International Journal on Advanced Science Engineering Information Technology

Vol.10 (2020) No. 6
ISSN: 2088-5334

*MobileNet*s: Efficient Convolutional Neural Network for Identification of Protected Birds

Yulius Harjoseputro[a,1], Ign. Pramana Yuda[b], Kefin Pudi Danukusumo[a,2]

[a] *Department of Informatics, Universitas Atma Jaya Yogyakarta*
E-mail: [1]yulius.harjoseputro@uajy.ac.id; [2]kefinpudi@gmail.com

[b]*Department of Biology, Universitas Atma Jaya Yogyakarta*
E-mail: pramana.yuda@uajy.ac.id

***Abstract*—Wildlife trade is one of the main factors causing endangered bird species. In Indonesia, trade has caused 28 bird species to be classified in the endangered bird category. Protection efforts have been made with the establishment of 564 species of Indonesian birds as protected birds. For law enforcement, certainty is needed in the identification of these bird species. This study begins with a Forum of Discussion Groups from relevant institutions in Java and Bali to determine the types of protected birds that are prioritized to developed in this application. Based on the results of the Forum of Discussion Group, a bird photo dataset compiled using 17 categories or types of bird photos as prioritized in this study. The method used in this study is the Convolutional Neural Network (CNN) method, which combined the structure of *MobileNet* and the weight of the network that has previously trained using *ImageNet.* The results of this study are the differences of results between CNN standards and those combined with the structure of *MobileNet.* For better accuracy, using the CNN standard, which is around 98.38% for the accuracy of the training, while in terms of size, combined with *MobileNet* has a relatively smaller model size, which is 68 megabytes.**

Keywords*— wildlife; endangered; forum of discussion group; dataset; convolutional neural network; *MobileNet; ImageNet.

I. INTRODUCTION

Wildlife trade is a large business with a trade value of $ 8-10 billion per year [1]. If carried out with the principles of sustainability, trade is one form of wildlife conservation. However, in reality, 80% of the traded animals are captured in nature, which does not guarantee its sustainability [2]. One day several species of birds were rarely found [3]. The wildest animals captured and traded were birds [4]. This condition, especially in Indonesia, is inseparable from the culture of the community. Birds have long cultural roots in Indonesian society [5]. To fulfill a high demand, then endemic and protected species of birds are often smuggled between islands and between countries. Based on the latest assessment from the International Union for Conservation of Nature's (IUCN), trade has caused twenty-eight Indonesian bird species to be classified as endangered birds [6].

The first step in protecting wildlife is determining species protected by legal protection. This effort has existed since the days of the Dutch East Indies government. The last legal umbrella is Government Regulation (PP) Number 7 of 1999 concerning Preservation of Plant and Animal Species and PP Number 8 of 1999 concerning Utilization of Plant and Wildlife Species. The appendix to the PP has been updated with the Indonesian Minister of Environment and Forestry Regulation No. P.20 / MENLHK / SETJEN / KUM.1 /6/2018 concerning species of plants and animals that are protected. Based on this regulation, 794 species of animals are protected in Indonesia; 564 of them are birds.

One problem that arises in implementing this regulation is one of the truths of certainty in the identification of species of animals found in the market or field. The ability of staff at the Natural Resources Conservation Center (KSDA), a government agency with authority in this field, in identifying species of animals is still limited. Therefore, we need tools to overcome this problem. Several mobile applications for identifying animal species are already available, for example, iNaturalist, eBird, Butterfly Collection, and Burungnesia. This application contains photographs of reference animals, which form the basis for identifying objects of observation or research. However, the application is still limited to species in certain areas or defined groups of animals, such as birds. Besides that, it still needs a relatively long time because identifying the characteristics still needs to be done. Applications based on photos of objects, such as CamFind, have not been able to identify the types of protected animals in Indonesia accurately. The classification of bird species has increasingly received attention in computer vision as a

promising application in the study of biology and the environment [7].

The purpose of this study is to apply the Convolutional Neural Network (CNN) method to develop bird recognition applications for the identification of protected animals in Indonesia. This application assisted KSDA staff in law enforcement efforts of protected bird species by giving high certainty about bird species confiscated from the trade or private collections.

II. Materials and Method

Before discussing the methods used in this research, it is better to know any theory or material about the research topic. Several types of research have been concluded in this paper, and some of them have high relevance to this paper. The high demand for wild animals for pets has created legal and illegal trade. Also, there is a high demand for certain animal parts or organs for ornament decoration (for example, elephant ivory, rhino horn, feather bird bones, snakeskin) for traditional medicinal ingredients (for example, rhino horn, crocodile stalk) or food sources of protein. Globally the value of this animal trade is very high, reaching $ 8-10 billion per year [1]. The value of wildlife trade is equivalent to the value of trade in weapons, drugs, and humans [8].

