JOIN (Jurnal Online Informatika)
p-ISSN: 2528-1682, e-ISSN: 2527-9165
Volume 9 Number 2 | June 2024: 148-158
DOI: 10.15575/join.v9i1.1300

Cassava Diseases Classification using EfficientNet Model with Imbalance Data Handling

Stephany Octaviani Ngesthi[1], Lili Ayu Wulandhari[2]
[1,2]Department of Computer Science, Binus University Jakarta, Indonesia

Article Info

Article history:

Received February 9, 2024
Revised March 26, 2024
Accepted April 2, 2024
Published August 24, 2024

Keywords:

Basic augmentation
Cassava diseases classification
EfficientNet
Imbalance data handling
Neural style transfer
SMOTE

ABSTRACT

This research highlights the urgent need for classifying cassava diseases into five classes, such as Cassava Bacterial Blight (CBB), Cassava Brown Streak Disease (CBSD), Cassava Green Mottle (CGM), and Cassava Mosaic Disease (CMD), and Healthy. The study proposes the utilization of the EfficientNet model, a lightweight deep learning architecture, for classifying cassava diseases based on leaf images. However, the datasets available for this classification task are all unbalanced, made it difficult for researchers to perform. To tackle this imbalance issue, the authors compared several imbalance data handling methods commonly used for image classification, including SMOTE (Synthetic Minority Oversampling Technique), basic augmentation, and neural style transfer, to be applied before fed into EfficientNet. Initially, EfficientNet model without addressing dataset imbalances, the F1-Score stands at 78%, with most images misclassified into the majority class. Integration with SMOTE notably boosts the F1-Score to 82%, showcasing the efficacy of oversampling methods in enhancing model performance. Conversely, employing data augmentation, both basic and deep learning-based, lowers the F1-Score to 74% and 65% respectively, yet it results in a more balanced distribution of true positives across disease classes. The findings suggest that SMOTE surpasses the other methods in handling imbalanced data.

Corresponding Author:

Stephany Octaviani Ngesthi,
Department of Computer Science, Binus University Jakarta, Indonesia
Jl. Kebon Jeruk Raya No. 27, Kebon Jeruk Jakarta Barat 11530, Indonesia
Email: stephany.ngesthi@binus.ac.id

1. INTRODUCTION

Cassava is the third-largest source of carbohydrates after rice and corn [1]. However, cultivating cassava plants faces significant challenges from four prominent diseases caused by bacterial and viral infections, which are Cassava Bacterial Blight (CBB), Cassava Brown Streak Disease (CBSD), Cassava Green Mottle (CGM), and Cassava Mosaic Disease (CMD). CBB induces wilting and blight, CBSD leads to brown streaks on roots, CGM results in severe leaf deformation, and CMD causes mottling and distortion, reducing photosynthesis [2]. Each disease requires a distinct treatment method. Farmers rely on government-funded agricultural experts for visual inspections and diagnoses, which is quite time-consuming and costly [2]. Therefore, implementing an automatic deep learning-based system for cassava leaf disease classification is essential to enhance the efficiency, resilience, and sustainability of cassava cultivation. A competition named iCassava Fine-Grained Visual Categorization Challenge [3] addressed classifying cassava leaf images into five classes, including four diseases and healthy. The competition dataset initially contained 2,317 images in 2019 and an expanded dataset of 21,397 images in 2021, where data can be obtained from Kaggle [4].

Previous research has been undertaken on this competition. A study by Ayu et al. [5] utilized MobileNet-V2 on a mobile application, achieving a 65.6% overall accuracy. Another experiment by

Sangbumrung et al. [6] used Faster R-CNN to classify cassava leaf images into healthy and unhealthy. This approach yielded an impressive F-measure of 96%. Maryum et al. [7] improved cassava disease classification by implementing U-Net segmentation before the EfficientNet-B4 model. These improvements aided the model in reducing background noise and increasing accuracy from 81.43% to 89.09%.

Meanwhile, Zhong et al. [8] introduced T-Rnet, a transformer-embedded ResNet with a focal angular margin penalty softmax (FAMP - Softmax) loss function, which enables effective learning of classification boundaries in imbalanced datasets and achieves an impressive accuracy of 91.2%. Liu et al. [9] introduced an EfficientNet with a multi-scale fusion model, which helps the model extract disease features containing essential information from cassava leaves and enhance the classification. This method outperformed the original EfficientNet model, achieving an almost 4% improvement to 88.1% in the average recognition rate. Another study by Nabende et al. [10] achieved a 90.1% accuracy using Haar-Cascade as the pre-processing method and the Deep Residual Neural (DRN) network for their proposed model. Gao et al. [11] employed an EfficientNet model and utilized the HSV color space for image pre-processing, resulting in an overall accuracy of 91.7%. Aravind et al. [12] demonstrated outstanding performance with the EfficientNet model, reaching an impressive accuracy rate of 96.74%. Shahriar et al. [13] conducted a comparative study of state-of-the-art models, resulting in Xception and EfficientNet-B0 with the highest overall accuracy rates at 91.3% and 91.1%, respectively.

