Highly Accurate Deep Learning Model for Olive Leaf Disease Classification: A Study in Tacna-Perú

Erbert F. Osco-Mamani[1], Israel N. Chaparro-Cruz[2]
Department of Computer Science and Systems Engineering[1,2]
Universidad Nacional Jorge Basadre Grohmann, Tacna, Perú[1,2]

***Abstract*—Deep learning applied to computer vision has different applications in agriculture, medicine, marketing, meteorology, etc. In agriculture, plant diseases can cause significant yield and quality losses. The treatment of these diseases depends on accurate and rapid classification. Olive leaf diseases are a problem that threatens the crop quality of olive growers. The objective of this work was to classify olive leaf diseases with Deep Learning in olive crops of the La Yarada-Los Palos area in the Tacna region, Peru. Disease classification is a critical task, nevertheless, for the most common diseases in the region: virosis, fumagina, and nutritional deficiencies, there is no dataset to train deep learning models. Due to the latter, a novel dataset of RGB olive leaf images is elaborated and published. Then, an extensive comparative experimental study was conducted using all possible configurations of Learning from Scratch, Transfer Learning, Fine-Tuning, and Data Augmentation state-of-the-art methods to train a modified VGG16 architecture for the classification of Olive Leaf Diseases. It was demonstrated experimentally: (*i*) The ineffectiveness of Data Augmentation when the model Learning from Scratch, (*ii*) A high improvement by using Transfer Learning vs Learning from Scratch, (*iii*) Similar performance using Transfer Learning vs Transfer Learning + Fine-Tuning vs Transfer Learning + Data Augmentation, and (*iv*) Very high improvement using Transfer Learning + Fine-Tuning + Data Augmentation. This led us to a Deep Learning Model with an accuracy of 100%, 99.93%, and 100% in the training, validation, and test sets and F1-Score on the validation set of 1, 0.9901, and 0.9899 in the Nutritional Deficiences, Fumagina, and Virosis olive leaf diseases respectively. Replication of the results is ensured by publishing the novel dataset and the final model on GitHub.**

Keywords*—Olive; leaf diseases; disease classification; deep learning; data augmentation; transfer learning; fine-tuning; VGG16

I. Introduction

Tacna is the leading nationally producer of olives [1]. Peru is the second-largest exporter of olives and third-largest exporter of olive oil in South America [1]. With over 22,897 hectares under olive cultivation and an average yield of 7,995 kg/ha, olive production and processing is critical to the local and regional economy [2]. Tacna accounts for 81.4% of the olive area that exists in Peru [3]. Despite its importance, olive cultivation is still managed traditionally and the use of data is incipient. Pests and diseases such as Orthezia Olivicola and fruit borer can significantly reduce the number of fruits per harvest [4]. This and other problems will be exacerbated by impending climate change [5].

The problems derived from pests and diseases affect the production and the production cycle as they affect leaves, flowers, and fruits and can initiate neighboring cycles [6]. In addition, the scarcity of diagnostic tools in underdeveloped countries has a devastating impact on their development and quality of life. Therefore, there is an urgent need to automatically classify plant diseases with affordable and easy-to-use solutions.

To reduce olive harvesting diseases, it is necessary to create and modernize technologies for efficient productivity. Adequate and fast classification of olive leaf diseases could prevent quality and crop losses. There is research related to the classification of olive leaf diseases [7] with datasets collected in countries such as Saudi Arabia [8], [9] or Turkey [10], [11], [12]. However, for the most common diseases in the region of Tacna-Perú: virosis, fumagina, and nutritional deficiencies, there is no dataset to train computer vision models for classification.

Deep Learning has become the artificial intelligence method by excellence for solving a variety of problems, including those related to computer vision [13]. The area of computer vision encompasses a large number of tasks such as segmentation, detection, and classification, among which are those related to diseases. In disease problems, Deep Learning has been shown to be state-of-the-art in both agricultural and medical applications using CNN architectures [14], [15], [16]. Deep Learning models learn directly from data and require large datasets to obtain good accuracies. To avoid the latter, some techniques have been well proven to obtain models with better results, such as Data Augmentation [17], [18], [19], Transfer Learning [20], [21], [22], and Fine-Tuning [23], [24]. But there is no study on the impact of each and combination of these techniques.

In this context, CNNs such as the well-known VGG16 architecture [25] could be trained to classify olive leaf diseases. VGG16 model was the winner of the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [26], which consists of classifying 1.2 million labeled images into a hundred classes using a 133GB dataset. The model already trained on this gigantic dataset is available, and through techniques such as Transfer Learning and Fine-Tuning, one can take advantage of feature extraction to train another classifier, if the new task and dataset have similarity to the initial task and dataset.

In this paper, our main contributions are: (*i*) A novel and public olive leaf disease classification dataset for the classification of virosis, fumagina, and nutritional deficiencies diseases that affect olive harvests in Tacna-Perú. (*ii*) Conduct an extensive comparative experimental study using all possible configurations of Data Augmentation, Transfer Learning, and Fine-Tuning techniques to train a modified VGG16 architecture for olive leaf disease classification. (*iii*) Obtain and publish a Highly Accurate Deep Learning model trained using

Data Augmentation, Transfer Learning, and Fine-Tuning that solves the olive leaf disease classification task.

The rest of this paper is organized as follows: Section II provides the necessary background to understand the work. Section III presents the novel olive leaf disease classification dataset. Section IV defines the materials and methods used to carry out the experiments. Section V sets the experimental setup and pipeline of experiments. Section VI presents the experimental results and performance analysis of them. Section VII presents the discussion. Finally, Section VIII provides the conclusion and further work.

II. Background

A. Deep Learning

Deep Learning is a sub-field of machine learning that consists of using multiple layers of non-linear processing to learn data representations with multiple levels of abstraction [27].

B. CNNs

Convolutional Neural Networks (CNNs) are designed to process multiple types of data, especially two-dimensional images, and are directly inspired by the visual cortex of the brain. In the visual cortex, there is a hierarchy of basic cells: simple and complex [28]. A CNN can extract the image features without the need for this process to be performed by hand as before 2011, finally, adding a neural network at the end allows the classification task to be performed.

Fig. 1 shows a typical CNN architecture. Feature extraction consists of convolution layers (C1, C2, C3) and pooling layers (P1, P2). The classification consists of fully connected layers (FC) and an output layer [29].

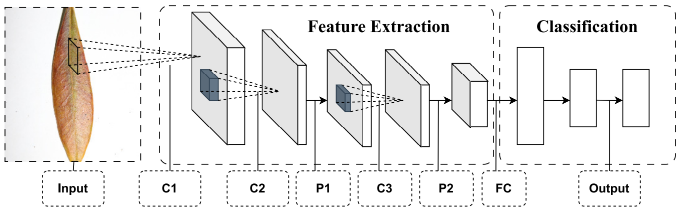

Fig. 1. Typical CNN architecture, based on [29].

