E-ISSN: 2962-6110
P-ISSN: 2964-7843
JSRET
jsret.knpub.com
Journal of Scientific Research, Education, and Technology

Application of Convolutional Neural Networks for Tomato Fruit Ripeness Identification

Julio Ignasius Wangjaya Rumbekwan, Joko Aryanto
University of Technology Yogyakarta

ABSTRACT

The identification of tomato ripeness is essential in agriculture to ensure quality and reduce spoilage. Traditional methods that rely on human observation are often inaccurate. This research aims to develop a CNN-based system to accurately identify tomato ripeness, focusing on the ripe and unripe categories. The images were taken while the tomatoes were still on the plant stalk with data collection involving 1.000 images of tomatoes, obtained from Kaggle and taking pictures of tomatoes at the Poktan Welan Asri Petinggen garden tour site, Yogyakarta. The dataset was divided into training (700 images), testing (150 images), and validation (150 images). The experimental design uses the CNN model with image processing steps such as resizing, labeling, and data augmentation. Testing on this system achieved an accuracy of 92.67%. These findings demonstrate the effectiveness of CNN in detecting tomato ripeness, providing a reliable solution for farmers and agricultural stakeholders.

Keywords: *Human Observation, Identification, Tomato Ripeness, Plant Stalk, CNN*

Corresponding author
Name: Julio Ignasius Wangjaya Rumbekwan
Email: juliowangjaya@gmail.com

INTRODUCTION

Tomatoes are one of the most perishable agricultural products, and accurate ripeness detection is essential to ensure optimal quality and minimize tomato spoilage. However, traditional ripeness detection methods, which rely heavily on human observation, are prone to errors due to subjective judgment and environmental factors. These inaccuracies often result in early harvest or spoilage, which impacts farmer productivity and market quality.

The objective of this research is to develop an automated tomato ripeness detection system using Convolutional Neural Networks (CNN), which can classify tomatoes into two categories: ripe and unripe. This research hypothesizes that a well-constructed CNN model, trained with diverse tomato images, can outperform traditional manual observation in terms of accuracy and reliability.

This research is important because it introduces a technological solution that can help farmers and all authors conducting research in the field of tomato fruit farming, especially in regions like Indonesia, where agriculture remains a vital economic sector. By

utilizing machine learning techniques, this research contributes to improving the efficiency and effectiveness of agricultural practices, providing a scalable and accurate approach to detecting ripeness.

LITERATURE REVIEW

In the study conducted by (Febriyanti, 2024), on image processing for human skin diseases, there are several steps in image processing, namely resizing, labelling the dataset, and dividing into training data and testing data. The resizing process is necessary to resize the dimensions of the dataset used to speed up the training. Next, the step taken is labelling according to the folder name where the dataset is stored. After that, the data separation stage for training, testing, and validation is performed.

In classifying tomatoes with Convolutional Neural Network, there are several stages of methods that can be used, including feature learning and classification. Classification of tomatoes can train CNN with training images. Training is done to find the desired model shape with processing data to get good and accurate results. Augmentation is done in training and data validation so that the data does not experience overfitting on CNN. The final result of this research, getting an accuracy rate of 96.6% from 30 tomato images using Confusion Matrix (Ahmad et al., 2023).

Tomato cultivation is carried out with a sorting system. This system is one of the important things that must be applied to get quality tomatoes, and this sorting system is still done manually. Therefore, CNN is needed in the classification of tomato fruit maturity levels. The classification is based on the color of the tomato and there are three colors used, green for unripe tomatoes, green for semi-ripe tomatoes, and red for ripe tomatoes. This study used 1.148 tomato images, and the final results showed that after testing 10 tomato images, almost 90% of the images were unripe, almost 90% were ripe, and almost 80% were semi-ripe (Ayunda et al., 2023).

Convolutional Neural Networks (CNN) offer a promising approach for the programmatic classification of fresh and rotten fruits and vegetables. This image processing methodology is implemented using the TensorFlow library. In our study, we utilized a dataset consisting of 12.220 images for the classification task. The results indicated that the training data achieved an impressive accuracy of 90,42%, while the validation accuracy reached 94,21%. This was accomplished using the Stochastic Gradient Descent (SGD) optimizer, with a configuration of 20 epochs, a batch size of 16, and a learning rate of 0,01. However, when applied to the testing data, we recorded a slightly lower accuracy of 80,83% (Munfaati & Witanti, 2024).

Research conducted by (Yanto et al., 2021), revealed that one of the limitations in determining fruit viability through visual observation lies in its reliance on human perception, which can be influenced by psychological factors and requires considerable time, particularly in large-scale plantations. A proposed solution to address this challenge is the use of the CNN (Convolutional Neural Network) algorithm. In a study focused on classifying the ripeness of sweet oranges based on color brightness levels, a dataset comprising 100 samples was utilized. The research achieved an overall accuracy of 97.52%,

with a testing accuracy graph indicating 92%. These results demonstrate that CNN is highly effective in classifying the ripeness of sweet oranges by analyzing their texture through color brightness levels.

In the research of mango species classification using CNN, the dataset is divided into three parts: 70% training, 20% validation, and 10% testing. In this study, Otsu segmentation consisting of three layers namely 16, 32, and 63 and Adam optimizer were used for the classification process. In this study, the highest accuracy result obtained was 99.56% with 100% precision, 100% recall, and 100% f1-score (Yati et al., 2023).

Another study conducted by (Putro & Hermawan, 2021), found that Cavendish banana is a local fruit that has high economic value and good marketability. However, there are still errors when assessing the ripeness of the fruit, for example when distinguishing good and bad fruit. The ripeness of the fruit can be affected by light intensity and image quality at the time of ripeness detection. In classification, Artificial Neural Network (ANN) is used as one of the classification methods. In this study, images were taken from morning to evening with a total of 80 images. The most accurate results were obtained when the images were of good quality and taken in the morning. It can be concluded that light intensity and image quality affect the classification of Cavendish banana ripeness based on the color characteristics used.

In comparing algorithms for best accuracy, the study conducted by (Kurniadi et al., 2021), on fruit used two algorithms that are comparable in terms of best accuracy: Support Vector Machine and Convolutional Neural Network algorithms. The dataset for this study was obtained from the Internet through the Kaggle website and contains a total of 11,219 data from 17 labels. The test results show that in fruit classification, the support vector machine algorithm achieved an accuracy of 93.09%, while the convolutional neural network algorithm achieved an accuracy of 96.87%. The number of epochs used for training is 25 epochs. Therefore, we can conclude that the convolutional neural network algorithm is better at classifying fruits than the support vector machine algorithm.

