ILKOM Jurnal Ilmiah Vol. 14, No. 3, December 2022, pp. 203-208
Accredited 2nd by RISTEKBRIN No. 200/M/KPT/2020; E-ISSN 2548-7779 | P-ISSN 2087-1716

Classification of dog and cat images using the CNN method

Teguh Adriyanto [a,1,*]; Risky Aswi Ramadhani [a,2];Risa Helilintar [a,3]; Aidina Risktyawan[a,4]
[a]*Universitas Nusantara Pgri Kediri Univ, Kediri and 64111, Indonesia*
teguhae37@gmail.com [1]; riskyaswiramadhani@gmail.com [2] ; risahelilintar@unpkediri.ac.id [3]; ristykdr@gmail.com [4]
** Corrresponding author*

Article history: Received July 15, 2022; Revised July 21, 2022; Accepted November 29, 2022; Available online December 20, 2022

Abstract

Blind people can be defined as those people who are unable to see objects or pictures around them with their eyes. This inability becomes an issue for them when dealing with objects or images in front of them. These problems lead to the novelty of this study that is to recognize objects or images around blind people with the CNN algorithm. Dogs and cats were used as objects in this study. These object recognitions used Deep Learning, a relatively new science in the field of machine learning. Deep learning works like the human brain's ability to recognize an object. In this study, the objects that were used were pictures of a dog and a cat. This study used 3 types of data, namely training, validation, and testing data. The data training consisted of dog data with a total of 1000 images and cat data with a total of 1000 images. Data validation consisted of 500 dog data and 500 cat data. The CCN architecture employed 3 convolution layers. The layer was convolution 1 using 16 filters of kernel size 3x3, the second convolution using 32 filters of kernel size 3x3 and the third using 64 filters of kernel size 3x3. While the data testing consisted of 51dog data and 27 cat data. The method used to analyze the image was CNN. The input was an image with a size of 150x150 pixels with 3 channels, namely R, G, and B. This classification went through a performance test with the Confusion Matrix, and it obtained 45% precision, 45% recall and 45% f1-score. From these results it can be concluded that the accuracy values should be improved.

.

Keywords: Blind; Cat; CNN; Classification; Dog; Image.

Introduction

Blind people are those who are unable to see objects or pictures in their surroundings [1]. Their inability to see becomes an issue for them when dealing with objects or images in front of them. The novelty of this research is based on the issue that is to recognize objects or images that are around the blind people using the CNN algorithm. In this study, only dogs and cats were used as the objects. The object recognition used Deep Learning.

Deep learning is a relatively new science in the field of machine learning. Deep learning works by adopting the ability of the human brain to recognize objects [2][3]. This study used CNN to identify 2 objects [4]-[7]. The objects used in this study were pictures of dogs and cats. The data in the study consisted of 3 parts, namely training, validation, and test datasets [8].

Training dataset is the data used to make sure that the employed data match with the model, the training data used in this process was 1000 dog images, and 1000 cat images. While the Validation dataset is the data used for model evaluation. Data validation consisted of 500 dog images and 500 cat images. The last is test dataset, this data is used to be run in the final model. Simulation data using the model in the real world. Test data should never be used by the previous model. The test data that would be used in this study was 51 dog pictures, and 27 cat pictures.

The method used to process training, validation, and test datasets was CNN[9][10]. For CNN to work properly the input from the system must be determined first. The input from the system was an image with a size of 150 x 150 pixels, with 3 channels, namely R, G, and B. The CCN architecture used 3 layers of convolution. The layer was convolution 1 using a 3x3 kernel, filter 16, the second convolution using a 3x3 kernel, filter 32, and the third using a 3x3 kernel, filter 64.

http://dx.doi.org/10.33096/ilkom.v14i3.1116.203-208

The success analysis of Image Classification with CNN was carried out using the Confusion Matrix. By using the Confusion matrix, accuracy, recall, precision, and F-Measure can be known [11]. This research will be used as the basis for developing assistive technology for blind people.

Method

A. Convolutional Neural Network Models Dog and Cat

Convolutional Neural Network (CNN) is one of the algorithms of Deep Learning. It was the result of the development of the Multi Layer Perceptron (MLP) designed to process data in the form of a grid, one of which was a two-dimensional image dimensions, for example images or sound. CNN was used for labeled data classification using the supervised method learning. Training data and targeted variables were required in the supervised learning, so that it can be concluded that the purpose of this method was to group a data to an existing variable. CNN was often used to identify an object or scene, as well as detecting and segmenting objects. CNN architecture consisted of Featured Learning and Classification as seen in **Figure 1**.

