

Fruit Freshness Detection Using Android-Based Transfer Learning MobileNetV2

Irfan Fajar Muttaqin¹, Riza Arifudin²

^{1,2}Computer Science Department, Faculty of Mathematics and Natural Sciences,
Universitas Negeri Semarang, Indonesia

Abstract. Fruit is an important part of the source of food nutrition in humans. Fruit freshness is one of the most important factors in selecting fruit that is suitable for consumption. Fruit freshness is also an important factor in determining the price of fruit in the market. So it is very necessary to detect fruit freshness which can be done by machine. Take apples, bananas, and oranges as samples. The machine learning algorithm used in this study uses MobileNetV2 with transfer learning techniques. MobileNetV2 introduces many new ideas aimed at reducing the number of parameters to make it more efficient to run on mobile devices and achieve high classification accuracy. Transfer learning is used so that data does not need training from the start, so it only takes several networks from MobileNetV2 that have previously been trained and then retrained with a different purpose to improve accuracy results. Then the models that have been created are inserted into the application using Android Studio. Software testing is done through black box testing.

Purpose: The purpose of this research is to design a machine-learning model to detect fruit freshness and then apply it to application Android smartphones.

Methods/Study design/approach: The algorithm used in this study uses MobileNetV2 with transfer learning techniques. Models that have been created are inserted into the application using Android Studio.

Result/Findings: The training results using MobileNetV2 transfer learning obtained an accuracy of 99.62% and the loss results obtained were 0.34%. The results of the application after testing using the black box testing method required improvements to the application and the machine learning model so that it can run optimally.

Novelty/Originality/Value: Machine learning models that have been created using transfer learning MobileNetV2 are applied to Android applications so that they can be used by the public.

Keywords: Fruit Freshness, Transfer Learning, MobileNetV2, Android.

Received July 03, 2023 / **Revised** January 05, 2024 / **Accepted** March 28, 2024

This work is licensed under a [Creative Commons Attribution 4.0 International License](https://creativecommons.org/licenses/by/4.0/).



INTRODUCTION

Computer Vision is one of the newest technological developments today that utilizes computers to obtain various information from an image or video. In contrast to image processing, which is primarily used to process an image into another image, in Computer Vision, when a computer obtains information from an image or video, the computer is also capable of processing and analyzing information so that the information can be used for user needs [1]. One example of the application of Computer Vision is the detection of fruit freshness.

Fruits are an important part of the human diet because they are the main source of dietary nutrients [2]. Apples, bananas, and oranges are one of several types of fruit that exist in Indonesia and are very popular with the general public, both young to old people who like to consume these fruits. This shows that apples, bananas, and oranges are widely consumed by the public and have competitiveness as well [3]. So it is very necessary to detect fruit freshness which can be done by machine. Since the first appearance of machine learning algorithms that have been implemented, various techniques have been proposed [4]. Classification of fresh and rotten fruit is among other studies using a variety of different techniques, such as Regression Tree [5], Convolutional Neural Network [6], and MobileNet [7].

¹*Corresponding author.

Email addresses: fajarmirfan@students.unnes.ac.id (Muttaqin)

DOI: 10.15294/rji.v2i1.70845

MobileNet is part of the Convolutional Neural Networks (CNN) algorithm family used for image classification proposed by the research team at Google [8]. Through different versions, MobileNet introduces many new ideas that aim to reduce the number of parameters to make it more efficient for mobile devices and achieve high classification accuracy [8]. To recognize fruit freshness in limited resources, MobileNetV2 with transfer learning is used [9]. MobileNetV2 which has been pre-trained or previously trained, is used to extract features from fruit images, then the convolutional layer and Softmax classifier are used to classify fruit images into 6 classes and then extracted using MobileNetV2 [9]. Compared to other models, this model implemented by Google open source deep with TensorFlow framework can make a good compromise between accuracy and speed [9]. Once developed, this model will be implemented into Android applications.

