with visual inspection can be of great benefit to users who have little to no knowledge of the product they are cultivating.

*Correspondence: mturkoglu@bingol.edu.tr

There have been numerous studies in the literature with regards to the detection of plant diseases. In the past decade, these studies have often been conducted based on the classification process by using features such as color, shape, and texture [4–7, 9, 9, 10]. The major disadvantages of methods based on these properties are:

- Very low performance when used alone, although attempts have been made to improve performance through a combination of several methods.
- Requires the use of segmented methods, meaning that plants have to be separated from their roots in order to extract geometric and similar features.
- Applied using datasets that contain images that are difficult to obtain in real life.

For these reasons, such systems and methods cannot be used in real life. Recently, deep convolutional neural networks (CNNs) have provided significant gains in areas of visual object and machine learning. One of the biggest advantages of models and methods improved regarding these areas is that they can perform feature extraction without applying segmented methods. Therefore, these methods and models can be easily used in real applications.

CNNs used as basic deep learning tools have obtained significant success in several plant disease detection studies. Amara et al. [11] used the LeNet architecture as a convolutional neural network to classify image datasets, including being able to contrast healthy and diseased banana leaves. These were evaluated with deep models fine-tuned by transfer learning. Mohanty et al. [12] used a dataset consisting of 54,306 images of diseased and healthy plant leaves. They were evaluated for performance using AlexNet and GoogleNet models based on a deep CNN to identify 14 crop species and 26 diseases. Fujita et al. [13] developed a new practical plant disease detection system that contained seven disease types. They used classification systems based on CNNs, which showed an average of 82.3% accuracy under a fourfold cross-validation strategy. Sladojevic et al. [14] proposed a new approach in order to recognize 13 different types of plant diseases by using deep convolutional networks. Fuentes et al. [15] developed a robust deep-learning-based detector for real-time usage that recognized nine different tomato diseases and pests.

In this paper, a system based on deep CNNs is developed for the detection of plant diseases and pests as a classification problem. In this study, we use a dataset consisting of real plant disease and pest images from Turkey. First, we use this dataset for deep feature extraction based on deep learning architectures: AlexNet, VGG16, VGG19, GoogleNet, ResNet50, ResNet101, InceptionV3, InceptionResNetV2, and SqueezeNet. The deep features obtained from these deep models are classified by SVM, ELM, and KNN. Later, we fine-tune using deep learning models based on transfer learning. We remove the last three layers of the architecture and then add new layers to adopt the pretrained CNNs to solve the problem. Finally, we evaluate the performance results by using transfer learning and deep feature extraction methods. The main contributions of the proposed plant disease classification method are the following:

- Plant disease and pest classification using images is problematic for machine learning. In order to classify these images, various methods have been proposed in the literature. Today, deep learning methods are phenomena of machine learning, and various deep learning networks have been proposed in the literature for supervised learning. In this article, nine of the most used deep learning networks (AlexNet, GoogleNet, VGG16, VGG19, ResNet50, ResNet101, InceptionV3, Inception ResNetV2, and SqueezeNet) are examined for plant diseases and pests using transfer learning and deep feature extraction, and their results are discussed comprehensively.

- This work proposes a novel classification architecture using pretrained deep learning networks and traditional classifiers. The nine deep learning networks are also utilized for feature extraction. Variable sizes of features are extracted using these networks. The traditional classifiers, which are SVM, ELM, and KNN, are used in the classification phase with 10-fold cross-validation. The results clearly indicate that the proposed architecture has higher classification accuracy than transfer-learning-based networks.

- The proposed architecture has lower computational complexity than transfer-learning-based models. This situation is proved using execution time calculation.

- In the literature, studies have generally been performed using synthetic plant disease and pest images. In this study, we use a dataset consisting of 1965 real plant disease and pest images within eight clusters. The proposed method was tested on this dataset and the results clearly prove that it can be used in real-world applications.

The study is organized as follows. While the proposed method is given in detail in Section 2, the proposed architectures are given in Section 3. The experimental work and results are given in Section 4. Finally, in Section 5, the results are discussed, as well as a presentation of the contributions of the study.

2. Materials and methods

In this section, the theoretical background and related algorithms, dataset, and recommended method are detailed under appropriate subheadings.

2.1. Deep learning and pretrained CNN models

Deep learning is a form of machine learning technique that uses calculation models composed of multiple processing layers in order to learn the characteristics of the data [16]. As a result of significantly high achievements in areas such as classification and recognition using deep learning, interest in this subject has increased. Recently, these methods have been used in many areas such as speech recognition, visual object recognition, and object detection [17]. Although the first studies on deep learning were based on a very long history, the main reasons for its success in recent years are the creation of large data and the generation of faster computers with large scale memory [18].

This study evaluated performance by trying different approaches of the nine most powerful architectures of deep neural networks for the issue of plant disease identification. These deep learning architectures are trained on a subset of the ImageNet database. The AlexNet architecture is a deep learning algorithm consisting of a total of 25 layers with weights that can be trained on eight layers [19]. The GoogleNet architecture, based on a network-in-network approach, uses architecture modules that use multiple convolutions in parallel to extract various feature points [20, 21]. The VGG network proposed by the Oxford Visual Geometry Group (VGG) is a homogeneous architecture used to obtain better results in the ILSVRC-2014 competition [22]. The ResNet network was developed by He et al. to train networks with even larger depth [23]. This architecture, based on microarchitecture modules, differs from traditional consecutive network architectures such as VGGNet and AlexNet. The Inception network was proposed by Szegedy et al. as a type of convolutional neural network model [21]. This network consists of a large number of convolution and maximum pooling steps. In the last stage, it has a fully connected neural network. The InceptionResNetV2 network [24] was based on the Inception-based network structure and residual connections. InceptionResNetV2 performs nearly identically to Inception architectures, but this architecture achieved significant acceleration in training using residual connections [25]. SqueezeNet was developed by Landola et al. as a smart architecture that achieves AlexNet-level accuracy on ImageNet with 50 times fewer parameters [26]. The characteristics of these architectures are detailed in Table 1.

Table 1. Details of the utilized architectures.

