Agrociencia

IDENTIFICATION OF DISEASE IN TOMATO LEAVES USING MACHINE LEARNING CLASSIFIERS AND DIGITAL IMAGES

Juan Pablo **Ambrosio-Ambrosio**[1], Juan Manuel **González-Camacho**[1*], Abraham **Rojano-Aguilar**[2], David Hebert **del Valle-Paniagua**[1]

[1] Colegio de Postgraduados Campus Montecillo. Posgrado en Socieconomía, Estadística e Informática-Cómputo Aplicado. Carretera México-Texcoco km 36.5, Montecillo, Texcoco, State of Mexico, Mexico. C. P. 56264.
[2] Universidad Autónoma Chapingo. Posgrado en Ingeniería Agrícola y Uso Integral del Agua. Carretera México-Texcoco km 38.5, Chapingo, Texcoco, State of Mexico, Mexico. C.P. 56227.
* Author for correspondence: jmgc@colpos.mx

Citation: Ambrosio-Ambrosio JP, González-Camacho JM, Rojano-Aguilar A, del Valle-Paniagua DH. 2023. Identification of disease in tomato leaves using machine learning classifiers and digital images. **Agrociencia** https://doi.org/ 10.47163/agrociencia.v57i3.2462

Editor in Chief:
Dr. Fernando C. Gómez Merino

Received: November 12, 2021.
Approved: February 28, 2023.
Published in Agrociencia: May 11, 2023.

ABSTRACT

Early identification of diseases in crops improves agronomic decision-making and has a positive impact on agricultural production. In this study, we evaluated three machine learning classifiers to identify three diseases in a tomato crop (*Solanum lycopersicum*) using chromatic characteristics of digital images of leaves, and a computational tool was developed for its practical use. The classifiers were support vector machine (SVM), multilayer perceptron (MLP), and histogram gradient boosting (HGB). The target classes were tomato yellow leaf curl virus (V), the fungus *Septoria lycopersici* (H), the acarid *Tetranychus urticae* (A), and healthy leaves (S). The images were preprocessed to eliminate anomalies and the selection algorithm by region was used to obtain pixels of representative color for each target class. The pixels were then transformed from RGB to the HSV color model to create the training database, which consisted of three-color characteristics (H, S and V) and the associated target class. The three classifiers achieved similar prediction performance. According to the Kruskal Wallis test, there were no significant differences (*p-value* = 0.5117). SVM obtained an overall accuracy (*Acc*) of 93.3 %, MLP obtained a value of 93.2 %, and HGB of 93.1 %. Moreover, in performance at the class level (diseases), SVM obtained a higher F1 = 96 % in identification of symptoms caused by *Septoria lycopersici* and a lower F1 = 90 % in identification of symptoms caused by *Tetranychus urticae*. The computational tool developed, IDENTO v1.0, facilitated identification of the three leaf diseases in tomato based on optimized classifiers and constitutes an option for promoting the use of artificial intelligence in agriculture.

Keywords: Artificial intelligence, multi-class classification, support vector machine, decision trees, neural networks, optimization.

INTRODUCTION

The growing world population demands more efficient agricultural production to guarantee food security. Tomato (*Solanum lycopersicum*) is one of the most cultivated plants in the world due to its consumption and economic importance. The increase in its demand has promoted an increase in production and cultivated area. In 2019 the

FAO (2021) reported a world production of 180 766 329 Mg on 5 030 545 ha. Mexico, at the close of the 2020 growing cycle reported a production of 3 249 186 Mg of tomatoes on an area of 44 814 ha (SIAP, 2021).

Tomato yield is affected by pests and diseases, and timely detection is important for taking preventative and corrective actions (Seminis, 2017). Padol and Yadav (2016) developed a disease classification system for grape leaves in which they applied the nearest neighbors algorithm to segment diseased regions in images, extract characteristics of color and texture, and implement a support vector machine with an overall accuracy of 88.9 %. Abdullah *et al*. (2007) applied a multilayer neural network to classify diseases in rubber tree leaves with two approaches: RGB dominant pixels (mean) and normalized data. These authors reported precision of 70 % and sensitivity of more than 80 %.

Saleem *et al*. (2019) presented a review of deep learning models to classify diseases in crops. Particularly, they reported that the models of convolutional neural networks have been applied in disease detection with overall accuracy of more than 95 %. Fuentes *et al*. (2017) proposed a detector of tomato leaf diseases in real time based on deep-learning meta-architectures. These authors were able to successfully recognize nine different categories of diseases and pests with an overall accuracy between 80 and 85 %.

The objective of this study was to evaluate the performance of three machine learning models (support vector machine, multilayer perceptron, and histogram gradient boosting) and develop a computational tool to facilitate identification of tomato leaf diseases, such as tomato yellow leaf curl virus (TYLCV), leaf spot *Septoria lycopersici*, and damage by the acarid *Tetranychus urticae*, and differentiate them from healthy leaves, using digital images. We assume that the paradigms of machine learning represent a viable, low-cost alternative for identification of plant diseases based on color.

MATERIALS AND METHODS

Data set

The database of images used in this study consisted of a sub-set of 80 digital images (20 per target class) reported by Hughes and Salathé (2015). These images of healthy and diseased leaves were captured in the laboratory under different conditions of illumination (https://www.kaggle.com/datasets/emmarex/plantdisease). Each image was saved in jpg graphic format with a size of 256 x 256 pixels and a resolution of 96 pixels per inch. Each pixel of the image was transformed to the RGB standard color model with three color channels: red (R), green (G) and blue (B), associated with one of the four target classes: V, H, A and S.

