

Classification System of Crystal Guava (*Psidium Guajava*) Using Convolutional Neural Network And Rectified Linear Unit Method Based on Android

Wiktasari¹, Tri Raharjo Yudiantoro^{2*}, Muhammad Fikry Alifiansyah³, Kurniangsih⁴, Liliek Triyono⁵, Abu Hasan⁶
^{1,2,3,4,5,6}*Electrical Engineering Departement, Politeknik Negeri Semarang, Central Java, 50275, Indonesia*

Abstract—These instructions *Abstract - However, determining the ripeness of fruit is frequently done by hand, which presents problems with consistency and efficiency. In order to improve the sorting of crystal guava fruit maturity, this study suggests combining machine learning technology with the creation of digital image-based apps. Fruit ripeness is classified using a convolutional neural network (CNN), a deep learning model, based on the color of its skin. It is anticipated that the method will increase productivity and offer superior precision while sorting crystal guava fruit. The System Development Life Cycle (SDLC) with a Waterfall approach is the methodology employed. The system design formed from the deep learning model resulted in excellent performance in classifying images of crystal guava fruit by utilizing model training from the base models ResNet50V2, DenseNet121, NASNetMobile, and MobileNetV2 with a combination of training using K-fold cross-validation with a 5-fold configuration. The best-trained model achieved an average highest accuracy of 99.92% in model training using MobileNetV2 with the lowest average loss value of 0.0088. The system application was developed using mobile Android, leveraging the Flutter framework and Dart programming language. The research results demonstrate a comparison of testing on crystal guava and local guava fruits against ripeness classification parameters*

Keywords-Crystal Guava Machine Learning Convolutional Neural Network Image-Based Application Development Ripeness Classification..

1. Introduction

The integration of technological collaboration with Taiwan has ushered in a new era for enhancing the genetic diversity of local guava (*Psidium guajava*) varieties in Indonesia. This endeavor aims to develop a novel variant of guava, known as "Jambu Kristal," which exhibits distinctive characteristics such as a sweet flavor with a Brix level between 11 and 12, a slightly flattened round shape, and occasional asymmetric forms. Notably, Jambu Kristal has a seed content of less than 3%, an unevenly textured surface, and typically weighs between 100 to 500 grams per fruit. The fruit's skin is a light green, while its flesh is pale white, resembling pear flesh [1].

Jambu Kristal offers several advantages, including a refreshing taste, an acidity profile akin to apples and pears, ease of cultivation, year-round fruiting, substantial entrepreneurial opportunities for both fruit production and seedling cultivation, and a high market value at both the farmer and supermarket levels (Pakpahan, 2011). The fruit size of Jambu Kristal varies with the seasons, growing larger during the rainy season and exhibiting a sweeter taste in the dry season. This variability makes Jambu Kristal a valuable crop for farmers seeking to enhance their economic prospects. The ripeness of Jambu Kristal is generally assessed through

parameters such as size, weight, skin color, and aroma [2].

Research on a deep learning-based date fruit classification model achieved an impressive 99% accuracy in classifying eight types of date fruits. They utilized Convolutional Neural Networks (CNN) with MobileNetV2 architecture and transfer learning. Despite the impressive results, the dataset used was limited, with only 203 to 240 images per date fruit type, which may constrain the model's ability to generalize across a wider variety of date fruits [2].

Research on fruit and vegetable classification models using deep learning with the CBAM (Convolutional Block Attention Module) and MobileNetV2 architecture reached accuracies of 95.86% and 93.78% for classifying two different fruit sets. However, the model faces challenges in implementation within embedded vision systems and remains limited to two-dimensional image research [3].

Research on a new date fruit classification model using deep transfer learning achieved a validation accuracy of 97.21% and a test accuracy of 95.21%. This study also created a new dataset consisting of images of date fruits in 27 classes. However, representation may not be uniform across classes,

especially for minority classes, which could affect the model's generalization [4].

A survey of deep learning methods for vegetable type recognition highlighted the success of using CNNs and attention models. It demonstrated that large and diverse datasets are crucial for optimal model training, though some models are still sensitive to lighting changes and environmental conditions [5].

Research on applying deep learning techniques for vegetable recognition using CNN with transfer learning successfully identified various vegetable types with high accuracy. Nonetheless, this research is limited by the variety of vegetables in the dataset and its reliance on transfer learning [6].

Research using K-Nearest Neighbor (KNN) and HSV color space transformation to detect the ripeness of crystal guavas achieved high accuracy for ripe and unripe guava categories. However, this method does not detect ripeness in real-time. (Wibowo et al., 2021). Research on CNN for classifying the ripeness of Badami mangoes reached a high accuracy of 97.2%. However, the study is limited to the Badami mango variety and does not provide real-time ripeness detection [7].

Research using deep learning with ANN for classifying the ripeness of passion fruits based on color achieved 80% accuracy. This research is limited to passion fruit and does not offer real-time ripeness detection [8].

Research on K-NN with HSV color feature extraction and GLCM texture for classifying the ripeness of guava bol achieved 93% accuracy. This study demonstrates a good application of methods for fruit ripeness classification [9].

Research on developing a mobile application for vegetable classification using CNN and SVM achieved a high accuracy of up to 99.98% for several vegetable types. The main challenge is the use of image datasets and the potential for further development in video classification media. (Jaelani Akbar et al., n.d.). Research aimed at improving fruit classification on mobile devices using transfer learning achieved 95% accuracy. A major drawback is the long training time depending on model complexity and dataset size.

Tamina employed VGG-16 with deep CNN for image classification, achieving 95.40% accuracy. However, additional classifiers are needed to achieve two-class classification goals [10]. Research on CNN for fruit image classification achieved 91.42% accuracy, with recommendations to increase the dataset size to improve accuracy. (Maulana et al., 2020). Research on using CNN for herbal plant identification achieved 96.54% accuracy. The main challenges are lighting and high-quality image sharpness [11]. Research on developing a mobile application for vegetable classification using CNN achieved 98.1% accuracy. This method still lacks comprehensive image classification data [12]

Overall, these studies show significant progress in using deep learning and machine learning for fruit and vegetable classification. However, common weaknesses include dataset limitations, sensitivity to lighting conditions, and reliance on transfer learning techniques. Further development requires larger and more diverse datasets and techniques capable of handling environmental variation more effectively.

