Journal of Applied Data Sciences
Vol. 6, No. 1, January 2025, pp. 1-19


# Improving Classification Accuracy of Local Coconut Fruits with Image Augmentation and Deep Learning Algorithm Convolutional Neural Networks (CNN)

Usman Usman[1,*], Fitri Yunita[2], Muh. Rasyid Ridha[3]

[1,2,3]*Department of Information Systems, Faculty of Engineering and Computer Science, Indragiri Islamic University, Riau Province, Indonesia*

(Received: August 23, 2024; Revised: September 15, 2024; Accepted: October 3, 2024; Available online: December 27, 2024)

**Abstract**

Local coconut varieties must be classified to maintain the quality and genetic diversity of coconuts as the main commodity in Indonesia's largest coconut-producing region. This study introduces a deep learning module for improved classification of coconuts, using color jitter as part of a data augmentation strategy to supplement the existing dataset and utilizing well-known CNN-based models like VGG16 for image analysis, with a focus on the needs of future research. The goal is to improve the classification accuracy of local coconut varieties through deep learning. We investigate both data augmentations and EDA, and we use VGG-16-based CNN models to enhance the classification performance. We used a confusion matrix for the model evaluation, containing metrics like accuracy, precision, recall, and f1-score. Results reveal that a color jitter augmentation model attained a training accuracy of 99.12%, testing accuracy of 97.33%, and validation accuracy of 97.33%. Model exploration using VGG16, on the other hand, improved all three: training accuracy—99.87%, testing accuracy—98.77%, and validation accuracy—98.97% average F1-score: 99%. Our research contributes massively to providing the best automatic classification method that will benefit and help farmers shorten their jobs while promoting economic growth in trading effectively across Indonesian regions. Its novelty is in combining image augmentation and CNNs, concerning the VGG16 model, showing better.

*Keywords:* Local Coconut Fruit, CNN, Image Augmentation, EDA, VGG16

## 1. Introduction

Artificial intelligence (AI) is a discipline within computer science that includes various models or methods such as backpropagation [1] and the double-layered perceptron algorithm [2]. It is an artificial representation of the human brain [3], [4]. Researchers have widely employed artificial intelligence in research, including the detection of rice quality [5]. One of the classification methods is CNN, which is capable of extracting characteristics from input images and then changing the image dimensions without changing the image characteristics [6]. Indonesia is one of the largest coconut producers in the world [7], making it one of the world's largest coconut areas and being referred to as the world's coconut belt [8]. According to Mawardati [9], coconut is a very profitable plantation commodity. Almost every part of the coconut has its advantages.

To maintain the quality of coconut products, improve production efficiency, and maintain genetic diversity, accurate classification of local coconut varieties is essential. The manual classification of local coconut varieties often encounters challenges due to its susceptibility to personal error and human error [10] and its unreliability for large volumes. Therefore, the development of accurate and efficient automatic classification methods is crucial.

a spike-based backpropagation training methodology for popular deep SNN architectures [11]. These studies have not fully addressed several issues. For example, the quality and variety of local coconut image datasets are still limited, which can hinder the performance evaluation of AI models. The NMC is developed based on the mean value of the samples [12] naive Bayes classifier The learned parameters were used to make a graph of the relationships between diseases and symptoms, and the knowledge graphs that were made were tested and proven to be correct [13]. The study

*Corresponding author: Usman Usman (usmanstmkom@unisi.ac.id)
DOI: https://doi.org/10.47738/jads.v6i1.389

 ISSN 2723-6471


of the NMC method is applied to a robust feature-based Automated Multi-view Human Action Recognition System [14], and research on Robust Hybrid Classification Methods and Applications [15] Researchers have used CNN in their studies to determine the maturity of oil palm fruit [16], and classifying seedling quality images to increase coconut production also showed an accuracy of 86.66% [17].

Researchers surveyed enhancing image data for deep learning [18], while another study enhanced CNN's performance in classifying liver lesions using GAN-based synthetic medical images by employing image augmentation [19]. Researchers [20] focused on making Deep Convolutional Generative Adversarial Network (DCGAN-CNN models with physical constraints more accurate at predicting porosity in additive manufacturing processes. Geology and seismic fault interpretation in some regions of the North Sea and Southwest Barents Sea can also utilize CNN [21], while denoising [22] and color balancing techniques for underwater images and river water segmentation [23] can also benefit from its application.

Health professionals can utilize CNN for various purposes, including diagnosing Parkinson's disease using CNN [24], enhancing COVID-19 detection from CXR [25], and approximating satellite images [26]. CNN can be combined with other methods, like the Rotation-Invariant Coordinate Convolutional Neural Network RIC-CNN for rotation-invariant coordinates [27]. It is also used to classify batik patterns with the help of CNN and the VGG16 architecture, which produces validation data that is 82.56% accurate [28].

The previously mentioned research describes the results of model testing on CNN's ability to categorize scans and other types of images taken with ordinary cameras. In general, the results show that CNN can categorize images. Like previous research, this study uses CNN architecture to perform classification on a data set of coconut fruit images.

This research aims to develop a more accurate classification model by combining image augmentation techniques with several models from CNN deep learning algorithms to overcome the limitations of the data set and improve accuracy. This research specifically concentrates on the classification of local coconut varieties, which have different characteristics. Identifying the unique characteristics of local coconut varieties will help in the preservation and development of superior varieties. In addition, this research aims to add classifications based on variety, maturity level, and quality.

**2. Related Works**

## 2.1. Convolutional Neural Network (CNN)

CNN is a subset of deep learning and artificial neural networks specifically designed for processing images. CNN models [29] have revolutionized the fields of computer vision and image processing, including facial recognition and object detection, image classification, and image segmentation [30]. The CNN architecture consists of many interconnected mathematical layers, each of which generates complex features [31]. CNN can identify objects better than other models because it can analyze visual patterns [32], [33]. There are several main layers in CNN that extract features and information from images, including the convolution layer, pooling layer, ReLU activation function layer, fully connected layer, and softmax function in the classification layer [34].

## 2.2. Exploratory Data Analysis (EDA)

EDA is the initial stage of data analysis that serves to examine and summarize the main characteristics of a data set [35], [36]. EDA's purpose is to identify the distribution structure of relationship patterns in the data before creating predictive or inferential models. The stages involved in EDA [37] start with collecting the field survey results, which consist of unprocessed raw data, followed by evaluating the dimensionality, i.e., the number of rows and columns in the dataset, and then selecting the categorical type at this stage. Cleaning involves detecting outliers, correcting inconsistencies, and summarizing using descriptive statistics of frequency distribution through metric calculations, analyzing the frequency distribution of categorical columns, and then displaying visualization results such as histograms, boxplots, scatterplots, and bar plots. EDA analysis [38] is very important in this situation because it can show the structure of the data, help understand its shape, find problems, mistakes, and inconsistencies that need to be fixed, give useful first insights for model development, and improve the quality of the analysis.

 

## 2.3. Image Augmentation with Color Jitter

Image augmentation [39], [40], [41] plays an important role when training CNN models as it creates artificial variations of the original images included in the training dataset. By effectively expanding the dataset using this approach, we can improve the overfitting performance [42], mitigate the generalization ability, and reap other benefits from images, thereby enhancing the accuracy and reliability of the trained images. Therefore, they should apply common techniques such as rotation, vertical cropping, translation, brightness, darkening, noise addition, contrast modification [43], and image filtering, where image augmentation applied to CNN ultimately results in more accurate and reliable models.

In this study, random color jitter [44] is used as an image augmentation technique [45] to change the color properties of images so that our dataset has a lot of different types. This should stop overfitting and allow better generalization by setting parameters for each property, such as hue saturation brightness.

## 2.4. Model VGG16

VGG16 is one of the influential CNN models [46], [47] used primarily for image recognition. It was designed by the Visual Geometry Group (VGG) at the University of Oxford, in 2014 during the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) competition, which won many awards due to its efficient learning mechanism based on the use of small filters across a deeper architecture capable of learning complex hierarchical representations captured in different levels of depth of representation learned through pre-training weights [48] derived from the imagenet dataset [49], the VGG16 model can be seen in figure 1. on which VGG16 was initially trained at the time of deployment, thus achieving superior results compared to other shallow alternatives such as AlexNet [50], LeNet [51], etc.

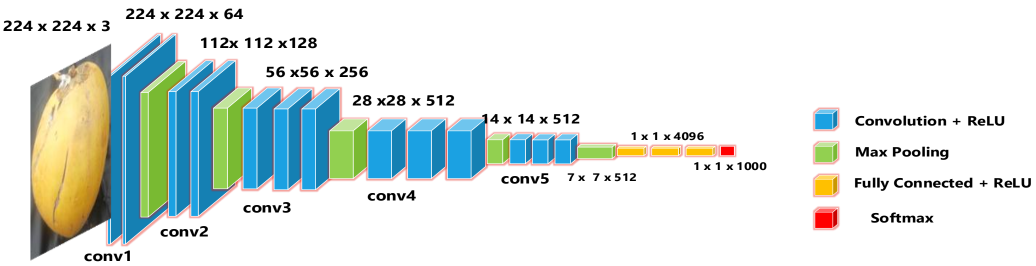


**Figure 1.** Model VGG16

## 2.5. Confusion Matrix

The confusion matrix is a table used to evaluate the performance of classification models, useful for providing a comprehensive summary of how good the model is. The confusion matrix [52] works to compare model predictions with actual values, the actual value of the data. The confusion matrix has 4 quadrants, including accuracy, precision, recall, and f1-score [53], [54], which contain the categories of true positive (TP), true negative (TN), false positive (FP), and false negative (FN).

$$\text{Accuracy} = \frac{\text{TP+TN}}{\text{TP+TN+FP+FN}} \tag{1}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP+FP}} \tag{2}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP+FP}} \tag{3}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP+FP}} \tag{4}$$

Accuracy is the count of correct predictions (both true positives and true negatives) divided by the total number of predictions, while precision refers to the ratio of true positives to all positive predictions, recall means the ratio of true positives to all relevant actual observations, and f1-score computes the average based on harmonics between precision and recall. This yields one metric which considers how well both are met.

