Received October 22, 2021, accepted November 19, 2021, date of publication November 25, 2021,
date of current version December 9, 2021.

*Digital Object Identifier 10.1109/ACCESS.2021.3131002*

# Convolutional Neural Networks for Texture Feature Extraction. Applications to Leaf Disease Classification in Precision Agriculture

**STEFANIA BARBURICEANU, SERBAN MEZA, (Member, IEEE), BOGDAN ORZA, RAUL MALUTAN, AND ROMULUS TEREBES, (Member, IEEE)**
Communications Department, Technical University of Cluj-Napoca, 400114 Cluj-Napoca, Romania

Corresponding author: Stefania Barburiceanu (stefania.barburiceanu@com.utcluj.ro)

This work was supported by the Project "Entrepreneurial Competences and Excellence Research in Doctoral and Postdoctoral Programs—ANTREDOC," Project co-funded by the European Social Fund Financing under Grant 56437/24.07.2019.

**ABSTRACT** This paper studies the use of deep-learning models (AlexNet, VggNet, ResNet) pre-trained on object categories (ImageNet) in applied texture classification problems such as plant disease detection tasks. Research related to precision agriculture is of high relevance due to its potential economic impact on agricultural productivity and quality. Within this context, we propose a deep learning-based feature extraction method for the identification of plant species and the classification of plant leaf diseases. We focus on results relevant to real-time processing scenarios that can be easily transferred to manned/unmanned agricultural smart machinery (e.g. tractors, drones, robots, IoT smart sensor networks, etc.) by reconsidering the common processing pipeline. In our approach, texture features are extracted from different layers of pre-trained Convolutional Neural Network models and are later applied to a machine-learning classifier. For the experimental evaluation, we used publicly available datasets consisting of RGB textured images and datasets containing images of healthy and non-healthy plant leaves of different species. We compared our method to feature vectors derived from traditional handcrafted feature extraction descriptors computed for the same images and end-to-end deep-learning approaches. The proposed method proves to be significantly more efficient in terms of processing times and discriminative power, being able to surpass traditional and end-to-end CNN-based methods and provide a solution also to the problem of the reduced datasets available for precision agriculture.

**INDEX TERMS** Applied convolutional neural networks, leaf disease detection, image classification, texture classification, texture feature extraction.

## I. INTRODUCTION

Image feature extraction and classification is a computer vision field that has been studied intensively by researchers due to its practical relevance for various scenarios, including that of precision agriculture, [1]. Plant diseases have a huge effect on the agricultural productivity [2]. They can easily degrade the quality of the products, so they must be detected as soon as possible. The current methodology for detection is the human perception of plant leaves [3]. However, this method is not efficient in terms of available resources, especially for large crops, and automatic image classification systems can be beneficial in this situation. In the literature, several plant disease classification problems were addressed, such as the classification of cucumber and citrus leaves [4], [5] which is performed by using the Gray-Level Co-occurrence Matrix (GLCM) for the extraction of relevant features. In [6], colour information is used along with GLCM - derived features and Gabor characteristics for the classification of mango leaves. Deep-learning methods are also mentioned for the classification of plant diseases in [7]–[9].

Until recently, [7], [9], the problem of image classification has been addressed as a two-stage approach: the extraction of handcrafted features and machine-learning classification. The feature extraction step is regarded as the most important stage because the subsequent classification task is based

The associate editor coordinating the review of this manuscript and approving it for publication was Gangyi Jiang.

on the derived image descriptors. Even the most powerful machine-learning classifier will provide a poor classification performance if the image features are not chosen appropriately. The extraction of relevant and discriminative features is a challenging task for real-world applications. Moreover, images are captured under various conditions and, to obtain good classification results, the extracted features should provide invariance to several transformations (such as scale, rotation, illumination conditions) and robustness to noise.

One of the most popular and efficient feature extraction methods is the Local Binary Patterns operator (LBP) [10] and its improved version, proposed in [11]. The LBP descriptor is based on the signs of differences between neighbouring pixels, and it is used to describe locally the texture of the analysed image. Later, several LBP-derived operators which provide improved invariance to different transformations and a greater discrimination power were proposed, such as the Median Robust Extended Local Binary Patterns (MRELBP) [12]. Also, in order to improve the robustness to Gaussian noise, the Block Matching and 3D Filtering Extended Local Binary Patterns (BM3DELBP) was introduced by us in [13]. Another popular texture feature descriptor is the Gray-Level Co-occurrence Matrix (GLCM) [14] which achieved significant performance for texture classification tasks as reported in the literature.

In the case of traditional machine-learning methods, expert-driven feature selection and extraction are needed. A specialist must design a feature extraction method capable of outputting the most relevant features and feed them into a conventional machine-learning classifier. The classifier is then trained to learn from data and apply the learnt information to new data in order to make a classification decision.

However, lately, [7], [9], impressive results were obtained with the use of deep-learning methods, revolutionising the image and object classification field. Rather than relying on handcrafted features, these methods can be used as end-to-end approaches because they work by automatically learning the relevant features themselves, without the need of expertise, from the raw data provided as input. Deep-learning methods are constructed to learn hierarchically, their architecture being composed of several hidden layers, and are generally trained on large datasets to obtain a good classification performance. Such a dataset is ImageNet [15]. The main disadvantage of these algorithms is the long training time, which in most situations is a lot larger compared to the case of traditional classification methods. This is due to the large number of parameters that have to be learnt from the data.

The Convolutional Neural Network (CNN) is a deep-learning technique that has been widely used in the past years with great success for many computer vision tasks, [1], [7], [9], [16]. The architecture of a CNN is composed of several types of layers: convolutional, nonlinear, pooling, fully connected, normalization and others. The stack of convolutional, nonlinear, and pooling layers act as a feature extractor. The second part of the CNN is composed of several fully connected layers that are used to make a classification decision based on the generated features. We show in Fig. 1 the general block scheme of an end-to-end CNN architecture for classification tasks.

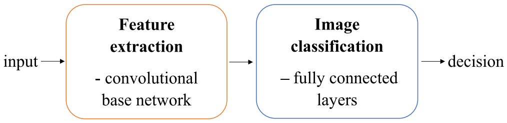


FIGURE 1. The architecture of an end-to-end CNN for image classification.

One of the main disadvantages of CNN-based methods is the fact that very large datasets are required in order to achieve significant results, [17], like with any deep-learning technique. However, there are applications in which the number of available training samples is limited, [18], especially because of the large resources (time, expertise, etc.) needed to acquire and label consistently a vast number of images (e.g. precision agriculture). This is largely addressed either by performing some sort of "data augmentation," where, from the existing data, "new" data is generated, or by deploying what is termed "transfer learning."

Data augmentation is a challenging approach, as it tries to create relevant variability in the data, and, with the use of generative adversarial networks contributes more to the increase in the overall complexity of the classification system. The work in [17] provides a relevant overview of the field, and [19] is an example of an applied case of vine leaf classification.

The "transfer learning" concept, developed by [20], [21], resembles the approaches we, as humans, take in our everyday life, as we do not learn everything from scratch, but rather use the knowledge gained in a particular previous task in other related new tasks. Practically, we transfer the knowledge acquired in the past to solve future problems. Isolated training models are designed specifically for a particular task and dataset, whereas in the transfer learning models, the gained knowledge can be transferred and used in another related new task which can imply a better performance obtained on a smaller dataset and less training time. CNN-based methods that explored the transfer learning approach by using features derived from pre-trained CNNs on large image datasets can be found in the work of [22]–[26] and others.

Typically, a new object classification problem is addressed by using a pre-trained model without its classification layers to extract the relevant features for the new problem. Practically, the weights of the network are not updated for the new task, but they are used in the new problem exactly as they were trained for the previous task and only the classification part is replaced. Popular CNN models and datasets which are widely used for feature extraction in the context of transfer learning and belonging to the object classification problem include: AlexNet [27], VggNet [28], GoogleNet [29], and ResNet [30]. Features can be extracted either from the convolutional layers or from the fully connected layers of the network. In general, it was shown, [23], [31], [32], that the features extracted from convolutional layers have a better generalization capability. The features extracted from fully connected layers have a poorer transferability because they are more specific to a particular task or dataset.

In the context of plant disease, the classification problem translates to a texture classification problem from the point of view of the image content, where the disease manifests itself more as a variation in the leaf texture rather than a type of object that is present in the image, [3], [7], [9]. Such, different state-of-the-art CNN architectures for texture classification and recognition have been proposed [33]–[36]. However, this strategy is highly impacted by the lack of large texture datasets compared to the object classification problem datasets. Fine-tuning (retraining only some layers) on small texture datasets does not bring enough improvements in the classification accuracy [33]. In [33], the authors proposed the Texture CNN architecture which is based on AlexNet but uses an energy measure derived from the last convolutional layer. They arrived at the conclusion that the size of the dataset strongly influences the performance. The authors also observed that the fine-tuning performed on a network pre-trained on textured images achieves better results than by using a network pre-trained on a dataset that contains mostly objects. This happens probably because an image from an object-oriented dataset can contain multiple textures. In [36], the authors propose Bilinear CNN Models in which the fully connected layers are replaced with bilinear pooling models.

Our paper is structured as follows. Section II describes the proposed technique which involves using pre-trained CNN models on object-oriented datasets to extract textural features. Section III details the experimental configuration setups. We use publicly available textures and images of different real-world plant species affected by disease datasets for evaluation. Section IV details the obtained experimental results together with a comparison between other handcrafted and deep-learning methods and the proposed technique in terms of classification performance and time efficiency. Section V is dedicated to final conclusions and remarks.

## II. THE PROPOSED METHOD

We are interested in the study of the performance of deep-learning pre-trained models in the classification of textured images even if the models were pre-trained on object categories. We show, therefore, how the chosen networks behave in a real task in which the textural characteristics are essential, namely in the classification of diseases that affect plant leaves. The underlying approach of the proposed method is to analyse which are the best pre-trained CNN models and their relevant layers for feature characterisation. We take advantage of the fact that there are large object datasets that allow for the pre-training of CNNs and keep the weights for the model and use this model in a new classification task.

The use of pre-trained models has several advantages. One of them is the fact that the feature extraction process is time-efficient because the images pass only once through the network. Secondly, relevant results can be obtained for small datasets for the classification task and no architecture handcrafting is required. This can be achieved because such models were trained on very large datasets, so there are many patterns and features already learnt that can be used to solve a different problem. For the significance of the results, the initial and the new task should be similar. Since we are interested in the classification of textures, even if the datasets on which popular CNN models were built are object-oriented, they are well-suited also for texture classification problems. This happens because of the hierarchical architecture of CNNs: while the early and mid-convolutional layers detect low-level features and texture structures, only the features computed from the last layers are more specific to the initial object classification task. We show in Fig. 2 the block scheme used to describe the considered texture classification system.

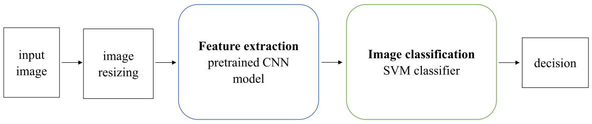


**FIGURE 2. The block scheme of the considered classification system based on pre-trained CNNs.**

We use a pre-trained CNN model from which a feature vector is obtained for each image of the dataset. The chosen supervised classifier is the Support Vector Machine with RBF kernel which is trained on 75% of the images from each class of the considered dataset and is evaluated on the rest (25%). In order to benefit from the advantages of the transfer learning concept and thus to keep the already learnt weights of the considered network, the classification layers at the end of the CNN network are removed because they are adapted to the number of classes on which the training of the CNN was performed, which is different to that of the current problem. Thus, pre-trained CNNs are used only for feature extraction in this work and the SVM is responsible for the classification. Although an artificial neural network consisting of a fully connected layer, a softmax layer, and an output layer could have been used for the classification part, it would not have surpassed the efficiency of SVM. According to [37], CNNs are very powerful as feature extractors due to their convolutional base, but less efficient for the classification operation since the classifier is in this case a linear one. On the other hand, SVM is better for the classification of more complex data [37] since by using the RBF kernel the initial feature space where data cannot be linearly separated is transformed into another higher dimensional space where the separation of data classes is possible. Using an SVM classifier on top of features extracted from CNNs instead of CNN classification layers provides better results in [38], [39].

The training and test sets are chosen randomly. For feature extraction, we considered several pre-trained models that are widely used in practice. They are presented in Table 1. Their default architecture is given in Appendix in Fig. 20-23. For these already existing models, the final classification layers are eliminated, and the features are extracted from several

**TABLE 1.** Considered pre-trained models.

| Network | Number of convolutional and fully connected layers | Parameters (millions) | Size of the input image |
|---|---|---|---|
| ResNet18 | 18 | 11.7 | 224×224×3 |
| AlexNet | 8 | 61 | 227×227×3 |
| Vgg16 | 16 | 138 | 224×224×3 |
| ResNet50 | 50 | 25.6 | 224×224×3 |

different layers in order to observe which are the most relevant for a texture classification task.

All these models were pre-trained on the ImageNet object-oriented dataset which contains more than a million images of objects classified into 1000 classes. Each pre-trained model requires input images to be of a fixed size, as given in Table 1. So, if the analysed textured images have a different size, before using the pre-trained CNN models to extract the features, the input images are resized. The convolutional base performs convolutional operations by means of several filters. The weights are the filter values and they are determined by the number and size of filters. This means that the weights corresponding to the convolutional base network do not depend on the size of the input image. So, the convolution operation is not influenced by the input image size. Filter sizes remain the same if the input image size is changed. However, the size of the feature maps will be different and that is why the number of neurons for the fully connected layers is changed depending on the input image size. This means that retraining is necessary for this situation. Changing the architecture of the model would require changing the weights which is done by training and, in this case, the purpose behind the transfer learning concept would be lost. So, to be able to rely on this concept, the images are resized to match the size required by the considered CNN models. If the difference between the size of the initial images and that imposed by CNN models is not very large and the aspect ratio is kept the same (1:1), resizing the images does not bring artifacts that could negatively influence the performance.

