Research Article

Optimal Deep Learning Model for Olive Disease Diagnosis Based on an Adaptive Genetic Algorithm

Hamoud Alshammari,1 Karim Gasmi,2 Moez Krichen,3 Lassaad Ben Ammar,4 Mohamed Osman Abdelhadi,2 Ammar Boukrara,2 and Mahmood A. Mahmood1,5

1Department of Information Systems, College of Computer and Information Sciences, Jouf University, Jouf, Saudi Arabia
2Department of Computer Science, College of Arts and Sciences at Tabarjal, Jouf University, Jouf, Saudi Arabia
3Faculty of CSIT, Al-Baha University, Saudi Arabia & ReDCAD Laboratory, University of Sfax, Tunisia
4College of Sciences and Humanities, Prince Sattam Bin Abdulaziz University, Al-Kharj, Saudi Arabia
5Department of Information Systems and Technology, FGSSR, Cairo University, Egypt

Correspondence should be addressed to Karim Gasmi; kgasmi@ju.edu.sa

Received 3 February 2022; Revised 21 February 2022; Accepted 24 February 2022; Published 14 March 2022

Academic Editor: Kuruva Lakshmanna

Copyright © 2022 Hamoud Alshammari et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Though many researchers have studied plant leaf disease, the timely diagnosis of diseases in olive leaves still presents an indisputable challenge. Infected leaves may display different symptoms from one plant to another, or even within the same plant. For this reason, many researchers studied the effects of those diseases on, at most, two plants. Since olive crops are affected by many pathogens, including bacteria wilt, olive knot, Aculus olearius, and olive peacock spot, the development of an efficient algorithm to detect the diseases was challenging because the diseases could be defined in many different ways. For this purpose, we introduce an optimal deep learning model for diagnosing olive leaf diseases. This approach is based on an adaptive genetic algorithm for selecting optimal parameters in deep learning model to provide rapid diagnosis. To evaluate our approach, we applied it in three famous deep learning models. For the comparative evaluation, we also tested other well-known machine learning methods. The experimental results presented in this paper show that our model outperformed the other algorithms, achieving an accuracy of approximately 96% for multiclass classification and 98% for binary classification.

1. Introduction

Nowadays, olive cultivation in some Middle Eastern countries depends on the latest scientific technology. Approximately 80% of olive cultivation in those countries focuses on olive oil production and the remaining 20% on table olives [1]. Olive crops are affected by many pathogens and deficiencies, including bacteria wilt, olive knot, bacterial spot, bacterial leaf blight, angular leaf spot, verticillium wilt, olive leaf spot, Aculus olearius, leaf mould, olive fruit flies, olive bark beetles, olive borers, and olive moths. Aculus olearius, leaf spot, and leaf mould can be observed visually on the host olive’s leaves [2].

Plant diseases have a great effect on crop yield, with approximately 80% healthy products and losses of 20% [3]. For instance, olive leaf spot and peacock leaf spot produce distinctive symptoms, such as dark green to black spots surrounded by a yellow halo on the leaves. Aculus olearius takes different shapes during different stages of the leaf life cycle (bud, young, and old): rust stains appear during the bud stage; the leaves develop malformations during the young stage; and dark, collapsed sections and yellow spots appear on the middle and tip sections of the leaves during the old stage [4].

In recent years, the growth of computer hardware, especially graphical processing unit (GPU)-embedded processors, has assisted in the development of artificial intelligence, machine learning techniques [5, 6], deep learning [7, 8], and IoT [9, 10]. Many researchers have used these advances to study the detection and diagnosis of olive leaf diseases.
However, the development of the disease algorithm was challenging because the diseases could be defined in many different ways. Infected leaves may display different symptoms from one plant to another, or even within the same plant. For this reason, many researchers studied the effects of those diseases on, at most, two plants [11]. Choosing the best neural network model requires careful consideration of a number of criteria, including the training data and feature extraction and the hyperparameters used. Even the tiniest changes in hyperparameters and network designs may have a large and often unanticipated impact on the ultimate performance of neural networks trained for complicated tasks like olive categorization. Any deep learning procedure needs hyperparameter tweaking. To solve this issue, an optimum deep neural network-based model for the categorization of olive diseases is proposed in this paper. In the proposed approach, olive pictures are used as inputs and an adaptive optimization algorithm “genetic algorithm” produces accurate results as outputs. The purpose of this research is to determine the ideal hyperparameters for deep learning architectures to obtain best results. Using olive leaf images meeting specified criteria, our goal was to find the optimal response time for categorizing olive diseases with a higher accuracy score. Deep learning and machine learning models could be used to solve the classification problem without considering response time. A traditional classification model usually proceed in two steps: training and testing. During the training phase, olive leaf image samples were used to generate a supervised model. The testing phase evaluated the performance of the established model by processing image samples without their labels. Our contributions are, but not limited to, (i) olive diseases recognizing and (ii) hyperparameter selection for deep learning model.

