agriculture

MDPI

Article

Evaluation of Convolutional Neural Networks' Hyperparameters with Transfer Learning to Determine Sorting of Ripe Medjool Dates

Blanca Dalila Pérez-Pérez [1], Juan Pablo García Vázquez [1,*] and Ricardo Salomón-Torres [2,*]

[1] Facultad de Ingeniería, Universidad Autónoma de Baja California (UABC), Mexicali 21100, Mexico; dalila.perez9@uabc.edu.mx
[2] Unidad Académica San Luis Río Colorado, Universidad Estatal de Sonora (UES), Sonora 83500, Mexico
* Correspondence: pablo.garcia@uabc.edu.mx (J.P.G.V.); ricardo.salomon@ues.mx (R.S.-T.); Tel.: +52-686-191-2431 (J.P.G.V.); +52-653-534-4255 (R.S.-T.)

Abstract: Convolutional neural networks (CNNs) have proven their efficiency in various applications in agriculture. In crops such as date, they have been mainly used in the identification and sorting of ripe fruits. The aim of this study was the performance evaluation of eight different CNNs, considering transfer learning for their training, as well as five hyperparameters. The CNN architectures evaluated were VGG-16, VGG-19, ResNet-50, ResNet-101, ResNet-152, AlexNet, Inception V3, and CNN from scratch. Likewise, the hyperparameters analyzed were the number of layers, the number of epochs, the batch size, optimizer, and learning rate. The accuracy and processing time were considered to determine the performance of CNN architectures, in the classification of mature dates' cultivar Medjool. The model obtained from VGG-19 architecture with a batch of 128 and Adam optimizer with a learning rate of 0.01 presented the best performance with an accuracy of 99.32%. We concluded that the VGG-19 model can be used to build computer vision systems that help producers improve their sorting process to detect the Tamar stage of a Medjool date.

Keywords: *Phoenix dactylifera* L.; Medjool dates; image classification; convolutional neural networks; deep learning; transfer learning

Citation: Pérez-Pérez, B.D.; García Vázquez, J.P.; Salomón-Torres, R. Evaluation of Convolutional Neural Networks' Hyperparameters with Transfer Learning to Determine Sorting of Ripe Medjool Dates. *Agriculture* **2021**, *11*, 115. https://doi.org/10.3390/agriculture11020115

Academic Editors: Sebastian Kujawa and Gniewko Niedbała
Received: 31 December 2020
Accepted: 25 January 2021
Published: 1 February 2021

1. Introduction

The date palm fruit (*Phoenix dactylifera* L.) is a berry composed of a fleshy mesocarp, covered by a thin epicarp and an endocarp covering all of its seed [1]. The name of this fruit is "date," which comes from the Greek word "Daktylos," which means "finger" [2]. This fruit has been the primary source of food in several countries in the Middle East, playing an essential role in the economy, society, and environment [3].

This fruit's growth presents a progressive maturity level in four stages known by their Arabic names: Kimri, Khalal, Rutab, and Tamar. At its first stage of growth (Kimri), the fruit is small, green, and with a hard texture. In its second stage (Khalal), the fruit reaches its maximum size and changes it is green color to yellow or red. In the third stage (Rutab), the fruit is losing weight and moisture, turning the fruit into a brown color. In the last stage (Tamar), the fruit is ripe and ready to be harvested [4].

According to Food and Agriculture Organization of the United Nations data, the world's largest date producers are Egypt, Saudi Arabia, Iran, Algeria, and Iraq, producing 66% of the world production in 2018 [5]. However, despite not being a native crop of the American continent, the date has also become a priority fruit for cultivation in southern California and Arizona in the United States and northwestern Mexico, where high-quality dates, such as Medjool cultivar, are grown [6].

The date palm producers face several challenges concerning harvesting, sorting, and packaging because they are mainly performed manually [7]. Therefore, many employers

are hiring for these activities that involve long working hours. People perform repetitive tasks, causing mistakes in the correct inspection of the fruit's quality attributes, such as color (maturity level), size, and texture.

Particularly in the Medjool date harvesting process, fruit pickers shake the palm bunch so that the ripe dates fall into containers. This can cause the ripe fruit to suffer damage to its texture or that fruits in other ripening stages are also harvested. The dates are placed in trays, where the immature ones will be extracted and grouped in other trays to dry them in the sun until they reach their full maturity. In contrast, the minute or damaged ones are commonly separated to develop date by-products or for animal consumption. Finally, the Medjool date sorting (which has the required degree of maturity) is packed.

To automate the processes related to harvesting, sorting, and packaging of dates, recently, there has been interest in exploring convolutional neural networks (CNNs) [8,9]. CNN has shown exceptional accuracy for classifying fruits and vegetables, considering several quality parameters, such as color (maturity level), shape, texture, and size [10–14].

Regarding dates, we identified that some studies use machine learning algorithms and image processing techniques to sort among date palm fruit or to detect among their different maturity stages [15–17]. Further, there are research works that propose using CNNs [8,9,18]. However, these studies do not present models to detect the maturity stage of the Medjool date.