In determining accuracy by comparing two methods, namely CNN and SVM, to classify Gedi leaves, papaya leaves, and yam leaves. The structure applied with the CNN method is VGG16. The result of this study shows that the accuracy rate for CNN is the highest, which is 100%, while SVM only reaches 44.07%. Therefore, the use of CNN method to classify Gedi leaves, papaya leaves, and yam leaves is the best choice in this study (Mulyana & Muthmainnah, 2024).

Another study conducted by (Siwilopo & Marcos, 2023), compared the K-Nearest Neighbor and Convolutional Neural Network methods. This research focuses on the classification of citrus fruits using two sets of data, namely training data and test data. The classification process involves converting the original image into grayscale format through MATLAB. The results of the test show that the accuracy of K-Nearest Neighbor is lower than that of Convolutional Neural Network. Therefore, CNN shows higher accuracy performance.

The development of an Android-based application using the CNN method has made it possible to accurately detect the ripeness of papaya fruit. CNN uses the papaya dataset to classify papaya into three ripeness categories namely unripe papaya, semi-ripe

papaya, and ripe papaya. This is different from the researcher's ripeness category which only has two ripeness categories. This application can not only predict the ripeness of papaya fruit but can also predict the harvest time of papaya based on the analysis of papaya fruit ripeness. The final result shows an accuracy rate of 96.97% (Hawibowo & Muhimmmah, 2024).

Similar research using three categories of ripeness was conducted by (Ristiana Betris Tosi et al., 2024), revealing that one of the algorithms that can determine the ripeness of chili peppers based on their color is a convolutional neural network. The CNN algorithm performs several preprocessing steps, including RGB (Red, Green, Blue) color extraction. In this study, the HSV (hue, saturation, value) model is used. Chili peppers are divided into several colors, such as red, blue, and yellow, and are distinguished by red and green. The CNN algorithm used in this study is very accurate in detecting the ripeness of chili peppers based on several colors. CNN classifies three classes: red chili, orange chili and raw chili, with each class having the same accuracy of 1.00.

METHOD

The stages of the research method include system research framework, tomato data collection, CNN architecture creation, and flowchart.

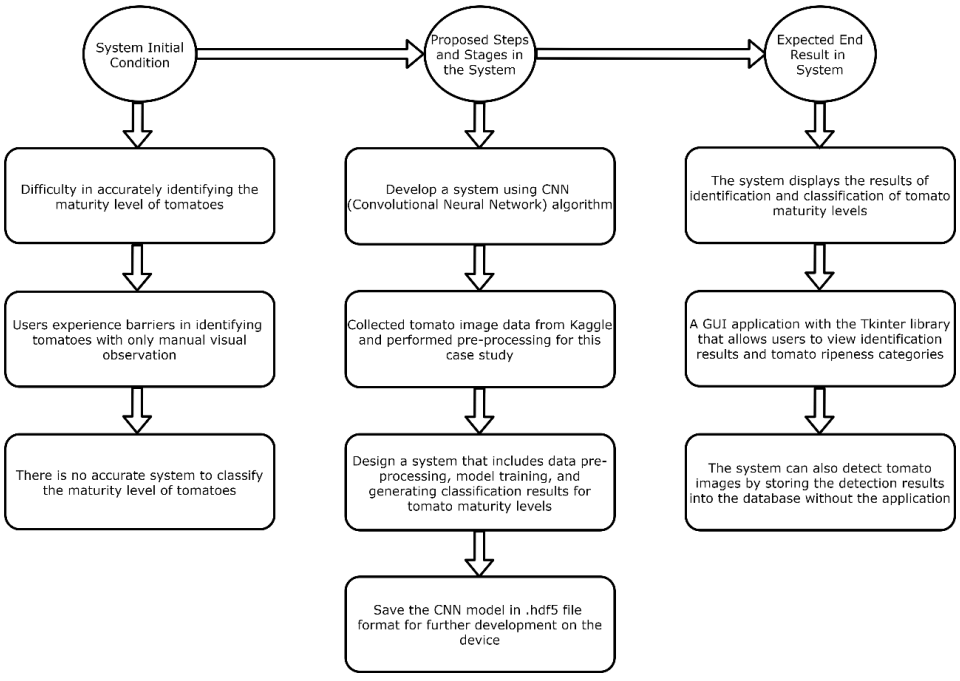


Figure 1. Research Framework System

In the research framework, the main conditions that are the focus of this investigation are outlined. Based on these recognized problems, suggested solutions and stages are introduced to overcome the difficulties encountered. In the anticipated result stage, the system created by using the CNN algorithm, which is stored in a model with the

". hdf5" file format, will present the results of identifying and classifying the tomato ripeness stage. Furthermore, the GUI-based system will show the detection results, sorting the tomatoes as ripe or unripe. The program can also directly detect the ripeness level of tomatoes by storing the detection results in the database without going through the GUI.

Tomato Data Retrieval

Data on tomatoes was obtained partly from the Kaggle platform and mostly through direct observation at the Poktan Welan Asri Petinggen garden tour, Yogyakarta. Direct data collection in the tourist garden was carried out using a Vivo Y17 smartphone equipped with a camera with a resolution of 13 MP. Photos of tomatoes were taken with two different maturity levels, namely green unripe tomatoes and red and yellowish ripe tomatoes, to generate data from various color conditions that are useful in determining the maturity level of tomatoes.

