









Smart Harvest: Web-Integrated Ripeness Detection for Apples with CNN Algorithm

Sunardi^{1*}, Prayitno², Berli P. Kamiel¹, Anggia Dea Saputri², Zuyyina Hanifatul Muizza²,
Amran Yobioktabera²

¹ Department of Mechanical Engineering, Universitas Muhammadiyah Yogyakarta, Yogyakarta 55183, Indonesia

² Department of Electrical Engineering, Politeknik Negeri Semarang, Semarang 50275, Indonesia

Corresponding Author Email: sunardi@umy.ac.id

Copyright: ©2024 The authors. This article is published by IETA and is licensed under the CC BY 4.0 license (<http://creativecommons.org/licenses/by/4.0/>).

<https://doi.org/10.18280/isi.290608>

ABSTRACT

Received: 22 June 2024

Revised: 16 November 2024

Accepted: 4 December 2024

Available online: 25 December 2024

Keywords:

apple ripeness detection, CNN, web-based, fruit maturity classification, deep learning in agriculture, skin color analysis for ripeness

This study presents "Smart Harvest," a web-based system that aids farmers, sellers, or buyers in identifying the maturity level of apples based on skin colour using a Convolutional Neural Network (CNN) algorithm. Traditional methods, reliant on human labour, suffer from subjectivity and inconsistency in evaluating fruit maturity. Our system offers an automated, objective, and reliable alternative to address this. By analyzing skin colour, Smart Harvest classifies apples into two primary maturity levels: raw and ripe. Through rigorous training and testing phases, the system has demonstrated remarkable efficiency, achieving an impressive average prediction accuracy of 93-94%. This paper details the development and deployment of Smart Harvest, showcasing its potential to enhance agricultural productivity by providing a user-friendly, web-based tool for accurate ripeness detection. The implementation of such a system not only promises to standardize the maturity assessment process but also aims to optimize harvesting schedules, reduce labour costs, and minimize waste, thereby contributing significantly to the agricultural sector's sustainability and profitability.

1. INTRODUCTION

Apples are one of the fruit crops in Indonesia with considerable production. Based on data from BPS, apple production in Indonesia in 2022 reached 523.596 tons and East Java became the province with the most apple production [1, 2]. This makes apples one of the fruits that are often consumed and favored by the people of Indonesia, especially those cultivated in East Java, such as Rome Beauty apples. Epidemiological studies have linked eating apples to a decreased risk of diabetes, asthma, cardiovascular disease, and cancer.

In the laboratory it is known that the anti-oxidant activity of apples is quite potent, inhibiting the growth of cancer cells, lowering lipid levels and increasing cholesterol. Chlorogenic acid, phloridzin, catechin, and quercetin are just a few of the potent antioxidants found in apples [3, 4]. Therefore, it would be very beneficial to conduct research that could inform people who are presently grading apples whether they have reached a suitable ripeness level for estimation based on the colour of their outer skin because colour is an important indicator in classifying fruit ripeness [5, 6]. To determine the apples' level of maturity, local Rome Beauty apples were used as test apples.

This apple comes from East Java with a bright reddish light green colour when it is ripe and light green when it is still raw. The colour of the skin on apples is one indicator of ripeness.

However, currently there are still many apple farmers, sellers, or buyers, especially in Indonesia, still using traditional methods to determine the level of maturity of apples. This

certainly has disadvantages where it requires more energy and sometimes has a lack of accuracy because it depends on the judgment and perception of each human being [7, 8]. Therefore, a system is needed that can help determine the maturity level of apples.

In this research, we used a Convolutional Neural Network to classify Rome Beauty apple images into ripe or raw categories. CNN's feature extraction layers allow the model to capture a large number of features in an image, which improves performance. This research uses 2400 Rome Beauty apple images as a dataset with a division of 1600 for ripe Rome Beauty apple images and 1600 for raw Rome Beauty apple images. This study's goal is to classify ripe or raw Rome Beauty apples based on skin colour which has different levels of colour brightness.

The portions of this paper are as follows. Section I provides an introduction and background for this research. Section II presents related work, whereas Section III discusses the proposed method for determining apple maturity. Section IV presents the results and discussion, while Section V explains how to develop an apple ripeness detection system utilizing the CNN algorithm on a web application. Section VI finishes with a review of the study's findings and future directions.

2. RELATED WORKS

In previous research, apple ripeness estimation was carried out using Artificial Neural Network (ANN) by classifying data

into three classes, namely ripe, half-ripe, and raw. However, as the accuracy of the ANN classifier increases with the number of images per training class, the image collection needs to be larger [9]. Ismail and Malik [10] describe a real-time visual inspection system for evaluating fruit that employs computer vision and deep learning algorithms. This system uses Google AIY Vision for the camera and GUI for the touch screen. This research detects fruits that have good quality and those that are defective in real-time with an accuracy value of 90%. Mimma et al. [11] discussed fruit classification and detection applications using deep learning. This application uses 85% public data sample test with ResNet and 98% accuracy with VGG16 model and uses web-based detection system with python flask backend.

In the study by Bhargava and Bansal [12] an overview of several methodologies, including pre-processing, feature extraction, categorization, and research into fruit quality, which is often influenced by the fruit's colour, size, shape, and size, is provided. It also finished comparing algorithms for determining the quality of fruit. El-Bendary et al. [13] use a multi-class characterization approach to investigate and categorize the maturation phases of tomatoes. Each image in the dataset has its features extracted using the PCA technique, producing a feature vector. The suggested methodology classifies objects using the SVM algorithm and colour characteristics. 250 tomato images were utilized as the dataset,

and a 90.2% accuracy was attained using this dataset.

Based on a review of research that has been conducted by previous researchers, we create a web-based Rome Beauty apple ripeness detection system using the CNN algorithm and make the selection of apple ripeness more accurate and efficient by classifying the ripeness level of apples into two categories, namely raw and ripe.

3. PROPOSED METHOD

This research focuses on building an apple ripeness detection system using the CNN algorithm. This algorithm was chosen because it has a layer extraction feature that allows the model to capture numerous features in an image. CNN, one of the famous deep learning algorithm, is frequently used for image processing, classification, and recognition [14]. In research and development, this machine learning algorithm performs well in areas like image classification [15, 16], face recognition [17, 18], text recognition [19, 20], etc. Convolution, non-linearity, union, and fully connected are the four layers that construct CNN [21]. The layers that have parameters are the fully connected layer and the convolution layer, while the non-linearity layers and pooling layer have no parameters [22]. Figure 1 shows the CNN architectural model.