The rest of the paper is organized as follows. Section 2 presents related works, while Section 3 describes our materials and methods in detail. Experimental results are discussed in Section 4, and Section 5 concludes this work.

2. Background

This section presents a brief description of the most known deep learning models. Then, it discusses related works in the field of classification and diagnosis plant disease.

2.1. Deep Learning Models. Mathematical models that replicate the general concepts of the human brain through interconnecting neurons and synapses are known as artificial neural networks (ANNs). Their primary trait is their ability to be taught during the supervised learning process. During that process, neural networks are built to model some systems using existing data containing specific matches to the inputs and outputs to be modelled. CNNs [12] evolved from conventional ANNs and focus primarily on applications with repeating patterns, especially image recognition, in different areas of the modeling space [13]. Depending on the architecture that a CNN is made of, several popular CNN models can be identified. We present hereafter the most famous ones.

(i) AlexNet is one of the CNN models. The network architecture of AlexNet is contained in eight layers; the first five are convolutional layers, and the last three are fully connected layers. It uses the thanh and sigmoid functions to improve the training performance of activation function [12]. Figure 1 represents an example of AlexNet model.

(ii) Resnet (the residual neural network) is one of the ANN models. It depends on skipped connections to avoid the vanishing gradients problem, which occurs when training an ANN with gradient-based learning methods and backpropagation. The skipped connections were implemented on double- or triple-layer skips with a weight matrix that was used to learn the skip weights [14]

(iii) DenseNet (the dense convolutional network) depends on several parallel skips, which occur through a direct connection between any two layers with the same feature map size. DenseNet requires significantly fewer parameters and less computation to achieve state of the art performances [15]. DenseNet is composed of dense blocks as shown below. Within those blocks, the layers are densely connected together: Each layer gets the input from previous layers output feature maps, as shown in Figure 2
whereas DensNet makes use of the characteristics of each and every previous convolutional block in the training set. The same theory that unites ResNet and DenseNet is that they both link to the feature maps of all preceding convolutional blocks, which is the same philosophy that unites ResNet and DenseNet. According to the state of the art, the performance of each architecture of existing deep learning models depends on several criteria such as dataset size and quality and model parameters. For this reason, we chose to apply our optimal deep learning model on the different models. In what follows, we describe the different blocks of our proposal.

2.2. Related Works. Most of the time, farmers are not able to harvest a good crop, which leads to lower income. This is due to the lack of nutrients, soil moisture, and temperature fluctuations. The frequent occurrence of plant diseases also affects the quality and quantity of the harvest. Smart agriculture is a unique idea that uses IoT devices [16] to collect data about agricultural landscapes. This technology allows any device to send or receive data to a server over the Internet. IoT technology can maximize resource efficiency and agricultural yields while minimizing operating costs. This technology allows farmers to monitor their crops without being physically present in the field [17]. Using cloud computing as the backbone [18], they investigated several popular applications of IoT sensor-monitoring network technologies for agriculture. All data collected by this technology must be analysing for several task, such as plant disease detection. In recent years, many researchers have relied on machine learning to recognize plant disease and carry out manual or automatic feature extractions for plant images.

Fredj et al. [8] applied manual feature extraction, using a gray-level cooccurrence matrix (GLCM) to determine which images showed diseased tomato leaves. The images were obtained manually from a dataset containing 800 tomato leaf images, which were labeled either “healthy” or “infected.” This dataset was split into training and testing sets. An accuracy score of 99.83% was obtained using a support vector machine (SVM).

