2023

Optimization techniques in deep convolutional neuronal networks applied to olive diseases classification

BY ghadeer alharbi

1.Abstract

In this summary we will discuss the effect of two optimization techniques used CNN model for the performance of olive disease classification. Agricultural production is adversely affected by plant diseases, both in terms of quality and quantity. Nevertheless, the prediction of these diseases is proving to improve crop quality and reduce production losses. There is no doubt that detecting plant diseases, whether using traditional methods or naked eye, is a cumbersome process in terms of time, availability, and accuracy. An analysis of various CNN architectures and optimization algorithms, along with classification techniques used for olive disease detection, is presented in the present work. To detect olive diseases, this study presents a dataset containing 5571 olive leaf images collected manually on real conditions in various regions of Morocco. This research also aimed to examine the correlations between CNN architectures and optimization algorithms based on accuracy and other performance metrics. In experiments without data augmentation, the highest rate was 92,59%, while in trained models it was 100 %. Neuronal network performance is also studied in this study in relation to optimization algorithms.Based on the results of experiments, MobileNet architectures using Rmsprop algorithms were found to produce better performance and efficiency than other combinations.[1]

2.INTRODUCTION

Through the collection of a bunch of information collected under genuine conditions in various districts of the nation and for a variety of illnesses affecting plants and particularly olive trees, the current study of this paper was conducted in Morocco to add to the identification disease (ID) of those illnesses that affect the olive tree. Education regions, both inside and outside, introduce themselves as basic components. [1]. In summary, this work covers the following main topics. Firstly, the presentation of CNN architectures and diseases used in the literature for classification of olive plant diseases. Secondly, this work presents an exploratory analysis with a simulation of performance based on the data assembled previously for olive tree diseases. Further, the primary aim of this review is to distinguish the side effects of six olive plant leaves illnesses commonly seen[1].

As a benchmark for relative works, the dataset includes 5571 images assembled to demonstrate the performance of the system detector. Several studies have demonstrated that olive plant leaves diseases can be detected without the help of an expert. In order to improve classification performance. The purpose of this work is to answer the following questions[1]:

- How does CNN Model affect the classification of olive diseases?
- How does optimization affect the performance of olive disease classification?
- Does the CNN Model used for classification correlate with the algorithm used to optimize loss function?

Following is a structure for the remainder of this paper. First of all, Section 2 presents background and related works, followed by Methodology used. Second, Sections 3 and 4 describe the experimental setup based on simulations and discussions. Section 4 concludes with some conclusions and some perspectives for future research.

3. Background

A large number of researchers and practitioners have been attracted to machine learning in recent years due to its rapid growth .Several fields rely on machine learning, including machine translation, speech recognition, image recognition, recommendation systems, etc[3]. In machine learning, optimization is a key component. The essence of most machine learning algorithms is to build an optimization model and learn the parameters in the objective function from the given data. A machine learning model's popularity and application are heavily influenced by the effectiveness and efficiency of the numerical optimization algorithms in an era of immense data. Several effective optimization methods have been developed to promote machine learning, which have improved its performance and efficiency.

According to gradient information, popular optimization methods fall into three categories: stochastic gradient methods are a good example of a first-order optimization method; Newton's method is a typical example of a high-order optimization method; and coordinate descent is a representative of a heuristic derivative-free optimization method [3]. When optimizing a design, the objective may simply be to minimize production costs or to maximize production efficiency. Optimisation is a procedure used to compare various solutions iteratively until a satisfactory or optimum solution is found[2]. It has become an integral part of computer-aided design. The optimization algorithms of today can be divided into two types. A deterministic algorithm uses specific rules for moving from one solution to another, while a stochastic algorithm uses probabilistic rules for translating between solutions [2]. However, the focus in this summary is about the stochastic algorithms.

3.1 Stochastic Methods.

Probabilistic translation rules are inherent to stochastic methods .There are certain properties of these algorithms that make them more popular than deterministic methods[2]. An example of a stochastic optimization methods are:

- Iterated Local Search
- Stochastic Hill Climbing
- Stochastic Gradient Descent
- Tabu Search
- Greedy Randomized Adaptive Search Procedur[4].

In this summary, two common methods of stochastic methods will be addresses and discusses with respect to the agriculture domain. These methods are Stochastic Gradient Descent and Adaptive Learning Rate Method

a) Stochastic Gradient Descent:

For large-scale data and online updates, stochastic gradient descent (SGD) has been proposed [5]. As opposed to calculating the exact gradient value directly, stochastic gradient descent updates the gradient using one sample randomly per iteration. Stochastic gradients are unbiased estimates of real gradients [5]. Stochastic gradient descent has a nonlinear convergence speed and is independent of sample numbers [11]. SGD reduces the update time for dealing with large numbers of samples and removes a certain amount of computational redundancy, which significantly accelerates the calculation. In the strong

convex problem, SGD can achieve the optimal convergence speed [13], [15], [15], [10]. Meanwhile, it overcomes the disadvantage of batch gradient descent that cannot be used for online learning.

- The loss function can be written as the following equation:

$$L(\theta) = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{2}(y^i - f_\theta(x^i))^2 = \frac{1}{N}\sum_{i=1}^{N} cost(\theta, (x^i, y^i)).$$

If a random sample i is selected in SGD, the loss function will be $L^*(\theta)$:

$$L^*(\theta) = cost(\theta, (x^i, y^i)) = \frac{1}{2}(y^i - f_\theta(x^i))^2.$$

The gradient update in SGD uses the random sample i rather than all samples in each iteration,

$$\theta' = \theta + \eta(y^i - f_\theta(x^i))x^i.$$

Due to the fact that SGD uses only one sample per iteration, each iteration's computation complexity is O(D), where D is the feature count. With large numbers of samples N, SGD has a faster update rate than batch gradient descent. With SGD, optimization efficiency is improved at the expense of more iterations, but the increased number of iterations is insignificant compared with the high computation complexity caused by a large number of samples. Despite hundreds of thousands of samples, it is possible to get the optimal solution using only thousands of samples. Therefore, compared with batch methods, SGD can effectively reduce the computational complexity and accelerate convergence.