If carried out with the principles of sustainability, wildlife trade is one form of conservation. However, unfortunately, the majority (80%) of traded animals come from capture in nature that does not guarantee its sustainability [2]. As a result of this trade, many vertebrate animals (mainly birds, mammals, and fish) in Southeast Asia are endangered [9]. At present, 564 species of birds in Indonesia have protected. This status would not prevent buying and selling these types. Fifty species of birds that have protected are still frequently traded, openly in bird markets or clandestinely, and through online systems. The types of birds that frequently traded include Brontok Eagle, Javanese Eagle, Green Peacock, Black-headed Kasturi, Raja Cockatoo, Jambulkuning Cockatoo, Golden Julang, Rowo Cucak, White Starling, Bali Starling, and Nias gold Tiong. Birds are indeed the most traded type of vertebrate [4].

In Indonesia, birds are popular animals and have long cultural roots [5]. Endemic and protected bird species are often smuggled between islands and between countries. As a result, 28 species of birds in Indonesia are included in the category of endangered birds (IUCN Red List). Trade, together with damage and literature review of no more than 1000 words by expressing the state of the art and road map in the field under study. Charts and road maps are made in the form of JPG / PNG, which is then inserted in this field. Primary sources of literature/references that are relevant and prioritizing the results of research in the latest scientific journals or patents. It recommended using library resources in the last ten years. Loss of habitat has become a significant threat factor for bird conservation [6].

The loss of birds from their habitat will have a negative impact. Birds have essential ecological functions. Some plants depend on birds to spread their seeds. Besides, insectivorous birds act as controllers of insect populations in natural habitats or cultivation or agricultural areas [10].

Some web-based applications or mobile applications to identify an object have developed. In agriculture, several identification applications are available to identify weeds, pest insect identification, identification, disease, and nutritional conditions [11]. For the identification of animals, there are already several applications, such as the iNaturalist application, eBird, and Butterfly Collection. This application contains photographs of reference animals, which form the basis for identifying objects of observation or research. However, this application still needs researchers to identify the characteristics of animals found and match them with references in the application.

A mobile application for the identification of species of birds in Indonesia, namely Burungnesia, is available. However, it has not been able to identify from photos taken with a smartphone directly. Applications that are based on photos of objects, such as CamFind, able to identify the type of object that is photographed with a smartphone. However, this application has not been able to identify the types of birds protected in Indonesia precisely.

One of the fields in computer science that is overgrowing is Computer Vision. Evidenced by the existence of several annual competitions around Computer Vision, including Large Scale Visual Recognition Challenge (ILSVRC) [12], PASCAL Visual Object Classes (VOC)[13], and Microsoft Common Objects in Context (MSCOCO) [14].

With this competition, each year, state of the art methods emerges in the fields of image classification, object detection, and image segmentation. In recent years, the Deep Learning method has succeeded in increasing performance in these competitions, far above the previous state of the art method. The beginning of the success of Deep Learning began with AlexNet, an image classification method based on Convolutional Neural Network (CNN), which won the ILSVRC 2012 with an error rate of 15.3% compared to the second position of 26.2% [15].

The use of Deep Learning techniques with the Convolutional Neural Network method was first successfully applied by Yann LeCun in 1998 [16]. In this study, Yann LeCun proposed the CNN method to recognize handwriting to read documents. The results obtained from these studies indicate that accuracy is high enough to reach a test error of only 1.7%.

For the application of the Convolutional Neural Network method, it can develop in terms of architecture and the many uses of layers on the network. The use of correct architecture will be perfect for the classification of images in various categories, for example, such as the *ImageNet* dataset, which has as many as 1000 categories. In 2012 the Deep Learning technique with the CNN method was popularized with the AlexNet inspector, who tested with the *ImageNet* dataset [15]. The architecture created by Alex Krizhevsky showed very significant results in the testing set with a test error of 17%. These results are considered to be very extraordinary because the images in the dataset used are very complex and many. AlexNet is faster than KCR GoogLeNet when they are trained using PHD08 [17].

There is research on the classification of birds using deep learning methods. Among them are studies analyzing experimental category of birds in the dataset (i.e., Caltech UCSD Birds 200 [CUB-200-2011]) [3]. Results from these studies indicate that the accuracy in identifying birds is between 80% and 90%. This experimental study was carried

out with the Ubuntu 16.04 operating system using the Tensor flow library [3].