CNN Architecture Creation

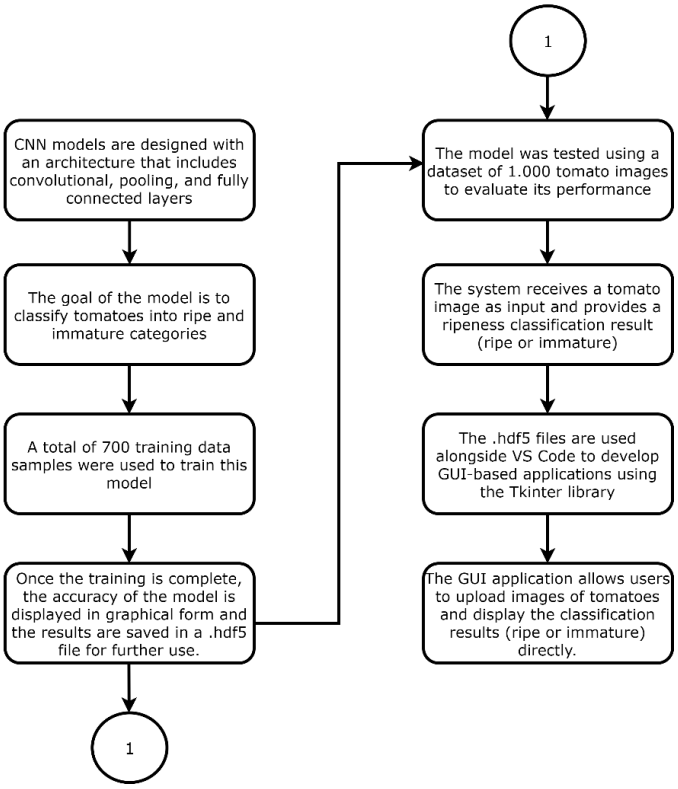


Figure 2. Architecture of CNN Model

The figure shows how the CNN model architecture is finalized. The initial process starts with building a CNN model structure that includes convolution, pooling, and fully connected layers. Next, the model is developed to classify the ripeness level of tomatoes into two types, namely ripe tomatoes and unripe tomatoes. The model training in this study uses 700 training data from a total of 1.000 available data. After the training process is complete, the accuracy of the model is presented through a graph, and the model is saved in the ". hdf5" format file. This model will be tested on each image in the dataset using CNN.

In the final stage, when the system receives an input image of a tomato, the system will provide an output that shows information about the tomato image, whether the tomato is a ripe tomato or an unripe tomato. After system testing is complete, additional testing can be done using the GUI with the Tkinter library in VS Code. The ". hdf5" file that has been saved will be integrated with VS Code to design a GUI using the Tkinter library in the Python programming language. The results obtained show that the GUI runs successfully.

System Flowchart

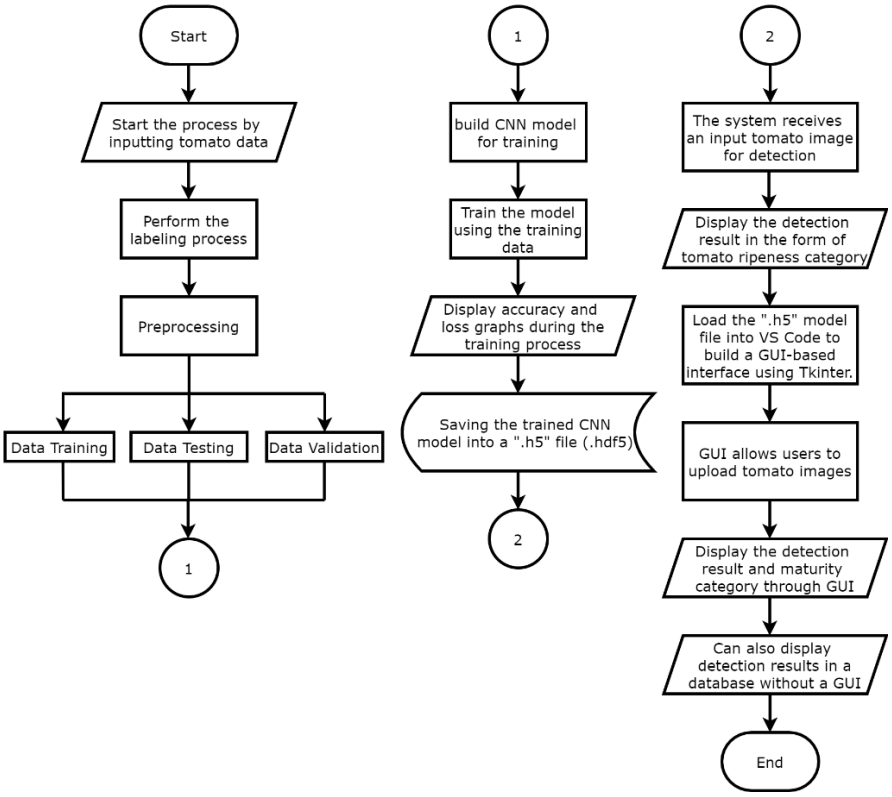


Figure 3. System Flowchart

The flowchart seen above illustrates the steps in the system that works to identify the maturity level of tomatoes. The stages begin with the input of tomato data, followed by an initial processing stage. After that, the data is divided into three parts: dataset for training, dataset for testing, and dataset for validation, where the system will build as well as train the model while showing accuracy-related graphs and losses. The trained CNN model will be saved in a file format with the extension .h5 or. hdf5. The file is then imported into VS Code to create a graphical user interface using the Tkinter library. On Google Colab platform, the system can detect uploaded images. After the image is detected properly, the system will provide maturity results as well as the maturity category of the recognized tomato image. Then, the user interface will display the results of the tomato ripeness analysis based on the input images received. The final process, the program that has been

developed is also able to present the detection results in the database without utilizing the user interface.

FINDING AND DISCUSSION

A system that uses the CNN algorithm to detect tomato maturity requires image data of tomatoes that are still attached to the plant stem or that have not been picked from the tomato plant. This system was developed using Google Colab and VS Code. The image data collected from the research location amounted to 1,000 tomato images which were divided into training, testing, and validation data for system testing. In this finding and discussion, the researcher starts by displaying the contents of the tomato dataset, running the program without a GUI with the Tkinter library, saving the program in a new file with the extension .h5 or (. hdf5), displaying the results of the research conducted through VS Code and displaying the detection results in the database without going through the GUI application.

