VOL. 16, NO. 13, JULY 2021 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences
©2006-2021 Asian Research Publishing Network (ARPN). All rights reserved.

www.arpnjournals.com

IDENTIFYING ROBELLINI PALM GROWTH STAGES THROUGH A CONVOLUTIONAL NEURONAL NETWORK

Fredys A. Simanca H., Jaime Alberto Paez, Jairo Cortés Méndez and Fabian Blanco Garrido
Systems Engineering, School of Engineering, Universidad Cooperativa de Colombia, Colombia
E-Mail: fredys.simanca@campusucc.edu.co

ABSTRACT

The Robellini palm or Pygmy date palm is a species native to Laos. Reaching a maximum height of 5 meters, it is considered one of the most exotic and elegant palms in the world. In fact, according to the Colombian Ministry of Agriculture and Rural Development, in the Colombian department of Cundinamarca, approximately 5000 hectares have been planted with flowers and foliage, including this palm. As its main market is the United States, this palm is grown at high levels of quality, procuring, and the correct leaf length. However, this can be cumbersome for the farmers owing to the height of the plants. In this study, we aim to build a convolutional neural network (CNN) model that can help Robellini Palm farmers identify current plant growth stages through aerial photographs taken with drones. Therefore, crop images were collected and classified into the four stages identified by the grower (seedbed, sowing, development and cutting) with 230 images used to train the model designed. After different tests, the accuracy of the Robellini Palm growth stage identification system was determined to be between 85% and 90%. Hence, the designed CNN satisfactorily classified the images loaded from the different crop stages.

Keywords: neural network, robellini palm, image classification, precision agriculture.

INTRODUCTION

This project seeks to create a technological tool using a convolutional neural network (CNN) [1] to identify the development stages of the Robellini palm [2], to foster agricultural progress by helping palm farmers identify the current growth stage of their crops. The solution proposed here is framed within the concept of precision agriculture (PA). Therefore, aerial photographs were taken using a drone. Then, these images were uploaded to the "Wit Farming" software, where the images are sorted by palm leaf stage using an algorithm created using a CNN.

The Robellini palm, also known as Pygmy date palm, is considered exotic and elegant palm trees in the world. These trees grow slowly and reach a maximum height of 5 m. The leaves are approximately 1-m-long, pinnate, with narrow and flexible greenish leaflets (20-cm-long) commonly arranged in a single plane with a rigid base structure, slender and sharp petioles, and turning dark after ripening. This palm is commonly used in small gardens, as well as an indoor plant as long as it is installed in well-lit places. It must be placed under direct sunlight or in partially shaded locations (Figure-1) [3].

Figure-1. Robellini palm.

This palm is categorized within the flowers and foliages, which includes plants with stems and leaves of different sizes and shapes, which are often used for flower arrangements. However, this category also includes several nonflowering plants. Colombian foliage is characterized by its large marketability among several high-demand countries because the properties of Colombian soils provide great advantages for these crops. Therefore, Colombian farmers can offer diverse genres and excellent production quality. Nevertheless, these crops are only produced for exporting, mainly to the United States because the domestic market is insignificant at the time.

Currently, the main issue faced by Robellini palms is the strict export quality requirements from the international market, that is, its leaflets to be cut at a minimum length. As this palm tree grows up to 5 m in height, farmers have problems monitoring leaflet development after the plant exceeds 2 m in height, thereby causing cut losses of up to 20%.

Therefore, a CNN project was developed, obtaining critical progress results because systematic geospatial aerial mechanisms (drones) [4] [5] [6] provide solutions to the different agricultural activities, facilitating the identification of crop issues through aerial images that, with the appropriate resolution, offer clear and accurate data on leaflet length or plant development stage or both. This allowed farmers to make quick decisions with a minimum margin of crop errors [7] [8].

The implementation of technological tools for these crops creates an important path in agricultural management by strengthening planting and irrigation, which accelerates the production development process and generates successful results at the right time for each crop development stage [9]. In fact, as production quality increases the farmers’ trust in technology increases, especially after having successfully implemented PA systems and, in our case, a CNN.