The V class represents leaves infected by TYLCV, whose symptomology is manifested by small, wrinkled leaves that are yellow between veins and have curled edges (Prasad *et al*., 2020). Class H represents leaves affected by *Septoria lycopersici*, a fungus that

causes small, dark, aqueous lesions that grow and become circular lesions with black or brown edges (Seminis, 2017). Class A represents leaves affected by *Tetranychus urticae*, an acarid that causes white dots, chlorosis and, in severe cases, necrosis and defoliation (Pérez-Hedo *et al*., 2018). Class S is that of healthy leaves (Figure 1).

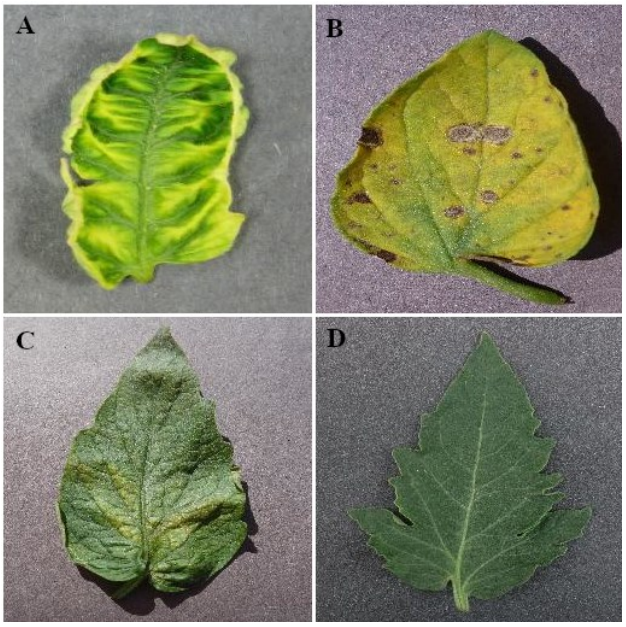

Figure 1. Images of tomato leaves with symptoms of disease. A: tomato yellow leaf curl virus; B. *Septoria lycopersici*; C: *Tetranychus urticae;* D: healthy leaf.

Computational tools

Images were processed and analyzed with the tools Opencv 4.5.1.48 and Scikit image 0.15.0. The machine learning algorithms were implemented in Python 3.7 language with the Scikit learn 0.21.3 library (Pedregosa *et al*., 2011) in the Spyder 3.6 development platform. For the vector and matrix operations, we used the tools provided by Pandas 0.25.1 and Numpy 1.16.5. with visualization of the outputs with Matplotlib 3.1.1. The Graphic interface was developed with Pyqt 5.9.2.

Images, data, and training of the learning models were processed with a computer system under Windows 10 environment of 64 bits, Intel Core i5 7th processor Gen @2.50 GHz, 500 SSD, 16 GB installed RAM memory.

Image processing

RGB images of the leaves were segmented by transformation to the HSV (*Hue*, *Saturation*, *Value*) color model to increase the visual difference between the leaf and the background. In HSV, H is the tone, which varies from 0 to 360 °, where each degree represents a color; S refers to colorimetric purity and varies from 0 to 100 % (maximum

color saturation); and V is the value or brightness and takes values from 0 % (black) to 100 % (white) (Camastra and Vinciarelli, 2015). The H, S, and V channels of each pixel of the image were then weighted with the following expression (Cuevas *et al*., 2010):

$$I_t = 0.2989 * H + 0.5870 * S + 0.1140 * V$$

where I_t is the weighted value of each pixel of the image.

With the set of I_t weighted values, the optimum threshold was determined with the OTSU algorithm to segment the leaf from the background of the image in binary form (1: leaf; 0: background) (Dey, 2020). A Gaussian filter with a 5 x 5 kernel and mean = 0 was applied to the segmented image of the leaf to homogenize it. Then, in images with damage on the edges, a morphological closure operation was applied to replace the 0 value pixels that formed part of the leaf with 1 value pixels. To eliminate noise in the image, the erosion operation was applied to 1 value replace pixels that did not form part of the leaf with value = 0. With the resulting binary image, the outer edge of the leaf was detected. With the image of the leaf profile, a mask with the values of white (255, 255, 255) was created to extract the pixels that form part of the original image in RGB format, and with the values of black (0, 0, 0) the leaf background (Cuevas *et al*., 2010). The images with problems of illumination and/or focus (salt and pepper noise), were smoothed with low pass filters such as the Gaussian, mean or median filters.

Extraction of characteristics

Color characteristics (obtaining samples of representative pixels of each target class) were extracted using the region growing algorithm (RG). This algorithm consists of selecting a seed pixel that represents the target class and determining by similarity the pixels that belong to a region defined by the seed pixel. RG compares each of the neighboring pixels (vicinity of four or eight pixels) with the seed pixel. If it complies to the similarity criterion, it is annexed to the region. To explore a new region of interest, another seed pixel is selected in the image and the search restarts. The algorithm terminates when adjacent pixels similar to the seed are not found, or when the entire image has been covered.

