Classification of Multiple Panamanian Watermelon Varieties Using Convolutional Neural Networks and Transfer Learning

Kenji Contreras[1], Danilo Cáceres-Hernández[1], Anel Henry[1], Fatima Rangel[1], Emmy Saez[1], Kang-Hyun Jo[2] and Javier E. Sánchez-Galán[1]

[1] Universidad Tecnológica de Panamá
{kenji.contreras, anel.henry, fatima.rangel, emmy.saez, danilo.caceres, javier.sanchezgalan}@utp.ac.pa
[2] Intelligent Systems Laboratory, Graduate School of Electrical Engineering, University of Ulsan, Ulsan, South Korea acejo@ulsan.ac.kr

Abstract: Panama is well regarded as an international exporter of watermelon in the Central American region. Most of the selection process is made by hand with empirical techniques, based on firmness, color, sound and random sampling from other specimens of the same batch. The overall goal of the project is to have an automated system able to distinguish between watermelon varieties. and that is able to classify watermelons that are ready for export or can be sold locally. For this matter, traditional and novel computer vision and spectral pattern recognition algorithms are used.

Objective: Assess the capability of models pre-trained on VGG19 and EfficientNetB0, are able to classify local watermelon varieties.

Materials and Methods: A data set comprised of watermelon images was created for each of 3 local export varieties (Joya, Anna, Quetzali) provided by "*La Asociación De Productores de Sandía de Exportación Cascajalillo Unido-APSECU*" (Herrera, Panama). These varieties were selected also, due to the fact that each one has very interesting characteristics, not only attending to pattern, color and stripes. Examples of the varieties of the dataset can be seen in **Figure 1**.

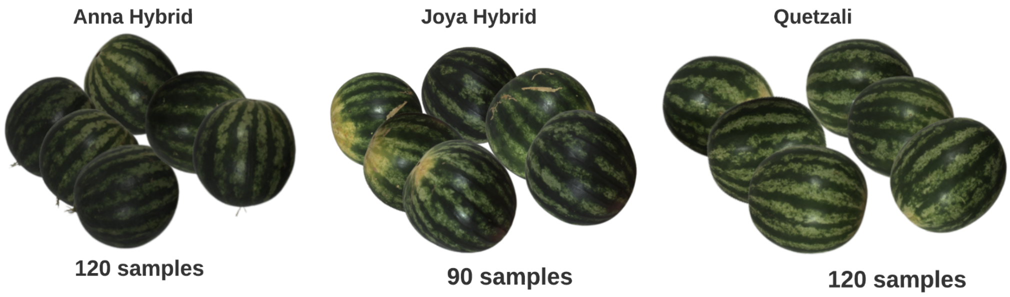

Figure 1 - Watermelon varieties used in this study.

RGB watermelon images were originally captured at a resolution of 360x270 pixels (with a white background for reference), but subsampled, cropped and resized to size of 40x40 pixels, as can be seen in Figure 2.

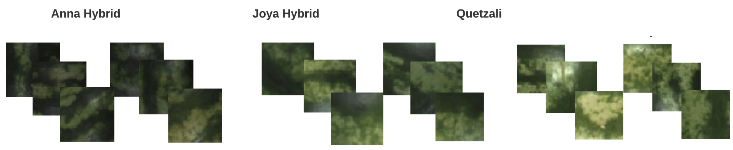

Figure 2 - Pigment samples for each variety.

Model Selection: Given the fact that the original data set has a significantly small sample size, the training of a fully customized Convolutional Neural Network (CNN) architecture was avoided, as it was prone to have high overfitting. Instead Transfer Learning (TL) was used as a Feature Extraction (FE) method, based on results from previous works from Pardede [1] and Zhang [2]. The initial architecture consisted of a basic model that resembled common CNN architectures such as VGG16 and 19. Used by freezing all pre-trained layers and replacing the top layers with customized fully connected layers (dense) layers.

Regularization: To avoid overfitting, several techniques could be included, however, according to Pardede [1], Dropout layers showed to outperform other techniques such as Batch Normalization for TL. Additionally, Global Average Pooling (GAP) was used as the final layer of the pre-trained block instead of a flattening layer. It was suggested by Chen [3] that a GAP layer is more native to the convolution structure by enforcing correspondences between feature maps and categories, including the aggregation of spatial information, and thus, adding robustness to spatial translations.