The above opens the way to continue advancing technologically in favor of the activity, estimating and better planning the processes of each crop is more feasible now, based on the main concept on which PA is based [10], which is to apply the correct amount of inputs, at the right time and in the right place. Hence, the application of information and communication technologies (ICT) for improving soil and crop management is effective [11]. For example, PA includes using geopositioning systems and advanced electronic devices, such as drones, to collect crop data, and perform irrigation and monitoring tasks [12] [13] [14].

For the Robellini palm, applying technological systems is essential because it is a foliage plant which can reach a maximum height of 5 m [15]. As this makes monitoring crops with the naked eye nearly impossible, this specialized invention generates great improvement opportunities for farmers. In the past, these monitoring activities were conducted manually, which generated great losses owing to inaccurate selection of cutting dates for the different batches. Therefore, using technology to collect data under a systematized procedure, such as the CNN, generates optimal results. [16].

Nevertheless, since this tool is optimized through image acquisition, miniature cameras must have an acceptable image resolution. In this sense, several image classification and processing methods have been created in many areas where image recognition is required, such as industrial fields, security activities, and the medical sector, among others. In addition, since the amount of information contained in a single image may prove too complex and vast for a linear algorithm, and even for an object-oriented algorithm, artificial intelligence must come into play, especially deep learning and neural networking methods based on the functions of biological structures and entities, such as the brain and human awareness [17] [18] [19].

Within this context, different artificial technology methods were assessed, determining that the most user-friendly option for the development of our application was the Python programming language. This structure features a built-in geoprocessing framework and an appropriate language for scripting, which facilitates the rapid development of the system structure and the creation of a simple and intuitive language that can be easily understood by the user [20]. This technological tool is expected to provide modern PA features, better leveraging the resources available and help farmers feel satisfied with the results obtained from its implementation, thus, becoming an essential farming solution. This generates improvements and progress in agriculture, as well as in the crop management and administration areas.

Furthermore, the implementation of crop technology is currently fostering crop development since it helps managing farming processes. Similarly, the information provided by technological equipment, such as PA, using drones to collect data and perform timely calculations that add value to treated crops, and, timely and localized crop support, help farmers streamline their agricultural processes, something that cannot be effectively developed manually [14].

Creating these technological structures is an important task since these structures are able to record a variety of objects in images through a significant number of “samples.” In addition, these images are processed through artificial neural networks, in which the “neurons” are part of associated tissues, thus bearing striking similarity to the visual neurons in a biological brain. Hence, they are extremely safe for artificial vision tasks, such as the programming and segmentation of pictorial representations, among other applications [1]. For example, when programmed with the corresponding features, they perceive a visual stimulus with the same orientation and position, they are able to be perfectly aligned with the patterns created, thus leaving the cell or artificial neuron activated and emitting the signal that has been programmed [21] [22] [23].

When implementing a method for the identification of the different Robellini palm development stages through a convolutional network, we detected several affectations in each palm growth process. Some of the most relevant were foliage damages caused by pests, humidity and pigmentation (based on the level of chlorophyll of the plant). However, we could successfully identify proper leaflet sizes for cutting, crop locations, the growth stages with the largest yield, current plant height, irregular size leaflets, temperature changes depending on seasonal variations, and other affectations that can occur during the development of the plants [24], which helped us determine the right time for cutting.

Our purpose here is to make this tool available to all Robellini palm farmers and help them offer high-yield products at high quality rates so they can gain an advantageous position in the market. Technology helps generate market competitiveness within the agricultural sector as it not only reduces costs, but also significantly increases productivity and quality and facilitates appropriate environmental crop management. This is especially relevant in Colombia since, owing to its tropical location, its soils are privileged for agricultural activities [7].

Nevertheless, crop management in Colombia still largely depends on traditional methods and there is little use of technology. In fact, Colombian crops are heavily affected by the climate change, especially by the high and low temperatures, without farmers having the technology available to protect their crops, providing the appropriate inputs and reducing their loss rates [25].

