MobileNetV2-D and multiple cameras for swiftlet nest classification based on feather intensity

Denny Indrajaya[1], Hanna Arini Parhusip[1], Suryasatriya Trihandaru[1], Djoko Hartanto[2]
[1]Master of Data Science, Faculty of Science and Mathematics, Satya Wacana Christian University, Salatiga, Indonesia
[2]PT Waleta Asia Jaya, Salatiga, Indonesia

Article Info

Article history:

Received Dec 22, 2023
Revised Jan 30, 2024
Accepted Feb 25, 2024

Keywords:

Image classification
MobileNetV2-D
Multiple cameras
Sorting machine
Swiftlet nest

ABSTRACT

MobileNetV2-D is a modified version of MobileNetV2, which is the novelty of this article. The algorithm is used to classify swiftlet nests into seven classes. In 2023, PT Waleta Asia Jaya is required to achieve a 7-fold increase in the export quota of swiftlet nests. To meet the quota, the company made a machine that can recognize swiftlet nest objects, which are classified into seven classes based on feather intensity, namely BRS, BR, BST, BS, BBT, BB, and BB2 for the light feathers to the heavy feathers, respectively. The input image is a combination of four images from four cameras with different positions, which adds to the novelty of MobileNetV2-D for the particular problem here. From the evaluation that has been carried out, the accuracy value of the MobileNetV2-D model was better than the MobileNetV2 model, i.e., the accuracy value of the MobileNetV2-D model was 99.9928% for the training dataset and 94.0723% for the testing dataset. Moreover, the speed of MobileNetV2-D is better than MobileNetV2-architecture.

Corresponding Author:

Hanna Arini Parhusip
Master of Data Science, Faculty of Science and Mathematics, Satya Wacana Christian University
Salatiga, Central Java, Indonesia
Email: hanna.parhusip@uksw.edu

1. INTRODUCTION

PT Waleta Asia is one of the companies engaged in washing swiftlet nests that sells its products abroad. From 2023 forward, the company will receive an additional quota for exports of more than 7 times the quota in 2022. Therefore, it is necessary to increase the productivity of the company. Considering the high number of export increases as the main problem, adding employees to meet market needs is not effective. Increasing the number of workers will require a larger space as well. As a solution to this problem, it is necessary to apply artificial intelligence (AI) to automate and increase the speed of the overall swiftlet nest washing process. The main contribution of the authors is to devise an algorithm that can help swiftlet nest sort quickly and easily. The article here proposes a new version of MobilenetV2 called MobilenetV2-D that solves this problem. The authors have not found any literature related to a similar work. The new contribution is made by presenting MobilenetV2-D here.

In washing swiftlet nests at PT Waleta Asia Jaya, several processes must be followed. One of the processes is sorting swiftlet nest raw materials based on feather intensity into as many as seven classes, namely BRS, BR, BST, BS, BBT, BB, and BB2 for nests with light to heavy feather intensity, respectively. This sorting process is usually done manually. As a result, the sorting result may vary from one person to another. Moreover, it allows optical illusions to occur that make the sorting results unstable. These are the major problems that will be handled. Therefore, the main contribution of this research is improving the classification of swiftlet nests from human activities to machine activities supported by AI. One method that can be used to make programs on machines that can classify images is the convolutional neural network (CNN) method.

CNN is one of the AI or deep learning (DL) methods that have been used in various fields, especially image recognition. The classification of chili conditions [1], fruit quality classification [2], and classification of unwashed eggs applied to a sorting machine [3] are the examples addressed in the literature. In the case of swiftlet nest detection, swiftlet nests are classified into 3 classes, which apply CNN for feature extraction as well [4]. CNN has been used for various things, including sorting machines. However, no authors have been found to use the CNN method for the classification of swiftlet nests based on feather intensity for a sorting machine in the business of swiftlet nests around Asia. Therefore, the article here contributes to making a classification model for swiftlet nests. The images in sorting machines are based on feather intensity (seven classes) with the updated MobileNetV2 algorithm in a new architecture called MobileNetV2-D.

Therefore, the major findings here will demonstrate the architecture of MobileNetV2-D and the algorithm in the related sorting machine for the novel classification of swiftlet nest. The architecture is compared to the standard MobileNetV2 that has been available previously. The reader will learn how the new algorithm of MobileNetV2-D improves the accuracy and reduces the number of parameters compared to MobileNetV2. As a result, the MobileNetV2-D has better speed than the MobileNetV2, which is the required algorithm to be implemented in high industrial activity.

2. METHOD

In this research, there are several steps to create a swiftlet nest image classification model that is grouped into seven classes based on feather intensity for the sorting machine. These steps are outlined in Figure 1. In addition, in carrying out this research, a literature study process regarding CNN was also carried out. A detailed explanation of the theory used in this research, along with the overall process, is explained in the next section.

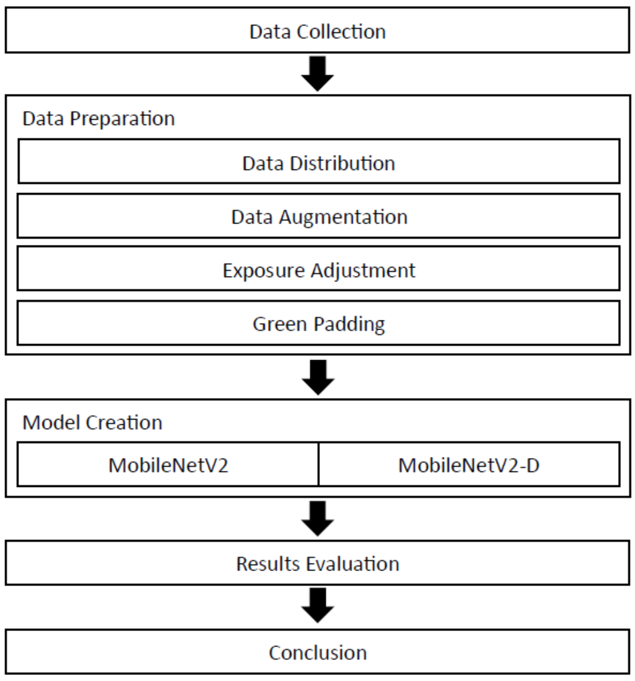


Figure 1. Research process outline

2.1. Materials and instruments

We collected data on seven classes of swiftlet nests from PT Waleta Asia Jaya in Salatiga named BRS, BR, BST, BS, BBT, BB, and BB2, where the swiftlet nest names were sorted based on feather intensity. The first is BRS, which states that swiftlet nests have the least feather intensity, so they are easy to remove and clean. The last, BB2, is the swiftlet nest that is the most difficult to clean. These types are illustrated in Figure 2. Initially, the swiftlet nests were sorted manually. Therefore, the picture-taking process in a prototype sorting machine was conducted for each class of the nests, and the images with the same class were also saved in one folder.