C. VGG16 Architecture

The VGG16 architecture [25] was proposed by Simonyan and Sisserman in 2014, they developed the convolutional neural network in the Oxford Visual Geometry Group (VGG16). This architecture is constituted of 13 convolutional layers, each group of convolution layers is followed by a max pooling layer composing the feature extraction or base model, which is followed by three fully connected layers as classification or top model, hence the name includes 16. Finally, a softmax layer is added to the classifier.

VGG16 is a classification model which is able to classify 14M images of 1000 different categories (Imagenet dataset - ILSVRC 2014) with 92.7% accuracy. Despite the existence of more recent models, the simplicity of this architecture makes it perfect for applying techniques such as Transfer Learning and Fine-Tuning. The network (model and trained weights) is available in Keras. Fig. 2 shows the architecture of VGG16.

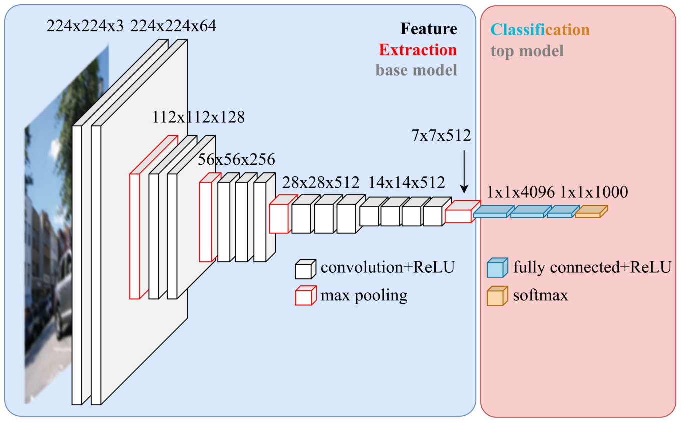

Fig. 2. VGG-16 architecture overview, based on [30].

D. Underfitting and Overfitting

Underfitting is a scenario where the model is not adjusted to the training dataset, causing a high error rate. In training across epochs, this happens when the model is not sufficiently capable of modeling the relationship between input and output.

When a model performs well with training data, it is necessary to consider a control group to ensure that the model performs well with data with which it has not been trained. That control group is better known as the validation set. It is possible for the model to perform very well on the training data but poorly on the validation data, which is known as overfitting (lack of generalization) [13].

E. Transfer Learning

Transfer Learning is a machine learning technique that seeks to leverage an already trained model for one task (pre-trained) into another [13], based on the assumption that the data with which the model has been initially trained are in the same feature space and have the same distribution as the new data. In cases where the latter is true, one could use the freeze feature extraction base of a pre-trained model and link it to a new classification network (with weights randomly initialized) to train only the latter and obtain good results.

This process results in a fast convergence of the model, either due to the constraints of feature extraction that have been frozen or due to the good compatibility of the dataset domains. When this is not sufficient, it is also necessary to train the feature extraction base, through a technique called Fine-Tuning.

F. Fine-Tuning

Fine-tuning is a technique that is applied over Transfer Learning to finish adjusting the model to the new dataset [13], for that purpose the feature extraction layers are unfrozen and the complete model is trained. It is important that this process is carried out after having trained the model classification layers and selected the best epoch for Fine-Tuning.

Fine-tuning allows the model to improve the feature extraction to obtain a better result in the classification task. The need to use Fine-Tuning, indicating that the feature space and distribution of the datasets (pre-trained and new) are not exactly the same, the distance between the feature space and distribution of datasets can be reflected in the model improvement.

G. Data Augmentation

Data augmentation is the generation of synthetic data by perturbing the original data [13], this artificially allows to have a larger amount of data and avoids biases related to the perturbations performed. The use of this technique reduces the need for large amounts of data to train Deep Learning models. In addition, when data are difficult to obtain, it is possible to use Data Augmentation as a regularization technique to generalize better.

Data augmentation also prevents model overfitting by adding diversity and some randomness to the dataset, which allows models to be trained over more epochs. The most used deformations or transformations are related to rotation, zoom, width and height shift, and horizontal and vertical flips. It is very important to use these transformations taking into consideration the problem domain in order not to obtain undesired results.

H. Learning from Scratch

Learning from Scratch is the process of training a model with randomly initialized parameters or weights. This is the usual process when pre-trained models are not used. When pre-trained models are used, the weights are not randomly initialized, but rather those that have been fitted to a pre-training dataset are loaded. This process may be necessary in cases where a pre-trained model compatible with the dataset being worked with is not encountered.

I. Confusion Matrix

A confusion Matrix is an NxN table that summarizes the number of correct and incorrect predictions that a classification model made, where N is the number of classes. The following values are calculated: TP are True Positive, FP False Positive, FN False Negative, and TN are True Negative. Taking these values into account, different classification metrics can be calculated.

J. Classification Metrics

1) Accuracy: Number of correct predictions over the total number of predictions.

2) Loss: It is a measure of how much error is being made in the prediction. Specifically, this value will be 0 when the prediction is equal to the desired output. For multiclass classification, the categorical cross-entropy loss is often used, the definition of which is as follows:

$$L_{\mathrm{CE}}=-\sum_{i=1}^{n} t_i \log \left(p_i\right), \text { for n classes} \tag{1}$$

where t_i is the truth label and p_i is the Softmax probability for the i^{th} class.

3) Precision: Try to answer the following question: What proportion of positive identifications was correct? In the case of multi-class classification, one metric per class is obtained; is defined as follows:

$$\text{Precision } =\frac{TP}{TP+FP} \tag{2}$$

4) Recall: Try to answer the following question: What proportion of real positives was correctly identified? In the case of multi-class classification, one metric per class is obtained; is defined as follows:

$$\text{Recall } =\frac{TP}{TP+FN} \tag{3}$$

5) F1-Score: It is a single score defined as the harmonic mean of precision and recall. In the case of multi-class classification, one metric per class is obtained; is defined as follows:

$$F_1=2 \frac{\text{precision } \cdot \text{ recall}}{\text{precision } + \text{ recall}}=\frac{2\mathrm{TP}}{2\mathrm{TP}+\mathrm{FP}+\mathrm{FN}} \tag{4}$$

III. Novel Olive Leaf Disease Classification Dataset

A. Data Acquisition and Processing

Images of olive leaves found in crops at La Yarada-Los Palos in the Tacna region of Peru were collected in August of 2019 with the next most common diseases in the region.