MATERIALS AND METHODS

The process used to identify the development stage of the Robellini palm using the CNN is described in Figure-2.

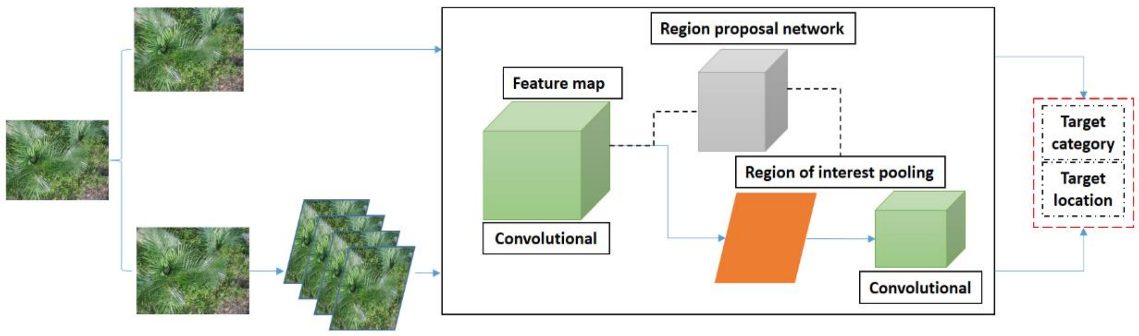


Figure-2. Designed convolutional neural network framework.

The convolutional network extracts the main characteristics from the images that best respond to the final objective. This convolutional activity uses an image as input and subsequently subjects this image to a filter or kernel, which generates a projection with the features of the original image, thus reducing the dimension of its measurements.

2.1 Study Design

The images used to train and test the neural network were obtained from a Robellini palm farmer. The categories specified by this farmer for this crop were: sowing, seedbed, development and cutting (Figure-3).

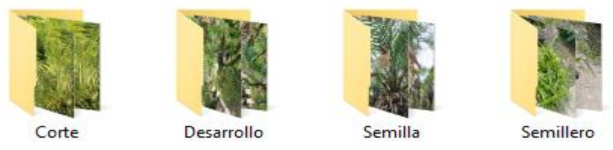


Figure-3. Training set.

2.2 Image Preparation

For each stage, the photographs provided by the farmer were compiled, obtaining the following number of images (Table-1).

Table-1. Number of images in the training set.

Stage	No. Images
Cutting	71
Development	93
Seed	40
Seedbed	26
TOTAL	230

2.3 Image Preprocessing

All images (230) were processed to a specific size (480 × 320). This is important because the programming code indicates the required image size. Hence, all images must have the same size. The format used was.jpg. In addition, all images were numbered sequentially although this is not a requirement for the application of the algorithm.

2.4 Image Processing

The CNN is implemented in Python mainly through libraries, such as TensorFlow and Keras, that establish in a simple way all the different layers and manners in which a model prediction can be obtained. Keras is an open-source library designed in Python based especially on the work by François Chollet, a Google programmer, in the ONEIROS (Open-ended Neuro-Electronic Intelligent Robot Operating System) project. The initial version of this systematic multiplatform application was launched on March 28, 2015, and it was aimed at accelerating the construction of neural networks. Hence, Keras does not work as a standalone framework, but as an automatic application programming interface (API) that facilitates access to different machine learning frameworks for their execution. The frameworks supported by Keras include Theano, Microsoft Cognitive Toolkit (formerly CNTK), and TensorFlow.

TensorFlow is a popular machine learning framework. Here the basic data structure is the tensor (a tensor is a generalization of vectors and matrices), and TensorFlow specializes in performing mathematical operations with tensors.

A machine learning application is the result of repeatedly calculating complex mathematical operations. In TensorFlow, these operations are represented using a graph known as "Data Flow Graph" where each node represents a mathematical operation, and the corners represent tensors to which these operations will be applied.