Shanwen et al. [19] grouped cucumber leaf images, taken with a digital camera in the agriculture demonstration zone of the Northwest Agriculture and Forestry University of China. They proposed a new segmentation method for leaf images by combining the superpixel clustering and EM algorithms. The computation time for the new segmentation method was lower than it was for the EM algorithm, k-means, and fuzzy c-mean.

Ramesh et al. [20] employed a random forest machine learning algorithm in their study to detect whether images depicted healthy or infected leaves. This method achieved an accuracy score of 70.14%. Kusumo et al. [21] utilized linear SVM and RGB as an image processing feature in their study to detect corn plant disease. The data was grouped from the Plant Village dataset, which contained 3,823 images. There were four different class labels: “gray leaf spot,” “common rust,” “healthy,” and “northern leaf blight.” This method produced an accuracy score of approximately 88%.

Kaur et al. [22] obtained grape leaf images from the Plant Village website. The data were categorized into four classes named “black rot,” “esca,” “leaf blight,” and “healthy.” They used fractional-order Zernike for feature extraction and SVM for classification. The accuracy score obtained with this method was 97.34%. Mustafa et al. [23] collected data on herb plants manually from a sample of actual leaves, with a total of 1,000 leaf images. These were divided equally among ten different herb plants, and each plant was labeled either “healthy” or “diseased.” The researchers employed hybrid models combining SVM, probabilistic neural network (PNN), Naive Bayes (NB), and fuzzy inference system (FIS). The accuracy score obtained with this method was 95.4%.

Nowadays, researchers prefer convolutional neural network (CNN) techniques for deep learning studies to detect...
and classify plant disease. Uğuz and Uysal [24] employed deep CNNs to classify olive leaf diseases; the data for this study were collected manually from Denizli, Turkey, during the spring and summer. They proposed a CNN architecture model. Their data consisted of 3,400 olive leaf images divided into three different classes named “Aculus olearius,” “peacock spot,” and “healthy.” The accuracy score obtained with this method was 95%.

Gokulnath and Usha [25] collected data on maize crop diseases from the Plant Village (3,852 images) and PDD (164 images) databases. The Plant Village images were sorted into four different classes named “Cercospora zeamaydis,” “Puccinia sorghi,” “Exserohilum turcicum,” and “healthy,” while the PDD images were sorted into five classes named “downy mildew,” “eyespot,” “healthy,” “northern leaf blight,” and “southern rust.” The researchers proposed a Boosted-DEPICT model for clustering maize crop disease images. This model achieved accuracy scores of 97.73% for the Plant Village database and 91.25% for the PDD database. Guo et al. [26] collected 1,000 leaf photos from the Plant Photo Bank of China (PPBC). The images were sorted into four different classes named “black rot,” “bacteria plaque,” “rust,” and “healthy.” The researchers proposed a mathematical model for plant disease detection and recognition based on deep learning. The accuracy score obtained with this method was 83.75%. Ramar et al. [27] proposed a CNN-based modified LeNet to classify maize leaf disease. They employed the Plant Village dataset. The images were sorted into four different classes (three diseases and one healthy). The accuracy score obtained with this model was 97.89%. Mohanty et al. [28] employed deep learning models to identify 14 crop species and 26 diseases. They utilized three different versions of the Plant Village dataset. The accuracy score obtained with the deep learning models was 99.35%. Goluguri et al. [29] proposed a hybrid CNN with long short-term memory (LSTM) to identify rice diseases. The accuracy score obtained with the model was 97.5%.

Our literature review reveals that the diagnosis of plant leaf disease presents significant challenges, though several approaches tried to solve these through various means; other works are presented in Table 1. We quote, for example, the traditional techniques of image processing, machine learning algorithms, and feature extraction. As for the olive leaf diseases, few works have been established. They present encouraging results. However, many more research works are needed to improve the obtained results. Starting from this report, the present paper investigates the use of a transfer deep learning technique based on a genetic algorithm to optimize the identification of olive leaf diseases.