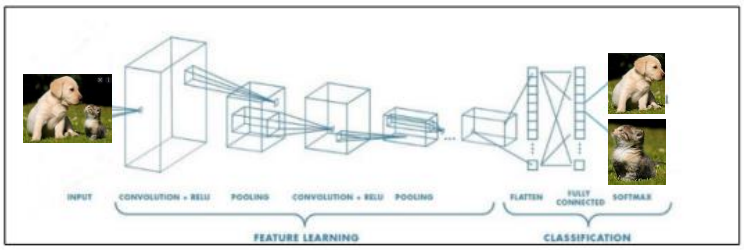

Figure 1. CNN Architecture Dog and Cat

The input was a two-dimensional image. On a computer, the input image was recognized as a numeric data based on the color code of each pixel. Color code was from 0 up to 255. Featured Learning consisted of Convolution Layer, Activation Layer (Relu) and Pooling Layer. The Convolution Layer consisted of an array of neuron form a filter with a certain length and height (pixel). Example pictures with dimensions of 32x32x3 was an image with a width of 32 pixels, 32 pixels high and had 3 pieces. The depth represented the color channel RGB (Red Green Blue).

Stride was a parameter to determine the number of filter shifts. Padding or zero padding was a parameter determining the number of pixels (contains the value 0) which would be added on each side of the input to manipulate the output dimensions of the convolutional layer (feature map). The activation function was at the stage after the convolution process. At this stage, the convoluted value is subjected to an activation function. The polling layer was usually after the convolution layer. Pool layer consisted of a filter of a certain size and stride shifted on the entire feature map area. Feature Learning produced a multidimensional feature map array.

B. Architecture of Convolutional Neural Network Models Dog and Cat

This study used 3 types of data, namely training, validation, and test datasets. The training data consisted of 1000 dog images and data 1000 cat images. Data validation consisted of 500 dog pictures and 500 cat pictures. While the test data consisted of 51 dog pictures and 27 cat pictures.

The CNN model used is shown in **Figure 2**. The input was an image with a size of 150x150 pixels with 3 channels, namely R, G, and B. CNN architecture used 3 layers of convolution.

```
Model: "model"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         [(None, 150, 150, 3)]     0
_________________________________________________________________
conv2d (Conv2D)              (None, 148, 148, 16)      448
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 74, 74, 16)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 72, 72, 32)        4640
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 71, 71, 32)        0
_________________________________________________________________
dropout (Dropout)            (None, 71, 71, 32)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 69, 69, 64)        18496
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 34, 34, 64)        0
_________________________________________________________________
flatten (Flatten)            (None, 73984)             0
_________________________________________________________________
dense (Dense)                (None, 512)               37880320
_________________________________________________________________
dropout_1 (Dropout)          (None, 512)               0
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 513
=================================================================
Total params: 37,904,417
Trainable params: 37,904,417
Non-trainable params: 0
_________________________________________________________________
```

Figure 2. CNN Architecture

A. Convolution Layer

Convolution layer 1 used a 3x3 kernel, 16 filters. The ReLU activation function was used to retrieve the highest nodes and then forward them to the next convolution layer. The image output was 148x148pixel. The pooling layer uses maxpooling with padding 2 and stride 2 which produced an output of 74x74pixel [12][13]. While the resulting parameters were ((3 x 3 x 3) + 1 bias) x 16 filters = 448 parameters.

Then the second convolution process was carried out using a 3x3 kernel, 32 filters. The ReLU activation function was used to take the highest nodes and then forward them to the next convolution layer. The image output was 72x72pixel. Pooling layer using maxpooling with padding 2 and stride 1 produced a 71x71pixel image. While the resulting parameters were ((3 x 3 x 16) + 1 bias) x 32 filters = 4640 parameters.

The dropout process used a value of 0.5. Therefore if the score was less than 0.5, the next convolution process is not continued.

The third convolution process used a 3x3 kernel, 64 filters. The ReLU (rectified linear unit) activation function was used to take the highest nodes and then forward them to the next convolution layer. The image output size was 69x69pixel. Pooling layer using maxpooling with padding 2 and stride 2 produced 34x34pixel output. While the resulting parameters were ((3 x 3 x 32) + 1 bias) x 64 filters = 18496 parameters.

B. Flaten Process

Next, the flattening process was carried out, namely changing the parameter value into a vector value (1-dimensional array) with a size of 34x34pixel 64 channels which produced 34x34x64=73984 nodes.

Dense was used to run the fully-connected layer. The first Dense used the value 512 and the ReLU activation function returns (73984*512)+512=37880320 parameters. The second Dense used a value of 1 and a sigmoid activation function that produced 513 parameters.

C. Compile Process

Furthermore, the process of compiling the model used the loss function binary_crossentropy and Adam's optimization with a learning rate of 0.001. Binary_crossentropy was used because the resulting class was only 2 (binary), namely dogs and cats.