1) Virosis: The name is descriptive of this disease of olive foliage due to the curved or sickle shape shown by the leaf affected with this disease. The disease is common in all commercial varieties of olives grown in South America [31].

2) Fumagina: It is a species of fungi, blackish in color, that covers plant tissues as a layer of soot, which hinders photosynthesis so that the affected olive trees see their productive capacity diminished. This layer comes off when passing the finger over the affected parts [32].

3) Nutritional Deficiency: Leaves with abnormal colorations at the terminal end of the leaves such as phosphorus and potassium [33].

The RGB images of 3984 x 2656 px were taken using a Canon EOS Rebel T6i 24.2MPX camera and fixed configurations (55 mm focal distance, 1/200s aperture, ISO-400). To highlight the characteristics of the leaves and keep the image focused and stable, a device was used to stabilize and focus the mobile device at a height of 30 cm. Each leaf was placed showing its frontal area on a blank sheet of paper.

One way to distinguish between diseased leaves is to focus on the level of green, brown, and yellow color. However, this is a criterion that does not apply to all cases, and an agronomic expert is usually required. Fig. 3 shows samples of the images taken.

The procedure for the construction of the olive leaf diseases dataset includes three phases. (i) First, images of olive trees

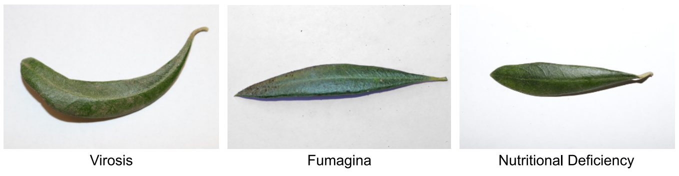

Fig. 3. Samples of images and classes of dataset.

are captured, selecting three diseases according to the study. (ii) Second, after the images are collected, they are given to an agronomist specialist expert in olive disease images. (iii) Finally, the specialist filtered the images and manually labeled them according to their characteristics.

The total images acquired resulted in 773 useful images, of which 258 are nutritional deficiency disease, 257 are fumagina, and 258 are virosis. In order to have class balance, a 77:19:3 split was performed to form the training, validation, and test sets. Table I specifies the final number of images per class and dataset.

TABLE I. NOVEL OLIVE LEAF DISEASE CLASSIFICATION DATASET SIZE

Category	Training Set	Validation Set	Test Set	Total
Deficiences	200	50	8	**258**
Fumagina	200	50	7	**257**
Virosis	200	50	8	**258**
Total	**600**	**150**	**23**	**773**

Finally, the novel olive leaf disease dataset is made available in our GitHub for further use and evaluation.

IV. MATERIALS AND METHOD

A. Experimental Platform

For the design of the experiments and their execution, the following computer hardware was made available to us through a Google Colab Pro subscription: GPU 1x Nvidia Tesla T4 with 15GB of RAM and CPU 1x Xeon Processors @2.3Ghz with 12.7 GB of RAM.

Keras, one of the main Deep Learning APIs written in Python, was used in this study. It is compatible with multiple back-end neural network computation engines and runs on the TensorFlow machine learning platform. Versions were: Keras 2.11.0 and Tensorflow 2.11.0 as the backend in a Python version 3.8 environment. Each epoch of training takes 12 seconds and the total amount of time for 10 runs of all experiments was 33.3 hours.

B. Deep Learning Architecture

The selected Deep Learning architecture was VGG16 [25], chosen because it extracts features at a low level by using a smaller kernel size, demonstrating a good feature extraction capability for image classification. As a Deep Learning classification architecture, it is composed of 2 sub-models: feature extraction (base) and classification (top). Feature extraction was carried out by a convolutional neural network. Table II presents the architecture of the feature extraction basis of VGG16 and the number of parameters.

TABLE II. VGG16 BASE MODEL ARCHITECTURE

Layer (type)	Output Shape	Number of Parameters
input_1 (InputLayer)	224 x 224 x 3	0
block1_conv1 (Conv2D)	224 x 224 x 64	1,792
block1_conv2 (Conv2D)	224 x 224 x 64	36,928
block1_pool (MaxPooling2D)	112 x 112 x 64	0
block2_conv1 (Conv2D)	112 x 112 x 128	73,856
block2_conv2 (Conv2D)	112 x 112 x 128	147,584
block2_pool (MaxPooling2D)	56 x 56 x 128	0
block3_conv1 (Conv2D)	56 x 56 x 256	295,168
block3_conv2 (Conv2D)	56 x 56 x 256	590,080
block3_conv3 (Conv2D)	56 x 56 x 256	590,080
block3_pool (MaxPooling2D)	28 x 28 x 256	0
block4_conv1 (Conv2D)	28 x 28 x 512	1,180,160
block4_conv2 (Conv2D)	28 x 28 x 512	2,359,808
block4_conv3 (Conv2D)	28 x 28 x 512	2,359,808
block4_pool (MaxPooling2D)	14 x 14 x 512	0
block5_conv1 (Conv2D)	14 x 14 x 512	2,359,808
block5_conv2 (Conv2D)	14 x 14 x 512	2,359,808
block5_conv3 (Conv2D)	14 x 14 x 512	2,359,808
block5_pool (MaxPooling2D)	7 x 7 x 512	0
Total		**14,714,688**

Classification was carried out by a dense neural network. Original VGG16 architecture [25] uses three sequential dense layers as classifier (top-model), that allow mapping the model input to 1000 classes (because it was conceived for Imagenet). The architecture was modified by replacing the top-model with a max-pooling layer (to reduce the dimensionality of the extracted features) connected to a dense layer that maps the model input to 3 classes. Table III presents our modified VGG16 architecture and the number of parameters.

It was hypothesized that only one dense layer is sufficient to map features to classes, due to the complexity of our dataset compared with ImageNet, and perform the problem task. This hypothesis is proven through our experiments.

TABLE III. OUR MODIFIED VGG16 ARCHITECTURE

Layer (component)	Output Shape	Number of Parameters
input_1 (Input Layer)	224 x 224 x 3	0
vgg16_base_model	7 x 7 x 512	14,714,688
Base Model (Feature Extraction)		**14,714,688**
GlobalAveragePooling2D (new)	512	0
Dense (new)	3	1,539
Top Model (Classification)		**1,539**
Total		**14,716,227**

V. EXPERIMENTAL SETUP

Fig. 4 presents the complete pipeline of every experiment carried out which is detailed below.

The following comparative objectives were considered in order to achieve the objective of the study:

1) To determine the effect of Data Augmentation.