The binary image is used as a mask to recover the values of the three RGB channels of the original image. Dissimilarity (d) determines how different the pixels should be to exclude them from the selection. We used 30 images of leaves for each target class (V, H, A and S) with different seeds and values of d in the interval [0.03, 0.08] (Li *et al*., 2015). The set of RGB color pixels was cleansed of pixels that were duplicated within and between target classes to create a set of unique R, G, and B input pixels.

Samples of selected images

The tomato leaves were segmented from the image background (Figures 2A and 2B) using the HSV color model and an elliptical kernel to smooth the outline of the leaf. In the closure operation, a 5 x 5 kernel was applied with six successive repetitions, and

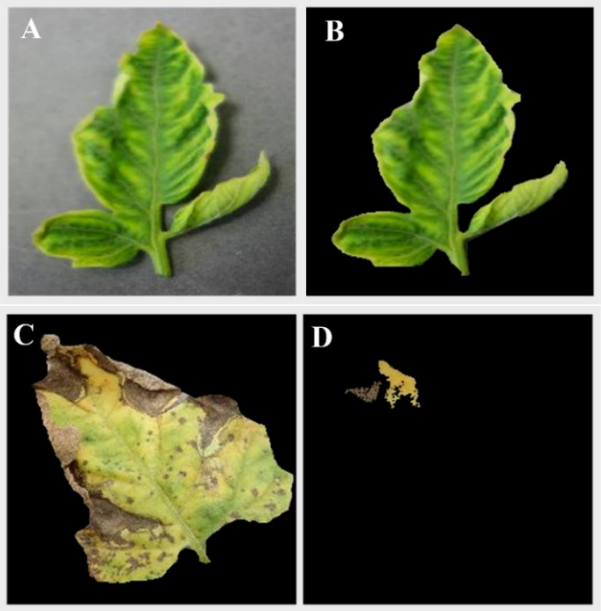

Figure 2. A: original tomato leaf; B: extraction of the object of interest, leaf infected by the tomato yellow leaf curl virus; C: leaf infected by *Septoria lycopersici*; D: Selection of samples, region growing algorithm with two seed pixels.

in the erosion operation, a 3 x 3 kernel was used with three successive repetitions. The images of the leaves with defined homogeneous edges were segmented automatically. However, the wrinkled leaves with shadows and noise were segmented manually to add and/or subtract small regions.

The selection of sample color pixels (R, G, B triplets) was obtained with the region growing algorithm and dissimilarity of 0.05 (Figure 2C) with two seed pixels (Figure 2D). In the input dataset, the value triplets (R, G, B) consisted of 40 000 samples (10 000 per target class), where each sample was associated with its target class (V, H, A, or S).

Machine learning models

Support vector machine

The support vector machine model (SVM) is a machine learning algorithm that is used to solve problems of classification or regression (Raschka and Mirjalili, 2017). SVM training consists of finding a hyperplane that separates the target classes in such a way that the margin between the support vectors is maximum. Smola and Schölkopf (2004) point out that, for the case of a bidimensional problem, the separation hyperplane is a line defined by:

$$f(x) = w^T x + b$$

where w is the vector of weights; x is the vector of input characteristics; and b is the bias. To find the maximum margin, one alternative is minimizing the norm of w, that is, $\|w\|^2 = w^T w$, and solving a problem of convex optimization. Slack variables ζ are introduced to smooth the linear restrictions and achieve convergence in linearly optimizing inseparable problems (Vapnik, 1995). With this approach, the loss function is expressed as (Géron, 2019):

$$L = \min_{w,b,\zeta} \frac{1}{2} \| w \|^2 + c \sum_{i=1}^{n} \zeta^{(i)}$$

$$Subject\ to \begin{cases} t^{(i)} \left(w^T x^{(i)} + b \right) \geq 1 - \zeta^{(i)} \\ \zeta^{(i)} \geq 0 \ for\ i = 1, 2, \ldots, n \end{cases}$$

where $y^{(i)}$ is the i^{th} predicted response; $x^{(i)}$ is a characteristic input vector; w is a vector of weights or parameters; b is the bias; $\zeta^{(i)}$ is a slack variable; n is the number of samples; $t^{(i)}$ is equal to -1 for negative samples (if $y^{(i)} = 0$) and 1 for positive samples (if $y^{(i)} = 1$) and C is the hyperparameter of balance between reducing the slack variables and maximizing the margin with a minimum w norm (Deisenroth *et al*., 2020).

Multilayer perceptron

The multilayer perceptron (MLP), or feed forward artificial neural network, is a generalization of the perceptron model proposed by Rosenblatt (1958). MLP architecture comprises three types of layers: the input layer (C_e), the hidden layer (C_o), and the output layer (C_s). The number of neurons in C_e is equal to the number of characteristics or input variables; the number of neurons in C_o is d (hyperparameter), and in C_s it is equal to the number of target classes (K). The neurons in each layer feed forward and are represented by a matrix of weights (W), with no connection between neurons of the same layer (Ramchoun *et al*., 2016).