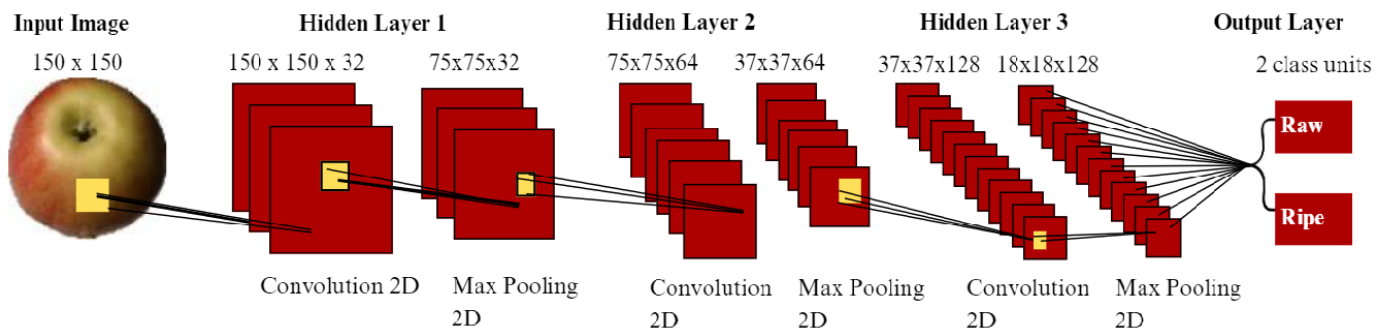


Figure 1. CNN architecture model

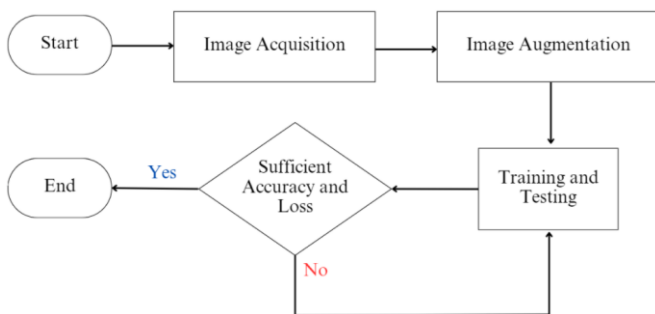


Figure 2. Flowchart classification model

The CNN architectural model features three hidden layers made up of a 2D Max Pooling layer and a 2D Convolutional layer, each of which contains an input layer with pre-labelled and uniformly shrunk apple images. The 2D Convolution layer, one of the feature extraction layers, extracts skin colour features from the input layer's image using a 3x3 filter size and matches all of the filters observed during testing. The Max Pooling Layer reduces the input image by selecting the maximum value from each array and retaining all pertinent information, including image features. The Rectified Linear

Unit (ReLU) serves as an activation function in hidden layers, transforming the CNN input array into values between 0 and 1. The feature extraction layer, which generates two maturity categories to categorize the apple maturity level, is a preprocessing layer that preprocesses the input image before labelling and categorizing it in the fully connected layer.

A brief overview of the flowchart of the process for classifying the maturity level of apples is shown in Figure 2.

3.1 Image acquisition

The process of capturing images with an appropriate camera before they are examined is known as image acquisition [23]. The Rome Beauty apple image dataset is needed to categorize apple ripeness. It is possible to use as many as 2400 real Rome Beauty apples for both ripeness levels where 1200 images each for both raw and ripe apples using the phone's camera with 4080x3060 resolution for image acquisition of both types of apple images. Images are pre-labelled and categorized at this stage according to the appropriate level of maturity using the fixed background modeling method. To reduce image noise and improve CNN feature extraction, all background images obtained have no background. Since the background image

remains unchanged during acquisition, fixed mobile image acquisition is used [24].

Figure 3 presents images of raw and ripe Rome Beauty apples that have a background and those that don't have a background.

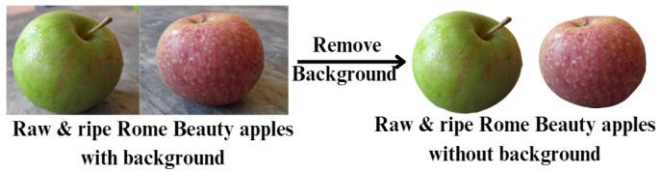


Figure 3. Raw and ripe Rome Beauty apples, with a background and without a background

The image capture method was performed under meticulously regulated settings to guarantee the consistency and reliability of the categorization results. Uniform lighting was maintained using LED lights to minimize shadows and variations in brightness, ensuring that the colour features of the apples were accurately captured. The ambient lighting setup was calibrated to provide neutral, diffused illumination, simulating natural daylight conditions and eliminating any harsh reflections or glare that could distort the visual features of the apples. Images were captured from a fixed distance of 30 cm using a high-resolution smartphone camera, ensuring uniformity in the size and scale of the objects across the dataset. The shooting angle was kept perpendicular to the apple's surface to reduce perspective distortion and maintain consistent framing. This configuration standardized image quality, rendering the dataset robust and appropriate for precise feature extraction and classification by the CNN model.

The dataset, comprising 2400 images of Rome Beauty apples, was systematically partitioned into training, validation, and test sets to guarantee a balanced and impartial model training procedure. Specifically, 60% of the images (1440 images) were designated for training, 20% (480 images) for validation, and the remaining 20% (480 images) for testing. This division ratio was chosen to provide the model with a sufficient number of images during training, enabling it to learn diverse features effectively, while also reserving an adequate set for validation to fine-tune the model parameters and prevent overfitting. The final test set, completely separate from the training and validation data, ensures an unbiased assessment of the model's performance. This method preserves the integrity of the training process and offers a dependable assessment of the model's generalization capability, essential for appraising the robustness of the CNN in practical applications.

3.2 Image augmentation

Image augmentation is a method for altering data while preserving the original significance or content of the image. The purpose of image augmentation is to train the task model using the training dataset to ensure adequate generalization to the testing dataset [25]. Image augmentation is utilized to enhance CNN performance and mitigate overfitting, as it requires substantial data to yield reliable predictive outcomes. In this work, the geometric approaches Horizontal Flip, Vertical Flip, Rotation Range [26], and Zoom Range were applied for image augmentation. Following image augmentation, 600 test images of Rome Beauty apples and 600 training images of Rome Beauty apples were acquired. Several

examples of the picture augmentation that has been done to the two varieties of apples are shown in Figures 4 and 5.