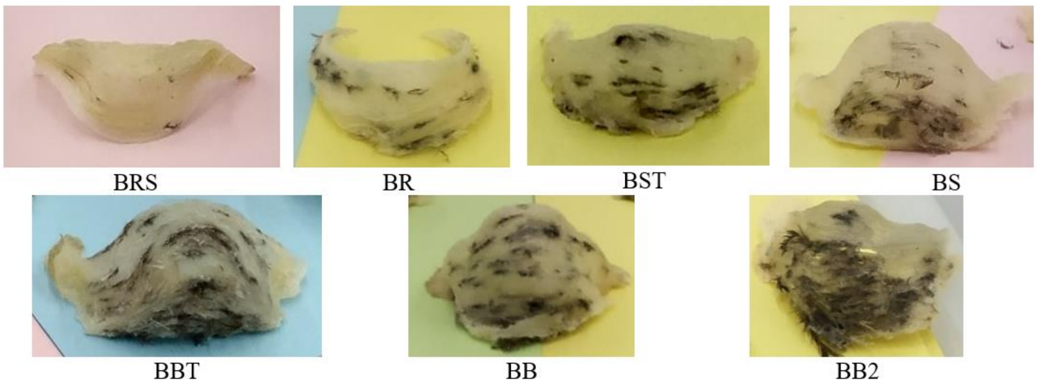


Figure 2. Swiftlet nest classes (source: company collections)

A prototype sorting machine was used to test algorithms in classification, while a sorting machine on an industrial scale is not shown here because the company does not publish the machine to the public. However, in order to apply the algorithm to classify an automatic sorting machine, a prototype sorting machine device is used to become a training device for the real sorting machine. The prototype that has been designed has a place to take pictures of swiftlet nests. The design is illustrated in Figure 3.

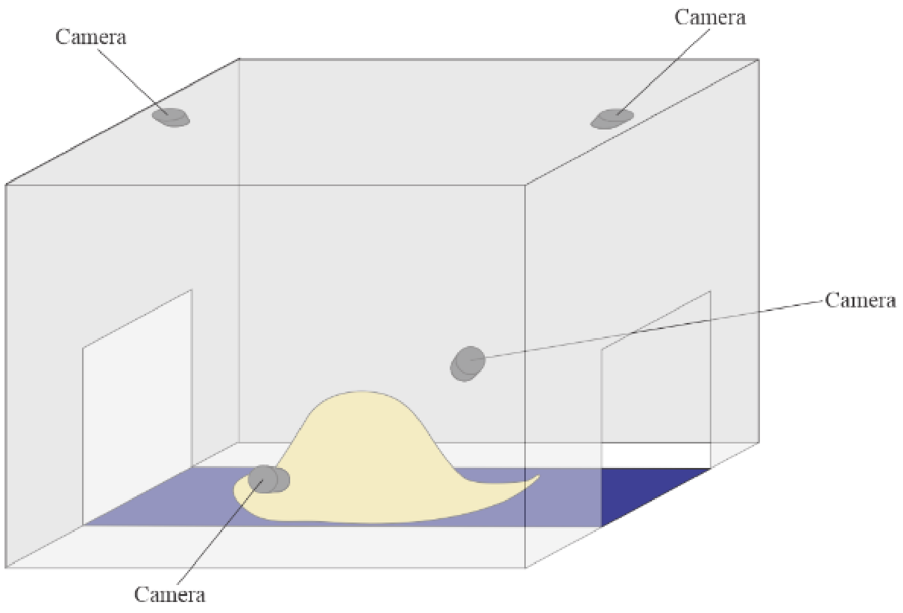


Figure 3. Swiftlet nest shooting place design in the sorting machine

The improvement of the sorting machine in this research is based on the use of four cameras producing four images in one object for classifying one object. In the development of sorting machines, the position of the camera and the conditions inside the shooting place are always adjusted so that taking pictures is always done for each new shooting location condition. Samples of image data obtained from a sorting machine using four cameras are shown in Figure 4, and a breakdown of the number of images for each class is shown in Table 1. The position of the nest is very important because the part of the nest that determines the nest class must be visible to the camera when taking pictures. Therefore, the preparation of training data is done by taking pictures of swiftlet nests in various positions and arranging them in such a way that no two or more nests are overlapping, as shown in Figure 4.

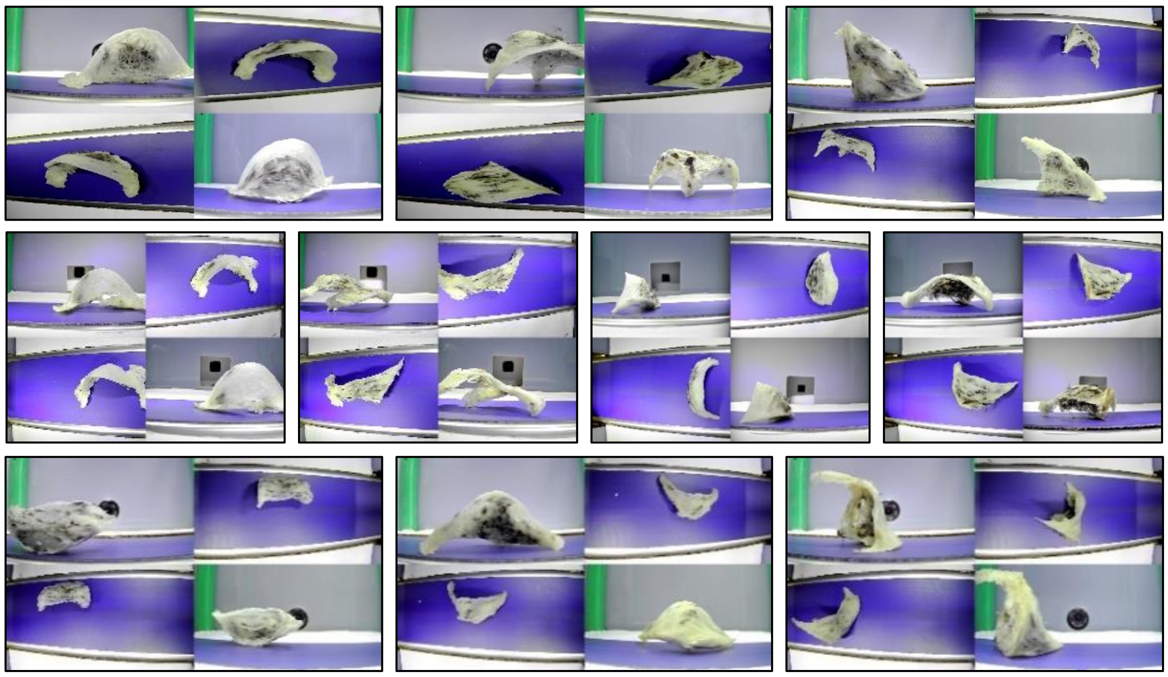


Figure 4. Image data sample from sorting machine (source: company collections)

Table 1. Number of images of each class on the whole images

Classes	Image number
BRS	1,097
BR	1,073
BST	1,276
BS	1,269
BBT	1,069
BB	1,322
BB2	1,359

2.2. Necessary techniques for producing the best data training

Before conducting further data processing, data division was carried out first. In this case, the data distribution for the testing dataset contains 241 images for each class, so there are 1,687 images (20% of all images) for the testing dataset, and the remaining 80% of all data are allocated to the training dataset. Then augmentation was carried out until the number of images for each class was 2,000, so the amount of data for the training dataset was 14,000 and for the testing dataset was 1,687.

To produce precisely identifiable training data, several technical problems must be trained and adjusted. This is done due to the complexity of swiftlet nests. Suppose the swiftlet nest position in different positions is expressed as an object in a different classification when it should be an object in the same class. Therefore, the use of four cameras and the lighting set determines the correct classification. With four cameras, there are four images for one object in one class. We employ a combination of four images from four cameras with different positions that are quite complex. A lot of data for the training process is required. The image augmentation involves randomizing the position and mirroring the four images in the image. We apply winter and summer color filters to the images on the training dataset so that more images are obtained. Then, on the training dataset, more images were obtained. With that, the model can learn the location and condition of the nest in the image better [5], [6].