We experiment with the extraction of features from several layers in the network. After feature extraction, the obtained feature vectors are fed into an SVM classifier whose parameters [40] are chosen through a grid search in order to obtain the best classification accuracy for each particular experiment and method.

## III. EXPERIMENTAL SETUP

We validate the approach by two different experimental setups. Firstly, we investigate how transfer learning can be used in general for the classification of textures when the CNN models were pre-trained on large object datasets and what are, in practice, the relevant layers that can be considered from the hierarchical CNN to extract features from. Then, we use the results to provide an applied example of texture classification for the plant disease detection problem in precision agriculture.

### A. TEXTURE DATABASE: OUTEX_TC_00013

For evaluating the proposed method, we used the Outex_TC_00013 dataset [41] which contains 68 categories of RGB textured images. There are 20 samples of size 128 × 128 pixels for each class, giving a total number of 1360 images. We show in Fig. 3 a sample for each image category. This dataset is challenging because the variability between different classes is rather small in some cases, such as the granite categories, the sandpaper ones, or the barleyrice classes. Therefore, the classification task can be difficult in such cases especially because the number of samples per class is limited.

### B. PLANTVILLAGE DATASET

For validation of the method, we used the PlantVillage dataset [42] containing several plant species, some of them healthy and some affected by different diseases. In [43], the authors use three versions of this dataset: the original RGB images, the grayscale version, and the segmented RGB variant. In this paper, there is considered only the segmented RGB set. In our experiments, we only considered the segmented RGB images from [43] since the color information is relevant to this classification problem (as the change in leaf color can be a sign of a certain disease) and because the use of the segmented variant excludes any potential bias that might be caused by the presence of the background information. Images from this dataset were captured under different conditions, the plant leaves suffer different rotations and have different shapes. Moreover, there are some segmentation problems because the leaves are not always perfectly segmented from the background. We discarded from the initial dataset the images that were poorly segmented and could no longer be recognized. We show in Fig. 4 some examples of images that have segmentation problems, some of them being kept and some being discarded.

We performed two experiments: plant species identification and disease detection. For the plant species identification, we considered only the categories with healthy plant leaves. Fig. 5 shows three samples for each considered category for this experiment and Table 2 shows the 12 classes used in the plant species identification scenario. For the disease detection experiment, we considered several setups described in detail in Table 3. We also show in Fig. 6 some sample images for each class considered in each setup.

## IV. EXPERIMENTAL RESULTS AND DISCUSSION

### A. OUTEX_TC_00013 RESULTS

In the first experiment, we considered extracting the features from the last layer located before the classification layers of the four pre-trained CNN models from Table 1: ResNet18, AlexNet, Vgg16, and ResNet50. For AlexNet and Vgg16, the last layer situated before the classification layers is fully connected (fc7 for both), whereas, for ResNet18 and ResNet50, the last layer is an average pooling layer (pool5 for ResNet18 and avg_pool for ResNet50). The pre-training of

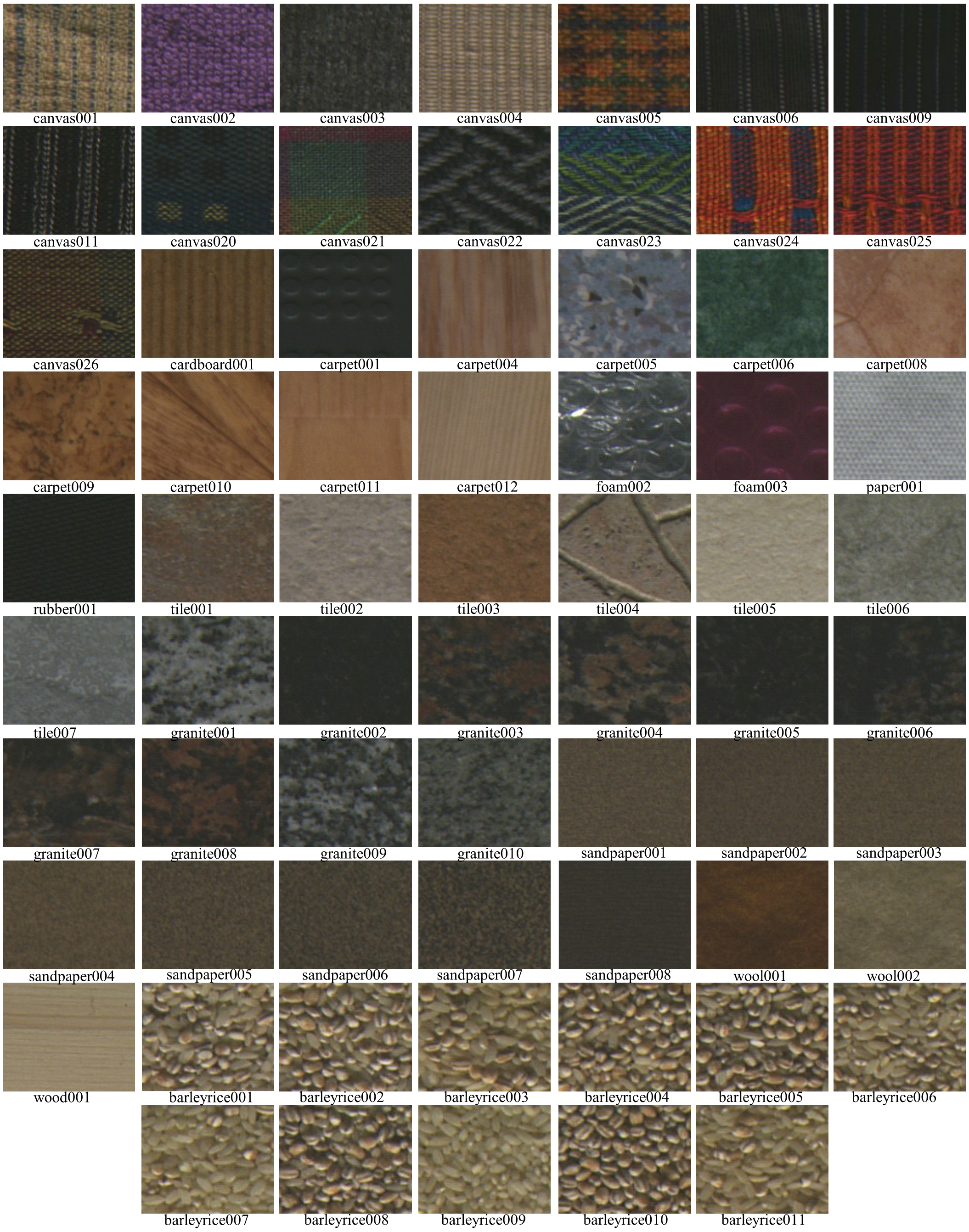


**FIGURE 3.** Samples for each class of the Outex_TC_00013 dataset.

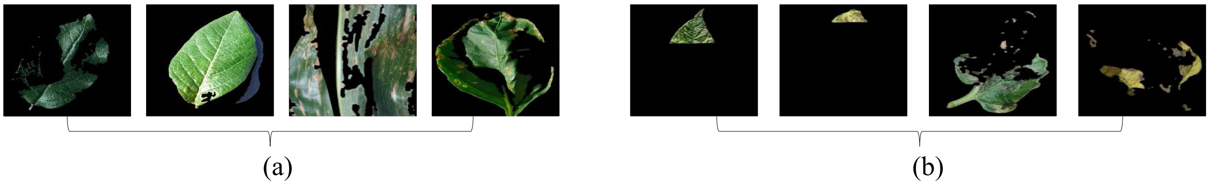


**FIGURE 4.** Examples of image samples with segmentation problems: (a) which were kept (b) which were discarded.

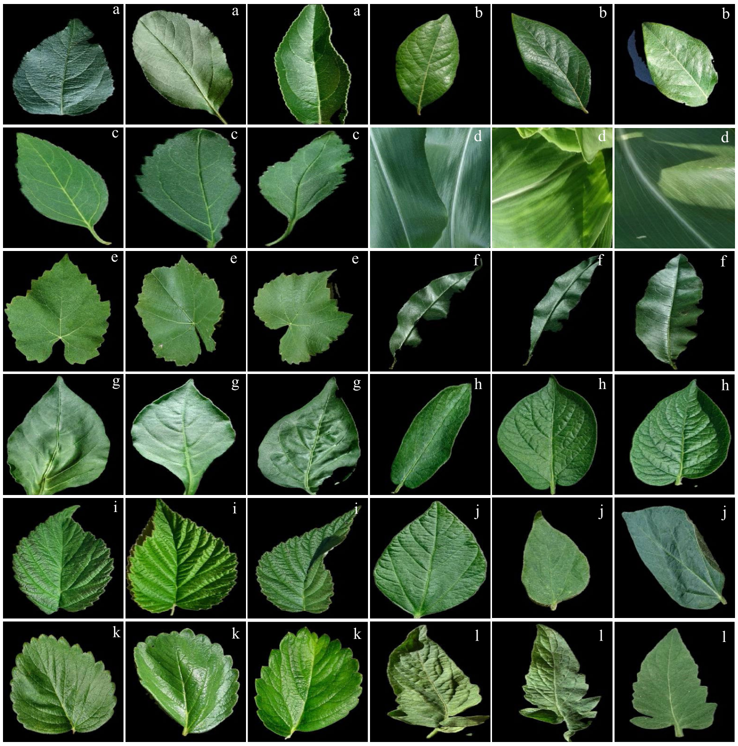


**FIGURE 5.** Image samples from category: (a) Apple healthy (b) Blueberry healthy (c) Cherry (including sour) healthy (d) Corn (maize) healthy (e) Grape healthy (f) Peach healthy (g) Pepper bell healthy (h) Potato healthy (i) Raspberry healthy (j) Soybean healthy (k) Strawberry healthy (l) Tomato healthy.

**TABLE 2.** The dataset configuration for plant species identification.

| Class | Number of samples |
|---|---|
| 1.Apple healthy | 1630 |
| 2.Blueberry healthy | 1481 |
| 3.Cherry (including sour) healthy | 852 |
| 4.Corn (maize) healthy | 1155 |
| 5.Grape healthy | 423 |
| 6.Peach healthy | 342 |
| 7.Pepper bell healthy | 1467 |
| 8.Potato healthy | 152 |
| 9.Raspberry healthy | 370 |
| 10.Soybean healthy | 5079 |
| 11.Strawberry healthy | 456 |
| 12.Tomato healthy | 1590 |
| TOTAL | 14997 images |

**TABLE 3.** The setups used for the disease detection experiment.

| Leaf disease detection setup | Classes and number of samples per class | | | | |
|---|---|---|---|---|---|
| 1.Apple | Scab (617) | Black Rot (621) | Cedar Rust (275) | Healthy (1630) | |
| 2.Cherry | Healthy (852) | Powdery mildew (1039) | | | |
| 3.Corn | Cercospora leaf spot (492) | Common Rust (1181) | Northern leaf blight (960) | Healthy (1155) | |
| 4.Grape | Black Rot (1161) | Black Measles (1379) | Leaf blight (1059) | Healthy (423) | |
| 5.Peach | Bacterial Spot (2264) | Healthy (342) | | | |
| 6.Pepper bell | Bacterial spot (980) | Healthy (1467) | | | |
| 7.Potato | Early blight (998) | Late blight (998) | Healthy (152) | | |
| 8.Strawberry | Leaf scorch (972) | Healthy (456) | | | |
| 9.Tomato | Bacterial spot (2109) | Early blight (845) | Late blight (1737) | Leaf mold (838) | Septoria leaf spot (1720) |
| | Spider mites (1665) | Target spot (1403) | Yellow leaf curl virus (5336) | Mosaic virus (370) | Healthy (1590) |

all the models was performed on the ImageNet dataset, [15]. We show in Table 4 the obtained results of the learning transfer from the pre-trained CNNs to the texture classification problem of the OUTEX_TC_00013 dataset. All feature vectors are normalised using a Z-score approach that is applied on columns (features are normalised independently from each other). We feed these feature vectors into the SVM classifier stage, with the parameters described in the second column of Table 4. All performance metrics are macro-averaging. From the results in Table 4, we can observe that the best performance is obtained by using the features extracted using the pre-trained ResNet50 model. Since the ResNet18 network achieved the worst classification results, we discarded it in the next experiments. In terms of time/resource efficiency, the fastest feature extraction for all images from the dataset is performed by the AlexNet model.

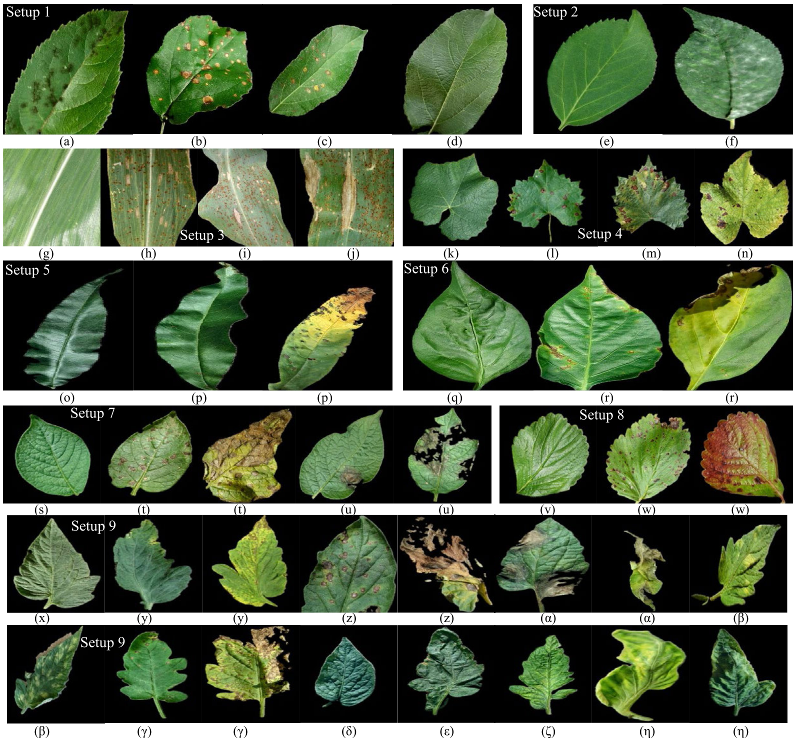


**FIGURE 6.** Image samples from category: (a) Apple Scab (b) Apple Black Rot (c) Apple Cedar Rust (d) Apple healthy (e) Cherry healthy (f) Cherry Powdery mildew (g) Corn healthy (h) Corn Cercospora leaf spot (i) Corn Common Rust (j) Corn Northern leaf blight (k) Grape healthy (l) Grape Black Rot (m) Grape Black Measles (n) Grape leaf blight (o) Peach healthy (p) Peach Bacterial Spot (q) Pepper bell healthy (r) Pepper bell Bacterial spot (s) Potato healthy (t) Potato Early blight (u) Potato Late blight (v) Strawberry healthy (w) Strawberry Leaf scorch (x) Tomato healthy (y) Tomato Bacterial spot (z) Tomato Early blight ($\alpha$) Tomato Late blight ($\beta$) Tomato Leaf mold ($\gamma$) Tomato Septoria leaf spot ($\delta$) Tomato Spider mites ($\varepsilon$) Tomato Target spot ($\zeta$) Tomato Mosaic virus ($\eta$) Tomato Yellow leaf curl virus.