2. Literature Review

The development of this system's Aplikasi Klasifikasi Buah Jambu Kristal app was driven by the primary purpose of detecting digital images captured by a smartphone's camera and the upload function for images taken from galleries. Deep learning models that have already been integrated into applications will work to analyze images that are associated with the classification of buah jambu kristal. System architecture can be seen in Figure 1.

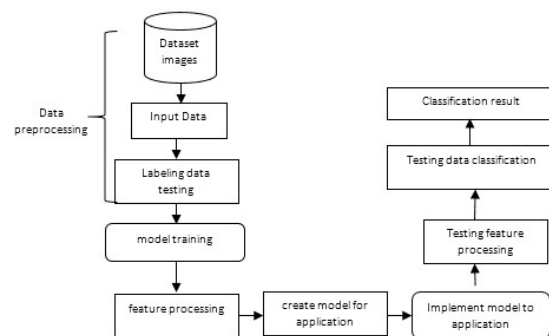


Fig 1. Research Method

2. 1. Data Preprocessing

This study uses a raw picture dataset of 2204 image

units to categorize 3 image classes on crystal guava fruit: ripe, not yet ripe, and rotten. Data preparation involves dividing the dataset into training and validation sets. Separate files comprising each image class category according to its number are used to construct the training and validation data. In Figure 2, the data labeling is displayed.

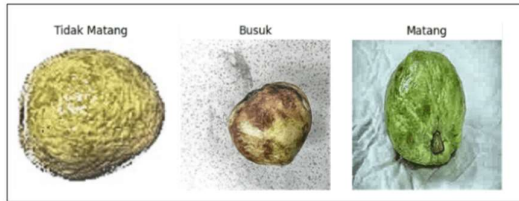


Fig 2. Labeling Crystal Guava Fruit Dataset

The dataset is divided into two parts: training and validation, following a 70:30 ratio. Details on the data split are shown in Table 1. For the "Rotten" category, there are 465 samples for training and 151 for validation. The "Ripe" category has 382 samples in training and 127 in validation. For "Not yet ripe," the split is 484 training samples and 161 validation samples.

Table 1. Division of Training and Validation Datasets

Class Label	Data Training	Data Validation
Rotten	465	151
Ripe	382	127
not yet ripe	484	161
Total	1331	439

The whole amount of data is displayed in the table, and Figure 3 also displays a comparison graph of the data.

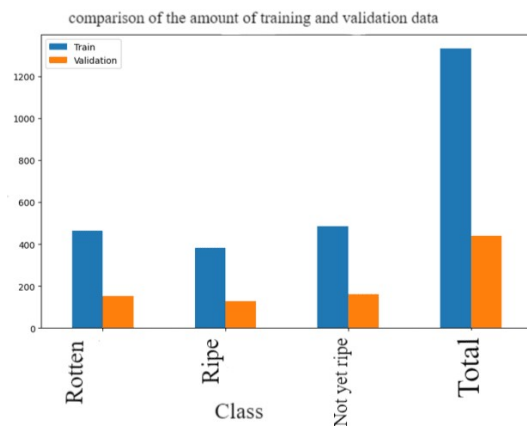


Fig 3. Comparison of the Amount of Training and Validation Data

2. 2.Creating Model

The model was developed through a classification system for crystal guava fruit, employing the Convolutional Neural Network (CNN) method, a form of deep learning architecture. This system is designed to identify and categorize crystal guava fruit using images captured by a camera or chosen from a gallery. The focus is on accurately classifying the fruit based on visual data. The selection process will determine the most effective CNN model. This model is preferred for its exceptional ability to recognize patterns within images. Key features such as shape, texture, and color play a significant role in this recognition. These characteristics are vital for effective visual object classification.

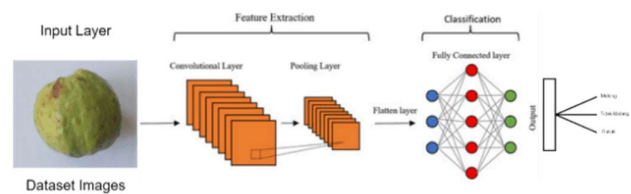


Fig 4. CNN Architecture Crystal Guava Fruit Model

The pre-trained CNN model architecture for the crystal guava fruit classification application, illustrated in Figure 3.10, demonstrates that the CNN structure comprises several distinct layers, including convolution layers, pooling layers, and fully connected layers. In the initial phase, the convolution layer is responsible for extracting key features from the crystal guava fruit images, such as edges, corners, and textures. These features are then condensed through a pooling layer to reduce the data's dimensionality and enhance computational efficiency. Following this, the features are integrated and classified using a fully connected layer, which links each neuron to all neurons in the preceding layer. Finally, the output layer generates a classification prediction for the detected crystal guava fruit.

The model is built using the TensorFlow and Keras frameworks, which facilitate more efficient development and training of deep learning models. The dataset comprises images of crystal guava fruit that have been labeled according to defined categories, including "Ripe," "Not Ripe," and "Rotten." This model aims to achieve high accuracy in the classification of crystal guava fruit.

The K-Fold Cross-Validation technique is utilized in

the advanced stages of deep learning model building to train a Convolutional Neural Network (CNN) architecture for the classification of crystal guava fruit. To make sure that the final model has good generalization to data that has never been seen before, the K-Fold Cross-Validation method was selected. Specifically, this method lessens the workload associated with overfitting-prone model training activities. The phases of model training with K-Fold are as follows:

1) Selection of Basic Model

The model employed in this study is made up of multiple pre-trained deep learning architecture models, including NASNetMobile, MobileNetV2, Densenet, and ResNet50V2, that were previously trained on the ImageNet dataset. In order to assess efficiency in terms of accuracy, performance, and computational resource usage, these models were compared. Several Fully Connected layers were added to this basic model in order to adapt it to the goal of categorizing photos of crystal guava fruit.