In making this model, an image exposure adjustment is made first before the image is inputted into the model, with the aim of making the color of the feathers in the nest more clearly visible. Then, because the input of the model is a square image, the green padding is also given. The green color is chosen because all pixels in the image are used in the classification process, where the green color is not found in the nest. Therefore, before the image is inserted into the designed CNN model, it is necessary to do similar things, namely exposure adjustment and green padding (preprocessing). The augmentation results in the form of randomization of the location of four frames in one image, applying winter and summer color filters, and also applying exposure adjustment and green padding are shown in Figure 5.

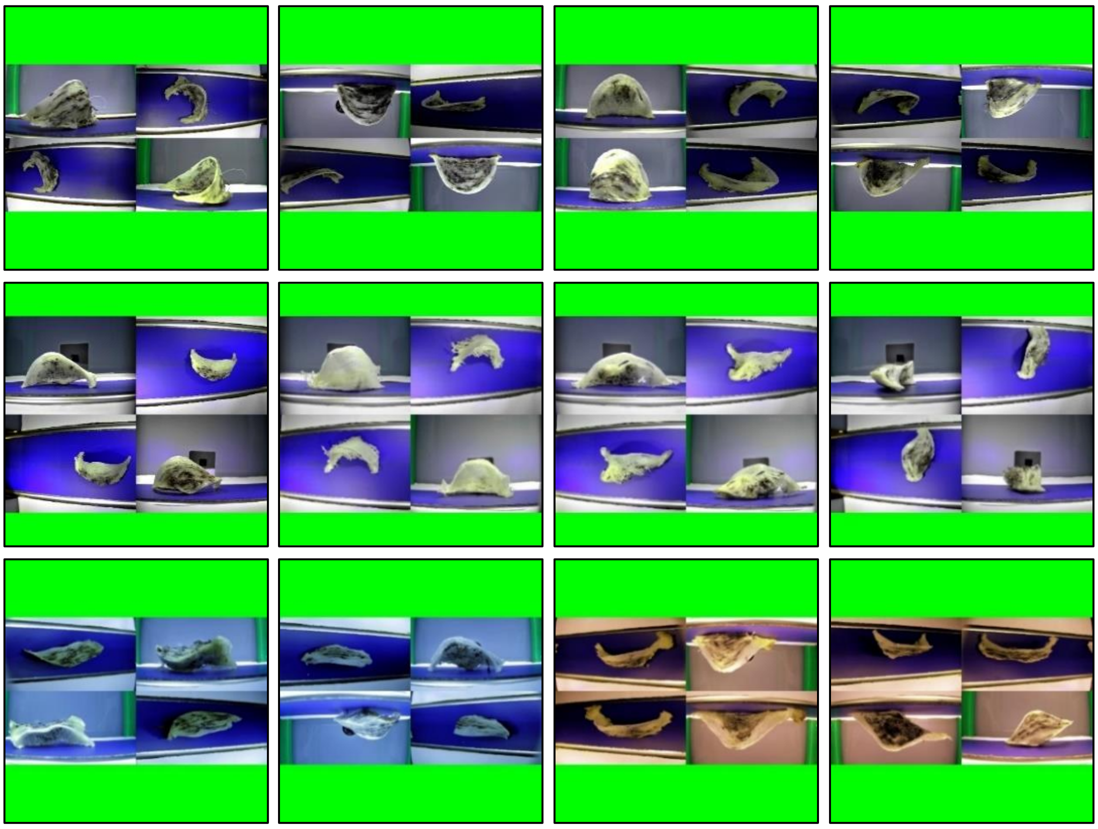


Figure 5. Examples of image augmentation include image preprocessing results

2.3. Convolutional neural network

CNN is one of the methods used in performing image recognition. CNN is known as a convolution process, which is a mathematical operation typically used for image processing [7]. The output of the convolution process is referred to as a feature map [8]. In CNN architecture, the formula of the convolutional layer is shown, i.e., [9], [10].

$$Y_{i,j,k} = f\left(\sum_{r=0}^{R-1}\sum_{s=0}^{S-1}\sum_{c=0}^{C-1} W_{r,s,c,k}X_{i+r,j+s,c} + b_k\right) \quad (1)$$

Where:
$Y_{i,j,k}$ = the output in the form of a feature map at position (i,j) and the k-th output channel,
$X_{i+r,j+s,c}$ = the input image or feature map at position $(i+r, j+s)$ and the c-th input channel,
$W_{r,s,c,k}$ = the weight of the filter at position (r,s), the c-th input channel, and the k-th output channel,
b_k = the bias for the k-th output channel,
R = the width of the filter,
S = the length of the filter,
C = the number of channels in the input image,
f = the activation formula,
i = $1, 2, 3$, ..., width of the feature map – R,
j = $1, 2, 3$, ..., length of the feature map – S.

There is a filter size on CNN used for the convolution process. In the process of convolution, it is possible to change the size of the image. Therefore, it is necessary to make adjustments according to needs,i.e., specifying the stride and padding sizes. The formula for determining the size of the feature map is shown here [11], i.e.,

$$O = \frac{W-F+2P}{S} + 1 \tag{2}$$

where:
O = the size of the feature map output,
W = the size of the image or feature map carried out by the convolution process,
F = the size of the filter,
P = the size of the padding on the sides of the image,
S = the stride size or filter step size when moved on an image or feature map.
One process that can be done after running a filter on an image or feature map is to apply an activation function such as the rectified linear unit (ReLU) [9], i.e.,

$$f(x) = max(0, x) \tag{3}$$

with this activation function, the negative values obtained earlier will be changed to 0. Furthermore, the batch normalization (BN) process is also carried out, which will normalize the data obtained from the previous process. BN is a technique used to speed up the training process and improve its accuracy [12].

In the architecture, there is also an average pooling process. This process is one of the ways to reduce the dimensions of images or feature maps [13]. This reduction is done so that fewer parameters are calculated. However, in the process of reducing the size of this image, important information about the data is maintained [14]. However, this average pooling was not used here. Instead, we employ the global average pooling as shown by (4), which is almost similar to average pooling [11], [15], i.e.,

$$y_k = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} x_{i,j,k} \tag{4}$$

where:
$x_{i,j,k}$ = the k-th feature map at position (i, j),
H = the height of the feature map,
W = the width of the feature map,
y_k = the average value of the k-th feature map of all positions.

Various architectures have been developed for image classification. However, there is one architecture that can provide good enough accuracy and has fairly good speed as well, namely the MobileNet architecture [16]. The MobileNet architecture uses BN and ReLU activation functions for depthwise convolution and pointwise convolution [17], while the MobileNetV2 structure uses the ReLU6 activation function [18], i.e.,

$$f(x) = min(max(x, 0), 6) \tag{5}$$

Now, we will propose the architecture of MobileNetV2-D. It was inspired by the MobileNetV2 architecture, i.e., we applied the bottleneck concept. We obtain a smaller number of parameters and layers than in MobileNetV2, making the computational process lighter and faster. The details are shown in the next subsection.

2.4. Model architecture of MobileNetV2-D

Using the bottleneck and conv blocks shown in Figures 6 and 7, the architecture of MobileNetV2-D is shown in Table 2. The algorithm applied bottleneck and ReLU6 operators [19]. This model was designed using the bottleneck operator as well. As mentioned above, it was made simpler with 1,263,943 parameters, which is less than the used MobileNetV2 architecture in general or the model provided by the TensorFlow framework, which has 2.266.951 parameters if the output units are seven. Thus, the model MobileNetV2-D architecture provides faster results.