However, both datasets in the competition in 2019 and 2021 [4] all exhibited class imbalances, which made it difficult for researchers to perform this classification task. To address the unbalanced data, Sambasiviam et al. [14] and Lilhore et al. [15] have demonstrated improved performances by implementing the Synthetic Minority Oversampling Technique (SMOTE), the common method used for imbalance data handling. Research [14] enhanced their Convolutional Neural Network (CNN) model by incorporating focal loss and SMOTE, then elevating the precision, recall, and F1-score from 83% to 92%. Simultaneously, research [15] introduced an Enhanced Convolutional Neural Network (E-CNN) model with an initial accuracy of 96%. When SMOTE was applied, the average accuracy experienced a substantial increase to 98.7%. As pioneered by Chawla et al. [16], SMOTE generates synthetic instances for the minority class, known as the selected seed, effectively rebalancing the dataset. The SMOTE algorithm initiates by randomly selecting an example from the minority class. Subsequently, the algorithm identifies the *k* nearest neighbors of the chosen seed within the feature space. From the set of *k*, one neighbor is randomly chosen. A synthetic example is created by selecting a random point in the feature space along the line connecting the seed and the chosen neighbor.

Besides SMOTE, basic augmentation and deep learning approaches in image classification are also commonly used to enhance the number of images, leading to imbalanced data handling Shorten & Khoshgoftaar [17]. Basic or traditional augmentation can be done to images by flipping, rotating, zooming, cropping, RGB shuffling, enhancing the contrast or brightness, etc. The deep learning approach can be done using neural style transfer. Neural style transfer involves merging a content image with a style reference image to produce an output image that preserves the content while incorporating the artistic style of the reference image [18]. A paper by Perez & Wang [19] shows some improvement results after applying traditional augmentation, such as flipping, rotating, zooming, cropping, and else, which improved the validation accuracy from 85.5% to 89% for Dogs vs. Goldfish classification, and 70.5% to 77.5% for Dogs vs. Cat.

The study of Zheng et al. [20] also delves into the potential of employing neural style transfer as a novel approach to augmenting data for image classification tasks. This study also shows some classification results on the Caltech-256 dataset, with the comparisons after implementing basic augmentation and neural style transfer. Without augmentation or style transfer, the baseline accuracy is 62%. Upon applying flipping as the conventional augmentation technique, the accuracy escalates to 66.67%. Introducing scream style transfer yields a notable improvement, resulting in an accuracy of 63.13%.

Similarly, employing wave style transfer enhances accuracy to 62.72%. Combining flipping and scream style transfer leads to a further increase, with accuracy reaching 67.28%. Meanwhile, combining flipping with wave-style transfer results in a slightly lower accuracy of 66.32%. Therefore, in this research, the authors aimed to compare SMOTE, basic augmentation, and neural style transfer, which will be used as imbalanced data handling methods before EfficientNet deep learning model in cassava disease classification.

2. METHOD

The method outlined in this research is segmented into five key stages: data gathering, data pre-processing, imbalance data handling, classification, and performance evaluation, as illustrated in Figure 1. For imbalanced data handling, the authors will compare several methods, including SMOTE, basic augmentation, and neural style transfer.

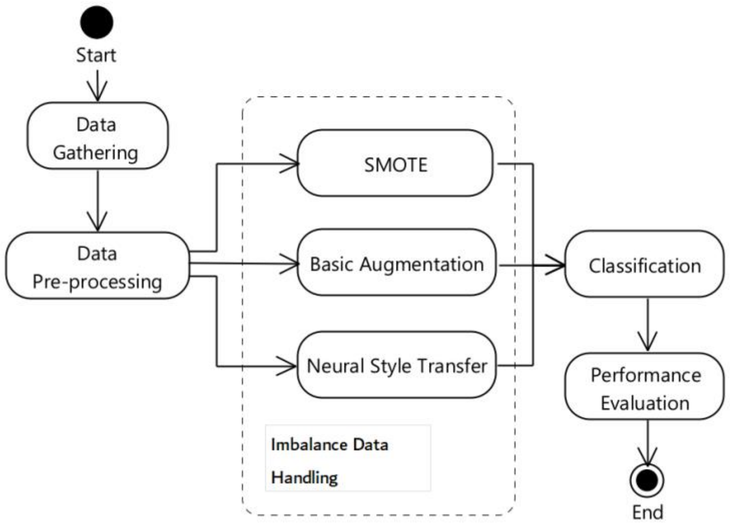
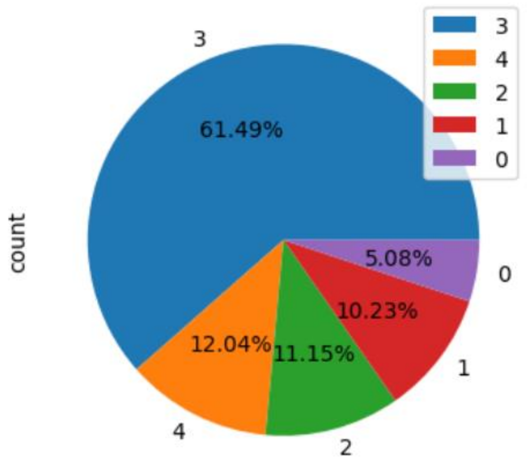
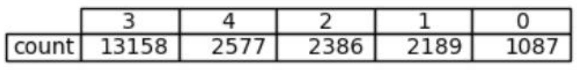


Figure 1. Proposed Method

2.1. Data Gathering

The dataset utilized in this study is sourced from the Cassava Disease Classification competition held in 2021. It comprises a total of 1,087 images representing Cassava Bacterial Blight (CBB), 2,189 images for Cassava Brown Streak Disease (CBSD), 2,386 images showcasing Cassava Green Mottle (CGM), 13,158 images featuring Cassava Mosaic Disease (CMD), and 2,577 images representing the Healthy category. The distribution and samples for each class within the dataset are visually represented in Figure 2 and 3, respectively. Class 0, 1, 2, 3, and 4 represent CBB, CBSD, CGM, CMD, and Healthy consecutively.