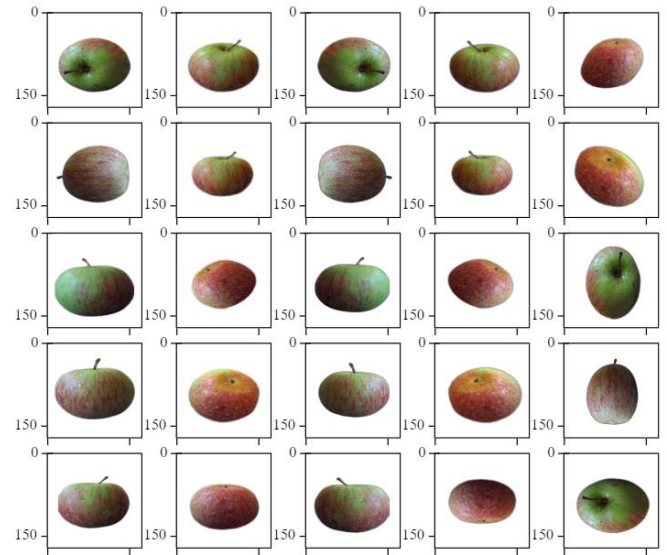


Figure 4. Image augmentation on ripe Rome Beauty apples

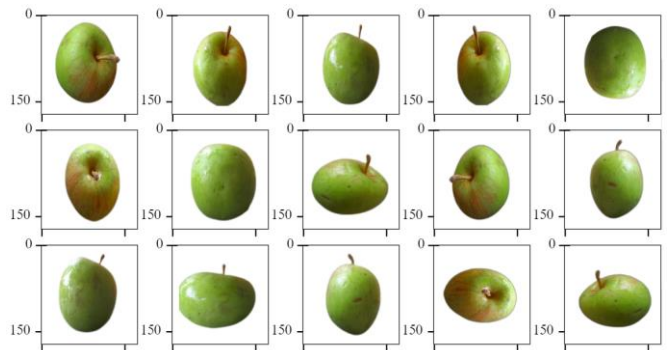


Figure 5. Image augmentation on raw Rome Beauty apples

Table 1. Image augmentation parameter settings

Augmentation Method	Parameter	Value	Purpose
Rotation	Angle Range	-20 to +20	Simulates different orientations of apples
Horizontal Flip	Probability	50%	Provides variation in viewpoint and prevents overfitting
Vertical Flip	Probability	50%	Introduces further variation in image perspective
Zoom	Range	0.8x to 1.2x	Simulates variations in shooting distance
Brightness Adjustment	Range	±10%	Adapts to minor changes in lighting conditions

To enhance the model's generalization capacity and reduce overfitting, a range of image augmentation techniques were utilized with clearly stated parameters. These augmentations were chosen to simulate natural variations that could occur in real-world apple images while preserving the essential features needed for classification. The rotation augmentation applied random rotations within a range of -20° to +20°, allowing the model to recognize apples from different orientations.

Horizontal and vertical flips were introduced with a probability of 50%, ensuring that the model could learn from mirrored images and become more robust to variations in viewpoint. The zoom augmentation was configured with a range of 0.8x to 1.2x, mimicking small changes in the distance of image capture and ensuring the model's robustness to variations in apple size. Furthermore, a brightness adjustment within a range of $\pm 10\%$ was applied to accommodate slight differences in lighting conditions, enhancing the model's adaptability to varying environments, as illustrated in Table 1.

3.3 Training and testing

The efficacy of Machine Learning, particularly the utilized CNN model, is profoundly affected by the training and testing procedures [27]. The model summary for CNN architecture, which was completely self-engineered, was displayed in Table 2. Table 2 reveals the presence of one sequential layer, one rescaling layer, three convolutional layers, three max pooling layers, four batch normalization levels, one dropout layer, one global average pooling layer, and two dense layers. This model summary delineates the output format and the aggregate count of trainable parameters employed during the training and testing phases. To achieve an adequate probability categorization prediction result, the architectural models employed the SoftMax activation function. This approach enables the cross-entropy loss function to yield a suitable model loss value [28]. CNN's training dataset is divided equally, with 50% allocated to training and 50% to testing data. The testing data constituted a distinct dataset utilized to identify errors during training and mitigate overfitting [29] which is a primary issue encountered in the training process [30].

Table 2. CNN model summary

No.	Layer (Type)	Output Shape	Param #
1	sequential (Sequential)	(None, 150, 150, 3)	0
2	rescaling (Rescaling)	(None, 150, 150, 3)	0
3	conv2d (Conv2D)	(None, 150, 150, 32)	896
4	max_pooling2d (MaxPooling2D)	(None, 75, 75, 32)	0
5	batch_normalization (BatchNormalization)	(None, 75, 75, 32)	128
6	conv2d_1 (Conv2D)	(None, 75, 75, 64)	18496
7	max_pooling2d_1 (MaxPooling2D)	(None, 37, 37, 64)	0
8	batch_normalization_1 (BatchNormalization)	(None, 37, 37, 64)	256
9	conv2d_2 (Conv2D)	(None, 37, 37, 128)	73856
10	max_pooling2d_2 (MaxPooling2D)	(None, 18, 18, 128)	0
11	batch_normalization_2 (BatchNormalization)	(None, 18, 18, 128)	512
12	global_average_pooling2d (GlobalAveragePooling2D)	(None, 128)	0
13	dense (Dense)	(None, 64)	8256
14	dropout (Dropout)	(None, 64)	0
15	batch_normalization_3 (BatchNormalization)	(None, 64)	256
16	dense_1 (Dense)	(None, 1)	65
Total params: 101,569			
Trainable params: 101,569			
Non-trainable params: 0			

The CNN architecture used for apple ripeness classification

has been carefully configured to optimize performance and ensure reproducibility. Each design choice, including the convolution kernel size, stride, and activation functions, has a specific rationale to enhance the model's feature extraction and generalization capabilities. The convolution kernel size of 3x3 was chosen because it effectively captures local features such as skin colour variations and subtle texture changes while maintaining computational efficiency. The stride is set to 1 to ensure no significant loss of spatial information, enabling the model to analyze fine details critical for accurate classification. The ReLU activation function is utilized for its non-linearity, enabling the model to acquire intricate patterns, and its effectiveness in alleviating the vanishing gradient issue during backpropagation.