RESULTS

We created the *train.py* file, which was used to build the neural network. First, the *keras* library is imported to support the creation of the neural network. The scikit-Learn (sklearn) library is also imported, which helps to efficiently structure the data so that the algorithm may start sorting.

```
import os
import numpy as np
import matplotlib.pyplot as plt
from keras.utils.np_utils import to_categorical
from sklearn.model_selection import train_test_split
from keras.models import Sequential
from keras.callbacks import EarlyStopping, ModelCheckpoint
from keras.layers.convolutional import Conv2D, MaxPooling2D
from keras.layers import Flatten, Dense, Activation, Dropout
```

Two vectors were also created (*images*, *labels*). The former stores the images, and the latter labels each image that will be loaded from the directories provided by the *directories* variable.

```
images = []
labels = []

directories = [["rubelina/corte/," 0],
               ["rubelina/desarrollo/," 1],
               ["rubelina/semilla/," 2],
               ["rubelina/semillero/," 3]]

```

The *for* cycle goes through the four directories, adding each of the images to the *images* vector, and identifying the label of each image, adding them to the *labels* vector.

```
for i in range(len(directories)):
  folder = os.listdir(directories[i][0])
  for file in folder:
    if os.path.isfile(os.path.join(directories[i][0], file)) & file.endswith('.jpg'):
      image = plt.imread(directories[i][0]+file)
      images.append(image)
      labels.append(directories[i][1])

```

Then, we defined the *X* variable, which contains the *images* parameter, and the *Y* variable, which contains the *labels* parameter. Subsequently, the set of images is divided into 4, *X_train* and *Y_train* to train the network using 80% of the images (184 images); *X_test* and *Y_test* for the corresponding network testing using 20% of the images (46 images)

```
X = np.array(images)
Y = to_categorical(labels)
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2)

```

The *Sequential ()* function creates a 6-layer sequential model, where each layer contains exactly one input tensor and one output tensor. In addition, within the *for* cycle, the *add ()* method is used to incrementally build the sequential model. Finally, the model is trained through the *fit* function. This model is saved with the following name: *mod_rubeline.h5*.

```
cnn = Sequential()
for i in range(6):
        cnn.add(Conv2D(filters=32, kernel_size=(3,3),
                input_shape=(320,480,3), activation="relu,"
                padding="same"))
        cnn.add(MaxPooling2D(pool_size=(2,2)))

cnn.add(Flatten())
cnn.add(Dense(256))
cnn.add(Activation("relu"))
cnn.add(Dropout(0.5))
cnn.add(Dense(4))
cnn.add(Activation("softmax"))
cnn.compile(loss="categorical_crossentropy," optimizer="adam,
 " metrics=["accuracy"])

 Cal = [EarlyStopping(monitor="val_loss," patience=20), Model
 Checkpoint(filepath="mod_Rubelina.h5," monitor="val_loss," s
 ave_best_only=True)]
cnn.fit(X_train, Y_train, batch_size=128, epochs=20, callback
 s=Cal, validation_split=0.1, verbose=1)
```

Using the *verbose* explanation, we determined that instruction progress must be specified on the screen. For each *epoch*, the screen will display how much time has elapsed (useful data to assess the total time it takes to execute all epochs), as well as the values of four metrically defined for each epoch: *loss*, *accuracy*, *validation_loss* and *validation_accuracy* (See Table-2 below).

Table-2. Neural network training results.