	3	4	2	1	0
count	13158	2577	2386	2189	1087

Figure 2. Dataset class distribution

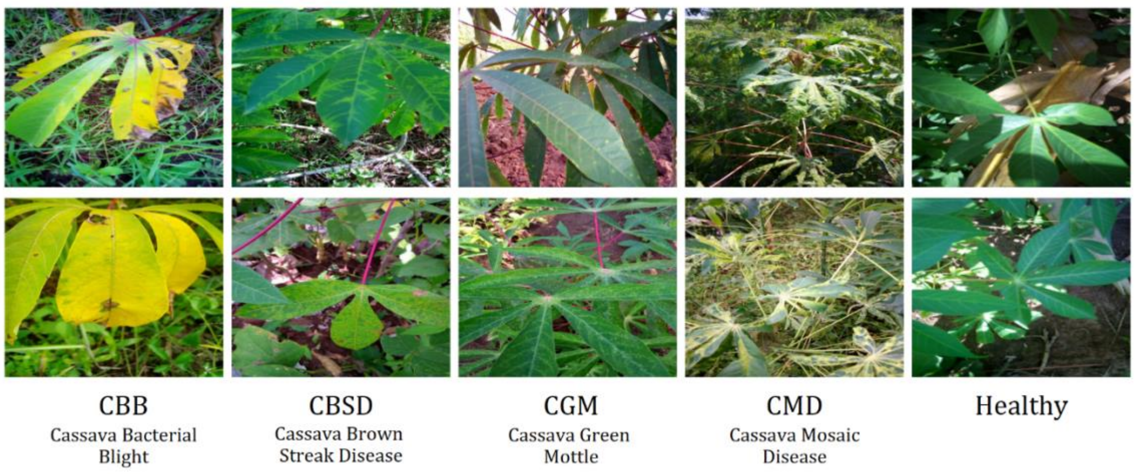


Figure 3. Dataset samples

2.2. Data Pre-processing

During the initial data pre-processing phase, image resizing was implemented, setting all images to a uniform size of 150 × 150 × 3 pixels. After that, a 50% split was performed, allocating half of the data for training and the remaining half for testing. The constraints of limited memory resources drove the decision to opt for a 50% split if the training data exceeds the 50% threshold. Following the split, the specific imbalance data handling techniques were exclusively applied to the training dataset.

2.3. Imbalance Data Handling

2.3.1. SMOTE

The train data distribution is initially depicted in Figure 3.5. Then, SMOTE, using the flow described in Figure 2.6 as pioneered by Chawla et al. [16], was applied only to the training data until the count of train data became as in Figure 3.6. In applying SMOTE, the train data, which consists of 10,698 images, were first resized from 150×150×3 to 67,500 and generated into an array of 67,500×10,698. Then, SMOTE is applied using the Python library by resampling all classes except the majority class. As the Introduction section explains, SMOTE generates random points using *k*-nearest neighbors [16]. Here, the value of *k* is set to 5. The flow of SMOTE is depicted in Figure 4.

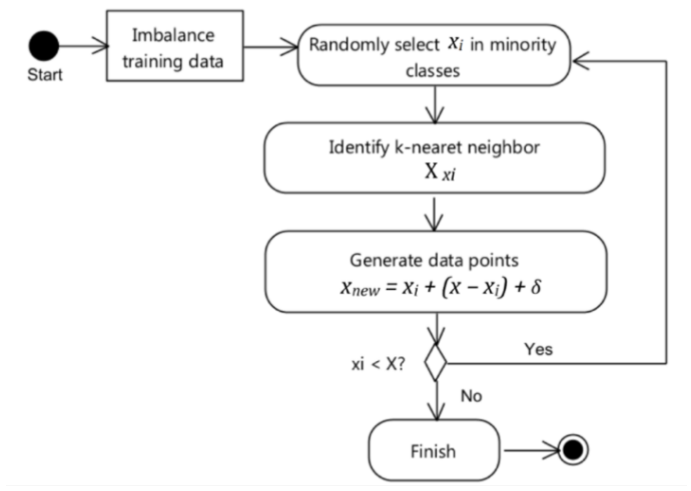
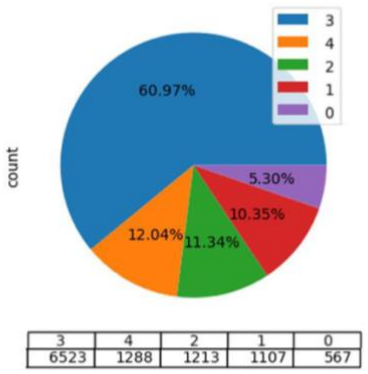

Start
Imbalance training data
Randomly select x_i in minority classes
Identify k-nearet neighbor X_{xi}
Generate data points $x_{new} = x_i + (x - x_i) + \delta$
xi < X?
Yes
No
Finish

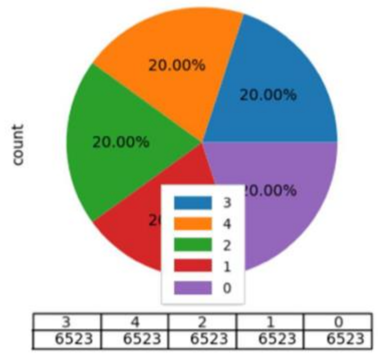
Figure 4. SMOTE flow

After SMOTE, the array became 67,500 × 32,615, where 32,615 contains 6,523 images each from 5 classes. The array is resized again to the input shape of images, which is 150 × 150 × 3 × 32,615, before being fed to the classification model.