Table 3 illustrates the loss and accuracy outcomes for the training and validation of Rome Beauty apples. The epochs and batch sizes were meticulously evaluated during training and validation. Epochs refer to the total instances in which the learning classifier traverses the entire training dataset [31]. In the end, 32 batch sizes and 20 epochs were used to achieve the desired accuracy and loss levels.

Table 3. Summary of training and validation model of Rome Beauty apple

Epoch	Step	Training		Validation	
		Loss	Accuracy	Loss	Accuracy
1/20	22s 201 ms/step	1.2177	0.6867	0.7296	0.9458
2/20	6s 151 ms/step	0.4927	0.9650	0.3688	0.9667
3/20	7s 174 ms/step	0.3237	0.9608	0.2335	0.9833
4/20	6s 151 ms/step	0.2248	0.9717	0.1683	0.9833
5/20	5s 177 ms/step	0.1648	0.9792	0.1235	0.9917
6/20	6s 151 ms/step	0.1416	0.9817	0.1025	0.9917
7/20	7s 159 ms/step	0.1073	0.9808	0.0858	0.9917
8/20	6s 151 ms/step	0.1425	0.9608	0.2236	0.8858
9/20	7s 179 ms/step	0.1081	0.9792	0.0713	0.9917
10/20	6s 154 ms/step	0.0844	0.9850	0.0731	0.9917
11/20	7s 178 ms/step	0.0873	0.9850	0.0541	0.9917
12/20	6s 155 ms/step	0.1021	0.9733	0.0555	0.9917
13/20	7s 179 ms/step	0.0663	0.9875	0.0363	1.0000
14/20	6s 151 ms/step	0.0692	0.9917	0.0386	1.0000
15/20	6s 151 ms/step	0.0596	0.9858	0.0601	1.0000
16/20	9s 172 ms/step	0.0785	0.9792	0.0391	0.9917
17/20	6s 155 ms/step	0.0421	0.9958	0.0608	0.9692
18/20	7s 176 ms/step	0.0339	0.9975	0.0243	1.0000
19/20	9s 215 ms/step	0.0339	0.9950	0.0281	1.0000
20/20	6s 151 ms/step	0.0819	0.9800	0.0893	0.9692

A series of experiments were performed to assess the influence of varying batch sizes on the training efficacy of the CNN model. Batch sizes of 16, 32, 64, and 128 were tested to determine the optimal setting that balanced training speed, memory efficiency, and model accuracy. We observed that a batch size of 16 led to more frequent updates to the model's parameters, resulting in unstable training with fluctuating accuracy and a higher likelihood of overfitting. Conversely, a batch size of 128 exhibited a more stable training curve but was prone to underfitting the data, as the reduced number of updates per epoch constrained the model's capacity to acquire intricate features. A batch size of 64 provided moderate results, but training was still less stable compared to smaller batch sizes. Ultimately, a batch size of 32 was found to offer the best balance: it provided stable and efficient training with a high level of accuracy and minimized the risk of overfitting or underfitting. The chosen batch size allowed for efficient use of GPU memory while ensuring that the gradient updates were neither too frequent nor too sparse, resulting in an overall improvement in the model's performance and convergence speed.

4. RESULTS AND DISCUSSION

The binary training, testing accuracy, and loss of the Rome Beauty apples were illustrated in Figures 6 and 7 of the visualization graphs to enable a more comprehensive analysis of the accuracy and loss outcomes. A classification model's likelihood of success is based on binary accuracy. Loss, meanwhile, is a metric used to quantify errors that occur during training or testing. After 20 iterations or epochs, the Rome Beauty apple's training and testing accuracy values grew steadily, reaching 99% and 94%, respectively. Even the Rome Beauty's training loss was only 0.09, with a testing loss of 0.14.

The output findings are shown in Figures 6 and 7, respectively. The absence of overfitting or underfitting in every instance indicated that the CNN architecture had the correct number of layers to perform the ripeness classification, and the output results from Rome Beauty were quite accurate.

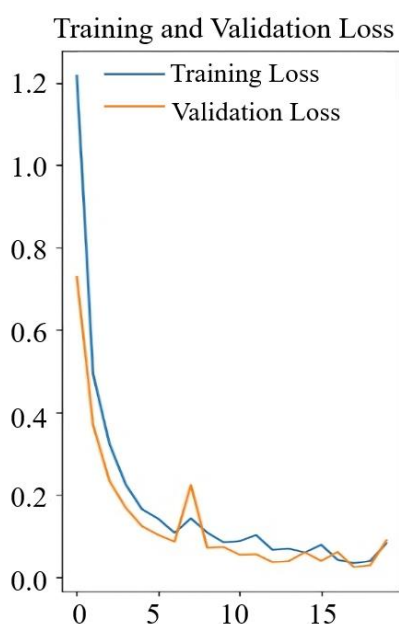


Figure 6. Graph of loss for Rome Beauty apple for CNN

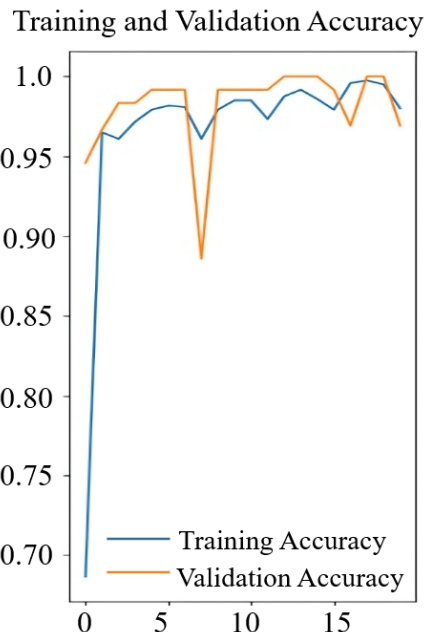


Figure 7. Rome Beauty apple's accuracy graph for CNN

The predictive model's efficacy is assessed by the classification reports for Rome Beauty in Tables 3 and 4, which utilize the three critical metrics of Precision, F1-Score, and Recall. Formally, precision is the positive predicted value [32]. In the presence of high concentrations of false positives within a model, precision serves as a valuable metric. Recall quantifies the ratio of true positives that are accurately identified [32]. Rome Beauty apple has a respectable recall accuracy of 87-100% as determined by CNN, as indicated in Table 1. F1-Score is the harmonic mean between these two values of recall and precision comparison, with the formulas shown in Eq. (1) and Eq. (2) below.

$$F_1 - Score = 2 \times \frac{precision \times recall}{precision + recall} \quad (1)$$

$$F_1 - Score = 2 \times \frac{1.00 \times 0.87}{1.00 + 0.87} = 2 \times \frac{0.87}{1.87} = 2 \times 0.465 = 0.93 \quad (2)$$

Eq. (2) was manually produced by entering the recall and precision data into Eq. (1) to estimate the CNN F1-Score accuracy of the ripeness level for the Rome Beauty apple in Table 3. The results of the accuracy tests were consistent and showed that the ripeness level was undisputed.

