Convolutional Neural Networks for Automatic Classification of Diseased Leaves: The Impact of Dataset Size and Fine-Tuning

Giovanny Caluña[1(✉)], Lorena Guachi-Guachi[1,2], and Ramiro Brito[3]

[1] Yachay Tech University, Hacienda San José, Urcuquí 100119, Ecuador
{giovanny.caluna,lguachi}@yachaytech.edu.ec
[2] SDAS Research Group, Ibarra, Ecuador
[3] Department of Mechatronics, Universidad Internacional del Ecuador, Av. Simon Bolivar, Quito 170411, Ecuador
http://www.sdas-group.com

Abstract. For agricultural productivity, one of the major concerns is the early detection of diseases for their crops. Recently, some researchers have begun to explore Convolutional Neural Networks (CNNs) in agricultural field for leaves diseases identification. A CNN is a category of deep artificial neural networks that has demonstrated great success in computer vision applications, such as video and image analysis. However, their drawbacks are the demand of huge quantity of data with a wide range of conditions, as well as a carefully fine-tuning to work properly. This work explores and compares the most outstanding five CNNs architectures to determine their ability to correctly classify a leaf image as healthy and unhealthy. Experimental tests are performed referring to an unbalanced and small dataset composed by healthy and diseased leaves. In order to achieve a high accuracy on the explored CNN models, a fine-tuning of their hyperparameters is performed. Furthermore, some variations are done on the raw dataset to increase the quality and variety of the leaves images. Preliminary results provide a point-of-view for selecting CNNs architectures for leaves diseases identification based on accuracy, precision, recall and F1 metrics. The comparison demonstrates that without considerably lengthening the training, ZFNet achieves a high accuracy and increases it by 10% after 50 K iterations being a suitable CNN model for identification of diseased leaves using datasets with a small variation, number of classes and dataset sizes.

Keywords: Leaf diseases classification · Convolutional Neural Networks (CNNs) · Image classification

1 Introduction

One of the major issues affecting farmers around the world are plant diseases caused by various pests, viruses, and insects, resulting in huge losses of money.

© Springer Nature Switzerland AG 2020
O. Gervasi et al. (Eds.): ICCSA 2020, LNCS 12249, pp. 951–966, 2020.
https://doi.org/10.1007/978-3-030-58799-4_68

The best way to relieve and combat this is timely detection. In almost all real solutions, early detection is determined by human, such an expensive and time-consuming solution.

Automatic approaches for leaf diseases identification aim to identify the signs of diseases at initial phase and decrease the huge effort of human observing in large farms. These methods involve some stages such image acquisition (to obtain images to be analyzed), image pre-processing (to enhance images quality), feature extraction (to get useful properties from images pixels), and classification (to categorize an image in classes based on its discriminating features).

Recent research have begun to explore CNNs approaches in medicine [1], mechanical [2] and in the agricultural field to identify plant disease and pests from images. CNNs, a category of deep learning approaches, have been extensively adopted for image classification purposes. They have been introduced for general purposes analysis of visual imagery and are characterized by their ability to learn key feature on their own. Their adoption for specific fields brings with it several challenges such as choosing the appropriate hyperparameters values such as learning rate, step and maximum number of iterations, as well as providing a sufficient quantity and quality of training dataset.

In order to determine how the CNNs structure, hyperparameter values and data quality influence performance reached in the classification of leaves diseases, this work explores and compare the five outstanding and standard CNN architectures such as Inception V3 [3], GoogLeNet [4], ZFNet [5], ResNet 50 [6], and ResNet 101 [6], which outperform other techniques of image classification. Furthermore, we evaluate some pre-processing tasks on the Plant Village dataset [7], a set of images of diseased and healthy crops such as pepper, tomato and potato, and perform fine-tuning of the hyperparameter of each CNN model x.

The remaining of this paper is organized as follows. Section 2 describes most relevant related works. The raw dataset and the operations applied for increasing its quality and size are described in Sect. 3. Explored CNN architectures are presented in Sect. 4. Experimental setup is presented in Sect. 5. Experimental results, obtained on processed datasets, are gathered and discussed in Sect. 6. Finally, Sect. 7 deals with the concluding remarks and future work.

2 Related Works

Among various CNN architectures used in agricultural field: CaffeNet, AlexNet, GoogleNet, VGG, ResNet, and InceptionV3 have been utilized as the underlying model structure, aiming at exploring the performance to identify types of leaves diseases from healthy leaves of single or multiple crops such as tomato, pear, cherry, peach, and grapevine [8–15]. The effectiveness of a CNN model in identifyng leaves diseases on single or multiple crops depends mainly on the quantity and quality images. The main benefit of working with multiple crops is to enrich image feature description learning but presents drawbacks when they have to identify diseases on early stages [16]. Several researchers have explored structural changes of some standard CNN models. In [17], LeNet CNN architecture

has been modified to identify three maize leaf diseases from healthy leaves. Based on ResNet50 CNN architecture, three different architectures were proposed in [16], to create a single multi-crop CNN model. The proposed architecture is capable of integrating contextual meta-data of the plant species information to identify seventeen diseases of five crops: wheat, barley, corn, rice and rape-seed. For experimental tests, the proposed multi-crop CNN model used a dataset constructed with more than one hundred-thousand of images taken by cellphone in real fields.