As a result of random selection, the gradient direction oscillates, and the search process in the solution space is blind. The variance of gradients in SGD is large, and the direction of movement is biased compared to batch gradient descent, whichalways moves toward the optimal value along the negative gradient direction. As a result, a compromise between these methods has been proposed [5], the mini-batch gradient descent method (MSGD). During each iteration, the MSGD updates the parameters using b independent identical samples (usually 50 to 256 [12]). As a result, the gradient variance is reduced and the convergence is more stable, which makes the optimization process faster. In the following sections, we will refer to MSGD as SGD. SGD has a better chance of finding a global optimal solution for complex problems than conventional optimization methods. When batch gradient descent is used for multimodal problems, deterministic gradients may cause the objective function to fall into a local minimum. However, the fluctuation in SGD always exists, so the objective function is able to jump to another possible minimum. However, the fluctuation in SGD may slow down the convergence process somewhat. It is still important to note that SGD can be used in the concrete optimization process [12], but there are many details that need to be considered, such as the choice of the right learning rate. The convergence rate will be slowed if the learning rate is too small, while the loss function will fluctuate too little if the learning rate is too large.

During the learning process, the learning rate can be adjusted based on a predefined list of learning rates or a certain threshold [16, 17]. It is important, however, to determine these lists or thresholds in advance based on the dataset's characteristics. A learning rate of the same for all parameters is also inappropriate. When data are sparse and features are occurring at different frequencies, it is not expected that variables will be updated at the same rate. Less frequently occurring features tend to have a higher learning rate [6], [9].

One of the challenges is to avoid the objective function becoming trapped in infinite numbers of local minimums, along with the learning rate. This difficulty is not caused by the local minimum values, but by the "saddle point" [18]. It is important for SGD to escape from saddle points since they have a positive slope in one direction and a negative slope in another direction, and gradients in all directions are zero. There have been some studies on escaping from saddle points [19], [20].

b) Adaptive Learning Rate:

The manually selected learning rate greatly influences the effect of the SGD method. Setting a learning rate that is appropriate is a tricky problem [6], [9], [21]. Adaptive methods have been proposed for automatically adjusting the learning rate. Deep neural networks use them extensively to solve optimization problems as these methods are free of parameter adjustments, fast to converge, and often deliver not badresults. AdaGrad is the most straightforward improvement to SGD. AdaGrad adjusts the learning rate dynamically based on the history of gradients. The update formulae are as follows:

where gt is the gradient of parameter θ at iteration t, Vt is the accumulate historical gradient of parameter θ

$$V_t = \sqrt{\beta V_{t-1} + (1-\beta)(g_t)^2},$$

at iteration t,andθtisthe value of parameter θ at iteration t.

In contrast to gradient descent, AdaGrad computes the learning rate using all of the historical gradients accumulated up to this iteration during parameter updating, rather than a fixed learning rate. AdaGrad has the advantage of eliminating the need to adjust the learning rate manually. Most implementations use 0.01 by default.

$$\begin{cases} g_t = \dfrac{\partial L(\theta_t)}{\partial \theta}, \\ V_t = \sqrt{\sum_{i=1}^{t} (g_i)^2 + \epsilon}, \\ \theta_{t+1} = \theta_t - \eta \dfrac{g_t}{V_t}, \end{cases}$$

AdaGrad adaptively adjusts the learning rate, but there are still two issues. [5] The algorithm still has to adjust the global learning rate manually. The accumulated gradient will increase as the training time increases, resulting in a zero learning rate, resulting in an ineffective parameter update as the learning rate tends to zero.

A further improvement to AdaGrad was AdaDelta [7] and RMSProp [8] to resolve the problem of eventually zero learning rates. By considering not accumulating all historical gradients, but only those in a

window over a period, the exponential moving average can be used to calculate second-order cumulative momentum.

$$\theta_{t+1} = m_t - \eta \frac{\sqrt{1-\beta_2}}{1-\beta_1} \frac{m_t}{V_t + \epsilon}.$$

where β is the exponential decay parameter. RMSProp and AdaDelta were both developed independently around the same time, to reduce the radically decreasing AdaGrad learning rates.

Another advanced SGD method is adaptive moment estimation (Adam) [9], which incorporates an adaptive learning rate for each parameter. This method combines the adaptive learning rate and momentum methods. As with AdaDelta and RMSProp, Adam also keeps an exponentially decaying average of past squared gradients Vt, just as momentum does:

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t,$$

$$V_t = \sqrt{\beta_2 V_{t-1} + (1-\beta_2)(g_t)^2},$$

where β1 and β2 are exponential decay rates. The final update formula for the parameter θ is

The default values of β_1, β_2, and ϙ are suggested to set to 0.9, 0.999, and 10^{-8}, respectively. Adam works well in practice and compares favorably to other adaptive learning rate algorithms.

