JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

A CNN-BASED APPROACH TO CLASSIFY ARECA NUTS BASED ON GRADES

[1] Ms. Amrutha Bhat, [2] Adarsh S Shetty, [3] Dishan Muralidhar Suvarna, [4] Goutham, [5] Rishika S Anchan

[1] Assistant Professor, [2 3 4 5] Student

[1 2 3 4 5] Department of Computer Science & Engineering, MITE, Moodabidri

Abstract: The classification and grading of arecanut is a crucial aspect of the arecanut industry. Arecanut is widely used as a stimulant and mouth freshener in many cultures, and its demand is increasing rapidly. The quality of the arecanut is determined by its size, colour, texture, and taste. Therefore, proper classification and grading of arecanut are necessary to maintain the quality of the product. The aim of this project is to develop a machine learning-based system that can classify and grade arecanut automatically. The proposed system will use image processing techniques to extract features from the arecanut images and then use machine learning algorithms to classify and grade them. The system will be trained using a dataset of arecanut images that have already been classified and graded by experts. The extracted features will be used to train a machine learning model that can classify the arecanut based on its size, colour and texture. The model will also be able to grade the arecanut based on its quality. The proposed system has several advantages over the traditional manual method of arecanut classification and grading. It is more accurate, efficient, and can process many arecanut samples in a short time. Moreover, it eliminates human errors and biases, thereby increasing the overall quality of the arecanut product. In conclusion, the proposed system for classification and grading of arecanut will be a significant contribution to the arecanut industry. It will enable the industry to maintain the quality of the product consistently, which will ultimately lead to increased customer satisfaction and higher profits.

***Index Terms* - Arecanut, classification, grading, machine learning, image processing, feature extraction, accuracy.**

I. INTRODUCTION

One of the most significant commercial plants in India is the areca nut. Right now, sorting areca nuts by hand is a labor-intensive and ineffective process. The current approach of areca nuts quality is simply observed with naked eye and farmers must carefully analyze each crop which is a very time-consuming process. The website was made to categorize areca nuts based on quality. The division of the areca nut is significant for grading. Finding the right yield is aided by accurate sorting. The CNN method of in-depth learning may take an image, assign value (readable weight and bias) to the many elements in the image, and distinguish between them. Comparatively speaking to other classification techniques, CNN requires very little preparatory processing. CNN can comprehend these filters / symbols with the proper training, even though the filters are manually created using conventional methods.

II. LITERATURE REVIEW

Ajit Danti -> In this paper, a novel method is proposed for classification of arecanut into two classes based on color. The proposed method has three steps: (i) Segmentation; (ii) Masking; and (iii) classification. Classification is done based on the red and green color components of the segmented region of the arecanuts. Test performance success rates ranged from 97 to 98 percent depending on the category.

Shabari Shedthi Billadi -> This paper provides a unique solution for classifying the good and defective arecanuts based on their color, texture, and density value. A unique density feature is considered here for better classification. The result of classifiers without considering the density feature is compared with respect to the density feature and it is observed that artificial neural networks work better than the others. The proposed method works effectively for classifying arecanut with an accuracy of 98.8%.

Dinesh R -> Colour, size, and texture are used by the author to identify the areca nut category. This article's major objective is to provide a thorough explanation of the areca nut, Computer Vision, and the technological requirements and applications based on areca nut classification and grading.

Siddesha S -> In this paper, we propose the texture-based grading of arecanut. The KNN method was used to create an 800-photographic database of four classes utilizing two-color features and four-grade scales. For classification Nearest Neighbor (NN) classifier is used. Experimentation conducted using a dataset of 700 images of 7 classes to demonstrate the proposed model's performance. 91.43% of classification rate is achieved with Gabor wavelet features.

Bharadwaj N K->The proposed approach makes use of global textural feature viz., Local Binary Pattern for feature extraction. Supports a vector classification machine for determining the range of areca nuts. The confusion matrix's accuracy, memory, and F-measure are all used to test and grade the system's performance.

III. METHODOLOGY

There are many methods involved in building this project. From collecting the datasets to loading them into the model and then training them to give accurate results. We have gone for Convolution Neural Network (CNN) which is a deep learning algorithm used for the sole purpose of classification. Then imported the trained model to the application that we built to give real time experience for the users.

3.1 Datasets

The Dataset for Areca Nut was collected manually by visiting different Areca Nut factories located nearby. By far we have successfully collected over 1800+ samples across industries. We also had to gain knowledge about the same in order to classify them so we asked the locals who have been well versed with the knowledge about areca nuts and we received many feedbacks that involved the type, texture, color, etc. So based on the Information given we have classified the Areca Nuts into three grades based on their quality and price in the market.