The MLP classifier is trained iteratively for each sample of the subset of A training data. The matrix of weights W of MLP is initialized with random values between 0 and 1. The subset A passes through C_e, the output of the activation function enters at C_o, then the output enters at C_s. This process in matrix form is expressed by:

$$A^{(s)} = \varphi(\varphi(A^{(e)} W^{(0)}) W^{(s)})$$

where $A^{(e)}$ is a matrix of input samples (n x m) where n is the number of samples, and m is the number of characteristics or input variables; $W^{(0)}$ is a matrix of weights (m x d), where d is the number of neurons of the intermediate layer; $\varphi(\cdot)$ is an activation function, such as *relu*, *sigmoide*, *softplus*, *tanh* or *softmax* (Atienza, 2020); $W^{(s)}$ is a matrix

of output weights (d x K) where K is the number of target classes of network outputs; and $A^{(s)}$ is a matrix of probabilities (n x K) that represents the outputs of the MLP network (Raschka and Mirjalili, 2017).

The error was measured using a loss function (L) that compares the desired and predicted responses (Atienza, 2020). The process of back propagation of the error begins with the calculation of partial derivates using the chain rule, the contribution to the error is calculated for each connection, and the values are updated with the descending gradient algorithm. This optimizer minimizes the logistic loss function, or crossed entropy, L, which is expressed by:

$$L = -\sum_{K=1}^{K} T_K \log(y_k) \quad (1)$$

where K is the total number of target classes; T_k is the observed target class; and y_k is the predicted target class (probability of membership).

Histogram gradient boosting

The histogram gradient boosting classifier (HGB) is an ensemble method that uses multiple machine learning models. The HGB architecture uses decision trees as the nucleus and integrates a sequential additive model in which a decision tree is trained with the residual errors of its antecessor. In the end, a more robust and powerful HGB model is obtained (Géron, 2019).

The sequential approach makes model training slower when the dataset is large (tens of thousands of samples). Unlike the random forest model that uses independent decision trees, HGB uses groups of input characteristics by means of classes that represent intervals of whole numbers. Moreover, it uses histograms to divide the samples and decrease training time. The size of the decision trees can be controlled using hyperparameters: maximum number of leaves per node (mln), maximum depth (md), and minimum samples per leaf (msl); the loss function to be minimized is the logistic function (Equation 1) (Pedregosa *et al*., 2011). Friedman (2001) presented a detailed mathematical description of the histogram gradient boosting model and different alternatives of loss functions.

Classifier performance metrics

The metrics used to evaluate the performance of a classifier are deduced from a confusion matrix (CM) in which the rows represent the observed classes, and the columns represent the classes predicted by the classifier. A sample that is classified correctly as class 1 is denominated true positive (TP) a sample that is classified correctly as class 0 is denominated true negative (TN). A false negative (FN) occurs when a sample of class 1 is classified as class 0. A false positive (FP) occurs when a sample of class 0 is classified as class 1 (Raschka and Mirjalili, 2017). The performance metrics of the classifiers used in this study were the following (Powers, 2011):

The overall accuracy (Acc) is the proportion of correct classifications relative to the total number of samples and is calculated by:

$$Acc = \frac{TN + TP}{TN + TP + FN + FP}$$

Precision (P) evaluates the proportion of positive predictions, and measures the reliability of the prediction to classify the target class and is defined as:

$$P = \frac{TP}{TP + FP}$$

Sensitivity (S) measures the capacity of the classifier to detect positive samples correctly and is calculated by:

$$S = \frac{TP}{TP + FN}$$

The metric F1 score is the harmonic mean of P and S, and is calculated by:

$$F1 = \frac{2 * P * S}{P + S}$$

Given that P and S vary inversely, the trade-off between the two metrics depends on the objective of classification. The precision-sensitivity curve (P-S) enables identification of an optimum point to balance the two metrics; the area under the P-S curve ($AUC_{p\text{-}s}$) is a metric that considers the imbalance between classes.

The ROC (*receiver operating characteristic*) curve is a graph of S *versus* TFP, where TFP is the rate of false positives, the proportion of negatives classified as positive, TFP = 1-E, where E is the specificity, the proportion of negative samples classified as negative. The ROC curve is graphed with values of S *versus* 1-E for different thresholds of probability of membership to each target class. The area under the ROC curve (AUC_{ROC}) measures the performance of a classifier, a value near 1 is considered optimal (Hand and Till, 2001).

Classifier training and prediction performance

The training of classifiers SVM, MLP and HGB was carried out in two stages. The first consisted of selecting the optimal hyperparameters of each classifier. The second was evaluating the predictive capacity of the classifiers based on the optimal hyperparameters.

Grid search and selection of hyperparameters

Selection of optimum hyperparameters of the classifiers was based on a grid search and a cross-validation procedure (VC). The grid search consists of defining ranges of values for each hyperparameter of interest. For each combination of values, the classifier trains using VC with p random partitions. VC uses a random partition of the original dataset as a training set for which p random disjointed partitions stratified per target class are generated. For each combination of hyperparameters, the model trains p times using a partition for the test and the rest for training. After p iterations, the average Acc is obtained. At the end of the search, the combination of hyperparameters that maximizes the average Acc is selected. In this study, 80 % of the data were used for training, 20 % for testing, and VC with $p = 5$. The range of values of the grid search for each hyperparameter was defined experimentally by trial and error (Table 1) (Raschka, 2018).

Table 1. Value ranges of hyperparameter defined in the grid search for the classifiers support vector machine (SVM), multilayer perceptron (MLP) and histogram gradient boosting (HGB).