ISSN 2723-6471


## 3. Methodology

### 3.1. Dataset Analysis

This study relies on a dataset of local coconut fruits sourced from field surveys and coconut plantations in several villages in the Indragiri Hilir Regency. Three main classes dominate the classification of this dataset, namely good, intermediate, and bad consisting of 3450 samples. Each category is differentiated through various characteristics such as skin, size, and fruit shape.

The morphological characteristics of coconuts cover a wide range of coconut fruits which include the color of the rind, shape, and size of the midrib. In general, all coconut fruits were classified into three shapes: elliptical, oval, and round. As for the color of coconut fruits, they can be yellow, green, orange, brown, and red. The unripe fruits of tall coconut varieties tend to be green in color from pale green to deep greenish bronze with an occasional blonde color seen in Pacific or coastal areas. Detailed information on the morphological characteristics of coconut fruits can be seen in table 1.

**Table 1.** Categories and morphological characteristics of local coconut fruits

| No | Aspect | Class Good | Class Intermediate | Class Bad |
|---|---|---|---|---|
| 1 | Skin Color | Skin color, Uniform golden brown or dark brown | uneven skin color, namely light brown and dark brown or slightly green and even orange | darkening and worsening of skin color |
| 2 | Fruit Shape | Symmetry, round, seamless | Slightly oval, within normal parameters, less smooth in some parts, and blotches in some parts | Oval, very irregular, dented or deformed, rough or wrinkled texture, |
| 3 | Fruit Size | > 1.5 kg | 1.0 kg - 1.5 kg | < 1.0 kg |
| 4 | Volume and Water Content | Large volume, high moisture content | Medium volume and water content | Small volume, low water content |

Classification of coconuts and varieties in most cases requires attention to measuring and capturing their characteristics. A standard color scale is used for skin color measurement to ensure uniformity. Fruit weight and volume are used to evaluate fruit size. Thicker coconuts are generally larger than thinner ones. This can be seen from the diameter and circumference of the fruit. Therefore, a heavy coconut will feel denser and more compact when grasped than a thin coconut. This means the structure has a lot of pulp and water content. Generally, coconuts that are said to have thicker skins tend to be heavier as the additional protection provided by the skin layer adds to the total weight of the structure. This quantitative data was then applied to a classification technique using a model that allocates coconuts into the most appropriate category.

For analysis purposes, the researcher divided the base dataset of 3450 samples into three parts: for training data, the division of class good, intermediate, and bad is 750 samples each, or about 70%; for testing and validation data, the division of class good, intermediate, and bad is 200 samples each, or a ratio of 70% for training data, while for testing data, 15%, and validation data, 15%. This division allows the model to be properly trained, tested, and validated for maximum efficiency.

### 3.2. Research Steps

This research employs a variety of CNN model combinations to enhance accuracy, and it is structured into five main stages. The first stage involves conducting field surveys and preprocessing data to collect local coconut fruit datasets. This process involves identifying and collecting a variety of local coconut fruit varieties, taking into account factors such as color, texture, type, maturity level, and quality. The second stage is developing image augmentation techniques with color jitter, such as cropping, rotation, flipping, and both horizontal and vertical. The third stage involves developing a CNN model. At this stage, what is done is to design a CNN architecture for local coconut fruit classification with dataset characteristics, considering the number of layers and layer type (convolutional, pooling, or fully connected). The research stages can be seen in figure 2. This research uses ReLU and softmax activation functions. Relu activation functions are used to learn more complex feature representations, while Softmax is used to classify inputs into various classes.

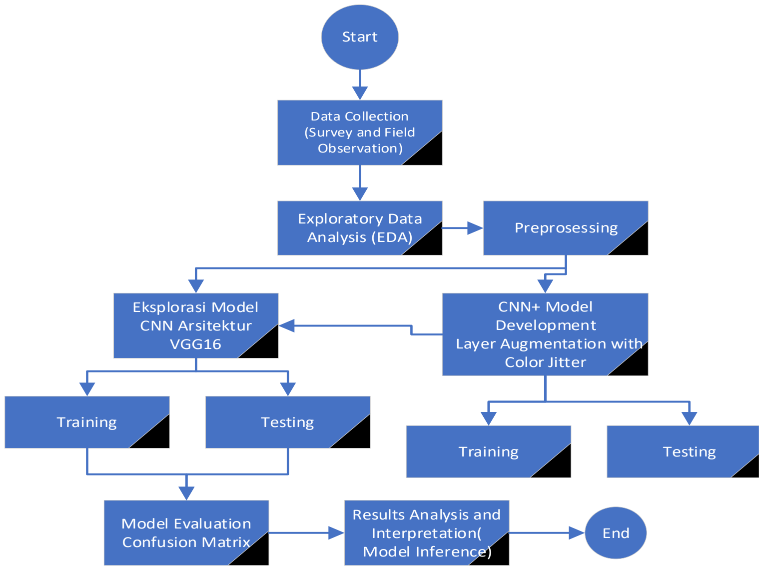


**Figure 2.** Research steps

The process involves initializing the model weights, determining parameters such as numerical filters, kernel size, activation function, and optimizer, assessing the model's performance with various parameters and configurations like configuration of learning rate, batch size, and epoch count, and identifying areas that require improvement in the model architecture, pre-processing techniques, and data augmentation. The fourth stage utilizes a sequential model to combine CNN and augmentation data with color jitter. In the fifth stage, in terms of increasing accuracy, the researcher explores the CNN architecture. In this research, the researcher explores the CNN architecture using the VGG16 model. After training and validating the model, the final stage executes interpretation analysis and inference models, generating predictions on fresh data.

## 4. Results and Discussion

### 4.1. Exploratory Data Analysis (EDA)

This stage aims to describe the distribution of classes or labels on the training dataset, which contains data about local coconuts. For data visualization using bar charts and pie charts, first, the researcher creates a subplot with two columns, type: XY, to show the cartesian graph for the bar chart, where x is the label that is counted based on the number of occurrences and y is the value of the number of occurrences of each label. The labels represent the counted labels, while values represent the number of occurrences for each label. The marker is used to set the pie slice's color, and the border is gray with a width of 3. The pull function creates an explosion effect on the first pie slice. Figure 3 divides the distribution of coconuts in training into 70% for the good category and 30% for the bad category.

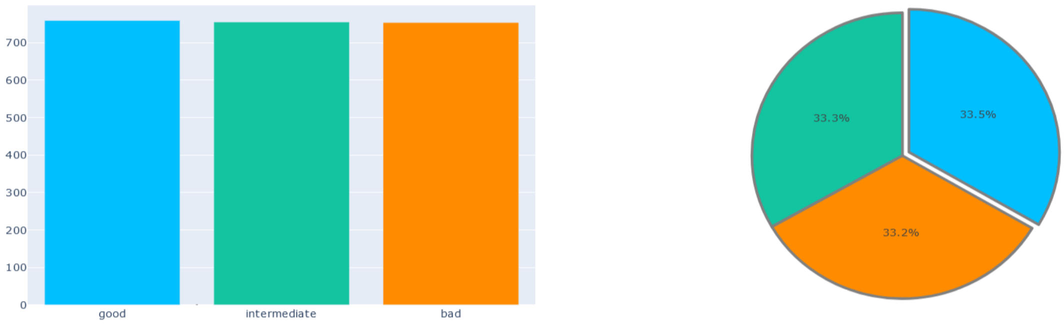


**Figure 3.** Distribution of coconuts in the training

The next step is to extract and estimate the average height and width of the image contained in the column in the form of height mean and width mean. The box plot of Image Dimensions can be seen in figure 4.

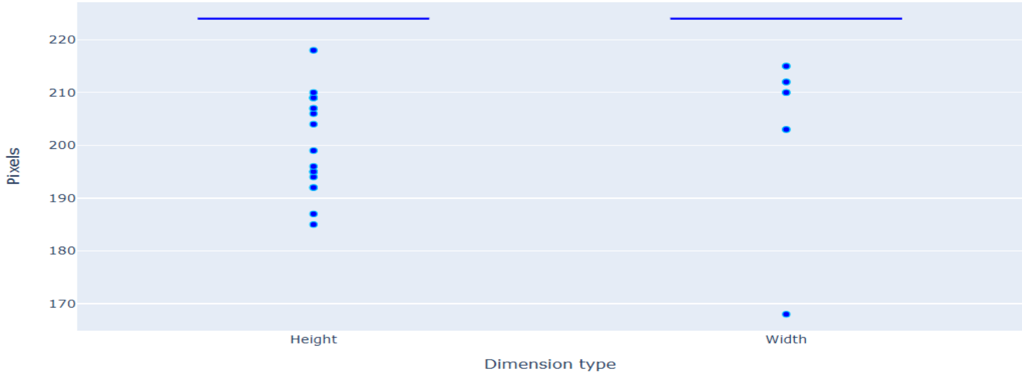


**Figure 4.** Boxplot of image dimensions (rain_df[dimensions].apply(lambda x: x[0]):)

Apply the lambda function to each element in the dimensions column, where x[0] takes the height value and width value, mean(), and Calculate the average of the extracted height and width values. We will obtain the average from these results to illustrate the distribution of the image's height and width in the dimensions column of the train DF data frame, which we will then visualize as a boxplot in figure 4. The dataset's maximum pixel value is 224 x 224.