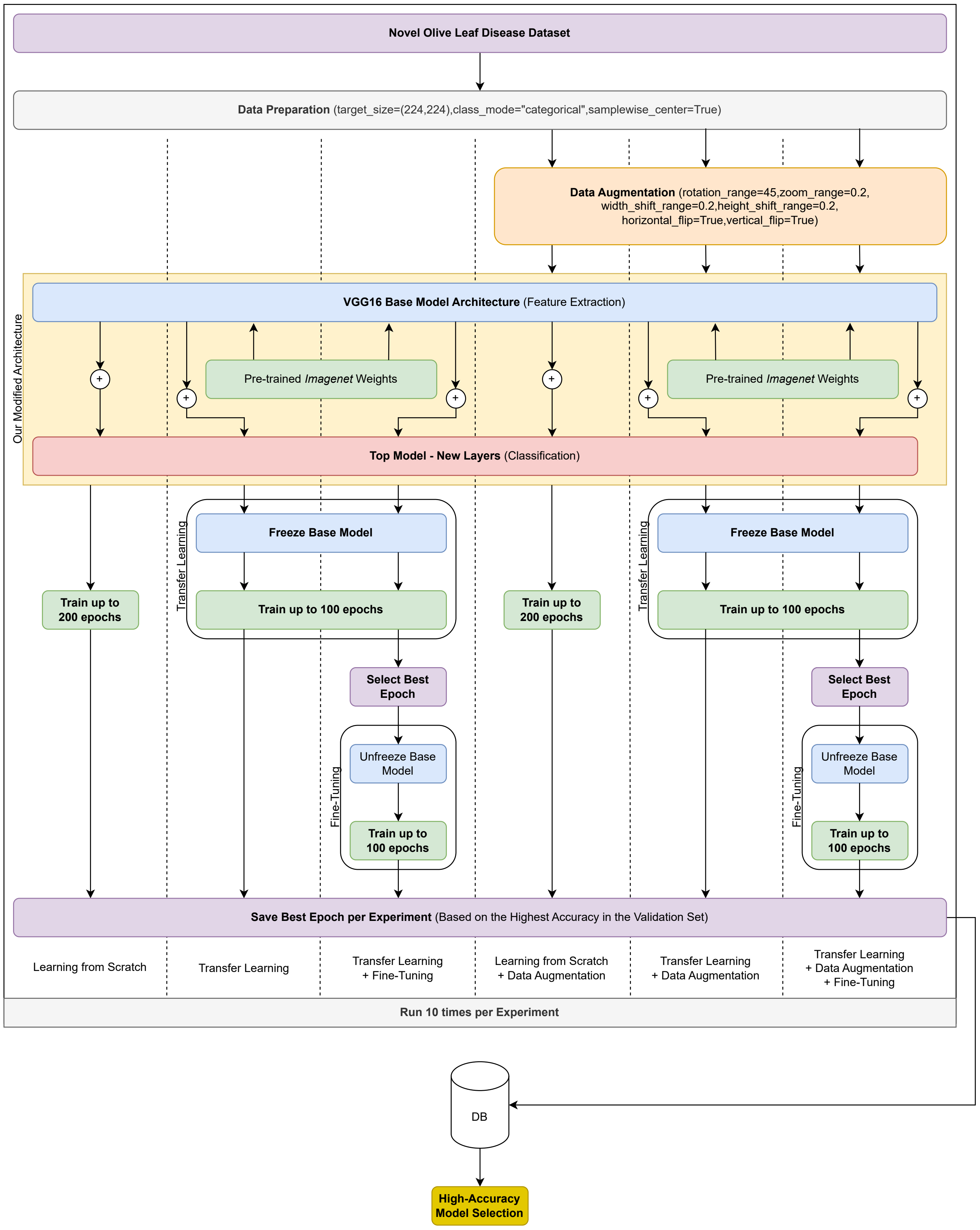
Novel Olive Leaf Disease Dataset
Data Preparation (target_size=(224,224),class_mode="categorical",samplewise_center=True)
Data Augmentation (rotation_range=45,zoom_range=0.2, width_shift_range=0.2,height_shift_range=0.2, horizontal_flip=True,vertical_flip=True)
Our Modified Architecture
VGG16 Base Model Architecture (Feature Extraction)
Pre-trained Imagenet Weights
Pre-trained Imagenet Weights
Top Model - New Layers (Classification)
Transfer Learning
Freeze Base Model
Train up to 100 epochs
Train up to 200 epochs
Select Best Epoch
Fine-Tuning
Unfreeze Base Model
Train up to 100 epochs
Save Best Epoch per Experiment (Based on the Highest Accuracy in the Validation Set)
Learning from Scratch
Transfer Learning
Transfer Learning + Fine-Tuning
Learning from Scratch + Data Augmentation
Transfer Learning + Data Augmentation
Transfer Learning + Data Augmentation + Fine-Tuning
Run 10 times per Experiment
DB
High-Accuracy Model Selection

Fig. 4. Pipeline of experiments.

2) To determine the effect of Transfer Learning.
3) To determine the effect of Fine-Tuning.

For data preparation, the RGB images in the dataset were resized to the input size of the selected architecture (224,224) and each sample was scaled to mean 0. In addition, the labels of each of the images are categorized as a dummy variable to match the output size of the modified architecture. This was done in all experiments.

For Data Augmentation, which is used in some experiments, the images sent to the model for training are artificially enhanced through the following random transformations:

1) 0-45 degree random rotation.
2) 0.8-1.2 zoom.
3) 0-0.2 of total width shift.
4) 0-0.2 of total height shift.
5) Horizontal flip.
6) Vertical flip.

We describe this configuration as “Aggressive Data Augmentation” because it achieves a transformation that rotates the image from 0º to 359º with a variety of zooms, shifts, and flips.

Finally, for training models, the loss was Categorical Cross Entropy and the optimizer was ADAM [34].

In order to obtain more trustworthy results, each of the following experiments of all possible setups of Transfer Learning, Fine-Tuning, and Data Augmentation was run 10 times:

A. *Learning from Scratch*

For this experiment, the modified architecture was trained up to 200 epochs and the best-performing epoch is selected.

B. *Learning from Scratch + Data Augmentation*

For this experiment, the images sent to the model for training were artificially enhanced through Data Augmentation. Then, the modified architecture was trained up to 200 epochs and the best-performing epoch was selected.

C. *Transfer Learning*

For this experiment, pre-trained Imagenet weights were loaded into the base model of our modified architecture. Then, the base model was frozen and only the top model was trained up to 100 epochs.

D. *Transfer Learning + Fine-Tuning*

For this experiment, the best epoch of the Transfer Learning experiment is taken to unfreeze the base model and train the whole architecture up to 100 epochs.