4.Related work

Related work can be classified into two categories: Firstly, techniques for detecting plant diseases and secondly, techniques used in the literature particularly for olive plant diseases classification.[1]

4.1. Techniques for detecting disease in plants

For the identification of plant diseases, this section describes techniques used in the literature. The first issue is that some works (Gavhale and Gawande, 2019; Saleem et al., 2019; Esgario et al., 2019) used leaf image datasets without specifying whether they were balanced or unbalanced. The authors in Moorthy et al. Using spectroscopy, Moorthy et al. (2020) studied the interaction between materials and light in terms of frequency joining or mirroring. As stated in the review, pictures were used as key information in most early research on leaves. The fact that the side effects are shown indicates that the sickness is now at a high level, and practically speaking, there is no hope of saving the tainted plants at this point.

In addition, Foysal et al. (2019) proposed a useful solution for identifying the classification areas of tomato plant illnesses and then obstructing them at an early stage. This study is aimed at finding a profound learning engineering that will handle units in a better way than acquiring actual examples (leaves, plants) and dissecting them in a lab, as has been done before. A convolutional neural network organization was prepared using the Inception v3 engineering in the exploration proposed by Saleem et al. (2019) using the cassava infection picture dataset collected from Tanzania using the Inceptionv3 technology. In order to distinguish three illnesses and two bugs, they demonstrated that determining how to move is an effective instrument for robotized disease discovery. Through a Tensor Flow application, the model is applied to cell phones to detect cassava plant diseases incrementally. In the exploration work (Abade et al., 2021) framed on 87,848 images, including 25 unique plants, some compositional models like AlexNet, GoogleNet, and VGG were examined. There were 58 specific types of plant sickness, including sound plants with the best results coming in at a triumph rate of 99.53 percent. Other works address a very important area regarding the limitations of data augmentation techniques, In fact, the authors in Tassis et al. (2021) propose an integrated framework using different convolutional neural networks (CNN) to automate the detection/recognition of lesions from field images collected via a smartphone containing part of the cof- fee tree. Additionally, Sharma et al. (2020) examine

Convolutional neural network (CNN) models can be trained using segmented image data to solve the model generalization problem on independent data. In this experiment, a proper dataset with an improved classification model is used instead of the previous one.

2.3.2. Techniques for detecting olive leaf diseases

According to Waleed et al. (2020), using segmentation of the k-mean algorithm that provides greater precision, texture analysis was applied using first to fourth order moments. This enabled the identification of the relationship between infection and one or more textures, in this case this allowed us to identify the infection relationship with a high correlation between the area of infection and texture features acting as homogeneity, entropy, which also helped to classify the two homogeneous diseases of neofabrie and leaf spot of peacock. Subsequently in Chandra and Matthias (2017), the authors present a proficient model utilizing the idea of move learning applied to distinguish olive tree diseases, a savvy increment of information with a weighted number of pictures in each class, and it works in more composite conditions with a broadened and further developed informational index. The simulation results show that the optimized model achieves higher estimates in terms of exactness, accuracy, recall, and F1 estimation with a general precision of 99.11%.

In Waleed et al. (2020), the authors describe an automated olive tree disease detection system based on an improved K-Means algorithm, including a model for implementing it based on classification steps. Therefore, Random Forest outperformed the others with an overall accuracy of 97.5% with a 70–30 ratio between training and testing. Additionally to the previous research presented, the experimental studies conducted by Uguz and Uysal (2020) using data augmentation techniques in trained models achieved a 95% performance rate, while the results of experiments without these techniques showed a higher value of 88%. In addition, this work examines how Adam, AdaGrad, SGD, and RMSProp streamlining calculations

impact network performance. As a result of the analyses performed, the authors assumed that Adam and SGD have relatively unrivaled outcomes as a whole. Nevertheless, when the Random Forest algorithm was

Table 1 Related works carried out of olive disease classification.

Paper	Validation accuracy	Augmentation	Nb. of classes	Transfer learning	Architecture	Dataset	Country source of images
Sinan Uguz, 2020	95%	Yes	(2 + 1 healthy)	Yes	VGG16 and VGG19 architectures	3400 olive leaves images	Turkey
Sinan Uguz, 2020	96%	Yes	(1 + 1 healthy)	No	SSD architecture	1460 olive leaves images	Turkey
Mario Milicevic, 2020	97.20% ± 0.57%	Yes	(1 + 1 healthy)	No	VGG-inspired network	1000 images	Croatia
Madallah Alruwaili et al, 2019	99.11% ± 0.75%	Yes	(6 + 1 healthy)	No	Alexnet architecture	2287 olive leaves images	-

applied to Olive Anthracnose, the results were satisfactory, but real data could have been collected to improve the results. Table 1 summarizes the results obtained from the literature on olive plant diseases classification and parameters used, including augmentation techniques, number of classes, transfer learning, CNN architecture, and dataset size.

5. Methodology used

In this section, different deep CNN architectures used to compare the performance of the two optimization techniques (i.e, Adagrad and SGD) in the agriculture domain.

5.1. Convolution Neuronal Network

Since 1989, various technological feats have been demonstrated in neural network engineering (Pedrycz, 2020). In a sense of continuous performance enhancement of its intelligent systems and based on the elaboration of the use cases, these innovations can be ordered by improving classification and regularization constraints.

5.1.1. AlexNet

In comparison to conventional techniques, this architecture was one of the main deep structures to evolve the performance of ImageNet classification by a crucial step, and in the works of the works, AlexNet introduced a new engineering (Alruwaili et al., 2019), using a rectified linear unit ReLu for the nonlinear fraction, rather than a tangent hyperbolic Tanh or Sigmoid function which had previously been integrated into classical neural networks. ReLu is calculated as follows:

$$f(x) = \max(0, x)$$

The strong point of the function of ReLu compared to the sigmoid func- tion is the computation time

This is justified by the fact that the function chooses max (0, x) rather than executing exponential operations during execution time, which is more expensive. The sigmoid turns out to be tiny in the immersion area, so weight updates almost disappear. It is called the vanishing gradient problem.