3. Proposed Method for Olive Disease Recognition

This section details our approach for classifying olive leaf images to “healthy,” “Aculus olearius disease,” or “peacock spot disease.” It is based on well-known deep learning models such as AlexNet, ResNet, and DenseNet.

As per Figure 3, the architecture of the approach goes through 3 stages. The first stage performs a preprocessing of the dataset to improve the quality of the images and remove the noise using the noise filter algorithm. The second stage executes an optimal deep learning parameter selection process. Finally, the third stage applies the aforementioned CNN models to the preprocessed dataset. It is worth mentioning that the main objective behind using this variety of models is to carry out several experiments and find out the best results.

3.1. Dataset Preprocessing. In this phase, the median noise filtering algorithm is utilized to enhance the images. The median filter replaces a pixel with the median of the gray levels in the vicinity of the pixel and provides a better method of removing or reducing the noise within captured images [43]. Geometric transformations such as rotation,
shifts, scaling, and flipping are also used to alter the images in a data collection. Data augmentation not only improves the generalizability of models, or in other words, models that prevent overfitting, but it also improves the results of imbalanced classification issues.

This study also utilized color histogram [44], moment shape [45], and hierarchical features [46] to apply feature extraction for machine learning techniques. The color histogram is based on RGB or HSV, which computes the number of pixels within the different colors of a given image. Moment shape depends on the concept of the moment in mathematics, which is utilized to analyse the contour and region of an image. The hierarchical feature uses a basic assumption in which each input can be divided into parts in hierarchical form according to the number of layers.

3.2 Hyperparameter Optimization. The main objective here is to obtain the optimum settings of the deep learning model by finding the best values for its hyperparameters. It is worth mentioning that doing such a step manually is usually time-consuming, increasing the complexity of the algorithm to be applied. Therefore, we resort to the genetic algorithm that is known as the best tool for solving similar problem. This process is presented in Algorithm 1.

In a genetic algorithm (GA) [47], a population of possible solutions to optimization problems is given a set of properties, which can be mutated and altered. Because of their constant size, the portions of these genetic representations are simple to align, making crossover processes simpler. For each generation, each person in the population’s fitness is assessed; the fitness is typically the value of the objective
function in the optimization problem being addressed. Adaptive fitness functions are constructed over a genetic representation and quantify the quality of the represented solution specified in Equation (1):

\[
\text{Fitness function} = w_1 \cdot \text{acc} + w_2 \cdot \frac{1}{\text{epoch}},
\]

where \( w_1 + w_2 = 1 \), acc represents the accuracy rate, and epoch represents the epoch number.

The three main genetic operators employed to form the next generation in a GA are reproduction, crossover, and mutation. The initial population was created by encoding the epoch number and batch size from decimal to binary, and each individual is a representation of epoch number and batch size choice. For the other steps of the GA, crossover or mutation will use the binary representations of each epoch number and batch size. The general GA process is presented in Figure 4.

Figure 5 shows the encoding technique for one individual in our evolutionary algorithm. As a genetic algorithm, each individual is encoded as a binary string, where the bits in each string indicate the solution’s unique traits. Having a huge population increases the search space and, as a result, the computing burden. As a result, there are 100 people in the initial population. Both the batch size and the epoch number hyperparameters use eight binary genes to represent their respective values. The epoch number, for example, is divided into a series of numeric digits before being encoded into the matching binary digits. Figure 5 shows the 16 binary genes used to represent an individual. We utilized a method called “uniform crossover,” in which each bit in the new offspring was randomly picked from the two parents by choosing parent gene values at each designated site. After selecting two parents, we produce a random binary vector with a length of each parent’s choice. A gene from the first parent is used to make the child if the value is 1. Otherwise, the second parent’s gene is chosen. The mutation procedure is subsequently applied to all of the crossover offspring. We employed a technique known as probabilistic bitwise mutation, in which a bit is randomly switched from 0 to 1 or vice versa. As the population grows, the likelihood of a mutation increases, which promotes genetic variety and, ultimately, global optimization. But the use of crossover and selection does not ensure optimality. By creating kids that are genetically distinct from their parents, mutations are exploited to mitigate this issue and avoid the local search space. Individuals in the population are randomly chosen to participate in a series of tournaments in which they compete against one another. Each tournament’s crossover operator selects a single winner. After selecting two people, the crossover operator selects the one with the highest fitness value.