**TABLE 4.** Experiment 1: The classification results [%] obtained by using the last layers of the pre-trained CNN models.

| Network | SVM parameters | Feature vector size | Feature extraction time for all images [s] | Average accuracy | Precision | Recall | F1 score |
|---|---|---|---|---|---|---|---|
| ResNet18-pool5 | C=$10^2$, $\gamma$=$10^{-3}$ | 512 | 68.79 | 75.04±1.77 | 76.6±1.77 | 75.04±1.77 | 75.82 |
| AlexNet-fc7 | C=$10^2$, $\gamma$ =$10^{-4}$ | 4096 | **42.13** | 77.82±1.98 | 78.9±2.04 | 77.82±1.98 | 78.36 |
| Vgg16-fc7 | C=$10^2$, $\gamma$ =$10^{-4}$ | 4096 | 891.37 | 76.56±1.43 | 78.21±1.45 | 76.56±1.43 | 77.38 |
| ResNet50-avg_pool | C=$10^4$, $\gamma$ =$10^{-5}$ | 2048 | 117.62 | **80.2±1.58** | **81.17±1.52** | **80.2±1.58** | **80.94** |

The results are averaged over 50 random partitions of the train and test sets

We were interested to see if the concatenation of the obtained features using two different models can increase the classification performance. We concatenated the feature vectors obtained by using AlexNet, Vgg16, and ResNet50. Table 5 details the obtained classification scores. The results achieved in the three cases are similar, the concatenation of two feature vectors generated using different models being able to slightly increase the performance. However, the size of the corresponding feature vectors is larger, which implies longer processing times.

For the two models, AlexNet and Vgg16, we pooled the features from the fully connected layers in the first experiment. However, features extracted from the fully connected layers are more specific to the initial task, on which the model was pre-trained, than features extracted from convolutional layers [23], [31], [32], [44]. In the third experiment, we also considered extracting features from the last convolutional layer (actually from the ReLU layer following the last convolutional layer) for the two models. For AlexNet, that is the relu5 layer which has 256 feature maps of size 13 × 13. In order to obtain the feature vector in this case, we averaged each feature map over all spatial locations. The layer relu5_3 of the Vgg16 model has 512 feature maps of size 14 × 14. The final feature vector is of size 512,

**TABLE 5.** Experiment 2: The classification results [%] obtained by concatenating the feature vectors of AlexNet, Vgg16, and ResNet50.

| Network | SVM parameters | Feature vector size | Feature extraction time for all images [s] | Average Accuracy | Precision | Recall | F1 score |
|---|---|---|---|---|---|---|---|
| AlexNet + Vgg fc7 | C=$10^2$, $\gamma=10^{-4}$ | 8192 | 933.5 | 82.38±2.12 | **83.92±1.84** | 82.38±2.12 | **83.14** |
| ResNet50-avg_pool + AlexNet-fc7 | C=$10^3$, $\gamma=10^{-5}$ | 6144 | **159.75** | **82.54±1.47** | 83.58±1.68 | **82.54±1.47** | 83.06 |
| ResNet50-avg_pool + Vgg16-fc7 | C=$10^2$, $\gamma=10^{-5}$ | 6144 | 1008.99 | 82.01±1.31 | 83.39±1.45 | 82.01±1.31 | 82.69 |

The results are averaged over 50 random partitions of the train and test sets

**TABLE 6.** Experiment 3: The classification results [%] obtained by considering the last convolutional layers of AlexNet and Vgg16.

| Network | SVM parameters | Feature vector size | Feature extraction time for all images [s] | Average Accuracy | Precision | Recall | F1 score |
|---|---|---|---|---|---|---|---|
| AlexNet-relu5 | C=$10^3$, $\gamma=10^{-3}$ | 256 | **44.7** | 75.26±1.86 | 76.98±1.78 | 75.26±1.86 | 76.11 |
| Vgg16-relu5_3 | C=$10^5$, $\gamma=10^{-5}$ | 512 | 1026.9 | 72.88±1.87 | 74.28±2.03 | 72.88±1.87 | 73.58 |
| AlexNet-relu5+ Vgg16-relu5_3 | C=$10^2$, $\gamma=10^{-4}$ | 768 | 1071.6 | **78.97±1.88** | **79.92±2.02** | **78.97±1.88** | **79.44** |

The results are averaged over 50 random partitions of the train and test sets

**TABLE 7.** Experiment 4: The classification results [%] obtained by considering several convolutional layers in AlexNet.

| Network | SVM parameters | Feature vector size | Feature extraction time for all images [s] | Average Accuracy | Precision | Recall | F1 score |
|---|---|---|---|---|---|---|---|
| AlexNet-relu1 | C=$10^5$, $\gamma=10^{-5}$ | 96 | **21.6** | 83.48±1.47 | 84.8±1.47 | 83.48±1.47 | 84.13 |
| AlexNet-relu2 | C=$10^3$, $\gamma=10^{-4}$ | 256 | 27.8 | 86.22±1.17 | 87.34±1.23 | 86.22±1.17 | 86.78 |
| AlexNet-relu3 | C=$10^3$, $\gamma=10^{-5}$ | 384 | 38.55 | **88.5±1.44** | **89.25±1.6** | **88.5±1.44** | **88.87** |
| AlexNet-relu4 | C=$10^3$, $\gamma=10^{-4}$ | 384 | 37.2 | 86.71±1.3 | 88±1.5 | 86.71±1.3 | 87.35 |
| AlexNet-relu5 | C=$10^3$, $\gamma=10^{-3}$ | 256 | 44.7 | 75.26±1.86 | 76.98±1.78 | 75.26±1.86 | 76.11 |

The results are averaged over 50 random partitions of the train and test sets

obtained after the averaging of the activations over all locations. We also concatenated the obtained features. The results are shown in Table 6. We can see from the obtained results the fact that the performance is marginally decreased by extracting the features from the last convolutional layers. This can happen because the features learnt by CNN up to that layer in the architecture are very complex and are more related to the initial task of object classification on which the network was pre-trained. Therefore, this validates experimentally that features from earlier layers in the network are more general and can be better at describing the texture in the context of transfer learning.

Consequently, we extracted features from several convolutional layers from AlexNet, ResNet, and Vgg16, the results being given in Tables 7-9. In all situations, we averaged each feature map over all spatial locations (the average of all values contained in the matrix corresponding to that feature map). We can observe from the obtained results the fact that extracting features from earlier layers improves the texture classification performance. This practical observation is in accordance with the theoretical understanding of CNNs. Earlier layers depict more general features, like texture structures, which are not specifically related to the initial classification problem on which the model was trained. Also, in terms of performance, the different models trade off feature extraction time to accuracy and precision as can be seen in the summary from Table 10.

Considering the model that provided the best performance (see Table 10) in terms of classification, Vgg16, we discuss hereafter the choice in the selection of the relevant layers for the transfer learning problem. For the architecture of the pre-trained Vgg16 model depicted in Fig. 7, we are interested to observe the features learnt by this model in earlier layers, such as relu2_1 (which achieves the best performance in terms of the proposed experiment for texture classification), in middle layers such as relu3_3 and relu5_1 and the features ‘‘seen’’ by the CNN model in deep layers such as relu5_3 (where the classification scores are worse; see Table 9). We consider a random choice of the training and test images for which we used the model to extract features and then classify in the SVM stage and we show in Fig. 8 a portion of the confusion matrices obtained for each selection of features corresponding to these layers.

For class 6 (barleyrice006), we can see from Fig. 8 that by considering the features extracted from the relu5_3 layer of the pre-trained Vgg16 model, none of the test images is correctly classified whereas all test samples (5) are predicted correctly by using the features extracted from the relu2_1 layer.

**TABLE 8. Experiment 4: The classification results [%] obtained by considering several convolutional layers in ResNet50.**

| Network | SVM parameters | Feature vector size | Feature extraction time for all images [s] | Average Accuracy | Precision | Recall | F1 score |
|---|---|---|---|---|---|---|---|
| ResNet50-activation_3_relu | C=$10^3$, $\gamma=10^{-2}$ | 64 | **90** | 79.39±1.48 | 80.81±1.72 | 79.39±1.48 | 80.09 |
| ResNet50-activation_4_relu | C=$10^3$, $\gamma=10^{-3}$ | 256 | 104.3 | **89.76±1.25** | **90.82±1.22** | **89.76±1.25** | **90.29** |
| ResNet50-activation_7_relu | C=$10^3$, $\gamma=10^{-4}$ | 256 | 122.7 | 87.7±1.42 | 88.85±1.4 | 87.7±1.42 | 88.27 |
| ResNet50-activation_13_relu | C=$10^3$, $\gamma=10^{-3}$ | 512 | 150.8 | 87.67±1.2 | 88.8±1.16 | 87.67±1.2 | 88.23 |
| ResNet50-activation_23_relu | C=$10^3$, $\gamma=10^{-5}$ | 256 | 207.1 | 86.87±1.22 | 87.67±1.41 | 86.87±1.22 | 87.27 |
| ResNet50-activation_24_relu | C=$10^3$, $\gamma=10^{-5}$ | 256 | 206.2 | 85.68±1.35 | 86.83±1.45 | 85.68±1.35 | 86.26 |
| ResNet50-activation_26_relu | C=$10^3$, $\gamma=10^{-5}$ | 256 | 222 | 85.65±1.4 | 86.42±1.61 | 85.65±1.4 | 86.03 |
| ResNet50-activation_32_relu | C=$10^2$, $\gamma=10^{-3}$ | 256 | 231 | 77.15±1.74 | 79.21±1.6 | 77.15±1.74 | 78.17 |
| ResNet50-activation_36_relu | C=10, $\gamma=10^{-3}$ | 256 | 235 | 75.45±1.74 | 77.1±2.04 | 75.45±1.74 | 76.27 |

The results are averaged over 50 random partitions of the train and test sets

**TABLE 9. Experiment 4: The classification results [%] obtained by considering several convolutional layers in Vgg16.**

| Network | SVM parameters | Feature vector size | Feature extraction time for all images [s] | Average Accuracy | Precision | Recall | F1 score |
|---|---|---|---|---|---|---|---|
| Vgg16-relu1_2 | C=$10^5$, $\gamma=10^{-4}$ | 64 | 7000 | 92.47±1.26 | 93.64±1.15 | 92.47±1.26 | 93.05 |
| Vgg16-relu2_1 | C=$10^4$, $\gamma=10^{-4}$ | 128 | 7919 | **93.94±1.2** | **94.76±1.15** | **93.94±1.2** | **94.35** |
| Vgg16-relu2_2 | C=$10^4$, $\gamma=10^{-4}$ | 128 | 9005 | 92.39±1.1 | 93.3±1.09 | 92.39±1.1 | 92.82 |
| Vgg16-relu3_1 | C=$10^3$, $\gamma=10^{-3}$ | 256 | 3201 | 90.33±1.18 | 91.26±1.2 | 90.33±1.18 | 90.79 |
| Vgg16-relu3_2 | C=$10^4$, $\gamma=10^{-5}$ | 256 | 3274.1 | 91.62±1.29 | 92.5±1.4 | 91.62±1.29 | 92.05 |
| Vgg16-relu3_3 | C=$10^3$, $\gamma=10^{-4}$ | 256 | 3679.5 | 89.01±1.36 | 89.96±1.45 | 89.01±1.36 | 89.48 |
| Vgg16-relu4_1 | C=$10^3$, $\gamma=10^{-5}$ | 512 | 1160 | 88.18±0.1 | 88.92±1.1 | 88.18±0.1 | 88.54 |
| Vgg16-relu4_2 | C=$10^2$, $\gamma=10^{-3}$ | 512 | 1164.8 | 87.67±1.32 | 88.6±1.38 | 87.67±1.32 | 88.14 |
| Vgg16-relu5_1 | C=$10^2$, $\gamma=10^{-4}$ | 512 | **1020** | 84.3±1.35 | 85.61±1.56 | 84.3±1.35 | 84.97 |
| Vgg16-relu5_3 | C=$10^5$, $\gamma=10^{-5}$ | 512 | 1026.9 | 72.88±1.87 | 74.28±2.03 | 72.88±1.87 | 73.58 |

The results are averaged over 50 random partitions of the train and test sets

**TABLE 10. Summary of the best performance [%] obtained for the considered pre-trained CNN models.**

| Network | SVM parameters | Feature vector size | Feature extraction time for all images [s] | Average Accuracy | Precision | Recall | F1 score |
|---|---|---|---|---|---|---|---|
| AlexNet-relu3 | C=$10^3$, $\gamma=10^{-5}$ | 384 | **38.55** | 88.5±1.44 | 89.25±1.6 | 88.5±1.44 | 88.87 |
| ResNet50-activation_4_relu | C=$10^3$, $\gamma=10^{-3}$ | 256 | 104.3 | 89.76±1.25 | 90.82±1.22 | 89.76±1.25 | 90.29 |
| Vgg16-relu2_1 | C=$10^4$, $\gamma=10^{-4}$ | 128 | 7919 | **93.94±1.2** | **94.76±1.15** | **93.94±1.2** | **94.35** |

The results are averaged over 50 random partitions of the train and test sets

Therefore, we are interested to observe the features learnt by the pre-trained Vgg16 model from the two layers and other intermediary layers by considering as input a test image from this class (barleyrice006). Fig. 9 presents the first 64 obtained feature maps. For better visualization, the following logarithmic transform is applied to all feature maps:

$$I_l = 3 \times ln(I + 1) \tag{1}$$

where $I$ is the initial feature map and $I_l$ is the feature map obtained after applying the logarithmic transform.