The main contribution of this article was the identification of the hyperparameters that best influenced the training of a CNN architecture that transfers learning to Medjool's mature date sorting. To achieve it, we performed a comparison of the performance of eight CNN architectures. Two versions of the CNN architecture are called the Visual Geometric Group (VGG) from Oxford University, VGG-16 and VGG-19. Three versions of the CNN architecture are called Residual Network from Microsoft research, ResNet-50, ResNet-101, and ResNet-152. WE also looked at AlexNet, Inception Version 3, and a CNN from scratch. The hyperparameters analyzed were the number of layers, the number of epochs, the batch size, optimizer, and learning rate.

2. Materials and Methods

2.1. Image Acquisition

The images corresponding to ripe and unripe dates in trays were taken in September 2020, during the first round of harvest of Medjool dates in the plantation located in Colonia La Herradura (32°36′56″ N, 115°15′36″ W) in the Mexicali Valley, Mexico. The acquisition of images was made with three different cameras, using natural light between 8:00 a.m. and 2:00 p.m. We used a Canon EOS Rebel T6 of 18 megapixels and the cameras of the smartphones Samsung, SM-N950F and SM-N960F, which have a dual camera of 12 megapixels.

2.2. Image Data Set

The image data set contained 1002 images in JPG format, which are of different sizes (5184 × 3456, 4449 × 3071, and 4376 × 3375 pixels). The network architectures were trained with JPG images because they are fed with low-quality images in real scenarios. We refer to low-quality as images with blur, noise, contrast, or compression. We considered that if you are trained in architecture with this type of image, the system will be able to classify the Medjool date in images with these features. Further, a study shows that convolutional neural networks are minimally affected in their performance by using JPG format [19].

The image data set was distributed as follows: 501 images of ripe dates and 501 images of unripe dates on trays (Figure 1). The dates in trays were previously classified as ripe or unripe by expert people.

Figure 1. Example of dataset images. (**A**–**C**) Trays with ripe dates. (**D**–**F**) Trays with unripe dates.

2.3. Convolutional Neural Networks

The convolutional neural network (CNN) is a type of artificial neural network, where neurons correspond to receptive fields similar to the neurons in the primary visual cortex of a biological brain [20]. Also, CNN is identical to ordinary neural networks such as multilayer perceptron. They are composed of neurons that have weights and biases that can learn. Each neuron receives some input, performs a scalar product, and then applies an activation function [21]. The CNN as a multilayer perceptron has a loss or cost function on the last layer, which will be fully connected.

Figure 2 presents a CNN structure, which consists of three blocks. The first is the input, an image. Next, we can see the block of feature extraction, which consists of convolutional and pooling layers. Finally, the third block is of classification, which consists of fully connected layers and softmax. The structure of the convolutional network changes as the number of convolution and pooling layers increases.

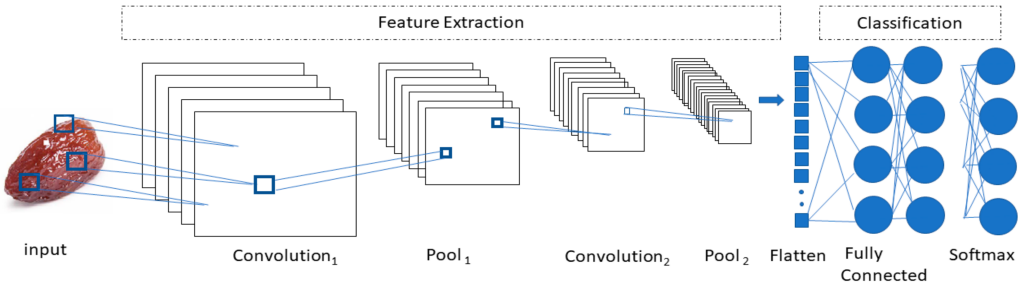

Figure 2. Representation of a basic convolutional neural network (CNN).

The main difference between convolutional neural networks from ordinary neural networks is that they explicitly assume that the inputs are images [21], allowing specific encoding properties in the architecture, gaining efficiency and reducing the number of

parameters in the network. In this way, CNNs can model complex variations and behaviors, giving quite accurate predictions. This study considered eight CNNs' architectures: VGG-16, VGG-19, Inception V3, ResNet-50, ResNet-101, ResNet-152, AlexNet, and one CNN from scratch.

2.3.1. VGG-16 and VGG-19 Architectures

VGG is the abbreviation for the Visual Geometric Group [22]. The VGG model was developed by Simonay and Zimmerman [23]. VGG uses 3 × 3 convolutional layers stacked on top of each other in increasing depth. The reduction of volume size is handled by max pooling. Two fully connected layers, each with 4096 nodes, are then followed by a softmax classifier [23]. The number 16 or 19 is the layer of networks considered deep.

2.3.2. Inception V3 Architecture (GoogLeNet V3)

This architecture was born with the name of GoogLeNet, but subsequent updates have been called Inception vN, where N refers to the version number put out by Google [22]. The basic module of Inception [24] consists of four branches concatenated in parallel: a 1 × 1 kernel convolution, followed by two 3 × 3 convolutions; a 1 × 1 convolution, followed by a 3 × 3 convolution; a pooling, followed by a 1 × 1 convolution; and finally a 1 × 1 convolution. Inception consists of 10 modules, although these modules are going slightly as the net gets deeper. Five of the modules are changed with the purpose of reducing the computational cost by replacing the n × n convolutions with two convolutions, a 1 × 7 followed by a 7 × 1. Two last modules replace the last two convolutions: 3 × 3 of the first branch with two convolutions each and one 1 × 3 followed by another 3 × 1, this time in parallel. In total, Inception V3 has 42 layers with parameters.