D. Fit Model Process

The next process was the fit model, namely the process of determining the model from the training image data and test data to determine the results of the accuracy values of the training image data and test image data. In

addition to the accuracy value, there would be a loss value. This process produced a model that was formed from the algorithm architecture in CNN that had been prepared. Prior to model fit, training data and validation, an augmentation process were performed using the ImageDataGenerator function. Image augmentation was a technique of applying different transformations to the original image resulting in multiple altered copies of the same image. However, each copy differed from the others in certain aspects depending on the augmentation techniques such as pan, rotate, flip, etc.

Results and Discussion

Fit model using epoch 20, step per epoch 100 and validation step 50 produced 64% accuracy as shown in figure 2.

```
Epoch 10/20
100/100 [==============================] - 48s 477ms/step - loss: 0.6536 - acc: 0.6010 - val_loss: 0.6596 - val_ac
c: 0.6040
Epoch 11/20
100/100 [==============================] - 45s 454ms/step - loss: 0.6527 - acc: 0.6095 - val_loss: 0.6647 - val_ac
c: 0.6610
Epoch 12/20
100/100 [==============================] - 45s 452ms/step - loss: 0.6488 - acc: 0.6065 - val_loss: 0.6525 - val_ac
c: 0.6110
Epoch 13/20
100/100 [==============================] - 45s 454ms/step - loss: 0.6464 - acc: 0.6115 - val_loss: 0.6560 - val_ac
c: 0.6590
Epoch 14/20
100/100 [==============================] - 45s 450ms/step - loss: 0.6562 - acc: 0.5960 - val_loss: 0.6429 - val_ac
c: 0.6140
Epoch 15/20
100/100 [==============================] - 45s 450ms/step - loss: 0.6400 - acc: 0.6145 - val_loss: 0.6420 - val_ac
c: 0.6310
Epoch 16/20
100/100 [==============================] - 45s 451ms/step - loss: 0.6403 - acc: 0.6295 - val_loss: 0.6402 - val_ac
c: 0.6140
Epoch 17/20
100/100 [==============================] - 51s 506ms/step - loss: 0.6315 - acc: 0.6290 - val_loss: 0.6314 - val_ac
c: 0.6500
Epoch 18/20
100/100 [==============================] - 51s 508ms/step - loss: 0.6306 - acc: 0.6280 - val_loss: 0.6274 - val_ac
c: 0.6780
Epoch 19/20
100/100 [==============================] - 51s 509ms/step - loss: 0.6371 - acc: 0.6215 - val_loss: 0.6281 - val_ac
c: 0.6650
Epoch 20/20
100/100 [==============================] - 46s 457ms/step - loss: 0.6217 - acc: 0.6445 - val_loss: 0.6217 - val_ac
c: 0.6440
```

Figure 3. Process of Training Model Diagram

The test results in **Figure 2** were the result data from the classification process with CNN. To determine the success rate of the system, a confusion matrix was needed to determine the level of accuracy, precision, recall, and f-measurement. The following are the test results that have been input into the confusion matrix. **Figure 4** is a measurement measurement chart.

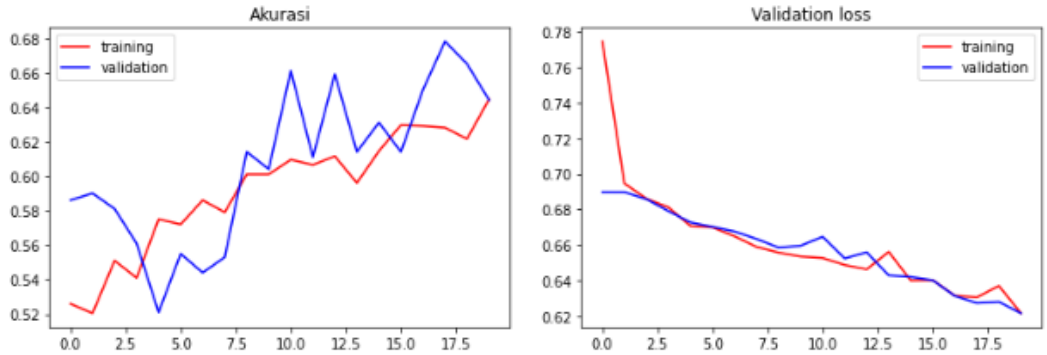

Figure 4. Training Accuracy Chart

Training accuracy diagram in **Figure 5**, it indicates that there is no overfitting [14][15]. Accuracy and validation loss training and validation data show that both have the same trend and overlap.