3.2.1. Crossover. The crossover is the most important operator working on the population of parents, and it is applied with a particular probability, which is denoted by the crossover rate \( P_c \) (typically close to unity). The crossing procedure consists of selecting two people who are represented by their chains of genes that have been randomly picked from the general population and defining one or more crossing locations. Afterwards, the new offspring is generated by interchanging distinct bits of each string between each other.

Let \( G_p \) and \( G_o \) be the two selected parent chromosomes, which are represented, respectively, as follows:

\[
G_p = G_{p1}^1, G_{p2}^1, \ldots G_{pn}^1; \quad G_o = G_{o1}^1, G_{o2}^1, \ldots G_{on}^1.
\]
Then,

\[ G_{\xi} = \begin{cases} G_i \text{if } 1 < i < k, \\ G_{\omega} \text{else,} \end{cases} \]  

(3)

with

\[ k = \text{random}(1, n-1), \; 1 < k < n - 1. \]  

(4)

3.2.2. Mutation. This process guards against the early loss of valuable information by genetic algorithms. It enables the introduction of certain information into the population that could otherwise be lost during the crossover procedure. So it contributes to the preservation of diversity, which is necessary for a thorough examination of the subject field. It is possible to apply the mutation operator with a given probability, known as the mutation rate \( P_{\text{m}} \), which is commonly between 0.05 and 0.10. Changing a one bit at bit zero, and vice versa, for each bit of the string with the probability \( P_{\text{m}} \) is referred to as a mutation in binary coding.

Let \( G_\alpha \) be the parent chromosome,

\[ G_\alpha = G^1_\alpha, G^2_\alpha, \ldots, G^n_\alpha. \]  

(5)

Then, \( G^n_\alpha = 1 - G^k_\alpha \), with \( k = \text{random}(1, n-1), \; 1 < k < n - 1. \)

4. Experiment Results and Discussion

In this section, we present the results of the experimental tests performed on the model presented in this paper. We also compare the results of different machine learning techniques using our modified deep learning model. This model was implemented in Python with the help of Google Colab deep learning server.

To apply our proposed model, we carried out several series of experiments on machine learning methods. Our model consists of three scenarios:

(1) Scenario 1: the classification task sorts the images into two classes: “sick” and “healthy”

\begin{figure}
\centering
\includegraphics[width=\textwidth]{fig4}
\caption{Genetic algorithm process.}
\end{figure}

\begin{figure}
\centering
\includegraphics[width=\textwidth]{fig5}
\caption{Coding of a chromosome.}
\end{figure}
(2) Scenario 2: the classification task sorts infected olive leaf images into two disease classes: “olive peacock spot” and “Aculus olearius”.

(3) Scenario 3: the third scenario concerns the transition from binary classification to multiclass classification. We have sorted our images into three classes: “healthy,” “peacock spot,” and “Aculus olearius.”

4.1. Dataset Description and Evaluation Metrics. To evaluate our proposed optimal deep learning-based model, we employed a dataset collected during the spring and summer. Our dataset contained 3,400 olive leaf images (https://github.com/sinanuguz/CNN_olive_dataset), which were grouped into three different classes—leaves infected with Aculus olearius, leaves infected with olive peacock spot, and healthy leaves—with the assistance of an agricultural engineer who is an expert in the field. In Figure 6, we give an example of a sample picture and the different types of the olive diseases.

Later, the dataset was split in two groups, with 80% used for training and 20% for testing, as shown in Table 2. The irregularities that can arise in the distribution of datasets when they are separated into training and testing sets may have a negative effect on CNN model results. To avoid this problem, we used the k-fold cross-validation method when forming the training and testing sets.

To evaluate the performance of the proposed model and the other different machine learning methods, we used the standard metrics employed in large-scale medical image sets for the estimation of classification tasks, such as accuracy, precision, recall, F-measure, ROC curve, and AUC.