Network	Depth	Size	Parameter (millions)	Image input size
AlexNet	8	227 MB	61	227 × 227
VGG16	16	515 MB	138	224 × 224
VGG19	19	535 MB	144	224 × 224
SqueezeNet	18	4.6 MB	1.24	227 × 227
GoogleNet	22	27 MB	7	224 × 224
Inceptionv3	48	89 MB	23.9	299 × 299
InceptionResNetv2	164	209 MB	55.9	299 × 299
ResNet50	50	96 MB	25.6	224 × 224
ResNet101	101	167 MB	44.6	224 × 224

2.2. Classifiers

In this paper, traditional classifier methods of SVM, ELM, and KNN are used to construct the classification model of deep features extracted from specific layers of pretrained deep learning models.

2.2.1. K-nearest neighbor (KNN)

The KNN classifier uses a supervised learning method. This method is widely used in fields such as machine learning, image processing, and statistical estimation. This method is an algorithm that makes a classification of the existing learning data when new data enter. The principle of this method is to assign new data input to the closest cluster in a previously established sample set. The distance between these two data points is calculated by using various distance functions. The most known functions are Euclidean distance, Minkowski distance, and Manhattan distance. In addition, the greatest advantage of this method is that its application is easy and understandable [27, 28].

2.2.2. Support vector machine (SVM)

SVM is a method developed by Vapnik and is based on statistical learning theory [29]. The SVM method aims to have a linear discriminant function with the largest marginal separating the classes from each other. The learning data closest to the hyperplane are called support vectors. SVM can classify sets of data as both linearly distinguishable and indistinguishable [30]. This classifier has been successfully applied in order to solve problems in many areas such as image and object recognition, voice recognition, fingerprint recognition, and handwriting recognition [29–31].

2.2.3. Extreme learning machine (ELM)

ELM is a learning algorithm for single hidden-layer feedforward networks (SLFNs) proposed by Huang et al. [32]. In ELM, the least squares method is used to calculate output weights while hidden layer weights are randomly generated [33]. ELM can be modeled as shown in Eq. (1);

$$\sum_{i=1}^{N} \beta_i g(\omega_i * x_j + b_i) = y_j, \qquad j = 1, 2, \cdots, M, \tag{1}$$

where $[x_i, y_i]$ is the input-output, M is the number of training samples, w_i is the input weight, and b_i is the bias of the hidden layer. If the network output assumes that the actual values converge with 0 error, $Y = H\beta$ can be written in matrix form. Y represents the output of ELM. The H hidden layer output matrix and β output weight are calculated by Eq. (2):

$$H = \begin{bmatrix} g(w_1.x_1 + b_1) & \cdots & g(w_N.x_1 + b_N) \\ \cdots & \cdots & \cdots \\ g(w_1.x_M + b_1) & \cdots & g(w_N.x_M + b_N) \end{bmatrix}, \qquad \beta = \beta_1, \beta_2, \cdots, \beta_M. \tag{2}$$

The β parameter is obtained by $\beta = H^{\dagger}Y$. The Moore–Penrose generalized inverse of H is shown with $H^{\dagger}$ [32–34].

2.3. Dataset

The dataset used to test the performance of the proposed method includes images of plant diseases common to the Malatya, Bingöl, and Elazığ regions of Turkey (Figure 1). These images were obtained with a Nikon 7200d camera. Each image in this dataset consists of 4000 × 6000 resolution and three-channel (RGB) color images. Table 2 lists the names and numbers of plant diseases and pests in this dataset.

Table 2. Details of the utilized dataset for detection of plant diseases and pests.

Label	Name	Train	Test	Total
1	Coryneum beijerinckii (disease)	545	61	606
2	Apricot monilia laxa (disease)	76	9	85
3	Walnut leaf mite ga (pest)	62	7	69
4	Peach sphaerolecanium prunastri (pest)	310	35	345
5	Peach monilia laxa (disease)	278	31	309
6	Cherry myzus cerasi (disease)	175	19	194
7	Xanthomonas arboricola (disease)	129	14	143
8	Erwinia amylovora (pest)	193	21	214

As can be seen in Table 2, the dataset includes a total of 1965 images belonging to eight different plant diseases. The dataset was obtained by academics in the field of plant protection in the Agricultural Faculty of Bingöl University and İnönü University, Turkey. These images were obtained during daytime on different days. In addition, images of the diseases were taken using a large number of different trees.

3. Proposed method

In this study, we applied deep feature extraction from various fully connected layers and transfer learning based on pretrained deep learning architectures. The scheme of the proposed study is illustrated in Figures 2 and 3. Transfer learning and deep feature extraction are detailed in the following subsections.

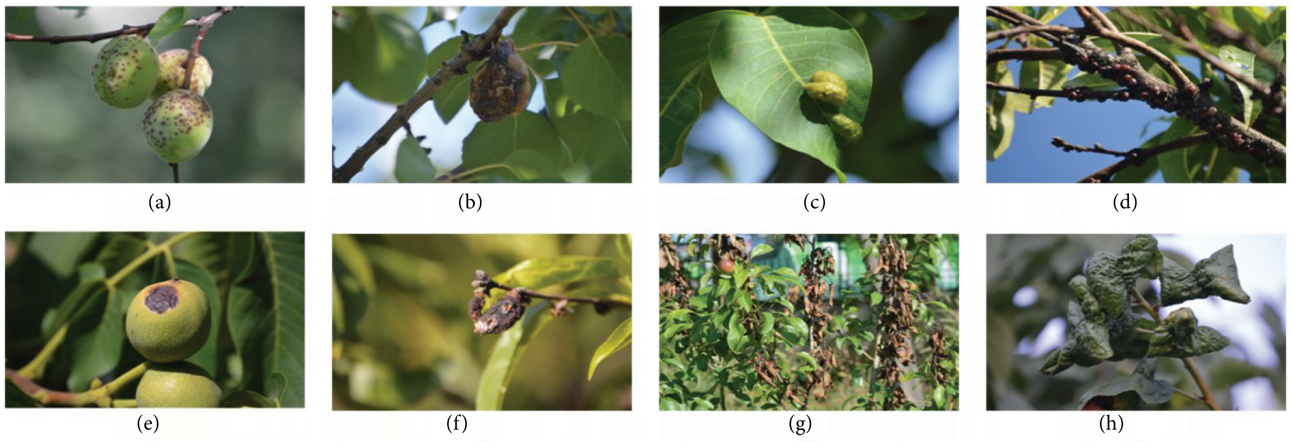

Figure 1. Samples of plant diseases and pests images used for experimentations: (a) Coryneum beijerinckii, (b) Apricot monilia laxa, (c) Walnut leaf mite ga, (d) Peach sphaerolecanium prunastri, (e) Xanthomonas arboricola, (f) Peach monilia laxa, (g) Erwinia amylovora, (h) Cherry myzus cerasi.