Epochs	loss	accuracy	val_loss	val_accuracy
1/20	43.8030	0.2788	22.2032	0.4211
2/20	25.4097	0.2788	5.4115	0.4737
3/20	7.6221	0.3091	1.6131	0.6316
4/20	3.9002	0.4061	1.0149	0.5789
5/20	2.0111	0.4667	0.7145	0.7368
6/20	1.3937	0.4970	0.7272	0.6316
7/20	1.2230	0.5697	0.9067	0.7368
8/20	1.0159	0.6242	0.7081	0.7895
9/20	0.9296	0.6788	0.7504	0.7368
10/20	0.7486	0.7333	0.8682	0.7368
11/20	0.7250	0.7273	0.6745	0.7895
12/20	0.6831	0.7697	0.7080	0.7368
13/20	0.5974	0.7939	0.7471	0.7368
14/20	0.6206	0.7818	0.6032	0.7368
15/20	0.5537	0.8364	0.6105	0.7368
16/20	0.5718	0.7939	0.7295	0.7368
17/20	0.5534	0.7879	0.6917	0.7368
18/20	0.5575	0.7818	0.6170	0.7368
19/20	0.4949	0.8364	0.6566	0.7368
20/20	0.4472	0.8545	0.5686	0.7368

The *Loss* and *Accuracy* metrics are indicators of training progress. Through this mechanism, an attempt is made to predict the categorization of the training data to later conduct the measurement with the known label (cutting, development, sowing or seedbed), and measure the corresponding results. The *accuracy* value denotes the fractions from the correct calculations. This is how we can also verify that values are increased to their maximum during each *epoch*. Contrarily, the *validation_accuracy* metric, which is executed with the validation records, and

VOL. 16, NO. 13, JULY 2021 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences
©2006-2021 Asian Research Publishing Network (ARPN). All rights reserved.

which has not been included in this exercise, is lower, thus obtaining a threshold that does not exceed the expected values. The above is consistent with the results from over fitting through a number of epochs.

After training and generating the model, a new file is created to classify the input images. This file was named as *classify.py*.

```
1.  import os
2.  import numpy as np
3.  import matplotlib.pyplot as plt
4.  from keras.utils.np_utils import to_categorical
5.  from sklearn.model_selection import train_test_split
6.  from keras.models import load_model
7.
8.  images = []
9.  labels = []
10.
11. directories = [["rubelina/corte/," 0],
12.                ["rubelina/desarrollo/," 1],
13.                ["rubelina/semilla/," 2],
14.                ["rubelina/semillero/," 3]]
15.
```

The *Etiqueta* function quantifies the labels from 0 to 3 since we are only implementing four development stages for the Robellini palm.

```
16. def Etiqueta(stage):
17.         if stage==0:
18.                 return "Corte"
19.         elif stage==1:
20.                 return "Desarrollo"
21.         elif stage==2:
22.                 return "Semilla"
23.         elif stage==3:
24.                 return "Semillero"
25.
26. for i in range(len(directories)):
27.    folder = os.listdir(directories[i][0])
28.    for file in folder:
29.      if os.path.isfile(os.path.join(directories[i][0], file))
    &  file.endswith('.jpg'):
30.         image = plt.imread(directories[i][0]+file)
31.         images.append(image)
32.         labels.append(directories[i][1])
33.
34. X = np.array(images)
35. Y = to_categorical(labels)
36. X_train, X_test, Y_train, Y_test =  train_test_split(X, Y, te
    st_size=0.2)
37.
```

In line 38, the previously trained model is loaded (*mod_Rubelina.h5*) and assessed to determine its accuracy based on the variables previously established. In line 40, model accuracy is printed out.

```
38. model = load_model("mod_Rubelina.h5")
39. evaluation = model.evaluate(X_test, Y_test, verbose=1)
40. print("Precisión del test:," evaluation[1])
41.
```

Finally, the values for the test images are predicted. For testing, we used 20% of the 230 images (46 images), and the images for the 10 first data are displayed herein below. Tables 3 and 4 denote the results from two algorithm runs.

```
42. prediction = model.predict_classes(X_test)
43. for i in range(10):
44.    plt.imshow(X_test[i])
45.   plt.title(f"Etiqueta: {Etiqueta (np.argmax(Y_test[i]))}
   \nClasificación: {Etiqueta(prediction[i])}")
46.    plt.show()
```

VOL. 16, NO. 13, JULY 2021 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences
©2006-2021 Asian Research Publishing Network (ARPN). All rights reserved.

Table-3. First run results.