3	4	2	1	0
6523	1288	1213	1107	567

Figure 5. Train Data Distribution



3	4	2	1	0
6523	6523	6523	6523	6523

Figure 6. Train Data Distribution after SMOTE

2.3.2. Basic Augmentation

In basic augmentation, several transformations are applied to the minority classes of training data until their quantity of images approaches that of the major class, specifically class 3. The details of the transformations are shown in Table 1.

Table 1. Basic Augmentation Process

Augmentation Transformations	***Images Count***				
	Class 0	***Class 1***	***Class 2***	***Class 3***	***Class 4***
Original Data	567	1,107	1,212	6,523	1,288
Flip Horizontal	567	1,107	-	-	1,288
Flip Vertical	567	-	1,212	-	-
Zoom and Crop	567	1,107	1,212	-	1,288
Gaussian Noise	567	1,107	1,212	-	1,288
Solarize	567	-	-	-	-
Gamma Contrast	567	1,107	1,212	-	1,288
Sigmoid Contrast	567	-	-	-	-
Linear Contrast	567	-	-	-	-
Elastic	567	-	-	-	-
Jigsaw	567	-	-	-	-
Total images after Augmentation	**6,237**	**6,642**	**6,060**	**6,523**	**6,440**

Examples of augmented images are depicted in Figure 7 to provide a visual representation.

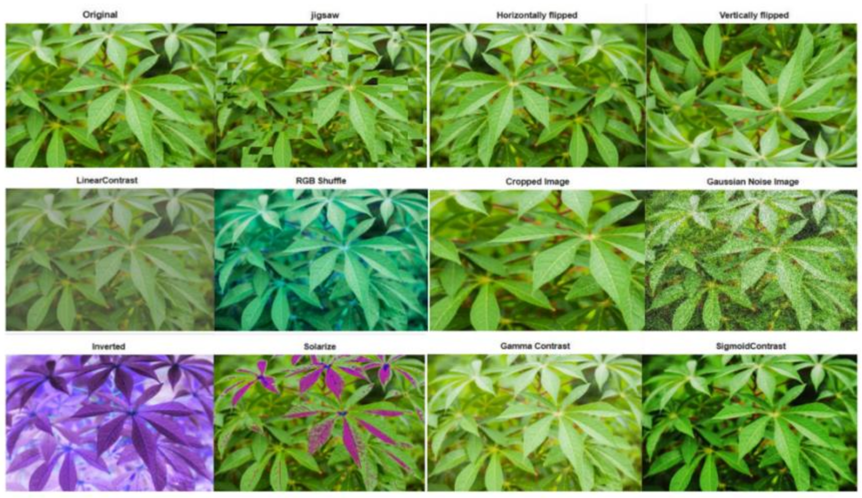


Figure 7. Results of basic augmentation approach

2.3.3. Neural Style Transfer

In this technique, the authors utilized a neural style transfer model outlined by Ghiasi et al [21]. The original images belonging to minority classes undergo stylization using the style images in Figure 8 until the count approached the quantity of images in class 3. The style images are obtained from the open-source datasets [22]. The detail of stylization process is shown in Table 2. The results samples of this process are illustrated in Figure 9.

Table 2. Neural Style Transfer Process

Style Transfer Images	***Images Count***				
	Class 0	***Class 1***	***Class 2***	***Class 3***	***Class 4***
Original Data	567	1,107	1,212	6,523	1,288
Style Image 1	567	1,107	-	-	1,288
Style Image 2	567	-	1,212	-	-
Style Image 3	567	1,107	1,212	-	1,288
Style Image 4	567	1,107	1,212	-	1,288
Style Image 5	567	-	-	-	-
Style Image 6	567	1,107	1,212	-	1,288
Style Image 7	567	-	-	-	-
Style Image 8	567	-	-	-	-
Style Image 9	567	-	-	-	-
Style Image 10	567	-	-	-	-
Total images after Augmentation	**6,237**	**6,642**	**6,060**	**6,523**	**6,440**

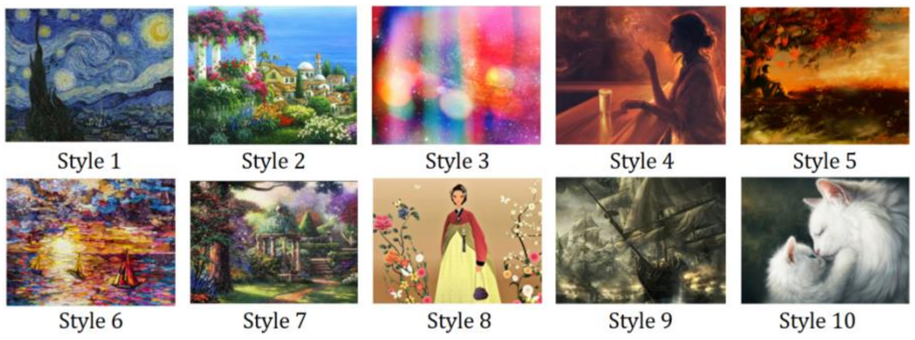


Figure 8. Neural style images [22]