3.1. Transfer learning

Transfer learning is a machine learning method that is reused as a starting point for solving a different problem by using knowledge obtained from a model developed in the solution of a problem. The current study fine-tuned this by using pretrained CNN models based on transfer learning. The reason for using pretrained CNN models is that they are faster and easier than training a CNN model with randomly initialized weights [35]. In addition, the fine-tuning process is based on transferring new layers instead of the last three layers of the pretrained networks to our classification task, as shown in Figure 2.

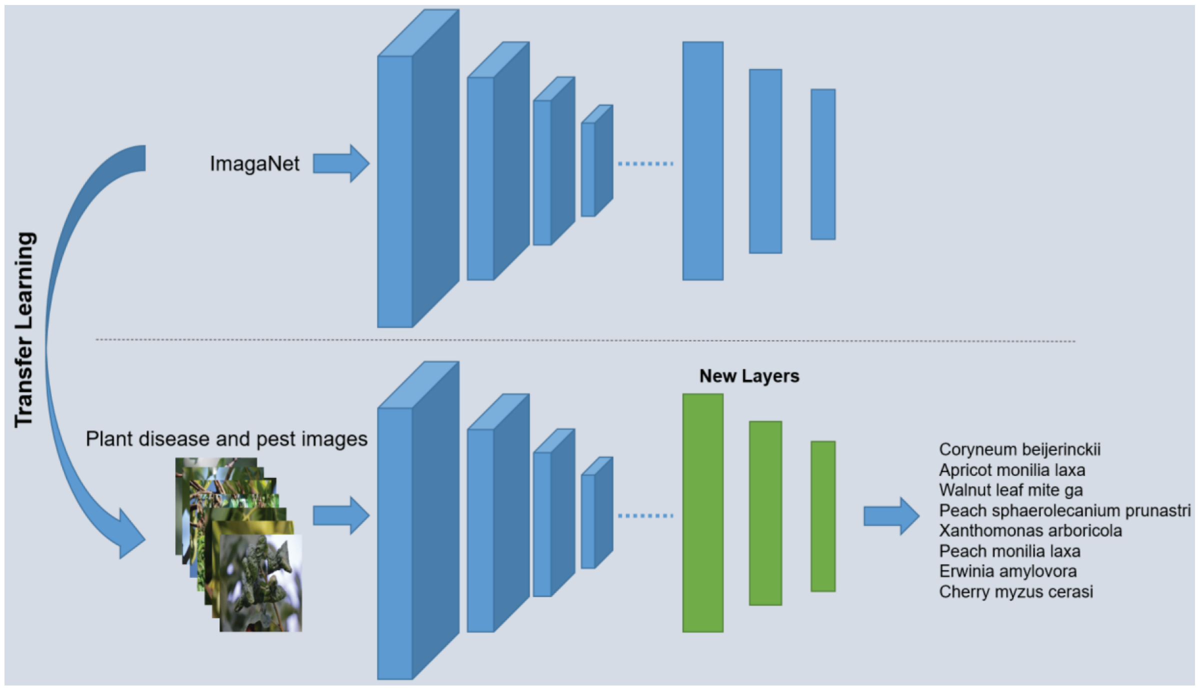

Figure 2. Simple representation of transfer learning.

3.2. Deep feature extraction

Deep feature extraction is based on extracting features learned from a pretrained convolutional neural network. These features are used to train machine learning classifiers. In other words, this method is based on the extraction of deep features from the fully connected layer of pretrained networks. In this study, effective deep features were extracted from a certain layer of deep learning models, which are GoogleNet, ResNet50, ResNet101, InceptionV3, InceptionResNetV2, and SqueezeNet and pool5-drop_7x7_s1, fc1000, fc1000, predictions, predictions, and pool10, respectively. In addition, feature vectors were obtained from the these layers of the GoogleNet, ResNet50, ResNet101, InceptionV3, InceptionResNetV2, and SqueezeNet models: 1024, 1000, 1000, 1000, 1000, and 4096, respectively. The obtained deep features are employed in the classification phase by using the traditional classifiers of SVM, ELM, and KNN, as shown in Figure 3.

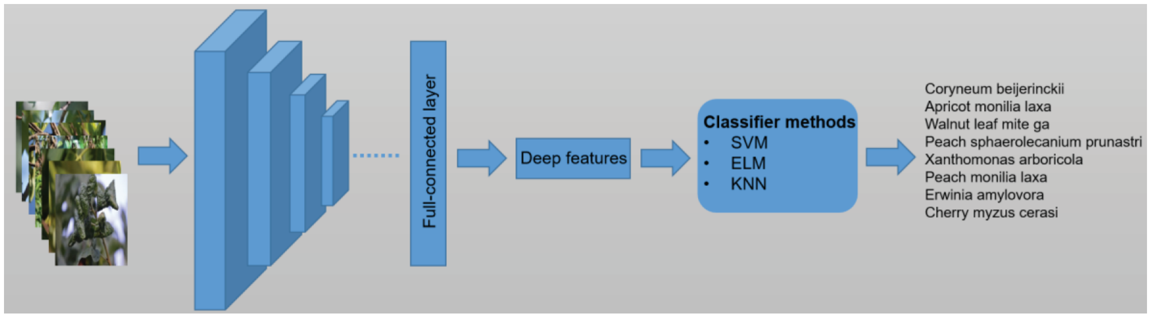

Figure 3. A simple representation for deep feature extraction and traditional classifiers.

In this study, deep features obtained from these three layers for each of the AlexNet, VGG16, and VGG19 models are used in the classification phase by using the traditional classifiers of SVM, ELM, and KNN in order to calculate performances by using classifier methods. Then the performances of these layers are compared, and the deep features obtained from the best layer are determined. In addition, while 4096 feature vectors are obtained from the fc6 and fc7 layers of the AlexNet, VGG16, and VGG19 models, 1000 feature vectors are obtained from the fc8 layer of these models.