2.3.3. ResNet-50, ResNet-101, and ResNet-152 Architectures (Residual Neural Network)

ResNet [25] does not have a fixed depth and depends on the number of consecutive modules used. However, increasing the network's depth to obtain a greater precision makes the network more difficult to optimize. ResNet addresses this problem by adjusting a residual application in place of the original and adding several connections between layers. These new connections skip several layers and perform an identity or a 1 × 1 convolution. The base block of this network is called the residual block. When the network has 50 or more layers, it is composed of three sequential convolutions, a 1 × 1, a 3 × 3, and a 1 × 1, and a connection that links the input of the first convolution to the output of the third convolution. This study used three models with this architecture, ResNet-50, ResNet-101, and ResNet-152, which are composed of 50, 101, and 152 layers, respectively.

2.3.4. AlexNet

This architecture consists of five convolutional layers and three fully connected layers. Some convolution layers are followed by max-pooling layers (1, 2, and 5 layers). The Rectified Linear Unit (ReLU) nonlinearity is applied to the output of every convolutional and fully connected layer. The fully connected layers have 4096 neurons each [26]. To avoid data over-adjustment, a regularization method is used, known as a dropout, which consists of "turning off" neurons with a predetermined probability during training.

2.3.5. CNN from Scratch

The CNN that we built from scratch was composed of four alternate convolutional and max-pooling layers, followed by a dropout after every other convolutional and pooling pair. After the last grouping layer, we attached a fully connected layer with 256 neurons, another dropout layer, and, finally, a softmax classification layer for our classes. The loss function was the cross entropy since it is useful with convolutional neural networks, most significantly for purposes of image classification [27]. In order to compare the performance of a network that learns from scratch against other architectures that start from transfer learning, a convolutional network was trained from scratch.

2.3.6. CNNs' Optimization Techniques and Hyperparameters

Techniques

All the above networks were too deep to train them from scratch with our data set. Therefore, we used transfer learning, which consists of taking the features learned in other contexts and using them in a new and similar problem [28]. Transfer learning is usually done for tasks where the data set has too little data to train a full-scale model from scratch. This was our case since we only had 1002 Medjool date images.

Transfer learning is commonly used in two ways: (1) pretraining model, which consists of using a pretrained model that replace its last layers with others, so that the characteristics are of the new data set and (2) convolutional network tuning, which is a strategy to tune the weights of the layers using backward propagation.

For this study, the application of transfer learning was the pretraining model. We used the pre-trained networks with ImageNet, which is a large visual database designed for use in visual object recognition [26]. We removed the final classification layer, the neuron softmax layer at the end, which corresponds to ImageNet, and instead replaced it with a new softmax layer for our image data set. A summary of the utilized CNN architectures is shown in Table 1.

Table 1. Characteristics of the CNNs' architectures used in this study.

Network	Depth (Hidden Layers)	Image Size	Parameters
CNN from scratch	24	224 × 224	1,209,058
VGG-16	16	224 × 224	134,268,738
VGG-19	19	224 × 224	143,667,240
ResNet-50	50	224 × 224	23,591,810
ResNet-101	101	224 × 224	42,662,274
ResNet-152	152	224 × 224	58,375,042
Inception v3	48	299 × 299	21,806,882
AlexNet	8	227 × 227	56,328,962

Hyperparameters

Hyperparameters are variables that define the structure of a convolutional network as well as allow it to be trained [29]. These hyperparameters are learning rate, epochs, optimizer, batch size, number of layers, and activation functions, among others, which can be adjusted to make CNN more efficient. In this study, we changed the values of the hyperparameters optimizer, learning rate, batch, and epochs. Our CNN used an optimizer Adaptive Moment Estimation (Adam) and Stochastic Gradient Descent (SGD) since those are well-known optimizers, which have good performance to classify images in CNN [30]. The learning rates for the optimizers were 0.01 and 0.001. The batch size value was 64 and 128, the epochs were 25 and 400, and the number of layers depended on the CNN architecture used (Table 1).

2.4. Experimental Framework

To implement and evaluate the CNN architectures presented in Section 2.3, we used the Google Colab cloud service based on Jupyter's Notebooks, which allows the free use of Google's GPUs or TPUs, with the libraries Scikit-learn, PyTorch, TensorFlow, Keras, and OpenCV [31]. The hardware specifications used in this experiment were GPU: Nvidia-Tesla-T4; CPU: Intel(R) Xeon(R) CPU @ 2.20 GHz; RAM: ~12.78 GB available; Hard Disk: ~32.20 GB available; and the software specifications were and Operating System: 18.04.5 LTS (Bionic Beaver) with the libraries Keras 1.0.8 and Tensorflow 1.15 as a back-end.