2) Model Architecture

There are multiple key parts to the developed model. The models' convolution layer is designed to extract features from photos of crystal guavas. The pooling layer uses GlobalAveragePooling2D to minimize dimensions and guard against overfitting. Rectified Linear Unit (ReLU) activation and 256-unit density layer are used to capture non-linear patterns. The output layer, which uses softmax activation, is classified into three groups: "mature," "immature," and "rotten."

3) K-Fold Cross-Validation Process

The training data is now separated into 5 subsets, or folds. Four subsets are utilized as training data and one subset is used as validation data for each iteration. Each fold was utilized once as validation data during the five training iterations. To avoid performance degradation, each model is trained with a low learning rate using the Adam Optimizer.

4) Implementation of Early Stopping

By monitoring the training model data based on validation loss and validation accuracy, early stopping is introduced during training to terminate training if the model does not show performance improvement on validation data after multiple epochs in the future. In doing so, training time is decreased and overfitting is avoided.

5) Evaluation and Aggregation of Results

The model is assessed at the end of training based on the accuracy and loss values obtained from the validation data at each fold. After that, an average of these numbers is calculated to give a general picture of the model's performance.

2. 3. Model Training Using K-Fold

The K-Fold Cross-Validation technique is utilized in the advanced stages of deep learning model building to train a Convolutional Neural Network (CNN) architecture for the classification of crystal guava fruit. To make sure that the final model has good generalization to data that has never been seen before, the K-Fold Cross-Validation method was selected. Specifically, this method lessens the workload associated with overfitting-prone model training activities. The phases of model training with K-Fold are as follows:

1) Selection of Basic Model

The model employed in this study is made up of multiple pre-trained deep learning architecture models, including NASNetMobile, MobileNetV2, Densenet, and ResNet50V2, that were previously trained on the ImageNet dataset. In order to assess efficiency in terms of accuracy, performance, and computational resource usage, these models were compared. Several Fully Connected layers were added to this basic model in order to adapt it to the goal of categorizing photos of crystal guava fruit.

2) Model Architecture

There are multiple key parts to the developed model. The models' convolution layer is designed to extract features from photos of crystal guavas. The pooling layer uses GlobalAveragePooling2D to minimize dimensions and guard against overfitting. Rectified Linear Unit (ReLU) activation and 256-unit density layer are used to capture non-linear patterns. The output layer, which uses softmax activation, is classified into three groups: "mature," "immature," and "rotten."

3) K-Fold Cross-Validation Process

The training data is now separated into 5 subsets, or folds. Four subsets are utilized as training data and one subset is used as validation data for each iteration. Each fold was utilized once as validation data during the five training iterations. To avoid performance degradation, each model is trained with a low learning

rate using the Adam Optimizer.

4) Implementation of Early Stopping

By monitoring the training model data based on validation loss and validation accuracy, early stopping is introduced during training to terminate training if the model does not show performance improvement on validation data after multiple epochs in the future. In doing so, training time is decreased and overfitting is avoided.

5) Evaluation and Aggregation of Results

The model is assessed at the end of training based on the accuracy and loss values obtained from the validation data at each fold. After that, an average of these numbers is calculated to give a general picture of the model's performance.

2. 4.Evaluation

To determine which model performs best, deep learning model testing must be designed and tested. Training data reporting is achieved by comparing the performance of the basic model with graphs and confusion matrices. True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN) are the four variables used in the confusion matrix to explain the outcomes of deep learning classification. Metrics for performance evaluation, including accuracy, precision, recall, and F1-score, will be computed using these four variables.

True Positive	True Negatif
False Positive	False Negative

1) Accuracy

The percentage of positive and negative forecasts that are accurate relative to the total number of predictions made is called accuracy. An overview of the model's accuracy in correctly classifying samples is given. The accuracy value is computed using the formula below:

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \quad (1)$$

2) Precision

The ratio of accurate positive predictions (True Positive) to all positive predictions (True Positive + False Positive) is known as precision. Since precision tells us how many real positive predictions

are accurate, it's a crucial indicator for reducing the amount of false positives. The precision value is computed using the formula below:

$$Precision = \frac{TP}{TP + FP} \quad (2)$$

3) Recall

A metric called recall is used to assess how well the model finds all real positive examples. This is the calculation formula:

$$Recall = \frac{TP}{TP + FN} \quad (3)$$

4) F1-Score

By averaging the two, the F1-Score measure integrates recall and precision into a single value. When there is a trade-off between these two parameters, the F1-Score is helpful since it serves to balance recall and precision. This is the calculation formula:

3. Results

This section presents the results of the research along with the tests conducted. Additionally, it includes a discussion of the research findings and the tests performed.

3.1. Result Evaluation Model

The model constucted consists of four types of pre-trained modls: ResNet50V2, DenseNet121, NASNetMobile, and MobileNetV2. Each model will be trained on the data individually. The training process is where the model learns from a pre-prepared dataset through data preprocessing. This process begins with inputting images of crystal guava into the CNN model. The images are then processed through various data transformations within the CNN layers. These CNN layers include convolutional layers, pooling layers, and fully connected layers, all aimed at identifying patterns, features, and important information in the guava images. After passing through these layers, the model generates predictions for the given images.

The predictions made by the model are then compared

to the actual results using a loss function. The loss function measures the discrepancy or error between the model's predictions and the expected results. This provides an assessment of the model's performance, represented by the loss score. An optimizer is then used to update the weights in each CNN layer based on the loss score obtained, with the goal of reducing prediction errors in the next iteration. This training process is repeated for a specified number of epochs, where one epoch represents a complete iteration through the entire training dataset. Additionally, 5-Fold Cross-Validation is employed, dividing the data into 5 subsets (folds). In each iteration, one subset is used as validation data, while the remaining 4 subsets serve as training data. The training is performed 5 times with each fold used once as validation data. Early stopping is used in this study to halt the training process if the performance on the validation data no longer improves or starts to decline.