3.3. Summary steps of proposed architecture and transfer learning

The following steps summarize the proposed deep feature extraction:
Step 1: Acquire plant images.
Step 2: Resize plant image according to deep networks using bilinear interpolation. For instance, color images sized 224×224 and 227×227 are used in ResNet50 and AlexNet, respectively.
Step 3: Features are extracted using the fully connected layers of the deep learning models.
Step 4: Classification is performed using the deep features with the SVM, KNN, and ELM classifiers.

The following steps summarize the transfer learning:
Step 1: Acquire plant images.
Step 2: Resize plant image according to deep networks using bilinear interpolation.
Step 3: Last three layers are removed from pretrained deep models and replaced by a fully connected layer, a softmax layer, and a classification output layer in order to adopt the pretrained CNN networks to solve the problem.
Step 4: Classification is performed using the newly created deep model.

4. Experimental studies

In this study, we evaluated the performance using different approaches of nine powerful architectures of deep neural networks for plant disease detection. The experimental studies were implemented using the MATLAB deep learning toolbox. All applications were run on a server with a 2xCore Intel Xeon E5 with 64 GB memory. The server was equipped with NVIDIA Quadro M4000 and 8 GB memory. We used the SVM, ELM, and KNN methods in order to test the classification accuracy of the deep features. The study used a one-versus-all approach and a cubic SVM as the classifier type for the SVM classifier parameters. In addition, we used a sigmoid function as an activation function and a hidden layer neuron number of 10,000 for the ELM classifier parameters. Finally, we used the Euclidean distance function and 1NN-3NN value of the KNN algorithm for the KNN classifier parameters.

In order to test the performance of the proposed method, we used a dataset containing our own plant disease images. This dataset consists of a total of 1965 images belonging to eight different plant diseases. The numerical split for training and testing for each plant species is given in Table 2. To calculate the performance of the proposed methods, a 10-fold cross-validation test was applied. In addition, we used performance measures of accuracy, sensitivity, specificity, and F1-score for classification performances of experimental studies. The experimental results and performance comparisons are detailed under the following subheadings.

4.1. Results based on deep feature extraction with AlexNet, VGG16, and VGG19 models

In this section, we use three different fully connected layers for deep feature extraction based on pretrained AlexNet, VGG16, and VGG19 models. For each of these models, deep features were extracted from the fc6, fc7, and fc8 layers. Then these features were calculated for their performance using the SVM, ELM, and KNN methods. The accuracy scores of these experimental studies are given in Table 3. In addition, these accuracy scores were evaluated according to the calculated average accuracy score across the folds and their standard deviations.

Table 3. Accuracy scores (%) and execution time for different training approaches based on deep feature extraction (bold font shows the best accuracy scores for each model).

Classifier	AlexNet			Vgg16			Vgg19		
methods	fc6	fc7	fc8	fc6	fc7	fc8	fc6	fc7	fc8
SVM	**95.5±1.2**	94.1 ± 0.65	92.2 ± 0.92	**95.0±1.4**	93.7 ± 1.21	92.5 ± 0.7	94.45 ± 0.8	93.99 ± 0.74	91.76 ± 1.26
ELM	93.41 ± 0.72	92.29 ± 0.5	91.53 ± 0.96	94.84 ± 1.12	93.57 ± 0.62	92.11 ± 0.8	**94.74±1.14**	94.38 ± 0.81	93.01 ± 0.54
KNN	89.41 ± 1.8	89.01 ± 1.41	84.5 ± 1.02	88.55 ± 1.49	87.0 ± 1.11	84.7 ± 0.82	89.01 ± 1.52	87.94 ± 1.38	85.5 ± 0.98
Time	31 s	32 s	34 s	1 min 6 s	1 min 7 s	1 min 9 se	1 min 15 s	1 min 16 s	1 min 19 s

As shown in Table 3, the deep features obtained from the fc6, fc7, and fc8 layers for each of the AlexNet, VGG16, and VGG19 models were tested for their performance using the SVM, ELM, and KNN methods. According to the obtained accuracy scores, the fc6 layer was found to have the best deep features for these models. The highest accuracy level for the AlexNet model was 95.5% using the SVM classifier. While the highest accuracy for the VGG16 model was 95% using the SVM classifier, the highest accuracy for the VGG19 model was 94.74% using the ELM classifier. In addition, in Table 3 the execution times for each of the fully connected layers of the AlexNet, VGG16, and VGG19 models are given. According to these results, for all fully connected layers, AlexNet had the lowest training time among the three models, while the training times for VGG16 and VGG19 were almost equal. In addition, the performance measures of sensitivity, specificity, and F1-score related to these experimental studies are given in Table 4.

Table 4. Performance measures (%) for different training approaches based on deep feature extraction.

Classifier methods	Performance measures	AlexNet			VGG16			VGG19		
		fc6	fc7	fc8	fc6	fc7	fc8	fc6	fc7	fc8
SVM	Sensitivity	94.34	93.03	89.61	93.02	91.91	89.96	92.22	92.41	89.34
	Specificity	99.35	99.22	98.78	99.31	99.12	98.88	99.19	99.13	98.79
	F1-score	94.58	93.49	90.02	93.19	91.69	89.87	92.47	92.44	89.10
ELM	Sensitivity	90.75	89.44	88.20	92.12	91.51	89.85	92.67	91.87	89.98
	Specificity	99.00	98.86	98.70	99.22	99.08	98.87	99.23	99.12	98.97
	F1-score	91.29	90.23	89.30	92.46	91.99	90.36	93.06	91.93	90.65
KNN	Sensitivity	87.94	87.07	76.31	86.56	83.87	82.61	87.28	85.43	82.88
	Specificity	98.38	98.35	97.38	98.35	98.01	97.75	98.40	98.23	97.85
	F1-score	87.35	86.33	78.03	85.46	82.69	80.94	86.33	84.87	82.03

4.2. Results based on deep feature extraction with other deep learning models

In this section, we used the fully connected layer for deep feature extraction, based on pretrained CNN models of GoogleNet, ResNet50, ResNet101, InceptionV3, InceptionResNetV2, and SqueezeNet. For each of these models, deep features were extracted from the layers specified in Section 3. Then these features were calculated for their performance using the SVM, ELM, and KNN methods. The accuracy scores of these experimental studies are given in Table 5. In addition, these accuracy scores were evaluated according to the calculated average accuracy score across the folds and their standard deviations.