Android is an open-source operating system that runs on the Linux kernel. Android applications are developed using the Java language [10]. Android is currently the most widely used smart mobile device platform in the world, occupying 85% of the world market [11]. Machine learning and artificial intelligence (AI) are starting to revolutionize mobile app development, especially on Android [12]. Machine learning applications can now identify speech, photos, and gestures, and translate speech with high accuracy. Machine learning is an important innovation in mobile applications that mobile developers must fully understand because it changes the way people visualize and experience mobile applications [12]. The programming language that will be used to create Android applications using Kotlin.

In May 2017, Google announced via official support for Kotlin on the Android platform. Kotlin is a statically typed programming language that runs on the Java Virtual Machine (JVM) and fully interoperates with Java. Immediately after the announcement, the language grew in popularity, appearing in rankings such as the 2018 and 2019 Stack Overflow Annual Developer Surveys [13]. Kotlin doesn't impact maintainability with respect to Java when working on two small applications. At the same time, Kotlin is moving toward more concise code when asked to develop new features for ongoing software projects [14].

METHODS

As research conducted by [15], datasets are important in the field of deep learning. Therefore, before the classification process, it is necessary to carry out data preprocessing to clean and balance the dataset. After that, the classification process is carried out, and the model that is successfully created will be tested using the evaluation method. The following are the steps in the method used.

Data Collection

At this stage, the author performs a dataset search for the research object. The datasets used in this study are Fruits fresh and rotten for classification and Fresh and Stale Images of Fruits and Vegetables [16] [17]. This dataset is taken from Kaggle by downloading the dataset. The Fruits fresh and rotten dataset for classification dataset has been divided into train data and test data, containing fresh and rotten fruit which include apples, bananas, and oranges. The Fresh and Stale Images of Fruits and Vegetables dataset contains images of six fruits and vegetables: apple, banana, bitter melon, capsicum, orange, and tomato. The pictures taken are simply apples, bananas, and oranges.

Data Preprocessing

At this stage, data processing is carried out from the results of the previous stage. The purpose of this stage is to obtain the best model which will later be applied to Android applications. Several stages are arranged sequentially, namely dividing the dataset into data train, data validation, and data test, creating data augmentation, training, and evaluation.

Split Dataset

In order to be able to evaluate the model, it is necessary to divide the dataset [18]. The dataset is divided into train data, validation data, and test data. The training data is used for learning (model-based), i.e. adjusting the parameters to the machine learning model. Data validation is used to provide an unbiased evaluation of the model fitted to the training data set when setting the model hyperparameters [19]. Test data is unseen data or data that has not been seen and is not included in the process of making a classification model [18]. In this study the method used to split the dataset using a random sampling technique. Random sampling or random sampling is a data sampling method that protects the data modeling process from being biased towards the possibility of different data characteristics [20]. However, random separation can cause problems if there is an uneven distribution of data.

Data Augmentation

In this study, the authors used the Data Augmentation technique or image augmentation. Augmentation is a technique that creates a new image from an existing one by changing its standard position. Image augmentation techniques make it possible to artificially increase the size of the training dataset, thereby providing more data to the model for training. This makes it possible to improve the accuracy of the training model by recognizing new variants of the training data [21]. Various augmentation techniques can be applied such as rescale, rotation, width shift, height shift, shear range, zoom range, horizontal flip, and vertical flip. The following is an explanation of the augmentation technique.

Training Model

The training process begins with creating a basic model using the transfer learning method. The model used is MobileNetV2. The MobileNetV2 layer used for feature extraction is the last layer before the flatten operation (bottleneck layer). MobileNetV2 is loaded using 3 input channels with ImageNet weights. Include_top = False is used because the layer used for feature extraction is the last layer before the flatten operation (bottleneck layer) [22]. All layers on MobileNetV2 will be frozen to prevent specific layer weights from updating during training. Then the feature extraction process is carried out by compiling the model, GlobalAveragePooling2D, and dense into sequential. The dense layer uses the Softmax activation function which will classify images into many classes [22]. The Softmax activation function can be seen in Equation 1.

$$\text{softmax}(x) = \frac{\exp(x_i)}{\sum_{j=1}^n x_j} \quad (1)$$

Information

x = input vector

x_i = exponential function for the input vector

x_j = exponential function for the output vector

n = jumlah class

The model is compiled using the adam optimizer with a learning rate of 0.01, 0.001, or 0.0001. The model training process uses an initial epoch value of 100 [22].