E. *Transfer Learning + Data Augmentation*

For this experiment, the images sent to the model for training were artificially enhanced through Data Augmentation, and pre-trained Imagenet weights were loaded into the base model of our modified architecture. Then, the base model was frozen and only the top model was trained up to 100 epochs.

F. *Transfer Learning + Data Augmentation + Finetuning*

For this experiment, the best epoch of the Transfer Learning + Data Augmentation experiment is taken to unfreeze the base model and train the whole architecture up to 100 epochs.

For all experiments, the evaluation for best epoch selection was made taking into account the accuracy in the validation set. Because the experiment was run 10 times, each selection of the best model was stored in a database in order to finally select the highest accurate model.

All numbers of epochs frame the underfitting and overfitting processes of model training, so in none of the cases was the number of training epochs insufficient.

VI. Experimental Results

A. *Impact of Data Augmentation*

From Fig. 5, the accuracy in the training set and validation set over 200 epochs is presented with an interquartile range on a model trained Learning from Scratch. The best validation accuracy was 0.86 with a validation loss of 0.558051.

From Fig. 6, the accuracy in the training set and validation set over 200 epochs is presented with an interquartile range on a model trained Learning from Scratch. The best validation accuracy was 0.8 with a validation loss of 0.641917.

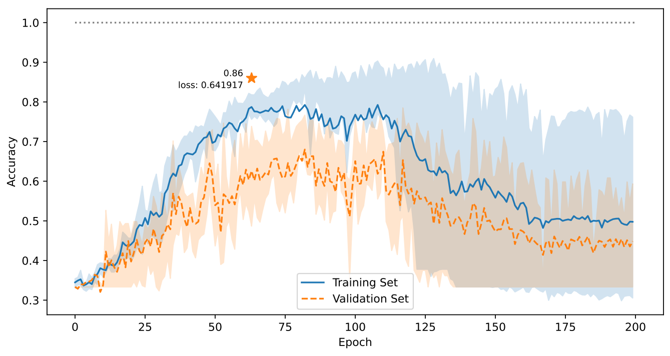

Fig. 5. Accuracy of learning from scratch.

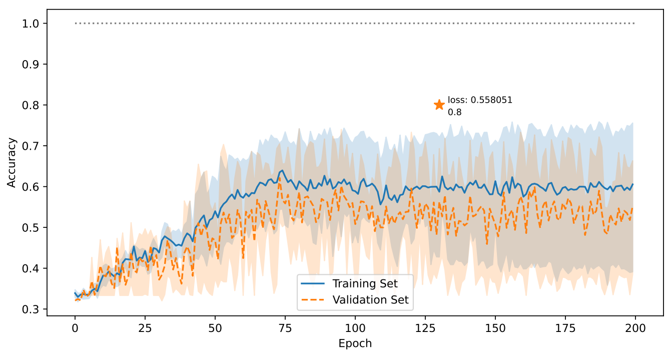

Fig. 6. Accuracy of learning from scratch + data augmentation.

Although the accuracy was higher without using data augmentation, the loss was lower when using data augmentation, which means that without data augmentation the model was less sure of its prediction. On the other hand, without the use of data augmentation, the best validation accuracy was reached in 63 epochs, with the use of Data Augmentation it was reached in 130 epochs, which shows that Data Augmentation allows for avoiding overfitting in training. Moreover, both models achieve similar performances.

It is worth mentioning that, while the model without data augmentation remains stationary, the model trained without data augmentation decreases in validation value. This denotes that Learning from Scratch was not able to update the model in such a way that it adjusts to the variability of the data caused by our aggressive data augmentation.

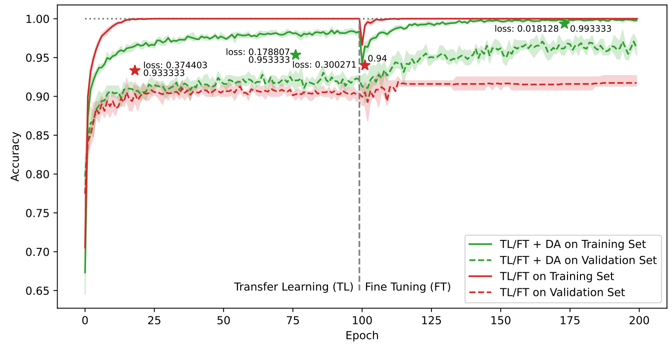

Fig. 7. Accuracy of transfer learning vs transfer learning with data augmentation.

From Fig. 7, the accuracy in the training set and validation set over 200 epochs is presented with an interquartile range on a model trained Transfer Learning (left side), Transfer Learning + Data Augmentation (left side), Transfer Learning + Fine-Tuning (right side), and Transfer Learning + Fine-Tuning + Data Augmentation (right side).

For Transfer Learning the best validation accuracy was 0.93333 with a validation loss of 0.374403 and was reached at epoch 18, for Transfer Learning + Data Augmentation the best validation accuracy was 0.953333 with a validation loss of 0.178807 and was reached at epoch 76, for Transfer Learning + Fine-Tuning the best validation accuracy was 0.94 with a validation loss of 0.300271 and was reached at epoch 101, and for Transfer Learning + Fine-Tuning + Data Augmentation the best validation accuracy was 0.993333 with a validation loss of 0.018128 and was reached at epoch 176.

The accuracy was higher in the validation set using Data Augmentation in Transfer Learning and Transfer Learning + Fine Tuning. On the other hand, without the use of Data Augmentation, the best validation accuracy was reached before vs with the use of Data Augmentation. Moreover, both models achieve similar performances.

Data Augmentation when applied improves accuracy by 2% in Transfer Learning and 5% in Transfer Learning + Fine Tuning. This difference is consistent with the fact that Data Augmentation generates a greater variety of input data and Fine-Tuning is able to take advantage of this variety because it has unfrozen feature extraction layers.

From Fig. 8, the loss in the training set and validation set over 200 epochs is presented with an interquartile range on a

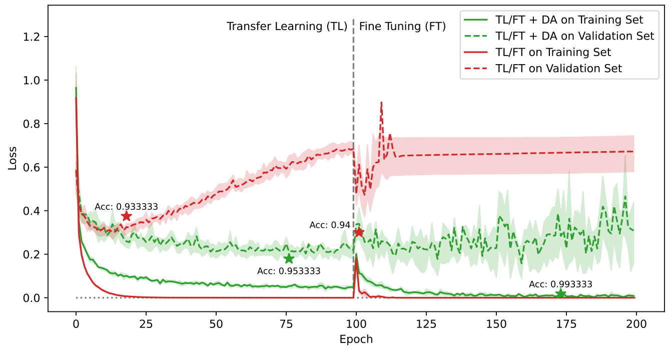

Fig. 8. Loss of transfer learning vs transfer learning with data augmentation.

model trained Transfer Learning (left side), Transfer Learning + Data Augmentation (left side), Transfer Learning + Fine-Tuning (right side), and Transfer Learning + Fine-Tuning + Data Augmentation (right side).