The MobileNetV2-D architecture shown in Table 2 has the main differences with MobileNetV2 in the number of hidden layers, the number of units in each layer, the size of kernels, and the expansion factor values in the bottleneck. In MobileNetV2-D architecture, the number of layers and the number of maximum units of all hidden layers are less than in MobileNetV2 architecture. The kernel sizes used in the

MobileNetV2-D also varied, namely (2×2), (3×3), (4×4), and (5×5), while in the MobileNetV2 architecture just (3×3). The expansion factor used in MobileNetV2-D is more varied than in MobileNetV2. The reason we use a smaller number of hidden layers and units is to make the model faster. Likewise, the reason the expansion factor values used are different and relatively smaller than the expansion factor values in the MobileNetV2 architecture is to make the number of units at the bottleneck smaller so that the model becomes faster. Then the kernel sizes used are different, which aims to make the process of extracting features better, where there are small and large sizes. The use of a large kernel size can provide a more representative feature value, but it can also require more computation. Therefore, to make the computational speed still not too much, a smaller kernel size is used in a few layers.

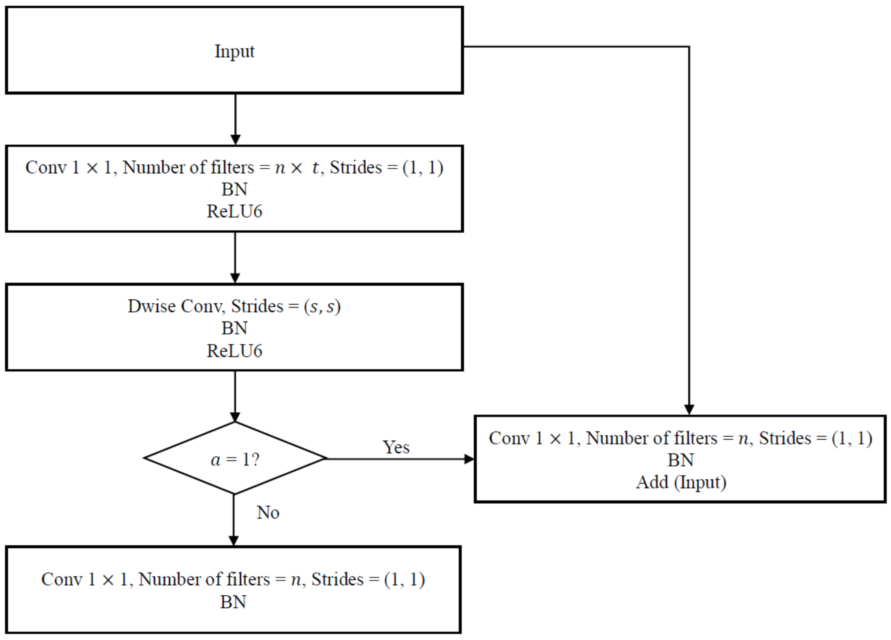

Input
Conv 1 × 1, Number of filters = $n \times t$, Strides = (1, 1)
BN
ReLU6
Dwise Conv, Strides = (s, s)
BN
ReLU6
$a = 1$?
Yes
No
Conv 1 × 1, Number of filters = n, Strides = (1, 1)
BN
Add (Input)
Conv 1 × 1, Number of filters = n, Strides = (1, 1)
BN

Figure 6. The bottleneck used in this study

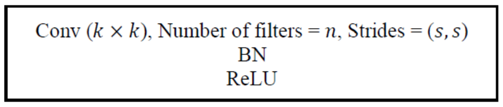

Conv $(k \times k)$, Number of filters = n, Strides = (s, s)
BN
ReLU

Figure 7. Conv block used in this study

Table 2. The MobileNetV2-D architecture

Input	Operator	n	k	t	s	a
480 × 480 × 3	Conv block	24	2	-	2	-
240 × 240 × 24	Bottleneck	24	2	1	1	1
240 × 240 × 24	Bottleneck	16	3	4	2	0
120 × 120 × 16	Bottleneck	16	2	4	1	1
120 × 120 × 16	Bottleneck	32	3	5	2	0
60 × 60 × 32	Bottleneck	32	3	5	1	1
60 × 60 × 32	Bottleneck	32	3	5	1	1
60 × 60 × 32	Bottleneck	32	3	5	1	1
60 × 60 × 32	Bottleneck	32	3	5	1	1
60 × 60 × 32	Bottleneck	64	4	6	2	0
30 × 30 × 64	Bottleneck	64	3	6	1	1
30 × 30 × 64	Bottleneck	128	4	6	2	0
15 × 15 × 128	Bottleneck	128	3	6	1	1
15 × 15 × 128	Conv block	256	3	-	1	-
15 × 15 × 256	Conv block	64	5	-	1	-
1 × 1 × 64	Global average pooling	64	-	-	-	-
1 × 1 × 64	Dense	7	-	-	-	-

In the model training process, the stochastic gradient descent (SGD) optimizer was used. SGD is a simple optimizer but very effective in the training model process. Using this optimizer, the number of epochs for the training process can be reduced [20], [21]. The learning rate for each epoch was not the same. This learning rate is designed so that at each epoch there is an increase and decrease in the value of the learning rate, as well as a constant learning rate value, so that the update weights of the model can be better [22].

$$lr_i = \begin{cases} lr_0 & \text{, if } i = 0 \\ lr_0 + (lm - lr_0)\sin\left(\frac{i\pi}{2M}\right) & \text{, if } i \leq M \\ lc + \frac{lm}{2}\left(sin\left(\frac{i-M}{N-M-C} + \frac{\pi}{2}\right) + 1\right) & \text{, if } i \leq N - C \\ lc & \text{, if } i \leq N \end{cases} \tag{6}$$

Where:
$i = 1, 2, \ldots, N$,
lr_i = the learning rate at the i-th epoch,
lr_0 = the initial learning rate value,
lm = the maximum value of the learning rate,
lc = the constant learning rate value,
N = the number of epochs,
M = the number of epochs that have an increased learning rate at the beginning of the training process,
C = the number of epochs that use a constant learning rate at the end of the training process.

The hyperparameters used in the model training process in this study are a batch size of eight, and the variable values of (6) used for the learning rate are $lr_0 = 0.001$, $lm = 0.01$, $lc = 0.00001$, $N = 30$, $M = 5$, and $C = 10$. In addition, from the 30 models obtained at each epoch, the best model will be chosen based on the smallest loss value obtained at each epoch. The used loss value in this case is the categorical cross entropy [23], i.e.,

$$L = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C} y_{i,j}\log(p_{i,j}) \tag{7}$$

where N is the amount of data, C is the number of classes, y is the actual class of data, and p is the prediction probability of the actual data class [23].

2.5. Accuracy

Evaluation of an image classification model can be done using an accuracy formula, which is based on the number of correctly and incorrectly classified images. The number can be tabulated in a matrix called a confusion matrix [24]. Table 3 is an example of a confusion matrix containing true positive (TP), false negative (FN), false positive (FP), and true negative (TN) values. These values are used to obtain accuracy values that indicate the performance of the classifier method. The formula for obtaining the accuracy value is shown in (8) [25].

$$Accuracy = \frac{TP+TN}{n} \tag{8}$$

Table 3. Confusion matrix

		Predictions	
		Class 1	Class 2
Actual	Class 1	TP	FN
	Class 2	FP	TN

Where:
TP = the number of classification results that are positive and correct,
TN = the number of classification results that are negative and correct,
FP = the number of classification results that are positive and false,
FN = the number of classification results that are negative and false.