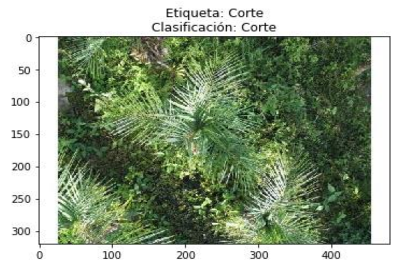
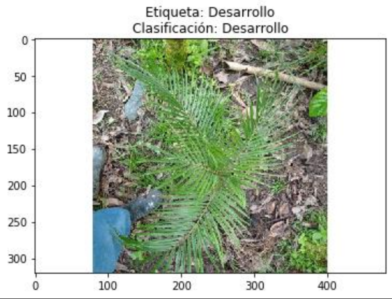
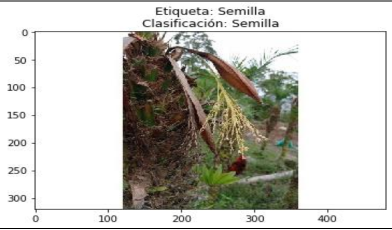
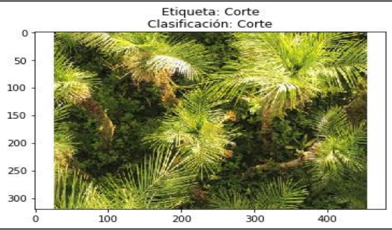
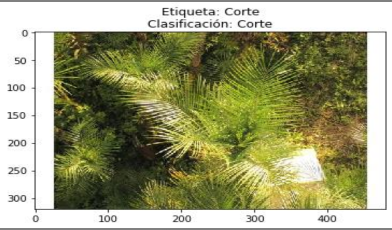
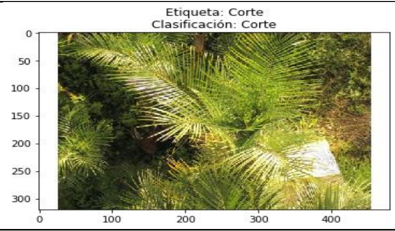
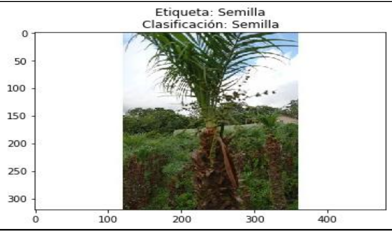
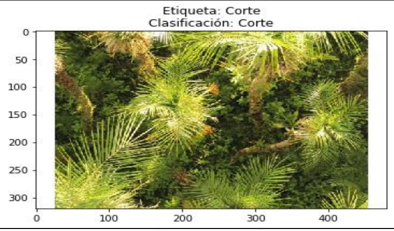
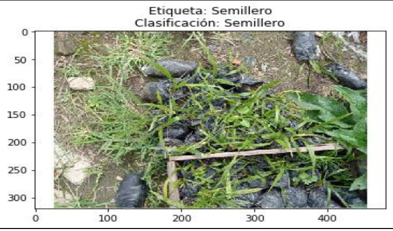
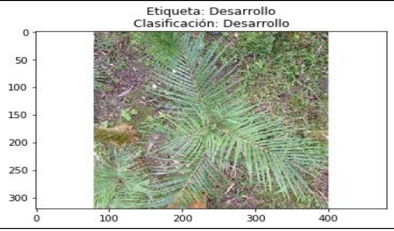
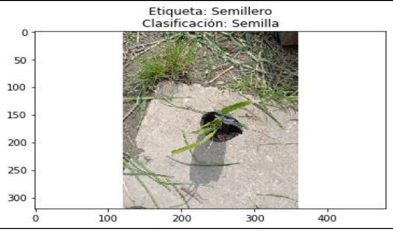
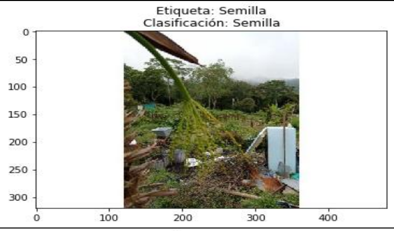
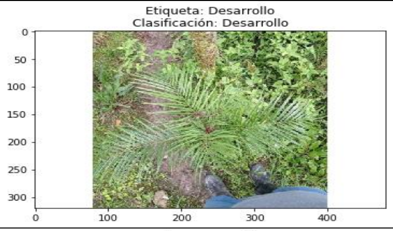
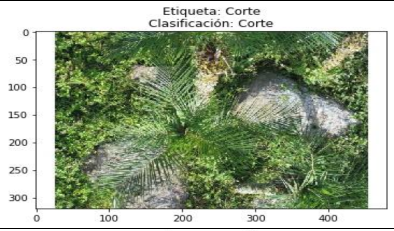
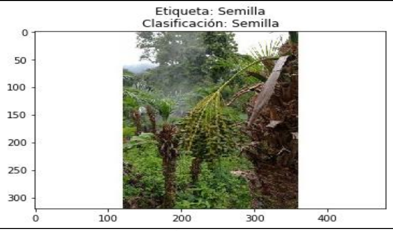
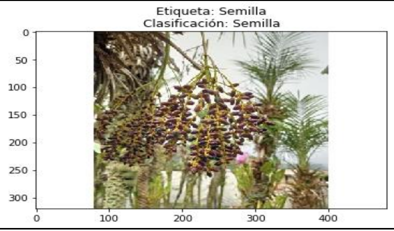
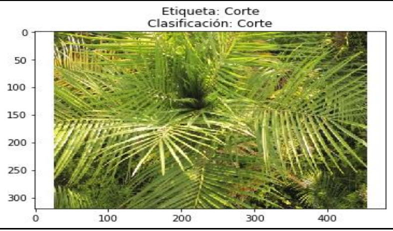
Test Accuracy:	**86.95%**
	