The research discusses the classification of bird species in 2018. In that study, they are using PreTrained Mask-RCNN and ensemble models consisting of Inception Nets (Inceptionv3 net & InceptionResnetv2) to obtain localization and species of birds from the image. The final model in the study achieved an F1 score of 0.5567 or 55.67% in the dataset provided at the CVIP 2018 Challenge [7]. Besides, research related to birds has also been carried out, especially regarding bird song. In that study, the MAP score reached more than 40% for the main species and 33% for the main species with species background [18].

In this research, the CNN model used does not require multiple layers to save learning time. The simplicity of the CNN architecture used is sufficient for image classification with a relatively small dataset. Mainly the categories for classification are only a few. As little as the amount of data to be studied can produce unsatisfactory accuracy, for that, several things must do in order to get reasonably good results. This technique is effortless to implement on the CNN model and will have an impact on the model's performance in training and reducing overfitting [19]. During various experiments, the CNN parameters are configured on several events to analyze their performance in different environments [20].

The test results in this study were compared based on the number of the epoch that is used, and after that, the optimal number of the epoch took

in terms of the speed and accuracy of the model in studying data related to the morphology of the bird. In this research, an adequate model is needed that can meet the needs of the application to make a classification of the dataset to use. The model used in this study includes the CNN architecture, which was supported by the structure of *MobileNet* and the network weights that have previously trained using *ImageNet*.

A. CNN (Convolutional Neural Network) Architecture

CNN is an artificial neural network that is devoted to receiving input consisting of 2-dimensional images. CNN is included in the type of Deep Neural Network because of the high network depth and widely applied to image data [21]. The processes contained in this CNN method include the training process. In the training process, there are three stages, namely the Convolutional layer, the Pooling layer, and the Fully connected layer [22]. Existing processes on this network include convolution operations or convolution layers, where the input image on the network was filtered every channel with a kernel whose size is determined from the start. This filtering process produced a feature map. In addition to this operation, the architecture that was used in this study was the Pooling and ReLu (Rectified Linear Unit) layers for each hidden layer.

The standard convolution process in the model created in this study uses a filter whose size is 3x3 and is used only for the first layer. In general, convolution operations can be written with the formula number 1 as follows.

$$S(i,j) = (K * I)(i,j) = \sum_m \sum_n I(i-m, j-n) K(m,n) \quad (1)$$

In formula number 1 is a multiplication formula between the image matrix and filter. Where variable i and j are the pixel dimensions of the image, K is the basis of the kernel, and i is the input while m, n is the dimension of the kernel. After the first layer, we used the Max Pooling layer, as shown in Fig 1.

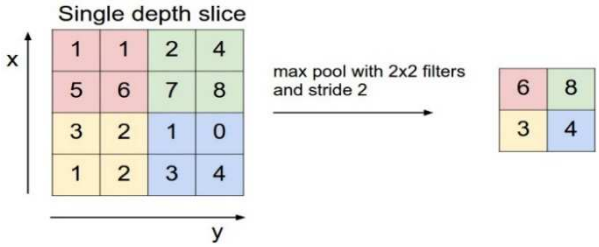

Fig. 1 Max Pooling Example [21]

A good CNN architecture always has a broad and deep layer stack. The representation of network depth is beneficial for the level of classification accuracy. Models that made with deep CNN networks also have an excellent performance on a variety of datasets [23]. Therefore usually, when using this CNN method, it is very much recommended to use GPU to speed up the training process [24].

B. MobileNet.

What needs to know in the CNN standard convolution process is how the operations carried out to require very high computational costs so that it produced a model with a huge size. This model becomes ineffective when it is used as a web service. *MobileNet* is one of the states of the art architectures that can significantly reduce the size of the model and only affects minimal accuracy.

The basis of the *MobileNet* Architecture is two types of convolution layers that are different from standard convolution processes. These layers are depthwise separable convolution and pointwise convolution layers. The depthwise layer uses a single filter for each input channel, different from the standard convolution that uses inputs with channel dimensions for one filter. For the pointwise layer, then perform a 1x1 filter multiplication operation to combine the feature map results from the depthwise layer [25].

C. Classification

The model in this study was built specifically to detect bird images that have 17 output classes. This classification can also dub as Fine-grained Object Classification, which aims to identify categories that are visually and semantically very similar in general categories, for example, bird species. Even with networks such as CNN, this classification is a challenging task because the object of the image to be detected has very little visual difference and is even painful for humans to recognize [26].