All images from Fig. 9 present only the positive activations since they are extracted from ReLU layers. As we can see, the relu2_1 layer extracts different textural features and most of the channels show large activations. As we move deeper into the network, fewer and fewer activations occur. In the case of relu5_3, most of the feature maps do not contain activations at all. This happens because the last convolutional layers explore abstract and complex structures related to the objects present in the images from the ImageNet dataset that was used in the pre-training step. So, the feature maps are

**TABLE 11.** Experiment 5: The classification results [%] obtained by considering the concatenation of features derived from convolutional layers in the pre-trained AlexNet, Vgg16, and ResNet50.

| Network | SVM parameters | Average Accuracy | Precision | Recall | F1 score |
|---|---|---|---|---|---|
| AlexNet-relu3 + ResNet50-activation_4_relu | C=$10^3$, $\gamma$ =$10^{-5}$ | 90.26±1.1 | 91.08±1.26 | 90.26±1.1 | 90.67 |
| AlexNet-relu3 + Vgg16-relu2_1 | C=$10^3$, $\gamma$ =$10^{-5}$ | 89.56±1.44 | 90.43±1.4 | 89.56±1.44 | 89.99 |
| Vgg16-relu2_1 + ResNet50-activation_4_relu | C=$10^3$, $\gamma$ =$10^{-3}$ | **90.76±1.13** | **91.85±1.07** | **90.76±1.13** | **91.3** |

The results are averaged over 50 random partitions of the train and test sets

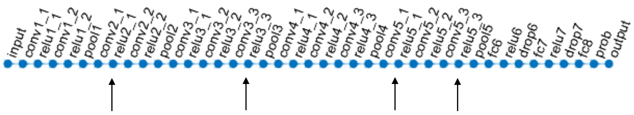


**FIGURE 7.** Pre-trained Vgg16 architecture setup for feature extraction.

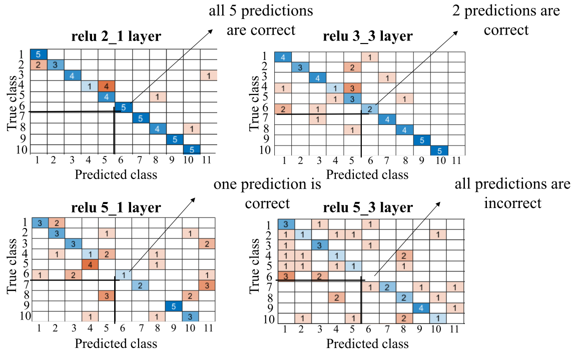


**FIGURE 8.** Confusion matrix portion for the classification of features extracted using several layers from the pre-trained Vgg16 model.

not activated in most cases and they are not useful for texture classification. Fig. 10 shows the considered test image along with three feature maps with large activations for the relu2_1 layer.

We can observe from Fig. 10 the fact that a lower layer such as relu2_1 is able to extract relevant features for the considered test image. There is no need to go very deep into the network for texture classification. Also, in some cases, pooling features from deeper layers can actually degrade the performance.

We also concatenated the feature vectors extracted from the considered models that achieved the best performance and obtained the results given in Table 11. By concatenating the feature vectors generated from the pre-trained AlexNet and ResNet50 models, only a slight increase in performance is observed compared to the individual scores. The results of Vgg16 are decreased when performing the concatenation to other feature vectors.

In order to compare the results obtained using the proposed architecture for texture feature extraction from pre-trained CNNs to the performance obtained using handcrafted feature vectors on the same datasets [12], [45], we show in Table 12 a synthesis of these results. We also consider the AlexNet and Vgg16 architectures as end-to-end approaches, where both the features and classification are made by the network. We directly train the AlexNet and Vgg16 CNNs on the Outex_TC_00013 dataset in order to observe if the obtained results surpass a model pre-trained on a large dataset consisting of object categories. The considered training parameters are given in Appendix. Fig. 11 shows the training progress on a random partition of the training and test sets for both networks. We can observe from Table 12 that the pre-trained AlexNet deep-learning model surpasses the MRELBP operator [12] which works on grayscale images.

However, when incorporating the colour information provided by OCCBM3DELBP [45], the pre-trained AlexNet is outperformed. This comes with the trade-off of much longer processing times for extracting features using OCCBM3DELBP, 2223.2 seconds being the average feature extraction time for all images in the dataset. By training AlexNet end-to-end on the Outex_TC_00013 dataset, we can observe from Fig. 11 the fact that the classification accuracy on the training set reached up to 100%. However, the classification scores obtained for the validation set are lower as the results for the two sets start to diverge around the 25th epoch. The difference is about 25%, meaning that the network has learnt the data from the training set, but it is not able to generalize well for new data. This happens because there is a small number of training images per class. The same applies to the trained end-to-end Vgg16 model where the difference between the training and validation accuracy is even higher, of approximately 36% (probably because there are more parameters in this case). This is consistent with the observation that CNNs require large datasets for satisfactory classification results. However, the handcrafted operators and the pre-trained models work very well even in this situation.

We can see from Table 12 that the best performance is achieved by considering the pre-trained Vgg16 model and by extracting features from the relu2_1 layer. A good compromise between classification accuracy and time efficiency can be obtained by extracting features from the relu3 layer of the AlexNet model since it is by far the fastest strategy: the average feature extraction time for all images is 38.55 seconds for the pre-trained AlexNet compared to more than 2 hours for the pre-trained Vgg16 relu2_1 model.

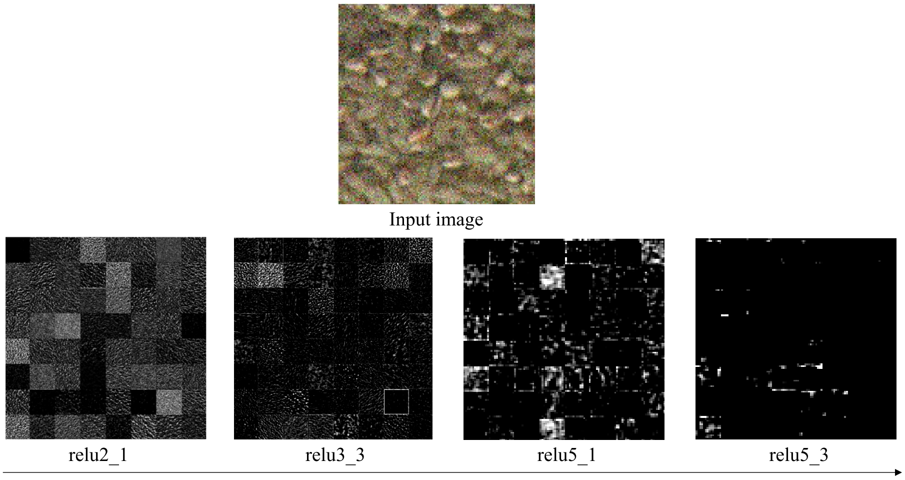


**FIGURE 9.** The input image and the first 64 feature maps for the considered layers in the pre-trained Vgg16 model.

**TABLE 12.** Comparison to other feature extraction techniques [%].

| Operator | Parameters | Feature vector size | Feature extraction time for all images [s] | Average Accuracy | Precision | Recall | F1 score |
|---|---|---|---|---|---|---|---|
| MRELBP [12] | C=$10^2$, $\gamma$ =$10^{-4}$ | 800 | 875.84 | 84.06±1.2 | 85.12±1.25 | 84.06±1.2 | 84.58 |
| OCCBM3DELBP [45] | C=$10^2$, $\gamma$ =$10^{-4}$, m=5 | 4800 | 2223.2 | 92.1±1.24 | 92.75±1.21 | 92.1±1.24 | 92.42 |
| AlexNet end-to-end trained on Outex_TC_00013 | Appendix A | - | - | 75.18±4.53 | 79.95±3.31 | 75.18±4.52 | 77.5 |
| Vgg16 end-to-end trained on Outex_TC_00013 | Appendix A | - | - | 65.15±3.86 | 68.14±3.99 | 65.15±3.86 | 66.61 |
| Pre-trained AlexNet-relu3 & SVM | C=$10^3$, $\gamma$ =$10^{-5}$ | 384 | **38.55** | 88.5±1.44 | 89.25±1.6 | 88.5±1.44 | 88.87 |
| Pre-trained ResNet50-activation_4_relu & SVM | C=$10^3$, $\gamma$ =$10^{-3}$ | 256 | 104.3 | 89.76±1.25 | 90.82±1.22 | 89.76±1.25 | 90.29 |
| Pre-trained Vgg16-relu2_1 & SVM | C=$10^4$, $\gamma$ =$10^{-4}$ | 128 | 7919 | **93.94±1.2** | **94.76±1.15** | **93.94±1.2** | **94.35** |

The results are averaged over 50 random partitions of the train and test sets

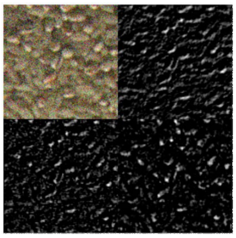
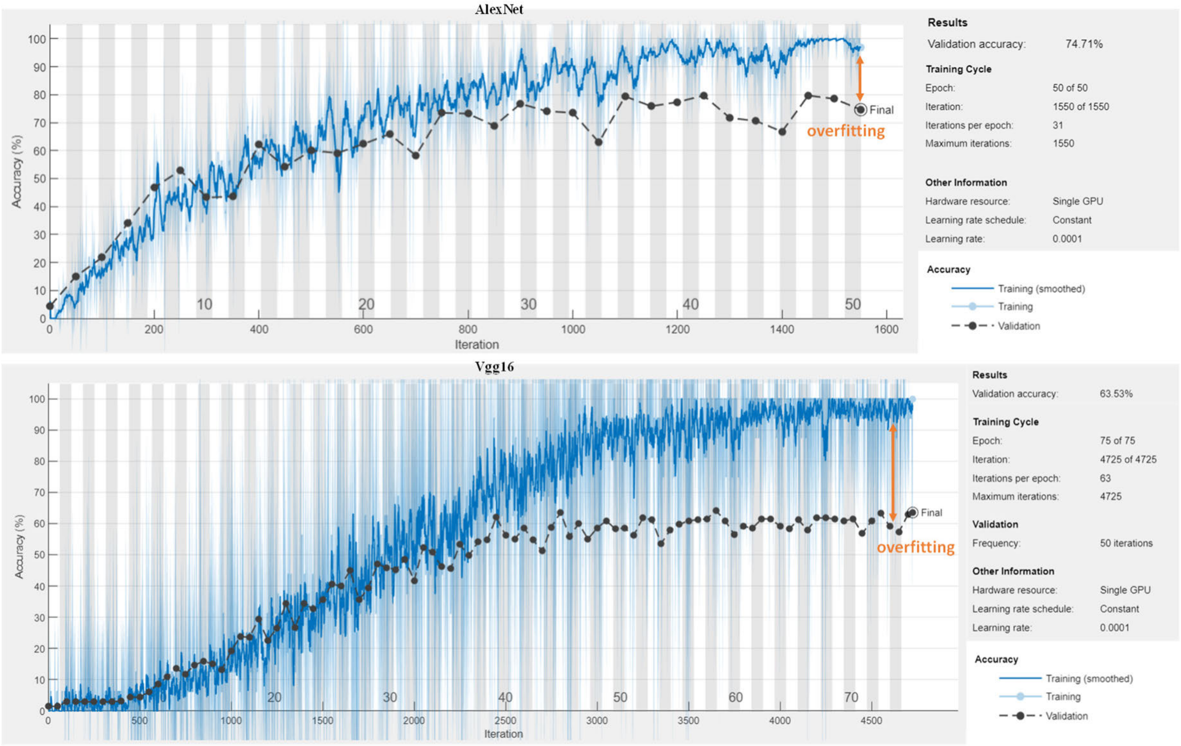


**FIGURE 10.** The input image and the associated feature maps with large activations for the relu2_1 layer of Vgg16.

From the obtained results we can conclude that, even if the considered models have been (pre)trained on object classes, they are also efficient for texture classification. Features extracted from early convolutional layers in the network represent less complex patterns being mostly related to features such as the textural content. Moreover, they are more general and are not related specifically to the initial classification problem on which the models were trained and, in combination with classifiers that support small datasets (like SVMs) provide important and relevant classification scores.