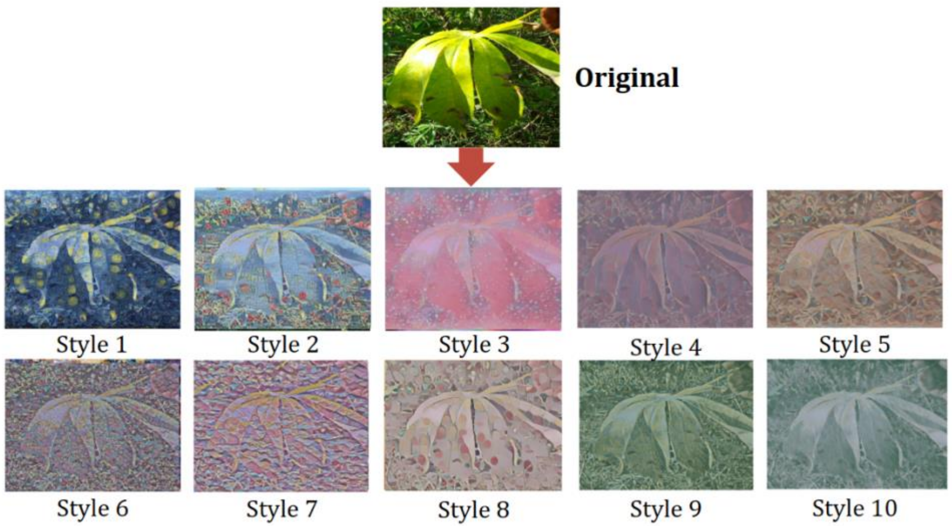


Figure 9. Samples results of neural style transfer

2.4. Classification

In the classification stage, the authors leverage the EfficientNet model [23], seamlessly integrated with a global average pooling 2D layer, a dropout layer, and a dense layer using Rectified Linear Units (ReLU) function, then followed by softmax activation. The softmax activation function is subsequently applied for the classification layer. The equation of ReLU and softmax [24] are shown as follows.

ReLU: $f(x) = x$ .... for $x \geq n$
$f(x) = 0$ .... for $x < n$

Softmax: $f(x) = \frac{e^{xi}}{\sum_{x=i}^{x=j} e^{xj}}$

In EfficientNet model, the authors use Adaptive Moment Estimation (Adam) as the optimizers, where the equation is shown below.

$m_t = \beta_1 * m_t - 1 + (1 - \beta_1) * gt$
$v_t = \beta_2 * v_t - 1 + (1 - \beta_2) * g^2t$
$f(x) = x - (\alpha * m_t / \sqrt{\square}\ (v_t + \varepsilon\ )$

The comprehensive flow of the classification stage is intricately depicted in Figure 10, as follows.

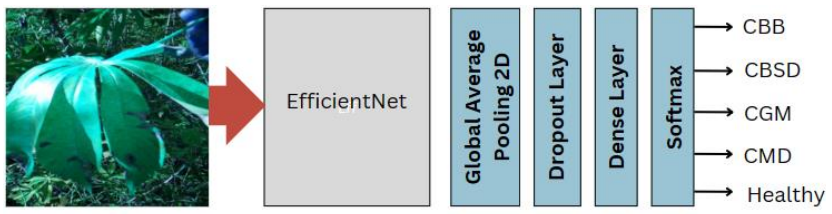


Figure 10. Classification flow

2.5. Performance Evaluation

The evaluation of testing performances will be conducted using the confusion matrix, and metrics such as F1-Score, precision, and recall will be calculated. The formulas employed for performance assessment are presented in Equation 1-4. In these equations, TP corresponds to True Positives, TN denotes True Negatives, FP represents False Positives, and FN stands for False Negatives.

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

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

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

$$\text{F1-Score} = \frac{2 \times \Pr ecision \times Recall}{\Pr ecision + Recall} \quad (4)$$

3. RESULTS AND DISCUSSION

3.1. Results

In this research, the authors conducted a series of experiments aimed at fine-tuning parameters to identify the optimal architecture for the EfficientNet model. The architectural models subjected to testing include EfficientNet-B0 and EfficientNet-B1, both chosen for their possession of fewer than 8 million trainable parameters, 5.4 M and 7.8 M consecutively [20], making them lightweight and suitable for training with 21,397 image data in the limited memory resources. The initial learning rate is set at 0.001, aligning with the default value in Keras. Subsequently, both increments and decrements of the learning rate by 0.0002 until a noticeable decline in accuracy is observed. The parameterization process extends to the number of epochs, spanning from 1 to 6, with the selection of the best accuracy results among them. Experiments and training are executed in Kaggle using Tensor Processing Units (TPUs). Table 3 shows the results of the parameterization.