Table 5. Accuracy scores (%) and execution time for other deep learning models based on deep feature extraction (bold font shows the best accuracy scores for each model).

Classifier methods	GoogleNet	ResNet50	Resnet101	Inceptionv3	InceptionResNetv2	SqueezeNet
SVM	**95.22±1.24**	**97.86±1.56**	96.74 ± 0.64	**94.96±1.22**	94.76 ± 1.04	**95.62±1.08**
ELM	94.18 ± 0.96	97.65 ± 1.34	**97.45±0.78**	94.54 ± 0.87	**95.20±1.16**	95.10 ± 0.94
KNN	89.16 ± 2.06	90.48 ± 2.24	91.20 ± 1.41	88.65 ± 1.52	88.80 ± 2.2	87.02 ± 3.82
Time	1 min 41 s	2 min 21 s	4 min 11 s	9 min 29 s	28 min 13 s	1 min 3 s

As shown in Table 5, the deep features obtained from the specific layers of the GoogleNet, ResNet50, ResNet101, InceptionV3, InceptionResNetV2, and SqueezeNet models were tested for their performance using the SVM, ELM, and KNN methods. The ResNet50 model and SVM classifier produced the highest accuracy among these other deep models with an accuracy of 97.86%. The confusion matrix of this model is shown in Table 6. When classified with the SVM method, the highest accuracy scores were 95.22% for the GoogleNet model, 94.96% for the InceptionV3 model, and 95.62% for the SqueezeNet model. Using the ELM classifier, the highest accuracy scores were 97.34% for the ResNet101 model and 95.35% for the InceptionResNetV2 model.

The execution time for each of these deep models is given in Table 5. According to these results, SqueezeNet (1 min, 3 s) showed the lowest training time among the three models, while InceptionResNetV2 (28 min, 13 s) had the highest training time. Comparatively, the GoogleNet model (1 min, 41 s) was slightly slower than the SqueezeNet model (1 min, 3 s); the ResNet101 model (4 min, 11 s) was almost twice as slow as the

Table 6. Confusion matrix based on ResNet50 model and SVM classifier (Legend: 1. Erwinia amylovora; 2. Xanthomonas arboricola; 3. Walnut leaf mite ga; 4. Apricot monilia laxa; 5. Coryneum beijerinckii; 6. Cherry myzus cerasi; 7. Peach sphaerolecanium prunastri; 8. Peach monilia laxa).

	1	2	3	4	5	6	7	8
1	**204**	1			2			6
2		**138**	4			1		
3		1	**68**					
4				**83**			1	1
5					**606**			
6	1	3				**186**		4
7			1				**343**	1
8	2	1	1	2	1	2	5	**295**

ResNet50 model (2 min, 21 s) in training time; and the InceptionV3 model (9 min, 29 s) was four times slower than the ResNet50 model (2 min, 21 s). In addition, the performance measures of sensitivity, specificity, and F1-score related to these experimental studies are given in Table 7.

Table 7. Performance measures (%) for other deep learning models based on deep feature extraction.

Classifier methods	Performance measures	GoogleNet	ResNet50	Resnet101	Inceptionv3	InceptionResNetv2	SqueezeNet
SVM	Sensitivity	94.21	97.35	95.02	92.66	91.94	93.91
	Specificity	99.30	99.69	99.53	99.25	99.24	99.36
	F1-score	94.13	97.14	94.92	93.20	92.13	93.83
ELM	Sensitivity	92.09	96.60	96.21	91.39	92.54	93.12
	Specificity	99.14	99.66	99.63	99.18	99.28	99.26
	F1-score	92.57	96.70	96.26	92.31	93.17	93.51
KNN	Sensitivity	86.12	88.80	89.34	84.84	84.52	83.78
	Specificity	98.40	98.59	98.71	98.25	98.33	98.10
	F1-score	86.02	88.01	88.40	85.65	84.39	83.19

In this study, the utilized dataset is heterogeneous. Accordingly, the performance measures of accuracy, sensitivity, specificity, and F1-score were calculated for each class and the obtained results using the ResNet50 model and SVM classifier are shown Figure 4.

4.3. Results based on transfer learning with deep learning models

In this section, we performed fine-tuning based on transfer learning using deep learning models from pretrained CNN networks. For transfer learning parameters, the batch size was chosen as 10, the maximum epoch number was set to 7, and the initial learning rate was assigned as 0.0004. In addition, training of the deep networks was achieved with the stochastic gradient descent with momentum (SGDM) learning method. The training procedure ended after 1250 iterations. The accuracy scores of these experimental studies are given in Table 8 and these scores were evaluated according to the calculated average accuracy score across the folds and their standard deviations.

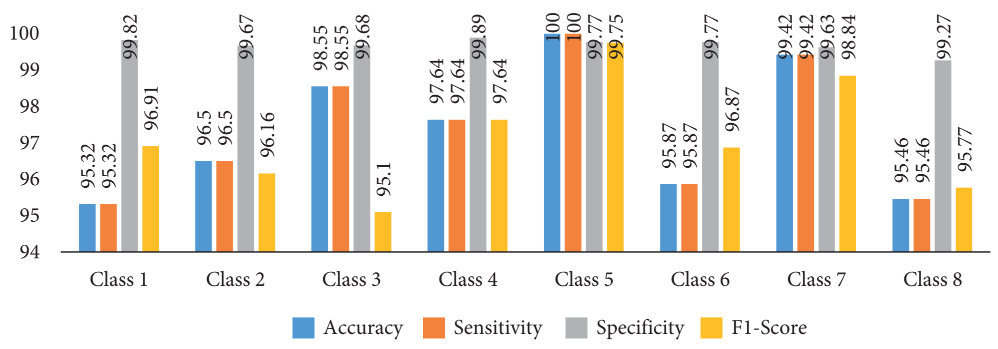

Figure 4. Performance results for each class of ResNet50 model and SVM classifier.

Table 8. Accuracy scores (%) and execution time for deep models based on transfer learning.