2.5. Performance Evaluation

The accuracy is the metric used to evaluate the classification performance of the architectures proposed in this paper. This metric calculates the percentage of samples that are correctly classified, and it is represented in the next equation:

$$\text{Accuracy} = \frac{\text{tp} + \text{tn}}{\text{tp} + \text{tn} + \text{fp} + \text{fn}} \quad (1)$$

where tp represents true positives, those that belonged to the class and were correctly classified in that class; tn represents true negatives, those that did not belong to the class and were correctly classified in another class; fp represents false positives, those that did not belong to the class and were wrongly assigned to the class; and finally, fn represents false negatives, those that belonged to the class and were mistakenly classified in another class.

3. Results

Using the Adam parameter as an optimizer, it can be observed in Table 2 that for the evaluation with 25 epochs, the highest performance percentage was for VGG-16 (96.63% and 95.27%), with a learning rate (0.001), and for VGG-19 (93.92% and 97.30%), with a learning rate (0.01). The lowest performance was for AlexNet (64.19%) and ResNet-152 (64.17%), for a learning rate (0.001), and CNN from scratch (46.62% and 53.38%), with a learning rate (0.01). On the other hand, for 400 epochs, the highest percentage was Inception V3 (98.65%) and VGG-19 (98.75%), both for a learning rate (0.001) and for Inception V3 (98.65%) and VGG-19 (99.32%), with a learning rate (0.01). Likewise, the lowest performance was for ResNet-101 and ResNet-152 (both with 80.41%) and ResNet-101 (79.05%), for a learning rate (0.001) and, finally, AlexNet (67.57%) and CNN from scratch (43.24%), both with a learning rate (0.01). It can also be observed that the two best results were for VGG-19 (99.32% and 98.65%) for a batch (128), followed by Inception V3 (98.65%) for both batches (64); all these for 400 epochs.

Table 2. Accuracy evaluation of eight CNN architectures, changing the values of the hyperparameters of a batch, learning rate, and epochs, using the Adaptive Moment Estimation (Adam) parameter as an optimizer.

Parameters	CNN from Scratch	VGG-16	VGG-19	ResNet-50	ResNet-101	ResNet-152	AlexNet	Inception V3
Epochs	25 400	25 400	25 400	25 400	25 400	25 400	25 400	25 400
Batch = 64, Optimizer = Adam, Learning Rate = 0.001								
Accuracy	93.24	96.63	90.54	68.92	71.62	74.32	64.19	96.62
(%)	94.59	96.62	95.95	81.08	80.41	80.41	88.51	98.65
Time	9	24	14	11	14	25	11	12
(min)	16	40	43	33	41	61	19	131
Batch = 128, Optimizer = Adam, Learning Rate = 0.001								
Accuracy	85.13	95.27	87.84	70.95	70.27	64.17	75.00	93.24
(%)	93.92	97.29	98.65	83.11	75.67	81.08	85.81	95.27
Time	11	12	5	13	7	9	11	13
(min)	12	34	46	45	16	54	19	48
Batch = 64, Optimizer = Adam, Learning Rate = 0.01								
Accuracy	46.62	85.81	93.92	83.78	65.54	75.68	85.81	95.95
(%)	95.27	95.95	96.62	86.49	79.05	84.46	67.57	98.65
Time	10	12	12	10	11	16	16	11
(min)	14	43	45	34	45	65	18	47
Batch = 128, Optimizer = Adam, Learning Rate = 0.01								
Accuracy	53.38	97.29	97.30	84.46	76.35	66.89	89.19	93.92
(%)	43.24	96.62	99.32	84.46	79.73	81.76	87.16	95.95
Time	11	12	13	12	11	15	11	13
(min)	16	38	43	33	46	60	18	44

Regarding the time parameter in Table 2, CNN from scratch had the lowest values for processing time. However, some values were higher than those reported by ResNet-50, ResNet-101, ResNet-152, and AlexNet architectures. Likewise, the highest processing times in 25 epochs were for ResNet-152 (25 min) and Inception V3 (13 min), with a learning rate (0.001), and for ResNet-152 and AlexNet (16 min) and ResNet-152 (15 min), for a learning rate (0.01). For 400 epochs, the highest process time was for Inception V3 (131 min) and ResNet-152 (54 min), both for a learning rate (0.001) and ResNet-152 (65 and 60 min), with a learning rate (0.01). The ResNet-152 architecture was the CNN that required the most processing time on its network for most hyperparameters. The highest processing times were not associated with high or low accuracy.

Table 3 reveals that using the Stochastic Gradient Descent (SGD) parameter as an optimizer, for an evaluation with 25 epochs, the highest performance percentage was for VGG-19 (87.16%) and VGG-16 (87.16%), with a learning rate (0.001), and for Inception V3 (92.56% and 91.89%), with a learning rate (0.01). While the lowest performance was for AlexNet (52.70%) and CNN from scratch (51.35%), for a learning rate (0.001), and for ResNet-50 and ResNet-152 (both with 45.94%) and ResNet-50 (45.94%), with a learning rate (0.01). On the other hand, for 400 epochs, the highest percentage was obtained by Inception V3 (95.94%) and CNN from scratch (94.59%), both for a learning rate (0.001), and VGG-19 (94.59%) and Inception V3 (95.27%), with a learning rate (0.01). Likewise, the lowest performance was obtained by AlexNet (56.08% and 60.81%), for a learning rate (0.001), and, finally, ResNet-50 (50% and 52.03%) with a learning rate (0.01). It can also be observed that the two best CNN architectures turned out to be CNN from scratch (94.59%) and Inception V3 (95.27%) for a batch (128), followed by Inception V3 (95.94%) and VGG-19 (94.59%) for a batch (64).