In addition, some optimization studies have been introduced aiming at the problems of too many parameters, computation and training time consumed by CNN models. For instance, a global pooling dilated CNN is implemented in [18] to address the problems of too many parameters of AlexNet CNN to recognize six common cucumber leaf diseases. The impact of the size and variety of the datasets on the effectiveness of CNN models is studied in [19]. The study discussed the use of transfer learning on GoogleNet to deal with the problem of leaf diseases recognition based on an image database of 12 plant species including rice, corn, potato, tomato, olive, and apple. An empirical exploration of fine-tuning and transfer learning on VGG 16, Inception V4, ResNet (with 50, 101 and 152 layers) and DensNet (with 121 layers) is addressed in [20]. Results obtained shown that DenseNet improves constantly its accuracy with increasing number of epochs, with no signs of overtting and performance degradation.

On the other hand, several specialized CNN architectures have been designed with a reduced number of hidden layers to provide a fast and accurate leaves diseases identification, such as [21], which used only two hidden layers to differentiate healthy and unhealthy images of rice leave. In [22], a novel architecture based on three-channel CNN with four hidden convolutional layers is presented. It is proposed for identification of vegetable leaf disease by combining the color components of diseases. Besides, a hybrid convolutional encoder network is stated in [23]. It identifies six classes of diseases on potato, tomato and maize crops.

3 Dataset

CNN demands huge amounts of varied data, otherwise the model may be not robust or overfitting. Therefore, the raw dataset was augmented performing preprocessing operations such as illumination changes mainly based on brightness and contrast variation, data augmentation, and the addition of random classes on their data. Figure 1 shows some examples from the original data set and the output after transformations performed over the raw data.

3.1 Raw Dataset

The current work used the Plant Village dataset [7], such a publicly accessible dataset that contains 13K RGB images of healthy and infected leaves of plants, with resolution 256×256. The dataset contains 26 different anomalies in 14 crop species including apple, blueberry, cherry, corn, grape, orange, peach, bell

pepper, potato, raspberry, soybean, squash, strawberry and tomato. In this work, all images of diseased leaves were labeled as unhealthy, resulting in a raw dataset with two classes (healthy and unhealthy leaves).

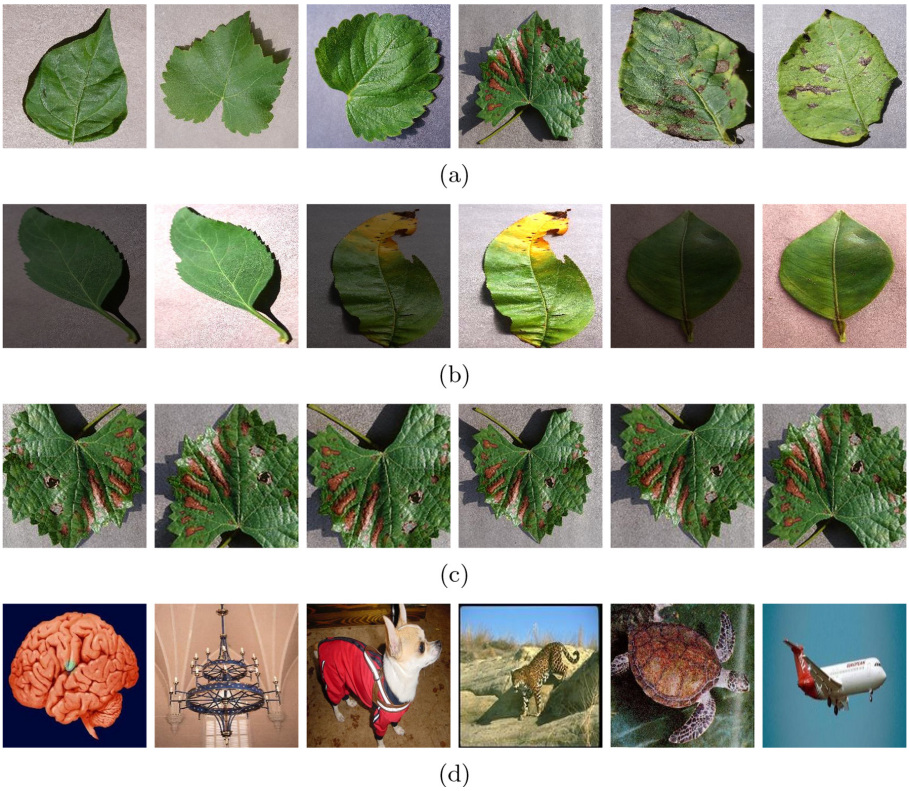

Fig. 1. Leaves samples related to: a) Raw dataset [7]; b) illumination changes (brightness and contrast variation), c) data augmentation transformations (rotation, translation, zoom and flip); d) random classes added.

3.2 Pre-processing Operations

A. **Illumination Changes (Brightness and Contrast Variation):** For each leaf image, this operation added two leaves images with high and low contrasts to the raw dataset, resulting in a total of 35,231 new leaves images. For this purpose, this work used **convertScaleAbs** function of openCV library, where $alpha = 1.5$ and $beta = 0$ parameters control the contrast and the brightness to create the new leaf image. The range of these values are [0.0–100.0] and [1.0–3.0], respectively.