Classifier	Hyperparameter	Values
SVM	*C*	0.5, 1, 10, 25, 50
	kernel	linear, polynomial, rbf[†], sigmoid
	Gamma	0.1, 0.3, 0.5, 0.7, 0.9
	fd[¶]	ovo[§], ovr[þ]
MLP	*Hidden layer*	100, 120, 150
	fa[¤]	relu, logistic
	optimizer	adam, [††]sgd
	ta[¶¶]	constant, adaptative
	alfa	0.0001, 0.001, 0.01
HGB	*ta*[¶¶]	0.05, 0.1, 0.2
	mhn[§§]	25, 31, 50
	mp[þþ]	9, 10, 11
	mmh[¤¤]	15, 20, 25

[†]*rbf*: kernel Gaussian; [¶]*fd*: decision function; [§]*ovo*: one against one; [þ]*ovr*: one against the rest; [¤]*fa*: activation function; [††]*sgd*: stochastic gradient descent; [¶¶]*ta*: learning rate; [§§]*mhn*: maximum number of leaves per node; [þþ]*mp*: maximum depth; [¤¤]*mmh*: maximum number of samples per leaf.

Classifier prediction performance

The performance of each classifier was evaluated based on the entire dataset and the optimal hyperparameters. Training and test of each classifier were carried out with VC ($p = 5$) and the Acc metric. In each iteration, the performance metrics are obtained based on the test partitions. At the end of VC, the Acc, F1 macro averages and standard deviations were obtained. The three classifiers with optimal values of Acc were then used to obtain the metrics P, S and F1 for each target class.

Prediction and identification with new images

In this study, the computational tool denominated IDENTO was developed to identify diseases on tomato leaves in unseen images (not used in the training step) based on optimal learning classifiers. Moreover, IDENTO allows image processing to create new datasets for training and test. The tool is intended to facilitate the use of machine learning models and their practical application in recognizing the analyzed diseases (Ambrosio-Ambrosio and González-Camacho, 2022).

RESULTS AND DISCUSSION

Optimal hyperparameters

To select optimal hyperparameters, a training dataset with 128 000 samples (32 000 per target class) selected by a random sampling of the full dataset was used and stratified for each target class (Table 2).

Table 2. Optimal hyperparameters obtained by means of a grid search and cross-validation of the classifiers support vector machine (SVM), multilayer perceptron (MLP), and histogram gradient boosting (HGB).

Classifier	Hyperparameter	Optimum
SVM	*C*	50
	kernel	rbf[†]
	Gamma	10
	fd[¶]	ovr[§]
MLP	*Co*	150
	fa[Þ]	Relu
	optimizer	Adam
	ta[¤]	0.001
	alfa	0.0001
HGB	*ta*[¤]	0.1
	mhn[††]	31
	mp[¶¶]	11
	mmh[§§]	15

[†]*rbf*: Gaussian *kernel*; [¶]*fd*: decision function; [§]*ovr*: one *versus* the rest; [Þ]*fa*: activation function; [¤]*ta*: learning rate; [††]*mhn*: maximum number of leaves per node; [¶¶]*mp*: maximum depth; [§§]*mmh*: maximum number of samples per leaf.

Evaluation of classifier performance

The three classifiers achieved good performance F1 > 0.95 for classifying class H, the fungus *Septoria lycopersici* that causes brown spots with highly distinctive yellow outline. For class A, the acarid *Tetranychus urticae,* a value of F1 > 0.89 was obtained since it was not notably different from class S (Table 3)

Table 3. Performance of the classifiers support vector machine (SVM), multilayer perceptron (MLP) and histogram gradient boosting (HGB); metrics precision (P), sensitivity (S), score (F1) for each target class.

Model	Class	P	S	F1	AUC_{ROC}	$AUC_{P\text{-}S}$
SVM	V[†]	0.95	0.92	0.93	0.98	0.95
	H[¶]	0.95	0.96	0.96	0.99	0.97
	A[§]	0.89	0.90	0.90	0.97	0.91
	S[Þ]	0.94	0.95	0.94	0.99	0.96
MLP	V	0.95	0.91	0.93	0.99	0.98
	H	0.95	0.96	0.95	1.00	0.99
	A	0.89	0.90	0.90	0.98	0.95
	S	0.94	0.96	0.95	0.99	0.98
HGB	V	0.94	0.91	0.93	0.99	0.98
	H	0.95	0.96	0.95	1.00	0.99
	A	0.88	0.90	0.89	0.98	0.94
	S	0.94	0.95	0.94	0.99	0.98

[†]V: tomato yellow leaf curl virus; [¶]H: fungus *Septoria lycopersici*; [§]A: acarid *Tetranychus urticae*; [Þ]S: healthy leaf.

The three models had similar performance (Table 3), the class H, fungus *Septoria lycopersici*, was identified more efficiently and class A, acarid *Tetranychus urticae*, less efficiently. SVM obtained the highest F1 score, 96 %, and HGB had the lowest score, F1 = 89 %. At the class level, the performance metrics were very similar.

The SVM confusion matrix for a test set of 8000 random samples describes a total of 7459 pixels classified correctly with Acc = 93.2 % (Figure 3A). The ROC and P-S curves confirm that SVM had better performance in classifying class H than for class A (Figure 3B and 3C). The RF and HGB classifier confusion matrixes were very similar to SVM.