After calculating the average height and width values, the next step is to create a function that displays several images from the dataset based on their labels. The plot_images function will present the images as a grid, labeled and sized for each image. The plot images function extracts image data from the train_df data frame, simplifying the EDA process and helping to understand the image data's characteristics. In the average calculation process, researchers also filter image files in the train_df dataset based on label and image file size. Based on these findings, we need to perform random rotation augmentation and zoom on images that exhibit a tendency to tilt to the right or left, appear cut off, or exhibit extreme brightness or darkness, as illustrated in figure 5. We display an image of the coconut fruit shape.


**Figure 5.** Shape image of coconut

## 4.2. Data Prepossessing

The purpose of the prepossession stage is to prepare images and labels for model training or evaluation. Data flows typically use the preprocess_image function to process images and labels before feeding them into the model. This function can be used in Python's TensorFlow function map or when creating a data set. The preprocess image function performs essential steps in image and label preprocessing, including reading, decoding, resizing, normalizing images, and converting labels to numerical formats. This helps to prepare the data for model training or evaluation after performing the preprocess image function. Next, we will look at the augmentation results using color_jitter. To perform color_jitter, we need to set channel = 3.

When performing deep learning model construction for classifying coconut images, image preprocessing and augmentation functions are applied for the purpose of preparing the images for use in model training. The preprocess_image function starts considering the files residing in the file_path, where it first scans the image using image_read_file in tf.io. The next step performed here is image conversion, where the read image is converted into a tensor by tf.image.decode jpeg, where in the n_channels parameter, the number of channels that appear is three (default is 3 for RGB). The obtained image is then maintained within the acceptable range or standard of an input image for deep learning models in general except the dimension, 224 x 224 pixels, through tf.image.resize. Proceeding to the next step, all pixel values in the image are reduced to a pixel intensity range of [0, 1] to maintain uniformity of data processing as in the previous function using tf.cast and previously divided by 255.0. Based on this experience and in this context, the so-called image labels that initially looked like good, intermediate, or bad evolved into a number structure, such as the words 0 is good, 1 is intermediate, and 2 is bad, by using tf.where. This function is designed in such a way that it can give back the transformed image and the numerical value of the image.

To improve the diversity and quality of the training data, the last color_jitter is used, which makes the image lighter or darker due to the random brightness of the image tf.image.random_brightness, tf.image.random_contrast, tf.image.random_saturation, and tf.image.random_hue. In conjunction with this function, this image has been discolored. With respect to the task of image augmentation for training data, the augment_train_image function also contributes to this by performing changes such as flipping the image from the left side to the right side and vice versa, and from the top side to the bottom side by using tf.image.random flip left left right and tf.image.random flip up down. Finally, color jitter is added using the color_jitter function.

To prepare the validation data, the augment_val_image function was developed and can incorporate augmentation if needed, but currently, it only returns the image and label without any transformation. At the same time, this augmentation approach is extended through the layers_augmentation object, which uses the Keras Sequential API. An example of augmentation is where some sequential operations are applied to the image. For example, operations like RandomFlip to make the image switch sides, RandomRotation to adjust the angular position of the image, and RandomZoom to slightly change the height by a certain amount are performed to maintain diversity in the training set. This entire set of operations is embedded into the Keras model and used during training to increase the flexibility of the model obtained as a result of training on various data

In the augmentation process, the researcher performed a visualization function, which is useful for comparing the original image with the augmented image. By displaying both images side by side, it is easy to evaluate the effect of the applied augmentation and ensure that the augmentation process works as expected, which helps in verifying and understanding the effect of color augmentation on images before they are used in machine learning models. In figure 6, an example of an augmented image from the original image to the augmented image is shown.

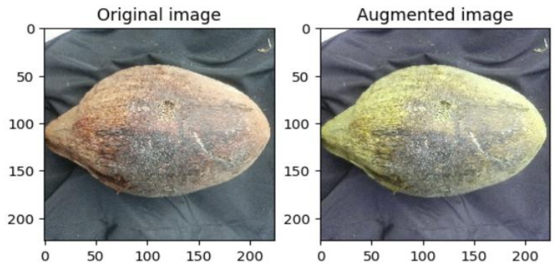


**Figure 6.** Image augmentation

We perform autotune and batch_size to automatically enhance the dataset's performance, using autotune as a constant. The system automatically adjusts the configuration to maximize speed and efficiency in dataset processing. The batch size parameter determines the size of the batch for model training. A batch is several data samples processed together in one training iteration. A batch size of 32 signifies that each training iteration will process 32 data samples simultaneously, thereby influencing both training speed and memory usage. In the augmentation process, researchers set random zoom and random rotation with a small value so that the resulting image looks clear and the model can recognize the image pattern properly. After the augmentation process, researchers prepare the dataset to parallelly optimize the preprocessing and training processes. This is useful because it makes training the model faster, especially when using a Graphics Processing Unit (GPU).

 ISSN 2723-6471 

To create and manage the training, validation, and testing datasets, it is necessary to create an array of images and labels, then apply a map, which preprocesses and augments each element of the dataset; shuffle, which randomizes the order of elements in the dataset; batch groups dataset elements of a specified size; and prefetch increases throughput by loading data into memory while the model is being trained. To iterate through the training dataset m, the researcher uses a loop to retrieve and print information about the batch of images and labels.

In addition to examining the individual channels of the images in the dataset, which is useful for understanding and verifying the image preprocessing that has been applied, viewing these image channels can help in debugging and ensuring that the data is processed correctly before it is used for model training, where the label is 0 for a bad image and 1 for a good value image. With Batch of images shape: (32, 224, 224, 3), Batch Size: 32, Labels: [1 0 2 2 1 1 1 2 0 2 0 1 0 1 2 0 2 2 0 2 0 2 1 1 2 0 1 2 2 0 1 2]. Displaying images from a single batch of training datasets aids in verifying preprocessing and augmentation and also ensures correct data loading. Figure 7 is an image of the results of iterating through the train data.

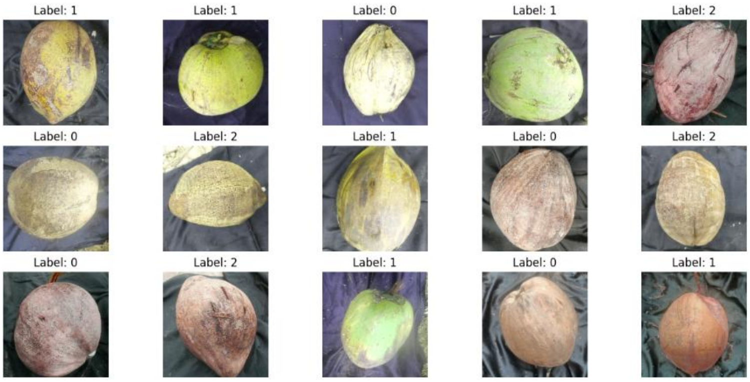


**Figure 7.** Train data iteration results

## 4.3. CNN Model Development

During the model development process, we will build a CNN model and train it on data that has undergone augmentation layers, specifically random zoom and random rotation. We have designed this model to be able to take images as input. The images should have dimensions of 224x224 pixels, and they should have three color channels (RGB). The structure of the model consists of an input layer that takes in images with dimensions of 224x224 pixels and three-color channels. Additionally, the model comprises four convolutional layers, each having a kernel size of 3×3, stride length equal to one, padding as valid, and activation function as Rectified Linear Unit (ReLU). These layers have filters numbering 32, 64, 64, and 128, respectively. There is a two-dimensional max-pooling layer following every convolutional layer to reduce spatial dimensions; it has a kernel size of 2×2 and a stride length equal to two. The flattened layer takes its previous convolutional layer output and converts it into a one-dimensional vector, while the dense layer—the first fully connected part of our architecture—uses Rectified Linear Units activation with a dropout rate of 0.2 to prevent overfitting among its 128 neurons. Finally, the output layer predicts class probabilities using a single neuron in the sigmoid function. Table 2 displays a CNN model that utilizes augmented data and incorporates color jitter.