Table 3. Accuracy evaluation of eight CNNs' architectures, changing the batch's hyperparameters values, learning rate, and epochs, using the Stochastic Gradient Descent (SGD) parameter as an optimizer.

Parameters	CNN from Scratch	VGG-16	VGG-19	ResNet-50	ResNet-101	ResNet-152	AlexNet	Inception V3
Epochs	25 400	25 400	25 400	25 400	25 400	25 400	25 400	25 400
Batch = 64, Optimizer = SGD, Learning Rate = 0.001								
Accuracy	54.05	86.49	87.16	66.89	53.38	54.05	52.70	86.50
(%)	93.24	93.24	91.21	75.68	75.00	64.19	56.08	95.94
Time	12	14	13	11	12	13	11	13
(min)	16	41	46	36	47	58	19	42
Batch = 128, Optimizer = SGD, Learning Rate = 0.001								
Accuracy	51.35	87.16	85.81	68.92	53.37	53.37	53.38	79.05
(%)	94.59	90.54	92.57	71.62	74.32	64.87	60.81	93.24
Time	10	23	13	12	12	13	11	14
(min)	28	50	48	32	44	115	18	43
Batch = 64, Optimizer = SGD, Learning Rate = 0.01								
Accuracy	83.11	88.51	77.10	45.94	46.62	45.94	53.38	92.56
(%)	89.86	92.57	94.59	50.00	66.89	65.54	83.78	93.92
Time	10	12	12	11	8	14	11	12
(min)	15	45	45	32	44	58	31	44
Batch = 128, Optimizer = SGD, Learning Rate = 0.01								
Accuracy	79.72	78.37	79.73	45.94	46.62	53.37	54.05	91.89
(%)	91.26	90.54	88.51	52.03	56.08	64.19	80.41	95.27
Time	11	11	12	11	69	13	34	12
(min)	16	41	44	53	44	54	21	42

Table 3 shows that, for the time parameter, there was no defined pattern to identify the architecture that presented the lowest processing time in all its hyperparameters. Low values mostly appeared for CNN from scratch. However, the lowest value was for the ResNet-101 model with 8 min, in epochs (25), batch (64), and learning rate (0.01). Likewise, the accuracy of CNN from scratch was better than that reported by ResNet-50, ResNet-101,

ResNet-152, and AlexNet architectures. The highest processing times in 25 epochs was for VGG-16 (14 and 23 min), with a learning rate (0.001), and for ResNet-152 (14 min) and ResNet-101 (69 min), for a learning rate (0.01). For 400 epochs, the highest process time was for ResNet-152 (58 and 115 min) for a learning rate (0.001) and (58 and 54 min), with a learning rate (0.01). Finally, ResNet-52 architecture required the most processing time for most hyperparameters. The highest processing times were not associated with high or low accuracy.

4. Discussion

Convolutional Neural Networks (CNNs) are used in several agriculture areas such as leaf and plant disease detection, land cover classification, crop type classification, plant recognition, segmentation of root and soil, crop yield estimation, fruit counting, obstacle detection in row crops and grass mowing, and identification of weeds, to mention a few [32,33]. For example, in Mohanty et al. [34], they presented the training of CNN architectures AlexNet and Google Net with a PlanVillage image data set to detect 26 types of diseases in 14 kinds of crops. Their results showed an accuracy of 99.35% to identify healthy and diseased plants. Meanwhile, Rahnemonfar and Sheppard [35] proposed using the CNN architectures' inception and Residual Networks (ResNet) architectures to estimate the yield of a tomato plant using synthetic images. Their results indicated that, with 91% accuracy, they can evaluate the yield.

Another example was presented in [36], where authors proposed training several convolutional networks to identify four fruits (mango, orange, apple, and banana). They were classified into two categories: fresh and rotten. The best performing models were Inception version 3 and the Visual Geometric Group of 16-layer (VGG-16) architectures, which received the learning transfer. Their results showed identification and classification percentages of 90% accuracy. A similar study was presented in [13], where the use of a VGG-16 network to classify vegetables and fruits was proposed. A total of 26 categories were classified: pumpkin, celery, cauliflower, pineapple, pomegranate, grapefruit, banana, cucumber, broccoli, onion, carrot, etc. The authors claimed to have 95.6% accuracy in classifying these fruits and vegetables. Regarding dates, we identified research works that proposed using CNNs to sort among dates or to detect among their different maturity stages [8,9,16].

Currently, determining the stage of maturity in the Medjool date using traditional image processing and machine learning methods is complicated. This is because these methods are trained to extract features in various cultivars such as their appearance, color (associated with the maturity stages), shape, and texture [7,16]. However, there are no studies where a feature extraction or predictive model for sorting Medjool dates that we are aware of. Furthermore, recent models cannot determine sorting Medjool because this cultivar is harvested, sorted, packaged, and consumed in its Tamar stage.

To contribute with a model that may be useful in sorting the Medjool date through images, we compared the performance of eight CNN architectures in this study. Additionally, some hyperparameters' values were modified, and transfer learning was used to identify and propose the use of CNN with the best precision.