Research Dataset

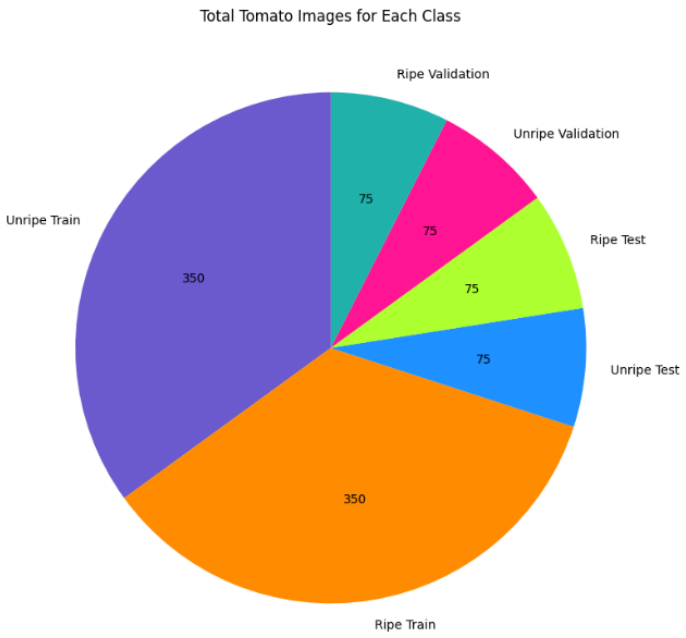


Figure 4. Dataset in Pie Chart

In the data set image above, the pie chart displays the number of images per label. Varying colors can make the pie chart more attractive. In this phase, the researcher presents the data content using pie charts to make the presentation more visually appealing. The pie chart shows a data set consisting of 1.000 tomato pictures labeled according to maturity categories. The data set was labeled as training data, testing data and validation data, with each label representing the corresponding tomato image after the data was split. The pie chart highlights the labels that contain tomato images. There are two tomato ripeness classes including, ripe tomato class, and unripe tomato class.

Data Augmentation

Research conducted by (Fadillah et al., 2021),uses the CNN algorithm and data augmentation approach to increase the number of datasets and achieve better performance than previous research that only uses 26 data. This research focuses on building a Bisindo alphabet translation model using CNN implementation. The accuracy achieved was 94.38%. Without data augmentation, the model only achieved 30% accuracy. Data augmentation is usually used to improve model accuracy with a limited amount of data without expanding the data set.

Table 1. Data Augmentation

NO	Parameter	Value
1	rescale	1./255
2	rotation_range	30
3	width_shift_range	0,2
4	height_shift_range	0,2
5	shear_range	0,3
6	zoom_range	0,1
7	horizontal_flip	True
8	fill_mode	nearest

The data augmentation table summarizes the data parameters used in the train_datagen, val_datagen, and test_datagen programs using ImageDataGenerator. The ImageDataGenerator included in these programs was used for image preprocessing and data extraction. Images are expanded by means of rescaling, rotation, shifting, horizontally, and zooming. This increases the variability in the training data, prevents overfitting, and allows the model to be better generalized.

Summary

The summary data attaches the CNN architecture information used in this research, shown in table 2 below.

Table 2. Summary

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 254, 254, 32)	896
max_pooling2d (MaxPooling2D)	(None, 127, 127, 32)	0
conv2d_1 (Conv2D)	(None, 125, 125, 64)	18,496

max_pooling2d_1 (MaxPooling2D)	(None, 62, 62, 64)	0
conv2d_2 (Conv2D)	(None, 60, 60, 128)	73,856
max_pooling2d_2 (MaxPooling2D)	(None, 30, 30, 128)	0
flatten (Flatten)	(None, 115200)	0
dense (Dense)	(None, 512)	58,982,912
dropout (Dropout)	(None, 512)	0
dense_1 (Dense)	(None, 2)	1,026

Total Params: 59, 077, 186 (225.36 MB)

The table above is a summary generated from the CNN architecture using TensorFlow. It presents the layers used in the model along with information on the output shape and number of parameters that can be trained on each layer. The layers include MaxPooling2D, Conv2D, Flatten, Dense, and Dropout. The output shape describes the size of the result after processing each layer. The None value in the first column indicates the batch size which is not specified during model design and can vary, while the other numbers indicate the image dimensions and the number of units in the dense layer. The Param # column describes the number of parameters to be trained at each layer, which includes the weights and biases that need to be optimized during training. Overall, this table gives an overview of the structure of the CNN model which includes convolution for feature extraction, pooling for dimensionality reduction, and fully connected layers for final classification. The total parameters involved are 59,077,186 (225.36 MB).

Training Epoch

Epoch training is done to train the CNN model to get good accuracy and loss for research. The training epoch is displayed as shown below.

```
21/21 - 25s - 1s/step - accuracy: 0.8750 - loss: 0.3477 - val_accuracy: 0.8929 - val_loss: 0.2417 - learning_rate: 0.0010
Epoch 3/50
21/21 - 210s - 10s/step - accuracy: 0.9401 - loss: 0.1889 - val_accuracy: 0.9628 - val_loss: 0.0924 - learning_rate: 0.0010
Epoch 4/50
21/21 - 6s - 301ms/step - accuracy: 0.9688 - loss: 0.0589 - val_accuracy: 0.9643 - val_loss: 0.0655 - learning_rate: 0.0010
Epoch 5/50
21/21 - 208s - 10s/step - accuracy: 0.9775 - loss: 0.0753 - val_accuracy: 0.9851 - val_loss: 0.0482 - learning_rate: 0.0010
Epoch 6/50
21/21 - 8s - 358ms/step - accuracy: 0.9375 - loss: 0.0773 - val_accuracy: 0.9643 - val_loss: 0.1126 - learning_rate: 0.0010
Epoch 7/50
21/21 - 199s - 9s/step - accuracy: 0.9760 - loss: 0.0622 - val_accuracy: 0.9792 - val_loss: 0.0520 - learning_rate: 0.0010
Epoch 8/50
21/21 - 6s - 300ms/step - accuracy: 0.9688 - loss: 0.0403 - val_accuracy: 1.0000 - val_loss: 0.0099 - learning_rate: 0.0010
Epoch 9/50
21/21 - 203s - 10s/step - accuracy: 0.9760 - loss: 0.0614 - val_accuracy: 0.9762 - val_loss: 0.0748 - learning_rate: 0.0010
Epoch 10/50
21/21 - 29s - 1s/step - accuracy: 0.9688 - loss: 0.2440 - val_accuracy: 0.9643 - val_loss: 0.1032 - learning_rate: 0.0010
Epoch 11/50
21/21 - 241s - 11s/step - accuracy: 0.9775 - loss: 0.0614 - val_accuracy: 0.9836 - val_loss: 0.0407 - learning_rate: 0.0010
Epoch 12/50
21/21 - 8s - 401ms/step - accuracy: 1.0000 - loss: 0.0099 - val_accuracy: 0.9643 - val_loss: 0.0417 - learning_rate: 0.0010
Epoch 13/50
21/21 - 198s - 9s/step - accuracy: 0.9641 - loss: 0.1059 - val_accuracy: 0.9792 - val_loss: 0.0935 - learning_rate: 0.0010
Epoch 14/50
21/21 - 9s - 432ms/step - accuracy: 0.9688 - loss: 0.1243 - val_accuracy: 0.9643 - val_loss: 0.0862 - learning_rate: 5.0000e-04
Epoch 15/50
21/21 - 266s - 13s/step - accuracy: 0.9835 - loss: 0.0526 - val_accuracy: 0.9821 - val_loss: 0.0424 - learning_rate: 5.0000e-04
Epoch 16/50
21/21 - 7s - 324ms/step - accuracy: 1.0000 - loss: 0.0177 - val_accuracy: 1.0000 - val_loss: 0.0360 - learning_rate: 5.0000e-04
Epoch 17/50
21/21 - 245s - 12s/step - accuracy: 0.9805 - loss: 0.0520 - val_accuracy: 0.9777 - val_loss: 0.0534 - learning_rate: 5.0000e-04
Epoch 18/50
21/21 - 29s - 1s/step - accuracy: 1.0000 - loss: 0.0102 - val_accuracy: 1.0000 - val_loss: 0.0344 - learning_rate: 5.0000e-04
5/5 - 13s - 3s/step - accuracy: 0.9267 - loss: 0.2140
Test Loss: 0.2140, Test Accuracy: 0.9267
```