5.1.2. VGG

As CNN networks have gained popularity in image recognition tasks, architectural design research has increased. Their layered presentation, called VGG, was separated from the main presentation presented in Pedrycz (2020), which recommends a simple and powerful plan directive for CNN structures.Unlike AlexNet, VGG draws inspiration from the relationship between depth and the capacity to represent the composition of the layers (Uguz and Uysal, 2020). Fig. 2 depicts a basic layout of VGG.

5.1.3. GoogleNet

A GoogleNet is introduced to achieve the best performance at the lowest cost (Pedrycz, 2020) by challenging a simple objective. Through the revolutionary concept of the beginning block in CNN, convolutional modifications at different scales are joined using the concepts of division, change, and union to achieve this. With GoogleNet, convolutional layers are exchanged with miniature neural networks as underlying each layer.

5.1.4. Residual network

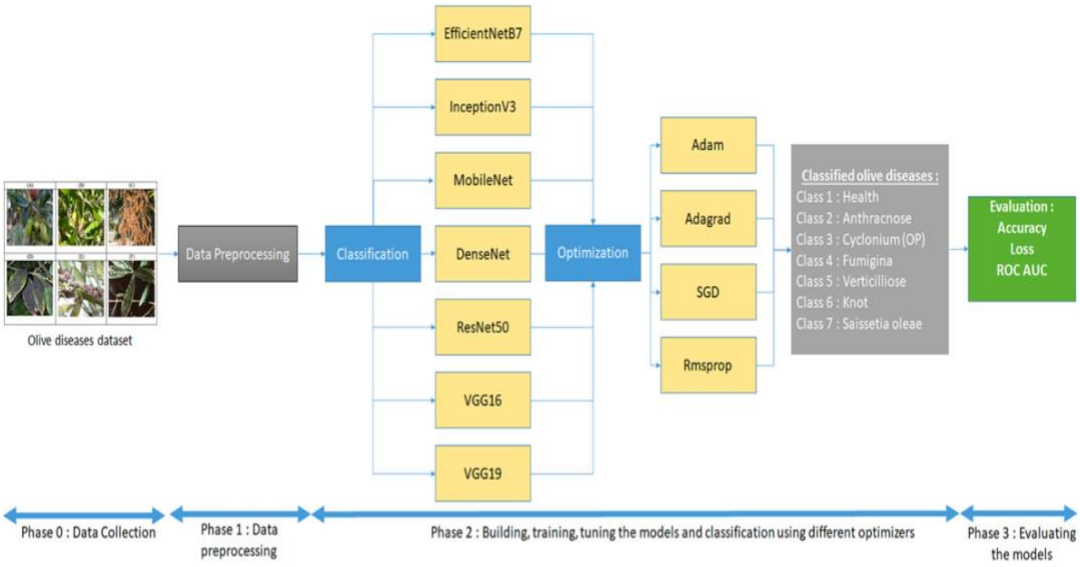
ResNet redesigned the engineering of CNN by introducing the idea of continuing to develop in CNNs for the preparation of deep organizations. For example, each of the past models of road networks used deep neural configurations that coherently superimposed several layers of convolution. Pedrycz (2020) found that deeper configurations are more efficient. In order to overcome these constraints, the authors of the ResNet architecture presented in Figure. Jump associations were introduced in layer 4 with the assumption that deeper layers will allow learning similar to shallower layers.

5.1.5. CNN characteristics

However, there are other challenges that a deep CNN architecture has been applied to, including time series data and grid topology. Deep CNN performs well in time series data or grid topology. With a great deal of strength and prattle, Table 1 describes the fundamental difficulties related to distinctive CNN models.

5.2. Simulations

Fig. 1 below shows the simulation process adopted for implementing the optimization techniques studied and the approach taken to obtain the best results.



5.2.1. Simulation workflow

Figure 1 General schema for experimental study

For the purpose of determining the CNN models most appropriate for detecting specific diseases, a training dataset consists of unknown samples used to train the model classification algorithm. As a result, a suitable model correlated with the best optimizer was obtained, and different performance indicators were used to assess the results. In this analysis, suitable classification methods are used to select features in an optimal manner. The most important tasks associated with the image classification techniques detailed in the schema above are the designation of a suitable classifier, the extraction of features, the choice of a training sample, the preprocessing of images and determining the optimal classification model, the optimization algorithm, processing after classification, and analyzing the accuracy of the evaluation (Dhingra et al., 2018).

5.2.2. Context of simulations

For the simulation, the following is the configuration of the computer hardware.

• CPU: Processeur i5-8250U @ 1.6 GHz (8 cpu)
• RAM: 16384MB
• Langage de programmation: Python Version 3.7 • Software: Anaconda 3 / Spyder Version 3.3.6

5.2.3. Data collection

In the dataset, there are 5571 images of different plant leaf classes (health and diseases) taken in the field from the crop olives. It contains seven classes of diseases, which are distributed unbalancedly. There are a variety of images taken at different stages of the evolution of the olive diseases presented in the dataset, which allows the classifier to obtain the best result on the validation set by enriching the learning set (Yousuf and Khan, 2021). Because few suitable datasets were available for real-time detection of olive diseases at the beginning of this work, a lot of human and material resources were used to collect diseased

olives. Seasonal changes as well as temperature, humidity, and radiance affect olive disease patterns. The expansion and diffusion of disease spots on affected leaves are facilitated by rainy weather, which can lead to germ generation and spread. For this reason,5571 images were collected by the author under a variety of weather conditions for more comprehensive applications. In addition, all diseased images in the dataset are manually annotated by experts.