To evaluate a model's performance, one can examine the accuracy and loss scores it produces. Accuracy provides insight into how precise the model's predictions are, while the loss score reflects the extent of errors in the model's predictions. A well-performing model is characterized by high accuracy and low loss, indicating its success in the training process. Below are the results for each model:

1) ResNet50V2 Model

Based on the training results of the ResNet50V2 model shown in Figures 3 and 4, the model was trained with a configuration of 100 epochs and 5 subsets (folds).

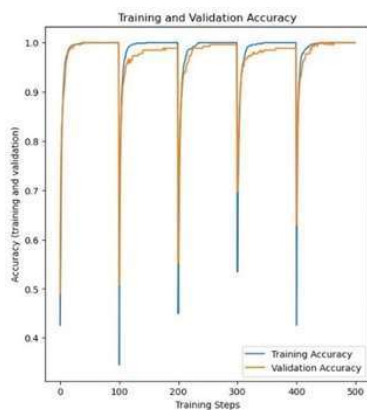


Fig 5. ResNet50V2 Accuracy Training and Validation Results

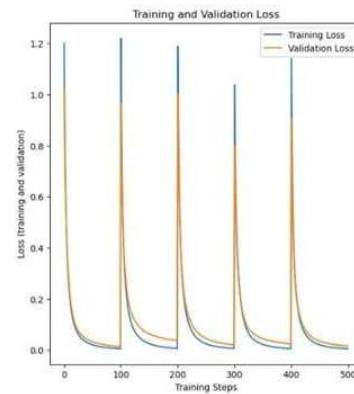


Fig 6. ResNet50V2 Training and Validation Loss Results

Based on the results displayed from the ResNet50V2 model during k-fold cross-validation, the data is divided into several subsets or "folds." Each fold produces its own loss and accuracy values. The model was evaluated using 5-fold cross-validation, and it demonstrated excellent performance across all folds. In Fold 1, the model achieved a loss of 0.0128 with a perfect accuracy of 100.00%. In Fold 2, the loss increased to 0.0376 with a slight drop in accuracy to 98.86%. Fold 3 recorded a loss of 0.0196 and a high accuracy of 99.62%. In Fold 4, the model exhibited a loss of 0.0235 with an accuracy of 98.86%. Finally, in Fold 5, the model once again reached a perfect accuracy of 100.00% with a loss of 0.0154. After evaluating all the folds, the average accuracy of the model was 99.47%, with an average loss of 0.0218, indicating consistent and reliable performance.

2) NASNetMobile Model

The training results of the NASNetMobile model, shown in Figures 5 and 6, show that the model was trained with a configuration of 100 epochs and 5 subsets (fold). In these images, it can be seen that training was carried out using 100 epochs and 5 folds. This configuration includes 100 epochs and 5 subsets to train the NASNetMobile model. Figures 5 and 6 depict the results of training with 100 epochs and 5 folds. The NASNetMobile model was trained using 100 epochs and divided into 5 subsets, as seen in Figures 7 and 8.

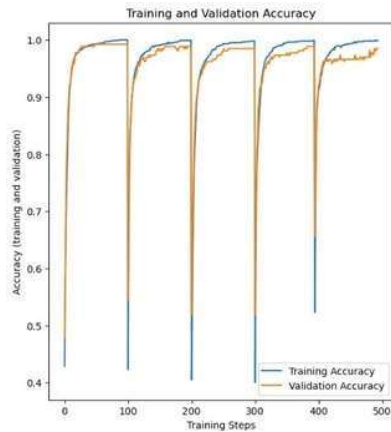


Fig 7. NASNetMobile Accuracy Training and Validation Results

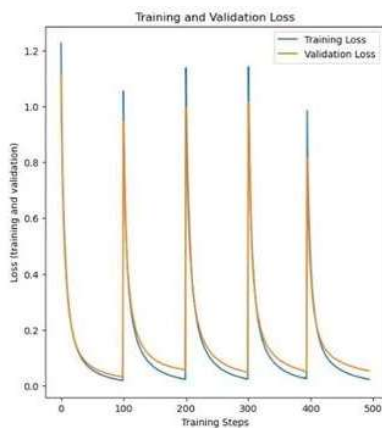


Fig 8. NASNetMobile Training and Validation Loss Results

It shows the k-fold cross-validation results for a machine learning model with five folds using the NASNetMobile model's findings. According to the findings of the 5-fold cross-validation model evaluation, the model achieved a loss of 0.0314 at Fold 1 with a 99.25% accuracy. Fold 2 recorded a loss of 0.0572 with an accuracy of 98.86%, and Fold 3 recorded a loss of 0.0482 with a 98.48% accuracy. Additionally, Fold 4 displays a 0.0504 loss with a 98.86% accuracy, and Fold 5 displays a 0.0534 loss with a 98.48% accuracy. These total findings yielded an average loss of 0.0481 and an accuracy of 98.79% on average. These numbers show that the model performs exceptionally well, with an error (loss) rate that is comparatively low and an accuracy that is close to 99%.

3) Model DenseNet121

The training results of the DenseNet121 model are illustrated in Figures 4.7 and 4.8. The model was trained using five subsets, known as folds.

Additionally, the training was configured for 100 epochs. These figures present the performance metrics obtained during the training process. Overall, the results provide insights into the model's effectiveness.