**Table 2.** CNN Model of Augmented Data

| No | Layer (Type) | Output shape | Parameter |
|---|---|---|---|
| 1 | Conv2D | (None, 222, 222, 32) | 896 |
| 2 | MaxPooling2D | (None, 111, 111, 32) | 0 |
| 3 | Conv2d_1 | (None, 109, 109, 64) | 18.496 |
| 4 | Max_pooling2d_1 | (None, 54, 54, 64) | 0 |
| 5 | Conv2d_2 | (None, 52, 52, 64) | 36.928 |
| 6 | Max_pooling2d_2 | (None, 26, 26, 64) | 0 |
| 7 | Conv2d_3 | (None, 24, 24, 128) | 73.856 |

 ISSN 2723-6471


| | | | |
|---|---|---|---|
| 8 | Max_pooling2d_3 | (None, 12, 12, 128) | 0 |
| 9 | flatten (Flatten) | (None, 18432) | 0 |
| 10 | dense (Dense) | (None, 128) | 235.942.4 |
| 11 | dropout (Dropout) | (None, 128) | 0 |
| 12 | dense_1 (Dense) | (None, 3) | 129 |
| | **Total params** | | **248.972.9** |
| | **Trainable params** | | **248.972.9** |
| | **Non-trainable params** | | **0** |

In setting up and building deep learning models, the application of the binary cross-entropy loss function is essential for binary classification tasks such as classifying images into two classes. One of the first steps taken is to set the learning rate to 0.01. This is intended to speed up the learning process of the model. However, it should be noted that if the step size is too large even after convergence has been achieved, the model may become oscillatory or non-convergent. Therefore, another optimization algorithm used is Adam, with a learning rate that can be adjusted to suit the situation and the architecture to be solved.

The adjustment of hyperparameters such as learning rate, batch size, optimizer type, etc. is at the core of understanding the problem in deep learning studies. The selection of the learning rate is done by looking at previous experiments that proved to converge well at that value. The selection of batch size is done by considering memory constraints and training speed. In general, quantitative procedures such as grid search or random search are used to find the best set of hyperparameters. Gradient descent is more sophisticated in training as the Adam optimizer is considered more appropriate as the learning rate can be changed during the training process, leading to smoother and better convergence. As we know, Adam can be aggressive when optimizing deeper networks with convolutional layers and processing large datasets full of interdependencies.

In addition, the parameter accuracy is defined to record the occurrence of the correct label given by the model during training. To further enhance the training process, several such callbacks have been designed. These callbacks allow you to track and improve the modeling in some way. One of the callbacks used is EarlyStopping, which serves to stop training if the validation metric shows no improvement to prevent overfitting. Model Checkpoint is used to save the best model from all the models trained by taking the best measures seen from the metrics specified during the training process. There is also ReduceLROnPlateau that helps accelerate the model plateau by appropriately reducing the learning rate if the model shows no improvement in the performance metrics. Hence, the effectiveness and efficiency of the training process with the help of such callbacks are greatly improved.

The training stage of the model specifies 200 epochs, during which the fit method of the hard model trains the model using the provided data. This function will iterate through the training dataset covering the designated epoch count, apply the specified callback, and monitor performance on the validation dataset. Table 3 is the result of epoch 22, which obtained the highest accuracy value of 99.47% and validation accuracy of 97.74%. Table 3 only displays 6 epochs, namely from Epoch 1 to Epoch 3 and 20 to Epoch 22.

**Table 3.** Accuracy and validation accuracy for CNN model and augmentation data

| Epoch | Loss | Accuracy | Val loss | Val accuracy |
|---|---|---|---|---|
| 1 | 1.9337 | 0.7209 | 5.5842 | 0.5298 |
| 2 | 0.4089 | 0.8607 | 5.0268 | 0.5257 |
| 3 | 0.2947 | 0.8977 | 5.5604 | 0.5257 |
| .. | ..... | .... | .... | ......... |
| 20 | 0.0230 | 0.9916 | 0.2464 | 0.9692 |
| 21 | 0.0367 | 0.9881 | 0.1779 | 0.9856 |
| 22 | 0.0163 | 0.9947 | 0.1485 | 0.9774 |

The following is a graph plot image for training and validation loss and accuracy, where there are 2 graphs, namely training and validation loss, which has a minimum learning rate of 0.01, and accuracy of training and validation graphs. Training and Validation Graph of CNN Accuracy can be seen in figure 8.

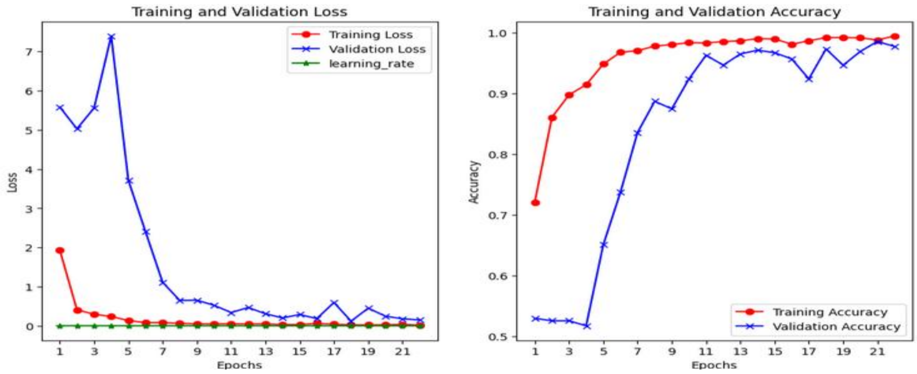


**Figure 8.** Training and validation graph of CNN accuracy

The model evaluation results suggest the model successfully obtained an accuracy of 99.12% with a loss of 21.46% for train data, testing accuracy of 97.33% with a loss of 5.52%, and validation accuracy of 97.33% with a loss of 11.82%. This is a good result. The value of the confusion matrix obtained was Class good: True Positive (TP): 151, False Positive (FP): 1, True Negative (TN): 323, False Negative (FN): 11. Class intermediate: True Positive (TP): 162, FP: 8, TN: 316, FN: 0. Class bad: T): 160, FP: 4, TN: 320, FN: 2. Moreover, we also looked at the classification model performance using the AUC-ROC analysis matrix, its ability to differentiate one class from another, or how well the classification model is interacting with itself by assessing the AUC with criterion thresholds of true positive rate (TPR) and false positive rate (FPR). For AUC-ROC analysis matrix: All Classes In all classes, it is observed that the model allows capturing the performance of all classes irrespective of how they classify the curve in that all the classes are scored with 1.00 Andrews, 2008. This means that the model does not fail to make a correct prediction for every class. There is no overlapping in the class predictions of the model. While the ROC Curve Reaches Towards the Upper Left-Hand Side: There are ROC lines for all classes that tend to reach the top left-hand side of the graph, which therefore depicts that the TPR is high while the FPR is fairly low. This means that the model barely misclassifies individuals into positive classes when they shouldn't be.

Low FPR and high TPR: From the above graph, we observe that the FPR is almost negligible (near zero) for all three classes, which are very few negative predictions by the model. Also, the TPR, on the other hand, is very high, at almost 1 for the majority of the classes, which implies that the number of positive examples that were predicted correctly for most of the classes was high (true positives). Predict would be the performance for three classes, with the value of AUC 1.00 indicating a high performance of the model. The graphical image of the confusion matrix and AUC-ROC can be seen in figure 9.

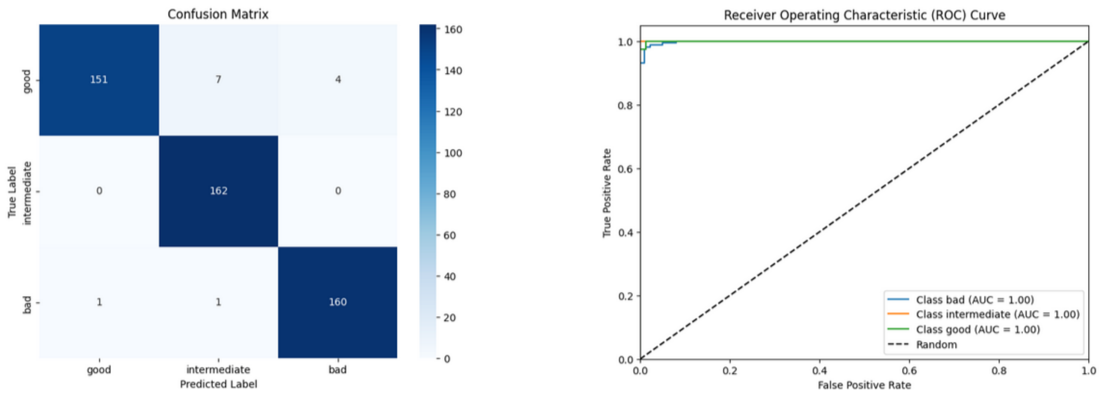


**Figure 9.** Confusion matrix and AUC-ROC curve

To evaluate the performance of the classification model, the first step is to store and evaluate the model. The following are the classification results based on precision, recall, and f1-score: For the good class, the accuracy precision is 99%, recall is 93%, and f1-score is 96%; for the intermediate class, the accuracy precision is 95%, recall is 100%, and f1-score is 98%; while for the bad class, the accuracy precision is around 98%, recall is around 99%, and f1-score is around 98%; accuracy AVG is around 97%; macro AVG is around 97%; and weight AVG is around 97% for f1-score. The performance of the class can be seen in table 4.

**Table 4.** Performance of class

| Class | Precision (%) | Recall (%) | F1-Score (%) |
|---|---|---|---|
| Good | 99% | 93% | 96% |
| Intermediate | 95% | 100% | 98% |
| Bad | 98% | 99% | 98% |
| Accuracy | | | 97% |
| Macro avg | 97% | 97% | 97% |
| Weight avg | 97% | 97% | 97% |

## 4.4. Exploring CNN with Architecture Model VGG16

The result of the development of CNN models with augmentation layers is the exploration of CNN using the VGG16 model. The researchers used max pooling to reduce the spatial dimensionality of the coconut case local data. In the feature map, max pooling occurs when the maximum number of units in each patch is calculated. This strategy increases model tractability and stops over-fitting without sacrificing relevant data.