Table 3. Summary of Parameterization Results

Model	*Learning Rate*	*Number of epochs*	*Testing Accuracy*
EfficientNet-B0	0.001	6	72.13%
	0.0008	5	73.24%
	0.0006	6	73.58%
	0.0004	5	73.54%
	0.0012	4	70.84%
EfficientNet-B1	0.001	6	72.34%
	0.0008	3	74.36%
	0.0006	**5**	**74.76%**
	0.0004	5	73.28%
	0.0012	5	69.98%

The findings in Table 3 demonstrate the impact of adjusting the learning rate for EfficientNet-B0 and decreasing the learning rate from the default 0.001 to 0.0008 improved testing accuracy, rising to 72.34% from 72.13%. Encouraged by this improvement, the authors further reduced the learning rate in increments of 0.0002 until observing a decline in accuracy at 0.0004. Conversely, increasing the learning rate to 0.0012 decreased testing accuracy to 70.84%, prompting the authors to halt the increment.

Like EfficientNet-B0, EfficientNet-B1 exhibited improved performance when the learning rate decreased to 0.0008, with a subsequent decline observed at a learning rate of 0.0004. In contrast, raising the learning rate to 0.0012 resulted in decreased performance for EfficientNet-B1, mirroring the observed trend in EfficientNet-B0. Table I also highlights that the EfficientNet-B1 model, trained with a learning rate of 0.0006 and over five epochs, demonstrated the highest performance, leading to its selection as the baseline model for cassava disease classification. Various training experiments were performed using the baseline model to assess the performance of four methods. These methods included using EfficientNet without addressing data imbalance (Method I), incorporating EfficientNet with SMOTE (Method II), employing EfficientNet with basic augmentation (Method III), and utilizing EfficientNet with neural style transfer (Method IV). The aim was to compare and analyze the performances and effectiveness of these approaches.

The confusion matrices generated by methods I, II, III, and IV are visually represented sequentially in Figure 11-14. Subsequently, accuracy, precision, recall, and F1-Score are computed using the formulas presented in Eq. 1-4. The outcomes of these calculations are tabulated in Table 4 for comprehensive examination and comparison.