Our approach to this research is to use a network that has previously trained. The use of *MobileNet* with trained weights provides a substantial increase in accuracy because network weights have optimized using *ImageNet*. That also means that the trained network can detect color differences, shapes, and edges optimally. An outline of the model architecture in this study is shown in Fig. 2 as follows.

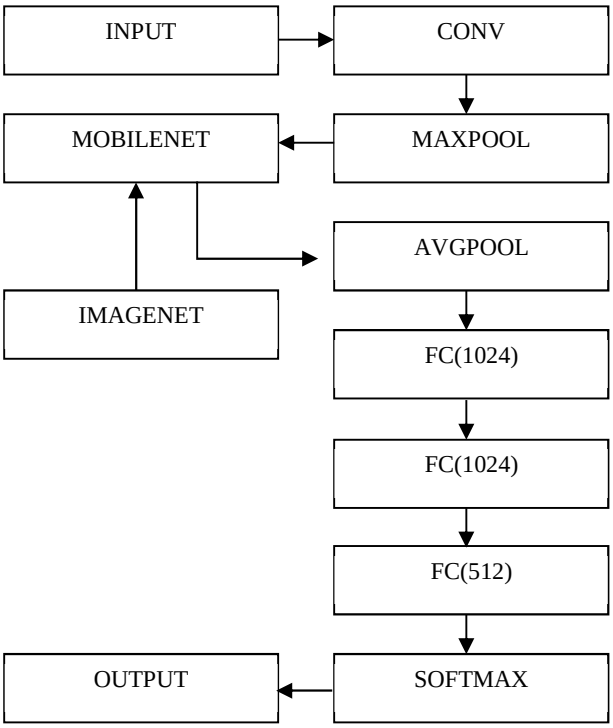

Fig. 2 Architectural Research Model

The architectural model developed in this study can see in Fig 2, which includes input with an image size of 150x150 and then continued with the usual convolution layer and the MaxPooling layer. After that, this model uses the *MobileNet* architecture, which already has training weights from *ImageNet*. The results of the operation of all these layers are then added to the AvgPooling layer. Then the results of this pooling layer were finalized with the Fully connected layer with the number of neurons 1024, 1024, and 512. ReLU activates each of these Fully Connected layers. After that, we use Softmax, which is a function to take vector input and distribute the output probability.

1) Input: In CNN architecture, input includes a size for the next layer shape. In this study, we used input with a 150x150 pixel shape. It means that all image inputs from the dataset are reshaped to match the next input layer. The CNN architectural code can see in Fig 3.

```
batch_size = 32

train_generator = train_datagen.flow_from_directory(
    'dataset/train',
    target_size=(150, 150),
    color_mode='rgb',
    batch_size=batch_size,
    class_mode='categorical',)
```

Fig. 3 CNN Architectural Code

The CNN architecture code that we created uses the python programming language with the TensorFlow backend and KERAS framework. Modeling from layer to layer is made easy by the structure provided by Keras. For input to the model, use the flow_from_directory function with dataset location parameters, target size with 150x150 pixel conditions, input color mode that is RGB, batch size with 32 contents, and categorical class mode. This function would be a condition or setting that is used as the basis of input before being added to the next layer.

2) Conv (Convolution Layer): This layer is the first *convolution* layer that uses a standard convolution method with 32 filters and kernel size 3x3. Input on the previous layer filtered with 32 kernels that produced output features that would be used for the next layer. The convolution process in this study can see in Fig 4.

```
model_bird = Sequential()

model_bird.add(Conv2D(32, (3, 3), input_shape=(3, 150, 150), padding='same'))
```

Fig. 4 Convolution Process

Using the Sequential () function of hard, we have an object to hold the layer stack. Then this object has the Add () function, which contains the layer to be used. Conv2D means we add a 2-dimensional convolution layer with an input form of 3x150x150 pixels.

3) MaxPool: The next layer is MaxPool, shown in Fig. 5, which is useful for handling overfitting and reducing the computing burden. MaxPool works by using a 2x2 kernel that filters the output from the previous layer and takes the maximum value from the filtered features section.

```
model_bird.add(MaxPooling2D(pool_size=(2, 2)))
```

Fig. 5 MaxPool Code

Just like adding a convolutional layer, this time using the add () function again to add a Max pooling layer with a 2x2 kernel size.