2.2.4. Data preprocessing

As part of the preprocessing procedure, the study confirmed that the data points are spanned. Next, the data is split into 80 training images and 20 testing images. Furthermore, improvement approaches were applied to enhance the distribution of pixels across a wide range of intensities, and direct discrepancy stretching was applied to the images.

2.2.5. Data annotation

It is crucial to label the positions and classes of object spots in the diseased images during image annotation. The authors used Python is used to develop an algorithm that provides a frame selection function at this stage. Using this algorithm and the knowledge provided by agricultural experts, it is possible to select and assign diseased areas to the appropriate classes (Pantazi et al., 2020). A black pixel indicates the background, while all other colors uniquely identify the leaves of the plants in the scene. Annotations are provided as images of the same size as the originals, stored in JPG or JIFF formats. To label occurrences on the same sheet across the time-lapse footage, the approach consistently used the same color code. In the binary mask of each plant, delimit each leaf, branch, and fruit individually, following an approach based solely on manual labeling (Garcia and Barbedo, 2018).

2.2.6. Data augmentation

In the training stage of CNNs, the overfitting were avoided by using data augmentation. Overfitting occurs when noise or errors are described instead of the underlying relationship, as shown by Cap et al. (2020). With more images after expansion via data augmentation techniques, the model will be able to learn as many irrelevant patterns as possible during the training process, which will reduce overfitting and result in higher performance.

Rotation transformations, horizontal and vertical flips, and intensity disturbances, including brightness, sharpness, and contrast distortions, are used to augment data. In addition, Gaussian noise processing is applied. Using the above operations, diseased images are generated for each image (Cap et al., 2020). According to Vega-Márquez et al. (Vega-Márquez et al., 2020), cross-validation is used before oversampling olive dataset classes, as is the case with feature selection. Resampling the data repeatedly allows randomness to be introduced into the olive disease dataset to prevent overfittingm.

After applying the techniques outlined in the above paragraphs, the distribution of olive disease datasets. As part of the data augmentation process, rotation_range = 15, width_shift_range = 0.2, height_shift_range = 0.2, shear_range = 0.2, zoom_range = 0.2, horizontal_flip = True, fill_mode = 'nearest' are fine-tuned.

2.2.7. Feature extraction

Some variables or characteristics are very important in forming the different models. In this case, and in order to keep only the most relevant variables and eliminate the harmful characteristics that can disturb the learning of the proposed system, a selection process of variables is then applied. To this extent, the methods of variable selection used in this study are the sequential backward selection of variable selection.

2.2.8. Classification

A classification task requires the use of machine learning algorithms algorithms which learn how to assign a marker to different classes from the problem sphere. Using spam emails as an example, you can quickly understand multi-class classification. This refers to classification functions and characteristics that have more than two class markers.

2.2.9. Evaluation

A performance metric provides decision makers with intelligence to support their evaluation. To determine the appropriate feature selection method for prediction [5], distinctive performance metrics such as Accuracy, Error Rate, Kappa, Precision, Recall, F1 Score, Mean Absolute Error, and Log Loss are used to evaluate the results.

2.2.9.1. Accuracy:

Accuracy is calculated according to the formula below.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

Where, TP- True Positive; TN- True Negative; FP- False Positive; FN- False Negative.

2.2.9.2. Precision.

The present of predicted positive that is in fact positive. It also called as Positive Predicted Value (PPV).

$$\text{Precision} = \frac{TP}{TP + FD}$$

2.2.9.3. Recall.

The proportion of positive results out of the number of samples which were in fact positive. It is recognized as Sensitivity.

$$\text{Recall} = \frac{\text{Tp}}{\text{TP} + \text{FN}}$$

2.2.9.4. F1 score.

The F1 Score is derived from the geometric mean of the Precision or PPV and

$$\text{F1Score} = 2 * \frac{\text{PPV.TPR}}{\text{PPV} + \text{TPR}}$$

2.2.9.5. F1 score.

$$\text{Loss} = -\frac{1}{\text{output}} \sum_{i=1}^{\text{outpet}} y_i \cdot \log \hat{y}_i + (1 - y_i) \cdot \log (1 - \hat{y}_i)$$

F1 Score is designed as the geometric mean of Precision or PPV and Recall or True Positive Rate (TPR).

$$\text{F1Score} = 2 * \frac{\text{PPV.TPR}}{\text{PPV} + \text{TPR}}$$

2.2.9.6. Loss.

The paired cross entropy misfortune work determines the scale 90% efficiency of a model by figuring the accompanying normal:

Model yield refers to scalar qualities, yi refers to the objective value that relates the model output, and yield size refers to the amount of scalar qualities that constitute the output.

There are several challenges associated with deep CNN, among which is a deep CNN architecture. Deep CNN works well in time series data or in grid topology. The main challenges associated with the different CNN architectures are listed in Table 2 along with major strengths and weaknesses.