B. **Data Augmentation:** It enriched the data with augmented images result of: rotating the image with $probability = 1$ from 5 to $-5°$, random distortions with $gridwidth = 4$, $gridheight = 4$ and $magnitude = 8$, left-right flips, top-bottom flips, random zooms with $probability = 0.5$ and percentage $area = 0.8$ (where probability corresponds how often the operation is applied and the magnitude is the magnitude of the distortions). They were performed by using the **augmentator library** [24]. It produced 9 K additional leaves images.
C. **Random Classes Added:** In order to distinguish healthy from diseased leaves, this procedure altered the raw dataset adding 6 classes including images of Airplane, Brain, Chandelier, Turtle, Jaguar, and Dog. This resulted in 1,238 additional images.

4 CNN Models

Inception V3 [3], GoogLeNet [4], ResNet 50 [6] and ResNet 101 [6] were selected because of their successful performance for image classification tasks [25,26]. On the other hand, ZFNet [5] was also explored due to their simplicity and low computation costs.

The architectures of explored CNN models are schematized in Fig. 2. Each one is characterized by a repeating sequence of convolutional layers, layers of activation functions, pooling layers, specialized modules (such as Inception and Residual modules aiming at efficient computations) and fully connected layers to output the classification label. Each layer type is named according to the performed operations. In this work, the last fully connected layer was adjusted to support two output classes.

– **Convolutional Layer:** It consists of a set of filters to extract different features from the input image. The main purpose of each filter is build a feature map to detect simple structures such as borders, lines, squares, etc. and grouping them into subsequent layers to construct more complex shapes. In mathematical terms, convolution computes dot products between the input data and the entries of the filters to learn and detect patterns from the previous layer, as is shown in Eq. 1.

$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(m,n)K(i-m, j-n) \tag{1}$$

where I represents the image and K the filter also often called kernel, m and n represent the dimensions of the filter typically of 3×3, 5×5, 7×7 or 11×11. *(i,j)* represents the pixels where the convolution operation will be performed. The input (image) is composed by $(i \times j \times r)$, where r is the number of channels of the image. The filters are slipped from the left up to the right down corner applying the convolutional operation to create the feature map.
– **Activation Functions:** In the convolution process many negative values are generated. These values are unuseful for the next layers and produce