Hyper Parameter Tuning: Furthermore, an optimization algorithm known as Grid Search in conjunction with K-Fold Cross-Validation was utilized to search over a given subset of the hyper parameters space of the training algorithm to optimize the top layers. The number of neurons per dense layer (4, 8, and 12) was chosen to take into account the low dimensionality of the input images (40x40 pixels) and dropout values were set iterate over 0.3, 0.5, and 0.7.

Testing other pre-trained models: Taking advantage of the wide variety of pre-trained models, more efficient and state-of-the-art models such as EfficientNet [4] were tested in conjunction with the previously developed fully connected layers. In comparison to older models such as VGG16 and 19, EfficientNet is among the most efficient models that reach state-of-the-art accuracy on both the ImageNet dataset and other transfer learning tasks. In this work, the first variant (B0) was chosen since the scaling architecture complexity of higher variants was deemed to be unnecessary when considering the dataset's complexity.

As it can be seen in Figure 3, The architecture consisted of an Input Layer, Preprocessing Layers (only for EfficientNetB0). Weights from Pre-trained Model, a Global Average Pooling 2D layer, a Dense Layer (8 neurons), an Activation function (Rectifier Linear Unit-ReLu), a Dense Layer (12 neurons), a second ReLu activation function, a Dropout layer with P=0.3, a dense Layer (consisting of 3 neurons) and finally an activation function with Softmax as output. The Loss Function was selected to be Categorical Cross-Entropy.

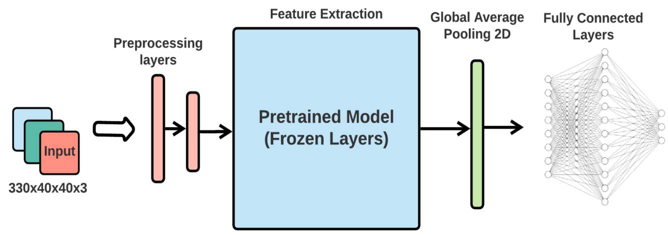

Figure 3 - Model Architecture

As an Optimizer the Adam algorithm was selected, with an alpha= 0.001, Beta 1=0.9, Beta 2=0.999, and a tolerance of Epsilon: 1e-07. A K=5 folds Cross Validation was made, with a 20%/80% train/test split, with 70 training epochs and a Batch size of 16. All models were built and tested using Python's Keras library in Jupyter Notebooks hosted within the Google Collaboratory platform. It is noteworthy that each of Keras' pre-trained model includes a custom preprocessing function to adequate the input image to the neural network, in the case of EfficientNetB0, the preprocessing function is replaced by default preprocessing layers inside the model that perform input rescaling and normalization.

Results: The following table shows the results after training for 70 epochs both models using (K=5) Fold Cross Validation:

	VGG19		EfficientNetB0	
	Val. Loss	Val. Accuracy	Val. Loss	Val. Accuracy
Fold 1	0.7639	71.21%	0.5172	78.78%
Fold 2	0.4681	89.39%	0.1884	90.90%
Fold 3	0.1997	95.45%	0.1034	96.96%
Fold 4	0.0300	98.48%	0.0796	95.45%
Fold 5	0.1865	93.93%	0.1268	95.45%
Average	0.3295	89.69%	0.2031	91.51%

Table 1 - Transfer Learning Results.

The curves depicted in **Figure 4** show the behaviour of each model during each epoch in terms of accuracy and loss:

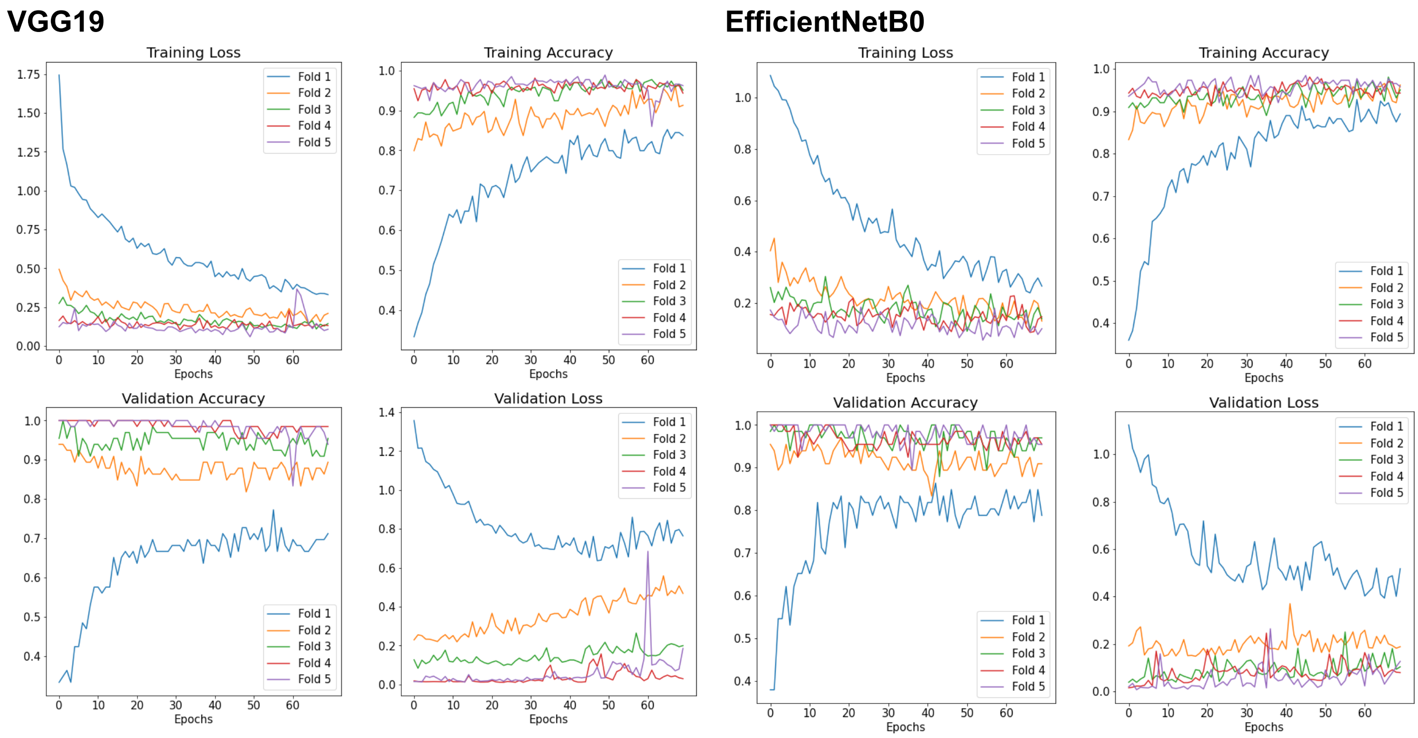

Figure 4 - Resulting Training / Test Accuracy and Loss Curves for the Models

Conclusions:

- Despite achieving an acceptable accuracy rate and relatively low loss rates, by observing the fluctuation of accuracy per epoch, it can be inferred that both models show signs of slight overfitting. The first fold shows a significant drop in accuracy rate in comparison to other folds, moreover, the several peaks might indicate that the model does not learn appropriately every batch of samples due to the low amount of samples of each class.
- During early experimentation, data augmentation techniques such as random image rotation, flips, height or width shifting did not improve results. This could be due to the fact that certain features pertaining to pigments which identify each watermelon variety should not be spatially distorted.
- The slight improvement in average accuracy and loss between VGG19 and EfficientNetB0 might indicate that more modern and complex models could hold the potential for a more optimized model.

Future Work:

- Test the models using a different dataset where instead of images of regions of 40x40 pixels, use a pre-processing ROI size and not a fixed bounding box.
- Test other pre-trained models.
- Optimize more parameters such as number of epochs, optimizer and number of dense layers using Grid Search or other algorithms.

Acknowledgements: This work is supported by the Secretaría Nacional de Ciencia, Tecnología e Innovación de Panamá (SENACYT) Grant funded by the Panamanian Government (Project 165- 2019-FID18-060). The authors would like to thank the Sistema Nacional de Investigación of Panamá, APSECU and the Universidad Tecnológica de Panamá for their administrative support and contribution.

References:

[1] J. Pardede, B. Sitohang, S. Akbar, and M. L. Khodra, “Implementation of transfer learning using vgg16 on fruit ripeness detection,” International Journal of Intelligent Systems & Applications, vol. 13, no. 2, 2021.
[2] Y.-D. Zhang, Z. Dong, X. Chen, W. Jia, S. Du, K. Muhammad, and S.-H. Wang, “Image based fruit category classification by 13-layer deep convolutional neural network and data augmentation,” Multimedia Tools and Applications, vol. 78, no. 3, pp. 3613–3632, 2019.
[3] M. Lin, Q. Chen, and S. Yan, “Network in network,” arXiv preprint arXiv:1312.4400, 2013.
[4] M. Tan and Q. Le, “Efficientnet: Rethinking model scaling for convolutional neural networks,” in International Conference on Machine Learning.PMLR, 2019, pp. 6105–6114.