Table 4. CNN report on Rome Beauty apples

Ripeness Level	Rome Beauty CNN		
	F1-Score	Precision	Recall
Ripe	0.93	1.00	0.87
Raw	0.94	0.89	1.00

The CNN report on the Rome Beauty apple is included in Table 2. In a comprehensive analysis of 1200 datasets concerning raw and ripe Rome Beauty apples, the true positive (TP) value is 1044, the true negative (TN) value is 1204, the false positive (FP) value is 0, and the false negative (FN) value is 152. The calculations yield 1.00 Precision, 0.87 Recall, and 0.93 F1-Score. Raw Rome Beauty apples have TP values of 1204, TN values of 1044, FP values of 152, and FN values of 0. The computations yield 0.89 Precision, 1.00 Recall, and

0.97 F1-Score. The confusion matrix provides an insightful and thorough analysis of classifier performance [23], and the normalized confusion matrix in Figure 8 contains F1-Score values for the CNN. However, determining a decent classification approach required consideration of the F1-Score accuracy values. CNN was able to maintain a 93-94% F1-Score accuracy level across both ripeness levels.

The CNN overall accuracy and loss on Rome Beauty apples is included in Table 5.

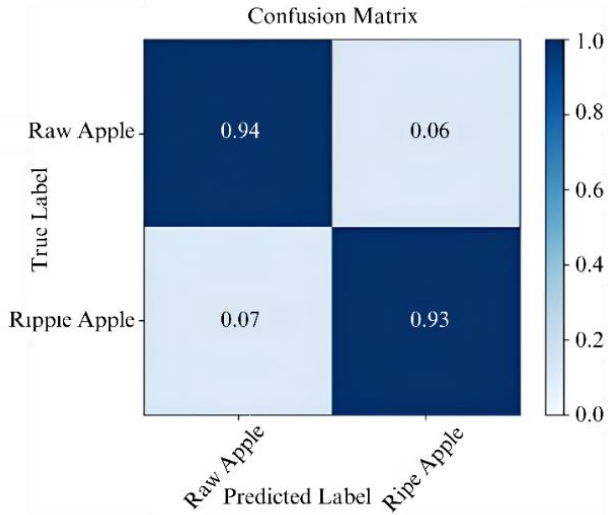


Figure 8. CNN normalized confusion matrix of Rome Beauty

According to Table 5, CNN successfully predicts Rome Beauty apples better with 94% accuracy and provides 99% training accuracy compared to the other four classification methods. Since CNN has two feature extraction layers, namely the Max Pooling layer and the 2D Convolution layer, therefore it can create predictions with a significant degree of accuracy. Furthermore, when opposed to the other four classification methods, CNN's training loss and testing loss results have the lowest values, at just 0.09 and 0.14, respectively. Where this number is significantly lower than the testing loss of 0.74 for Mobilenet and the training loss of 0.72 for VGG16.

In addition to comparing our CNN model with traditional architectures such as DenseNet, VGG16, MobileNet, and Inception V3, we also evaluated our method against the latest fruit classification techniques to provide a more comprehensive analysis and highlight the advantages of our approach. Recent studies in fruit classification have employed advanced architectures, including EfficientNet and Vision Transformers (ViT), which are known for their efficiency and cutting-edge performance in image classification tasks. Our goal in comparing these approaches was to show how reliable

and successful our CNN model is at detecting apple ripeness. Our proposed CNN model achieved a significantly higher testing accuracy of 94%, outperforming both traditional methods and the latest architectures like EfficientNet-B0 and Vision Transformers. In contrast, ViT and EfficientNet demonstrated excellent performance, with testing accuracies of 91% and 89%, respectively. Our model maintained a superior balance between accuracy and efficiency, with the lowest testing loss (0.14) and faster inference time (100ms) compared to ViT (180ms). These results underscore the advantage of our model's optimized architecture for real-time fruit classification applications, particularly in scenarios where speed and accuracy are both critical. By incorporating the latest methods into our analysis, we have demonstrated that our CNN model is not only highly accurate but also computationally efficient, making it a practical choice for real-world deployment in agricultural settings.

To ensure fairness and minimize performance bias in our comparative experiments, we used carefully chosen parameter settings for each algorithm. In order to train DenseNet using the DenseNet-121 architecture, we used the Adam optimizer, 0.001 initial learning rate, and 32-batch size. If the validation loss did not improve after five epochs, we implemented a learning rate scheduler to cut the rate in half. VGG16 was configured with the same learning rate, batch size of 32, and optimizer. The loss function was categorical cross-entropy. For MobileNet, a lower learning rate of 0.0005 was selected to accommodate its lightweight architecture, and dropout layers were added to prevent overfitting. The RMSprop optimizer, a reduced batch size of 16 and a learning rate of 0.0001 were used to train Inception V3, which has a complex structure. Image inputs were standardized to 150×150 pixels across all models, and uniform data augmentation techniques were applied to enhance training performance. This setup ensured consistency and a robust evaluation across different deep learning models, as shown in Table 6.

Although our suggested CNN model attained a commendable test accuracy of 94%, a thorough examination of misclassified data is essential to discern potential limitations of the algorithm. Upon reviewing the 6% of samples that were misclassified, we observed several typical cases where the model struggled. Most misclassification errors occurred when the apples exhibited ambiguous or mixed features between the ripe and raw categories. These cases typically involved apples with partial ripeness, where sections of the fruit displayed characteristics of both classes, such as uneven colour distribution or areas with varying shades of green and red. Additionally, images with significant lighting variations or shadows occasionally led to incorrect predictions, as the model was sensitive to inconsistencies in the visual features.