3. RESULTS AND DISCUSSION

As mentioned above, this article demonstrates a new architecture, MobileNetV2-D, specifically designed for the special purpose of classifying swiftlet nest based on feather intensity. In this section, the results of the architecture that has been created are applied and discussed. The MobileNetV2-D architecture and MobileNetV2 architecture are trialed five times for experiments for classifying swiftlet nest images from four cameras into seven classes. We assume that by observing five times the accuracy and loss values for each architecture, we have guaranteed that the results can be used for other experiments with the new data in further research. Thus, the used hyperparameter values are the same in five experiments, yielding the same learning rate used for each epoch with 30 epochs. The learning rate given for each epoch is shown in Figure 8. Using the same data and the same hyperparameter values, this section will show the comparisons of five experiments from the MobileNetV2-D architecture and the MobileNetV2 architecture.

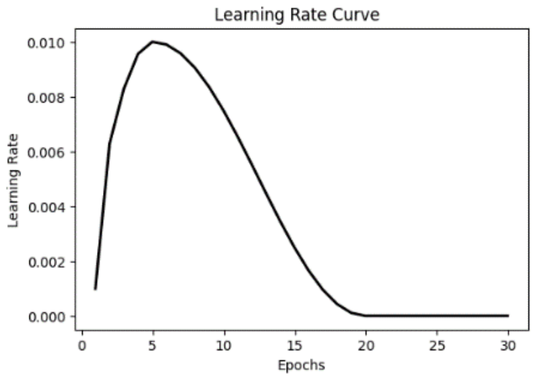


Figure 8. Learning rate curve

3.1. Observing the stability of the architecture through five trials of experiments

Section 2 has shown the MobileNetV2-D architecture that is characterized in Table 2. Here, we employ the architecture to show one of the training results by considering the accuracy and loss values of 30 epochs with five experiments. One of these experiments is depicted in Figure 9. The architecture gives good results for the training process with 30 epochs. With the five experiments that have been conducted, the graphs obtained are also similar, which are not shown here. We conclude that the architecture of MobileNetV2-D is quite stable.

In Figure 9(a), the accuracy obtained for each epoch does not fluctuate significantly. This also applies to the change in the loss value obtained at each epoch, which also tends to decrease, as shown in Figure 9(b). This shows that the model obtained is quite optimal for the hyperparameters. Thus, increasing the number of epochs will not significantly change the performance of the model.

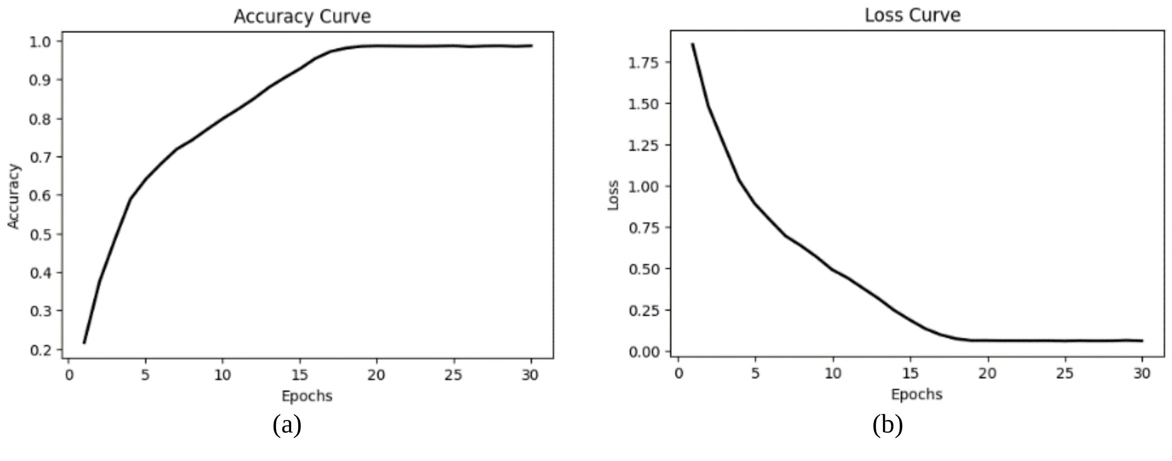


Figure 9. Graph of training image classification results from MobileNetV2-D model (a) accuracy curve and (b) loss curve

Let us consider the five experiments of the MobileNetV2 architecture with the same hyperparameters as above. One of the results is shown in Figure 10. The five experiments that have been conducted also produce a similar graphical shape, as shown in Figure 10(a) for the accuracy values and Figure 10(b) for the loss values. Therefore, we are not present here. We also conclude that the MobileNetV2 model can provide fairly stable results for each training process performed.

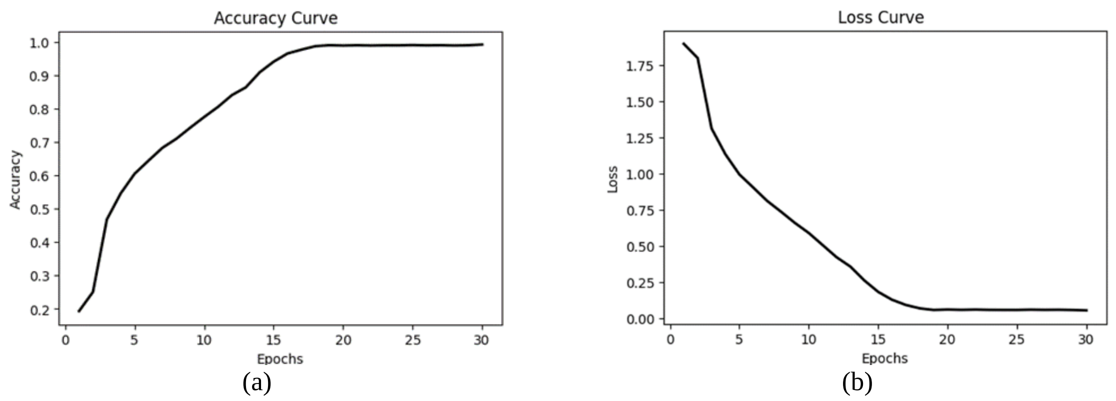


Figure 10. Graph of training image classification results from MobileNetV2 model (a) accuracy curve and (b) loss curve

3.2. Accuracy and loss values vs. the number of epochs

Now, we mimic the influence of the number of epochs used in our observation. The accuracy values from the MobileNetV2 model obtained for each epoch tend to increase to 100%, as shown in Figure 10(a). This implies that the loss values for each epoch tend to decrease to zero, as shown in Figure 10(b). Just like the results of the previous MobileNetV2-D model process, the use of an epoch value of 30 is appropriate because the model already looks optimal with the used hyperparameters. Therefore, models from the MobileNetV2-D architecture and the MobileNetV2 architecture give optimum results, indicating a tendency toward 100% accuracy and zero loss for both architectures.

3.3. Values of accuracy and loss in relation to learning rate

The use of different learning rate values in each epoch formulated in this research gives quite good results, as depicted in Figures 9 and 10. The performance of the MobileNetV2-D and MobileNetV2 models cannot be further improved by increasing the learning rate. The use of high enough learning rate values in the first five epochs can provide a fairly fast increase in accuracy value and a fairly fast decrease in loss value. The learning rate then begins to decrease from the 6th epoch to the 20th epoch. The value of loss tends to decrease without fluctuating significantly. Likewise, the accuracy value obtained also increases without fluctuating significantly. After about the 20th epoch, the accuracy of the model was close to 100%, so from the 21st epoch to the 30th epoch, the change in accuracy value and loss was not so significant. From the 21st epoch to the 30th epoch, there was also no significant decrease in accuracy. The learning rate used from the 21st epoch to the 30th epoch had a small and constant value. Thus, the weight change on the model during the training process was not too significant, which had an impact on the changes in the accuracy value obtained in each epoch.