The final evaluation of performance of the classifiers SVM, MLP and HGB was carried out with a VC, $p = 5$ random partitions of 32 000 samples for training and 8000 for test prediction based on the metric Acc. Comparison of the three classifiers shows that, in terms of the median and the mean, Acc was more than 93 % (Figure 4A).

The classifiers SVM, MLP, and HGB had no problems of data overfitting. This occurs when the classifier achieves very high performance with the training set and low performance with the test, or prediction, set. The HGB classifier obtained very large differences in Acc between training and testing in the five random partitions of VC. However, these differences did not reflect an overfitting problem (Figure 4B).

SVM obtained Acc = 0.947 (±0.001) in training and in testing 0.934 (±0.004). MLP achieved Acc = 0.932 (±0.001) in training and 0.931 (±0.005) in testing; HGB reached a training Acc = 0.959 (±0.001) and a test Acc of 0.931 (±0.003). Based on overall precision, average Acc, the three classifiers obtained very similar prediction performance, *i.e.*, SVM (93.4 %), MLP (93.2 %) and HGB (93.1 %). The non-parametric Kruskal Wallis

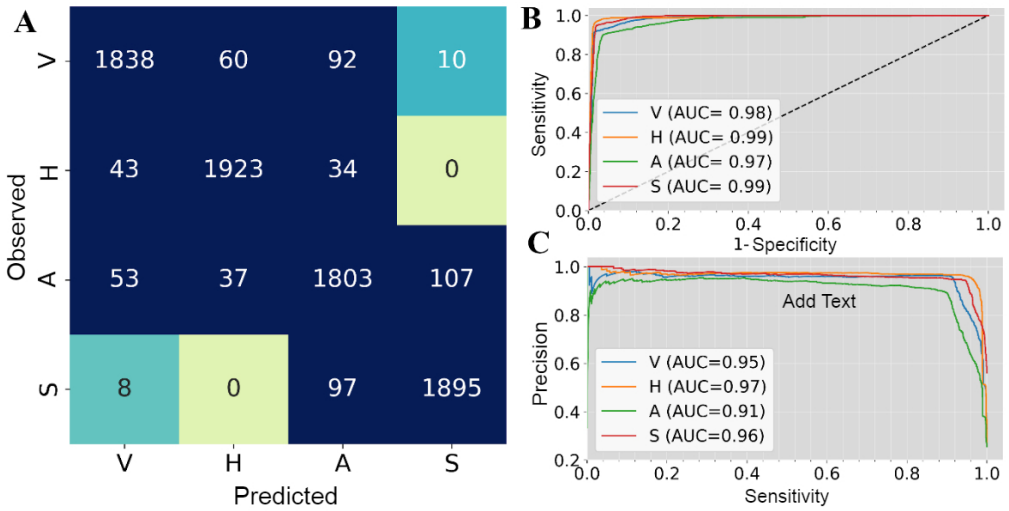

Figure 3. Support vector machine (SVM) performance. A: confusion matrix; B: ROC curves; C: P-S curves, precision *versus* sensitivity. Target classes: tomato yellow leaf curl virus (V), fungus *Septoria lycopersici* (H), acarid *Tetranychus urticae* (A), and healthy leaf (S).

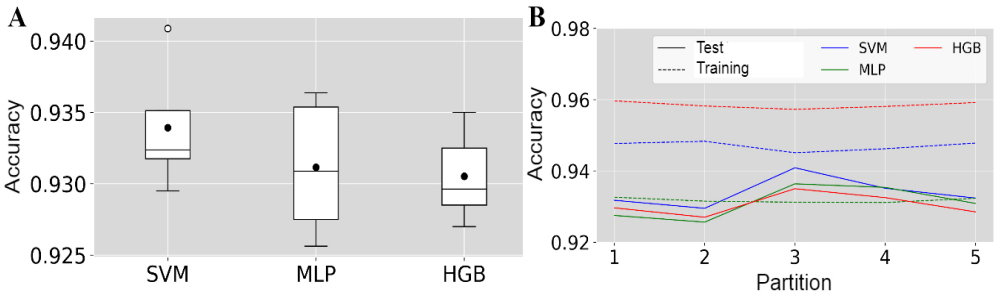

Figure 4. Comparison of overall accuracy (Acc) of the classifiers support vector machine (SVM), multilayer perceptron (MLP), and histogram gradient boosting (HGB). A. box plot of Acc; B. Graph of Acc in training and test for each partition k obtained by cross-validation.

test was applied to compare the response in terms of Acc performance of the three classifiers with five replications, and $\chi^2 = 1.34$, df = 2, p-*value* = 0.5117 was obtained and thus confirms that there were no significant differences in performance.

Regarding the performance achieved by the three classifiers, Acc was more than 93 % and was acceptable for identifying the diseases described and under the conditions of acquisition of the leaf images. In terms of computation time, the algorithms SVM and HGB are faster than MLP, and SVM was more stable in training and in testing (Figure 4B). SVM, MLP and HGB represent three different supervised machine learning paradigms. SVM is grounded in a process of quadratic optimization (Smola and Schölkop, 2004; Vapnik, 1995), MLP is a non-linear approximation model (Ramchoun *et al*., 2016), and HGB is based on a geometric approach of decision tree assembly

(Friedman, 2001). The relationship precision *versus* computation time of a machine learning algorithm, relative to deep learning paradigms is highly superior. However, to increase the predictive capacity of the learning algorithms even more, the use of deep learning is a viable option with a higher computational cost (Saleem *et al*., 2019; Padol and Yadav, 2016).