## B. PLANTVILLAGE RESULTS

We are interested to evaluate the performance of pre-trained CNNs on real-world images of plant leaves. For feature extraction, we consider the pre-trained AlexNet model since it achieves a promising performance for the texture dataset compared to the rest of the analysed models in Section IV A, also being the fastest in terms of processing time. In such practical applications, the feature extraction time should be as small as possible for real-time processing and classification. The relu2, relu3, and relu4 layers were chosen for extracting features using the pre-trained AlexNet model based on the exhibited performance for the more general, in terms of texture classification tasks, Outex_TC_00013 dataset. Even if the pre-trained Resnet50 model offered a performance similar



AlexNet

**Results**

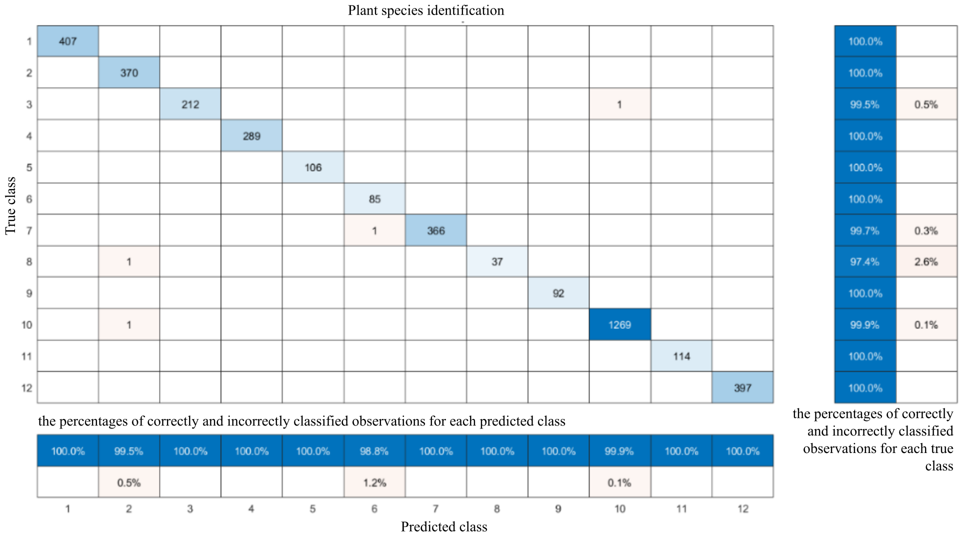
| | |
|---|---|
| Validation accuracy: | 74.71% |
| **Training Cycle** | |
| Epoch: | 50 of 50 |
| Iteration: | 1550 of 1550 |
| Iterations per epoch: | 31 |
| Maximum iterations: | 1550 |
| **Other Information** | |
| Hardware resource: | Single GPU |
| Learning rate schedule: | Constant |
| Learning rate: | 0.0001 |

Vgg16

**Results**

| | |
|---|---|
| Validation accuracy: | 63.53% |
| **Training Cycle** | |
| Epoch: | 75 of 75 |
| Iteration: | 4725 of 4725 |
| Iterations per epoch: | 63 |
| Maximum iterations: | 4725 |
| **Validation** | |
| Frequency: | 50 iterations |
| **Other Information** | |
| Hardware resource: | Single GPU |
| Learning rate schedule: | Constant |
| Learning rate: | 0.0001 |

**FIGURE 11.** The training progress on a random partition of the training and test sets using the AlexNet and Vgg16 end-to-end architectures.


Plant species identification
True class
Predicted class

| True class \ Predicted class | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 407 | | | | | | | | | | | | 100.0% | |
| 2 | | 370 | | | | | | | | | | | 100.0% | |
| 3 | | | 212 | | | | | | | 1 | | | 99.5% | 0.5% |
| 4 | | | | 289 | | | | | | | | | 100.0% | |
| 5 | | | | | 106 | | | | | | | | 100.0% | |
| 6 | | | | | | 85 | | | | | | | 100.0% | |
| 7 | | | | | | 1 | 366 | | | | | | 99.7% | 0.3% |
| 8 | | 1 | | | | | | 37 | | | | | 97.4% | 2.6% |
| 9 | | | | | | | | | 92 | | | | 100.0% | |
| 10 | | 1 | | | | | | | | 1269 | | | 99.9% | 0.1% |
| 11 | | | | | | | | | | | 114 | | 100.0% | |
| 12 | | | | | | | | | | | | 397 | 100.0% | |
| | 100.0% | 99.5% | 100.0% | 100.0% | 100.0% | 98.8% | 100.0% | 100.0% | 100.0% | 99.9% | 100.0% | 100.0% | | |
| | | 0.5% | | | | 1.2% | | | | 0.1% | | | | |

the percentages of correctly and incorrectly classified observations for each predicted class

the percentages of correctly and incorrectly classified observations for each true class

**FIGURE 12.** The obtained confusion matrix for one run in the plant species identification experiment.

to that of AlexNet and the pre-trained Vgg16 model proved to support better classification results for the Outex dataset, these models, however, did not qualify for consideration for real-time processing applications. Thus, we did not consider them for the PlantVillage dataset.

The first experiment consists in the identification of 12 plant species by considering only healthy leaves. We compare our results with the results obtained using other handcrafted feature vectors on the same dataset. The obtained classification scores are shown in Table 13. The obtained results show that the pre-trained AlexNet model achieves the best classification scores by considering the relu3 layer, with an average feature extraction time of only 321 sec (compared to OCCBM3DELBP which achieves more than 30 hours). Fig. 12 shows the confusion matrix computed for one run (particular random choice of the training and test sets) using

**TABLE 13. Plant species identification results [%].**

| Operator | SVM parameters | Feature vector size | Feature extraction time for all images [s] | Average Accuracy | Precision | Recall | F1 score |
|---|---|---|---|---|---|---|---|
| AlexNet-relu2 | C=$10^3$, $\gamma=10^{-3}$ | 256 | **248** | 99.75±0.07 | 99.77±0.12 | 99.47±0.24 | 99.62 |
| AlexNet-relu3 | C=$10^2$, $\gamma=10^{-3}$ | 384 | 321 | **99.86±0.05** | **99.86±0.07** | **99.69±0.21** | **99.77** |
| AlexNet-relu4 | C=$10^3$, $\gamma=10^{-4}$ | 384 | 310 | 99.81±0.07 | 99.79±0.13 | 99.52±0.26 | 99.66 |
| MRELBP [12] | C=$10^2$, $\gamma=10^{-4}$ | 800 | 25962 | 98.36±0.2 | 98.23±0.34 | 97.38±0.4 | 97.8 |
| OCCBM3DELBP (m=5) [45] | C=$10^2$, $\gamma=10^{-5}$ | 4800 | 109177 | 99.69±0.08 | 99.69±0.09 | 99.42±0.2 | 99.56 |

The results are averaged over 50 random partitions of the train and test sets

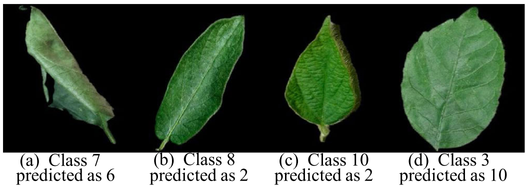

(a) Class 7 predicted as 6 (b) Class 8 predicted as 2 (c) Class 10 predicted as 2 (d) Class 3 predicted as 10

**FIGURE 13. Incorrectly classified samples in the plant species identification experiment.**

the pre-trained AlexNet model and relu3 layer for feature extraction and an SVM classifier. We also show in Fig. 13 the four incorrectly classified images. We can observe that in a) and b), the plant leaves are rotated at different angles which does not allow a complete exposure of the leaf and thus, the incorrect classification appears. The image from Fig.13 c) presents some segmentation errors which can be the reason for the erroneous prediction. The image sample shown in Fig.13 d) is very similar visually to the training images from the predicted class and probably, due to the high intra-class variability for category 3, the misclassification occurs.

The second experiment contains nine setup configurations used for disease detection in several plant species. We show in Table 14 the obtained results for all setups in comparison to the other handcrafted methods. We can see that the pre-trained AlexNet model is much faster than the other ones and also achieves the best classification results for all setups. For cherries and strawberries, a perfect classification is obtained. The highest improvement is achieved for the disease detection of tomato leaves which is one of the most difficult classification problems due to the high number of classes (10 classes) compared to the other setups.

Since the lowest scores are obtained for the classification of corn and tomato, we are going to analyse some of the incorrectly classified samples for these setups.

In the third setup corresponding to corn leaves, there are three classes associated with different leaf diseases and one class of healthy leaves. We show the confusion matrix obtained for one random partition using the AlexNet relu2 method in Fig. 14. As we can see, 23 samples are misclassified in this case. We show in Fig. 15 some of these images.

From the total of 23 misclassified samples, 22 were due to the confusion between two diseases, classes 1 and 3 (Cercospora leaf spot and Northern leaf blight). Even visually in some cases, it is difficult to make a distinction between the two categories.

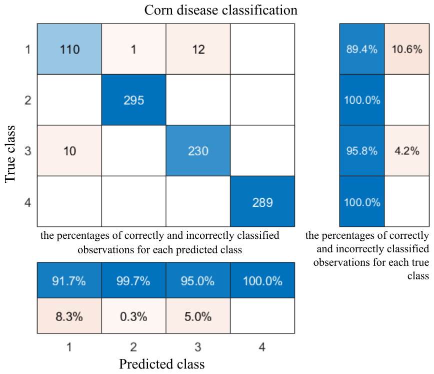


**FIGURE 14. The obtained confusion matrix for one run in the disease detection of corn leaves.**

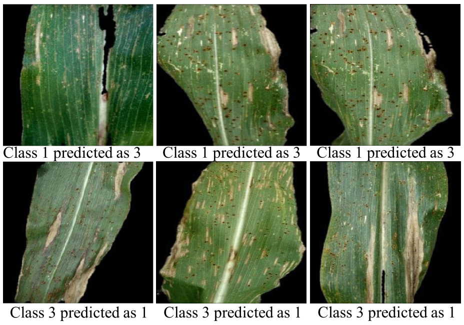

Class 1 predicted as 3 Class 1 predicted as 3 Class 1 predicted as 3
Class 3 predicted as 1 Class 3 predicted as 1 Class 3 predicted as 1

**FIGURE 15. Incorrectly classified samples for disease detection in corn leaves.**

The last setup corresponding to the tomato leaves comprises images from 10 categories: tomato leaves affected by nine different diseases and healthy ones. Fig. 16 shows the computed confusion matrix for one run on this setup by considering the pre-trained AlexNet model and relu3 layer for feature extraction and the SVM classifier. We can see that for this run, there are 82 misclassifications. The true class with the smallest percentage of correctly classified samples is 2. We show some misclassified observations in Fig. 17. There is a high intra-class variability for class 2 and that is why the classification results are poorer for the images comprised in this category. However, taking into account the fact that