Figure 5. Training Epoch

This CNN model was trained with tomato data using augmentation to increase data variation which showed good training performance. Callbacks such as EarlyStopping and ReduceLROnPlateau help optimize training by stopping training when there is no improvement and adjusting the learning rate. The graph shows a consistent trend in increasing accuracy and decreasing loss, which indicates the model has reached convergence without overfitting. The accuracy on the test data was 92.67% with a low loss (0.214), indicating that the model can classify tomatoes into “unripe” and “ripe” categories accurately.

Train and Validation Accuracy and Loss Graph

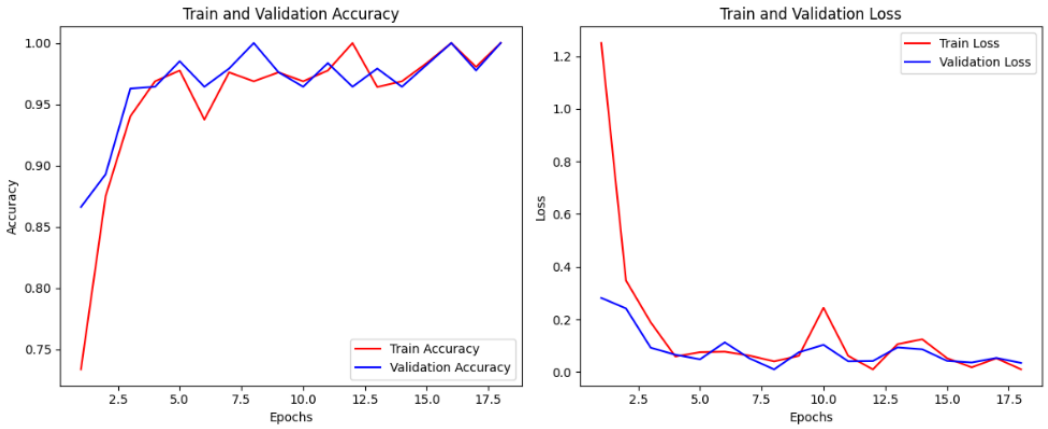


Figure 6. Train and Validation Accuracy and Loss

Figure 6 shows a significantly improved accuracy graph, indicating that the model learns well from the training data and generalizes well to the validation data. The loss graph shows a significant decrease at the beginning of training, with training loss approaching zero and validation loss stabilizing at a low value. This shows that the model does not suffer from

overfitting despite very high training accuracy. Overall, this graph illustrates that the trained CNN model has a good balance between training and validation and consistent generalization performance.

Maximum Accuracy and Minimum Loss

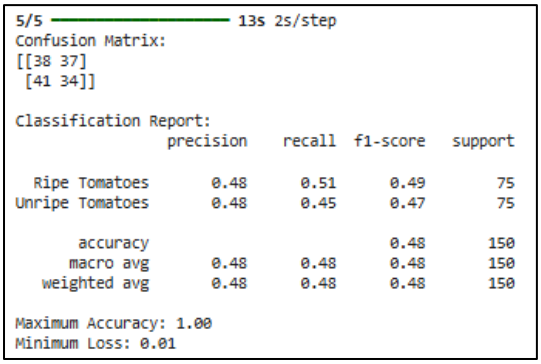


Figure 7. Maximum Accuracy and Minimum Loss

From the model training results, the maximum accuracy of 1.00 indicates that the model achieved perfect accuracy on the training data during the training process. This means that the model was able to learn the patterns from the training data to the maximum extent. However, this does not necessarily reflect the performance on unseen data (validation or testing). A minimum loss of 0.01 indicates that the model's prediction error on the training data is at a very low level. This very small loss value indicates that the model can accurately predict the labels on the training data.

However, from the classification report on the test data, the model still performed poorly with precision, recall, and f1-score around 0,48 for both classes. This indicates that while the model achieved maximum accuracy and minimum loss on the training data, it performed poorly on the test data, possibly due to data imbalance or lack of variation in the training data.

Tomato Ripeness Detection System Results

At this stage, three images are uploaded to be detected directly by the system. It consists of one image of an unripe tomato and two images of different colored tomatoes that are ripe. The detection results will show the level of tomato ripeness according to the tomato image uploaded into the system.

In performing detection, the response time of the system to process each image varies. The CNN system will detect various image conditions, including lighting differences in tomatoes.

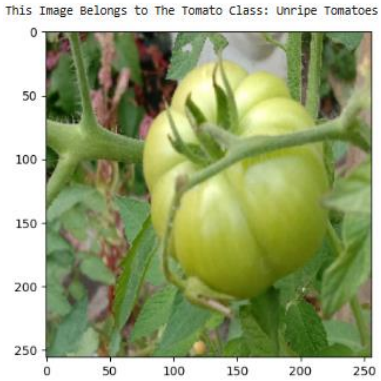


Figure 8. Result Detection of Unripe Tomatoes

The unripe tomato image shows the detection result of the system when the tomato image is uploaded into the system. The detection results show a very accurate detection rate of the tomato image.