It can be observed that training using Data Augmentation avoids the rapid overfitting to which the model is prone due to the low variability of the data.

Furthermore, the use of Data Augmentation in Fine-Tuning did not show an improvement over 1% when Data Augmentation was not applied, where the model tends even more quickly to fall into overfitting than in Transfer Learning.

B. *Impact of Transfer Learning*

Fig. 9 shows the average accuracy over all experiments. The difference between using any configuration of Learning from Scratch vs any configuration using Transfer Learning was noticeable in the graph so that the lines do not overlap at any point. The use of transfer learning, regardless of also using data augmentation, improved the accuracy by at least 10% on average.

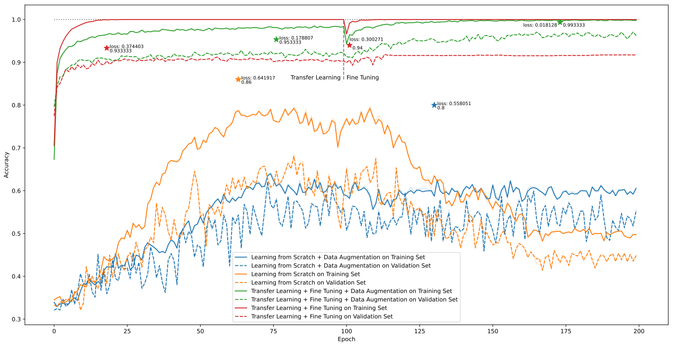

Fig. 9. Average accuracy of all experiments.

Fig. 11 shows the run with the best validation accuracy over all experiments. Again, the difference between using any configuration with Learning from Scratch vs any configuration using Transfer Learning is noticeable in the graph so that the lines do not overlap at any point. The use of transfer learning, regardless of also using data augmentation, improved the accuracy by at least 7.33333% on the best experiment run.

C. Impact of Fine-Tuning

From Fig. 7 (described above), for Transfer Learning (with or without Data Augmentation) the best validation accuracy was 0.953333 with a validation loss of 0.178807 and was reached at epoch 76, for Transfer Learning + Fine-Tuning the best validation accuracy was 0.993333 with a validation loss of 0.018128 and was reached at epoch 173.

Fine-Tuning when applied improves accuracy by less than 1% in Transfer Learning and 4% in Transfer Learning + Data Augmentation. This difference is consistent with the fact that some specific-domain problems need an adjustment in the feature extraction layers of the model to improve the performance.

Additionally, without Data Augmentation the difference between Transfer Learning and Transfer Learning + Fine-Tuning was similar.

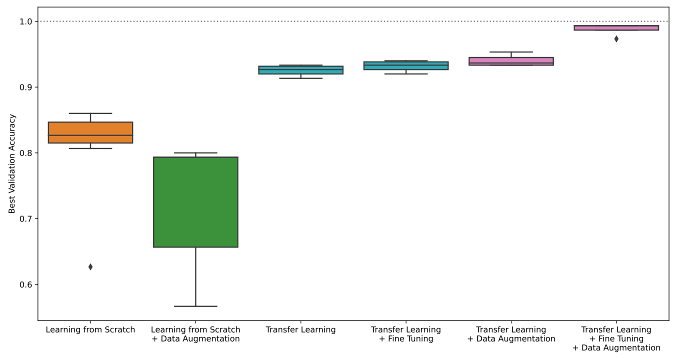

Fig. 10. Boxplot of validation accuracy.

From Fig. 10, all experiments are presented in a boxplot that resumes the best validation accuracies over 10 runs of each experiment. It can be observed that the use of Fine-Tuning over Transfer Learning without Data Augmentation gave similar results. Additionally, there was a clear improvement and separation between the values with respect to the combined use of Transfer Learning + Fine-Tuning + Data Augmentation. Furthermore, the range of values was lowest than in any other experiment.

D. Highly Accurate Deep Learning Model

From Fig. 11, the best run of all experiments is presented. The criterion used was to choose the model in whose: best epoch with the highest accuracy and lowest loss in the validation set has been obtained. Therefore, the model of run 0, epoch 173, loss 0.000008, accuracy 1.0, validation loss 0.018128, validation accuracy 0.993333 that made use of Transfer Learning + Fine-Tuning + Data Augmentation was selected as a Highly Accurate Deep Learning Model.

From Table IV, metrics of all experiments are presented, and the best values are highlighted in bold. In the validation set, the best result was obtained for all metrics in the combined use of Transfer Learning + Fine-Tuning + Data Augmentation, except for the interquartile range. However, taking into consideration the Q4-Q1 range this result was still the best. This demonstrates, due to the execution of 10 runs, that the result

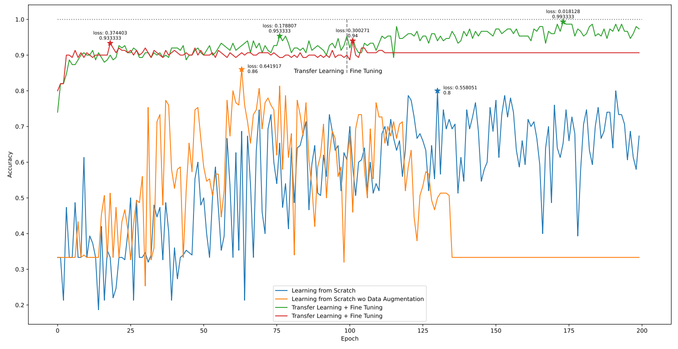

Fig. 11. Best run accuracy of all experiments.

is the product of a consistent improvement provided by the experiment setup.

Once the model has been selected, is submitted to the test set to corroborate the performance. Table V shows the confusion matrix of the selected model over training, validation, and test sets. In overall sets, only in the validation set, there is one sample that belongs to the virosis class but is classified as fumagina by the model.

Table VI shows additional metrics for each disease to be classified. The average F1-Score over all diseases was 0.9933 on the validation set. The value coincides with the average accuracy of the model in the validation set due to the fact that the dataset is balanced with respect to its classes.

Finally, the highly accurate model is made available on our GitHub for further use and evaluation.

VII. Discussions

Regarding other studies related to the detection of olive leaf diseases [7], [8], [9], [10], [11], [12], it is not possible to make a direct comparison because those studies and the present are case studies of different olive leaf diseases in different countries. However, we agree with them that it is possible to classify olive leaf diseases using Deep Learning.