	AlexNet	VGG16	VGG19	GoogleNet	ResNet50	Resnet101	Inceptionv3	InceptionResNetv2	SqueezeNet
Accuracy	93.33 ± 2.41	96.92 ± 1.26	94.87 ± 1.3	95.3 ± 1.02	95.38 ± 0.9	96.41 ± 0.86	92.31 ± 2.12	89.23 ± 3.46	91.28 ± 1.96
Time	5 min 48 s	15 min 26 s	42 min 53 s	11 min 30 s	20 min 49 s	31 min 30 s	40 min 39 s	111 min 21 s	8 min 33 s

As shown in Table 8, the VGG16 model showed the best accuracy score of 96.92% among the other deep models, while the InceptionResNetV2 model was the worst at 89.23%. While the second best accuracy score was produced as 96.41% for the ResNet101 model, the third best accuracy score was 95.38% using the GoogleNet and ResNet50 models. In addition, Table 8 shows the execution time of the utilized deep models. According to these results, AlexNet had the fastest execution time (5 min, 48 s) among the three models, while InceptionResNetV2 had the slowest execution time (1 h, 51 min, 21 s). Comparatively, the SqueezeNet model had a faster execution time (8 min, 33 s) than the GoogleNet (11 min, 30 s) and VGG16 (15 min, 26 s) models. In addition, the ResNet50 model (20 min, 49 s) was almost twice as quick as the VGG19 model (42 min, 53 s) in execution time, and one-third quicker than the ResNet101 model (31 min, 30 s). In addition, the performance measures of sensitivity, specificity, and F1-score related to these experimental studies are given in Table 9.

Table 9. Performance measures (%) for deep models based on transfer learning.

Performance measures	AlexNet	VGG16	VGG19	GoogleNet	ResNet50	ResNet101	Inceptionv3	InceptionResNetv2	SqueezeNet
Sensitivity	88.02	96.64	92.24	92.29	94.91	95.81	87.05	85.09	89.22
Specificity	99.0	99.54	99.14	95.75	99.36	99.43	98.90	98.48	98.70
F1-score	88.30	96.38	93.87	93.53	93.28	95.94	88.76	85.15	88.33

4.4. Results based on traditional methods

In this section, traditional methods of LBP, HOG, GLCM, and color features, which are widely used for plant disease detection systems, have been applied. These methods extracted features directly from the images without the requirement for a segmented operation. Properties obtained from these methods were calculated for performance using the SVM, ELM, and KNN methods. The accuracy scores related to these studies are given in Table 10.

As seen in Table 10, the best performances obtained using the SVM classifier were 70.9% for the LBP method, 56.8% for the HOG method, 45.3% for color features, and 45.2% for the GLCM method.

Table 10. Accuracy scores (%) for traditional methods (bold font shows the best accuracy scores for each method).

Classifier methods	LBP	HOG	Color features	GLCM
SVM	**70.9**	**56.8**	**45.3**	**45.2**
ELM	68.26	53.36	43.57	42.85
KNN	61.4	46.5	35.9	38.5

4.5. Discussion and comparison of simulation results

The proposed study used pretrained CNN models to obtain the best performance in the detection of plant diseases. We evaluated the performance results of transfer learning and deep feature extraction based on the AlexNet, VGG16, VGG19, GoogleNet, ResNet50, ResNet101, InceptionV3, InceptionResNetV2, and SqueezeNet deep models. In addition, traditional methods of LBP, HOG, color features, and GLCM, which are widely used in object recognition, were applied and their accuracy scores were evaluated.

First, we extracted features based on deep feature extraction by using the AlexNet, VGG16, and VGG19 models. For each of these models we separately extracted features from the three fully connected layers of fc6, fc7, and fc8. The obtained deep features were calculated for their performance using SVM, ELM, and KNN classifiers. According to the results, the best performance for these three models was obtained using deep features extracted from the fc6 layer. For the AlexNet model, using the features from fc6, the highest classification accuracy obtained was 95.5% with SVM and the lowest classification accuracy obtained was 89.41% with KNN. For the VGG16 model, the highest classification accuracy obtained was 95.0% with SVM using the features from fc6 and the lowest classification accuracy obtained was 88.55% with KNN. For the VGG19 model, the highest classification accuracy obtained was 94.74% with ELM using the features from fc6 and the lowest classification accuracy obtained was 88.01% with KNN. As a result, the features extracted from fc6 of the AlexNet, VGG16, and VGG19 models produced better accuracy scores than the features extracted from either fc7 or fc8. In addition, both the SVM and ELM methods achieved much better results than the KNN classifiers.

Next, we extracted features from a specific layer using the GoogleNet, ResNet50, ResNet101, InceptionV3, InceptionResNetV2, and SqueezeNet deep learning models. The obtained deep features were calculated for their performance using SVM, ELM, and KNN classifiers. According to the results, the GoogleNet model's highest classification accuracy was 95.22% with SVM and the lowest classification accuracy was 89.16% with KNN. For the ResNet50 model, the highest classification accuracy was 97.86% with SVM and the lowest classification accuracy was 90.48% with KNN. For the ResNet101 model, the highest classification accuracy was 97.45% with ELM and the lowest classification accuracy was 91.2% with KNN. For the InceptionV3 model, the highest classification accuracy was 94.96% with SVM and the lowest classification accuracy was 88.65% with KNN. For the InceptionResNetV2 model, the highest classification accuracy was 95.2% with ELM and the lowest classification accuracy was 88.8% with KNN. For the SqueezeNet model, the highest classification accuracy was 95.62% with ELM and the lowest classification accuracy was 87.02% with KNN. The GoogleNet, ResNet50, InceptionV3, and SqueezeNet models obtained the best performance with the SVM classifier while the ResNet101 and InceptionResNetV2 models obtained the best performance with the ELM classifier. In addition, the worst performance obtained for all the models was achieved using the KNN classifier.

Later, we performed fine-tuning based on transfer learning using pretrained CNN networks. According to these performance results, while the VGG16 model produced the highest accuracy among the deep models, the second best accuracy was achieved using the ResNet101 model. The accuracy score for the GoogleNet model was equal to that of the ResNet50 model. In addition, the VGG19 model had a better accuracy score than the InceptionV3, InceptionResNetV2, and SqueezeNet models.