Table 1 List of Datasets Collected

Quality	Datasets
Grade A	720
Grade B	535
Grade C	616

3.2 Convolution Neural Network (CNN)

CNN is an abbreviation for Convolutional Neural Network, a deep learning technique that is extensively used for image classification, object recognition, and other image and video analysis applications.

CNNs are meant to automatically learn and extract information from pictures by employing a sequence of convolutional layers, followed by pooling layers, fully connected layers, and an output layer. The convolutional layers use a mathematical technique known as convolution to the input picture to extract significant characteristics such as edges, textures, and shapes. To decrease computation costs and prevent overfitting, the pooling layers down sample the feature maps created by the convolutional layers. The fully connected layers conduct the final classification or regression operation based on the retrieved characteristics.

CNNs have transformed the area of computer vision by delivering cutting-edge performance on a variety of image-related tasks such as picture classification, object recognition, segmentation, and more. They have been employed in a variety of applications, including self-driving cars, medical picture processing, and surveillance systems.

3.3 Implementation

To Implement the CNN technique, we had to import some libraries into our Project. It includes TensorFlow which is used to build the CNN model in our Project. We also imported matplotlib library, which is used for creating visualizations. Next, we imported the required layers from the TensorFlow Keras API for building a CNN model. These layers include Conv2D, MaxPooling2D, Dropout, Flatten, and Dense. We also imported the Sequential model from the TensorFlow Keras API, which is a linear stack of layers.

We use pathlib module, which provides an object-oriented way to manipulate the file system paths in Python through which we import the datasets into the project. After loading image data from a directory, we split it into training and validation datasets, and create TensorFlow Dataset objects for each of them. We also apply preprocessing to both train and validation dataset that includes path the dataset, validation split, image size, batch size, subset, etc. Each of these parameters are passed as the arguments to the TensorFlow preprocessing method.

After the preprocessing the dataset are sent to augmentation. Data augmentation is a technique used to artificially increase the size of a dataset by generating new samples through random transformations of the existing samples. This can help improve the generalization and robustness of a machine learning model by increasing its exposure to a wider variety of training examples.

We then load the Sequential() method which is the base architecture of the model. Sequential() creates a new sequential model object in Keras. A sequential model is a linear stack of layers, where the output from one layer is fed as input to the next layer. This is a common type of neural network architecture used in deep learning. Once the Sequential() object is created, layers can be added to it using the .add() method. The layers can be various types, such as convolutional, pooling, activation, or dense layers. The order in which the layers are added determines the flow of data through the network. For example, in a typical CNN, convolutional and pooling layers are used to extract features from the input image, followed by one or more dense layers to perform classification.

First, we add the data augmentation layer to the model. This will randomly apply transformations to the input images during training to help improve generalization performance. Next, we add a 2D convolutional layer to the model with 64 filters of size 5x5, an input shape of (512,512,3), and the relu activation function. This layer applies a set of 64 convolutional filters to the input image to extract features. Then we add a max pooling layer to the model with a pool size of 2x2. This layer down samples the feature maps from the previous convolutional layer to reduce the spatial dimensionality and help reduce overfitting. Then we add a dropout layer to the model with a dropout rate of 0.2. Dropout randomly drops out a fraction of the inputs to the layer during training to help prevent overfitting.

We continue the above process for another 3 layers. Each of the convolutional layers applies a set of filters to the input image or feature maps to extract relevant features. The number and size of the filters can impact the complexity and expressive power of the model. Relu activation function is used to introduce non-linearity into the model. Max pooling layers are used to down sample the feature maps and help reduce overfitting. Dropout layers are used to prevent overfitting by randomly dropping out a fraction of the inputs to the layer during training.

After the last layer, the Flatten() layer is used to convert the 2D feature maps into a 1D vector, which is then passed to a fully connected layer. The Dense(units=180, activation='relu') creates a fully connected layer with 180 hidden units and a ReLU activation function. The number of hidden units in the fully connected layer is usually determined by experimenting with different values to achieve the best performance for the given task. A higher number of units can lead to a more complex model, but may also lead to overfitting. The Dropout(dropout rate) layer is used to randomly drop out a certain proportion of the output from the previous layer during training. In this case the dropout rate is 0.2. This is a regularization technique that helps prevent overfitting by reducing the reliance of the model on any one input feature.

Finally, the last Dense layer with 3 units and softmax activation is used to output the class probabilities for the three possible output classes. The softmax activation function is used here to ensure that the sum of the output probabilities is equal to 1, making it suitable for multi-class classification problems. The number of units in this layer is determined by the number of classes in the output. The choice of 180 units for the first fully connected layer is specific to the given model and dataset. It was likely determined through experimentation to achieve the best trade-off between model complexity and performance. However, the optimal number of units can vary depending on the specific problem and dataset, and it is always recommended to experiment with different values to find the best performing model.