Max pooling has advantages, one of which is the elimination of redundant information by emphasizing the most defining characteristics. This means that, instead of averaging the map, max pooling extracts the strongest features important for classification from the area. Max pooling affects the size of the feature map by reducing its width and height, which also helps address the memory and processing requirements of the paper, thus ensuring a faster model in both training and application. Max pooling also achieves some rotational invariance in the internal representation of the model, in the sense that if an image is shifted gently, the internal model will still recognize certain patterns. max pooling while capturing important features and reducing the possibility of overfitting. There are many other applications of max pooling, but generally, it is only used in CNN structures such as VGG16.

To limit overfitting, max pooling facilitates compression of the parameters associated with the model, which in some cases is critical on small datasets as it reduces overfitting bias and guarantees that only the most useful information is retained, thus minimizing noise. Max pooling is most effective when applied in conjunction with other measures to address overfitting, such as dropout and l2 regularization. Max pooling in the VGG16 model admin max pooling: V max-pooling spatial-five-input matrix-sized (2242563 gaps) constraints, showing pollution level dependence. This relates to the dense layers of the fully connected model, where the first dense layer consists of 198 and 128 neurons activated by ReLU, followed by another dense layer (3 neurons, activation=softmax) for multi-classification. This combination of strategies helps maintain the performance of the model in terms of other local coconut data. Table 5 presents the sequential model for multi-class classification.

**Table 5.** Model Sequential VGG16

| No | Layer (Type) | Output shape | Parameter |
|---|---|---|---|
| 1 | Vgg-16 (Functional) | 7, 7, 512 | 14714688 |
| 2 | max_pooling2d | 3, 3, 512 | 0 |
| 3 | flatten_1 (Flatten) | 250.88 | 0 |
| 3 | Dense_1 | 198 | 4.967.622 |
| 4 | Dense_2 | 128 | 25.472 |
| 5 | Dropout | 128 | 0 |
| 6 | Dense_3 | 3 | 129 |
| | **Total params** | | **19.707.911** |
| | **Trainable params** | | **4.993.223** |
| | **Non-trainable params** | | **147.14.688** |

ISSN 2723-6471


In the above model, we use flatten, the value on the test is better, and the learning rate that has been set is 0.01, whereas in the early stopping callback process (monitor=val loss, patience=4, restore_best_weights=true) and in reducelronplateau (monitor=val loss, factor=0.2, patience=2, min_lr=1e-7), the training process is done with the number of epochs = 200, and on reducelronplateau (monitor=val loss, factor=0.2, min_lr=1e-7), the training process is done with the number of epochs = 200, patience=2, min_lr=1e-7), table 6 is the result of epic training and validation of accuracy and loss for VGG16. The results of the testing process of the CNN development model and the augmentation layer, where the test results are used as training data for the VGG16 model, show that the accuracy value increases by around 99.91%, with a value accuracy of around 98.77%, resulting in very good accuracy in recognizing local coconut fruit images. Table 6 only displays 6 epochs, namely from Epoch 1 to Epoch 3 and 20 to Epoch 22.

**Table 6**. Accuracy and loss for CNN model VGG16

| Epoch | Loss | Accuracy | Val loss | Val accuracy |
|---|---|---|---|---|
| 1 | 1.2808 | 0.7743 | 9.7145 | 0.3347 |
| 2 | 0.3216 | 0.8880 | 7.5374 | 0.5195 |
| 3 | 0.1702 | 0.9352 | 14.0213 | 0.4661 |
| .. | ..... | .... | .... | ......... |
| 20 | 0.050 | 0.9987 | 0.454 | 0.9877 |
| 21 | 0.0062 | 0.9987 | 0.0437 | 0.9877 |
| 22 | 0.0045 | 0.9991 | 0.0430 | 0.9877 |

When the model evaluation is obtained, it shows that the trained model achieves an accuracy of 99.87% and a loss of 1.40% for training data, a testing accuracy of 98.77%, a loss of 10.55%, and a validation accuracy of 98.97% and a loss of 4.03%. This is a very good result. The value of the confusion matrix obtained was Class good: TP: 158, FP: 2, TN: 322, F): 4. Class intermediate: TP: 161, FP: 3, TN: 321, FN: 1. Class bad: TP: 161, FP: 1, TN: 323, FN: 1. VGG16 Graphics can be seen in figure 10.

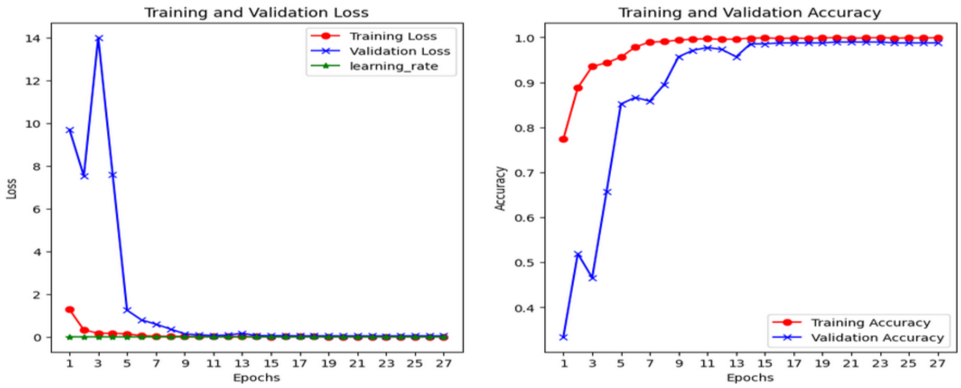


**Figure 10.** VGG16 graphics

The ROC curve is an important tool for measuring the efficiency of classification models. In the X-axis, the FPR is taken, which is called the proportion of negative cases that were incorrectly classified as positives, and bargained by the Y, the TPR, which is the proportion of positive cases that are categorized as positive. Focusing on the curves of this graph, are for class bad (blue), intermediate (orange), and good (green) classes with AUC curve scores equal to all at 1.00. This implies that the model can distinguish all classes successfully and accurately. The curve that is located on the left-hand side of the graph is a representation that a true positive rate has been achieved and, at the same time, a false positive rate is too low. This clear differentiation is represented in the graph with the dashed horizontal black line where the effectiveness is that of the random model (0.5 AUC), yet all the curves in this graph even go as far as to the opposite of the random line where performance is better than any other, hence no average. On the whole, the model's perfect execution is at classifying and defining all classes (bad, intermediate, and good) reversely, which is performed with AUC = 1.00 for all classes, showing the perfect zero error when classifying each class. The graphical image of the confusion matrix and AUC-ROC can be seen in figure 11.

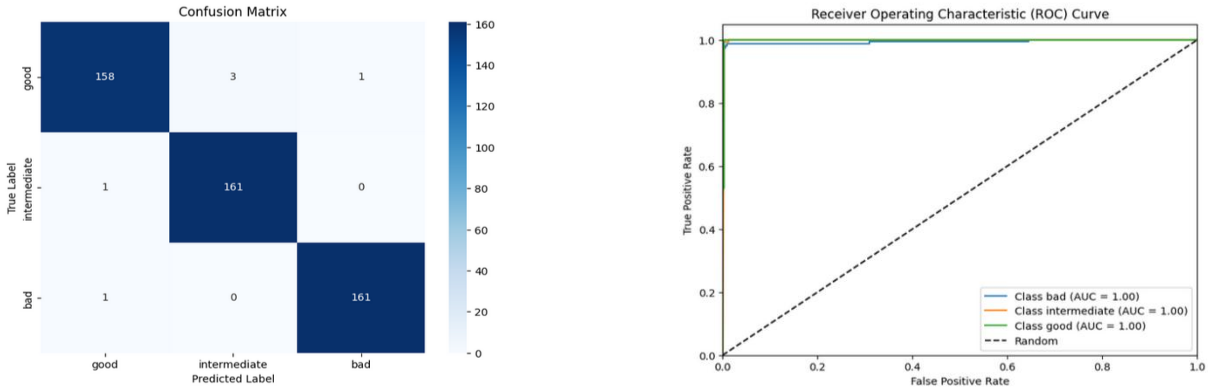


**Figure 11.** Confusion matrix and AUC-ROC curve

To evaluate the performance of the classification model, the first step is to store and evaluate the model. The following are the classification results based on precision, recall, and f1-score: For the good class, the accuracy precision is 99%, recall is 98%, and f1-score is 99%; for the intermediate class, the accuracy precision is 98%, recall is 99%, and f1-score is 99%; while for the bad class, the accuracy precision is around 99.9%, recall is around 99%, and f1-score is around 99%; accuracy AVG is around 99%; macro AVG is around 99%; and weight AVG is around 99% for f1-score. The performance of the Class of VGG16 can be seen in table 7.

**Table 7.** Performance of class

| Class | Precision (%) | Recall (%) | F1-Score (%) |
|---|---|---|---|
| Good | 99% | 98% | 98% |
| Intermediate | 98% | 99% | 99% |
| Bad | 99% | 99% | 99% |
| Accuracy | | | 99% |
| Macro avg | 99% | 99% | 99% |
| Weight avg | 99% | 99% | 99% |

After seeing the confusion matrix and the test data evaluation, the results of exploratory training with VGG-16 produce a better accuracy score than the previous model.