**TABLE 14. Disease detection results [%].**

| Setup | Operator (feature descriptor) | SVM parameters | Average feature extraction time for all images [s] | Average Accuracy | Precision | Recall | F1 score |
|---|---|---|---|---|---|---|---|
| 1-Apple | Pre-trained AlexNet-relu2 | C=10, $\gamma=10^{-3}$ | **54** | 99.54±0.2 | 99.53±0.23 | 99.41±0.29 | 99.47 |
| | Pre-trained AlexNet-relu3 | C=10, $\gamma=10^{-3}$ | 68.83 | 99.66±0.22 | 99.7±0.2 | 99.58±0.3 | 99.64 |
| | Pre-trained AlexNet-relu4 | C=10, $\gamma=10^{-3}$ | 66.1 | **99.74±0.17** | **99.75±0.16** | **99.66±0.26** | **99.7** |
| | MRELBP [12] | C=$10^2$, $\gamma=10^{-4}$ | 5440.9 | 97.21±0.45 | 96.93±0.54 | 96.27±0.77 | 96.6 |
| | OCCBM3DELBP (m=5) [45] | C=$10^2$, $\gamma=10^{-5}$ | 22881 | 99.53±0.26 | 99.56±0.21 | 99.48±0.3 | 99.52 |
| 2-Cherry | Pre-trained AlexNet-relu2 | C=1, $\gamma=10^{-3}$ | **30.2** | 99.97±0.07 | 99.97±0.07 | 99.97±0.07 | 99.97 |
| | Pre-trained AlexNet-relu3 | C=$10^{-1}$, $\gamma=10^{-3}$ | 42.16 | **100±0** | **100±0** | **100±0** | **100** |
| | Pre-trained AlexNet-relu4 | C=1, $\gamma=10^{-3}$ | 44 | **100±0** | **100±0** | **100±0** | **100** |
| | MRELBP [12] | C=$10^2$, $\gamma=10^{-4}$ | 3271.43 | 99.9±0.12 | 99.9±0.12 | 99.9±0.12 | 99.9 |
| | OCCBM3DELBP (m=5) [45] | C=$10^2$, $\gamma=10^{-5}$ | 13750.4 | 99.9±0.12 | 99.9±0.12 | 99.9±0.12 | 99.9 |
| 3-Corn | Pre-trained AlexNet-relu2 | C=$10^2$, $\gamma=10^{-3}$ | **72** | **97.48±0.46** | 96.46±0.68 | **96.52±0.65** | **96.49** |
| | Pre-trained AlexNet-relu3 | C=10, $\gamma=10^{-3}$ | 84.99 | 97.45±0.49 | **96.52±0.7** | 96.39±0.77 | 96.46 |
| | Pre-trained AlexNet-relu4 | C=10, $\gamma=10^{-3}$ | 81.3 | 97.41±0.5 | 96.42±0.7 | 96.43±0.72 | 96.42 |
| | MRELBP [12] | C=$10^2$, $\gamma=10^{-4}$ | 6558.2 | 94±0.6 | 92±0.83 | 91.5±0.89 | 91.7 |
| | OCCBM3DELBP (m=5) [45] | C=$10^2$, $\gamma=10^{-5}$ | 27570.6 | 96.77±0.6 | 95.46±0.78 | 95.44±0.81 | 95.45 |
| 4-Grape | Pre-trained AlexNet-relu2 | C=10, $\gamma=10^{-3}$ | **90** | 99.09±0.32 | 99.24±0.27 | 99.29±0.25 | 99.26 |
| | Pre-trained AlexNet-relu3 | C=$10^2$, $\gamma=10^{-3}$ | 105.35 | **99.53±0.18** | **99.61±0.15** | **99.63±0.14** | **99.62** |
| | Pre-trained AlexNet-relu4 | C=$10^2$, $\gamma=10^{-3}$ | 104 | 99.44±0.19 | 99.53±0.15 | 99.56±0.15 | 99.55 |
| | MRELBP [12] | C=$10^2$, $\gamma=10^{-4}$ | 7001 | 93.48±0.76 | 94.73±0.65 | 94.7±0.63 | 94.72 |
| | OCCBM3DELBP (m=5) [45] | C=$10^2$, $\gamma=10^{-5}$ | 29275.5 | 99.07±0.21 | 99.25±0.17 | 99.25±0.17 | 99.25 |
| 5-Peach | Pre-trained AlexNet-relu2 | C=$10^3$, $\gamma=10^{-5}$ | **43.4** | **99.65±0.18** | 99.31±0.49 | **99.14±0.51** | **99.22** |
| | Pre-trained AlexNet-relu3 | C=$10^3$, $\gamma=10^{-5}$ | 58.58 | 99.53±0.2 | 99.04±0.6 | 98.88±0.52 | 98.96 |
| | Pre-trained AlexNet-relu4 | C=10, $\gamma=10^{-3}$ | 56 | 99.62±0.24 | **99.38±0.58** | 98.94±0.64 | 99.16 |
| | MRELBP [12] | C=$10^2$, $\gamma=10^{-4}$ | 4510.3 | 98.1±0.52 | 96.15±1.32 | 95.83±1.51 | 96 |
| | OCCBM3DELBP (m=5) [45] | C=$10^2$, $\gamma=10^{-5}$ | 18961.6 | 99.48±0.25 | 98.95±0.66 | 98.8±0.74 | 98.87 |
| 6-Pepper bell | Pre-trained AlexNet-relu2 | C=$10^2$, $\gamma=10^{-3}$ | **43** | 99.78±0.17 | 99.8±0.16 | 99.74±0.2 | 99.77 |
| | Pre-trained AlexNet-relu3 | C=10, $\gamma=10^{-3}$ | 54.7 | 99.71±0.23 | 99.73±0.22 | 99.67±0.27 | 99.7 |
| | Pre-trained AlexNet-relu4 | C=10, $\gamma=10^{-3}$ | 52 | **99.82±0.15** | **99.81±0.17** | **99.83±0.15** | **99.82** |
| | MRELBP [12] | C=$10^2$, $\gamma=10^{-4}$ | 4233.7 | 91.26±0.87 | 91.03±0.93 | 90.83±0.91 | 90.93 |
| | OCCBM3DELBP (m=5) [45] | C=$10^2$, $\gamma=10^{-5}$ | 17814.1 | 99.46±0.32 | 99.51±0.3 | 99.37±0.38 | 99.44 |
| 7-Potato | Pre-trained AlexNet-relu2 | C=$10^3$, $\gamma=10^{-4}$ | **43** | 99.18±0.4 | 98.3±1.15 | 97.82±1.44 | 98.06 |
| | Pre-trained AlexNet-relu3 | C=$10^3$, $\gamma=10^{-4}$ | 55.47 | 99.34±0.33 | 98.51±0.89 | **98.56±0.93** | 98.54 |
| | Pre-trained AlexNet-relu4 | C=$10^3$, $\gamma=10^{-5}$ | 53.4 | **99.43±0.32** | **98.87±0.75** | 98.45±1.12 | **98.66** |
| | MRELBP [12] | C=$10^2$, $\gamma=10^{-4}$ | 3716.3 | 95.07±0.9 | 93.28±1.7 | 90.86±2.12 | 92.06 |
| | OCCBM3DELBP (m=5) [45] | C=$10^2$, $\gamma=10^{-5}$ | 15635.4 | 98.9±0.41 | 97.58±1.23 | 97.66±1.12 | 97.62 |
| 8-Strawberry | Pre-trained AlexNet-relu2 | C=$10^{-1}$, $\gamma=10^{-3}$ | **27** | **100±0** | **100±0** | **100±0** | **100** |
| | Pre-trained AlexNet-relu3 | C=$10^{-1}$, $\gamma=10^{-3}$ | 29.67 | **100±0** | **100±0** | **100±0** | **100** |
| | Pre-trained AlexNet-relu4 | C=$10^{-1}$, $\gamma=10^{-3}$ | 28 | **100±0** | **100±0** | **100±0** | **100** |
| | MRELBP [12] | C=$10^2$, $\gamma=10^{-4}$ | 2505 | 99.41±0.38 | 99.31±0.47 | 99.37±0.44 | 99.34 |
| | OCCBM3DELBP (m=5) [45] | C=$10^2$, $\gamma=10^{-5}$ | 10385.8 | 99.98±0.08 | 99.98±0.06 | 99.97±0.12 | 99.97 |
| 9-Tomato | Pre-trained AlexNet-relu2 | C=$10^2$, $\gamma=10^{-3}$ | **280** | 97.69±0.19 | 96.8±0.75 | 96.68±0.29 | 96.74 |
| | Pre-trained AlexNet-relu3 | C=$10^2$, $\gamma=10^{-3}$ | 348.01 | **97.99±0.22** | **97.14±0.37** | **97.04±0.34** | **97.09** |
| | Pre-trained AlexNet-relu4 | C=$10^3$, $\gamma=10^{-3}$ | 330 | 97.49±0.2 | 96.53±0.32 | 96.27±0.3 | 96.4 |
| | MRELBP [12] | C=$10^2$, $\gamma=10^{-4}$ | 30475 | 85.41±0.46 | 81.15±0.68 | 80.56±0.73 | 80.85 |
| | OCCBM3DELBP (m=5) [45] | C=$10^2$, $\gamma=10^{-5}$ | 128212.6 | 94.8±0.41 | 94.75±0.44 | 94.57±0.44 | 94.99 |

The results are averaged over 50 random partitions of the train and test sets

for this setup there were considered nine different diseases and images are exposed to various conditions, the achieved performance is promising.

For each of the two setups corresponding to cherry and strawberry leaves, there are two categories: healthy and non-healthy. Fig. 18 presents some image samples. We can see that even visually the classification task is not challenging in these cases and therefore perfect classification scores are obtained.

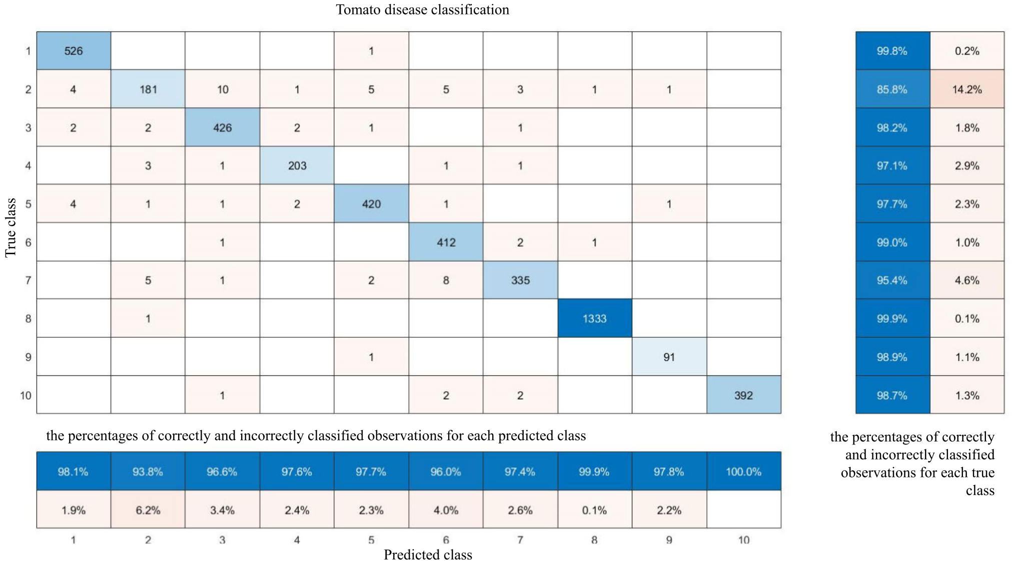
Table 15 shows the results obtained by other state-of-the-art methods for different experiments performed on images from the PlantVillage dataset. We can observe that no other work obtains better performances for the considered experiments than the method proposed in this paper.

### C. APPLICABILITY OF THE METHOD

The greatest impact of the proposed method is represented by its applicability for texture classification in cases where large datasets are not available and time performance approaching real-time scenarios is required, like, for instance, precision agriculture. It is in such instances that constructing the model, and training it can be challenging and may require either expertise in tailoring the feature extraction step to the task (and a good understanding of the variability in the data) or,

**TABLE 15. Comparison to state-of-the-art methods for the PlantVillage dataset [%].**

| Paper | Method used | Experiment | Number of classes | Maximum average accuracy | Maximum precision | Maximum recall | Maximum F1 score |
|---|---|---|---|---|---|---|---|
| [46] | GLCM and others +SVM | Potato | 3 | 95 | 95 | 95 | 95 |
| [47] | Color and GEV features +SVM | Tomato | 7 | 84.7 | - | - | - |
| [48] | CNNs | Tomato | 10 | 95.65 | - | - | - |
| [49] | CNN | Tomato | 9 | 92 | 90 | 92 | 91 |
| [50] | High-Order Residual Network | Tomato | 6 | 91.79 | - | - | - |
| [51] | Convolutional autoencoder | Potato<br>Corn | 3<br>4 | 87.01<br>81.84 | - | - | - |
| [52] | CNNs | Tomato | 10 | 95.24 | - | - | - |
| [53] | CNNs | Apple | 4 | 98.54 | - | - | - |



Tomato disease classification

| True class \ Predicted class | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |  |  |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 526 |  |  |  | 1 |  |  |  |  |  | 99.8% | 0.2% |
| 2 | 4 | 181 | 10 | 1 | 5 | 5 | 3 | 1 | 1 |  | 85.8% | 14.2% |
| 3 | 2 | 2 | 426 | 2 | 1 |  | 1 |  |  |  | 98.2% | 1.8% |
| 4 |  | 3 | 1 | 203 |  | 1 | 1 |  |  |  | 97.1% | 2.9% |
| 5 | 4 | 1 | 1 | 2 | 420 | 1 |  |  | 1 |  | 97.7% | 2.3% |
| 6 |  |  | 1 |  |  | 412 | 2 | 1 |  |  | 99.0% | 1.0% |
| 7 |  | 5 | 1 |  | 2 | 8 | 335 |  |  |  | 95.4% | 4.6% |
| 8 |  | 1 |  |  |  |  |  | 1333 |  |  | 99.9% | 0.1% |
| 9 |  |  |  |  | 1 |  |  |  | 91 |  | 98.9% | 1.1% |
| 10 |  |  | 1 |  |  | 2 | 2 |  |  | 392 | 98.7% | 1.3% |

the percentages of correctly and incorrectly classified observations for each predicted class

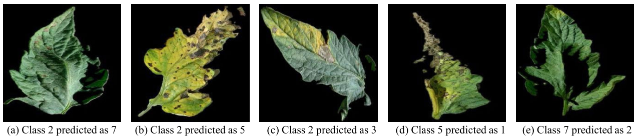
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|
| 98.1% | 93.8% | 96.6% | 97.6% | 97.7% | 96.0% | 97.4% | 99.9% | 97.8% | 100.0% |
| 1.9% | 6.2% | 3.4% | 2.4% | 2.3% | 4.0% | 2.6% | 0.1% | 2.2% |  |

the percentages of correctly and incorrectly classified observations for each true class

**FIGURE 16. The obtained confusion matrix for one run in the disease detection of tomato leaves.**

if end-to-end deep-learning-based models are used, be dependent on the size of the training dataset. To address this, we proposed the pre-training of existing popular models (like AlexNet) on object-based large datasets (ImageNet) and the use of the description some of the hidden layers in the model provide as features for an SVM classifier. If the choice of the SVM classifier is also approached through a grid search, in order to maximize the classification accuracy, the level of expertise adapted to the particularities of the texture classification task is greatly reduced, together with no increase in the available dataset size or complexity. Also, by reconsidering the common processing pipeline for real-time processing scenarios that can be easily transferred to manned/unmanned agricultural smart machinery (e.g. tractors, drones, robots, IoT smart sensor networks, etc.), the classification system becomes a single image prediction approach where the model is trained once and then it is stored locally, on the machinery.

We have shown that feature descriptors like MRELBP are not sufficiently discriminative. Whereas OCCBM3DELBP and the pre-trained AlexNet model are close in terms of classification accuracy, they are not in terms of time efficiency.



(a) Class 2 predicted as 7 (b) Class 2 predicted as 5 (c) Class 2 predicted as 3 (d) Class 5 predicted as 1 (e) Class 7 predicted as 2

**FIGURE 17. Some incorrectly classified samples for disease detection in tomato leaves.**

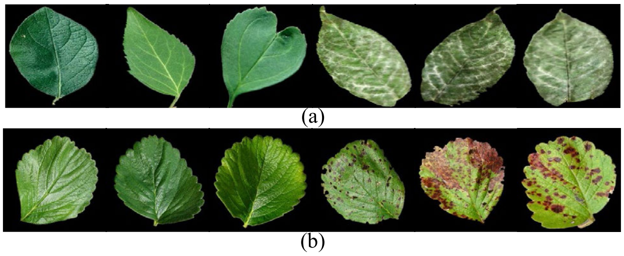


**FIGURE 18. Examples of healthy and non-healthy leaves of (a) cherry (b) strawberry.**

We show in Fig. 19 the estimated time required to classify an image from the PlantVillage database on the spot for

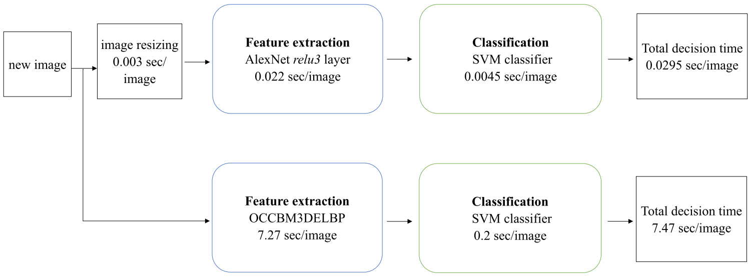

new image
image resizing 0.003 sec/ image
Feature extraction AlexNet relu3 layer 0.022 sec/image
Classification SVM classifier 0.0045 sec/image
Total decision time 0.0295 sec/image
Feature extraction OCCBM3DELBP 7.27 sec/image
Classification SVM classifier 0.2 sec/image
Total decision time 7.47 sec/image

**FIGURE 19.** Time efficiency comparison between OCCBM3DELBP and the pre-trained AlexNet model with relu3 layer referenced for feature extraction (the time was averaged on a random selection of the images from the dataset).