The accuracy curves and loss curves of the MobileNetV2-D and MobileNetV2 models are almost the same. Both models obtain high accuracy values for the training dataset during the training process. This also shows that the used training dataset is quite good, even though there are still some nest classifications by employees that are slightly less consistent. Thus, the difference in the accuracy value of the training dataset obtained is quite small for both architectures. Now, the goodness of the model in the testing dataset shown in the next subsection.

3.4. Models' performance

Five experiments of MobileNetV2-D architecture and MobileNetV2 architecture were evaluated by calculating accuracy values and loss values. The results of the accuracy value calculation are shown in Table 4. From the five training processes for each architecture, different accuracy values are obtained, but the differences are not significant. This shows that the model obtained is quite optimal using the learning rate formula compiled in this study.

Table 4 represents the accuracy values obtained from the MobileNetV2 models using the training dataset. Three experiments reach 100%. However, the accuracy values obtained from the testing dataset are below 90%. This can happen because the training dataset still allows there to be incorrect data. However, this problem is quite solved by the MobileNetV2-D models. Out of five experiments, the accuracy values obtained for testing datasets are above 90%, although the accuracy for the training dataset does not reach 100%. Furthermore, all five experiments using the testing dataset show that the MobileNetV2-D architecture has higher accuracy values than the MobileNetV2 architecture. Therefore, the best model from five experiments with higher accuracy obtained is the MobileNetV2-D model in the first experiment, with an accuracy value of 99.9928% for the training dataset and 94.0723% for the testing dataset.

Table 4. Accuracy values of 5 MobileNetV2-D models and 5 MobileNetV2 models (learning rate is given by values shown in Figure 8 for each epoch)

Experiments	Training dataset		Testing dataset	
	MobileNetV2-D	MobileNetV2	MobileNetV2-D	MobileNetV2
1	99.9928%	99.9929%	94.0723%	89.3894%
2	99.9786%	100%	93.1239%	87.6111%
3	99.9714%	100%	93.7759%	86.9591%
4	99.9857%	99.9929%	93.4203%	89.6266%
5	99.9643%	100%	91.2270%	88.6781%

The MobileNetV2-D and MobileNetV2 architectures have been explained in section 2 where the modifications are aimed at making the model faster and providing better accuracy with the device specification is listed in Table 5. We observe that the obtained frame per second (FPS) values for the MobileNetV2-D model and MobileNetV2 were 14.24 and 13.59, respectively. Therefore, the MobileNetV2-D model not only provides higher accuracy values but also provides better speed than MobileNetV2.

Table 5. Device specification

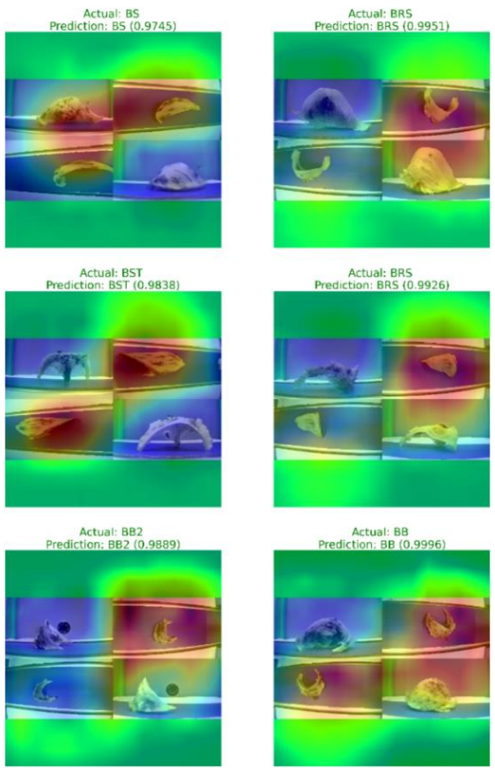
Parameters	Information
Processor	AMD Ryzen 7 6800H with Radeon Graphics (16 CPUs), ~3.2 GHz
RAM	16 GB
Operating system	Windows 11

3.5. Heatmap results

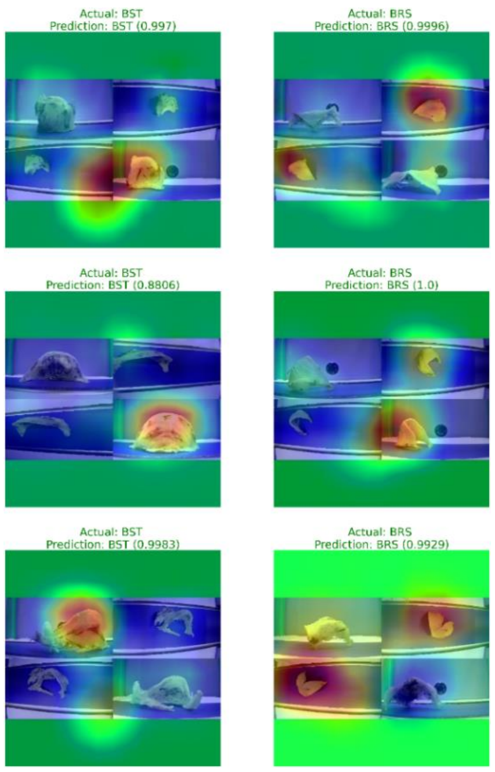
In testing the performance of image classification models, it is important to pay attention to the feature maps generated by the model. During the training process, the model may produce unexpected feature maps. This typically occurs due to insufficiently varied data, causing the model to extract features that are not appropriate, as well as due to the hyperparameters used. Therefore, during model testing with the testing dataset, feature maps are displayed by heatmaps.

The classification results, along with the heatmaps indicating the most important parts of the image in classifying the images obtained from testing the model. Figure 11 shows the correct classification using the testing dataset, and Figure 12 shows incorrect classifications. If we look at all the results, most error rates that occur are only 1 level. This means that if the class is sorted from the light feather intensity to the heavy feather intensity, most classification errors obtained are very small. Similar errors occur in manual sorting by employees. The classification error generated by the models is still tolerable.

To make a more in-depth evaluation of the models obtained, Figures 11 and 12 already have a heatmap that shows the features on the hidden layer of the last convolution process before global average pooling is carried out. In Figure 11(a), the correct classification results with the MobileNetV2-D model are shown. Many images have appropriate feature maps, although there are still some feature maps with less precise locations. In this case, it does not mean that they are entirely inaccurate; rather, within one image, the generated feature maps may include both accurate and inaccurate parts. This also applies to testing the MobileNetV2 model with correct classification results, as shown in Figure 11(b). It is evident from the result that the feature maps generated by the MobileNetV2 model appear slightly more precise in capturing the objects that determine the image class. However, there are still some areas of imprecision with objects due to the number of hidden layers and parameters, i.e., MobileNetV2 has more than MobileNetV2-D. For incorrect classification results, the generated feature maps are appropriate. Figures 12(a) and 12(b) depict the MobileNetV2-D model and the MobileNetV2 model, respectively.