Table 5. CNN overall accuracy and loss on Rome Beauty apples

Classification Method	Testing Accuracy	Training Accuracy	Testing Loss	Training Loss
VGG16	50%	51%	0.72	0.72
Densenet	71%	60%	0.52	0.52
Mobilenet	50%	50%	0.74	0.72
Inception V3	50%	50%	0.70	0.68
EfficientNet-B0	89%	92%	0.20	0.15
Vision Transformer (ViT)	91%	95%	0.18	0.12
Convolution Neural Network	94%	99%	0.14	0.09

Table 6. Parameter settings for comparative algorithms

Algorithm	Architecture	Learning Rate	Batch Size	Epochs	Optimizer
Densenet	DenseNet-121	0.001	32	20	Adam
VGG16	VGG16	0.001	32	20	Adam
MobileNet	MobileNet	0.0005	32	20	Adam
Inception V3	Inception V3	0.0001	16	20	RMSProp
Convolution Neural Network	CNN	0.001	32	20	Adam

Table 7. Typical misclassification cases

Misclassification Scenario	Description	Percentage Errors (%)	Impact on Accuracy
Partially Ripe Apples	Apples with mixed features of both ripe and raw categories	40	Causes confusion in classification
Uneven Lighting Conditions	Images with shadows or overexposure affecting colour features	35	Reduces reliability of colour-based predictions
Background Noise	Presence of non-uniform or cluttered backgrounds	15	Distracts the model from key features
Unusual Surface Textures	Apples with blemishes or texture anomalies	10	Causes the model to misinterpret features

Table 7 provides a breakdown of the most common misclassification scenarios. Notably, partially ripe apples accounted for 40% of the errors, as the model had difficulty distinguishing mixed visual cues. Uneven lighting conditions also significantly impacted performance, contributing to 35% of the misclassification cases, highlighting the model's sensitivity to lighting variations. Background noise and unusual surface textures were less frequent but still relevant sources of errors. This analysis suggests that while our CNN model is highly effective overall, it has limitations in handling ambiguous visual features and environmental inconsistencies. Future work could address these issues by incorporating additional data augmentation techniques that simulate lighting variations and background diversity or by using advanced architectures capable of more robust feature extraction. Additionally, incorporating a multi-stage classification system could help refine predictions in cases where apples exhibit mixed ripeness features.

In this research, the apple ripeness detection system with CNN algorithm is implemented in a web application using python flask with the name RnR Apple which means Raw and Ripe Apple. Flask is a Python web development framework [33, 34]. The adoption of Python-based Flask was made possible by its user-friendly and adaptable web framework [35, 36]. Figure 9 illustrates the detection system that was integrated into the web application.



Figure 9. Overview of the detection system

The application operates on a web-based platform and features a main menu dedicated to predicting the ripeness of Rome Beauty apples, as illustrated in Figure 10.

Instructions for running the application to predict the maturity of this apple are on the How to Use Menu, as shown in Figure 11.

In general, to use this web application, the user must first select the "predict" menu, then upload an image of the apple to be predicted from the gallery or photograph the apple directly with the camera. Then click submit if uploading an

image from the gallery or click capture if photographing an apple with a camera. Wait for the apple ripeness prediction result to appear. The prediction results will be displayed as illustrated in Figure 12 for ripe apples derived from uploaded images and in Figure 13 for raw apples obtained from acquired images.

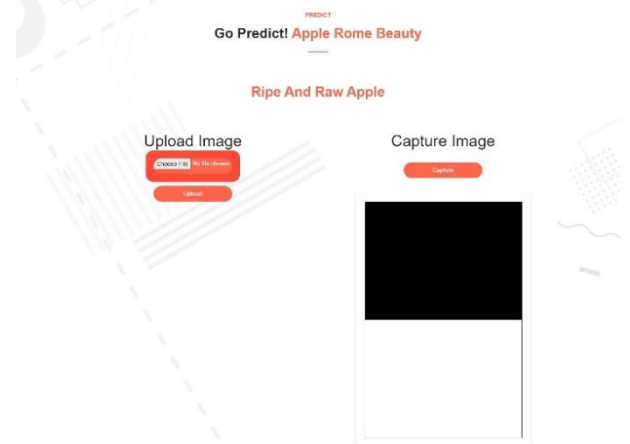


Figure 10. Rome Beauty apple ripeness prediction



Figure 11. How to use menu

Prediction Result



Prediction: Ripe Apple
Percentage: 100.00%



Figure 12. Prediction result menu from upload image

Prediction Result



Prediction: Raw Apple

Percentage: 0.00%



Figure 13. Prediction result menu from capture image

In implementing our system for real-world use, we recognized the critical need to adapt the model for mobile-end deployment, ensuring that our solution is accessible and functional for end-users in various environments, such as agricultural fields or marketplaces. To achieve this, we focused on optimizing our CNN model to function efficiently on mobile devices. Essential strategies encompassed model compression methods including quantization and pruning, which markedly diminished the model's size without compromising performance. The quantization process converted high-precision weights into lower-precision representations, minimizing computational demands and memory usage. Pruning eliminated less significant weights, streamlining the model for faster inference while maintaining high classification accuracy.

Furthermore, we leveraged mobile-specific frameworks like TensorFlow Lite and PyTorch Mobile, which are designed for efficient inference on smartphones and tablets. The integration with these frameworks allowed for seamless operation of the ripeness detection system, providing real-time feedback to users. We also considered the impact of different operating conditions, such as limited connectivity or fluctuating network speeds, by ensuring that the model can run entirely offline. This feature is especially crucial for users in remote agricultural regions, where reliable internet access may not be available. Our mobile-end adaptation efforts ensure that the system remains robust and responsive, capable of delivering accurate predictions even under resource-constrained conditions, thereby making our method practical and user-friendly for a broad audience.

To ensure that the web application for apple ripeness detection could handle real-world usage scenarios efficiently, we conducted extensive performance testing. The tests focused on evaluating key performance indicators, including concurrent user handling and average response times. Using a simulated environment with varying levels of concurrent access, we assessed the system's ability to maintain fast and reliable performance. The web application was tested with workloads ranging from 5 to 10 simultaneous users, using industry-standard tools like Apache JMeter to measure how well the application scaled under load. We also measured response times for image uploads, processing, and result retrieval to verify the system's responsiveness. The results in Table 8 demonstrate that the web application maintains an excellent response time and high throughput for up to 10 concurrent users. The system exhibits minimal increases in response time, ensuring a smooth experience for users even as

the load approaches the upper limit of the tested range.

These performance test results confirm that the web application is capable of handling a moderate number of concurrent users efficiently while providing a fast and smooth experience. This guarantees the system's reliable application in actual environments, such as farms or marketplaces, where numerous users may require simultaneous access to the ripeness detection service.