$$m_t = \beta_1 * m_{t-1} - 1(1 - \beta_1) * g_t \quad (2)$$

$$v_t = \beta_2 * v_{t-1} - 1(1 - \beta_2) * g_t^2 \quad (3)$$

Information

β_1, β_2 = Hyperparameter

g_t = Gradient

m_t = The exponential moving average of the gradient

v_t = The exponential moving average of the squared gradient

Evaluation

Evaluation is the process during model development to check whether the model is the best fit for a given problem and the appropriate data. The training result model is tested using accuracy and loss evaluation metrics. Accuracy is defined as the level of truth of a model to classify data correctly. Loss is the value of not knowing the model recognizes the data [23].

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \quad (4)$$

$$\text{Loss} = \frac{FP+FN}{TP+TN+FP+FN} \quad (5)$$

Information

TP = True Positive, the right data is classified by the model as a positive or true value.

TN = True Negative, the right data is classified by the model as a negative or wrong value.

FP = False Positive, the data is incorrect but classified as correct data.

FN = False Negative, data that is correct but classified as wrong as wrong data.

System Planning

In this study, application design was carried out to create a tflite (TensorflowLite) training model with the MobilnetV2 algorithm to detect fruit freshness using Google Collaboratory in Python. Models that have been trained are implemented in Android Studio using the Kotlin language. To make it easier to make an application system on Android, a scheme is needed on the Android application, so that it can run as desired. For the application scheme on the Android, system can be seen in Figure 1.

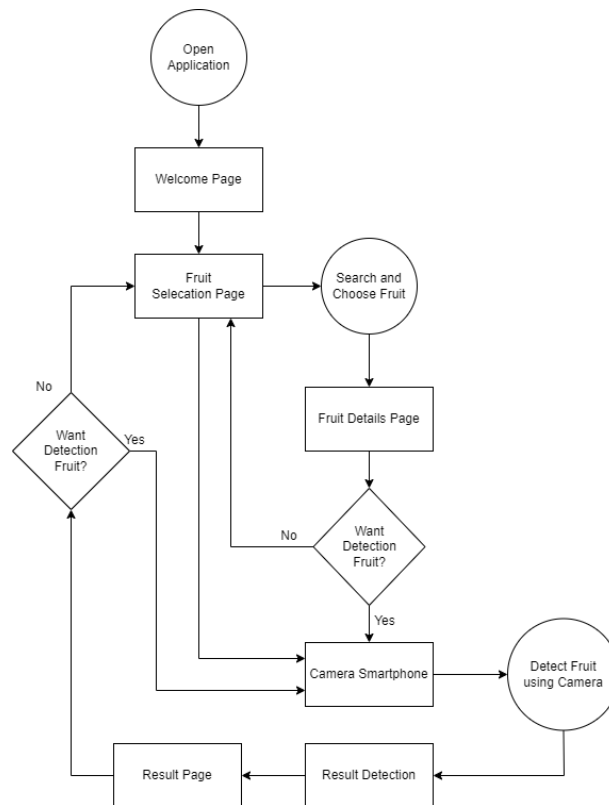


Figure 1. Scheme Android

In the Android application scheme in Figure 1, the first user will open the application and go directly to the welcome page, then go to the fruit selection page which contains a list of bananas, apples, and oranges. If you click on a fruit, it will go to the fruit detail page which contains a fruit description page. To detect fruit using a smartphone camera, you can directly from the fruit selection page. After performing the detection, it will go directly to the detection results page to display the detection results using the smartphone camera.

Testing

After all the coding of the program has been completed, the next step is to carry out the testing phase. The testing phase is carried out to ensure that the functions contained in the application are running according to design [24]. Testing was carried out using the blackbox testing method.