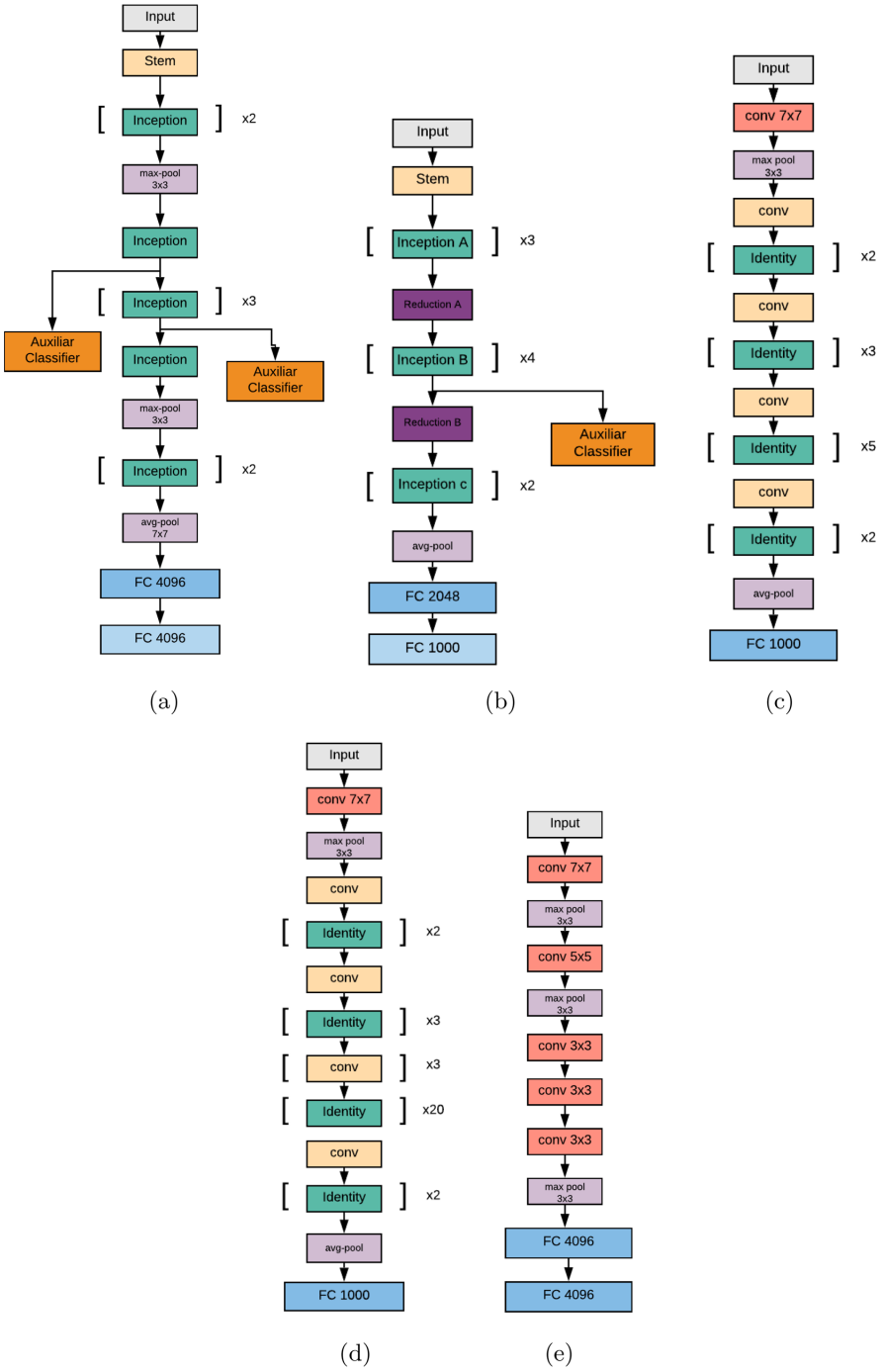

Fig. 2. State-of-the-art outstanding CNN architectures: a) GoogLeNet [4]; b) Inception V3 [3]; c) ResNet 50 [6]; d) ResNet 101 [6]; e) ZFNet [5].

more computational load. Therefore, after a convolutional layer, an activation function is applied. The activation function sets to zero negative values, which helps to get a faster and effective training. Rectified Linear Unit (ReLU) is the most widely used by Deep Learning techniques. ReLU aims to increase the non-linearity in the images, setting the value to 0 if the input is less than 0, and raw otherwise.

- **Pooling layer:** The pooling operation splits the convolved features into disjoint regions of a size given by a filter *F*. Then, the 'max' or 'mean' values are taken from each disjoint region to save main features and reduce progressively the parameters, and thus the computational load by sub sampling the spatial size of its input.
- **Fully Connected Layer (FC):** It aims to take the values from the previous layer and order them to drop out the class which the input belongs within a single vector of probabilities.
- **Inception Module:** It is used in deeper CNNs [4] for providing efficient computations and dimensional reduction. This module is designed with stacked 1×1 convolutions after the max-pooling layer. It is a different method to apply the convolution operation through different size filters to extract the most different and relevant parameters during the training. The different map features obtained are concatenated before the next layer. A significant change over this kind of modules was proposed by [3]. They applied a factorization operation over the inception modules by using large filters and dividing into two or more convolutions with smaller filters. Factorized inception module produced $\times 12$ lesser parameters than AlexNet [27] (one of the most common implemented architecture).
- **Residual Module:** In this module, each layer is the input for the next layer and directly feeds into the layers about 2–3 hops away. Residual Module characterizes the ResNet [6] CNN architecture aiming to face the vanish gradient [28]. The learning residual function is given by Eq. 2.

$$H(X) = F(X) - X \tag{2}$$

where X is the input of the block, *F(X)* is the residual function and *H(X)* is the final output of a residual block. This reformulation was done on the basis that a shadow network has less training error than the deeper counterpart. So, the residual modules support the deeper networks have an error which is no greater than the shallow ones.

The five explored CNNs architectures for the classification of leaves images are described in the following subsections.

4.1 ZFNet

It is an enhanced modification of AlexNet [27] architecture formed by eight layers. It is characterized by its simplicity. This has its convolutional layers followed by a max pooling layer. Each convolutional layer has decreasing filters starting

with a filter size of 7×7 as shown in Fig. 2e). ZFNet supports the hypothesis that, bigger filters loss more pixel information which can be conserved with the smaller ones to improve the accuracy. This model uses the ReLu activation function and the iterative optimization of batch stochastic gradient descent as learning algorithm for computing the cost function of a certain number of training examples for each mini-batch.

4.2 GoogLeNet - Inception V1

GoogLeNet [4] also called Inception V1 uses Inception Modules to build deeper networks with more efficient computation. It includes 1×1 Convolution at the middle and global average pooling at the end of the network that replaces a fully connected layers to save a lot of parameters and improve the accuracy. GoogleNet has 22 layers of deep, starting with a size filter of 7×7 decreasing through the network, followed by 9 inception modules, as illustrated in Fig. 2a).

4.3 GoogleNet - Inception V3

GoogleNet-Inception V3 [3] reflects an improved version of the previous version. In this architecture, computational efficiency and fewer parameters are realized. With 42 layers depth, the computation cost is only around 2.5 higher than GoogLeNet [4], and much more efficient than VGGNet [29]. Its most relevant features are: factorization technique applied trough the network and inceptions modules. The factorization is applied in order to reduce the number of parameters keeping the network efficiency. Inception V3 starts with small filters of 3×3 and keeps them until reach the inception modules, as shown in Fig. 2b).

4.4 ResNet 50 & ResNet 101

The Residual Networks, ResNets, represent a very deep network up to 152 layers. They learn from residual representation functions instead of learning the signal representation directly. ResNet 101 and ResNet 50 [6] included skip connection, also called shortcut connections, to fit two or more stacked layers into a desired residual mapping instead of hoping they directly fit. The skip connection is a solution to the degradation problem (accuracy saturation) generated in the convergence of deeper networks. They provide effectiveness in deeper networks, improving the learning process. The main difference between ResNet 101 and ResNet 50 is the number of convolutional layers, ResNet 101 has twice the number of layers than ResNet 50, as depicted in Fig. 2d) and c) respectively.