4) MobileNet + *ImageNet: MobileNet* here represents many layers that have been prepared previously. The arrangement of layers in the *MobileNet* architecture resembles the VGG architecture. The composition of this layer is then assisted by the weight of the *ImageNet* that has previously been trained. Using of *MobileNet* and *ImageNet* can be seen in Fig 6.

```
base_model = model_bird.output

base_model = MobileNet(weights='imagenet',
                       include_top=False)(base_model)
```

Fig. 6 Code for using *MobileNet* and *ImageNet*

The model is formed as output for the base_model object. Then this object is formed by using layer arrangement from *MobileNet* with the function provided by Keras. This function has a weight parameter with the contents of *ImageNet*, which means to take the weight of the *ImageNet* that has trained. Then there is also the include_top = false parameter, which means not to take the final layer layout of the existing *MobileNet* structure.

5) AVGPool (AveragePool Layer): This layer shown in Fig 7 indicates the output from the *MobileNet* process and then be processed by the regular pooling operation by calculating the average output. Operations at this layer can reduce data significantly and prepare the model for the last few layers for classification.

```
x = base_model.output
x = GlobalAveragePooling2D()(x)
```

Fig. 7 Code for using AVGPool

The base_model object that has formed then taken its output to accommodate the x variable, and then this variable used to enter the GlobalAveragePooling2D function, which implements the average collection layer with a 2x2 kernel.

6) FC(1024,1024,512): Fully Connected Layer with 1024 neurons shown in Fig 8. At the fully connected layer, the output model results formed into a one-dimensional vector, and then each weight value in the model is processed with neurons on the layer as many as 1024. These neurons were linked again to the next layer with the number of neurons 1024 then 512. The number of neurons here usually has an impact on the training time but can increase the level of accuracy. However, the more neurons also, the accuracy not necessarily increase, so it needs optimal values as parameters in this layer.

```
x = Dense(1024, activation='relu')(x)
x = Dense(1024, activation='relu')(x)
x = Dense(512, activation='relu')(x)
```

Fig. 8 Fully Connected Layer with 1024 neuron

The results of the previous average pooling layer are accommodated again into the x variable, which is the input for three fully connected layer layers, or where it can be called a dense layer with 1024 size neurons for the first two layers, and the last layer is 512. Each layer uses Activation Relu or Rectified Linear Unit to normalize the weight value on the network.

7) Softmax: Softmax is a function that normalizes the input values into vectors that produce a probability distribution, which, if added together, all have a value of 1. On this layer, all outputs are connected to 17 neurons according to the class output in this dataset—the results of 17 outputs calculated by softmax operation, which produce a probability value. Using of Softmax can be seen in Fig 9.

```
prediction = Dense(17, activation='softmax')(x)
```

Fig. 9 Using of Softmax

As the last layer, the prediction variable contains a fully connected layer with a total of 17 neurons, according to the number of output classes to predict. The activation used here is softmax, which is used for output normalization.

8) Output: All output generated by the model is compiled and optimized using Adam's optimizer. Each of the probabilities of the loss function calculated, which then the value of this loss were corrected with an optimizer so that the results of the training better.

```
model_output = Model(inputs=base_model.input, outputs=prediction)
model_output.compile(optimizer='Adam', loss='categorical_crossentropy', metrics=['accuracy'])
```

Fig. 10 Output generated by the model

The output shown in Fig 10, uses the Model function to hold all output layers with the output parameters. The input parameters contain the input layer of the previous base_model object. This model is then compiled with Adam's optimizer, and the loss functions categorical cross-cropping with metrics based on accuracy.

III. RESULT AND DISCUSSION

In this study, the training process carried out by using the Keras library with the TensorFlow backend, and in Table I, is the result of a summary model based on the operation of the library:

TABLE I
MODEL SUMMARY

No.	Layer Type	Output Shape	Param
1	InputLayer	(3, 3, 150, 150)	0
2	Conv2D	(3, 32, 150, 150)	864
3	MaxPooling	(3, 32, 150, 150)	128
4	ReLU	(3, 32, 150, 150)	0
5	DepthwiseConv2D	(3, 32, 150, 150)	288
6	BatchNormalization	(3, 32, 150, 150)	128
7	ReLU	(3, 32, 150, 150)	0
8	Conv2D	(3, 64, 150, 150)	2048
9	BatchNormalization	(3, 64, 150, 150)	25
10	ReLU	(3, 64, 150, 150)	0
11	DepthwiseConv2D	(3, 64, 150, 150)	576
12	BatchNormalization	(3, 64, 150, 150)	256
13	ReLU	(3, 64, 150, 150)	0
14	Conv2D	(3, 128, 150, 150)	8192
15	BatchNormalization	(3, 128, 150, 150)	512
16	ReLU	(3, 128, 150, 150)	0
17	DepthwiseConv2D	(3, 128, 150, 150)	1152
18	BatchNormalization	(3, 128, 150, 150)	512
19	ReLU	(3, 128, 150, 150)	0
20	Conv2D	(3, 128, 150, 150)	16384
21	BatchNormalization	(3, 128, 150, 150)	512
22	ReLU	(3, 128, 150, 150)	0
23	ZeroPadding2D	(3, 128, 150, 150)	0
24	DepthwiseConv2D	(3, 128, 150, 150)	1152
25	BatchNormalization	(3, 128, 150, 150)	512
26	ReLU	(3, 128, 150, 150)	0
27	Conv2D	(3, 256, 150, 150)	32768
28	BatchNormalization	(3, 256, 150, 150)	1024
29	ReLU	(3, 256, 150, 150)	0
30	DepthwiseConv2D	(3, 256, 150, 150)	2304
31	BatchNormalization	(3, 256, 150, 150)	1024
32	ReLU	(3, 256, 150, 150)	0
33	Conv2D	(3, 256, 150, 150)	65536

34	BatchNormalization	(3, 256, 150, 150)	1024
35	ReLU	(3, 256, 150, 150)	0
36	ZeroPadding2D	(3, 256, 150, 150)	0
37	DepthwiseConv2D	(3, 256, 150, 150)	2304
38	BatchNormalization	(3, 256, 150, 150)	1024
39	ReLU	(3, 256, 150, 150)	0
40	Conv2D	(3, 512, 150, 150)	131072
41	BatchNormalization	(3, 512, 150, 150)	2048
42	ReLU	(3, 512, 150, 150)	0
43	DepthwiseConv2D	(3, 512, 150, 150)	4608
44	BatchNormalization	(3, 512, 150, 150)	2048
45	ReLU	(3, 512, 150, 150)	0
46	Conv2D	(3, 512, 150, 150)	262144
47	BatchNormalization	(3, 512, 150, 150)	2048
48	ReLU	(3, 512, 150, 150)	0
49	DepthwiseConv2D	(3, 512, 150, 150)	4608
50	BatchNormalization	(3, 512, 150, 150)	2048
51	ReLU	(3, 512, 150, 150)	0
52	Conv2D	(3, 512, 150, 150)	262144
53	BatchNormalization	(3, 512, 150, 150)	2048
54	ReLU	(3, 512, 150, 150)	0
55	DepthwiseConv2D	(3, 512, 150, 150)	4608
56	BatchNormalization	(3, 512, 150, 150)	2048
57	ReLU	(3, 512, 150, 150)	0
58	Conv2D	(3, 512, 150, 150)	262144
59	BatchNormalization	(3, 512, 150, 150)	2048
60	ReLU	(3, 512, 150, 150)	0
61	DepthwiseConv2D	(3, 512, 150, 150)	262144
62	BatchNormalization	(3, 512, 150, 150)	2048
63	ReLU	(3, 512, 150, 150)	0
64	Conv2D	(3, 512, 150, 150)	262144
65	BatchNormalization	(3, 512, 150, 150)	2048
66	ReLU	(3, 512, 150, 150)	0
67	DepthwiseConv2D	(3, 512, 150, 150)	4608
68	BatchNormalization	(3, 512, 150, 150)	2048
69	ReLU	(3, 512, 150, 150)	0
70	Conv2D	(3, 512, 150, 150)	262144
71	BatchNormalization	(3, 512, 150, 150)	2048
72	ReLU	(3, 512, 150, 150)	0
73	ZeroPadding2D	(3, 512, 150, 150)	0
74	DepthwiseConv2D	(3, 512, 150, 150)	4608
75	BatchNormalization	(3, 512, 150, 150)	2048
76	ReLU	(3, 512, 150, 150)	0
77	Conv2D	(3, 1024, 150, 150)	524288
78	BatchNormalization	(3, 1024, 150, 150)	4096
79	ReLU	(3, 1024, 150, 150)	0
80	DepthwiseConv2D	(3, 1024, 150, 150)	9216
81	BatchNormalization	(3, 1024, 150, 150)	4096
82	ReLU	(3, 1024, 150, 150)	0
83	Conv2D	(3, 1024, 150, 150)	1048576
84	BatchNormalization	(3, 1024, 150, 150)	4096
85	ReLU	(3, 1024, 150, 150)	0
86	AveragePooling	(None, 1024)	0
87	Dense	(None, 1024)	1049600
88	Dense	(None, 1024)	1049600
89	Dense	(None, 512)	524800
90	Dense	(none, 17)	8721
		Total Parameter	6118890