RESULT AND DISCUSSION

Data Collection

The dataset needed in this study is a dataset that contains images of fresh apples, rotten apples, fresh bananas, rotten bananas, fresh oranges, and rotten oranges. Image data is obtained from a previously prepared dataset. The dataset used in this study comes from two different datasets, namely Fruits fresh and rotten for classification and Fresh and Stale Images of Fruits and Vegetables. The Fruits fresh and rotten dataset for classification totals 13,600 files and this dataset has been divided into train data and test data. In the training data and test data, the fruit image data has been divided into 6 classes, namely freshapples, freshbanana, freshoranges, rottenapple, rottenbanana, and rottenoranges. Whereas the Fresh and Stale Images of Fruits and Vegetable dataset contains various classes of types of apples, bananas, pumpkins, peppers, oranges, and tomatoes. According to the limitations of the problem mentioned in the previous

chapter, the fruit images used are only apples, bananas, and oranges. The fresh and rotten fruit dataset for image classification has been divided into training data and test data. After all the datasets have been collected, they will be stored on Google Drive for further processing using the Google Collaboratory application using the Python. Figure 2 shows an example of some pictures representing each fruit.

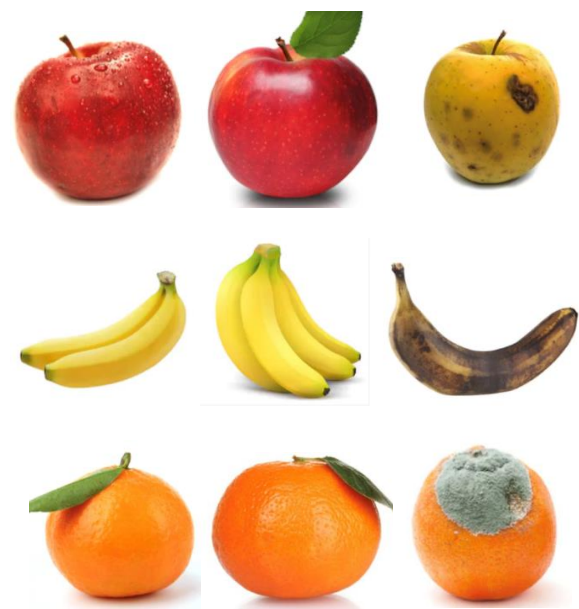


Figure 2. Dataset Sample

Data Preprocessing
Split Dataset

The first stage of dividing the dataset or split dataset is defining the path of each training data, data validation, and test data. Path creation is based on the class to be created, namely fresh orange, rotten orange, fresh banana, rotten banana, fresh apple, and rotten apple. The division of the dataset is done with a ratio of 90:10 for training data and validation data, and 90:10 for test data and validation data. To split the dataset into train data, validation data, and test data using the Random Sample technique. The dataset is divided based on several classes, namely fresh orange, rotten orange, fresh banana, rotten banana, fresh apple and rotten apple. The following results of the split dataset can be seen in Table 1.

Table 1. Result Split Dataset			
Name Class	Amount (File)		
	<i>Train</i>	<i>Validation</i>	<i>Test</i>
Fresh Orange	1320	496	38
Rotten Orange	1436	522	40
Fresh Banana	1423	501	38
Rotten Banana	2002	699	53
Fresh Apple	1524	525	39
Rotten Apple	2108	775	60

Data Augmentation

The results of the split dataset into train data, validation data, and test data will be further processed using augmentation techniques using the Image Data Generator. An Image Data Generator will be created so that the model receives new image variations at each epoch during the training process such as zooming, rotating, and inverting the image. The author applies several transformation techniques to each data train, data validation, and test data such as rescale, rotation, width shift, height shift, shear range, zoom range, horizontal flip, vertical flip, and fill mode. The division of augmentation techniques for training data, validation data, and test data is shown in Table 2.