Table 2 Major challenges associated with implantation of depth based CNN architectures[1]

Architecture	Strength	Shortcomings
AlexNet	- No more convultive layers and parameters used to adapt to the high volume data set. - The introduction of the rectified unit and the preprocessing represent an important advance in computer vision tasks.	- Limited efficiency compared to new CNN architectures.
ResNet	- Simplification of the process of knowledge as an extential layer in neuronal networks as an identity function. - Inputs can propagate faster via residual connections between layers. - The convolutional and sequential characteristic of the network.	- Overfitting due to stacking the same modules.
VGG	- Based on a compact and efficient design of complex networks. - Multitudes of layers of deep and reduced convolutions performed better than fewer layers of wider convolutions.	- The cost of calculation using fully associated layers.
GoogLeNet	- Convolutions 1×1 minimize the dimensionality of the channel on the pixel pane. Thus the maximum pooling reduces the resolution. - Offers a validation performance close to similar architectures with a complexity (it can't be considered as an advantage I guess) and a more optimal calculation time.	- Risk of data loss due to a bottleneck

5.3. Olive diseases

There are two types of olive plant disease research: first, techniques used in the literature to detect plant diseases, and second, methods used to classify olive plant diseases. In the following, a brief description of the olive disease that have been addresses by [1] is provided.

The Mediterranean region accounts for 95% of the 750 million hectares cultivated. A variety of factors can affect the olive crop, including insects, nematodes, and pathogens, the latter of which can severely damage the production of olives throughout the European Union (Pedrycz, 2020). There has been an increase in the introduction, spread, and establishment of certain diseases in olive production due to commercial operations of goods and people, climate change and changes in agricultural practices (Sinha and Shekhawat, 2020). F. Diseases such as: (Moorthy et al., 2020; Chliyeh et al., 2014) Negative influences on olive tree yields (Moorthy et al., 2020):

- Black olive disease
- Peacock eye disease
- Verticulosis disease
- Anthracnose disease
- Tuberculosis disease
- Saissetia oleae diseas

6. Results obtained

For the purpose of predicting specific diseases, models are implemented using the appropriate optimizer. In order to find the most suitable classification method for a particular disease, augmentation techniques are used. The techniques are then evaluated using metrics such as Accuracy.

With regard to Table 3, CNN MobileNet offers superior results. The optimal number of epochs using Early Stopping call back function is 100 iterations. This will further minimize the oscillations in the loss graph. Additionally, the augmentation would significantly extend the experimental duration. To avoid overfitting and increase the generalization capacity of the neural network, the neural network should be trained for the optimal number of epochs. The training dataset is used for model validation, which involves evaluating the model's performance after each training epoch. During training and validation, loss and accuracy are monitored to determine when overfitting begins. A MobileNet model with a precision value of 0.98 on a validation dataset was found to have the best precision values for six disease classes. Only models developed with Adagrad and SGD algorithms are shown with precision and loss value graphs. Table 3 shows the precision and loss values of the training and validation datasets after the application of data augmentation techniques.

According to the classification report, classes 2 and 4 outperform the other classes in terms of precision. As a result, we can conclude that in addition to the quality and the quantity of images, the types of classes used also play a role in improving performance. AUC is also clearly indicated in the Table 3, which clearly demonstrates that the MobileNet model combined with SGD optimization offers the best results.

Table 3: Performances of both Adagrad and SGD across different CNN architectures (10 epochs)[1].

Optimization	CNN model	Training loss	Training accuracy	Validation loss	Validation accuracy
Adagrad	*ConvNet*	0.1809	0,9242	0,305	0,8749
	EfficientNetB7	0.9578	1.000	0,1899	0,1968
	InceptionV3	0.4050	1.000	0,1885	0,9625
	MobileNet	0.0664	1.000	0,0159	0,9792
	DenseNet	0.2370	0,9843	0,0946	0,9583
	VGG19	0.4084	0,9686	0124	0,8333
	ResNet50	0.5161	0,8340	0,5074	0,7875
	VGG16	0.4037	1.000	0,1483	0875
SGD	*ConvNet*	0.1479	0,9982	0,2661	0,8969
	EfficientNetB7	0,1601	0,9351	0,2714	0,8749
	InceptionV3	0.1534	0,9952	0,1244	0,9792
	MobileNet	0.0241	1.000	0,0435	0,9801
	DenseNet	0.1228	1.000	0,1937	0,9625
	VGG19	0.3500	0,8831	0,2936	0,8969
	ResNet50	0,1855	0,9707	0,2777	0,8312
	VGG16	0.2581	0,9460	0,2488	0,9405

7. Discussions

Table 4 details the results of the Adagrad algorithms, which present the weakest performances with the largest absolute errors of 0.2682, and the largest number of iterations 143 (for the stochastic algorithm, the number of iterations is fixed at 75). According to CNN MobileNet, both mean absolute error and accuracy are superior. In addition, the results indicated a significant difference between the solvers - Adagrad's algorithm converged after 143 iterations. Furthermore, the Adagrad algorithms represents the best MAEs which are of the order of 0.2682, with an increase in the number of iterations. Due to its high convergence speed and MAE, the Stochastic algorithm is of particular interest for mobile phones. Secondly, to the best of our knowledge, this research paper has addressed the issue of the choice of optimization techniques at the level of convolutional neural networks and its impact on their performance in a way that is unprecedented. The batch norm and ReLU are the main characteristics that distinguish MobileNet architecture from other CNNs. In MobileNet, classification, identification, and segmentation are supported. As a result of the number of parameters and layers, the ability to run deep networks on terminal mobile devices enhances the user experience, providing the security and energy efficiency advantages associated with mobile applications. In addition, more simulations with parameter values are necessary to enhance a model that performs better than the pretrained model. In addition, increasing the number of simulations and periods will also result in a longer model compilation time, even if it improves performance. Furthermore, the performance of the same solver will change if a different set of hyperparameters and starting conditions are used. As a result, larger learning rate values tend to exceed gradient values, which makes it difficult for weights to converge to global minimums. Additionally, when the learning rate is too low, the progress of the global minimum is very slow, as seen in validation and training losses.