5 Experimental Setup

For experimental tests, each dataset was splitted into three sets with percentage ratio of (85%), (10%), and (5%) for training, validation, and testing respectively. The training set was processed to generate the LMDB dataset. LMDB is a compressed format for working with large dataset, where all images were sampled randomly from the training one. Explored CNNs received as input the generated LMDB. CNN models were trained using Python interfaces of Caffe framework available to face with research code and rapid prototyping. All experiments were performed using Tesla K80 Nvidia Graphics Processing Unit (GPU) that provides 11,441MiB of GPU memory, which allowed to use the original image resolution of 256×256.

In order to determine how the complex conditions on the dataset impact the performance of the CNN models applied to leaves diseases identification, this work train the explored CNN models with the original raw data, and with the training datasets augmented using three diverse operations (illumination changes, data augmentation and random classes added). In addition, the crucial hyperparameters such as learning rate (lr), step, and maximum number of iterations (#It.) were ne-tuned aiming at identifying a fast and accurate CNN model. Lr is the most critical hyperparameter to be tuned to achieve good performance. It controls how quickly or slowly a CNN model learns a problem, a value that is too small might produce a long training process that could get stuck, while a value that is too large might generate a sub-optimal set of weights, too fast or an unstable training. Step is a decaying lr function, which reduces the lr value over the total number of iterations at a given step size. A smaller step value results in more frequent reduction of the lr value over the total number of training iterations, which may result in slow training process due to small lr value. #It. determines at which iteration the highest accuracy might be achieved.

CNNs models differ in their hyperparameters, the default values of all hyperparameters are shown in Table 1. This work used Stochastic Gradient Descent (SGD) in the training process to compute the gradient descent of the cost function for each batch size.

Table 1. Default values of all hyperparameters.

Hyper parameter	Description	GoogLeNet	ZFNet	InceptionV3 - ResNet 50	ResNet101
Batch size	# of training samples in one forward/backward pass. Higher batch size needs more memory space	64	64	15	10
Test iter	# of test iterations per test interval. It must cover the validation set	100	200	100	100
lr policy	How the learning rate should change over the time	"poly"	"step"	"poly"	"poly"
Gamma	It sets how much the lr should change every time	0.96	0.1	0.96	0.96
Power	Helps to calculate the effective learning rate follows a polynomial decay in the "poly" lr policy	1.0	-	1.0	1.0
Momentum	How much of the previous weight will be retained in the new calculation	0.9	0.9	0.9	0.9
Weight_decay	The factor of (regularization) penalization of large weights	0.0002	0.0002	0.0002	0.0002
Solver mode	The mode used in solving the network	GPU	GPU	GPU	GPU
lr	It sets up the start learning rate in the network	0.001	0.01	0.001	0.001
Step	It sets how many iterations count before refresh the lr	320 K	200 K	320 K	320 K
#It	It indicates when the network should stop training	10 K	700 K	5 K	5 K

6 Experimental Results

Accuracy $((T_p + T_n)/(T_p + F_n + T_n + F_p))$, precision $(T_p/(T_p + F_p))$, recall $(T_p/(T_p + F_n))$ and F1 $((2 \times (recall \times precision)/(recall + precision))$ metrics were computed to measure the ability of explored CNNs models to classify leaves images into the corresponding class, and to determine how the different operations over the raw dataset and the hyperparameters fine-tune influence on the performance achieved. The parameters T_p, T_n, F_p, and F_n refer to the healthy leaf correctly classified as healthy one, unhealthy leaf correctly classified as unhealthy one, healthy leaf classified as unhealthy one, and unhealthy leaves classified as healthy one, respectively. In this sense, accuracy quantifies the ratio of correctly classified samples, precision quantifies the ratio of correctly classified positive samples over the total classified positive instances, recall measures

the ratio of correctly classified positive samples over the total number of samples, and F1 quantifies the harmonic meaning between accuracy and recall which shows how accurate and robust a classifier is.

6.1 Training with Raw Data

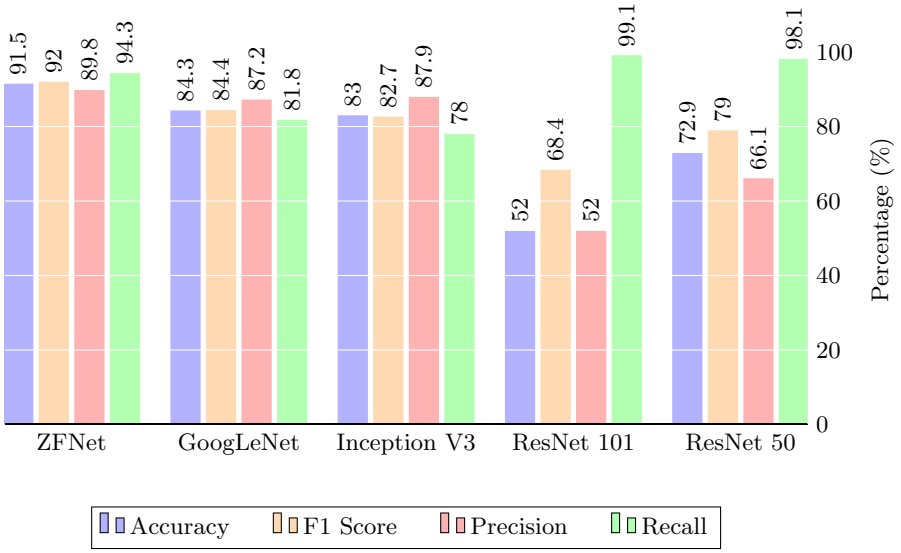

Fig. 3. Performance metrics values reached by the models with raw data.