## 4.5. Comparison of VGG16 Models

This research also compares the accuracy between VGG16 and other models; the researchers chose the Xception model as a comparison model on the grounds of computational efficiency. Xception is an efficient convolutional neural network (CNN) architecture and is used primarily for image classification and object detection tasks. Lowers the computation cost, meaning that models can be deployed in environments with limitations like mobile devices or edge computing. Enhances performance using a smaller number of parameters while still maintaining precision. Image Classification: Categorization of images into different classes based on the properties that have been extracted from the image. Transfer Learning: Serves as a guide for reassembling them on other data sets without having to start training them from square one. Backbone for Object Detection: Comprehends features of architectures used to build more advanced object detection models. Image Segmentation: assists in the division of image pixels in cases like segmentation, such as semantic segmentation. Image Processing Efficiency: Lowers the computation and parameters with depth-wise separable convolution.

The results of testing with the Xception model using the same dataset as VGG16, which consists of a local coconut fruit dataset, showed training model evaluation with an accuracy of 99.87%, testing evaluation of 95.69%, validation evaluation of 90.74%. A comparison graph between the VGG16 model and Xception can be seen in figure 12.

**VGG16 graphics accurcy and validation** **Xception graphics accurcy and validation**

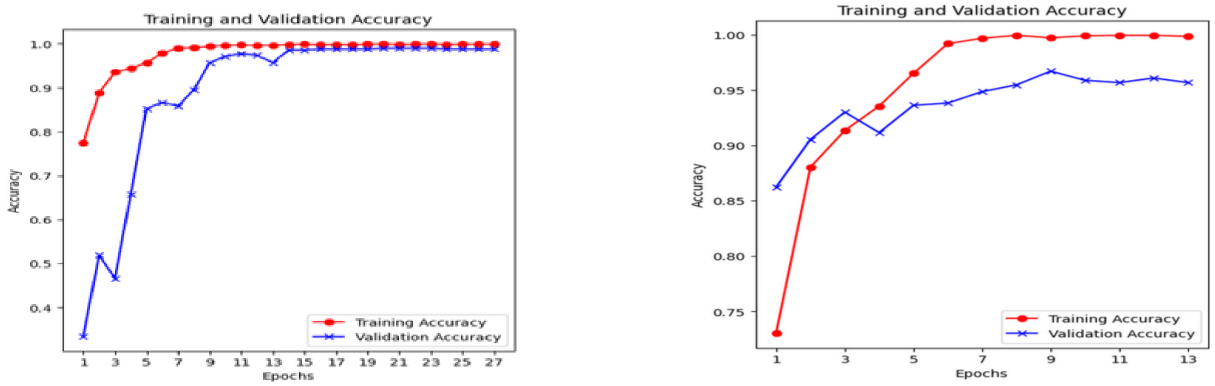


**Figure 12.** Comparison chart between vgg16 model and exception

Table 8 measures the performance of VGG16 and Xception models by comparing their performance in training, testing, validation, and AUC-ROC curves. VGG16 achieves very high training accuracy at an astounding 99.87% and maintains good testing and validation results at 98.77% and 98.97%, respectively. On the contrary, Xception registered lower training accuracy at 98.97% while scoring 95.06% and 96.71% on the testing and validation sets, respectively, but still performed slightly worse than VGG16.

In the aspect concerning the AUC-ROC curve, VGG16 recorded an AUC value of 1.00, representing a hundred percent ability in the classification task separating positive and negative classes. As good as it went, Xception's AUC is 0.99, which is still good but less than what VGG16 achieved. It is very clear that VGG16 has achieved outstanding accuracy.

**Table 8.** Comparison of evaluation models

| Model | Training (%) | Testing (%) | Validation (%) | AUC-ROC curve (%) |
|---|---|---|---|---|
| VGG16 | 99.87% | 98.77% | 98.97% | 1.00 |
| Xception | 99.87% | 95.06% | 96.71% | 0.99 |

Table 9 presents the performance of the CNN Convolutional Neural Network model with and without image augmentation. During the training of the model with image augmentation, the accuracy was 99.12%, while the model without image augmentation was slightly better at 99.82% accuracy. In the testing phase, on the other hand, the model with image augmentation managed to get an accuracy of 97.33% in contrast to the model without augmentation, which achieved only an accuracy of 96.09%. This was also the situation during the validation stage when the model with image augmentation recorded an accuracy of 97.33% as opposed to the model without augmentation, which could only achieve 95.28% accuracy. Moreover, the model with image augmentation performed excellently with an AUC-ROC value of 1.00, which means perfect classification was achieved, while the model without augmentation recorded an AUC-ROC value of 0.99, which was still excellent but not the best.

**Table 9.** Comparison of CNN with and without augmentation

| Model | Training (%) | Testing (%) | Validation (%) | AUC-ROC curve (%) |
|---|---|---|---|---|
| CNN With Image Augmentation | 99.12% | 97.33% | 97.33% | 1.00 |
| CNN Without Image Augmentation | 99.82% | 96.09% | 95.28% | 0.99 |

## 4.6. Discussion

There are five main discussion points of this research, including classification accuracy, image augmentation techniques, and the use of CNN models; comparison with other models; discussion of confusion matrices; and classification accuracy.

First, the primary concern pertains to the precision of classification. Current research results. The model, which utilized color jitter for image augmentation, demonstrated an accuracy of 99.12% for training data, 97.33% for test data, and 97.33% for validation data. The model explored using VGG16 achieved an accuracy of 99.87% for training data, an

ISSN 2723-6471


accuracy of 98.77% for test data, and an accuracy of 98.97% for validation data, when compared to the results of previous research, such as classification using the Backpropagation method only getting an accuracy of about 67% [11], then using the Naive Bayes Classifier (NBC) method, an average accuracy of 87.02% for the classification of coconut seeds [13], and the Nearest Mean Classifier (NMC) method, an accuracy of about 86.66% [12] for the classification of coconut fruit and coconut products.

Second, it refers to image augmentation techniques. The results of the current research use image augmentation techniques such as color jitter [45], rotation, and cropping. However, the results of previous research indicate that the augmentation techniques used in the same image were not explicitly specified in those reports.

Third, the current study's results demonstrate that the use of image augmentation techniques with color jitter in CNN model development, aimed at increasing dataset diversity, preventing overfitting, and enhancing the model's generalization ability, resulted in a significant increase in accuracy of up to 99.87%. For training data, an accuracy of 98.77% for test data, and an accuracy of 98.97% for validation data. This is different from earlier studies that used CNN to enhance batik classification. These studies were conducted using DenseNet169 and VGG16 two CNN models, which yielded accuracy rates of only 84.62% and 82.56%, respectively [27].

Fourth, as a comparison between the VGG16 model and other models, the results of VGG16 accuracy of 99.87% and validation accuracy of 98.97% while Xception only gets an accuracy of 99.87% and validation accuracy of 96.71%. Thus the use of the VGG16 model is better than the Xception model, while other traditional machine learning models such as Support Vector Machine (SVM) or K-Nearest Neighbors (KNN) can also be used as image classification but are not as efficient as deep learning such as CNN in handling complexity and variation in image data, especially for very large datasets, and KNN can be slow and inefficient for large image datasets because it has to calculate the distance to all training data points. Moreover, their performance is highly dependent on the selection of the distance metric and the k value; although SVM and KNN can be used for image classification tasks, they tend to be more suitable for the purpose of small datasets/retrieved features. On the other hand, for more complicated and large-scale image classification tasks, the deep learning model CNN is preferred due to its ability to perform automatic feature extraction on images as well as accommodate more variations in the data.

Fifth, as a matter of discussion, this study considers the confusion matrix as a suitable tool, especially for model accuracy assessment related to predictive modeling of binary classifications by providing classification predictions and explorations related to the actual class. In the context of local coconut fruit classification, for example, identifying whether a given image is a good coconut, a damaged coconut, or an intermediate coconut, the emergence of the confusion matrix makes it easy to see where the strengths and weaknesses of the models are. There are four components that make up the matrix classified as TP, which is the correct diagnosis of a good coconut by the model, and FP, where good coconuts are found in the image but they are actually damaged or intermediate.TN is the identification of damaged or intermediate coconuts; where healthy coconuts are incorrectly detected as damaged or intermediate, this is known as FN. More specifically, high TP and low FN indicate the model's performance in classifying good coconuts, while low FP and high TN indicate the model's functionality in classifying damaged and intermediate coconuts.

Confusion matrix analysis gives us the opportunity to refine model parameters such as prediction thresholds or training data requirements to reduce errors and improve performance. In addition, this matrix helps in the trade-off between precision and recall, depending on the needs of an application, ensuring that the model is not only correct on paper but also correct in practice. By adding intermediate classes, the model becomes more precise in handling mismatches caused by different coconut conditions and thus improves decision-making.

In this study, the VGG16 model, which has been pre-trained on large datasets like ImageNet, is relatively easier to train as it only requires modifications to a few of the last layers. Almost all of the last layers are adjusted and retrained, as VGG16 is already equipped with knowledge about common feature datasets. Lower computational and memory requirements can be achieved without the need to train the entire model from scratch. The VGG16 model has already been pre-trained, so in this case, the model can also be slightly modified for other tasks and still remain useful. This means that because the model already has a substantial amount of knowledge, one can use a minimal dataset and achieve better accuracy. The ability to quickly transfer models to different applications without requiring extensive

ISSN 2723-6471


training procedures is made possible by the flexibility of those applications. This increases how quickly alternative prototypes are filtered and the design process is completed.