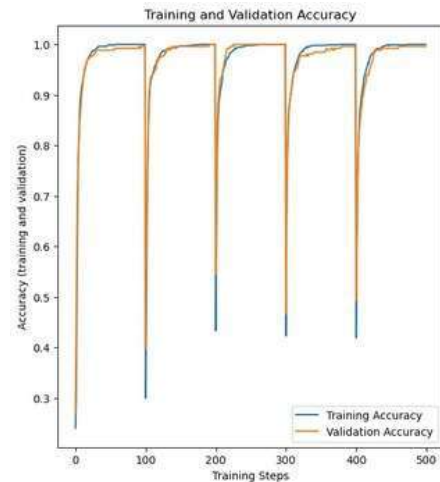


Fig 9. DenseNet121 Accuracy Training and Validation Results

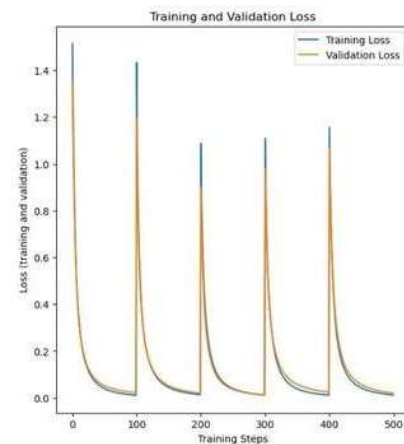


Fig 10. DenseNet121 Training and Validation Loss Results

The DenseNet121 model's output illustrates the outcomes of k-fold cross-validation for a five-fold machine learning model. Extremely high performance is demonstrated by the 5-fold cross-validation model evaluation findings. On Fold 1, the model recorded a loss of 0.0210 with an accuracy of 99.62%. Fold 2 shows a loss of 0.0187 with an accuracy of 100.00%, whereas Fold 3 records a loss of 0.0112 with an accuracy of 100.00%. 99.62% accuracy is lost in Fold 4 with a loss of 0.0234, and 99.62% accuracy is lost in Fold 5 with a loss of 0.0198. An average accuracy of 99.77% and an average loss of 0.0188 were found after calculating all

fold. These findings show that the model operates exceptionally effectively and consistently over the range of evaluated folds, with nearly perfect accuracy and a very low error rate.

4) Model MobileNetV2

Based on the results from training the MobileNetV2 model shown in Figures 9 and 10, the model was trained using a configuration of 100 epochs. It employed 5 subsets, or folds, during the training process. The training aimed to optimize the model's performance. The visuals illustrate the outcomes of the training sessions. Overall, the model was carefully evaluated across multiple folds to ensure robustness.

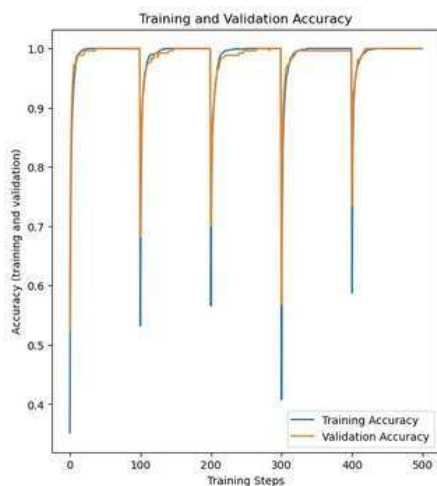


Fig 11. MobileNetV2 Accuracy Training and Validation Results

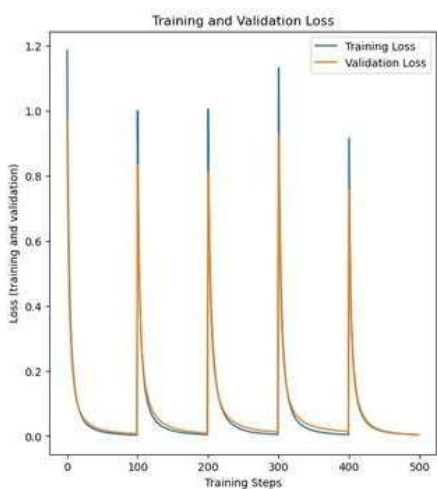


Fig 12. MobileNetV2 Loss Training and Validation Results

It displays the outcomes of k-fold cross-validation for a machine learning model with five folds using

the MobileNetV2 model's outputs. Very outstanding performance is shown by the 5-fold cross-validation results of the model evaluation. The model yielded an accuracy of 100.00% and a loss of 0.0068 on Fold 1. With the same precision of 100.00%, Fold 2 yields a loss of 0.0084. The loss on Fold 3 was accurately recorded at 0.0121, or 100.00%. Fold 5 obtained the lowest loss of 0.0036 with an accuracy of 100.00%, while Fold 4 recorded a loss of 0.0134 with a 99.62% accuracy. An average accuracy of 99.92% and an average loss of 0.0088 were found by averaging all folds. These findings demonstrate the model's extremely high performance in comparison to other models, demonstrating its efficacy and dependability in forecasting. Consequently, it is believed that using this model is the best option.

Table 2. Comparison of Training Models

Model	Accuration (%)	Loss
ResNet50V2	99.47	0.0218
NASNetMobile	98.79	0.0481
DenseNet121	99,77	0.0188
MobileNetV2	99,92	0.0088

The total performance of the ResNet50V2, NASNetMobile, DenseNet121, and MobileNetV2 training models is compared in Table 2. This table shows that the MobileNetV2 model has the lowest average loss value (0.0088) and the best average accuracy (99.92%). As a result, out of all the trained models, the MobileNetV2 model is regarded as the best. To highlight their superiority in terms of accuracy and overall performance, the best model results are printed bold.

3.2. Confusion Matrix Evaluation Results

The model is used to predict the classification of crystal guava test image data. Following this, the prediction outcomes are assessed with a confusion matrix. This matrix includes key metrics such as accuracy, recall, precision, and the F1-score. These evaluations help determine the model's effectiveness. By analyzing these metrics, we can gain insights into the model's performance. Ultimately, this process aids in refining the classification results.