Table 4 Comparison of the performances of studied algorithms.

Algorithm	**Number of iterations**	**MAE**	**Tensorflow**
Adagrad	143	0,2682	AdagradOptimizer
Stochastic	75(fixed)	0,2263	SGDRegressoe

8. Conclusions

In this summary a review of two optimization methods, namely Adagrad and SGD have been provided for the task of olive disease classification in the agriculture domain. These two techniques have studied with the respect of the CNN performance with different architectures of deep neural networks. Also, the study has developed a customized dataset for the identification task with 5571 images. A proper annotation process has been applied to obtain a reliable data. Results show that SGD outperformed the Adagrad. According to the study, the proposed method of automatic identification of olive diseases has obtained satisfactory results, however, further research is needed to improve its accuracy and reliability, as follows:

- Develop the olive tree disease dataset and set up a drone-based intelligent olive tree disease diagnosis system.
- Increase the accuracy of the system by integrating an image segmentation layer.
- Enhance the accuracy, reliability, and robustness of olive disease diagnostic systems by investigating different architectures of deep neural networks.

Reference:

[1] https://www.sciencedirect.com/science/article/pii/S258972172200006X

[2] https://mech.iitm.ac.in/nspch52.pdf

[3] https://arxiv.org/pdf/1906.06821.pdf

[4] Page 7, Essentials of Metaheuristics, 2011

[5] H. Robbins and S. Monro, "A stochastic approximation method," The Annals of Mathematical Statistics, pp. 400–407, 1951.

[6] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," Journal of Ma chine Learning Research, vol. 12, pp. 2121–2159, 2011.

[7] M. D. Zeiler, "AdaDelta: An adaptive learning rate method," arXiv preprint arXiv:1212.5701, 2012.

[8] T. Tieleman and G. Hinton, "Divide the gradient by a running average of its recent magnitude," COURSERA: Neural Networks for Machine Learning, pp. 26–31, 2012.

[9] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in International Conference on Learning Representations, 2014, pp. 1– 15.

[10] N. L. Roux, M. Schmidt, and F. R. Bach, "A stochastic gradient method with an exponential convergence rate for finite training sets," in Advances in Neural Information Processing Systems, 2012, pp. 2663– 2671.

[11] R. Johnson and T. Zhang, "Accelerating stochastic gradient descent using predictive variance reduction," in Advances in Neural Information Processing Systems, 2013, pp. 315–323.

[12] S. Ruder, "An overview of gradient descent optimization algorithms," arXiv preprint arXiv:1609.04747, 2016.

[13] A. S. Nemirovsky and D. B. Yudin, Problem Complexity and Method Efficiency in Optimization. John Wiley & Sons, 1983.

[14] A. Nemirovski, A. Juditsky, G. Lan, and A. Shapiro, "Robust stochastic approximation approach to stochastic programming," SIAM Journal on Optimization, vol. 19, pp. 1574–1609, 2009.

[15] A. Agarwal, M. J. Wainwright, P. L. Bartlett, and P. K. Ravikumar, "Information-theoretic lower bounds on the oracle complexity of convex optimization," in Advances in Neural Information Processing Systems, 2009, pp. 1–9.

[16] H. Robbins and S. Monro, "A stochastic approximation method," The Annals of Mathematical Statistics, pp. 400–407, 1951.
[17] C. Darken, J. Chang, and J. Moody, "Learning rate schedules for faster stochastic gradient search," in Neural Networks for Signal Processing, 1992, pp. 3–12.

[18] I. Sutskever, "Training recurrent neural networks," Ph.D. dissertation, University of Toronto, Ontario, Canada, 2013.

[19] Z. Allen-Zhu, "Natasha 2: Faster non-convex optimization than SGD," in Advances in Neural Information Processing Systems, 2018, pp. 2675–2686.

[20] R. Ge, F. Huang, C. Jin, and Y. Yuan, "Escaping from saddle pointson- line stochastic gradient for tensor decomposition," in Conference on Learning Theory, 2015, pp. 797–842.

[21] C. Darken and J. E. Moody, "Note on learning rate schedules for stochastic optimization," in Advances in Neural Information Processing Systems, 1991, pp. 832–838.

[22] Abade, A., Ferreira, P.A., Vidal, F.B., 2021. Plant diseases recognition on images using convolutional neural networks: A systematic review. Computers and Electronics in Agriculture. Elsevier, https://doi.org/10.1016/j.compag.2021.106125.

[23] Alruwaili, M., Alanazi, S., El-Ghany, S.A., Shehab, A., 2019. An efficient deep learning model for olive diseases detection. Int. J. Adv. Comput. Sci. Appl. 10, 486–492. https://doi.org/10.14569/ijacsa.2019.0100863.

[24] Altarawneh, M., 2015. An empirical investigation of olive leave spot disease using auto- cropping segmentation and fuzzy c-means classification. World Appl. Sci. Emp. In- vest. Olive Leav. Spot Diseas. https://doi.org/10.5829/idosi.wasj.2013.23.09.1000.

[25] Alves, L., Silva, R.R., Bernardino, J., 2019. System to predict diseases in vineyards and olive groves using data mining and geolocation ICSOFT 2018. Proceedings of the 13th International Conference on Software Technologies, pp. 679–687 https://doi.org/10.5220/0006914306790687.

[26] Cap, Q.H., Uga, H., Kagiwada, S., Iyatomi, H., 2020. LeafGAN: an effective data augmenta- tion method for practical plant disease diagnosis. IEEE Trans. Autom. Sci. Eng. https://doi.org/10.1109/TASE. 2020.3041499.