Finally, traditional methods of LBP, HOG, GLCM, and color features were used, and the obtained feature values were calculated for their performance using SVM, ELM, and KNN classifiers. According to the results, among these four methods, the highest classification accuracy was 70.9% using the LBP method and SVM classifier. In addition, the best performance of these four methods was obtained using the SVM classifier and the lowest performance was achieved with KNN. The comparisons of accuracy scores based on deep feature extraction, transfer learning, and traditional methods are shown in Figure 5.

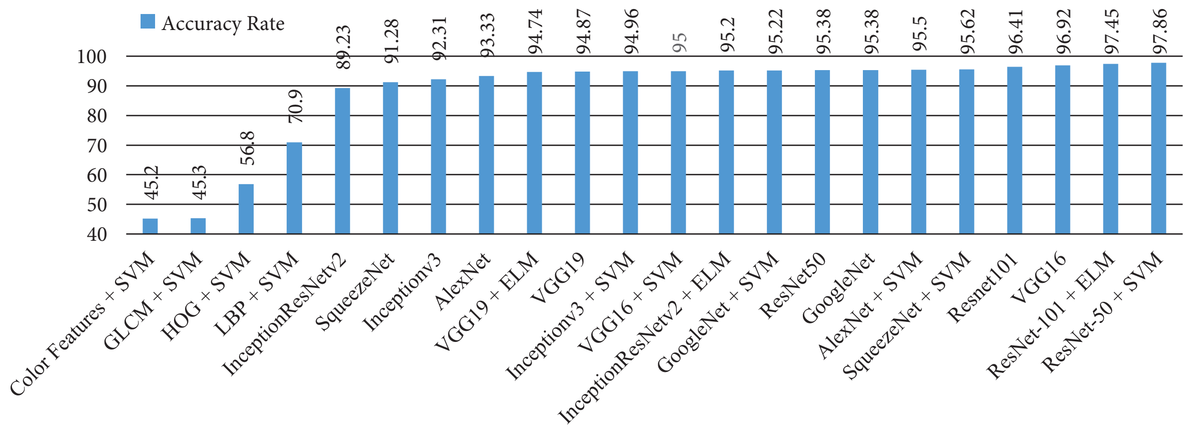

Figure 5. Comparison of accuracy scores of utilized method and models.

As shown in Figure 5, the highest accuracy score was 97.86% using the ResNet50 model and SVM classifier, while the second best accuracy score of 97.45% was produced using the ResNet101 model and SVM classifier. The best accuracy score based on transfer learning was 96.92% with the VGG16 model. In addition, conventional methods were found to have very poor levels of performance compared to the deep learning models. Among the traditional methods, the highest accuracy score of 70.9% was produced for the LBP method and SVM classifier. The accuracy scores based on deep feature extraction and transfer learning using deep models were as follows:

- ResNet50 features and the SVM classifier had a better accuracy score than the fine-tuned ResNet50 model based on transfer learning. ResNet101 features and the ELM classifier had a better accuracy score than the fine-tuned ResNet101 model. SqueezeNet features and the SVM classifier had a better accuracy score than the fine-tuned SqueezeNet model. AlexNet features and the SVM classifier had a better accuracy score than the fine-tuned AlexNet model. InceptionResNetV2 features and the ELM classifier had a better accuracy score than the fine-tuned InceptionResNetV2 model. InceptionV3 features and the SVM classifier had a better accuracy score than the fine-tuned InceptionV3 model.

- The fine-tuned VGG16 model had a better accuracy score than the VGG16 features and SVM classifier. The fine-tuned GoogleNet model had a better accuracy score than the GoogleNet features and SVM classifier. The fine-tuned VGG19 model had a better accuracy score than the VGG19 features and SVM classifier.

The evaluation results show that deep feature extraction and the SVM/ELM classifier obtained a higher classification accuracy than networks based on transfer learning. In addition, each of the utilized deep models has unique features such as number of layers, number of connections, and filter types. Therefore, performance results differed from each other. For example, the ResNet50 model has fewer layers and connections than InceptionResNetV2, but its performance was higher.

In this study, we calculated execution times for deep feature extraction and transfer learning based on deep learning models, as shown in Tables 5 and 8. According to the results, deep feature extraction has a much lower training time compared to transfer learning. For example, while the AlexNet deep feature extraction was almost ten times lower than the fine-tuned AlexNet in training time, the ResNet50-based deep feature extraction was almost five times lower than the fine-tuned ResNet50. In addition, the testing times for transfer learning and deep feature extraction based on deep models were mostly 10–30 s due to low test data, so there was no need for a separate table. As a result, although the proposed architecture based on deep feature extraction resulted in lower training times than transfer learning, it had higher accuracy scores.

5. Conclusion

This paper compared the performance results of deep feature extraction and transfer learning for the detection of plant diseases and pests. This study used nine powerful architectures of deep neural networks for both deep feature extraction and transfer learning. First, we extracted deep features for fully connected layers of these deep models. The obtained deep features had their performances calculated using SVM, ELM, and KNN classifiers. Then these deep models were fine-tuned based on plant disease and pest images. Finally, we compared the performance results with deep learning models using traditional methods. The evaluation results showed that the deep learning models produced better results compared to the traditional methods. The results for deep feature extraction were better than the transfer learning achievements. In the studies based on deep feature extraction and classifier methods, the fc6 layers of the AlexNet, VGG16, and VGG19 models produced better accuracy scores compared to the other layers. As a result, the highest level accuracy scored was 97.86%, obtained with the ResNet50 model and SVM classifier. In addition, the calculated execution time for deep feature extraction and transfer learning based on deep learning models and the proposed architecture based on deep feature extraction showed lower computational complexity than models based on transfer learning.

In future works, we are planning to use regional images and improved CNN models in order to improve the classification performance. Additionally, we will gather images of different diseases in order to enrich the database. The main goal for subsequent research will be developing a smart mobile device application that can detect various plant diseases. This application, which will provide automatic plant disease diagnosis with visual inspection, could be of great benefit to users with little to no knowledge of the plants that they are cultivating.

References

[1] Turkoglu M, Hanbay D. Apricot disease identification based on attributes obtained from deep learning algorithms. In: IEEE 2018 International Conference on Artificial Intelligence and Data Processing; Malatya, Turkey; 2018. pp. 1-4.

[2] Güzel M. The importance of good agricultural practices (GAP) in the context of quality practices in agriculture and a sample application. PhD, Dokuz Eylül University, İzmir, Turkey, 2012.