1) Confusion Matrix results on the DenseNet121 Model

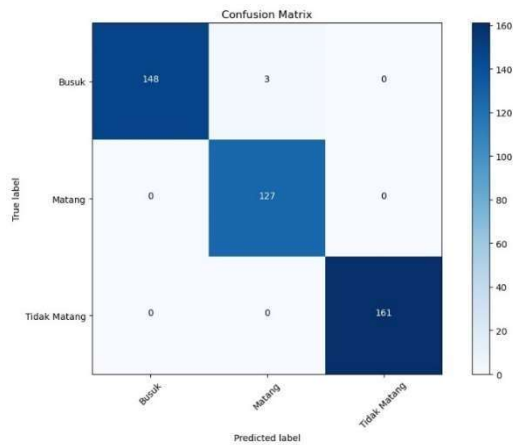


Fig 13. Confusion Matrix DenseNet121

From the confusion matrix of the DenseNet121 model in Figure 13 it can be concluded:

- a. Of a total of 151 Rotten images, 148 were correctly predicted as Rotten, 3 images were predicted as Ripe, and 0 images were predicted as Unripe.
- b. Of the total 127 Ripe images, 127 were correctly predicted as Ripe, 0 images were predicted as Rotten, and 0 images were predicted as Not Ripe.
- c. Of the total 161 Unripe images, 161 were correctly predicted as Not Ripe, 0 images were predicted as Rotten, and 0 images were predicted as Ripe.

2) Confusion Matrix results on the MobileNetV2 Model

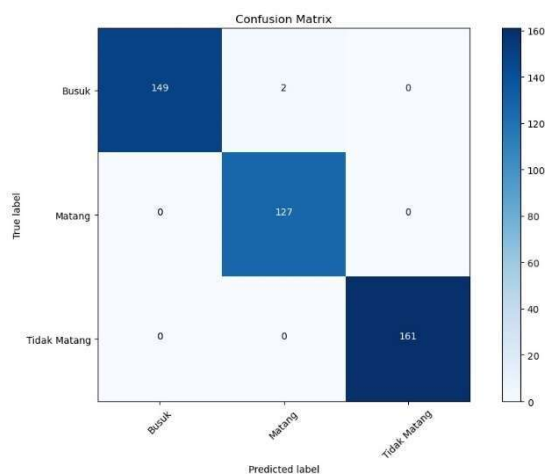


Fig 14. Confusion Matrix MobileNetV2

The following conclusions can be drawn from the MobileNetV2 model's confusion matrix in Figure 14:

- a. Out of 151 photos that were classified as Rotten, 149 were accurately forecasted as such, 2 as Ripe, and 0 as Unripe.
- b. Out of the 127 photos that were classified as ripe, 127 were accurately forecasted as such, 0 as rotten, and 0 as not ripe.
- c. Out of the 161 unripe photos in total, 161 were accurately predicted as Not Ripe, 0 as Rotten, and 0 as Ripe.

3) Confusion Matrix results on the DenseNet121 Model

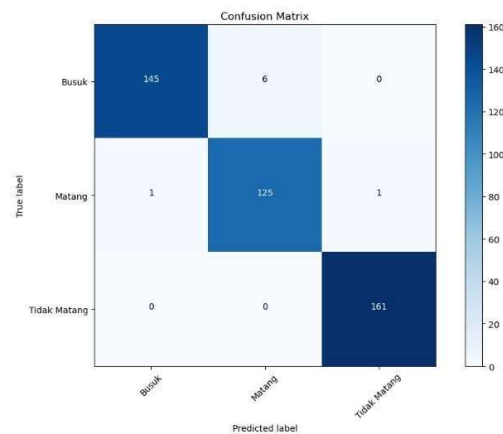


Fig 15. Confusion Matrix DenseNet121

The following conclusions can be drawn from the DenseNet121 model's confusion matrix in Figure 15:

- a. Out of the 151 photos that were categorized as Rotten, 145 were accurately forecasted as such, 6 as Ripe, and 0 as Not Ripe.
- b. One image was classified as Rotten, one as Not Ripe, and 125 of the 127 Ripe images were accurately forecasted as such.
- c. Out of the 161 unripe photos in total, 161 were accurately predicted as Not Ripe, 0 as Rotten, and 0 as Ripe.

4) Confusion Matrix results on the Resnet50V2 Model

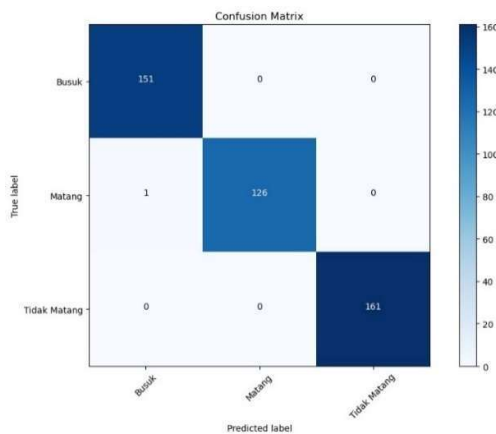


Fig 16. Confusion Matrix Resnet50V2


The following conclusions can be drawn from the Resnet50V2 model's confusion matrix in Figure 16:

- a) Out of 151 photos that were categorized as Rotten, 151 were accurately predicted as such, 0 as Ripe, and 0 as Unripe.
- b) 126 of the 127 ripe photos were accurately projected to be ripe, one was expected to be rotten, and none were forecast to be not ripe.
- c) Out of the 161 unripe photos in total, 161 were accurately predicted as Not Ripe, 0 as Rotten, and 0 as Ripe.

3.3. Application Functionality Analysis

The application for classifying Crystal Guava (Psidium Guajava) using a convolutional neural network method is designed for the Android operating system. It has been installed on an Infinix Note 12 2023 device with the specified specifications.

Table 3. Device specifications

No	Device	Spesification
1		Type of smartphone: Infinix 12 2023 chipset MediaTek Helio G99 8GB RAM and 256GB of storage System software: Android 12 XOS Display Dimensions: 6.78 inches; Pixel Resolution: 1080 x 240

The findings of evaluating the application display on various devices used in table 3 are summarized as follows:

- 1) Page with Splash Screens after around three seconds, the Splash Screen page is successfully displayed by the system, and Figure 15 shows the Onboarding Screen page.