	
	
	
	

Table-4. Second run results.

Test Accuracy:	**89.13%**
	
	
	
	
	

CONCLUSIONS

The Robellini palm development stage identification system proposed here could efficiently sort the sample images loaded for classification. The different tests yielded an accuracy between 85% and 90%, which is good considering the number of images used to train the model. This algorithm will help Robellini palm farmers identify the cutting stage of the palm, thus reducing cutting losses.

This tool can also help farmers save time when going through crops to identify batches that are ready for cutting. In addition, compared with other traditional methods, this system is robust and exhibits low computational costs.

REFERENCES

[1] A. Krizhevsky, I. Sutskever y G. E. Hinton. 2017. «ImageNet classification with deep convolutional neural networks» Communications of the ACM. 60(6): 84-90.

[2] E. Iossi, R. Sader, F. V. Môro y J. C. Barbosa. 2007. «Phisiological maturation of Phoenix roebelenii O'Brien seeds» Revista Brasileira de Sementes. 29(1): 147-154.

[3] ambientessostenibles.com, «ambientessostenibles.com,» ambientessostenibles.com, [En línea]. Available: https://www.ambientessostenibles.com/producto/palma-robelina/#:~:text=Es%20una%20de%20las%20palmeras,Su%20crecimiento%20es%20lento. [Último acceso: 30 04 2021].

[4] A. L. A. R. Subashini Panchanathan. 2015. «MyGeo-Explorer: A Semantic Search Tool For Querying Geospatial Information» Arpn Journal of Engineering and Applied Sciences. 10(23): 18012.

[5] V. Berrío Meneses, D. F. Alzate Velásquez, J. A. Ramón Valencia y J. L. Ramón Valencia. 2018. «Optimization system of planning techniques in precision agriculture through drones» Espacios. 39(45): 18-34.

[6] V. A. Berrío Meneses, J. Mosquera Téllez y D. F. Alzate Velasquez. 2015. «Use of drones for multispectral image analysis in precision agriculture» @limentech. 13(1): 28-40.