In this research, we have utilized a dataset derived from field surveys, designed to reflect the conditions and variability that would be encountered in real-world applications. This dataset includes data collected from various locations and conditions, providing a more realistic representation of field situations while also focusing on the performance analysis of the model in a controlled environment

The use of a dataset from field surveys allows us to conduct tests that are more relevant and closer to real-world conditions. However, we recognize that applying the model to a new set of data that has not been previously seen remains an important step to further validate its utility and reliability. As a next step in our research, we plan to expand testing with additional data collected from broader and more diverse field surveys. This will help us identify potential weaknesses in the model and make the necessary adjustments to enhance its reliability. In addition, we will consider feedback from end users to ensure that the model is not only accurate but also relevant and useful.

This study, in totality, indicates notable strides made in the classification of local coconut fruits over previous methods and research findings, which are important for the advancement of more accurate and efficient classifiers. This research demonstrates that the application of image augmentation, which develops the augmentation technique using color jitter and CNN deep learning algorithms, along with exploration using the VGG16 architecture, yields superior classification results for local coconut fruits compared to existing methods.

The findings of this study also show that there has been significant progress in coconut fruit classification compared to previous studies. By achieving the highest accuracy reported in this study, it makes a valuable contribution to improving the accuracy of coconut fruit classification. Accurate prediction of local coconut fruits is likely to be an effective approach to improving coconut fruit accuracy and performance to assist farmers in the rapid classification of local coconut fruits. The results also show that it achieves high accuracy. It provides a valuable contribution in terms of developing better local coconut fruit recognition methods and classification. This study can be a reference for coconut farmers and traders because the limited availability of coconut experts can hinder the classification process. Even for coconut industry companies, the development of accurate and efficient automatic classification methods is very important. Further research is needed to improve the performance of coconut fruit classification and overcome the challenges of recognizing variations by adding and multiplying more datasets and more varied categories such as taste, content thickness, harvest period, and coconut age, so that the accuracy is higher.

## 5. Conclusion

In this research, we have successfully improved the accuracy of local coconut fruit classification using image augmentation and deep learning with the CNN algorithm. This research developed a CNN model using color jitter augmentation data to improve the performance of local coconut fruit recognition. The model trained and evaluated showed good results, with a training accuracy of 99.12%, testing accuracy of 97.33%, and model evaluation accuracy of 97.33%. For accuracy, the AVG f1-score average was 0.97, the macro AVG average was 0.97, and the weight AVG average was 0.97. Researchers also explored using the VGG16 model architecture, where the results of models that have been trained and evaluated increase and are very good, with training data accuracy of 99.87%, testing data of 98.77%, and validation data of 98.97%. On the evaluation of model performance for AVG f1-score around 0.99%, macro AVG around 0.99%, and weight AVG around 0.99%, the results of this research reveal an advancement in the precision of local coconut fruit classification using image augmentation and deep learning using the CNN algorithm compared to previous methods.

The results of this study can be a recommendation for coconut farmers and traders, as well as the palm oil industry in the Indragiri Hilir district, to improve the accuracy of local coconut fruit classification automatically. We also recommend further research in the form of increasing the dataset to tens of thousands of datasets, and further research may also be able to improve and further develop both the model and CNN architecture. This research can be the basis for further research on coconut fruit classification, with a focus on more complex modeling.

**6. Declarations**

## 6.1. Author Contributions

Conceptualization: U.U., F.Y., and M.R.R.; Methodology: F.Y. and M.R.R.; Software: U.U.; Validation: U.U., F.Y., and M.R.R.; Formal Analysis: U.U., F.Y., and M.R.R.; Investigation: U.U.; Resources: F.Y.; Data Curation: F.Y.; Writing Original Draft Preparation: U.U., F.Y., and M.R.R.; Writing Review and Editing: F.Y., U.U., and M.R.R.; Visualization: U.U.; All authors have read and agreed to the published version of the manuscript.

## 6.2. Data Availability Statement

The data presented in this study are available on request from the corresponding author.

## 6.3. Funding

The authors would like to thank the Ministry of Education, Culture, Research, and Technology, Directorate General of Higher Education, Research, and Technology of Indonesia, for providing funding for the Fundamental Research Scheme Regular Research Program in 2024, as well as Indragiri Islamic University for providing facilities for conducting research.

## 6.4. Institutional Review Board Statement

Not applicable.

## 6.5. Informed Consent Statement

Not applicable.

## 6.6. Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

**References**

[1] S Xu et al, “Optical coherent dot-product chip for sophisticated deep learning regression,” *Light Sci Appl*, vol 10, no 1, pp 1–12, 2021, doi: 101038/s41377-021-00666-8

[2] Y Liu, X Yu, J Zhao, C Pan, and K Sun, “Development of a Robust Data-Driven Soft Sensor for Multivariate Industrial Processes with Non-Gaussian Noise and Outliers,” *Mathematics*, vol 10, no 20, pp 1–16, 2022, doi: 103390/math10203837

[3] G M van de Ven, H T Siegelmann, and A S Tolias, “Brain-inspired replay for continual learning with artificial neural networks,” *Nat Commun*, vol 11, no 1, pp 1–14, 2020, doi: 101038/s41467-020-17866-2

[4] K Dwivedi, M K Dutta, and J P Pandey, “EMViT-Net: A novel transformer-based network utilizing CNN and multilayer perceptron for the classification of environmental microorganisms using microscopic images,” *Ecol Inform*, vol 79, no December 2023, pp 1-20, 2024, doi: 101016/jecoinf2023102451

[5] M Xie, Z Wang, X Xu, X Zheng, H Liu, and P Shi, “Quantitative estimation of the nutrient uptake requirements of peanut,” *Agronomy*, vol 10, no 1, pp 151-163, 2020, doi: 103390/agronomy10010117

[6] Y Shima, “Image Augmentation for Object Image Classification Based on Combination of Pre-Trained CNN and SVM,” *J Phys Conf Ser*, vol 1004, no 1, pp 85–97, 2018, doi: 101088/1742-6596/1004/1/012001

[7] H Mardesci, Santosa, N Nazir, and R A Hadiguna, “Identification of prospective product for the development of integrated coconut agroindustry in Indonesia,” *Int J Adv Sci Eng Inf Technol*, vol 9, no 2, pp 511–517, 2019, doi: 1018517/ijaseit927172

[8] A D Prabaswari and N P Al-Bana, “Product Analysis of Safety Belt and Body Harness Coconut Sugar Climber: An Ergonomic, Durability, and Economic Feasibility Principle,” *Math Model Eng Probl*, vol 11, no 8, pp 2143–2151, 2024.

[9] Mawardati, Jullimursyida, Suryadi, E T Kembaren, and I Sinta, “Prospects for the Development of Coconut and Oil Palm Commodities in Community Plantations in North Aceh Regency,” *Univers J Agric Res*, vol 10, no 6, pp 722–730, 2022.

[10] A Poranen, A Kouvonen, and H Nordquist, “Human errors in emergency medical services: a qualitative analysis of contributing factors,” *Scand J Trauma Resusc Emerg Med*, vol 32, no 1, pp 1–11, 2024, doi: 101186/s13049-024-01253-7

[11] C Lee, S S Sarwar, P Panda, G Srinivasan, and K Roy, “Enabling Spike-Based Backpropagation for Training Deep Neural Network Architectures,” *Front Neurosci*, vol 14, no February, pp 1–22, 2020, doi: 103389/fnins202000119

[12] A Srinivasulu and N Sriraam, “Signal processing framework for the detection of ventricular ectopic beat episodes,” *J Med Signals Sens*, vol 13, no 3, pp 239–251, 2023, doi: 104103/jmssjmss_12_22

[13] M Rotmensch, Y Halpern, A Tlimat, S Horng, and D Sontag, “Learning a Health Knowledge Graph from Electronic Medical Records,” *Sci Rep*, vol 7, no 1, pp 1–11, 2017, doi: 101038/s41598-017-05778-z

[14] K P Chou et al, “Robust Feature-Based Automated Multi-View Human Action Recognition System,” *IEEE Access*, vol 6, no February, pp 15283–15296, 2018, doi: 101109/ACCESS20182809552

[15] F Z Okwonu, N A Ahad, and I E Okoloko, “Robust Hybrid Classification Methods and Applications,” *Sci Technol*, vol 30, no 4, pp 2831–2850, 2022, doi: 1047836/pjst30429

[16] C Augsornthip, P Neranon, P Phukpattaranont, and A Romyen, “Sound Classification for Non-Destructive Diagnosis of Basal Stem Rot Disease based on Stem Density in Oil Palm Trunks,” *J Adv Res Appl Mech*, vol 124, no 1, pp 19–38, 2024.