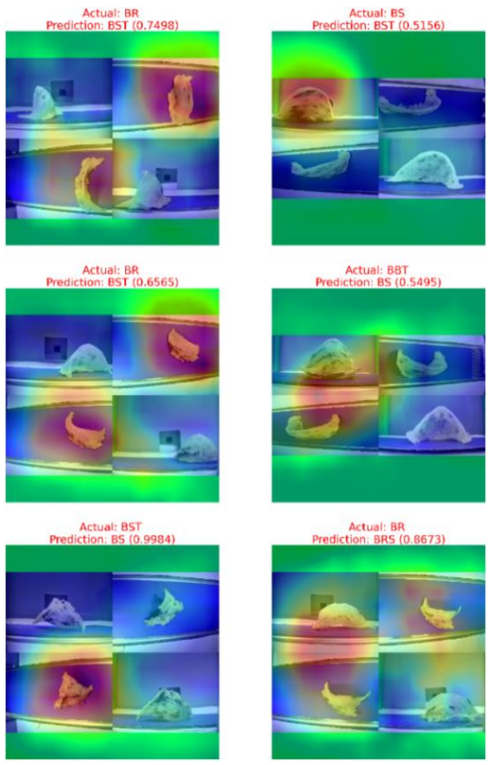


(a)

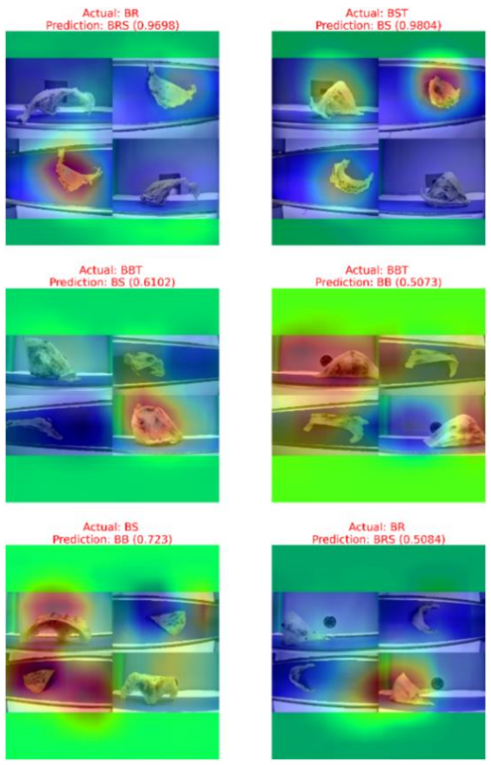


(b)

Figure 11. Correct classification results (a) MobileNetV2-D and (b) MobileNetV2



(a)



(b)

Figure 12. Incorrect classification results (a) MobileNetV2-D and (b) MobileNetV2

3.6. Probability results

From the evaluation of the overall data with examples in Figures 11 and 12, the probability value of the correct classification result tends to have a probability value above 90%. As for the incorrect classification, results tend to be below 90%; many results are even below 80%, although a few results are still above 90%. Incorrect classification results with probability values above 80%, some classification results appear to give correct results when reconsidering the swiftlet nest class. This means that the actual data class is not correct. However, it needs to be noted that classifying the nest image based only on images from the camera is influenced by the camera's angle of view, which affects the intensity of feather color in the nest.

3.7. Remark

Our study demonstrates that the MobileNetV2-D architecture could provide better results than MobileNetV2. With the obtained accuracy value, the model can be used for sorting machine purposes for further research or next generation development. The influence of data is very important in making image classification models. Particularly, the classification of swiftlet nest images, which are grouped into seven classes based on feather intensity, the quality of the data needs to be improved. The quality of the data in question is more consistent grouping of swiftlet nests because the used data are the result of sorting from the several experts in this field manually, which allows for different perspectives in determining nest classes.

Based on the overall evaluation results, besides the fact that MobileNetV2-D has good accuracy, this model also has good speed, so it can be an alternative to MobileNetV2 when applied to sorting machines with four cameras arranged as shown in Figure 3. Note that MobileNetV2-D provides better accuracy than MobileNetV2 if it is used to classify images from four cameras with a total of seven classes. In addition, the object being classified is a swiftlet nest, which is distinguished based on the feather intensity instead of different items. This is the limitation of our results. Therefore, further and more in-depth studies may be needed. The MobileNetV2-D architecture must be confirmed with several experiments with different objects and a larger number of classes, as well as a different number of cameras, to see the performance of this architecture. Further development can also include modifications to the MobileNetV2-D architecture with consideration of the architectural modifications described in the method section.

4. CONCLUSION

The article here addresses the new algorithm in the CNN method called MobileNetV2-D. It was proposed to classify swiftlet nests implemented on a sorting machine with four cameras. The major finding was the MobileNetV2-D architecture as a modification of the MobileNetV2 architecture. MobilenetV2-D has a smaller number of parameters to classify the swiftlet nest into seven classes based on the feather intensity obtained. MobileNetV2-D can provide better accuracy and speed than MobileNetV2, i.e., 99.9928% and 94.0723% for the training dataset and the testing dataset, respectively. We also gain the FPS value of 14.24. In addition, the model can classify a swiftlet nest image captured using four cameras from various camera angles and extract the precise features, namely the side of the used nest, for determining classification results. This finding does not refute previous research. The description of the architecture development in the method chapter shows that the new MobileNetV2-D can improve the quality of the model. In addition, the architecture has not been used for the classification of other images with a larger number of classes. Therefore, to conduct further research on the MobileNetV2-D architecture, modeling experiments can be carried out with various data and various numbers of cameras. Readers can utilize this model architecture for image classification modeling, especially for the purpose of building a sorting machine with four cameras. In addition, readers can also see the modified parts of the MobileNetV2 architecture that can be used as a reference for the development of new models. MobileNetV2-D can be employed as an alternative for the development of sorting machines with various cameras.

ACKNOWLEDGEMENTS

The article here is supported by Satya Wacana Christian University in the Bootcamp Writing Activity 2023 and PT Waleta Asia Jaya for providing funds, facilities, and data for this article.

REFERENCES

[1] T. Purwaningsih, I. A. Anjani, and P. B. Utami, "Convolutional neural networks implementation for chili classification," *Proceeding - 2018 International Symposium on Advanced Intelligent Informatics: Revolutionize Intelligent Informatics Spectrum for Humanity, SAIN 2018*, 2018, pp. 190–194, doi: 10.1109/SAIN.2018.8673373.

[2] H. Suhendar, V. Efelina, and M. Ziveria, "Fruit quality classification using convolutional neural network," *Journal of Physics: Conference Series*, vol. 2377, no. 1, 2022, doi: 10.1088/1742-6596/2377/1/012015.