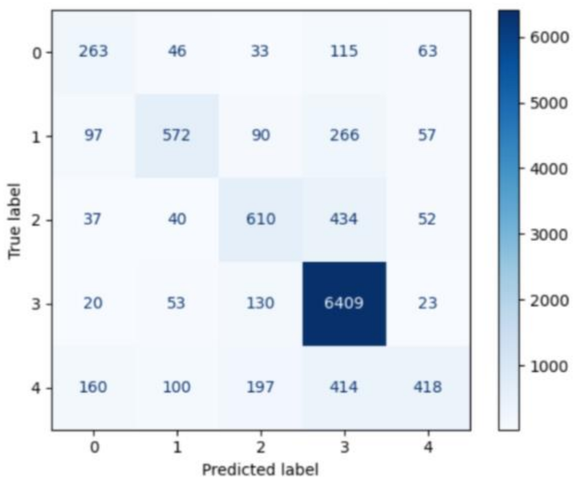


Figure 11. Confusion matrix of method I

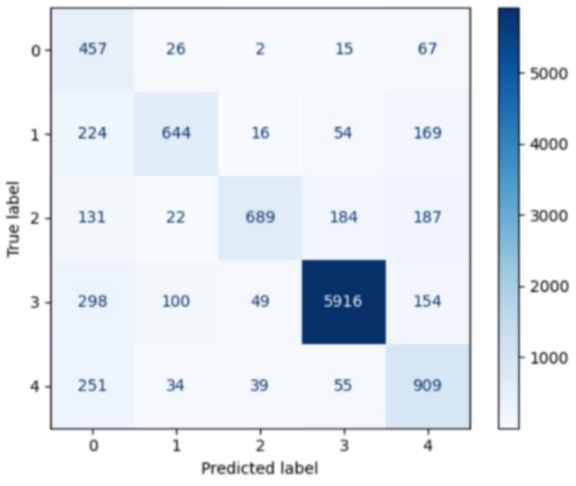


Figure 12. Confusion matrix of method II

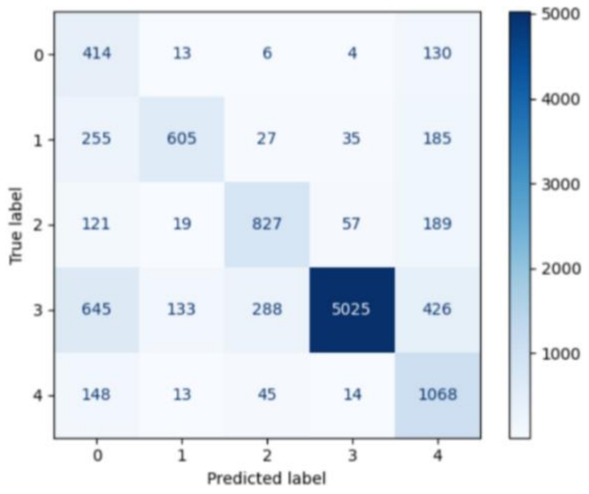


Figure 13. Confusion matrix of method III

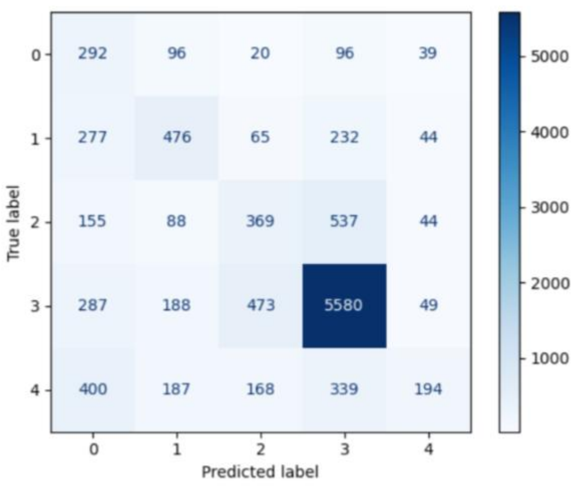


Figure 14. Confusion matrix of method IV

Table 4. Performance Evaluation

Method	*Class*	*Performances*			
		Precision	*Recall*	*F-Measure*	*Accuracy*
EfficientNet-B1	0	56%	42%	48%	-
	1	71%	53%	60%	-
	2	58%	52%	55%	-
	3	84%	97%	90%	-
	4	68%	32%	44%	-
	Avg.	**78%**	**77%**	**78%**	**77%**
EfficientNet-B1 + SMOTE	0	34%	81%	47%	-
	1	78%	58%	67%	-
	2	87%	57%	69%	-
	3	95%	91%	93%	-
	4	61%	71%	66%	-
	Avg.	**85%**	**81%**	**82%**	**81%**
EfficientNet-B1 +Non-Deep Learning Augmentation	0	26%	73%	39%	-
	1	77%	55%	64%	-
	2	69%	68%	69%	-
	3	98%	77%	86%	-
	4	53%	83%	65%	-
	Avg.	**83%**	**74%**	**77%**	**74%**
EfficientNet-B1 + Deep Learning Augmentation	0	21%	54%	30%	-
	1	46%	44%	45%	-
	2	34%	31%	32%	-
	3	82%	85%	84%	-
	4	52%	15%	23%	-
	Avg.	**66%**	**65%**	**65%**	**65%**

3.2. Discussions

When using the first method, EfficientNet, without imbalanced data handling, the confusion matrix in Figure 11 shows that class 3 (CMD) achieved the highest true positives. While in the other classes, most images are predicted to be in class 3. Table 4 also shows that class 3 (CMD) scored highest in all performances. While in the other classes, the scores are not larger than 60-70%. This happens because class 3 has the largest proportion of data that can be biased heavily towards the minority classes.

When the use of SMOTE is applied to method II, the confusion matrix in Figure 12 shows that the true positives are spread more evenly to all classes, and only a small portion of data from minority classes that are miss-predicted to class 3. Besides, the average scores for precision, recall, F1-Score, and accuracy in Table 4 are also enhanced by 7%, 4%, 4%, and 4% respectively.

When basic augmentation is applied before EfficientNet in method III, we can see in Figure 13 that the true positives are spread more evenly to all classes. However, compared to the results in Figure 8, the true positives of class 3 decreased from 6,409 to 5,580. The true positives of the other classes, which are a minority, increased from 263 to 292, from 572 to 605, from 610 to 827, and from 418 to 1,068 for each class 0, 1, 2, 4, respectively. Table 4 also shows the declining scores of about 3%, 1%, and 3% in accuracy, recall, and F1-score, respectively.

When neural style transfer is applied before EfficientNet in method IV, we can see in Figure 14 that the number of true positives for classes 1, 2, 3, and 4 decreased from 6,409 to 5,025, from 572 to 476, from 610 to 369, and from 418 to 194, respectively. Conversely, the true positive count for class 0 increased from 263 to 414. Notably, many images originally classified as class 1, 2, and 4 were misclassified as either class 0 or class 3. Table 4 also shows that all scores decreased to 65% and 66% for precision.

The results of applying basic augmentation and neural style transfer became worse because the transformations may disrupt and harm the original images. Unlike augmentation, SMOTE does not harm the original images because it creates synthetic data points using interpolation from the original images. The performance scores of neural style transfer are inferior to those achieved with basic augmentation. This disparity arises from the application of neural style transfer, which simultaneously alters all colors, styles, and patterns of the original image. In contrast, basic augmentation disrupts only one aspect at a time in each augmented image, leading to more controlled disruptions and, ultimately, better performance scores.

4. CONCLUSIONS

In classifying cassava leaf images into five classes, including Cassava Bacterial Blight (CBB), Cassava Brown Streak Disease (CBSD), Cassava Green Mottle (CGM), Cassava Mosaic Disease (CMD), and Healthy, the authors chose EfficientNet for the classification model. Following experimentation on choosing the best parameters, EfficientNet-B1 emerged as the chosen baseline model, configured with five epochs and a learning rate of 0.0006.

However, this classification task has unbalanced data issues. Therefore, the authors compared several imbalanced data handling methods to be applied to training data before being fed into EfficientNet, which are SMOTE, basic augmentation, and neural style transfer. Before applying any imbalance data handling method, EfficientNet demonstrated an F1-Score of 78%.

The application of SMOTE to the EfficientNet model surpasses the original model, yielding an impressive F1-Score of 82%. On the contrary, the utilization of basic augmentation reduces the F1-Score to 74% and to 65% for the utilization of neural style transfer. The declining scores are attributed to potential disruptions caused by the transformations in the original images. Notably, integrating SMOTE with EfficientNet-B1 with the chosen parameters became our baseline method for cassava disease classification based on leaf images.