[17] C Shorten and T M Khoshgoftaar, “A survey on Image Data Augmentation for Deep Learning,” *J Big Data*, vol 6, no 1, pp 60-72, 2019, doi: 101186/s40537-019-0197-0

[18] M Frid-Adar, I Diamant, E Klang, M Amitai, J Goldberger, and H Greenspan, “GAN-based synthetic medical image augmentation for increased CNN performance in liver lesion classification,” *Neurocomputing*, vol 321, no. 1, pp 321–331, 2018, doi: 101016/jneucom201809013

[19] M Chen and W “Grace” Guo, “DCGAN-CNN with physical constraints for porosity prediction in laser metal deposition with unbalanced data,” *Manuf Lett*, vol 35, no 51, pp 1146–1154, 2023, doi: 101016/jmfglet202308116

[20] W Bönke, B Alaei, A Torabi, and D Oikonomou, “Data augmentation for 3D seismic fault interpretation using deep learning,” *Mar Pet Geol*, vol 162, no January, pp. 1-11, 2024, doi: 101016/jmarpetgeo2024106706

[21] I S Chandra, R K Shastri, D Kavitha, K R Kumar, S Manochitra, and P B Babu, “CNN based color balancing and denoising technique for underwater images: CNN-CBDT,” *Meas Sensors*, vol 28, no February, pp. 1-9, 2023, doi: 101016/jmeasen2023100835

[22] F Wagner, A Eltner, and H G Maas, “River water segmentation in surveillance camera images: A comparative study of offline and online augmentation using 32 CNNs,” *Int J Appl Earth Obs Geoinf*, vol 119, no February, pp 1-13, 2023, doi: 101016/jjag2023103305

[23] W Baccouch, S Oueslati, B Solaiman, and S Labidi, “A comparative study of CNN and U-Net performance for automatic segmentation of medical images: Application to cardiac MRI,” *Procedia Comput Sci*, vol 219, no 2022, pp 1089–1096, 2023, doi: 101016/jprocs202301388

[24] E Dzotsenidze, E Valla, S Nõmm, K Medijainen, P Taba, and A Toomela, “Generative Adversarial Networks as a Data Augmentation Tool for CNN-Based Parkinson’s Disease Diagnostics,” *IFAC-PapersOnLine*, vol 55, no 29, pp 108–113, 2022, doi: 101016/jifacol202210240

[25] B Marty, R Gaudin, T Piperno, D Rouquette, C Schwob, and L Mezeix, “Methodology to classify high voltage transmission poles using CNN approach from satellite images for safety public regulation application : Study case of rural area in Thailand,” *Syst Soft Comput*, vol 6, no November 2023, pp. 1-11, 2024, doi: 101016/jsasc2024200080

[26] H Mo and G Zhao, “RIC-CNN: Rotation-Invariant Coordinate Convolutional Neural Network,” *Pattern Recognit*, vol 146, no September 2023, pp 1-13, 2024, doi: 101016/jpatcog2023109994

[27] B J Filia et al, “Improving Batik Pattern Classification using CNN with Advanced Augmentation and Oversampling on Imbalanced Dataset,” *Procedia Comput Sci*, vol 227, no. 1, pp 508–517, 2023, doi: 101016/jprocs202310552

[28] I Slamet, D C Susilotomo, E Zukhronah, S Subanti, and I Susanto, “Retinopathy Classification using Convolutional Neural Network Method with Adam Optimization and Applied Batch Normalization,” *J Appl Data Sci*, vol 5, no 3, pp 1123–1133, 2024, doi: 1047738/jadsv5i3309

[29] R O Alnashwan, S A Chelloug, N S Almalki, I Issaoui, A Motwakel, and A Sayed, “Lighting Search Algorithm With Convolutional Neural Network-Based Image Captioning System for Natural Language Processing,” *IEEE Access*, vol 11, no December, pp 142643–142651, 2023, doi: 101109/ACCESS20233342703

[30] L Alzubaidi et al, “Review of deep learning: concepts, CNN architectures, challenges, applications, future directions,” vol 8, no 1 Springer International Publishing, 2021 doi: 101186/s40537-021-00444-8

[31] V V, C R K, and R A C, “Real Time Object Detection System with YOLO and CNN Models: A Review,” *J Xi’an Univ Archit Technol*, vol XIV, no 7, pp 144–151, 2022, doi: 1048550/arXiv220800773

[32] Y Man, X Weng, P K Sivakumar, M O'Toole, and K Kitani, "Multi-Echo LiDAR for 3D Object Detection," *Comput Vis Pattern Recognit*, vol 2107, no July, pp 3743–3752, 2021, doi: 1048550/arXiv210711470

[33] A Thapa et al, "Deep learning for breast cancer classification: Enhanced tangent function," *Comput Intell*, vol 38, no 2, pp 506–529, 2022, doi: 101111/coin12476

[34] R Ranjan et al, "Crystal Loss and Quality Pooling for Unconstrained Face Verification and Recognition," *ArXiv*, vol 14, no 8, pp 1–15, 2018, doi: 1048550/arXiv180401159

[35] Y Song and Y Wang, "Towards High-Performance Exploratory Data Analysis (EDA) via Stable Equilibrium Point," *ACM Digit Libr*, vol 14450 LNCS, no November, pp 483–492, 2024, doi: 101007/978-981-99-8070-3_37

[36] R Indrakumari, T Poongodi, and S R Jena, "Heart Disease Prediction using Exploratory Data Analysis," *Procedia Comput Sci*, vol 173, no 2019, pp 130–139, 2020, doi: 101016/jprocs202006017

[37] A S Paramita and A Ramadhan, "An Unsupervised Learning and EDA Approach for Specialized High School Admissions," *J Appl Data Sci*, vol 5, no 2, pp 316–325, 2024, doi: 1047738/jadsv5i2178

[38] H Liu, S He, and J Peng, "K-nearest neighbor based on exploratory data analysis of curriculum models of Chinese early childhood education," *Heliyon*, vol 10, no 14, pp. 1-21, 2024, doi: 101016/jheliyon2024e33781

[39] T Islam, M S Hafiz, J R Jim, M M Kabir, and M F Mridha, "A systematic review of deep learning data augmentation in medical imaging: Recent advances and future research directions," *Healthc Anal*, vol 5, no December 2023, pp. 1-18, 2024.

[40] A Tatar, M Haghighi, and A Zeinijahromi, "Experiments on image data augmentation techniques for geological rock type classification with convolutional neural networks," *J Rock Mech Geotech Eng*, vol X, no Mei, pp 1–21, 2024.

[41] S V, K Prasad, C Mukhopadhyay, and B Banerjee, "Data augmentation for Gram-stain images based on Vector Quantized Variational AutoEncoder," *Neurocomputing*, vol 600, no June, pp 1-22, 2024, doi: 101016/jneucom2024128123

[42] C Harvey and A Noheria, "Deep Learning Encoded Ecg - Avoiding Overfitting in Ecg Machine Learning," *J Am Coll Cardiol*, vol 83, no 13, pp. 172-172, 2024, doi: 101016/s0735-1097(24)02162-4

[43] G Wang, W Li, S Ourselin, and T Vercauteren, "Automatic brain tumor segmentation using convolutional neural networks with test-time augmentation," *Lect Notes Comput Sci (including Subser Lect Notes Artif Intell Lect Notes Bioinformatics)*, vol 11384 LNCS, no November, pp 61–72, 2019, doi: 101007/978-3-030-11726-9_6

[44] S Zini, A Gomez-villa, M Buzzelli, and M Bicocca, "Planckian Jitter : countering the color-crippling effects of color jitter on self-supervised training," *Comput Vis Pattern Recognit*, vol 1, no feb, pp 1–15, 2023, doi: 1048550/arXiv220207993

[45] K Hulliyah, N S A Abu Bakar, S Aripiyanto, and D Khairani, "Revolutionizing Digit Image Recognition: Pushing the Limits with Simple CNN and Challenging Image Augmentation Techniques on MNIST," *J Appl Data Sci*, vol 4, no 3, pp 119–129, 2023, doi: 1047738/jadsv4i3104

[46] F Gapsari et al, "A convolutional neural network -VGG16 method for corrosion inhibition of 304SS in sulfuric acid solution by timoho leaf extract," *J Mater Res Technol*, vol 30, no March, pp 1116–1127, 2024, doi: 101016/jjmrt202403156

[47] Y Chen, Y Chen, S Fu, W Yin, K Liu, and S Qian, "VGG16-based intelligent image analysis in the pathological diagnosis of IgA nephropathy," *J Radiat Res Appl Sci*, vol 16, no 3, pp. 1-9, 2023, doi: 101016/jjrras2023100626

[48] T G W Boers et al, "Medical Image Analysis," *Med Image Anal*, vol 002223, no August, pp. 1-11, 2024.

[49] A O Topal, R Chitic, and F Leprévost, "One evolutionary algorithm deceives humans and ten convolutional neural networks trained on ImageNet at image recognition," *Appl Soft Comput*, vol 143, no May, pp 1-18, 2023.

[50] X Chen, H Chen, J Wan, J Li, and F Wei, "An Enhanced AlexNet-Based Model for Femoral Bone Tumor Classification and Diagnosis Using Magnetic," *J Bone Oncol*, vol 00106, no July, pp. 1-7, 2024, doi: 101016/jjbo2024100626

[51] B Yadav, A Indian, and G Meena, "Recognizing Off-line Devanagari Handwritten Characters Using Modified Lenet-5 Deep Neural Network," *Procedia Comput Sci*, vol 235, no 2023, pp 799–809, 2024, doi: 101016/jprocs202404076

[52] D Valero-Carreras, J Alcaraz, and M Landete, "Comparing two SVM models through different metrics based on the confusion matrix," *Comput Oper Res*, vol 152, no April 2022, pp. 1-12, 2023, doi: 101016/jcor2022106131

[53] J M Jung et al, "Multi-algorithmic approach for detecting outliers in cattle intake data," *J Agric Food Res*, vol 15, no December 2023, pp. 1-8, 2024, doi: 101016/jjafr2024101021

[54] O Peretz, M Koren, and O Koren, "Naive Bayes classifier – An ensemble procedure for recall and precision enrichment," *Eng Appl Artif Intell*, vol 136, no PB, pp 1-12, 2024, doi: 101016/jengappai2024108972