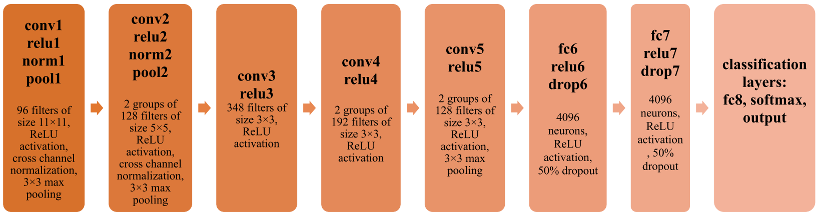

conv1 relu1 norm1 pool1
96 filters of size 11×11, ReLU activation, cross channel normalization, 3×3 max pooling
conv2 relu2 norm2 pool2
2 groups of 128 filters of size 5×5, ReLU activation, cross channel normalization, 3×3 max pooling
conv3 relu3
348 filters of size 3×3, ReLU activation
conv4 relu4
2 groups of 192 filters of size 3×3, ReLU activation
conv5 relu5
2 groups of 128 filters of size 3×3, ReLU activation, 3×3 max pooling
fc6 relu6 drop6
4096 neurons, ReLU activation, 50% dropout
fc7 relu7 drop7
4096 neurons, ReLU activation , 50% dropout
classification layers: fc8, softmax, output

**FIGURE 20.** The default AlexNet architecture.

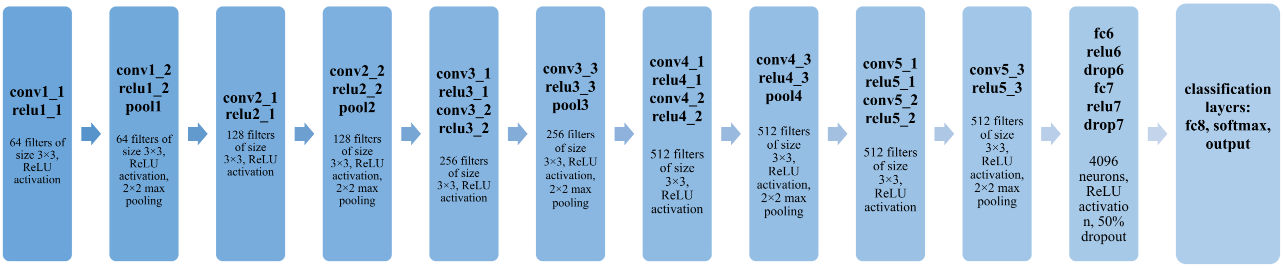

conv1_1 relu1_1
64 filters of size 3×3, ReLU activation
conv1_2 relu1_2 pool1
64 filters of size 3×3, ReLU activation, 2×2 max pooling
conv2_1 relu2_1
128 filters of size 3×3, ReLU activation
conv2_2 relu2_2 pool2
128 filters of size 3×3, ReLU activation, 2×2 max pooling
conv3_1 relu3_1 conv3_2 relu3_2
256 filters of size 3×3, ReLU activation
conv3_3 relu3_3 pool3
256 filters of size 3×3, ReLU activation, 2×2 max pooling
conv4_1 relu4_1 conv4_2 relu4_2
512 filters of size 3×3, ReLU activation
conv4_3 relu4_3 pool4
512 filters of size 3×3, ReLU activation, 2×2 max pooling
conv5_1 relu5_1 conv5_2 relu5_2
512 filters of size 3×3, ReLU activation
conv5_3 relu5_3
512 filters of size 3×3, ReLU activation, 2×2 max pooling
fc6 relu6 drop6 fc7 relu7 drop7
4096 neurons, ReLU activatio n, 50% dropout
classification layers: fc8, softmax, output

**FIGURE 21.** The default Vgg16 architecture.

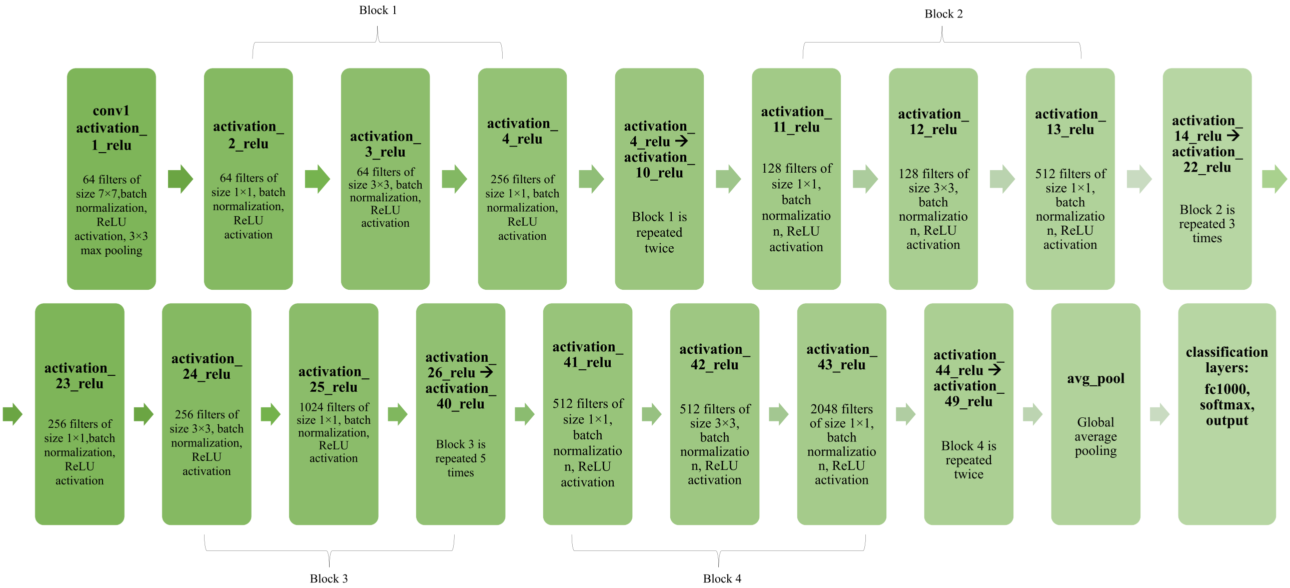

Block 1
Block 2
conv1 activation_ 1_relu
64 filters of size 7×7,batch normalization, ReLU activation, 3×3 max pooling
activation_ 2_relu
64 filters of size 1×1, batch normalization, ReLU activation
activation_ 3_relu
64 filters of size 3×3, batch normalization, ReLU activation
activation_ 4_relu
256 filters of size 1×1, batch normalization, ReLU activation
activation_ 4_relu → activation_ 10_relu
Block 1 is repeated twice
activation_ 11_relu
128 filters of size 1×1, batch normalizatio n, ReLU activation
activation_ 12_relu
128 filters of size 3×3, batch normalizatio n, ReLU activation
activation_ 13_relu
512 filters of size 1×1, batch normalizatio n, ReLU activation
activation_ 14_relu → activation_ 22_relu
Block 2 is repeated 3 times
activation_ 23_relu
256 filters of size 1×1,batch normalization, ReLU activation
activation_ 24_relu
256 filters of size 3×3, batch normalization, ReLU activation
activation_ 25_relu
1024 filters of size 1×1, batch normalization, ReLU activation
activation_ 26_relu → activation_ 40_relu
Block 3 is repeated 5 times
activation_ 41_relu
512 filters of size 1×1, batch normalizatio n, ReLU activation
activation_ 42_relu
512 filters of size 3×3, batch normalizatio n, ReLU activation
activation_ 43_relu
2048 filters of size 1×1, batch normalizatio n, ReLU activation
activation_ 44_relu → activation_ 49_relu
Block 4 is repeated twice
avg_pool
Global average pooling
classification layers: fc1000, softmax, output
Block 3
Block 4

**FIGURE 22.** The default Resnet50 architecture.

the two methods (based on the interpretation of the results in Section IV B).

The total decision time for the proposed approach based on the pre-trained AlexNet relu3 layer model is

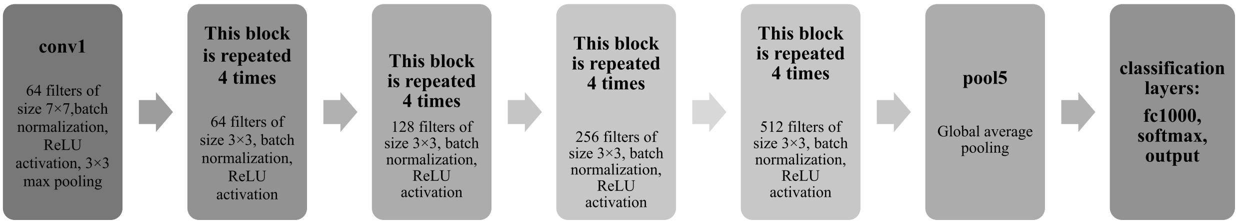

conv1
64 filters of size 7×7,batch normalization, ReLU activation, 3×3 max pooling
This block is repeated 4 times
64 filters of size 3×3, batch normalization, ReLU activation
This block is repeated 4 times
128 filters of size 3×3, batch normalization, ReLU activation
This block is repeated 4 times
256 filters of size 3×3, batch normalization, ReLU activation
This block is repeated 4 times
512 filters of size 3×3, batch normalization, ReLU activation
pool5
Global average pooling
classification layers:
fc1000, softmax, output

**FIGURE 23.** The default Resnet18 architecture.

approximately 30 ms which can be easily considered for any real-time case. For the OCCBM3DELBP operator, in addition to the fact that the feature extraction step takes longer, the classification is longer too. This happens because the feature vector in this situation has 4800 values compared to AlexNet which generates only 384 features, for the same input image.

In the case of the pre-trained AlexNet model with the relu3 layer used for feature extraction, the required memory is 216 MB for storing the pre-trained model. For the SVM train model file, the required memory depends on the number of generated features and the number of training images. For example, in the case of the tomato setup, when considering 13212 training images, 10.2 MB are required to store the SVM train model file for the AlexNet relu3 layer (384 features).

## V. CONCLUSION

We proposed using a deep-learning-based method for texture classification with performance compatible with real-time processing scenarios. We considered using CNNs as feature descriptors rather than end-to-end classifiers and combine them with SVMs. To obtain a relevant classification performance even for small datasets, we based our work on the transfer learning concept and adapted to the task popular CNN models (AlexNet, Vgg16, ResNet) pre-trained on the very large ImageNet object-based dataset. In the experimental section, we considered two datasets: a public one with generic RGB textures (for initial validation of the proposed approach) and a dataset from the applied field of precision agriculture consisting of images with leaves from several plant species and affected by several diseases (for illustrating the applicability of our work).

We analysed the classification results obtained by extracting features from several different layers of the different CNN pre-trained models and using them for describing the textures in the proposed datasets. We showed experimentally that the extraction of features from early convolutional layers is relevant for texture classification as the generated characteristics are more general and not necessarily specific to the task, a result consistent with the theoretical understanding of the CNNs presented in the literature. We compared the results with handcrafted features derived for the same dataset and we concluded that the proposed CNN-based system achieved the most satisfying overall performance (time and classification score). For the PlantVillage dataset, we performed plant species identification and we proposed nine setups in the experimental section for disease detection. We compared the obtained results with the performance achieved using classical machine-learning texture extractors and end-to-end deep-learning techniques. The pre-trained AlexNet model was chosen for feature extraction since it provided a promising performance in the general texture dataset evaluation and exhibited the smallest processing time. Thus, for the PlantVillage dataset, only the pre-trained AlexNet model was employed since the other considered models didn't meet the criteria for real-time processing applications. The proposed architecture (based on the use of the pre-trained AlexNet model on the ImageNet dataset and the selection of the relu3 layer as a descriptor together with an SVM classifier whose parameters were obtained through a grid search) surpasses the other operators in the considered cases in terms of both classification performance and processing times, making it a relevant candidate for real-time processing tasks.

## APPENDIX

The training parameters for the end-to-end AlexNet and Vgg16 networks are given below:

- Solver: Adam [54];
- Mini-batch size: 32 for AlexNet and 16 for Vgg16;
- Number of epochs: 50 for AlexNet and 75 for Vgg16;
- Learning rate: $10^{-4}$;
- L2 regularization factor: $10^{-4}$;
- Gradient decay rate [54]: 0.9;
- Squared gradient decay rate [54]: 0.999.

For running the tests, we used an Intel Core i7-4510U 2.00 GHz processor and NVIDIA GeForce GT 840M 4GB video card. The software code was run in Matlab R2020a.

## REFERENCES

[1] D. I. Patrício and R. Rieder, "Computer vision and artificial intelligence in precision agriculture for grain crops: A systematic review," *Comput. Electron. Agricult.*, vol. 153, pp. 69–81, Oct. 2018, doi: 10.1016/j.compag.2018.08.001.

[2] M. Donatelli, R. D. Magarey, S. Bregaglio, L. Willocquet, J. P. M. Whish, and S. Savary, "Modelling the impacts of pests and diseases on agricultural systems," *Agricult. Syst.*, vol. 155, pp. 213–224, Jul. 2017, doi: 10.1016/j.agsy.2017.01.019.

[3] B. Aglave, *Handbook of Plant Disease Identification and Management*. Boca Raton, FL, USA: CRC Press, 2018. Accessed: Apr. 26, 2021. [Online]. Available: https://www.routledge.com/Handbook-of-Plant-Disease-Identification-and-Management/Aglave/p/book/9781138585478

[4] P. Krithika and S. Veni, "Leaf disease detection on cucumber leaves using multiclass support vector machine," in *Proc. Int. Conf. Wireless Commun., Signal Process. Netw. (WiSPNET)*, Mar. 2017, pp. 1276–1281, doi: 10.1109/WiSPNET.2017.8299969.

[5] R. M. Prakash, G. P. Saraswathy, G. Ramalakshmi, K. H. Mangaleswari, and T. Kaviya, "Detection of leaf diseases and classification using digital image processing," in *Proc. Int. Conf. Innov. Inf., Embedded Commun. Syst. (ICIIECS)*, Mar. 2017, pp. 1–4, doi: 10.1109/ICIIECS.2017.8275915.

[6] S. C. Madiwalar and M. V. Wyawahare, "Plant disease identification: A comparative study," in *Proc. Int. Conf. Data Manage., Analytics Innov. (ICDMAI)*, Feb. 2017, pp. 13–18, doi: 10.1109/ICDMAI.2017.8073478.