ACKNOWLEDGEMENTS

The authors thank Bina Nusantara University for the research grant and for supporting this research.

REFERENCES

[1] Okwuonu et al., “Opportunities and challenges for biofortification of cassava to address iron and zinc deficiency in Nigeria,” *Global Food Security*, vol.28, 2021

[2] C. Fauquet, “Cassava Stories CMD Outbreak in Mainland SEA,” Global Cassava Partnership 21. [Online]. Available: https:// gcp21.org/. [Accessed: 25-Sep-2023].

[3] E. Mwebaze, T. Gebru, A. Frome, S. Nsumba, J. Tusubira, “iCassava 2019Fine-Grained Visual Categorization Challenge,” *arXiv preprint arXiv:1908.02900*, 2019.

[4] Makerere University AI Lab, “Cassava Leaf Disease Classification”, Kaggle. [Online]. Available: https://www.kaggle.com /competitions/cassava-disease/data/. [Accessed: 25-Sep-2023].

[5] H. Ayu, A. Surtono, D. Apriyanto, “Deep learning for detection cassava leaf disease”. *Journal of Physics: Conference Series*, vol. 1751, 2021.

[6] I. Sangbumrung, P. Praneetpholkrang, S. Kanjanawattana, “A Novel Automatic Method for Cassava Disease Classification Using Deep Learning. Journal of Advances in Information Technology,” *Journal of Advances in Information Technology*, vol. 11, no. 4, pp. 241–248, 2020.

[7] A. Maryum et al., “Cassava Leaf Disease Classification using Deep Neural Networks,” in *IEEE 18th International Conference on Smart Communities: Improving Quality of Life Using ICT, IoT and AI (HONET)*, 2021, pp. 32–37.

[8] Y. Zhong, B. Huang, C. Tang, “Classification of Cassava Leaf Disease Based on a Non-Balanced Dataset Using Transformer-Embedded ResNet,” *Agriculture*, vol. 12, no. 9, p. 1260, Sep. 2022.

[9] M. Liu, H. Liang, M. Hou, “Research on cassava disease classification using the multi-scale fusion model based on EfficientNet and attention mechanism,” *Journal of Frontiers in Plant Science*, vol. 13, Dec. 2022.

[10] J. N. Nabende et al., “Improving In-field Cassava Whitefly Pest Surveillance with Machine Learning,” *Journal of Frontiers in Plant Science,* vol. 41, Feb. 2022.

[11] F. Gao, J. Sa, Z. Wang, Z. Zhao, “Cassava disease detection method based on EfficientNet,” in 2*021 7th International Conference on Systems and Informatics (ICSAI)*, Chongqing, China, 2021, pp. 1–6.

[12] S. Aravind, S. Harini, Varun kumar K A, “Cassava leaf disease classification using Deep Learning,” *Natural Volatiles and Essential Oils (NVEO) Journal,* vol. 8, no. 5, pp. 9375–9389, 2021.

[13] H. Shahriar, P.S. Shuvo, Md.S.H. Fahim, Md.S. Sordar, Md.E. Haque, “Cassava leaf disease classification using deep learning and convolutional neural network ensemble,” Brac University, Bangladesh, 2022.

[14] G. Sambasivam, G. D. Opiyo, “A predictive machine learning application in agriculture: Cassava disease detection and classification with imbalanced dataset using convolutional neural networks,” *Egyptian Informatics Journal*, vol. 22, no. 1, pp. 27–34, 2021.

[15] U.K. Lilhore et al., “Enhanced Convolutional Neural Network Model for Cassava Leaf Disease Identification and Classification,” *Mathematics*, MDPI, vol. 10, no. 4, pp. 1–19, Feb. 2022.

[16] N. Chawla, K. Bowyer, L. Hall, W. Kegelmeyer, “SMOTE: Synthetic Minority Over-sampling Technique”. *Journal of Artificial. Intelligence Research (JAIR)*, vol. 16, no. 1, pp.321-357, 2022.

[17] Shorten, Connor and Taghi M. Khoshgoftaar. “A survey on Image Data Augmentation for Deep Learning.” *Journal of Big Data 6* (2019): 1-48.

[18] Q. Cai et al, " Image neural style transfer: A review", *Journal of Computers and Electrical Engineering* (2023), vol. 10.

[19] L. Perez & J. Wang, “The Effectiveness of Data Augmentation in Image Classification using Deep Learning.” *ArXiv abs/1712.04621* (2017).

[20] X. Zheng et al., “Stada: Style transfer as data augmentation”, *arXiv preprint arXiv:1909.01056* (2019).

[21] Ghiasi, G., Lee, H., Kudlur, M., Dumoulin, V., & Shlens, J. (2017). Exploring the structure of a real-time, arbitrary neural artistic stylization network. ArXiv, abs/1705.06830.

[22] Papers with Code. [Online] Available: https://paperswithcode.com/datasets?task=style-transfer/. [Accessed: 23-Jan-2024]

[23] M. Tan, Q.V. Le, “EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks”, in *36th International Conference on Machine Learning (ICML) 2019,* Long Beach, 9-15 June 2019, pp. 6105-6114.

[24] J. Lederer, “Activation functions in artificial neural networks: A systematic overview”. *arXiv preprint arXiv:2101.09957* (2021).