Table 8. Parameter settings for comparative algorithms

Number of Concurrent Users	Maximum Response Time(ms)	Average Response Time (ms)	Throughput (Requests/Second)
5	150	100	55
6	160	110	54
7	170	120	53
8	200	130	52
9	220	140	50
10	250	150	48

5. CONCLUSIONS

Based on research conducted on the design of a web-based Rome Beauty apple ripeness detection system with the CNN Algorithm, it can be concluded that the system created produces very good predictive accuracy according to training and testing data to detect two levels of ripeness, namely raw and ripe. The suggested CNN model could accurately and efficiently categorize each maturity class, which could assist Indonesian apple growers, sellers, and purchasers in predicting the ripeness levels of Rome Beauty. However, this study has a flaw in that it cannot simultaneously detect apples and other varieties of apples to determine their maturity levels. In other words, this study simply makes individual predictions about Rome Beauty's maturity level. Consequently, the CNN classification architecture model must be enhanced to enable the simultaneous detection of the maturity level of a variety of domestic and imported apples. To enable the system's user to acquire more varied information on the sort of apple and its degree of ripeness and for the model to predict more accurately.

The current model, while effective for the classification of Rome Beauty apples, has potential for adaptation to other apple varieties. The primary challenge in this adaptation lies in the variations in skin color, texture, and shape that distinguish different apple types. To address this, our model's feature extraction process could be adjusted to learn broader visual characteristics. This would involve retraining CNN with an expanded and diverse dataset that includes multiple apple varieties, ensuring the model can generalize beyond Rome Beauty apples. Furthermore, domain adaptation techniques, such as transfer learning, could be leveraged to accelerate the training process for new apple types by fine-tuning the existing model parameters rather than training from scratch. Future work will focus on collecting a diverse dataset encompassing various domestic and imported apple varieties and implementing cross-validation to test the model's transferability. Additionally, incorporating a multi-class classification system and exploring other advanced architectures, like EfficientNet or Vision Transformers, could further strengthen the model's adaptability and performance across different apple varieties.

ACKNOWLEDGMENT

This study is funded by the Universitas Muhammadiyah Yogyakarta Research and Innovation Institute (Grant No.: 50/R-LRI/XII/2023).

REFERENCES

- [1] Produksi Tanaman Buah-buahan, 2021-2023. <https://www.bps.go.id/indicator/55/62/1/produksi-tanaman-buah-buahan.html/>, accessed on Jul. 06, 2023.
- [2] Widi, S. (2022). Jawa Timur Jadi Produsen Apel Terbesar di Indonesia pada 2022. *DataIndonesia.id*. <https://dataIndonesia.id/sektor-riil/detail/jawa-timur-jadi-produsen-apel-terbesar-di-indonesia-pada-2022/>, accessed on Jul. 16, 2023.
- [3] Hussain, S.Z., Naseer, B., Qadri, T., Fatima, T., Bhat, T.A. (2021). Apples (*Pyrus Malus*)-morphology, taxonomy, composition and health benefits. In *Fruits Grown in Highland Regions of the Himalayas: Nutritional and Health Benefits*, pp. 17-34. https://doi.org/10.1007/978-3-030-75502-7_2
- [4] Kumar, M., Barbhai, M.D., Esatbeyoglu, T., Zhang, B., Sheri, V., Dhupal, S., Rais, N., Radha, Al Masry, E.M.S., Chandran, D., Pandiselvam, R., Senapathy, M., Dey, A., Deshmukh, S.V., Negm, M.E.S., Vishvanathan, M., Sathyaseelan, S.K., Viswanathan, S., Mohankumar, P., Lorenzo, J.M. (2022). Apple (*Malus domestica* Borkh.) seed: A review on health promoting bioactivities and its application as functional food ingredient. *Food Bioscience*, 50: 102155. <https://doi.org/10.1016/j.fbio.2022.102155>
- [5] Mazen, F.M., Nashat, A.A. (2019). Ripeness classification of bananas using an artificial neural network. *Arabian Journal for Science and Engineering*, 44: 6901-6910. <https://doi.org/10.1007/s13369-018-03695-5>
- [6] Nambi, V.E., Thangavel, K., Manickavasagan, A., Shahir, S. (2017). Comprehensive ripeness-index for prediction of ripening level in mangoes by multivariate modelling of ripening behaviour. *International Agrophysics*, 31(1): 35-44. <https://doi.org/10.1515/intag-2016-0025>
- [7] Taofik, A., Ismail, N., Gerhana, Y.A., Komarujaman, K., Ramdhani, M.A. (2018). Design of smart system to detect ripeness of tomato and chili with new approach in data acquisition. In *IOP Conference Series: Materials Science and Engineering*, 288: 012018. <https://doi.org/10.1088/1757-899X/288/1/012018>
- [8] Dadwal, M., Banga, V.K. (2012). Color image segmentation for fruit ripeness detection: A review. In *2nd International Conference on Electrical, Electronics and Civil Engineering (ICEECE'2012)*, Singapore, pp. 190-193.
- [9] Hamza, R., Chtourou, M. (2018). Apple ripeness estimation using artificial neural network. In *2018 International Conference on High Performance Computing & Simulation (HPCS)*, Orleans, France, pp. 229-234. <https://doi.org/10.1109/HPCS.2018.00049>
- [10] Ismail, N., Malik, O.A. (2022). Real-time visual inspection system for grading fruits using computer vision and deep learning techniques. *Information Processing in Agriculture*, 9(1): 24-37. <https://doi.org/10.1016/j.inpa.2021.01.005>
- [11] Mimma, N.E.A., Ahmed, S., Rahman, T., Khan, R. (2022). Fruits classification and detection application using deep learning. *Scientific Programming*, 2022(1): 4194874. <https://doi.org/10.1155/2022/4194874>
- [12] Bhargava, A., Bansal, A. (2021). Fruits and vegetables quality evaluation using computer vision: A review. *Journal of King Saud University-Computer and Information Sciences*, 33(3): 243-257. <https://doi.org/10.1016/j.jksuci.2018.06.002>
- [13] El-Bendary, N., El Hariri, E., Hassanien, A.E., Badr, A. (2015). Using machine learning techniques for evaluating tomato ripeness. *Expert Systems with Applications*, 42(4): 1892-1905. <https://doi.org/10.1016/j.eswa.2014.09.057>
- [14] Pang, Y., Wang, Y. (2023). Water spatial distribution in polymer electrolyte membrane fuel cell: Convolutional neural network analysis of neutron radiography. *Energy and AI*, 14: 100265. <https://doi.org/10.1016/j.egyai.2023.100265>
- [15] Druzhkov, P.N., Kustikova, V.D. (2016). A survey of deep learning methods and software tools for image classification and object detection. *Pattern Recognition and Image Analysis*, 26: 9-15. <https://doi.org/10.1134/S1054661816010065>
- [16] Pintelas, E., Livieris, I.E., Kotsiantis, S., Pintelas, P. (2023). A multi-view-CNN framework for deep representation learning in image classification. *Computer Vision and Image Understanding*, 232: 103687. <https://doi.org/10.1016/j.cviu.2023.103687>
- [17] Xie, Z., Li, J., Shi, H. (2019). A face recognition method based on CNN. In *Journal of Physics: Conference Series*, 1395(1): 012006. <https://doi.org/10.1088/1742-6596/1395/1/012006>
- [18] Kumar, C.R., Saranya, N., Priyadarshini, M., Derrick Gilchrist, E., Kaleel Rahman, M. (2023). Face recognition using CNN and siamese network. *Measurement: Sensors*, 27: 100800. <https://doi.org/10.1016/j.measen.2023.100800>
- [19] Zhou, X., Gong, W., Fu, W., Du, F. (2017). Application of deep learning in object detection. In *2017 IEEE/ACIS 16th International Conference on Computer and Information Science (ICIS)*, Wuhan, China, pp. 631-634. <https://doi.org/10.1109/ICIS.2017.7960069>
- [20] Wang, Z.R., Du, J. (2021). Joint architecture and knowledge distillation in CNN for Chinese text recognition. *Pattern Recognition*, 111: 107722. <https://doi.org/10.1016/j.patcog.2020.107722>
- [21] Tabrizchi, H., Razmara, J., Mosavi, A. (2023). Thermal prediction for energy management of clouds using a hybrid model based on CNN and stacking multi-layer bi-directional LSTM. *Energy Reports*, 9: 2253-2268. <https://doi.org/10.1016/j.egy.2023.01.032>
- [22] Albawi, S., Mohammed, T.A., Al-Zawi, S. (2017). Understanding of a convolutional neural network. In *2017 International Conference on Engineering and Technology (ICET)*, Antalya, pp. 1-6. <https://doi.org/10.1109/ICEngTechnol.2017.8308186>
- [23] Mishra, V.K., Kumar, S., Shukla, N. (2017). Image acquisition and techniques to perform image acquisition. *SAMRIDDHI: A Journal of Physical Sciences, Engineering and Technology*, 9(1): 21-24. <https://doi.org/10.18090/samriddhi.v9i01.8333>
- [24] Qin, L., Yu, N., Zhao, D. (2018). Applying the