As shown in Table 2, our findings indicated that when we use an Adam optimizer, the VGG architectures show the best accuracy, with the VGG-19 model that reached the highest percentage of accuracy with 99.32%. Likewise, the ResNet and CNN from scratch architectures showed the lowest performance percentages; the CNN from a scratch model achieved the most insufficient precision, with 43.24%. The highest average percentage generated among the eight architectures was 89.53%, using the combination of batch (64), learning rate (0.01), and epochs (400), with an average time of 48.71 min, while the lowest was 75%, combining a batch (64), learning rate (0.001), and epochs (25), with an average time of 12.25 min.

Likewise, Table 3 indicates that no architecture showed the best accuracy when we used an SGD parameter as an optimizer. However, the ResNet-50 architecture showed

the lowest performance percentages, with batch (64 and 128) and learning rate (0.001). The highest percentage generated among the eight architectures was 80.57%, using the combination of batch (64), learning rate (0.001), and epochs (400), with an average time of 38.13 min, while the lowest was 66.21%, combining a batch (128), learning rate (0.01), and epochs (25), with an average time of 21.63 min.

It was noticeable that, if the number of epochs for all models was increased, the percentage of accuracy and required processing time also increased. Likewise, we observed that the highest processing times corresponded to the ResNet-152 architecture, which could be associated with the fact that this architecture had the highest number of layers. However, none of its precision was higher than 85% performance.

The optimizer can help us minimize the error function that allows us to conform to the training set examples. In this study, the accuracy was higher for Adam than for SGD.

Several studies have focused on identifying the CNN that offers the best precision for selecting dates from cultivars in their various stages of maturity [8,9,18]. However, there are currently no reported studies that use any CNN to classify the date cultivar Medjool.

Table 4 compares similar studies to ours, where CNNs' architectures with the best performance have been reported. Nasiri et al., 2019 [9], only worked on VGG-16 with two hyperparameters, obtaining the highest accuracy of 96.98%. Likewise, Altaheri et al., 2019 [8], worked on two CNN with transfer learning and fine-tuning, modifying three hyperparameters twice, obtaining the highest percentage for VGG-16 with an accuracy of 97.25%. Faisal et al., 2020 [16], compared four CNNs' performances, evaluating four hyperparameters, resulting in ResNet as the best model, with an accuracy of 99.01%. Finally, our study evaluated eight CNNs' performances, using transfer learning and modifying four hyperparameters twice, resulting in the VGG-19 model with the highest performance, with 99.32% accuracy.

Table 4. Comparison of studies that report CNN architectures in the detection of various stages of maturity in the date palm fruit.

Reference	Date Palm Cultivar	Maturity Stages	Number of Images (Dataset)	CNN Architectures	Hyperparameters	Best Accuracy
Nasiri et al., 2019 [9]	Shahani	Khalal, Rutab, Tamar, and defective date	+1300 images	VGG-16	Epochs = 15 Batch = 32	VGG-16 96.98%
Altaheri et al., 2019 [8]	Barhi, Khalas, Meneifi, Naboot Saif and Sullaj	Immature-1, Immature-2, pre-Khalal, Khalal, Khalal-with-Rutab, pre-Tamar, and Tamar.	8072 images	AlexNet, VGG-16, Transfer learning and Fine-Tuning	Epochs = 50 and 200 Batch = 32 and 128 Learning rate = 0.0001, 0.0002	VGG-16 97.25%
Faisal et al., 2020 [18]	Barhi, Khalas, Meneifi, Naboot Saif, and Sullaj	Immature, Khalal, Khalal with Rutab, Pre-Tamar, and Tamar	8079 images	ResNet, VGG-19, Inception V3, NASNet and support vector machine (SVM) (regression and linear)	Epochs = 50 Batch = 16 Optimizer = Adam Learning rate = 0.0001	ResNet 99.01%
This Study	Medjool	Ripe and unripe	1002 images	VGG-16, VGG-19, Inception V3, ResNet-50, ResNet-101, ResNet-152, AlexNet, CNN from scratch	Epochs = 25 and 400 Batch = 64 and 128 Optimizers = Adam, Stochastic Gradient Descent Learning rate = 0.001, 0.01	VGG-19 99.32%

One aspect to consider in this comparison is that the Medjool date is only consumed in its Tamar stage. Therefore, this study only used two stages for its sorting. The number of images was lower compared to the rest of the studies. However, in our work, the percentage of accuracy was higher due to the application of transfer learning and modification in various hyperparameters, which influence architectures' performance [37,38].

In our study, resulting from choosing the hyperparameters epochs (400), batch (128), optimizer (Adam), and a learning rate (0.01), we identified that VGG-19 architecture had the best performance. Likewise, this architecture could be included as part of the software that controlled a robotic mechanism to support the date palm farmer in an automated system of sorting ripe fruits.

5. Conclusions

This study evaluated the precision and processing time of eight CNN architectures. Seven of them were pretrained by an extensive image database designed for object recognition (ImageNet). These models were named VGG-16, VGG-19, Inception V3, ResNet-50, ResNet-101, ResNet-152, and AlexNet, which received transfer learning when their last classification layer was replaced. Additionally, a model that learns from scratch was used, that is, without obtaining learning.