From the summary in Table I, can be seen that the first layer is an input layer image with a 150x150 pixel shape using three filters with three channels (RGB) so that the shape is (3,3,150,150) or can also describe as (channel, number of filters, size, size). Then in the second layer, there is a standard convolution layer with as many as 32 filters, and from here, the parameters at this layer are 864. Things to know that the more complex the operation is used and the more filters used, the more parameters are also used. After that, there is a max-pooling layer to filter the search for the most significant number of previous inputs and then use a ReLU (Rectified Linear Unit) on the hidden layer to activate the layer to eliminate negative nodes. This ReLU activation facilitates the network for training. Then from layer 5-85 is part of the *MobileNet* architecture that uses the concept of a network with multiple and deep layers as in the VGG model. Models with many layers undoubtedly affect the number of parameters so that they have the potential for more comfortable and more accurate training. In this model, there is also a layer named Depthwise Convolution, and as previously explained where this layer is a type of convolution layer on *MobileNet* that replaces standard convolution operations. It can see from this layer does not require significant computing and efficient in the number of parameters required.

After the depthwise layer, there is also a layer called Batch Normalization, which functions to normalize the input layer so that it is faster to train. After the layer from the MobileNet model, it is connected again with the Average Pooling layer, which takes the average value from the previous input, then reconnected to a one-dimensional vector with neurons totaling 1024, 1024 and 512 on the Dense layer or fully connected layer. After that, the last layer is the output layer by taking the probability of 17 classes. The total parameters generated from this model also amounted to 6118890 and are still quite efficient and accurate, given the small number of existing datasets.

Besides, in this study, we compared the model we made with the standard CNN model with 18 layers consisting of convolution, pooling, and ReLU layers and referred to the accuracy for each epoch. The results of the accuracy of the training dataset for each epoch can see in Table II and Table III.

TABLE II
RESULTS OF ACCURACY USING THE TRAINING DATASET

Model	Epoch-5	Epoch-10	Epoch-20
Standard CNN	92.33 %	96.77%	98.38%
MobileNet (*ImageNet* Pretrained)	90.54%	95.02%	96,27%

TABLE III
RESULTS OF ACCURACY USING THE VALIDATION DATASET

Model	Epoch-5	Epoch-10	Epoch-20
Standard CNN	58.29 %	67.94%	72.82%
MobileNet (*ImageNet* Pretrained)	55.52%	63.76%	70.59%

The model previously described it was trained using a maximum of 20 epochs and used 812 training datasets for 17 classes. The results of the training produced an accuracy of 96.27%. In this study, using a dataset with 81 images for each class, with 17 classes and the results reached 70.59%. Although there is a slight difference in testing accuracy between the two models, however when compared to its size, the model we use, for its size, only reaches 68 megabytes, far compared to using a standard CNN model that can produce models up to 1.5 gigabytes.

IV. CONCLUSION

The results of this study indicate the possibility of identifying protected birds using the Convolutional Neural Network method. Also, overall, for the results of this study, there are differences between the models produced using the CNN standard and the use of *MobileNet* on the CNN architecture. The CNN standard on the 20th epoch resulted in a training accuracy of 98.38% and a testing accuracy of 72.82%. While on *MobileNet*, the result is slightly different from the CNN standard, namely for the results of the training accuracy of 96.27% and the testing accuracy of 70.59%. From these results, when viewed from the accuracy of training and testing, the CNN standard has better accuracy than using *MobileNet*. Still, when viewed from the resulting model size, the CNN standard has a more substantial model size of 1.5 gigabytes, while the model using *MobileNet* produces a size of 68 megabytes. For future research, a focus on increasing accuracy becomes an essential factor in future research. Besides, the development of the implementation of this research should be directed towards mobile applications so that they can use more flexible.

ACKNOWLEDGMENT

The author would like to show the gratefully acknowledges the support from "Ministry of Research, Technology, and Higher Education of the Republic of Indonesia" and from Universitas Atma Jaya Yogyakarta (UAJY), Indonesia.

REFERENCES

[1] K. Lawson and A. Vines, *Global Impacts of the Illegal Wildlife Trade*, no. February. 2014.

[2] A. Haryanta, D. N. Adhiasto, and N. Hardianto, *Pendataan & Pengenalan Jenis Satwa Liar di Pasar Burung Yang Sering diperdagangkan*. 2013.