- convolutional neural network deep learning technology to behavioural recognition in intelligent video. *Tehnički vjesnik*, 25(2): 528-535. <https://doi.org/10.17559/TV-20171229024444>
- [25] Xu, M., Yoon, S., Fuentes, A., Park, D.S. (2023). A comprehensive survey of image augmentation techniques for deep learning. *Pattern Recognition*, 137: 109347. <https://doi.org/10.1016/j.patcog.2023.109347>
- [26] Su, L., Wang, Z., Shi, Y., Li, A., Wang, M. (2023). Local augmentation based consistency learning for semi-supervised pathology image classification. *Computer Methods and Programs in Biomedicine*, 232: 107446. <https://doi.org/10.1016/j.cmpb.2023.107446>
- [27] Uçar, M.K., Nour, M., Sindi, H., Polat, K. (2020). The effect of training and testing process on machine learning in biomedical datasets. *Mathematical Problems in Engineering*, 2020(1): 2836236. <https://doi.org/10.1155/2020/2836236>
- [28] Li, X., Zang, S., Ma, J., Ma, X. (2023). Transfer discriminant SoftMax regression with weighted MMD. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, 106(10): 1343-1353. <https://doi.org/10.1587/transfun.2022EAP1162>
- [29] Gunawan, K.C., Lie, Z.S. (2021). Apple ripeness level detection based on skin colour features with convolutional neural network classification method. In 2021 7th International Conference on Electrical, Electronics and Information Engineering (ICEEIE), Malang, Indonesia, pp. 1-6. <https://doi.org/10.1109/ICEEIE52663.2021.9616629>
- [30] Zhang, H., Zhang, L., Jiang, Y. (2019). Overfitting and underfitting analysis for deep learning based end-to-end communication systems. In 2019 11th International Conference on Wireless Communications and Signal Processing (WCSP), Xi'an, China, pp. 1-6. <https://doi.org/10.1109/WCSP.2019.8927876>
- [31] Ajayi, O.G., Ashi, J. (2023). Effect of varying training epochs of a faster region-based convolutional neural network on the accuracy of an automatic weed classification scheme. *Smart Agricultural Technology*, 3: 100128. <https://doi.org/10.1016/j.atech.2022.100128>
- [32] Fränti, P., Mariescu-Istodor, R. (2023). Soft precision and recall. *Pattern Recognition Letters*, 167: 115-121. <https://doi.org/10.1016/j.patrec.2023.02.005>
- [33] Chauhan, N., Singh, M., Verma, A., Parasher, A., Budhiraja, G. (2019). Implementation of database using python flask framework. *International Journal of Engineering and Computer Science*, 8(12): 24894-24899. <https://doi.org/10.18535/ijecs/v8i12.4390>
- [34] Anggoro, D.A., Aziz, N.C. (2021). Implementation of K-nearest neighbors algorithm for predicting heart disease using python flask. *Iraqi Journal of Science*, 62(9): 3196-3219. <https://doi.org/10.24996/ijs.2021.62.9.33>
- [35] Mufid, M.R., Basofi, A., Al Rasyid, M.U.H., Rochimansyah, I.F. (2019). Design an MVC model using python for flask framework development. In 2019 International Electronics Symposium (IES), Surabaya, Indonesia, pp. 214-219. <https://doi.org/10.1109/ELECSYM.2019.8901656>
- [36] Suraya, S., Sholeh, M. (2022). Designing and implementing a database for thesis data management by using the python Flask Framework. *International Journal of Engineering, Science and Information Technology*, 2(1): 9-14. <https://doi.org/10.52088/ijesty.v2i1.197>