Fig 17. Splash Screen

- 2) Dashboard Page



Fig 18. Dashboard Page

The dashboard screen page is successfully shown by the system. Figure 18 shows the button features on the dashboard screen, which include home and Crystal Guava Classification.

- 3) Examining the Classification Display for Crystal Guavas

The user can categorize the ripeness of crystal guava by taking a picture or choosing one from the gallery using the two primary options on the home screen of this view, as shown in Figure 4.19.



Fig 19. Crystal Guava Classification Display Testing

- 4) Scan Page Testing

The system successfully opens the front and rear cameras. Users can take photos directly as in Figure 20.



Fig 20. Scan Page Testing

- 5) Testing of Local Guava Classification Results

In order to examine the local guava fruit, ripe samples of the fruit were collected. Using the application, the

system was able to determine the ripeness of the local guava fruit and produced results with a ripe label, as shown in Figure 21.



Fig 21. Testing Local Guava Classification Results

6) Testing of Crystal Guava Classification Results
The system succeeded in detecting the ripeness level of crystal guava fruit using photos directly or from the gallery as in Figure 22.



Fig 22. Testing of Crystal Guava Classification Results

4. Discussions

The use of multiple CNN architectures in this study—ResNet50V2, DenseNet121, NASNetMobile, and MobileNetV2—demonstrates a comparative approach to identifying the most effective model for classifying the ripeness of crystal guava images. Each model was individually trained on the dataset with extensive preprocessing and through various CNN layers for feature extraction, followed by optimization through a loss function and weight updates. This iterative process, reinforced by 5-fold cross-validation and early stopping, allowed each model to demonstrate its strengths and weaknesses, resulting in comprehensive accuracy and loss scores for each fold. The MobileNetV2 model, achieving an impressive 99.92% accuracy with a minimal average loss of 0.0088, outperformed the other models, indicating its high reliability and robustness in handling the dataset.

Further evaluation using confusion matrices for each model reveals the classification accuracy for each ripeness category (Rotten, Ripe, Not Ripe) and validates the effectiveness of these models in real-world applications. MobileNetV2 and DenseNet121, with near-perfect results across all categories, are especially effective in distinguishing between different levels of ripeness. Additionally, the

classification functionality is successfully implemented on an Android application, offering practical usability with the ability to capture or select images and determine guava ripeness efficiently. This usability on a mobile platform, combined with high classification accuracy, supports the MobileNetV2 model as the best choice for deploying a reliable and user-friendly tool for guava ripeness classification.

5. Conclusion

Based on the results of the thesis titled “Development of a Guava Classification Application (Psidium Guajava) Using Convolutional Neural Network Method on a Mobile Platform Based on Digital Images,” the following conclusions can be concluded that this research successfully designed a machine learning model based on Convolutional Neural Network (CNN) that is accurate and effective for classifying the ripeness of guava based on its skin color, achieving an average accuracy of 99.92% and the lowest average loss of 0.0088 with the MobileNetV2 base model. The comparative classification results indicate that the approach applied to guava is more precise than that for local guava, highlighting the significant potential of AI technology in sorting various types of fruits. The application was successfully implemented into a mobile app, with user satisfaction survey results showing a satisfaction percentage of 90.19%, qualifying as very satisfied.

References

- [1] Akani, S. A., & Kelvyn. (2023). USAGE OF FLUTTER FRAMEWORK IN DESIGN AND DEVELOP MLEARNING APPLICATION AND ITS EFFECTIVENESS ANALYSIS AMONG WORKERS IN BATAM CITY. *JURNAL TEKNOLOGI INFORMASI DAN KOMUNIKASI*, 14(1). doi: 10.51903/jtikp.v14i1.536
- [2] Albarrak, K., Gulzar, Y., Hamid, Y., Mehmood, A., & Soomro, A. B. (2022). A Deep Learning-Based Model for Date Fruit Classification. *Sustainability*, 14(10), 6339. doi: 10.3390/su14106339
- [3] Xue, G., Liu, S., & Ma, Y. (2023). A hybrid deep learning-based fruit classification using attention model and convolution autoencoder. *Complex & Intelligent Systems*, 9(3), 2209–2219. doi: 10.1007/s40747-020-00192-x