Using the raw dataset with only two classes and minor variations among all samples, experimental results depicted in Fig. 3 clearly show that ZFnet takes advantages of the used default training settings parameters presented in Table 1. Indeed, ZFnet reaches the highest values (up 90%) overcoming all the explored models at the iteration 10K, it is attributed to its shallowness and simplicity. Experimental tests also demonstrate that deeper networks achieve lower values which can be improved only if more training iterations are performed with larger and more varied datasets. In this sense, ResNet 101 achieves the worst accuracy, precision, recall and F1 values. These results are due to the dataset characteristics which tend to cause overfitting problems in deeper models. In addition, ResNet 50 presents a better improvement in all the measures compared to ResNet 101, particularly because of the less layers and parameters managed by the model.

6.2 Pre-processing Impact

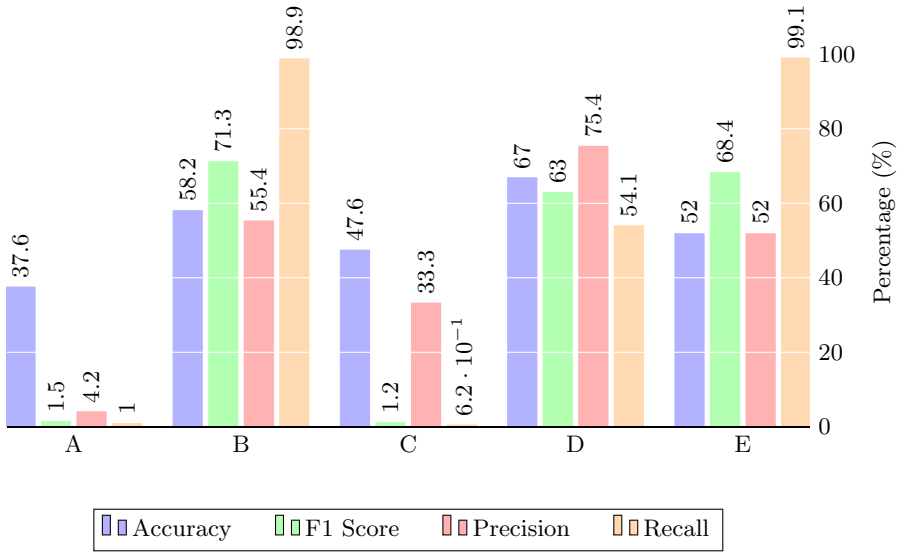

Fig. 4. Performance reached by the model ResNet 101 with the data sets: A) Illumination changes; B) data augmentation; C) random classes added; D) all changes; E) raw data.

As it is well know, the dataset size and poor variation data have a fundamental impact on the overall performance of deeper CNN. For this reason, ResNet101 was examined to determine the impact of different pre-processing operations to vary the leaves raw images and dataset size. Results reported in Fig. 4 show that both illumination variation and additional random classes datasets, experiments B and C, respectively, achieve lower accuracy than the raw dataset. In the illumination variation case, it is attributed to an overfitting produced by the repetition of the same image. Whereas with the random classes included, the accuracy performance decreases by 5%. This is produced because although the variation in the dataset size is increased, the difference between the two main classes (healthy and unhealthy leaves) is the same. Overall, augmented dataset reaches a notable increasing in accuracy due to the addition of new relevant information to the main classes, which avoids the overfitting produced in the previous cases. It can be affirmed because ResNet101 accuracy increases a significant percentage around 20% in experiment D as more data variability is applied.

Similar behaviour can be seen in the precision values. Precision decreases with illumination changes, increases a little with data augmentation and the eight additional classes, while in experiment D the precision with all the changes obtains the best performance. It is important to highlight that precision is a quite

relevant measure in this work, since it establishes the ability of the explored CNN model to identify diseased leaves. On the other hand, F1 and Recall scores reach the highest values in experiment B by using augmented dataset. In other words, applying the variations hinders the depth CNN model when it has to identify healthy leaves, but its performance improves when it has to identify diseased leaves.

6.3 Fine-Tuning Impact

Fine-tuning aims to increase the performance of the explored CNNs models by making small variations on their critical hyperparameters. Results obtained from training with the integrated dataset (raw dataset with all changes applied to the raw one) are shown in Table 2. From the scores obtained by ZFNet it is clear that shallow CNN models often leads to a higher amount of parameters. Besides, they can be efficient models with a limited number of classes, data variations and dataset sizes for the diseased leaves classification process. With few iterations in test 5, ZFNet achieves a high accuracy and increases it by 10% after 50 K iterations.