The results obtained reaffirm the findings regarding the use of Data Augmentation [17], [18], [19], Transfer Learning [20], [21], [22], and Fine-Tuning [23], [24]. In addition, our extensive experimental comparative study introduces new findings on the combination of these techniques, which are presented in the conclusions.

VIII. Conclusions

A novel and public olive leaf dataset for the classification of Virosis, Fumagina, and nutritional deficiencies diseases that affect olive harvests at La Yarada-Los Palos in Tacna Region, Perú was presented.

In addition, extensive comparative experimental studies were conducted using all possible configurations of Data Augmentation, Transfer Learning, and Fine-Tuning with the next conclusions: (i) Ineffectiveness of Data Augmentation when the model Learning from Scratch, (ii) High improvement by using Transfer Learning vs Learning from Scratch, (iii) Similar performance using Transfer Learning vs Transfer Learning

TABLE IV. Metrics of All Experiments

Set	Metric	Config:	LFS	LFS+DA	TL	TL+DA	TL+FT	TL+FT+DA
Training Set	**Loss**	min ↓	0.40205	0.64825	0.00002	0.03743	0.00016	**0.00001**
		max ↓	4.36961	2.36000	0.05210	0.07445	0.08969	**0.03955**
		median ↓	0.65133	0.92559	**0.00060**	0.05794	0.01349	0.00242
		mean ↓	1.20267	1.13607	**0.00710**	0.05766	0.02150	0.00854
		iqr ↓	0.63291	0.45144	**0.00356**	0.01647	0.01919	0.01357
	Accuracy	min ↑	0.36833	0.42000	0.98167	0.97000	0.98167	**0.99500**
		max ↑	0.94500	0.77000	**1.00000**	0.98500	**1.00000**	**1.00000**
		median ↑	0.83250	0.69750	**1.00000**	0.97750	0.99667	0.99833
		mean ↑	0.80983	0.64067	0.99800	0.97800	0.99450	**0.99850**
		iqr ↓	0.08542	0.21583	**0.00000**	0.00958	0.00500	0.00167
Validation Set	**Loss**	min ↓	0.37960	0.51447	0.30650	0.15781	0.30027	**0.01813**
		max ↓	1.50668	3.66409	0.51401	0.29471	1.06716	**0.23049**
		median ↓	0.75769	0.66646	0.39528	0.19313	0.50248	**0.07109**
		mean ↓	0.79619	0.99644	0.40222	0.20341	0.53828	**0.07605**
		iqr ↓	0.42499	0.32405	0.08068	**0.03825**	0.15090	0.04152
	Accuracy	min ↑	0.62667	0.56667	0.91333	0.93333	0.92000	**0.97333**
		max ↑	0.86000	0.80000	0.93333	0.95333	0.94000	**0.99333**
		median ↑	0.82667	0.79333	0.92667	0.93667	0.93333	**0.99333**
		mean ↑	0.81267	0.73333	0.92467	0.93933	0.93200	**0.98933**
		iqr ↓	0.03167	0.13667	0.01167	0.01167	0.01167	**0.00667**

TABLE V. Confusion Matrix

	Training Set			
	Class	**Deficiency**	**Fumagina**	**Virosis**
True Class	**Deficiency**	200	0	0
	Fumagina	0	200	0
	Virosis	0	0	200
	Validation Set			
	Class	**Deficiency**	**Fumagina**	**Virosis**
	Deficiency	50	0	0
	Fumagina	0	50	0
	Virosis	0	1	49
	Test Set			
	Class	**Deficiency**	**Fumagina**	**Virosis**
	Deficiency	8	0	0
	Fumagina	0	7	0
	Virosis	0	0	8
	Predicted Class			

TABLE VI. Precision, Recall, and F1-Score per Disease

	Training Set		
Disease	**Precision**	**Recall**	**F1-Score**
Deficiences	1.0000	1.0000	1.0000
Fumagina	1.0000	1.0000	1.0000
Virosis	1.0000	1.0000	1.0000
	Validation Set		
Disease	**Precision**	**Recall**	**F1-Score**
Deficiences	1.0000	1.0000	1.0000
Fumagina	0.9804	1.0000	0.9901
Virosis	1.0000	0.9800	0.9899
	Test Set		
Disease	**Precision**	**Recall**	**F1-Score**
Deficiences	1.0000	1.0000	1.0000
Fumagina	1.0000	1.0000	1.0000
Virosis	1.0000	1.0000	1.0000

+ Fine-Tuning vs Transfer Learning + Data Augmentation, and (iv) Very high improvement using Transfer Learning + Fine-Tuning + Data Augmentation.

Finally, a highly accurate Deep Learning model (100%, 99.33%, and 100% of accuracy in the training, validation, and test set respectively) based on modified VGG16 architecture using Data Augmentation, Transfer Learning, and Fine-Tuning to solve the olive leaf disease classification task was obtained and published. F1-Score was 1 for the diseases in training and test sets, and 1, 0.9901, and 0.9899 in the Nutritional Deficiences, Fumagina, and Virosis diseases.

Making the dataset and selected model public allows for the reproducibility of the results. Future works could deploy this trained model to a mobile or edge device for validation and agricultural use, experiment with different Data Augmentation configurations, and compare VGG16 with other models.

Acknowledgment

We would like to express our appreciation and thanks to the NVIDIA Deep Learning Institute (DLI) for providing the Free Faculty Development Workshop "Fundamentals of Deep Learning" on February 18, 2022.

This research was supported by Canon, Sobrecanon, and Mining Royalties of the Universidad Nacional Jorge Basadre Grohmann - Tacna - Perú.

References

[1] MINCETUR. Perfil de mercado y competitividad exportadora de la aceituna. https://www.mincetur.gob.pe/wp-content/uploads/

documentos/comercio_exterior/plan_exportador/publicaciones/Aceituna.pdf, 2022.

[2] DIRESA. Anuario estadístico agrario 2020 región tacna. www.agritacna.gob.pe, 2020.

[3] H Baumann. Agraria.pe. exportaciones de la aceituna perú. https://agraria.pe/buscar?q=exportaciones+de+aceituna+peru. Accessed: 2022-09-30.

[4] SENASA. Tacna promueve control de plagas. https://www.senasa.gob.pe/senasacontigo/tacna-senasa-promueve-control-de-plagas-del-olivo-con-metodos-no-contaminantes/, 2021.

[5] FAO. El trabajo de la fao sobre el cambio climático conferencia de las naciones unidas sobre el cambio climático 2019. http://www.fao.org/3/a-i8037s.pdf.

[6] Martín Eloy Casilla García. Análisis de los factores que influyen en la vecería del olivo (olea europea l.) en la región tacna. 2011.