Table 2. Technique Transformation

Technique	Data Train	Data Validation	Data Test
Rescale	1.0/255.0	1.0/255.0	1.0/255.0
Rotation	40	40	-
Width Shift	0.2	0.2	-
Height Shift	0.2	0.2	-
Shear Range	0.2	0.2	-
Zoom Range	0.2	0.2	-
Horizontal Flip	True	True	-
Vertical Flip	True	True	-
Fill Mode	Nearest	Nearest	-

Training Model

The data is trained using the Adam optimizer, the loss function uses categorical cross-entropy and accuracy metrics. The author uses the Adam optimizer which has many advantages such as more efficient computational processes, uses less memory, and is easy to implement. The loss function exists to calculate the loss in the model so that the weights of the neurons can be updated so that in the next evaluation it is hoped that the loss can be reduced. The categorical cross-entropy loss function is used for binary and multiclass assignments. The accuracy metric is used to get model accuracy. The data is trained on 100 epochs. The training process stops at epoch 91 when the metrics are monitored to prevent overfitting. The training and validation process lasted 14,105 seconds overnight. The best epoch is the 80th epoch with a validation accuracy value of 99%, a loss value of 0,04, and an accuracy of 98%. The graphical training and validation processes are shown in Figure 3.

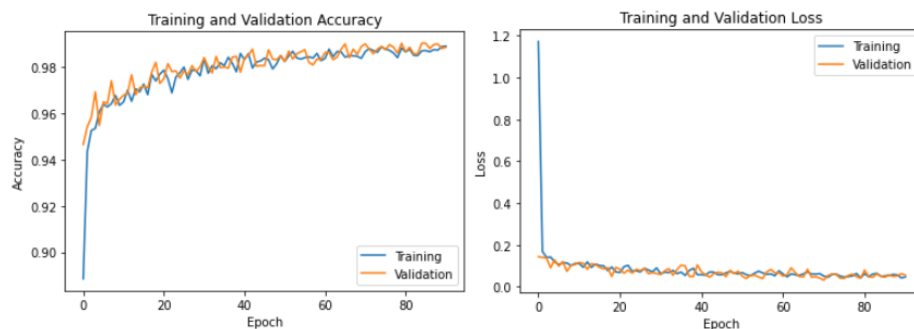


Figure 3. Graph of Training and Validation Parameters

Result

The following are the results of the model training process using MobileNetV2 transfer learning.

1. Accuracy : 98,91%
2. Loss : 0,47%
3. Validation Accuracy : 98,86%
4. Validation Loss : 0,55%
5. Accuracy Test : 99,62%
6. Loss Test : 0,34 %

System Planning

In implementing it into the Android system using the Kotlin programming language. After creating a project in Android Studio, new settings are needed in the AndroidManifest.xml section by adding a new function, namely uses-permission. The uses-permission function functions to add new permissions to the created application. New permission added permission to use smartphone cameras, read data storage, and write data storage. The new permission was added to detect fruit, upload images or fruit images to the application, and download images that have been detected on the results page. The Gradle section requires a new dependency so that Android Studio can read the tflite model. Next, add a new function to the build gradle (module) file, namely the view binding function. View binding functionality is a feature that makes it easy to write code that interacts with views. to display a fruit list using the recycler view, while transferring data between other activities using parcelable.

To display the results using a model that has been previously trained using MobileNetV2 transfer learning. The model used in this study is the tflite model because Android Studio only recognizes the tflite format rather than h5. In the coding for the bitmap or image size to be able to do a detection of 100 x 100 pixels, therefore the bitmap that has been obtained must be changed according to the coding. After that, create a label containing the classification result class which is made sequentially based on the model made from the first, namely Rotten Apple, Fresh Apple, Rotten Banana, Fresh Banana, Rotten Orange, and Fresh Orange. The application display can be seen in Figure 4.