On the other hand, each explored CNN model has a particular behaviour for the evaluated hyperparameter values. For instance, GoogLeNet is highly susceptible to high lr and hence the values obtained from test 1 and 2 are lower than 50% in all metrics. The improvement reached in the rest of the tests is produced because of their default large batch size, which gives fast learning with an smaller lr. GoogLeNet also gives great precision reaching values up to 90% in test 5, which was trained with just 547 iterations. It marks an improvement of the model of 10% against the scores obtained with raw data. Inception V3 reaches its highest accuracy (77%) under conditions established in test 2. A relevant fact is presented in test 6, where the model presents an abrupt change from the tendency. The accuracy of test 6 with a small lr and several iterations (apparently the optimal conditions for Inception V3) dropped to its worst point (under 50%). It is presumed to a possible overfitting because of the large #It. and small lr. A highest precision score is obtained with a lr = 0.01 and a large number of iterations (around 40K). The recall and F1 scores retain the tendency to increase with high values, except in test 6 where all results were deficient.

ResNet 101 scores depend on the number of iterations and, to a lesser extent, on the lr value. This is demonstrated by the accuracy reached by the model, which does not change significantly unless the #It. is increased. This fact is attributed to the depth of the model. Its highest accuracy is 86%, which shows a significant increase from experiments with raw data. This is attributed to the new variation on the data set. The model also shows great performance in precision, recall and f1 score metrics. On the contrary the accuracy obtained from ResNet 50 was under 55% in all tests, a very low performance against the other models. This fact is mostly attributed to the dataset used to the different tests (called all changes) because in all the variations performed on the model, the accuracy obtained was poor.

Table 2. Fine-tuning results

Model	Amount of Parameters	Test #	lr	Step	#It.	Accuracy	Precision	Recall	F1
GoogLeNet	12.36 M	1	0.1	438	547	48%	50%	0.6%	1%
		2	0.1	40 K	50 K	48%	50%	0.6%	1%
		3	0.01	438	547	80%	77%	94%	84%
		4	0.01	40 K	50 K	**87%**	83%	**95%**	**89%**
		5	0.001	438	547	77%	**94%**	63%	75%
		6	0.001	40 K	50 K	81%	76%	96%	85%
Inception V3	21.8 M	1	0.1	1,750	2,188	55%	54%	**89%**	67%
		2	0.1	40 K	50 K	**77%**	78%	79%	**79%**
		3	0.01	1,750	2,188	71%	77%	64%	70%
		4	0.01	40 K	50 K	74%	**96%**	51%	67%
		5	0.001	1,750	2,188	73%	82%	62%	71%
		6	0.001	40 K	50 K	48%	50%	0.6%	1%
ResNet 50	23.51 M	1	0.1	1,750	2,188	32%	37%	**41%**	**39%**
		2	0.1	40 K	50 K	9%	3%	1%	2%
		3	0.01	1,750	2,188	24%	**53%**	25%	34%
		4	0.01	40 K	50 K	10%	6%	4%	5%
		5	0.001	1,750	2,188	32%	30%	12%	17%
		6	0.001	40 K	50 K	**52%**	3%	1%	2%
ResNet 101	42.51 M	1	0.1	2,800	3,500	44%	42%	21%	28%
		2	0.1	40 K	50 K	79%	**94%**	65%	77%
		3	0.01	2,800	3,500	48%	50%	0.6%	1%
		4	0.01	40 K	50 K	81%	**94%**	68%	78%
		5	0.001	2,800	3,500	52%	52%	**100%**	69%
		6	0.001	40 K	50 K	**86%**	84%	91%	**87%**
ZFNet	62.36 M	1	0.1	438	547	48%	50%	0.6%	1%
		2	0.1	40 K	50 K	48%	50%	0.6%	1%
		3	0.01	438	547	52%	52%	**99%**	69%
		4	0.01	40 K	50 K	52%	52%	**99%**	68%
		5	0.001	438	547	83%	83%	89%	86%
		6	0.001	40 K	50 K	**93%**	**91%**	97%	**94%**

7 Conclusion

In this work, an empirical analysis among five state-of-the-art CNNs was carried out. The explored CNN architectures include GoogLeNet, Inception V3, ResNet 50, ResNet 101 and ZFNet. They differ in the number of convolutional layers, number of parameters and in the way they are arranged. From the experiments, ZFNet has a growing trend in accuracy, precision and F1 scores with no over fitting or performance decreasing manifestations. ZFNet obtained a 93% validation accuracy score for 50 K iterations beating the rest of the explored CNN models for identification of diseased leaves using datasets with a small number of classes, data variations and dataset sizes. Based on obtained results, as it was expected, all CNN models requiere a careful fine-tuning of their hyperparameters because

they not show direct correlations among hyperparameter values and their depth. As future work, we propose to extend this work exploring more preprocessing techniques to improve the quality and quantity of a limited dataset an thus to accomplish a reliable classification for several leaves diseases.

Acknowledgement. This work used the supercomputer 'Quinde I' from the public company Siembra EP in the Republic of Ecuador.