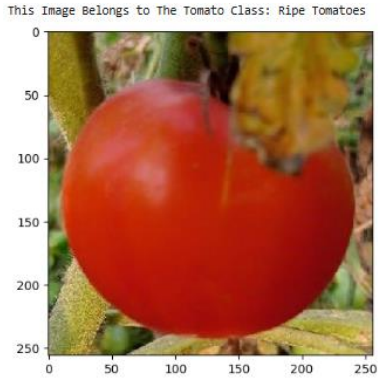


Figure 9. Orange/Reddish Ripe Tomato Detection Result

Figure 9 shows an image of a tomato that is ripe and reddish in color. The image is displayed when the user uploads a tomato image to be detected by the system, then the detection result shows that the image is a ripe tomato.

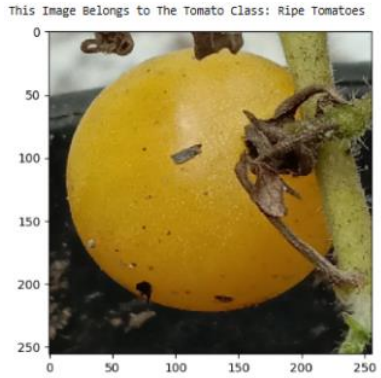


Figure 10. Yellowish Ripe Tomato Detection Result

Figure 10 shows an image of a yellowish ripe tomato. This image is detected as a ripe tomato, because the research that has been done by surveying the location on the farm provides information on ripe tomatoes but with different types of tomatoes. There are many types of tomatoes in the plantation at the research location. However, tomatoes with a yellowish color are ripe tomatoes as well as orange or reddish tomatoes. The system detection results in this study show accurate detection results that tomatoes with yellowish color belong to the class of ripe tomatoes.

Save CNN Model

```
# model disimpan
model.save('/content/tomato.h5')

from google.colab import files
files.download('/content/tomato.h5')
```

Figure 11. Program to Save CNN Model

After completing the entire process in Google Colab, the researcher saved the CNN model that had been developed and trained into a .h5 format file (. hdf5). This file will be utilized in the next research at VS Code to build a user interface (GUI) using the Tkinter library. In this way, researcher do not need to repeat the process of creating and training CNN models when developing GUI-based desktop application interfaces.

GUI Application with Tkinter Library

At this stage, the researcher will attach the interface that has been created by the researcher using VS Code, starting from the GUI dialog display, and the application detection display of tomato maturity.

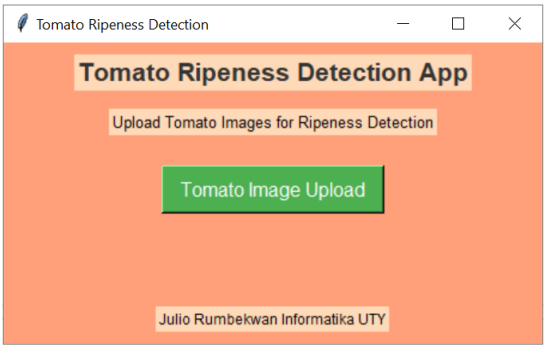


Figure 12. GUI Dialog

The GUI image shows a dialog with a button to upload a tomato image. The dialog is created before a message box appears that will display the detection results of the uploaded tomato image in the GUI dialog.

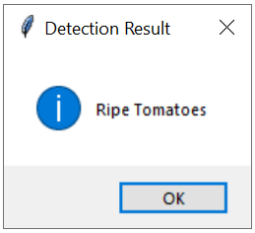


Figure 13. Ripe Tomato Detection Result via Message Box

The message box displays the result of the ripe tomato detection. The uploaded tomato images of orange/reddish, or yellowish tomatoes are detected as ripe according to the detection results displayed by the message box.

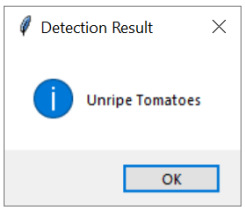


Figure 14. Unripe Tomato Detection Result via Message Box

The message box displays the detection results of unripe tomatoes. The uploaded tomato image is a green tomato image that is detected into an unripe tomato according to the detection result displayed by the message box.

Testing Maturity Detection in Database

At this stage, researcher tested tomato images using Visual Studio Code and displayed the detection results in the database. The database will display a total of 150 images according to the 150 images contained in testing data. Testing data is used for testing in database.

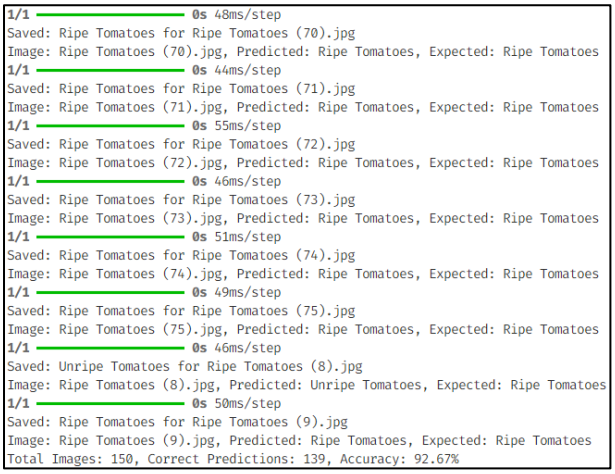
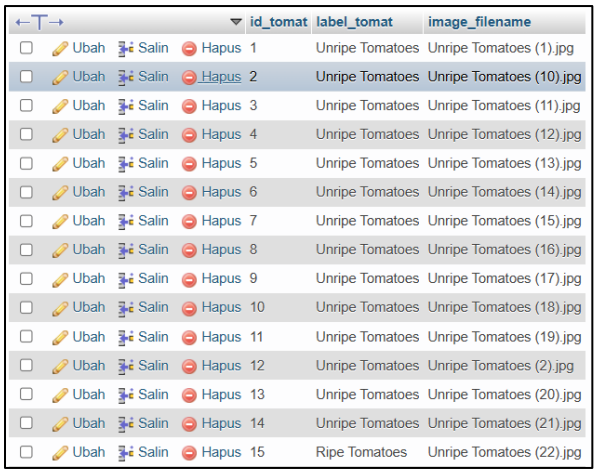