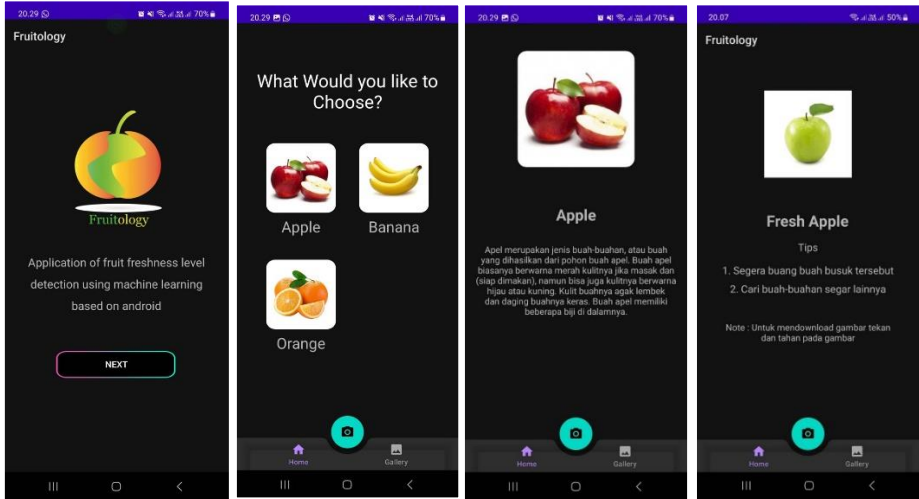
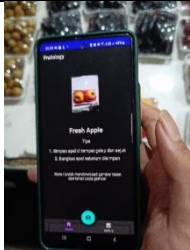


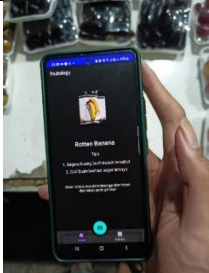
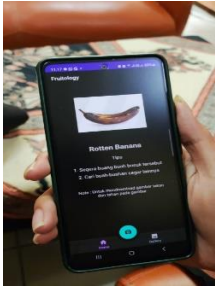
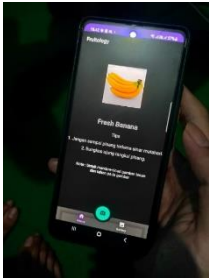
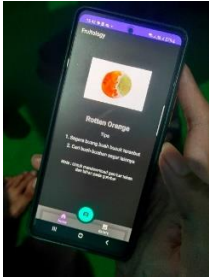
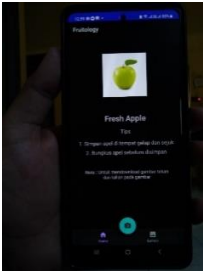
Figure 4. Display on Android

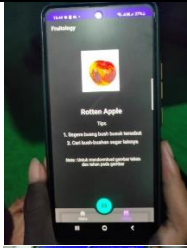
Testing

Testing or testing of fruit freshness detection applications using the blackbox method. Blackbox testing focuses on testing by looking at the functions in the application. In this application, testing is carried out on the functions that are owned. Then compare with the expected results. If the results of the tests carried out are the expected results, the application that has been made is by what has been previously determined. If the results of the tests carried out are not following the expected results, it is necessary to check and improve the application. The following are the results of tests carried out on the fruit freshness detection application using the blackbox method, which can be seen in Table 3.

Table 3. The results of test

No.	Sample Test	Detection Result	Test Result
1.		Fresh Apple	According to expectations
2.		Fresh Banana	According to expectations

No.	Sample Test	Detection Result	Test Result
3.		Rotten Banana	Not as expected
4.		Rotten Banana	According to expectations
5.		Fresh Banana	According to expectations
6.		Rotten Orange	According to expectations
7.		Fresh Apple	According to expectations
8.		Fresh Banana	According to expectations

No.	Sample Test	Detection Result	Test Result
9.		Rotten Apple	According to expectations
10.		Rotten Apple	Not as expected

Discussion

The application of MobileNetV2 transfer learning begins with first creating a model from the first, namely collecting datasets from Kaggle and then processing the data in the form of dividing the dataset by separating train data, validation data, and test data. Furthermore, data augmentation is carried out to increase the dataset using various transformation techniques. Finally, training was carried out using transfer learning by taking certain networks from MobileNetV2 and then training again with a different purpose. After training, the model is created in tflite format. Models that have been created using MobileNetV2 transfer learning are implemented on Android by incorporating the tflite model into the Android Studio application project. The project in an Android Studio is rearranged, especially in the Gradle section so that it can apply models to Android Studio. The model entered must be the tflite model, because Android Studio only recognizes the tflite format instead of h5. Coding on the uploaded tflite model is copied to one of the classes in the project. After that, create a label that contains the classification result class which is made sequentially based on the model made from the first, namely Rotten Apple, Fresh Apple, Rotten Banana, Fresh Banana, Rotten Orange, and Fresh Orange. MobileNetV2 transfer learning can be applied in detecting fruit freshness on the Android system so that the classification of fruit freshness images can be done properly.