[7] Mohamed Lachgar, Hamid Hrimech, Ali Kartit, et al. Optimization techniques in deep convolutional neuronal networks applied to olive diseases classification. *Artificial Intelligence in Agriculture*, 6:77–89, 2022.

[8] Madallah Alruwaili, Saad Alanazi, Sameh Abd El-Ghany, and Abdulaziz Shehab. An efficient deep learning model for olive diseases detection. *International Journal of Advanced Computer Science and Applications*, 10(8), 2019.

[9] Amel Ksibi, Manel Ayadi, Ben Othman Soufiene, Mona M Jamjoom, and Zahid Ullah. Mobires-net: A hybrid deep learning model for detecting and classifying olive leaf diseases. *Applied Sciences*, 12(20):10278, 2022.

[10] Sinan Uğuz and Nese Uysal. Classification of olive leaf diseases using deep convolutional neural networks. *Neural computing and applications*, 33(9):4133–4149, 2021.

[11] Hamoud H Alshammari, Ahmed I Taloba, and Osama R Shahin. Identification of olive leaf disease through optimized deep learning approach. *Alexandria Engineering Journal*, 72:213–224, 2023.

[12] Hamoud Alshammari, Karim Gasmi, Ibtihel Ben Ltaifa, Moez Krichen, Lassaad Ben Ammar, and Mahmood A Mahmood. Olive disease classification based on vision transformer and cnn models. *Computational Intelligence and Neuroscience*, 2022, 2022.

[13] Magnus Ekman. *Learning Deep Learning: Theory and Practice of Neural Networks, Computer Vision, NLP, and Transformers Using TensorFlow*. Addison-Wesley Professional, 2021.

[14] Jiraporn Thomkaew and Sarun Intakosum. Improvement classification approach in tomato leaf disease using modified visual geometry group (vgg)-inceptionv3. *International Journal of Advanced Computer Science and Applications*, 13(12), 2022.

[15] Orlando Iparraguirre-Villanueva, Victor Guevara-Ponce, Ofelia Roque Paredes, Fernando Sierra-Liñan, Joselyn Zapata-Paulini, and Michael Cabanillas-Carbonell. Convolutional neural networks with transfer learning for pneumonia detection. 2022.

[16] Mimoun Yandouzi, Mounir Grari, Idriss Idrissi, Mohammed Boukabous, Omar Moussaoui, Mostafa Azizi, Kamal Ghoumid, and Aissa Kerkour Elmiad. Forest fires detection using deep transfer learning. *International Journal of Advanced Computer Science and Applications*, 13(8), 2022.

[17] Hong Chun Teo, Ummi Rabaah Hashim, Sabrina Ahmad, Lizawati Salahuddin, Hea Choon Ngo, and Kasturi Kanchymalay. Efficacy of the image augmentation method using cnn transfer learning in identification of timber defect. *International Journal of Advanced Computer Science and Applications*, 13(5), 2022.

[18] Walid Al-Dhabyani, Mohammed Gomaa, Hussien Khaled, and Fahmy Aly. Deep learning approaches for data augmentation and classification of breast masses using ultrasound images. *Int. J. Adv. Comput. Sci. Appl*, 10(5):1–11, 2019.

[19] Sudeepthi Govathoti, A Mallikarjuna Reddy, Deepthi Kamidi, G BalaKrishna, Sri Silpa Padmanabhuni, and Pradeepini Gera. Data augmentation techniques on chilly plants to classify healthy and bacterial blight disease leaves. *International Journal of Advanced Computer Science and Applications*, 13(6), 2022.

[20] Priyanka Jaiswal, Vijay Katkar, and SG Bhirud. Multi oral disease classification from panoramic radiograph using transfer learning and xgboost. *International Journal of Advanced Computer Science and Applications*, 13(12), 2022.

[21] Farhana Alam, Farhana Chowdhury Tisha, Sara Anisa Rahman, Samia Sultana, Md Ahied Mahi Chowdhury, Wasif Reza Ahmed, and Mohammad Shamsul Arefin. Automated brain disease classification using transfer learning based deep learning models. *International Journal of Advanced Computer Science and Applications*, 13(9), 2022.

[22] Jayapalan Senthil Kumar, Syahid Anuar, and Noor Hafizah Hassan. Transfer learning based performance comparison of the pre-trained deep neural networks. *International Journal of Advanced Computer Science and Applications*, 13(1), 2022.

[23] Usman Bello Abubakar, Moussa Mahamat Boukar, and Steve Adeshina. Evaluation of parameter fine-tuning with transfer learning for osteoporosis classification in knee radiograph. *International Journal of Advanced Computer Science and Applications*, 13(8), 2022.

[24] Fadi Alharbi, Murtada K Elbashir, Mohanad Mohammed, and Mohamed Elhafiz Mustafa. Fine-tuning pre-trained convolutional neural networks for women common cancer classification using rna-seq gene expression. *International Journal of Advanced Computer Science and Applications*, 11(11), 2020.

[25] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[26] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[27] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[28] Seonwoo Min, Byunghan Lee, and Sungroh Yoon. Deep learning in bioinformatics. *Briefings in bioinformatics*, 18(5):851–869, 2017.

[29] Shanwen Zhang, Wenzhun Huang, and Chuanlei Zhang. Three-channel convolutional neural networks for vegetable leaf disease recognition. *Cognitive Systems Research*, 53:31–41, 2019.

[30] D Frossard. Vgg in tensorflow: Model and pre-trained parameters for vgg16 in tensorflow. *Department of Computer Science, University of Toronto, Toronto, ON*, 2016.

[31] René Chávez Alfaro, Eloy Casilla Garcia, Luis Salazar, Germán Sepulveda, Alfredo Huarachi, and Ida Bartolini. Avances en la investigación colaborativa y control integrado de la" hoja de hoz" en los cultivos de olivo de tacna y arica. *Ciencia & Desarrollo*, (7):7–12, 2003.

[32] JL Molina de la Rosa, B Jiménez Herrera, F Ruiz Coleto, F García Zamorano, J Cano Rodríguez, and J Pérez García. Técnicas de cultivo: Plagas y enfermedades del olivo. *Consejería de Agricultura, Pesca y Desarrollo Rural: Instituto de Investigación y Formación Agraria y Pesquera. Recuperado de https://www. juntadeandalucia. es/export/drupaljda/publicacion/17/07/1.% 20Plagas% 20y% 20enferm_olivo_2017% 20BAJA. pdf*, 2017.

[33] R Fernández-Escobar, M Guerreiro, M Benlloch, and M Benlloch-González. Symptoms of nutrient deficiencies in young olive trees and leaf nutrient concentration at which such symptoms appear. *Scientia Horticulturae*, 209:279–285, 2016.

[34] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.