The build() method is used to build the model and specify the input shape. The batch size is kept none which is not fixed and can vary during training. The compile() method is used to compile the model. It specifies the optimizer to use for training, the loss function to use, and the evaluation metric to use. Here, we are using the Adam optimizer, categorical cross entropy as the loss function, and accuracy as the evaluation metric. Categorical cross entropy is a loss function commonly used for multi-class classification problems where there are two or more mutually exclusive classes. It measures the dissimilarity between the predicted class probabilities and the true class probabilities, and the goal of the training process is to minimize this loss. The summary() method is used to print a summary of the model architecture, including the number of layers, the output shape of each layer, and the number of trainable parameters in the model. This summary can be helpful to get an overview of the model's structure and to check if the model is constructed as intended.

The fit() method is used to train the model on the training data and validate it on the validation data for 10 epochs. The fit() function returns a history object which contains information about the training process such as the loss and accuracy at each epoch.

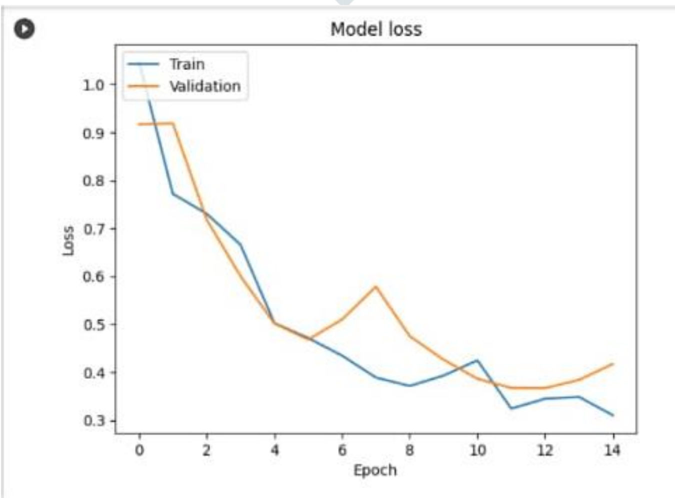
The dataset has been divided into batches of 32 images as specified by the batch size variable in the code. So, the total number of batches is equal to the total number of images divided by the batch size. So, during each epoch of training or validation, the progress bar displays the number of completed batches out of the total number of batches for that dataset.