We chose all these metrics to avoid the problem of unbalanced collection, which occurs when the problem is highly imbalanced, and the accuracy score is calculated simply by predicting that all observations belong to the majority class, producing unreliable results. Thus, we had to add other metrics to ensure that the classification task would produce credible results:

\[
\text{Precision} = \frac{TP}{TP + FP}, \\
\text{Recall} = \frac{TP}{TP + FN}, \\
F1 - \text{score} = \frac{2}{\left(\frac{1}{P}\right) + \left(\frac{1}{R}\right)}, \\
\text{ROC} = \frac{TPR}{TP + FN}, \\
\text{AUC} = \frac{TPR + FN}{FP + TN},
\]

where TP stands for true positive, FP stands for false positive, P stands for precision, R stands for recall, TPR stands for true positive rate, and FPR stands for false positive rate. Comparing the outcomes of various machine learning methods is also part of our research; for that,
we use sklearn (https://scikit-learn.org/stable/) as a python package and keras application (https://www.tensorflow.org/api_docs/python/tf/keras/applications) for deep learning models.

### 4.2. Machine Learning Results and Discussion

In this part, we analyse and compare the performance of different supervised machine learning classifiers such as logistic regression, linear discriminant analysis (LDA), KNN, decision tree, random forest, Naive Bayes, and SVM in order to determine the best classifier for olive disease diagnosis. The purpose of this comparison is to determine the performance of different machine learning algorithms in classifying olive diseases. In Tables 3 and 4, we present the binary classification results of scenario 1 and scenario 2.

In the first scenario, the highest results in terms of average accuracy values were achieved by random forest and SVM, which scored 0.92 and 0.88, respectively. It should be noted that the highest overall accuracy was computed as 0.92 when the random forest classifier outperforms all classification algorithms on the rate of accuracy. This is justified because random forest as a machine learning algorithm for classification works effectively when the predicted variable is binary, assumes that all predictors are independent of each other, and assumes that the data is free of missing values. Furthermore, the lowest overall accuracy was predicted with Naive Bayes classifier method as 0.81. On the other hand, in the second scenario, we see that the outcome obtained by random forest is always better for classification into two classes: “olive peacock spot” and “Aculus olearius.” Thus, the highest classification accuracy for random forest classifier method was predicted as 0.95. The results noticeably indicated that random forest classifier outperformed all classification methods in the two scenario.

Finally, we have move from a binary classification based on two classes to a multiclass classification. We found that the best results were obtained by the random forest model at about 0.89 for the third scenario. Therefore, it could be recommended to use random forest classifier in both binary classification and multiclass classification process. However, this outcome was not enough. To make our evaluation more relevant to practitioners, we focus not only on machine learning methods that are in general known to have strong performance in classification but also include other advanced deep learning methods for optimal classification. The most popular deep learning algorithms such as AlexNet, DensNet, and ResNet are considered. In addition, we proposed to make some changes to their initial architecture by opting an adaptive genetic algorithm for the hyperparameter selection process. We argue that this change is susceptible to strongly improve the quality of the results. To improve these

| Table 3: Machine learning method evaluation (scenario1). |
|----------------------------------------------------------|
|              | Accuracy | roc_auc | Scenario 1 | Precision | Recall | fBeta |
| Logistic regression | 0.87    | 0.92    | 0.89      | 0.95      | 0.93   |
| Linear discriminant analysis | 0.86    | 0.89    | 0.88      | 0.93      | 0.92   |
| KNN           | 0.89    | 0.92    | 0.91      | 0.93      | 0.92   |
| Decision tree | 0.85    | 0.83    | 0.90      | 0.89      | 0.89   |
| Random forest | 0.92    | 0.97    | 0.91      | 0.98      | 0.96   |
| Naive Bayes   | 0.70    | 0.89    | 0.96      | 0.59      | 0.64   |
| SVM           | 0.88    | 0.92    | 0.88      | 0.96      | 0.94   |