[7] Artifice innovación. 2013. Cómo las TIC complementan la agricultura de precisión?, [How do ICTs supplement Precision Agriculture?]. 06 05. [Online]. Available: https://colombiadigital.net/opinion/columnistas/artifice-innovacion/item/4866-como-las-tic-complementan-la-agricultura-de-precision.html.

[8] W. F. Moreno, H. I. Tangarife y A. Escobar Díaz. 2017. «Image analysis aplications in precision agriculture» Visi´on Electr´onica. 11(2): 200-210.

[9] J. N. Pretty. 2008. «Agricultural sustainability: Concepts, principles and evidence,» Philosophical Transactions of the Royal Society B: Biological Sciences. 363(1491): 447-465.

[10] M. Dholu y K. A. Ghodinde. 2018. «Internet of Things (IoT) for Precision Agriculture Application» de Proceedings of the 2nd International Conference on Trends in Electronics and Informatics, Pune-India.

[11] Mintic. 2018. Plan TIC 2018-2022/ El futuro digital es de todos., [ICT Plan 2018-2022 / The Digital Future belongs to Everybody] Mintic, Bogotá D.C.

[12] DGIAR. 2015. Manual del cálculo de eficiencia para Sistemas de Riego [Irrigation System Efficiency Calculation Guidelines], Lima: DGIAR.

[13] C. Flores, E. Holzapfel y O. Lagos. 2010. Un Sistema de Soporte Dinámico de Decisión para la Gestión del Agua Predial en Riego Superficial: Desarrollo y Aplicación del Modelo [A Dynamic Decision Support System for Property Water Management in Surface Irrigation: Model Development and Application]. Chilean journal of agricultural research. 70(2): 278-286.

[14] A. N. Rao y L. Jagdish. 2012. «Economic weed management approaches for rice in Asia» ARPN Journal of Agricultural and Biological Science. 7(7): 508.

[15] Finagro. 2018. Agroguía [Agro-Guide]. Minagricultura, Bogotá D. C.

[16] G. A. M. A. F. S. Adrian Gonzales. 2015. Drones Aplicados a la Agricultura de Precisión [Drones applied to Precision Agriculture], Bogotá.

[17] K. Simonyan y A. P. Zisserman. 2015. «Very deep convolutional networks for large-scale image recognition,» de 3rd International-Conference on Learning Representations, San Diego - USA.

[18] A. Kamilaris y F. X. Prenafeta-Boldú. 2018. «Deep learning in agriculture: A survey,» Computers and Electronics in Agriculture. 147(1): 70-90.

[19] Y. Chen, Z. Lin, X. Zhao, G. Wang y Y. Gu. 2014. «Deep learning-based classification of hyperspectral data,» IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing. 7(6): 2094-107.

[20] resources.arcgis.com, «resources.arcgis.com» [En línea]. Available: https://resources.arcgis.com/es/communities/python/01r500000005000000.htm. [Último acceso: 03 05 2021].

[21] Y. Xu y M. Vaziri-Pashkam. 2021. «Limits to visual representational correspondence between convolutional neural networks and the human brain» Nature Communications. 12(1): 1-16.

[22] Health Big Data. 2021. «juanbarrios.com,» Health Big Data. [En línea]. Available: https://www.juanbarrios.com/redes-neurales-convolucionales/. [Último acceso: 29 04 2021].

[23] S. Kugunavar y C. J. Prabhakar. 2021. «Convolutional neural networks for the diagnosis and prognosis of the coronavirus disease pandemic» Visual Computing for Industry, Biomedicine, and Art. 4(1): 4-12.

[24] J. Morales. 2009. «Infojardin,» 25 03. [En línea]. Available: http://archivo.infojardin.com/tema/phoenix-roebelenii-danada-por-un-invierno-tan-lluvioso-y-frio-fotos.136051/. [Último acceso: 17 Enero 2019].

[25] J. Barros Sierra, «http://www.seminis.mx,» 5 Junio 2017. [En línea]. Available: http://www.seminis.mx/blog-como-afectan-las-altas-temperaturas-nuestros-cultivos/. [Último acceso: 30 05 2021].