[3] P. P. Gavali, M. Prachi, A. Mhetre, M. Neha, and C. Patil, "Bird Species Identification using Deep Learning," *Int. J. Eng. Res. Technol.*, vol. 8, no. 04, pp. 68–72, 2019.

[4] E. R. Bush, S. E. Baker, and D. W. Macdonald, "Global trade in exotic pets 2006-2012," *Conserv. Biol.*, vol. 28, no. 3, pp. 663–676, 2014.

[5] P. Jepson and R. J. Ladle, "Bird-keeping in Indonesia: Conservation impacts and the potential for substitution-based conservation responses," *Oryx*, vol. 39, no. 4, pp. 442–448, 2005.

[6] J. Eaton *et al.*, "Trade-driven extinctions and near-extinctions of avian taxa in Sundaic Indonesia," *Forktail*, vol. 31, no. January 2015, pp. 0–12, 2016.

[7] S. D. Das and A. Kumar, "Bird Species Classification using Transfer Learning with Multistage Training," pp. 1–9, 2018.

[8] L. Wilson-Wilde, "Wildlife crime: A global problem," *Forensic Sci. Med. Pathol.*, vol. 6, no. 3, pp. 221–222, 2010.

[9] V. Nijman, "An overview of international wildlife trade from Southeast Asia," *Biodivers. Conserv.*, vol. 19, no. 4, pp. 1101–1114, 2010.

[10] R. Dirzo, H. Young, M. Galetti, G. Ceballos, J. Nick, and B. Collen, "Defaunation in the antropocene_dirzo2014.pdf," *Science (80-.).*, vol. 345, no. 6195, p. 401, 2014.

[11] I. A. Ciampitti and J. Albers, "Agricultural Mobile Apps: A review and update of ID apps," 2015.

[12] O. Russakovsky *et al.*, "*ImageNet* Large Scale Visual Recognition Challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, 2015.

[13] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (VOC) challenge," *Int. J. Comput. Vis.*, vol. 88, no. 2, pp. 303–338, 2010.

[14] T. Y. Lin *et al.*, "Microsoft COCO: Common objects in context," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 8693 LNCS, no. PART V, pp. 740–755, 2014.

[15] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "*ImageNet* Classification with Deep Convolutional Neural Networks," *Adv. neural Inf. Process. Syst.*, pp. 1097–1105, 2012.

[16] Y. Lecun, L. Bottou, Y. Bengio, and P. Ha, "GradientBased Learning Applied to DocumentRecognition," in *Proceedings of the IEEE*, 1998, no. November, pp. 1–46.

[17] S. G. Lee, Y. Sung, Y. G. Kim, and E. Y. Cha, "Variations of AlexNet and GoogLeNet to improve Korean character recognition performance," *J. Inf. Process. Syst.*, vol. 14, no. 1, pp. 205–217, 2018.

[18] B. P. Tóth and B. Czeba, "Convolutional neural networks for large-scale bird song classification in noisy environment," *CEUR Workshop Proc.*, vol. 1609, pp. 560–568, 2016.

[19] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," *J. Mach. Learn. Res.*, vol. 15, pp. 1929–1958, 2014.

[20] A. Jain and B. K. Sharma, "Analysis of Activation Functions for Convolutional Neural Network based MNIST Handwritten Character Recognition," *Int. J. Adv. Stud. Sci. Res.*, vol. 3, no. 9, pp. 68–74, 2018.

[21] W. S. Eka Putra, A. Y. Wijaya, and R. Soelaiman, "Klasifikasi Citra Menggunakan Convolutional Neural Network (CNN) pada Caltech 101," *J. Tek. ITS*, vol. 5, no. 1, pp. A65–A69, 2016.

[22] A. Santoso and G. Ariyanto, "Implementasi Deep Learning Berbasis Keras Untuk Pengenalan Wajah," *Emit. J. Tek. Elektro*, vol. 18, no. 01, pp. 15–21, 2018.

[23] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," 2014, pp. 1–14.

[24] C. K. Dewa, A. L. Fadhilah, and A. Afiahayati, "Convolutional Neural Networks for Handwritten Javanese Character Recognition," *IJCCS (Indonesian J. Comput. Cybern. Syst.*, vol. 12, no. 1, p. 83, 2018

[25] A. G. Howard *et al.*, "*MobileNet* s: Efficient Convolutional Neural Networks for Mobile Vision Applications," 2017.

[26] B. Zhao, X. Wu, J. Feng, Q. Peng, and S. Yan, "Diversified Visual Attention Networks for Fine-Grained Object Classification," *IEEE Trans. Multimed.*, vol. 19, no. 6, pp. 1245–1256, 2017.