- [4] Alsirhani, A., Siddiqi, M. H., Mostafa, A. M., Ezz, M., & Mahmoud, A. A. (2023). A Novel Classification Model of Date Fruit Dataset Using Deep Transfer Learning. *Electronics*, 12(3), 665. doi: 10.3390/electronics12030665
- [5] Yang, B., & Xu, Y. (2021). Applications of deep-learning approaches in horticultural research: a review. *Horticulture Research*, 8(1), 123. doi: 10.1038/s41438-021-00560-9
- [6] Jin, X., Che, J., & Chen, Y. (2021). Weed Identification Using Deep Learning and Image Processing in Vegetable Plantation. *IEEE Access*, 9, 10940–10950. doi: 10.1109/ACCESS.2021.3050296
- [7] Arvi Arkadia, Sekar Ayu Damayanti, & Desta Sandya Prasvita. (2021). Klasifikasi Buah Mangga Badami Untuk Menentukan Tingkat Kematangan dengan Metode CNN. 1, 158–165.
- [8] Andi Baso Kaswar, Andi Akram Nur Risal, Fatiah, & Nurjannah. (2020). KLASIFIKASI TINGKAT KEMATANGAN BUAH MARKISA MENGGUNAKAN JARINGAN SYARAF TIRUAN BERBASIS PENGOLAHAN CITRA DIGITAL. *JESSI*, 01, 1–8. Retrieved from <https://ojs.unm.ac.id/JESSI/index>
- [9] Aminatus Syarifah, Aditya Akbar Riadi, & Arief Susanto. (2022). Klasifikasi Tingkat Kematangan Jambu Bol Berbasis Pengolahan Citra Digital Menggunakan Metode K-Nearest Neighbor. 3, 27–35. doi: <http://dx.doi.org/10.37438/jimp.v7i1.417>
- [10] Tammina, S. (2019). Transfer learning using VGG-16 with Deep Convolutional Neural Network for Classifying Images. *International Journal of Scientific and Research Publications (IJSRP)*, 9(10), p9420. doi: 10.29322/IJSRP.9.10.2019.p9420
- [11] Rahayu, M. I., Jaenal, R., & Risyandi, M. H. (2024). IDENTIFIKASI TANAMAN OBAT HERBAL BERBASIS CITRA. *Jurnal Teknologi Informasi Dan Komunikasi*, 12(2), 57–63. doi: 10.58761/jurtikstmikbandung.v12i2.5763
- [12] Jaelani Akbar, M., Sardjono, W., Cahyanti, M., Ericks, D., & Swedia, R. (n.d.). PERANCANGAN APLIKASI MOBILE UNTUK KLASIFIKASI SAYURAN MENGGUNAKAN DEEP LEARNING CONVOLUTIONAL NEURAL NETWORK. Fakultas Teknologi Industri, 2(100), 300–306.
- [13] Arockia Raj, J., Dinakaran, S. J., & kumar, K. (2021). Android Application using Android Studio Android Application using Android Studio. In *Turkish Online Journal of Qualitative Inquiry (TOJQI)* (Vol. 6). Retrieved from <https://www.researchgate.net/publication/358165232>
- [14] Carolina, A. (2023). SISTEM PENJUALAN DENGAN PENGENALAN PRODUK SECARA OTOMATIS MENGGUNAKAN METODE YOLO. 8(2).
- [15] Choudhury, A., Talukdar, A. K., & Sarma, K. K. (2015). A review on vision-based hand gesture recognition and applications. In *Intelligent Applications for Heterogeneous System Modeling and Design* (pp. 256–281). IGI Global. doi: 10.4018/978-1-4666-8493-5.ch011
- [16] Femling, F., Olsson, A., & Alonso-Fernandez, F. (2018). Fruit and Vegetable Identification Using Machine Learning for Retail Applications. 2018 14th International Conference on Signal-Image Technology & Internet-Based Systems (SITIS), 9–15. doi: 10.1109/SITIS.2018.00013
- [17] Goularas, D., & Kamis, S. (2019). Evaluation of Deep Learning Techniques in Sentiment Analysis from Twitter Data. 2019 International Conference on Deep Learning and Machine Learning in Emerging Applications (Deep-ML), 12–17. doi: 10.1109/Deep-ML.2019.00011
- [18] Gulzar, Y. (2023). Fruit Image Classification Model Based on MobileNetV2 with Deep Transfer Learning Technique. *Sustainability*, 15(3), 1906. doi: 10.3390/su15031906
- [19] Inoue, K. (2019). Expressive Numbers of Two or More Hidden Layer ReLU Neural Networks. 2019 Seventh International Symposium on Computing and Networking Workshops (CANDARW), 129–135. doi: 10.1109/CANDARW.2019.00031
- [20] Kuhlman, D. (2009). *A Python Book: Beginning Python, Advanced Python, and Python Exercises*. Retrieved from <http://www.davekuhlman.org>
- [21] Maulana, F. F., & Rochmawati, N. (2020). Klasifikasi Citra Buah Menggunakan Convolutional Neural Network. *Journal of Informatics and Computer Science (JINACS)*,

- 1(02), 104–108. doi: 10.26740/jinacs.v1n02.p104-108
- [22] Nandan Challapalli, S. S., Mishra, G., Pachauri, Y., Mishra, A., Kumar, S. R., & Kumar, L. (2023). Comparing TensorFlow.js and TensorFlow in Python: An Accessibility and Usage Analysis. 2023 6th International Conference on Contemporary Computing and Informatics (IC3I), 6, 250–254. doi: 10.1109/IC3I59117.2023.10397702
- [23] Nurhikmat, T. (2018). IMPLEMENTASI DEEP LEARNING UNTUK IMAGE CLASSIFICATION MENGGUNAKAN ALGORITMA CONVOLUTIONAL NEURAL NETWORK (CNN) PADA CITRA WAYANG GOLEK. doi: 10.13140/RG.2.2.10880.53768
- [24] Pressman, R. S. (2010). Rekayasa perangkat lunak : pendekatan praktisi (buku 1) (7th ed.). ANDI.
- [25] Ramadhan, M. A. (2022). COMPUTER VISION UNTUK MENGETAHUI KEMATANGAN JAMBU KRISTAL MENGGUNAKAN METODE CONVOLUTIONAL NEURAL NETWORK.
- [26] Ratnasari, E. K., Mentari, M., Dewi, R. K., & Hari Ginardi, R. V. (2014). Sugarcane leaf disease detection and severity estimation based on segmented spots image. Proceedings of International Conference on Information, Communication Technology and System (ICTS) 2014, 93–98. doi: 10.1109/ICTS.2014.7010564
- [27] Şafak, E., & Barışçı, N. (2024). Detection of fake face images using lightweight convolutional neural networks with stacking ensemble learning method. PeerJ Computer Science, 10, e2103. doi: 10.7717/peerj-cs.2103
- [28] Wibowo, A., Hermanto, D. M. C., Lestari, K. I., & Wijoyo, H. (2021). Deteksi Kematangan Buah Jambu Kristal Berdasarkan Fitur Warna Menggunakan Metode Transformasi Ruang Warna Hsv (Hue Saturation Value) Dan K-Nearest Neighbor. INCODING: Journal of Informatics and Computer Science Engineering, 1(2), 76–88. doi: 10.34007/incoding.v2i1.131
- [29] Zhou, L., Zhang, C., Liu, F., Qiu, Z., & He, Y. (2019). Application of Deep Learning in Food: A Review. Comprehensive Reviews in Food Science and Food Safety, 18(6), 1793–1811. doi: 10.1111/1541-4337.1249. Quality of Service (QoS).” Etsi Tr 101 329 V2.1.1, 1, 1–37.