Figure 15. Accuracy Results in Python Terminal

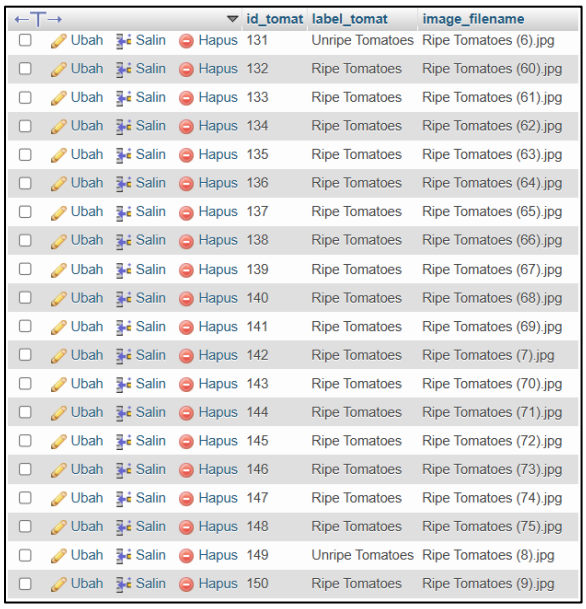
The image shows the running program of tomato ripeness detection. The system detects the images one by one, and the tomato ripeness detection shows an accuracy of 92.67%. The detection results will be displayed in the database in phpMyAdmin.



	id_tomat	label_tomat	image_filename
Ubah Salin Hapus	1	Unripe Tomatoes	Unripe Tomatoes (1).jpg
Ubah Salin Hapus	2	Unripe Tomatoes	Unripe Tomatoes (10).jpg
Ubah Salin Hapus	3	Unripe Tomatoes	Unripe Tomatoes (11).jpg
Ubah Salin Hapus	4	Unripe Tomatoes	Unripe Tomatoes (12).jpg
Ubah Salin Hapus	5	Unripe Tomatoes	Unripe Tomatoes (13).jpg
Ubah Salin Hapus	6	Unripe Tomatoes	Unripe Tomatoes (14).jpg
Ubah Salin Hapus	7	Unripe Tomatoes	Unripe Tomatoes (15).jpg
Ubah Salin Hapus	8	Unripe Tomatoes	Unripe Tomatoes (16).jpg
Ubah Salin Hapus	9	Unripe Tomatoes	Unripe Tomatoes (17).jpg
Ubah Salin Hapus	10	Unripe Tomatoes	Unripe Tomatoes (18).jpg
Ubah Salin Hapus	11	Unripe Tomatoes	Unripe Tomatoes (19).jpg
Ubah Salin Hapus	12	Unripe Tomatoes	Unripe Tomatoes (2).jpg
Ubah Salin Hapus	13	Unripe Tomatoes	Unripe Tomatoes (20).jpg
Ubah Salin Hapus	14	Unripe Tomatoes	Unripe Tomatoes (21).jpg
Ubah Salin Hapus	15	Ripe Tomatoes	Unripe Tomatoes (22).jpg

Figure 16. Initial Row Detection Result in Database

Figure 16 shows the detection results in the database on the first row. Researcher will only display the first line and the last line so that it is more efficient in attaching images in this study. There are 3 columns in the database, namely the id_tomato, label_tomato, and image_filename columns. Id as the row numbering, label to display the image detection result, and filename as the tomato image in the dataset detected by the label.



	id_tomat	label_tomat	image_filename
Ubah Salin Hapus	131	Unripe Tomatoes	Ripe Tomatoes (6).jpg
Ubah Salin Hapus	132	Ripe Tomatoes	Ripe Tomatoes (60).jpg
Ubah Salin Hapus	133	Ripe Tomatoes	Ripe Tomatoes (61).jpg
Ubah Salin Hapus	134	Ripe Tomatoes	Ripe Tomatoes (62).jpg
Ubah Salin Hapus	135	Ripe Tomatoes	Ripe Tomatoes (63).jpg
Ubah Salin Hapus	136	Ripe Tomatoes	Ripe Tomatoes (64).jpg
Ubah Salin Hapus	137	Ripe Tomatoes	Ripe Tomatoes (65).jpg
Ubah Salin Hapus	138	Ripe Tomatoes	Ripe Tomatoes (66).jpg
Ubah Salin Hapus	139	Ripe Tomatoes	Ripe Tomatoes (67).jpg
Ubah Salin Hapus	140	Ripe Tomatoes	Ripe Tomatoes (68).jpg
Ubah Salin Hapus	141	Ripe Tomatoes	Ripe Tomatoes (69).jpg
Ubah Salin Hapus	142	Ripe Tomatoes	Ripe Tomatoes (7).jpg
Ubah Salin Hapus	143	Ripe Tomatoes	Ripe Tomatoes (70).jpg
Ubah Salin Hapus	144	Ripe Tomatoes	Ripe Tomatoes (71).jpg
Ubah Salin Hapus	145	Ripe Tomatoes	Ripe Tomatoes (72).jpg
Ubah Salin Hapus	146	Ripe Tomatoes	Ripe Tomatoes (73).jpg
Ubah Salin Hapus	147	Ripe Tomatoes	Ripe Tomatoes (74).jpg
Ubah Salin Hapus	148	Ripe Tomatoes	Ripe Tomatoes (75).jpg
Ubah Salin Hapus	149	Unripe Tomatoes	Ripe Tomatoes (8).jpg
Ubah Salin Hapus	150	Ripe Tomatoes	Ripe Tomatoes (9).jpg

Figure 17. End of Line Detection Result in Database

Figure 17 shows the detection results in the database on the last row. With a total of 150 images in the test data, 139 images were predicted correctly, and 11 images were predicted incorrectly according to the detection results of all tomato test data images in Figure 15. This shows that the tomato ripeness identification system using CNN is good in this study.

DISCUSSION

Based on the research results, it was found that the CNN algorithm is accurate in detecting the ripeness level of tomatoes. Testing was conducted in three stages, starting from testing using Google Colab, testing with GUI applications using the Tkinter library in Visual Studio Code, and testing in the database on phpMyAdmin.

Similar research results with CNN also obtained good accuracy conducted by (Putri Ananda et al., 2023), using CNN for classification of papaya fruit maturity. The model was built using a data set of 300 papaya fruit photos for the training and testing process. The results obtained in this study showed an accuracy of 99% and a validation value of 97%. Another similar study conducted by (Sutrisna et al., 2024), revealed that determining the ripeness of papaya fruit using the CNN method was done by focusing on the texture and color of the fruit skin. The dataset included samples representing various stages of papaya ripeness and was trained using a CNN model. The study achieved an impressive accuracy of 96.63%, indicating that CNN is an effective solution for detecting the ripeness of papaya fruit.