All CNN architectures were evaluated by modifying the epochs, batch, optimizer, and learning rate hyperparameters since these parameters have been reported to have positive effects on the performance of convolutional networks. The results indicated that the CNN with the best performance for the sorting Medjool date was the architecture of the VGG group, which used the Adam optimizer. From these architectures, the VGG-19 model was the one that reported the best accuracy, with 99.32%. Likewise, the ResNet group architectures were the ones that reported the lowest performance using the same optimizer, the ResNet-152 model, which reported the most insufficient accuracy, with 64.17%. The use of the SGD optimizer did not have a significant effect on obtaining high accuracies.

Finally, it will be necessary to continue working on the best accuracy and the shortest processing time, with the modification of other hyperparameters. The inclusion in the evaluation of different fruit attributes, such as its size, gives it a high commercial value. It is essential in the packing process of this fruit.

Author Contributions: Conceptualization, B.D.P.-P. and J.P.G.V.; methodology, J.P.G.V. and R.S.-T.; software, B.D.P.-P.; validation, B.D.P.-P., J.P.G.V., and R.S.-T.; formal analysis, R.S.-T. and J.P.G.V.; investigation, J.P.G.V. and R.S.-T.; resources, B.D.P.-P.; data curation, B.D.P.-P.; writing, original draft preparation, J.P.G.V. and R.S.-T.; writing, review and editing, R.S.-T.; visualization, B.D.P.-P. All authors contributed to its editing and approved the final draft. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Dataset is available on https://data.mendeley.com/datasets/872xk9npmz/1.

Acknowledgments: We would like to thank CONACyT for the scholarship granted to the first author (CVU-409617). We would like to thank Ramiro Quiroz, of the company Palmeras RQ. We accessed his plantation to take the photographs. We thank Emmanuel Santiago Durazo for his contributions to the work done.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Salomon-Torres, R.; Sol-Uribe, J.A.; Salasab, B.; García, C.; Krueger, R.; Hernández-Balbuena, D.; Norzagaray-Plasencia, S.; García-Vázquez, J.P.; Ortiz-Uribe, N. Effect of Four Pollinating Sources on Nutritional Properties of Medjool Date (*Phoenix dactylifera* L.) Seeds. *Agriculture* **2020**, *10*, 45. [CrossRef]
2. Chao, C.T.; Krueger, R.R. The Date Palm (*Phoenix dactylifera* L.): Overview of Biology, Uses, and Cultivation. *HortScience* **2007**, *42*, 1077–1082. [CrossRef]
3. Radwan, E.; Radwan, E. The current status of the date palm (Phoenix dactylifera) and its uses in the Gaza Strip, Palestine. *Biodiversitas J. Biol. Divers.* **2017**, *18*, 1047–1061. [CrossRef]
4. Aleid, S.M. Dates. In *Tropical and Subtropical Fruits: Postharvest Physiology, Processing and Packaging*; Siddid, M., Ahmed, J., Lobo, M.G., Ozadali, F., Eds.; Wiley: New York, NY, USA, 2012; pp. 179–202.
5. Food and Agriculture Organization. FAOSTAT. Available online: http://www.fao.org/faostat/en/#data/QL (accessed on 11 December 2020).
6. Ortiz-Uribe, N.; Salomon-Torres, R.; Krueger, R. Date Palm Status and Perspective in Mexico. *Agriculture* **2019**, *9*, 46. [CrossRef]
7. Altaheri, H.; Alsulaiman, M.; Muhammad, G.; Amin, S.U.; Bencherif, M.; Mekhtiche, M. Date fruit dataset for intelligent harvesting. *Data Brief* **2019**, *26*, 104514. [CrossRef] [PubMed]