[27] Chandra, M., Matthias, M., 2017. SC-Adagrad and SC-RMSProp. arXiv:1706.05507v2. Chliyeh, M., Selmaoui, K., Touhami, A.O., Abdelkarim, F., 2014. Survey of the fungal species associated to olive-tree (Olea europaea L.). IJRB Survey of the Fungal Species Associ- ated to Olive-tree (Olea europaea L.) in Morocco.

[28] Dhingra, G., Kumar, V., Joshi, H.D., 2018. Study of digital image processing techniques for leaf disease detection and classification. Multimed. Tools Appl. 77, 19951–20000. https://doi.org/10.1007/s11042-017-5445-8.

[29] Esgario, J.G., Krohling, R.A., Ventura, J.A., 2019. Deep Learning for Classification and Sever- ity Estimation of Coffee Leaf Biotic Stress. http://arxiv.org/abs/1907.11561.

[30] Foysal, F.A., Islam, M.S., Abujar, S., 2019. A Novel Approach for Tomato Diseases Classifica- tion Based on Deep Convolutional Neural Networks A Novel Approach for Tomato Diseases Classification Based on Deep Convolutional Neural Networks. July. Springer, Singapore https://doi.org/10.1007/978-981-13-7564-4.

[31] Garcia, J., Barbedo, A., 2018. Impact of dataset size and variety on the e ff ectiveness of deep learning and transfer learning for plant disease classi fi cation. Comput. Electron. Agricult. 153, 46–53. https://doi.org/10.1016/j.compag.2018.08.013.

[32] Gavhale, M.K.R., Gawande, P.U., 2019. An Overview of the research on plant leaves disease detection using image an overview of the research on plant leaves. Diseas. Detect. Image Proess. Techn. https://doi.org/10.9790/0661-16151016.

[33] Hussain, S.A., Hasan, R., Hussain, S.J., 2018. Classification and Detection of Plant Disease using Feature Extraction Methods13 pp. 4219–4226.

[34] Kurmi, Y., Gangwar, S., Agrawal, D., Kumar, S., Srivastava, H.S., 2020. Leaf Image Analysis- Based Crop Diseases Classification Orre Cted Unc Pro Of. Signal, Image and Video Processing. Springer-Verlag London Ltd., part of Springer Nature 2020 https://doi.org/10.1007/s11760-020-01780-7.

[35] Liu, J., Wang, X., 2021. Plant diseases and pests detection based on deep learning: a re- view. Plant Meth., 1–18 https://doi.org/10.1186/s13007-021-00722-9.

[36] Moorthy, S.G., Meenakshi, K., Nithya, M., 2020. Plant leaf disease classification and detec- tion system using machine learning plant leaf. Diseas. Classific. Detect. Sys. Mach. Learn. https://doi.org/10.1088/1742-6596/1712/1/012012.

[37] Pantazi, X.E., Moshou, D., Bochtis, D., 2020. Artificial Intelligence in Agriculture. Intelligent Data Mining and Fusion Systems in Agriculture. Springer, pp. 17–101 https://doi.org/ 10.1016/b978-0-12-814391-9.00002-9.

[38] Pedrycz, W., Chen, S.m., 2020. Deep Learning: Algorithms and Applications. Studies in Computational Intelligence865. Springer. https://doi.org/10.1007/978-3-030-31760-7.

[39] Saleem, M.H., Potgieter, J., Arif, K.M., 2019. Plant disease detection and classification by deep learning. Plants 8, 32–34. https://doi.org/10.3390/plants8110468.

[40] Sharma, P., Berwal, Y.P.S., Ghai, W., 2020. Performance analysis of deep learning CNN models for disease detection in plants using image segmentation. Inf. Process. Agricult. 7, 566–574. https://doi.org/10.1016/j.inpa.2019.11.001.

[41] Singh, D., Jain, N., Jain, P., Kayal, P., Kumawat, S., Batra, N., 2020. PlantDoc: a dataset for visual plant disease detection. ACM Int. Conf. Proceed. Ser., 249–253 https://doi.org/ 10.1145/3371158.3371196.

[42] Sinha, A., Shekhawat, R.S., 2020. Olive spot disease detection and classification using anal- ysis of leaf image textures. Proc. Comput. Sci. 167, 2328–2336. https://doi.org/10. 1016/j.procs.2020.03.285.

[43] Tassis, L.M., Tozzi de Souza, J.E., Krohling, R.A., 2021. A deep learning approach combining instance and semantic segmentation to identify diseases and pests of coffee leaves from in-field images. Comput. Electron. Agricult. 186. https://doi.org/10.1016/j.compag.2021.106191.

[44] Uguz, S., Uysal, N., 2020. Classification of olive leaf diseases using deep convolutional neu- ral networks. Neur. Comput. Appl. 5. https://doi.org/10.1007/s00521-020-05235-5.

[45] Vega-Márquez, B., Nepomuceno-Chamorro, I., Jurado-Campos, N., Rubio-Escudero, C., 2020. Deep learning techniques to improve the performance of olive oil classification. Front. Chem. 7, 1–10. https://doi.org/10.3389/fchem.2019.00929.

[46] Waleed, M., Um, T.W., Khan, A., Khan, U., 2020. Automatic detection system of olive trees using improved K-means algorithm. Rem. Sens. 12, 1–16. https://doi.org/10.3390/ rs12050760.

[47] Yousuf, A., Khan, U., 2021. Ensemble Classifier for Plant Disease Detection. 1st. 10. IJCSMC, pp. 14–22. https://doi.org/10.47760/ijcsmc.2021.v10i01.003.