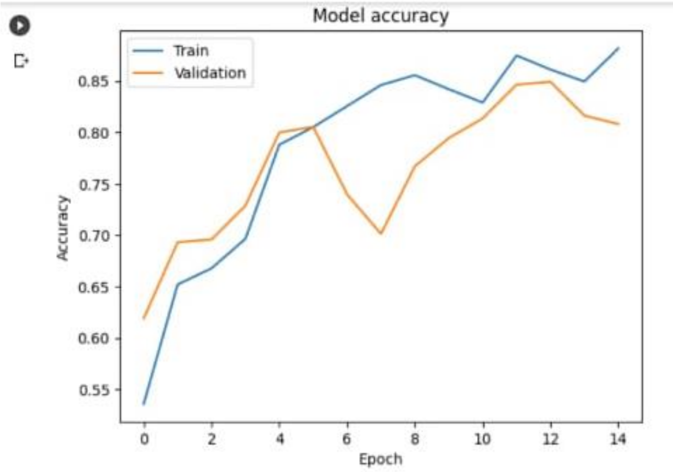
IV. RESULTS AND DISCUSSIONS

The model was trained using the above specified methodologies and we achieved an Accuracy of 80.82% using the customized CNN model.

```
history=classifier.fit(train_ds,validation_data=val_ds,epochs=15)
classifier.save("Custom_CNN_Model.h5")
```

```
Epoch 1/15
46/46 [==============================] - 173s 3s/step - loss: 1.0473 - accuracy: 0.5359 - val_loss: 0.9172 - val_accuracy: 0.6192
Epoch 2/15
46/46 [==============================] - 169s 3s/step - loss: 0.7719 - accuracy: 0.6521 - val_loss: 0.9190 - val_accuracy: 0.6932
Epoch 3/15
46/46 [==============================] - 169s 3s/step - loss: 0.7305 - accuracy: 0.6678 - val_loss: 0.7185 - val_accuracy: 0.6959
Epoch 4/15
46/46 [==============================] - 167s 3s/step - loss: 0.6657 - accuracy: 0.6965 - val_loss: 0.6001 - val_accuracy: 0.7288
Epoch 5/15
46/46 [==============================] - 168s 3s/step - loss: 0.5024 - accuracy: 0.7881 - val_loss: 0.5015 - val_accuracy: 0.8000
Epoch 6/15
46/46 [==============================] - 171s 3s/step - loss: 0.4712 - accuracy: 0.8052 - val_loss: 0.4679 - val_accuracy: 0.8055
Epoch 7/15
46/46 [==============================] - 169s 3s/step - loss: 0.4345 - accuracy: 0.8257 - val_loss: 0.5099 - val_accuracy: 0.7397
Epoch 8/15
46/46 [==============================] - 168s 3s/step - loss: 0.3887 - accuracy: 0.8462 - val_loss: 0.5784 - val_accuracy: 0.7014
Epoch 9/15
46/46 [==============================] - 168s 3s/step - loss: 0.3710 - accuracy: 0.8558 - val_loss: 0.4751 - val_accuracy: 0.7671
Epoch 10/15
46/46 [==============================] - 168s 3s/step - loss: 0.3928 - accuracy: 0.8421 - val_loss: 0.4260 - val_accuracy: 0.7945
Epoch 11/15
46/46 [==============================] - 168s 3s/step - loss: 0.4245 - accuracy: 0.8291 - val_loss: 0.3861 - val_accuracy: 0.8137
Epoch 12/15
46/46 [==============================] - 156s 3s/step - loss: 0.3239 - accuracy: 0.8749 - val_loss: 0.3670 - val_accuracy: 0.8466
Epoch 13/15
46/46 [==============================] - 169s 3s/step - loss: 0.3445 - accuracy: 0.8612 - val_loss: 0.3667 - val_accuracy: 0.8493
Epoch 14/15
46/46 [==============================] - 158s 3s/step - loss: 0.3483 - accuracy: 0.8496 - val_loss: 0.3837 - val_accuracy: 0.8164
Epoch 15/15
46/46 [==============================] - 168s 3s/step - loss: 0.3101 - accuracy: 0.8817 - val_loss: 0.4170 - val_accuracy: 0.8082
```





V. CONCLUSIONS

Many farmers in the coastal region grow arecanut, mostly for commercial purposes. One of the most popular and well appreciated crops right now is arecanut. Experimentation is done out using a 1700-image dataset of high, medium and bad quality arecanuts. The overall accuracy of the Convolution Neural Network is 97 percent, and the Support Vector Machine is 93 percent. As a result, this technology assists farmers in practicing smart farming and making better production decisions by allowing them to take critical preventive and corrective actions on their arecanut crop.

REFERENCES

[1] Mahajan, R. P., Bhagat, R. P., & Mahajan, N. P. (2021). Areca Nut Classification using CNN. International Journal of Computer Applications, 179(36), 23-27.

[2] Devi, R. K., Kumar, D. G., & Prasad, D. (2020). Areca Nut Classification using Deep Learning Techniques. In 2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS) (pp. 1105-1110). IEEE.

[3] Ratanpara, K., & Bhatt, J. K. (2020). Image classification of areca nut using deep learning. In 2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE) (pp. 1-6). IEEE.

[4] Kaur, P., & Kaur, G. (2019). Image classification using convolutional neural networks: A review. Journal of King Saud University-Computer and Information Sciences, 31(4), 397-408.

[5] Suryavanshi, S. S., & Bhaviskar, D. R. (2019). Classification of Arecanut using Convolutional Neural Network. In 2019 International Conference on Communication Information and Computing Technology (ICCICT) (pp. 1-5). IEEE.

[6] Mir, A. H., Asghar, M. N., & Tariq, M. A. (2019). Areca nut detection using machine learning. International Journal of Advanced Computer Science and Applications, 10(6), 274-280.

[7] Nayak, S., & Rao, V. P. (2018). Convolutional Neural Network based areca nut classification. In 2018 International Conference on Recent Innovations in Electrical, Electronics & Communication Engineering (ICRIEECE) (pp. 1-4). IEEE.

[8] Subramanian, M., Srinivasan, S., & Sivaprasad, S. (2018). A novel image classification approach for areca nut detection using convolutional neural networks. In 2018 International Conference on Computer Communication and Informatics (ICCCI) (pp. 1-5). IEEE.

[9] Raj, R. R., & Swarnkar, A. (2018). Areca nut classification using deep convolutional neural network. In 2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA) (pp. 1-5). IEEE.

[10] Singh, A., & Kapoor, G. (2018). Image classification using convolutional neural network: A survey. In 2018 3rd International Conference on Computing Methodologies and Communication (ICCMC) (pp. 714-718). IEEE.