Conclusion

The accuracy results obtained from the MobileNetV2 transfer learning algorithm for detecting fruit freshness is 99.62% and the loss results obtained are 0.34% of the total dataset of 14,000 fruit images, namely apples, bananas, and oranges. Based on the test results, it is expected to add a new dataset containing real-time fruit images in Indonesia, especially in the fruit market area so that the results obtained are in accordance with the fruits currently available in Indonesia. This Android application can be further developed by adding other features such as fruit duration, detailed fruit description, or other features that can help to choose fresh fruit.

REFERENCES

- [1] A. O. P. Dewi, "Kecerdasan Buatan sebagai Konsep Baru pada Perpustakaan," *Anuva J. Kaji. Budaya, Perpustakaan, dan Inf.*, vol. 4, no. 4, pp. 453–460, 2020, doi: 10.14710/anuva.4.4.453-460.
- [2] A. Ren *et al.*, "Machine Learning Driven Approach towards the Quality Assessment of Fresh Fruits Using Non-Invasive Sensing," *IEEE Sens. J.*, vol. 20, no. 4, pp. 2075–2083, 2020, doi: 10.1109/JSEN.2019.2949528.
- [3] F. N. Cahya, R. Pebrianto, and T. A. M., "Klasifikasi Buah Segar dan Busuk Menggunakan Ekstraksi Fitur Hu-Moment, Haralick dan Histogram," *IJCIT (Indonesian J. Comput. Inf. Technol.)*, vol. 6, no. 1, pp. 57–62, 2021, doi: 10.31294/ijcit.v6i1.10052.
- [4] N. Özkurt, T. Yıldırım, Yaşar Üniversitesi, Institute of Electrical and Electronics Engineers. Turkey Section., and Institute of Electrical and Electronics Engineers, *2019 Innovations in Intelligent Systems and Applications Conference (ASYU) : proceedings : 31 October-2 November 2019, Izmir, Turkey.*