Disease prediction in new leaf images

The computational tool IDENTO v1.0 allows identification of diseases based on the optimal learning models used in this study. Identification comprises two stages. The first processes an RGB image to segment the leaf from the background (homogeneous). The second uses the segmented leaf to classify and identify the tomato disease. The classifiers SVM, MLP and HGB are activated from the Python platform and predict the disease.

The predictive capability of IDENTO v1.0 was demonstrated with an unseen tomato leaf image, not used in the classifier training or test stages, from the database consulted for this study. The image was 256 x 256 pixels with three color channels, R, G, B (Figure 5A). The three classifiers coincided in indicating *Septoria lycopersici* as the

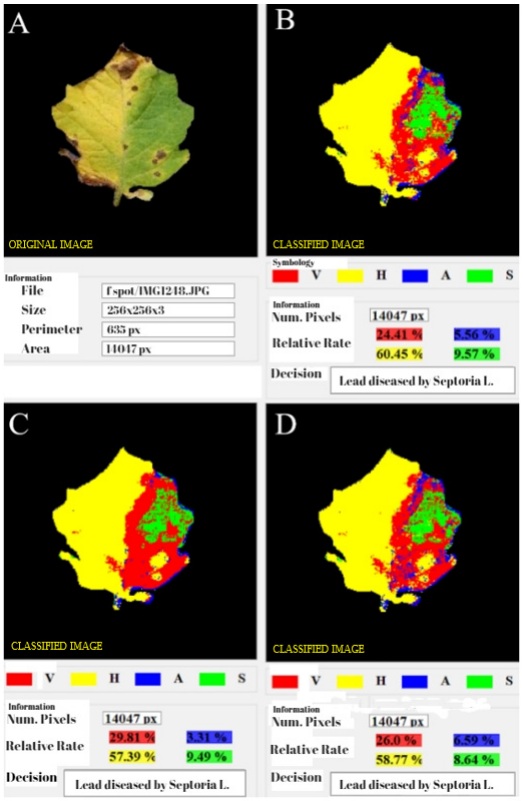

Figure 5. Identification of three diseases on tomato leaves, based on the classifiers support vector machine (SVM), multilayer perceptron (MLP), and histogram gradient boosting (HGB). A: preprocessed tomato leaf image; B: identification with SVM; C: identification with MLP; D: identification with HGB.

causal agent of the symptoms. The difference in Acc global precision of the models causes a change in the distribution of the pixels by class. Class H with SVM obtained the largest number of predicted pixels, 60.5 % (Figure 5B), and MLP had the lowest number, 57.4 % (Figure 5C). Given that the difference in the proportions of pixels is small, the visualized outputs are very similar.

The tool incorporates heuristic decision rules obtained by trial and error for identification of the most relevant disease, based on the relative rate of pixels (TR), that is, the proportion of pixels of a class of the total number of pixels of the leaf. If TR is higher than 70 % for class S, the leaf is identified as healthy. If TR is higher than 35 % for class V, it is identified as a leaf diseased by TYLCV, and if TR is more than 17 % for class H, it is identified as a leaf diseased by *Septoria lycopersici*, while if TR is more than 20 % for A, the disease is identified as caused by *Tetranychus urticae*. The thresholds of TR were defined in function of the analysis of multiple TR obtained when prediction was executed on the set of test images; overall precision in classification was more than 93 % (Kulkarni *et al.,* 2021**).**

CONCLUSIONS

The three classifiers, support vector machine (SVM), multilayer perceptron (MLP), and histogram gradient boosting (HGB) reached an overall accuracy (Acc) of more than 93 % in predicting the target classes: tomato yellow leaf curl virus (V), the fungus *Septoria lycopersici* (H), the acarid *Tetranychus urticae (A*), and healthy leaves (S). The fungus *Septoria lycopersici* was classified with a value of F1 = 96 %, healthy leaves with F1 = 95 %, tomato yellow leaf curl virus with F1 = 93 %, and the acarid *Tetranychus urticae* with F1 = 90 %.

The computational tool IDENTO v1.0 we developed enables practical application of the machine learning classifiers evaluated for identification of the diseases in tomato leaves based on the proposed heuristic rules. The user guide of the software is available at https://github.com/JPAAPSEICOA/Manual-IA-IMAGE-PROV1.0. This study shows the importance of applying machine learning paradigms using digital images for disease identification.

REFERENCES

Abdullah NE, Rahim AA, Hashim H, Kamal MM. 2007. Classification of rubber tree leaf diseases using multilayer perceptron neural network. *In* 2007 5th Student Conference on Research and Development. IEEE Xplore: Selangor, Malaysia, pp: 1–6. https://doi.org/10.1109/SCORED.2007.4451369

Ambrosio-Ambrosio JP, González-Camacho JM. (2022). IDENTO v1.0: una herramienta computacional para identificación automática de enfermedades en hojas de tomate. Colegio de Postgraduados. INDAUTOR, registro: 03-2022-061711283400-01.

Atienza R. 2020. Advanced deep learning with TensorFlow 2 and Keras (Second edition). Packt Publishing Ltd.: Birmingham, UK. 491 p.