[3] Asma BM, Birhanlı O. Mişmiş. Malatya, Turkey: Evin Ofset, 2004 (in Turkish).

[4] Al-Hiary H, Bani-Ahmad S, Reyalat M, Braik M, Al-Rahamneh Z. Fast and accurate detection and classification of plant diseases. International Journal of Computer Applications 2011; 17 (1): 31-38.

[5] Bashir S, Sharma N. Remote area plant disease detection using image processing. IOSR Journal of Electronics and Communication Engineering 2012; 2 (6): 31-34.

[6] Kulkarni AH, Patil A. Applying image processing technique to detect plant diseases. International Journal of Modern Engineering Research 2012; 2 (5): 3661-3664.

[7] Arivazhagan S, Shebiah RN, Ananthi S, Varthini SV. Detection of unhealthy region of plant leaves and classification of plant leaf diseases using texture features. Agricultural Engineering International CIGR Journal 2013; 15 (1): 211-217.

[8] Khirade SD, Patil AB. Plant disease detection using image processing. In: IEEE International Conference on Computing Communication Control and Automation; Pune, India; 2015. pp. 768–771.

[9] Athanikar G, Badar P. Potato leaf diseases detection and classification system. International Journal of Computer Science and Mobile Computing 2016; 5: 76-78.

[10] Barbedo JGA, Koenigkan LV, Santos TT. Identifying multiple plant diseases using digital image processing. Biosystems Engineering 2016; 147: 104-116.

[11] Amara J, Bouaziz B, Algergawy A. A deep learning-based approach for banana leaf diseases classification. In: BTW Workshops; Bonn, Germany; 2017. pp. 79-88.

[12] Mohanty SP, Hughes DP, Salathé M. Using deep learning for image-based plant disease detection. Frontiers in Plant Science 2016; 7: 1419.

[13] Fujita E, Kawasaki Y, Uga H, Kagiwada S, Iyatomi H. Basic investigation on a robust and practical plant diagnostic system. In: Proceedings of the 15th IEEE International Conference on Machine Learning and Applications; Anaheim, CA, USA; 2016. pp. 989–992.

[14] Sladojevic S, Arsenovic M, Anderla A, Culibrk D, Stefanovic D. Deep neural networks based recognition of plant diseases by leaf image classification. Computational Intelligence and Neuroscience 2016; 2016: 3289801.

[15] Fuentes A, Yoon S, Kim SC, Park DS. A robust deep-learning-based detector for real-time tomato plant diseases and pests recognition. Sensors 2017; 17 (9): 2022.

[16] LeCun Y, Bengio Y, Hinton G. Deep learning. Nature 2015; 521 (7553): 436.

[17] Liew SS, Hani MK, Radzi SA, Bakhteri R. Gender classification: a convolutional neural network approach. Turkish Journal of Electrical Engineering & Computer Sciences 2016; 24 (3): 1248-1264.

[18] Inik O, Ulker E. Deep learning and deep learning models used in image analysis. Gaziosmanpasa Journal of Scientific Research 2017; 6(3): 85-104.

[19] Krizhevsk A, Sutskever I, Hinton GE. Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems; 2012 pp. 1097-1105.

[20] Ghazi MM, Yanikoglu B, Aptoula E. Plant identification using deep neural networks via optimization of transfer learning parameters. Neurocomputing 2017; 235: 228-235.

[21] Szegedy C, Liu W, Jia Y, Sermanet P, Reed S et al. Going deeper with convolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition; Boston, MA, USA; 2015. pp. 1-9.

[22] Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition. arXiv preprint, arXiv:1409.1556, 2014.

[23] He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition; Las Vegas, NV, USA; 2016. pp. 770-778.

[24] Szegedy C, Loffe S, Vanhoucke V, Alemi AA. Inception-v4, inception-resnet and the impact of residual connections on learning. In: Thirty-First AAAI Conference on Artificial Intelligence; San Francisco, CA, USA; 2017. pp. 4278-4284.

[25] Nguyen LD, Lin D, Lin Z, Cao J. Deep CNNs for microscopic image classification by exploiting transfer learning and feature concatenation. In: 2018 IEEE International Symposium on Circuits and Systems; Florence, Italy; 2018. pp. 1-5.

[26] Landola FN, Han S, Moskewicz MW, Ashraf K, Dally WJ, Keutzer K. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and 0.5 mb model size. arXiv preprint, arXiv:1602.07360, 2016.

[27] Sengur A, Akilotu BN, Tuncer SA, Kadiroglu Z, Yavuzkilic S, Budak U, Deniz E. Optic disc determination in retinal images with deep features. In: IEEE 26th Signal Processing and Communications Applications Conference; İzmir, Turkey; 2018. pp. 1-4.

[28] Buttrey SE, Karo C. Using k-nearest-neighbor classification in the leaves of a tree. Computational Statistics & Data Analysis 2002; 40: 27-37.

[29] Vapnik VN. The Nature of Statistical Learning Theory. New York, NY, USA: Springer, 1995.

[30] Ozgur A, Erdem H. Comparison of out-of-box machine learning algorithms used in intrusion detection systems. International Journal of Informatics Technologies 2012; 5 (2): 41-48.

[31] Hajibandeh N, Faghihi F, Ranjbar H, Kazari H. Classifications of disturbances using wavelet transform and support vector machine. Turkish Journal of Electrical Engineering & Computer Sciences 2017; 25: 832-843.

[32] Huang GB, Zhu QY, Siew CK. Extreme learning machine: a new learning scheme of feedforward neural networks. In: Proceedings of the International Joint Conference on Neural Networks; Budapest, Hungary; 2004. pp. 985–990.

[33] Alcin OF, Sengur A, Ince MC. Forward-backward pursuit based sparse extreme learning machine. Journal of the Faculty of Engineering and Architecture of Gazi University 2015; 30: 111-117.

[34] Ertuğrul ÖF, Tağluk ME. A fast feature selection approach based on extreme learning machine and coefficient of variation. Turkish Journal of Electrical Engineering & Computer Sciences 2017; 25: 3409-3420.

[35] Deniz E, Şengür A, Kadiroğlu Z, Guo Y, Bajaj V et al. Transfer learning based histopathologic image classification for breast cancer detection. Health Information Science and Systems 2018; 6 (1): 18.