8. Altaheri, H.; Alsulaiman, M.; Muhammad, G. Date Fruit Classification for Robotic Harvesting in a Natural Environment Using Deep Learning. *IEEE Access* **2019**, *7*, 117115–117133. [CrossRef]
9. Nasiri, A.; Taheri-Garavand, A.; Zhang, Y.-D. Image-based deep learning automated sorting of date fruit. *Postharvest Biol. Technol.* **2019**, *153*, 133–141. [CrossRef]
10. Guo, Z.; Zhang, M.; Lee, D.-J.; Simons, T. Smart Camera for Quality Inspection and Grading of Food Products. *Electronics* **2020**, *9*, 505. [CrossRef]
11. Sa, I.; Ge, Z.; Dayoub, F.; Upcroft, B.; Perez, T.; McCool, C. DeepFruits: A Fruit Detection System Using Deep Neural Networks. *Sensors* **2016**, *16*, 1222. [CrossRef]
12. Steinbrener, J.; Posch, K.; Leitner, R. Hyperspectral fruit and vegetable classification using convolutional neural networks. *Comput. Electron. Agric.* **2019**, *162*, 364–372. [CrossRef]
13. Zeng, G. Fruit and vegetables classification system using image saliency and convolutional neural network. In Proceedings of the 2017 IEEE 3rd Information Technology and Mechatronics Engineering Conference (ITOEC), Chongqing, China, 3–5 October 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 613–617.
14. Rudnik, K.; Michalski, P. A Vision-Based Method Utilizing Deep Convolutional Neural Networks for Fruit Variety Classification in Uncertainty Conditions of Retail Sales. *Appl. Sci.* **2019**, *9*, 3971. [CrossRef]
15. Muhammad, G. Date fruits classification using texture descriptors and shape-size features. *Eng. Appl. Artif. Intell.* **2015**, *37*, 361–367. [CrossRef]
16. Haidar, A.; Dong, H.; Mavridis, N. Image-based date fruit classification. In Proceedings of the 2012 IV International Congress on Ultra Modern Telecommunications and Control Systems, St. Petersburg, Russia, 3–5 October 2012; IEEE: Piscataway, NJ, USA, 2012; pp. 357–363.
17. Manickavasagan, A.; Al-Mezeini, N.; Al-Shekaili, H. RGB color imaging technique for grading of dates. *Sci. Hortic.* **2014**, *175*, 87–94. [CrossRef]
18. Faisal, M.; Albogamy, F.; ElGibreen, H.; Algabri, M.; AlQershi, F.A. Deep Learning and Computer Vision for Estimating Date Fruits Type, Maturity Level, and Weight. *IEEE Access* **2020**, *8*, 206770–206782. [CrossRef]
19. Dodge, S.; Karam, L. Understanding how image quality affects deep neural networks. In Proceedings of the 2016 Eighth International Conference on Quality of Multimedia Experience (QoMEX), Lisbon, Portugal, 6–8 June 2016; IEEE: Piscataway, NJ, USA, 2016; pp. 1–6.
20. Geirhos, R.; Janssen, D.H.; Schütt, H.H.; Rauber, J.; Bethge, M.; Wichmann, F.A. Comparing deep neural networks against humans: Object recognition when the signal gets weaker. *arXiv* **2017**, arXiv:1706.06969.
21. Feng, J.; Darrell, T. Learning the Structure of Deep Convolutional Networks. In Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015; IEEE: Piscataway, NJ, USA, 2015; pp. 2749–2757.
22. Zaccone, G.; Karim, M.R. *Deep Learning with TensorFlow: Explore Neural Networks and Build Intelligent Systems with Python*; Packt Publishing Ltd.: Birmingham, UK, 2018.
23. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv* **2014**, arXiv:1409.1556.
24. Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; Wojna, Z. Rethinking the inception architecture for computer vision. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 2818–2826.
25. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
26. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*; Curran Associates, Inc.: New York, NY, USA, 2012; pp. 1097–1105.
27. Zhu, Q.; He, Z.; Zhang, T.; Cui, W. Improving Classification Performance of Softmax Loss Function Based on Scalable Batch-Normalization. *Appl. Sci.* **2020**, *10*, 2950. [CrossRef]
28. Pan, S.J.; Yang, Q. A Survey on Transfer Learning. *IEEE Trans. Knowl. Data Eng.* **2009**, *22*, 1345–1359. [CrossRef]
29. Han, J.-H.; Choi, D.-J.; Park, S.-U.; Hong, S.-K. Hyperparameter Optimization Using a Genetic Algorithm Considering Verification Time in a Convolutional Neural Network. *J. Electr. Eng. Technol.* **2020**, *15*, 721–726. [CrossRef]
30. Bera, S.; Shrivastava, V.K. Analysis of various optimizers on deep convolutional neural network model in the application of hyperspectral remote sensing image classification. *Int. J. Remote. Sens.* **2019**, *41*, 2664–2683. [CrossRef]
31. Bisong, E. Google Colaboratory. In *Building Machine Learning and Deep Learning Models on Google Cloud Platform*; Apress: Berkeley, CA, USA, 2019; pp. 59–64.
32. Kamilaris, A.; Prenafeta-Boldú, F.X. A review of the use of convolutional neural networks in agriculture. *J. Agric. Sci.* **2018**, *156*, 312–322. [CrossRef]
33. Kamilaris, A.; Prenafeta-Boldú, F.X. Deep learning in agriculture: A survey. *Comput. Electron. Agric.* **2018**, *147*, 70–90. [CrossRef]
34. Mohanty, S.P.; Hughes, D.P.; Salathé, M. Using Deep Learning for Image-Based Plant Disease Detection. *Front. Plant. Sci.* **2016**, *7*, 1419. [CrossRef]
35. Rahnemoonfar, M.; Sheppard, C. Deep Count: Fruit Counting Based on Deep Simulated Learning. *Sensors* **2017**, *17*, 905. [CrossRef] [PubMed]
36. Ashraf, S.; Kadery, I.; Chowdhury, A.A.; Mahbub, T.Z.; Rahman, R.M. Fruit Image Classification Using Convolutional Neural Networks. *Int. J. Softw. Innov.* **2019**, *7*, 51–70. [CrossRef]

37. Kandel, I.; Castelli, M. The effect of batch size on the generalizability of the convolutional neural networks on a histopathology dataset. *ICT Express* **2020**, *6*, 312–315. [CrossRef]
38. Motta, D.; Santos, A.; Álisson, B.; Machado, B.A.S.; Ribeiro-Filho, O.G.V.; Camargo, L.O.A.; Valdenegro-Toro, M.A.; Kirchner, F.; Badaró, R. Optimization of convolutional neural network hyperparameters for automatic classification of adult mosquitoes. *PLoS ONE* **2020**, *15*, e0234959. [CrossRef]