- [5] K. Skedgell and C. A. Kearney, "Predictors of school absenteeism severity at multiple levels: A classification and regression tree analysis," *Child. Youth Serv. Rev.*, vol. 86, no. 2017, pp. 236–245, 2018, doi: 10.1016/j.childyouth.2018.01.043.
- [6] N. Bindal, R. Modi, and B. K. Kaushik, "Fiscal classification using convolutional neural network," in *Emerging Topics in Artificial Intelligence 2020*, Aug. 2020, no. July, p. 55. doi: 10.1117/12.2568314.
- [7] W. Wang, Y. Li, T. Zou, X. Wang, J. You, and Y. Luo, "A novel image classification approach via dense-mobilenet models," *Mob. Inf. Syst.*, vol. 2020, 2020, doi: 10.1155/2020/7602384.
- [8] A. Alsenan, B. Ben Youssef, and H. Alhichri, "MobileUNetV3—A Combined UNet and MobileNetV3 Architecture for Spinal Cord Gray Matter Segmentation," *Electron.*, vol. 11, no. 15, 2022, doi: 10.3390/electronics11152388.
- [9] Q. Xiang, G. Zhang, X. Wang, J. Lai, R. Li, and Q. Hu, "Fruit image classification based on Mobilenetv2 with transfer learning technique," *ACM Int. Conf. Proceeding Ser.*, 2019, doi: 10.1145/3331453.3361658.
- [10] S. Lenka, S. Kumar, S. Mishra, K. K. Jena, M. L. Liya, and ..., "4th International Conference on I-SMAC (IOT IN SOCIAL, MOBILE, ANALYTICS AND CLOUD)," *IEEE Trans. Softw. Eng.*, pp. 73–79, 2019, [Online]. Available: <https://ieeexplore.ieee.org/abstract/document/9243441/>
- [11] J. Li, L. Sun, Q. Yan, Z. Li, W. Srisa-An, and H. Ye, "Significant Permission Identification for Machine-Learning-Based Android Malware Detection," *IEEE Trans. Ind. Informatics*, vol. 14, no. 7, pp. 3216–3225, 2018, doi: 10.1109/TII.2017.2789219.
- [12] V. Ganesan, "Machine Learning in Mobile Applications," *Int. J. Comput. Sci. Mob. Comput.*, vol. 11, no. 2, pp. 110–118, 2022, doi: 10.47760/ijcsmc.2022.v11i02.013.
- [13] V. Oliveira, L. Teixeira, and F. Ebert, "On the Adoption of Kotlin on Android Development: A Triangulation Study," *SANER 2020 - Proc. 2020 IEEE 27th Int. Conf. Softw. Anal. Evol. Reengineering*, pp. 206–216, 2020, doi: 10.1109/SANER48275.2020.9054859.
- [14] L. Ardito, R. Coppola, G. Malnati, and M. Torchiano, "Effectiveness of Kotlin vs. Java in android app development tasks," *Inf. Softw. Technol.*, vol. 127, p. 106374, 2020, doi: 10.1016/j.infsof.2020.106374.
- [15] S. Krishnan, M. J. Franklin, K. Goldberg, J. Wang, and E. Wu, "ActiveClean: An interactive data cleaning framework for modern machine learning," *Proc. ACM SIGMOD Int. Conf. Manag. Data*, vol. 26-June-20, pp. 2117–2120, 2016, doi: 10.1145/2882903.2899409.
- [16] A. Shrivastava, R. Sohandani, and N. Khatwani, "Fresh and Stale Images of Fruits and Vegetables | Kaggle," *Kaggle*, 2021. <https://www.kaggle.com/datasets/raghavrpotdar/fresh-and-stale-images-of-fruits-and-vegetables> (accessed Feb. 20, 2023).
- [17] Sriram Reddy Kalluri, "Fruits fresh and rotten for classification," 2018. <https://www.kaggle.com/sriramr/fruits-fresh-and-rotten-for-classification>
- [18] V. R. Joseph and A. Vakayil, "SPlit: An Optimal Method for Data Splitting," *Technometrics*, vol. 64, no. 2, pp. 166–176, 2022, doi: 10.1080/00401706.2021.1921037.
- [19] E. Breck, N. Polyzotis, S. Roy, S. E. Whang, and M. Zinkevich, "Data Validation for Machine Learning," *Proc. Mach. Learn. Syst. 1 (MLSys 2019)*, pp. 334–347, 2019.
- [20] S. Basso, A. Ceselli, and A. Tettamanzi, "Random sampling and machine learning to understand good decompositions," *Ann. Oper. Res.*, vol. 284, no. 2, pp. 501–526, 2020, doi: 10.1007/s10479-018-3067-9.
- [21] A. Mikołajczyk and M. Grochowski, "Data augmentation for improving deep learning in image classification problem," *2018 Int. Interdiscip. PhD Work. IIPhDW 2018*, no. August 2019, pp. 117–122, 2018, doi: 10.1109/IIPHDW.2018.8388338.
- [22] N. Hikmatia and M. Zul, "Aplikasi Penerjemah Bahasa Isyarat Indonesia menjadi Suara berbasis Android menggunakan Tensorflow," *J. Komput. Terap.*, vol. 7, no. Vol. 7 No. 1 (2021), pp. 74–83, 2021, doi: 10.35143/jkt.v7i1.4629.
- [23] S. Ruuska, W. Härmäläinen, S. Kajava, M. Mughal, P. Matilainen, and J. Mononen, "Evaluation of the confusion matrix method in the validation of an automated system for measuring feeding behaviour of cattle," *Behav. Processes*, vol. 148, pp. 56–62, 2018, doi: 10.1016/j.beproc.2018.01.004.
- [24] S. Supriyono, "Software Testing with the approach of Blackbox Testing on the Academic Information System," *IJISTECH (International J. Inf. Syst. Technol.)*, vol. 3, no. 2, pp. 227–233, 2020,