However, in a study conducted by researcher on tomato ripeness, the classification results in Classification Report showed a lower overall accuracy of 0,48. This can be caused by imbalances in the data, in addition to the dataset used of 1.000 tomato images requiring a long training process in building a CNN model.

CONCLUSION

Based on all stages of research that have been carried out by researcher in identifying tomato maturity using the CNN algorithm, the test accuracy results get an accuracy of 92.67%. With these accuracy results, the CNN algorithm is accurate in detecting tomato ripeness. The tomato image data used for this research amounted to 1.000 tomato images with a division of 70% (700 tomato images) for training data, 15% (150 tomato images) for test data, and 15% (150 tomato images) for validation data. The system created to detect tomato maturity has run accurately, but researcher hope that further development can expand the scope of this limited research. This research only focuses on binary classification (ripe or not ripe) without exploring other maturity levels. For that, it is necessary to add other maturities so that the research is more widespread which does not only focus on ripe tomatoes and unripe tomatoes. The research conducted is expected to help readers or writers who conduct similar research in the field of tomato fruit farming, and farmers in the Poktan Welan Asri Petinggen Garden Tour, Yogyakarta in accurately detecting the ripeness of tomatoes.

REFERENCES

Ahmad, A., Idris, I. S. K., & Bode, A. (2023). Tomato Fruit Type Classification Using Covolutional Neural network. *Jurnal Ilmiah Ilmu Komputer*, *2*(2), 83–89.

Ayunda, N. A., Haryatmi, E., & Riyadi, T. A. (2023). Classification of Tomato Ripeness Based on Convolutional Neural Network Methods. *Journal of Information Systems and Informatics*, *5*(4), 1658–1675. https://doi.org/10.51519/journalisi.v5i4.613

Fadillah, R. Z., Irawan, A., Susanty, M., & Artikel, I. (2021). Data Augmentation to Overcome Data Limitations in the Indonesian Sign Language Interpreter Model (BISINDO). *Jurnal Informatika*, *8*(2), 208–214. https://ejournal.bsi.ac.id/ejurnal/index.php/ji/article/view/10768

Febriyanti, F. A. (2024). Image Processing with Convolutional Neural Network (CNN) Method for Skin Disease Detection in Humans. *Kohesi: Jurnal Sains Dan Teknologi*, *3*(10), 21–30. https://ejournal.warunayama.org/kohesi

Hawibowo, M. S., & Muhimmmah, I. (2024). Application of Papaya Maturity Level Detection using Android-based Convolutional Neural Network (CNN) Method. *Jurnal Edukasi Dan Penelitian Informatika (JEPIN)*, *10*(1), 162. https://doi.org/10.26418/jp.v10i1.77819

Kurniadi, B. W., Prasetyo, H., Ahmad, G. L., Aditya Wibisono, B., & Sandya Prasvita, D. (2021). Comparative Analysis of SVM and CNN Algorithms for Fruit Classification. *Seminar Nasional Mahasiswa Ilmu Komputer Dan Aplikasinya (SENAMIKA) Jakarta-Indonesia*, *September*, 1–11.

Mulyana, S., & Muthmainnah, I. (2024). Comparative Analysis of the Accuracy Level of Cnn and Svm Algorithms in Classification on Gedi Leaves, Papaya Leaves and Ubi Leaves. *JATI (Jurnal Mahasiswa Teknik Informatika)*, *8*(4), 6157–6162. https://doi.org/10.36040/jati.v8i4.10144

Munfaati, E. A. N., & Witanti, A. (2024). Classification of Fresh or Rotten Fruits and Vegetables Using Convolutional Neural Network. *JISKA (Jurnal Informatika Sunan Kalijaga)*, *9*(1), 27–38. https://doi.org/10.14421/jiska.2024.9.1.27-38

Putri Ananda, T., Viola Widyasari, S., Ihsan Muttaqin, M., & Stefanie, A. (2023). Identification of Papaya Fruit Maturity Level Using Convolutional Neural Network (CNN) Method. *JATI (Jurnal Mahasiswa Teknik Informatika)*, *7*(3), 2094–2097. https://doi.org/10.36040/jati.v7i3.7137

Putro, A. D., & Hermawan, A. (2021). Effect of Light and Image Quality in Classification of Cavendish Banana Maturity Based on Color Characteristics Using Artificial Neural Network. *MATRIK : Jurnal Manajemen, Teknik Informatika Dan Rekayasa Komputer*, *21*(1), 215–228. https://doi.org/10.30812/matrik.v21i1.1396

Ristiana Betris Tosi, Helena dorothea Mbura, & Yampi R Kaesmetan. (2024). CNN Implementation in Identifying Chili Maturity Based on Color. *Indonesian Journal of Education And Computer Science*, *2*(1), 34–42. https://doi.org/10.60076/indotech.v2i1.385

Siwilopo, K. P., & Marcos, H. (2023). Comparing Classification on Citrus Fruits Using Convolutional Neural Network and K-Nearest Neighbor Methods. *Komputa : Jurnal Ilmiah Komputer Dan Informatika*, *12*(1), 57–64. https://doi.org/10.34010/komputa.v12i1.9068

Sutrisna, N. P., Sahirah, R. A., Laksono, K. S. S., Permadhi, R. A. S., Nurannisa, N., Larasati, S. S., Asmani, W. W., & Yudistira, N. (2024). Papaya Fruit Maturity Level Detection using Convolutional Neural Network Model. *Jurnal Teknologi Informasi Dan Ilmu Komputer*, *11*(3), 569–578. https://doi.org/10.25126/jtiik.938119

Yanto, B., Fimawahib, L., Supriyanto, A., Hayadi, B. H., & Pratama, R. R. (2021). Classification of Sweet Orange Fruit Maturity Texture Based on Color Brightness Level with Deep Learning Convolutional Neural Network Method. *INOVTEK Polbeng - Seri Informatika*, *6*(2), 259. https://doi.org/10.35314/isi.v6i2.2104

Yati, R., Rohana, T., & Pratama, A. R. (2023). Mango Type Classification Using Convolutional Neural Network Algorithm. *Jurnal Media Informatika Budidarma*, *7*(3), 1265. https://doi.org/10.30865/mib.v7i3.6445