| Table 4: Machine learning method evaluation (scenario2). |
|----------------------------------------------------------|
|              | Accuracy | roc_auc | Scenario 2 | Precision | Recall | fBeta |
| Logistic regression | 0.88    | 0.96    | 0.89      | 0.93      | 0.92   |
| Linear discriminant analysis | 0.86    | 0.91    | 0.88      | 0.91      | 0.90   |
| KNN           | 0.89    | 0.94    | 0.89      | 0.94      | 0.93   |
| Decision tree | 0.89    | 0.88    | 0.91      | 0.91      | 0.91   |
| Random forest | 0.95    | 0.99    | 0.96      | 0.96      | 0.96   |
| Naive Bayes   | 0.81    | 0.92    | 0.95      | 0.75      | 0.78   |
| SVM           | 0.89    | 0.96    | 0.89      | 0.94      | 0.93   |

| Table 5: GA parameters. |
|-------------------------|
| **GA property** | **Value/method** |
| Size of generation | 100 |
| Initial population size | 30 |
| Selection method | Tourni |
| Number of crossover points | 1 |
| Crossover probability | 0.9 |
| Mutation method | Uniform mutation |
| Mutation probability | 0.05 |

we use sklearn (https://scikit-learn.org/stable/) as a python package and keras application (https://www.tensorflow.org/api_docs/python/tf/keras/applications) for deep learning models.
results, we propose an optimal deep learning model based on the adaptive genetic algorithm version.

4.3. Optimal Deep Learning Results and Discussion. In this section, we will detail the comparison between the different deep learning models: AlexNet, ResNet, and DenseNet. We utilized the adapted version of the genetic algorithm (GA) with three deep learning models to minimize the epoch number. Table 5 presents the parameters of the GA.

Tables 6 and 7 present the results obtained by the optimal deep learning models based on an adaptive genetic algorithm for two binary classification scenarios. The first scenario present those results that classify our dataset into two classes: “diseased” and “healthy.” The second scenario present the results of the disease types: “olive peacock spot” and “Aculus olearius.” We note that the best results obtained by the DenseNet model were around 0.98 for binary classification.

Table 8 summarizes the results achieved using optimum deep learning models based on an adaptable evolutionary algorithm for multiclass categorization. We observe that the DenseNet model achieved the best results for multiclass classification with the highest accuracy about 0.96.

When we use our genetic algorithm-based technique, we observe that we can reach a high rate for the various assessment metrics with a little number of epochs. For AlexNet, we obtained a ROC-AUC value of roughly 0.97 using a 40-epoch number rather than a 60-epoch number. This decrease in the number of epochs was also observed in ResNet and DenseNet. For DenseNet, we were able to reduce the number of epochs to 20 while maintaining a higher level of accuracy.

The minimization of the epoch number gave us more time. Knowing that an epoch takes two minutes using the three models allowed us to save more than 30 minutes for each execution.
Through experimental results, we observe that our proposed model based on the genetic algorithm enhanced our accuracy and precision, but our greatest gain was in time.

In conclusion, branching out of machine learning methods into the depths of deep learning methods, the advancements of neural network makes trivial problems such as classifications so much easier and faster to process. In particular, it is very clear that our improved optimal deep learning model for diagnosing olive leaf diseases has given satisfactory results and attains a high classification accuracy rate. We can therefore note that our proposed model based on the genetic algorithm attempts to classify dataset by finding an optimal hyperplane and thus solves a quadratic optimization problem and provides rapid diagnosis of olive leaf images.

5. Conclusion

In this paper, we proposed an optimal deep learning model to classify olive leaf diseases. To this end, we used three deep learning models and an adapted version of the genetic algorithm.

The main goal of our proposed approach was to find an optimal batch size and epoch number to minimize response time and guarantee a higher accuracy rate. Before evaluating the proposed method, we compared the performances of different machine learning models.

The models were trained and tested using a database comprising 3,400 olive leaf images. The results showed that the random forest model achieved the best accuracy rate compared to other machine learning models. To evaluate our optimal deep learning models, we applied them to binary classification and multiclass classification.

Our results showed that optimal deep learning models improve the accuracy rate compared to all machine learning models, especially DenseNet-GA and ResNet-GA. On the other hand, our model could minimize response time as well as minimize the epoch number up to 15 epoch. The most successful model, a DenseNet model, achieved an accuracy rate of 98% in binary classification. As a future perspective, we plan to apply the optimal deep learning model to other plant collections and will try to collect other olive disease images.