References

1. Guachi Guachi, L., Guachi, R., Bini, F., Marinozzi, F.: Automatic colorectal segmentation with convolutional neural network. Comput. Aided Des. Appl. **16**, 836–845 (2019). https://doi.org/10.14733/cadaps.2019.836-845
2. Guachi Guachi, L., Guachi, R., Perri, S., Corsonello, P., Fabiano, B., Marinozzi, F.: Automatic microstructural classification with convolutional neural network, pp. 170–181, January 2019. https://doi.org/10.1007/978-3-030-02828-213
3. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. CoRR abs/1512.00567 https://arxiv.org/abs/1512.00567
4. Szegedy, C., et al.: Going deeper with convolutions. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1–9, June 2015. https://doi.org/10.1109/CVPR.2015.7298594
5. Zeiler, M., Fergus, R.: Visualizing and understanding convolutional networks. CoRR abs/1311.2901 (2013) http://arxiv.org/abs/1311.2901
6. Kaiming, H., Xiangyu, Z., Shaoqing, R., Jian, S.: Deep residual learning for image recognition. CoRR abs/1512.03385 (2015) http://arxiv.org/abs/1512.03385
7. Hughes, D., Salathe, M.: An open access repository of images on plant health to enable the development of mobile disease diagnostics through machine learning and crowdsourcing, November 2015
8. Sladojevic, S., et al.: Deep neural networks based recognition of plant diseases by leaf image classification. Comput. Intell. Neurosci. **2016**, 11 (2016). https://www.hindawi.com/journals/cin/2016/3289801/
9. Ballester, P., Correa, U., Birck, M., Araujo, R.: Assessing the performance of convolutional neural networks on classifying disorders in apple tree leaves, pp. 31–38, November 2017. https://doi.org/10.1007/978-3-319-71011-23
10. Bin, L., Zhang, Y., He, D., Li, Y.: Identification of apple leaf diseases based on deep convolutional neural networks. Symmetry **10**, 11 (2017). https://doi.org/10.3390/sym10010011
11. Zhang, K., Wu, Q., Liu, A., Meng, X.: Can deep learning identify tomato leaf disease? Adv. Multimedia **2018**, 1–10 (2018). https://doi.org/10.1155/2018/6710865
12. Baranwal, S., Khandelwal, S., Arora, A.: Deep learning convolutional neural network for apple leaves disease detection. SSRN Electron. J. (2019). https://doi.org/10.2139/ssrn.3351641
13. Ferentinos, K.: Deep learning models for plant disease detection and diagnosis. Comput. Electron. Agric. **145**, 311–318 (2018). https://doi.org/10.1016/j.compag.2018.01.009
14. Toda, Y., Okura, F.: How convolutional neural networks diagnose plant disease. Plant Phenomics 2019, March 2019. https://doi.org/10.1155/2019/9237136

15. Ashqar, B., Abu-Naser, S.: Image-based tomato leaves diseases detection using deep learning. Int. J. Eng. Res. **2**, 10–16 (2019)
16. Picon, A., Seitz, M., Alvarez-Gila, A., Mohnke, P., Ortiz Barredo, A., Echazarra, J.: Crop conditional convolutional neural networks for massive multi-crop plant disease classification over cell phone acquired images taken on real field conditions. Comput. Electron. Agric. **167**, 105093 (2019). https://doi.org/10.1016/j.compag.2019.105093
17. Ahila Priyadharshini, R., Arivazhagan, S., Arun, M., Mirnalini, A.: Maize leaf disease classification using deep convolutional neural networks. Neural Comput. Appl. **31**(12), 8887–8895 (2019). https://doi.org/10.1007/s00521-019-04228-3
18. Zhang, S., Zhang, S., Zhang, C., Wang, X., Shi, Y.: Cucumber leaf disease identification with global pooling dilated convolutional neural network. Comput. Electron. Agric. **162**, 422–430 (2019). https://doi.org/10.1016/j.compag.2019.03.012
19. Barbedo, J.: Impact of dataset size and variety on the effectiveness of deep learning and transfer learning for plant disease classification. Comput. Electron. Agric. **153**, 46–53 (2018). https://doi.org/10.1016/j.compag.2018.08.013
20. Too, E., Yujian, L., Njuki, S., Yingchun, L.: A comparative study of fine-tuning deep learning models for plant disease identification. Comput. Electron. Agric., March 2018. https://doi.org/10.1016/j.compag.2018.03.032
21. Bhattacharya, S., Mukherjee, A., Phadikar, S.: A Deep Learning Approach for the Classification of Rice Leaf Diseases, pp. 61–69, January 2020. https://doi.org/10.1007/978-981-15-2021-18
22. Zhang, S., Huang, W., Zhang, C.: Three-channel convolutional neural networks for vegetable leaf disease recognition. Cogn. Syst. Res. **53**, May 2018. https://doi.org/10.1016/j.cogsys.2018.04.006
23. Khamparia, A., Saini, G., Gupta, D., Khanna, A., Tiwari, S., de Albuquerque, V.H.C.: Seasonal crops disease prediction and classification using deep convolutional encoder network. Circuits Syst. Sig. Process. **39**(2), 818–836 (2019). https://doi.org/10.1007/s00034-019-01041-0
24. Augmentor. https://augmentor.readthedocs.io/en/master/index.html. Accessed 15 Mar 2020
25. Large Scale Visual Recognition Challenge (ILSVRC). http://www.image-net.org/challenges/LSVRC/. Accessed 15 Mar 2020
26. Canziani, A., Paszke, A., Culurciello, E.: An analysis of deep neural network models for practical applications, March 2016
27. Krizhevsky, A., Sutskever, I., Hinton, G.: Imagenet classification with deep convolutional neural networks, pp. 1097–1105 (2012). http://dl.acm.org/citation.cfm?id=2999134.2999257
28. Huang, F., Ash, J., Langford, J., Schapire, R.: Learning deep resnet blocks sequentially using boosting theory, June 2017
29. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv:1409.1556, September 2014