Camastra F, Vinciarelli A. 2015. Machine learning for audio, image and video analysis (Second edition). Springer: London, UK. https://doi.org/10.1007/978-1-4471-6735-8

Cuevas E, Zaldívar D, Pérez-Cisneros M. 2010. Procesamiento digital de imágenes usando Matlab & Simulink. Alfaomega: Ciudad de México, México. 815 p.

Deisenroth MP, Faisal AA, Ong CS. 2020. Mathematics for machine learning. Cambridge University Press: Cambridge, UK. 407 p. https://mml-book.com (Recuperado: abril 2021).

Dey S. 2020. Python image processing cookbook. Packt Publishing Ltd.: Birmingham, UK. 438 p.

FAO (Organización de las Naciones Unidas para la Alimentación y la Agricultura). 2021. Production/yield quantities of tomatoes in world + (total) 1994–2019. Organización de las Naciones Unidas para la Alimentación y la Agricultura. Roma, Italia. http://www.fao.org/faostat/en/#data/QC/visualize (Recuperado: mayo 2018).

Friedman JH. 2001. Greedy function approximation: a gradient boosting machine. The Annals of Statistics 29 (5): 1189–1232.

Fuentes A, Yoon S, Kim SC, Park DS. 2017. A robust deep learning-based detector for real time tomato plant diseases and pests recognition. Sensors 17 (9): 1–21. https://doi.org/10.3390/s17092022

Géron A. 2019. Hands-on machine learning with Scikit-Learn, Keras & TensorFlow (Second edition). O' Reilly Media, Inc.: Sebastopol, CA, USA. 819 p.

Hand DJ, Till RJ. 2001. A simple generalisation of the area under the ROC curve for multiple class classification problems. Machine Learning 45: 171–186. https://doi.org/10.1023/A:1010920819831

Hughes DP, Salathé M. 2015. An open access repository of images on plant health to enable the development of mobile disease diagnostics. arXiv. https://doi.org/10.48550/arXiv.1511.08060

Kulkarni V, Gawali M, Kharat A. 2021. Key technology considerations in developing and deploying machine learning models in clinical radiology practice. JMIR Medical Informatics 9 (9): e28776. https://doi.org/10.2196/28776

Li W, Du Y, Li H, Wang X, Zhu J. 2015. Decision tree algorithm based on regional growth for the automatic oil field road. Proceedings of Science 6: 1–7.

Padol PB, Yadav AA. 2016. SVM classifier based grape leaf disease detection. *In* 2016 Conference on Advances in Signal Processing. IEEE Xplore: Pune, India, pp: 175–179. https://doi.org/10.1109/CASP.2016.7746160

Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V *et al*. 2011. Scikit-learn: machine learning in python. Journal of Machine Learning Research 12: 2825–2830.

Pérez-Hedo M, Arias-Sanguino AM, Urbaneja A. 2018. Induced tomato plant resistance against *Tetranychus urticae* triggered by the phytophagy of *Nesidiocoris tenuis*. Frontiers in Plant Science 9: 1–8. https://doi.org/10.3389/fpls.2018.01419

Powers DM. 2011. Evaluation: from precision, recall and f-measure to roc, informedness, markedness & correlation. Journal of Machine Learning Technologies 2 (1): 37–63.

Prasad A, Sharma N, Hari-Gowthem G, Muthamilarasan M, Prasad M. 2020. Tomato yellow leaf curl virus: impact, challenges, and management. Trends in Plant Science 25 (9): 897–911. https://doi.org/10.1016/j.tplants.2020.03.015

Ramchoun H, Janati MA, Ghanou Y, Ettaouil M. 2016. Multilayer perceptron: architecture optimization and training. International Journal of Interactive Multimedia and Artificial Intelligence 4 (1): 1–5. https://doi.org/10.9781/ijimai.2016.415

Raschka S, Mirjalili V. 2017. Python Machine Learning (Second edition). Packt Publishing Ltd.: Birmingham, UK. 850 p.

Raschka S. 2018. Model evaluation, model selection, and algorithm selection in machine learning. arXiv. https://doi.org/10.48550/arXiv.1811.12808

Rosenblatt F. 1958. The Perceptron: a theory of statistical separability in cognitive systems (Project Para). Cornell Aeronautical Laboratory: Washington, DC, USA. 268 p.

Saleem MH, Potgieter J, Arif KM. 2019. Plant disease detection and classification by deep learning. Plants 8 (11): 468. https://doi.org/10.3390/plants8110468

Seminis. 2017. Tomato disease field guide. Seminis Vegetable Seeds. Inc. De Ruiter: St. Louis, MO, USA, 168 p. https://issuu.com/sureshlm9/docs/tomato_disease_guide (Recuperado: mayo 2021).

SIAP (Servicio de Información Agroalimentaria y Pesquera). 2021. Anuario estadístico de la producción agrícola. Servicio de Información Agroalimentaria y Pesquera. Ciudad de México, México. https://nube.siap.gob.mx/cierreagricola/ (Recuperado: mayo 2021).

Smola AJ, Schölkopf B. 2004. A tutorial on support vector regression. Statistics and Computing 14: 199–222. https://doi.org/10.1023/B:STCO.0000035301.49549.88

Vapnik VN. 1995. The nature of statistical learning theory. Springer: New York, NY, USA. https://doi.org/10.1007/978-1-4757-2440-0