Data Availability

The Olive dataset used to support the findings of this study has been deposited in the https://github.com/sinanuguz/CNN_olive_dataset. Comparing the outcomes of various machine learning methods is also part of our research; for that, we use sklearn (https://scikit-learn.org/stable/) as a python package and keras application (https://www.tensorflow.org/api_docs/python/tf/keras/applications) for deep learning models.

Disclosure

The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

Conflicts of Interest

We declare no conflict of interest.

Acknowledgments

The authors extend their appreciation to the Deanship for Research Innovation, Ministry of Education in Saudi Arabia for funding this research through the project number 400682337.

References

[1] P. Vossen, “Olive oil: history, production, and characteristics of the world’s classic oils,” HortScience, vol. 42, no. 5, pp. 1093–1100, 2007.

[2] M. E. Hernández, A. Ruiz Davila, A. Perez De Algaba, M. A. López, A. Trapero, and Casas, “Occurrence and etiology of death of young olive trees in Southern Spain,” European Journal of Plant Pathology, vol. 104, no. 4, pp. 347–357, 1998.

[3] E.-C. Oerke, “Crop losses to pests,” The Journal of Agricultural Science, vol. 144, no. 1, pp. 31–43, 2006.

[4] H. Cetin and O. Alaoğlu, “Eriophyid mites and their damage on olive trees in the Mut Mersin district of Turkey,” Journal of Entomology, vol. 30, no. 4, pp. 303–315, 2006.

[5] Q. Abu Al-Haija, M. Krichen, and W. Abu Elhaija, “Machine-learning-based darknet traffic detection system for IoT applications,” Electronics, vol. 11, no. 4, p. 556, 2022.

[6] A. Mihoub, H. Snoun, M. Krichen, R. B. H. Salah, and M. Kahia, “Predicting covid-19 spread level using socioeconomic indicators and machine learning techniques,” in In 2020 First International Conference of Smart Systems and Emerging Technologies, pp. 128–133, Riyadh, Saudi Arabia, Nov. 2020.

[7] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” Nature, vol. 521, no. 7553, pp. 436–444, 2015.

[8] O. B. Fredj, A. Mihoub, M. Krichen, O. Cheikhrouhou, and A. Derhab, “Cybersecurity attack prediction: a deep learning approach,” in In 13th International Conference on Security of Information and Networkspp. 1–6, Merkez Turkey, 2020.

[9] T. Wang, Y. Quan, X. S. Shen, T. R. Gadekallu, W. Wang, and K. Dev, “A privacy-enhanced retrieval technology for the cloud-assisted Internet of things,” IEEE transactions on industrial informatics, p. 1, 2021.

[10] H. Mukhtar, S. Rubaiee, M. Krichen, and R. Alroobaee, “An IoT framework for screening of covid-19 using real-time data from wearable sensors,” International Journal of Environmental Research and Public Health, vol. 18, no. 8, p. 4022, 2021.

[11] A. Sinha and R. S. Shekhawat, “Olive spot disease detection and classification using analysis of leaf image textures,” Procedia Computer Science, vol. 167, pp. 2328–2336, 2020.

[12] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” Advances in Neural Information Processing Systems, vol. 25, 2012.

[13] K. Gasmi, I. B. Tlaifa, G. Lejeune, H. Alshammar, L. B. Ammar, and M. A. Mahmood, “Optimal deep neural network-based model for answering visual medical question,” Cybernetics and Systems, pp. 1–22, 2021.

[14] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in In Proceedings of the IEEE conference on
[45] Y. Mingqiang, K. Kidiyo, and R. Joseph, “A survey of shape feature extraction techniques,” *Pattern Recognition*, vol. 15, no. 7, pp. 43–90, 2008.

[46] H. Li, Y. Wei, L. Li, C. L. Philip, and Chen, “Hierarchical feature extraction with local neural response for image recognition,” *IEEE Transactions on Cybernetics*, vol. 43, no. 2, pp. 412–424, 2013.

[47] L. Wang, P. Wang, S. Liang, Y. Zhu, J. Khan, and S. Fang, “Monitoring maize growth on the North China plain using a hybrid genetic algorithm-based back-propagation neural network model,” *Computers and Electronics in Agriculture*, vol. 170, article 105238, 2020.