[3] A. Nasiri, M. Omid, and A. Taheri-Garavand, "An automatic sorting system for unwashed eggs using deep learning," *Journal of Food Engineering*, vol. 283, no. 1, 2020, doi: 10.1016/j.jfoodeng.2020.110036.
[4] D. Indrajaya, A. Setiawan, D. Hartanto, and H. Hariyanto, "Object detection to identify shapes of swallow nests using a deep learning algorithm," *Khazanah Informatika : Jurnal Ilmu Komputer dan Informatika*, vol. 8, no. 1, pp. 139–148, 2022, doi: 10.23917/khif.v8i2.16489.
[5] S. N. Saydirasulovich, M. Mukhiddinov, O. Djuraev, A. Abdusalomov, and Y.-I. Cho, "An improved wildfire smoke detection based on YOLOv8 and UAV images," *Sensors*, vol. 23, no. 20, 2023, doi: 10.3390/s23208374.
[6] F. F. Alkhalid, A. Q. Albayati, and A. A. Alhammad, "Expansion dataset COVID-19 chest x-ray using data augmentation and histogram equalization," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 12, no. 2, pp. 1904–1909, 2022, doi: 10.11591/ijece.v12i2.pp1904-1909.
[7] M. Xin and Y. Wang, "Research on image classification model based on deep convolution neural network," *EURASIP Journal on Image and Video Processing*, vol. 2019, no. 1, 2019, doi: 10.1186/s13640-019-0417-8.
[8] Purwon, A. Ma'arif, W. Rahmaniar, H. I. K. Fathurrahman, A. Z. K. Frisky, and Q. M. ul Haq, "Understanding of convolutional neural network (CNN): a review," *International Journal of Robotics and Control Systems*, vol. 2, no. 4, pp. 739–748, 2022, doi: 10.31763/ijrcs.v2i4.888.
[9] C. K. Dewa, A. L. Fadhilah, and Afiahayati, "Convolutional neural networks for handwritten javanese character recognition," *IJCCS (Indonesian Journal of Computing and Cybernetics Systems)*, vol. 12, no. 1, pp. 83–94, 2018, doi: 10.22146/ijccs.31144.
[10] T. Li, "Research on convolutional neural network in the field of oil and gas exploration," *Open Access Library Journal*, vol. 10, no, 03, 2023, doi: 10.4236/oalib.1109738.
[11] T. D. Pham, D. T. Nguyen, J. K. Kang, and K. R. Park, "Deep learning-based multinational banknote fitness classification with a combination of visible-light reflection and infrared-light transmission images," *Symmetry*, vol. 10, no. 10, 2018, doi: 10.3390/sym10100431.
[12] B. K. Triwijoyo, A. Adil, and A. Anggrawan, "Convolutional neural network with batch normalization for classification of emotional expressions based on facial images," *MATRIK : Jurnal Manajemen, Teknik Informatika dan Rekayasa Komputer*, vol. 21, no. 1, pp. 197–204, 2021, doi: 10.30812/matrik.v21i1.1526.
[13] N. F. Nissa, A. Janiati, N. Cahya, and P. Astuti, "Application of deep learning using convolutional neural network (CNN) method for women's skin classification," *Scientific Journal of Informatics*, vol. 8, no. 1, pp. 144–153, 2021, doi: 10.15294/sji.v8i1.26888.
[14] N. Sabri, H. N. A. Hamed, Z. Ibrahim, and K. Ibrahim, "A comparison between average and max-pooling in convolutional neural network for scoliosis classification," *International Journal of Advanced Trends in Computer Science and Engineering*, vol. 9, no. 1.4, pp. 689–696, 2020, doi: 10.30534/ijatcse/2020/9791.42020.
[15] S. Qiu, "Global weighted average pooling bridges pixel-level localization and image-level classification," *arXiv*, 2018, doi: 10.48550/arXiv.1809.08264.
[16] J. Huang *et al.*, "Speed/Accuracy Trade-offs for modern convolutional object detectors," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 3296–3305. doi: 10.1109/CVPR.2017.351.
[17] A. G. Howard *et al.*, "MobileNets: efficient convolutional neural networks for mobile vision applications," *arXiv*, 2017, doi: 10.48550/arXiv.1704.04861.
[18] A. M. Husein, Christopher, A. Gracia, R. Brandlee, and M. H. Hasibuan, "Deep neural networks approach for monitoring vehicles on the highway," *SinkrOn: Jurnal dan Penelitian Teknik Informatika*, vol. 4, no. 2, pp. 163–171, 2020, doi: 10.33395/sinkron.v4i2.10553.
[19] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. C. Chen, "MobileNetV2: inverted residuals and linear bottlenecks," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4510–4520. doi: 10.1109/CVPR.2018.00474.
[20] S. Mandt, M. D. Hoffman, and D. M. Blei, "Stochastic gradient descent as approximate bayesian inference stephan," *Journal of Machine Learning Research*, vol. 18, 2017, doi: 10.48550/arXiv.1704.04289.
[21] Y. Tian, Y. Zhang, and H. Zhang, "Recent advances in stochastic gradient descent in deep learning," *Mathematics*, vol. 11, no. 3, 2023, doi: 10.3390/math11030682.
[22] Y. Xiong, L.-C. Lan, X. Chen, R. Wang, and C.-J. Hsieh, "Learning to schedule learning rate with graph neural networks," in *International Conference on Learning Representations (ICLR)*, 2022.
[23] J. Terven, D. M. Cordova-Esparza, A. Ramirez-Pedraza, and E. A. Chavez-Urbiola, "Loss functions and metrics in deep learning," *arXiv*, 2023, doi: https://doi.org/10.48550/arXiv.2307.02694.
[24] F. Gorunescu, *Data Mining Concepts, Models and Techniques*. New York: Springer-Verlag Berlin Heidelberg, 2011.
[25] P. Theerthagiri, I. J. Jacob, A. U. Ruby, and Y. Vamsidhar, "Prediction of COVID-19 possibilities using KNN classification algorithm," *International Journal of Current Research and Review*, vol. 13, no. 6, pp. 156–164, 2021, doi: 10.31782/IJCRR.2021.SP173.

BIOGRAPHIES OF AUTHORS

Denny Indrajaya, S.Si. is a student in the Master of Data Science at Satya Wacana Christian University (SWCU). He has published papers related to artificial intelligence and modeling. He is also the first winner in Indonesia's Nasional Data Science Tournament 2022. During his time as a Master of Data Science student, he has been working at the PT Waleta Asia Jaya as an artificial intelligence developer in that industry. He is also a programmer who has created websites and applications for smartphones and computers that have artificial intelligence features. He can be contacted at email: 632022002@student.uksw.edu.

Hanna Arini Parhusip holds a Doctor of Mathematics degree from ITB, 2005. She received her B.Sc. of Mathematics from Gadjah Mada University in Indonesia and her M.Sc. (Industrial Mathematics) from Kaiserslautern University, Germany in 1992 and 1997, respectively. In her master program, she was DAAD scholar, and the sandwich program, was supported by DAAD, ICTP, and the Indonesian government for her doctor program. She is currently an associate professor in the Master of Data Science Department at Satya Wacana Christian University in Indonesia. She has been one of the founders in the Master program in Data Science at this university since 2020 and has been developing the application of artificial intelligence for mining and the internet of things for several purposes in some surrounding companies as a collaboration between the university and the industries. Her research includes machine learning, artificial intelligence, internet of things, and the application of ordinary differential equations. She has published over 10 papers in international journals and conferences in the last 5 years. She has been the head of the Master program in Data Science. She can be contacted at email: hanna.parhusip@uksw.edu.

Suryasatriya Trihandaru holds a Doctor of Mathematics degree from ITB, 2005. He received her B.Sc. in Physics from Gadjah Mada University in Indonesia and M.Sc. (Industrial Mathematics) from Kaiserslautern University, in 1992 and 1998, respectively. In his master program he was a DAAD scholar, and sandwich program supported by DAAD, ICTP, and the Indonesian government for his doctoral program. He is currently an associate professor in the Master of Data Science Department at Satya Wacana Christian University in Indonesia. He is one of the founders of Master program in Data Science in this university since 2020 and has been developing the application of artificial intelligence for mining and the internet of things for several purposes in some surrounding companies as a collaboration between the university and the industries. His research includes machine learning, artificial intelligence, internet of things, and medical physics. He has published over 10 papers in international journals and conferences in the last 5 years. He can be contacted at email: suryasatriya@uksw.edu.

Djoko Hartanto, S.E., M.Des. is the CEO of PT Waleta Asia Jaya,at Salatiga city, Indonesia. In addition, he has also lectured to students at several universities in Indonesia. One of his topics of interest is the application of Artificial Intelligence in the real world, especially in the food industry. He can be contacted at email: djoko.hartanto019@gmail.com