[7] N. Ganatra and A. Patel, "A survey on diseases detection and classification of agriculture products using image processing and machine learning," *Int. J. Comput. Appl.*, vol. 180, no. 13, pp. 7–12, Jan. 2018, doi: 10.5120/ijca2018916249.

[8] J. Hang, D. Zhang, P. Chen, J. Zhang, and B. Wang, "Classification of plant leaf diseases based on improved convolutional neural network," *Sensors*, vol. 19, no. 19, p. 4161, Sep. 2019, doi: 10.3390/s19194161.

[9] S. S. Kumar and B. K. Raghavendra, "Diseases detection of various plant leaf using image processing techniques: A review," in *Proc. 5th Int. Conf. Adv. Comput. Commun. Syst. (ICACCS)*, Mar. 2019, pp. 313–316, doi: 10.1109/ICACCS.2019.8728325.

[10] T. Ojala, M. Pietikäinen, and D. Harwood, "A comparative study of texture measures with classification based on featured distributions," *Pattern Recognit.*, vol. 29, no. 1, pp. 51–59, Jan. 1996, doi: 10.1016/0031-3203(95)00067-4.

[11] T. Ojala, M. Pietikäinen, and T. Mäenpää, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 7, pp. 971–987, Jul. 2002, doi: 10.1109/TPAMI.2002.1017623.

[12] L. Liu, S. Lao, P. W. Fieguth, Y. Guo, X. Wang, and M. Pietikäinen, "Median robust extended local binary pattern for texture classification," *IEEE Trans. Image Process.*, vol. 25, no. 3, pp. 1368–1381, Mar. 2016, doi: 10.1109/TIP.2016.2522378.

[13] S. R. Barburiceanu, S. Meza, C. Germain, and R. Terebes, "An improved feature extraction method for texture classification with increased noise robustness," in *Proc. 27th Eur. Signal Process. Conf. (EUSIPCO)*, Sep. 2019, pp. 1–5, doi: 10.23919/EUSIPCO.2019.8902765.

[14] R. M. Haralick, K. Shanmugam, and I. H. Dinstein, "Textural features for image classification," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-3, no. 6, pp. 610–621, Nov. 1973, doi: 10.1109/TSMC.1973.4309314.

[15] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, and A. C. Berg, "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, Dec. 2015, doi: 10.1007/s11263-015-0816-y.

[16] A. Khan, A. Sohail, U. Zahoora, and A. S. Qureshi, "A survey of the recent architectures of deep convolutional neural networks," *Artif. Intell. Rev.*, vol. 53, no. 8, pp. 5455–5516, 2020, doi: 10.1007/s10462-020-09825-6.

[17] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *J. Big Data*, vol. 6, no. 1, p. 60, 2019, doi: 10.1186/s40537-019-0197-0.

[18] Y. Lu and S. Young, "A survey of public datasets for computer vision tasks in precision agriculture," *Comput. Electron. Agricult.*, vol. 178, Nov. 2020, Art. no. 105760, doi: 10.1016/j.compag.2020.105760.

[19] B. Liu, Z. Ding, L. Tian, D. He, S. Li, and H. Wang, "Grape leaf disease identification using improved deep convolutional neural networks," *Frontiers Plant Sci.*, vol. 11, p. 1082, Jul. 2020, doi: 10.3389/fpls.2020.01082.

[20] D. Han, Q. Liu, and W. Fan, "A new image classification method using CNN transfer learning and web data augmentation," *Expert Syst. Appl.*, vol. 95, pp. 43–56, Apr. 2018, doi: 10.1016/j.eswa.2017.11.028.

[21] M. Shaha and M. Pawar, "Transfer learning for image classification," in *Proc. 2nd Int. Conf. Electron., Commun. Aerosp. Technol. (ICECA)*, Mar. 2018, pp. 656–660, doi: 10.1109/ICECA.2018.8474802.

[22] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman, "Return of the devil in the details: Delving deep into convolutional nets," 2014, *arXiv:1405.3531*.

[23] M. Cimpoi, S. Maji, and A. Vedaldi, "Deep filter banks for texture recognition and segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 3828–3836, doi: 10.1109/CVPR.2015.7299007.

[24] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 580–587, doi: 10.1109/CVPR.2014.81.

[25] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, "Learning and transferring mid-level image representations using convolutional neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 1717–1724, doi: 10.1109/CVPR.2014.222.

[26] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "CNN features Off-the-shelf: An astounding baseline for recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, Jun. 2014, pp. 512–519, doi: 10.1109/CVPRW.2014.131.

[27] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, vol. 25, Dec. 2012, pp. 1097–1105, doi: 10.1145/3065386.

[28] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*.

[29] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 1–9, doi: 10.1109/CVPR.2015.7298594.

[30] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778, doi: 10.1109/CVPR.2016.90.

[31] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, and A. Vedaldi, "Describing textures in the wild," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 3606–3613, doi: 10.1109/CVPR.2014.461.

[32] P. Napoletano, "Hand-crafted vs learned descriptors for color texture classification," in *Computational Color Imaging*, vol. 10213. Cham, Switzerland: Springer, 2017, pp. 259–271, doi: 10.1007/978-3-319-56010-6_22.

[33] V. Andrearczyk and P. F. Whelan, "Using filter banks in convolutional neural networks for texture classification," *Pattern Recognit. Lett.*, vol. 84, pp. 63–69, Dec. 2016, doi: 10.1016/j.patrec.2016.08.016.

[34] R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla, and J. Sivic, "NetVLAD: CNN architecture for weakly supervised place recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 5297–5307, doi: 10.1109/CVPR.2016.572.

[35] X. Dai, J. Y.-H. Ng, and L. S. Davis, "FASON: First and second order information fusion network for texture recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 6100–6108, doi: 10.1109/CVPR.2017.646.

[36] T.-Y. Lin, A. RoyChowdhury, and S. Maji, "Bilinear convolutional neural networks for fine-grained visual recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 6, pp. 1309–1322, Jun. 2018, doi: 10.1109/TPAMI.2017.2723400.

[37] S. Jiang, R. Hartley, and B. Fernando, "Kernel support vector machines and convolutional neural networks," in *Proc. Digit. Image Comput., Techn. Appl. (DICTA)*, Dec. 2018, pp. 1–7, doi: 10.1109/DICTA.2018.8615840.

[38] F. J. Huang and Y. LeCun, "Large-scale learning with SVM and convolutional for generic object categorization," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, vol. 1, Jun. 2006, pp. 284–291, doi: 10.1109/CVPR.2006.164.

[39] X.-X. Niu and C. Y. Suen, "A novel hybrid CNN–SVM classifier for recognizing handwritten digits," *Pattern Recognit.*, vol. 45, no. 4, pp. 1318–1325, Apr. 2012, doi: 10.1016/j.patcog.2011.09.021.

[40] C. J. C. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining Knowl. Discovery*, vol. 2, no. 2, pp. 121–167, Jun. 1998, doi: 10.1023/A:1009715923555.

[41] *Outex Texture Database*. Accessed: Jan. 11, 2018. [Online]. Available: https://www.outex.oulu.fi/index.php?page=classification

[42] D. P. Hughes and M. Salathe, "An open access repository of images on plant health to enable the development of mobile disease diagnostics," 2015, *arXiv:1511.08060*.

[43] S. P. Mohanty, D. P. Hughes, and M. Salathé, "Using deep learning for image-based plant disease detection," *Frontiers Plant Sci.*, vol. 7, p. 1419, Sep. 2016, doi: 10.3389/fpls.2016.01419.

[44] M. Cimpoi, S. Maji, I. Kokkinos, and A. Vedaldi, "Deep filter banks for texture recognition, description, and segmentation," *Int. J. Comput. Vis.*, vol. 118, no. 1, pp. 65–94, May 2016, doi: 10.1007/s11263-015-0872-3.

[45] S. Barburiceanu, R. Terebes, and S. Meza, "Grape leaf disease classification using LBP-derived texture operators and colour," in *Proc. IEEE Int. Conf. Automat., Quality Test., Robot. (AQTR)*, May 2020, pp. 1–6, doi: 10.1109/AQTR49680.2020.9130019.

[46] M. Islam, A. Dinh, K. Wahid, and P. Bhowmik, "Detection of potato diseases using image segmentation and multiclass support vector machine," in *Proc. IEEE 30th Can. Conf. Electr. Comput. Eng. (CCECE)*, Apr. 2017, pp. 1–4, doi: 10.1109/CCECE.2017.7946594.

[47] C. S. Hlaing and S. M. M. Zaw, "Model-based statistical features for mobile phone image of tomato plant disease classification," in *Proc. 18th Int. Conf. Parallel Distrib. Comput., Appl. Technol. (PDCAT)*, Dec. 2017, pp. 223–229, doi: 10.1109/PDCAT.2017.00044.

[48] H. Durmus, E. O. Gunes, and M. Kirci, "Disease detection on the leaves of the tomato plants by using deep learning," in *Proc. 6th Int. Conf. Agro-Geoinformatics*, Aug. 2017, pp. 1–5, doi: 10.1109/Agro-Geoinformatics.2017.8047016.

[49] K. Yamamoto, T. Togami, and N. Yamaguchi, "Super-resolution of plant disease images for the acceleration of image-based phenotyping and vigor diagnosis in agriculture," *Sensors*, vol. 17, no. 11, p. 2557, Nov. 2017, doi: 10.3390/s17112557.

[50] W. Zeng, M. Li, J. Zhang, L. Chen, S. Fang, and J. Wang, "High-order residual convolutional neural network for robust crop disease recognition," in *Proc. 2nd Int. Conf. Comput. Sci. Appl. Eng. (CSAE)*, New York, NY, USA, Oct. 2018, pp. 1–5, doi: 10.1145/3207677.3277952.

[51] H. F. Pardede, E. Suryawati, R. Sustika, and V. Zilvan, "Unsupervised convolutional autoencoder-based feature learning for automatic detection of plant diseases," in *Proc. Int. Conf. Comput., Control, Informat. Appl. (IC3INA)*, Nov. 2018, pp. 158–162, doi: 10.1109/IC3INA.2018.8629518.

[52] E. Suryawati, R. Sustika, R. S. Yuwana, A. Subekti, and H. F. Pardede, "Deep structured convolutional neural network for tomato diseases detection," in *Proc. Int. Conf. Adv. Comput. Sci. Inf. Syst. (ICACSIS)*, Oct. 2018, pp. 385–390, doi: 10.1109/ICACSIS.2018.8618169.

[53] S. Baranwal, S. Khandelwal, and A. Arora, "Deep learning convolutional neural network for apple leaves disease detection," in *Proc. Int. Conf. Sustain. Comput. Sci., Technol. Manage. (SUSCOM)*. Jaipur, India: Amity Univ. Rajasthan, Feb. 2019, pp. 260–267, doi: 10.2139/ssrn.3351641.

[54] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.

**STEFANIA BARBURICEANU** was born in Sinaia, Romania, in 1993. She received the B.Sc. degree in electronics and telecommunications from the Technical University of Cluj-Napoca, Romania, in 2016, the M.Sc. double diploma degree in image and signal processing from the University of Bordeaux, France, and from the TUC-N, in 2018. She is currently pursuing the Ph.D. degree.

Her Ph.D. thesis is focused on texture feature extraction and classification. From 2018 to 2020, she was a Research Assistant with the Communications Department, TUC-N, where she has been an Assistant Professor, since 2020. Her work experience also includes being a member in several research projects involving image classification problems. Her research interests include image classification, machine-learning, and artificial intelligence domains. She is a EURASIP Member.

**SERBAN MEZA** (Member, IEEE) received the B.Sc. degree *(magna cum laude)* from the Faculty of Applied Electronics and Telecommunications, Technical University of Cluj-Napoca, Romania, in 2007, and the Ph.D. degree from the Technical University of Cluj-Napoca, in 2013.

His Ph.D. thesis is focused on contributions to 2D and 3D imaging. He pursued an academic career in the field of image and video engineering and multimedia applications after interning in several industrial Research and Development departments, such as GrassValley (The Netherlands) where, he contributed to projects related to machine vision and image processing. He is currently an Associate Professor with the Communications Department, TUC-N.

**BOGDAN ORZA** received the bachelor's, master's, and Ph.D. degrees in electronics and telecommunications from the Technical University of Cluj-Napoca, in 1996, 1997, and 2005, respectively. He acts as the Coordinator of the Multimedia Systems and Applications Laboratory of the Communications Department of the TUC-N. He focuses both at education and research level on information management, video and television engineering, digital image processing, multimedia systems and applications, E-learning solutions, database design, and programming. He was being actively involved in national and regional coordination of over 20 projects.

**RAUL MALUTAN** was born in Gherla, Romania, in September 1980. He received the B.Sc. degree in telecommunications from the Technical University of Cluj-Napoca, Romania, in 2004, and the joint Ph.D. degree in electronics and telecommunications from the Technical University of Cluj-Napoca, Romania, and in informatics from the Polytechnic University of Madrid, Spain, in 2010.

He is currently an Associate Professor with the Communications Department, TUC-N. He has authored more than 40 research articles published in ISI indexed journals, as book chapters and in peer-reviewed conferences. He is the coauthor of two Romanian patents. His research interests include high-order statistics, genomic processing, image classification, and biometry.

**ROMULUS TEREBES** (Member, IEEE) was born in Livada, Romania. He received the B.Sc. degree in electronics and telecommunications from the Technical University of Cluj-Napoca (TUC-N), Cluj-Napoca, Romania, in 1994, and the Ph.D. degree from the University of Bordeaux 1, Talence, France, and TUC-N (co-advised Ph.D. thesis), in 2004. The subject of his Ph.D. thesis dealt with partial-derivative-equation-based image processing techniques.

He is currently an IEEE Signal Processing Society Member and a Professor with the Department of Communications, Faculty of Electronics, Telecommunications and Information Technology, TUC-N. His publication list includes 48 articles and proceeding papers indexed in the Web of Science, the Core Collection, 41 being also IEEE Xplore indexed. His research interests include the area of image processing and computer vision.