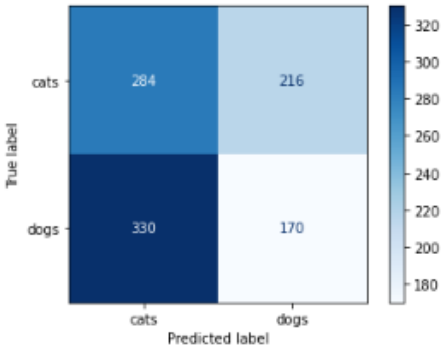

Figure 5. Test Results Using Confusion Matrix

The experimental data in **Figure 6** was then processed, so that the accuracy, precision, recall, and f1-score values were obtained as follows.

	precision	recall	f1-score	support
cats	0.46254	0.56800	0.50987	500
dogs	0.44041	0.34000	0.38375	500
accuracy			0.45400	1000
macro avg	0.45148	0.45400	0.44681	1000
weighted avg	0.45148	0.45400	0.44681	1000

Figure 6. Test Result Using Confusion Matrix

From the results of the test data testing, the results of the analysis with the confusion matrix are as follows: the precision values is 45%, recall value is 45% and f1-score is 45%. The level of precision, recall and f1-score is small, therefore it is necessary to improve the system.

Conclusion

In several previous studies and theories, CNN has performed quite good when it was used to recognize an object. However, the system in the current research should be improved, because CNN has not been able to work optimally. This was proven after the system was analyzed using Confusions Matric with 45% precision, 45% recall and 45% f1-score. The level of accuracy, precision, recall and f1-score is small, therefore improvement is necessary for the system. This research can be used as a basis for developing assistive technology for blind people.

Acknowledgement

We express our gratitude to Universitas Nusantara PGRI Kediri for providing financial support and equipment. With this support the research has been finished.

References

[1] D. Rocha, V. Carvalho, E. Oliveira, J. Goncalves, and F. Azevedo, "MyEyes-automatic combination system of clothing parts to blind people: First insights," *2017 IEEE 5th Int. Conf. Serious Games Appl. Heal. SeGAH 2017*, Jun. 2017, doi: 10.1109/SEGAH.2017.7939298.

[2] K. B. Lee and H. S. Shin , "On certain integrals of Lipschitz-Hankel type involving products of Bessel functions," International Conference on Deep Learning and Machine Learning in Emerging Applications. Turkey, 2019.

[3] W.L Hakim, F. Rezaiz , A.S. Nur and M. Panahi, "Convolutional Neural Network (CNN) with metaheuristic optimization algorithms for landslide susceptibility mapping in Icheon, South Korea," Journal of Environmental Management., vol. 3045, Marc 2022.

[4] M.R. Asegaf and A.T. Wibowo, Klasifikasi spesies tanaman monstera berdasarkan citra daun menggunakan metode Convolutional Neural Network (Cnn), . E-Proceeding of Engineering vol. 8, 2021, pp. 4195–4215.

[5] T. Bariyah, M.A. Rayidi and N. Ngatini," Convolutional Neural Network untuk metode klasifikasi multi-label pada motif batik", Tecno.com, vol. 20, 2021, pp. 155–165

[6] E.N. Arrofiqoh, and Harintaka, "Implementasi metode Convolutional Neural Network untuk klasifikasi tanaman pada citra resolusi tinggi", Geomatika, Vol.24, 2018.

[7] Z.F. Abror "Klasifikasi citra kebakaran dan non kebakaran menggunakan Convolutional Neural Network", Jurnal Ilmiah Teknologi Dan Rekayasa, Vol.24, 2018.

[8] H.N. Falah and K.K Purnamasari," Implementasi Convolutional Neural Network pada pengenalan tulisan tangan", 2019.

[9] K. Jhang and J. Cho," CNN training for face photo based gender and age group prediction with camera", International Conference on Artificial Intelligence in Information and Communication, Japan,2019.

[10] M. Martin, B. Sciolla, M. Sdika, P Quetin and P. Delachartre," Segmentation of neonates cerebral ventricles with 2D CNN in 3D US data: suitable training-set size and data augmentation strategies", 2019.

[11] R.A. Ramadhani, I.K.G.D. Putra, M. Sudarma, and I.A.D. Giriantari "A new technology on translating Indonesian spoken language into Indonesian sign language system", IJECE, Vol.11, No.4, 2021.

[12] M. T. Pavlova, "A comparison of the accuracies of a convolution neural network built on different types of convolution layers," 2021 56th International Scientific Conference on Information, Communication and Energy Systems and Technologies (ICEST), 2021, pp. 81-84, doi: 10.1109/ICEST52640.2021.9483569.

[13] M. Mody, M. Mathew, S. Jagannathan, A. Redfern, J. Jones and T. Lorenzen, "CNN inference: VLSI architecture for convolution layer for 1.2 TOPS," 2017 30th IEEE International System-on-Chip Conference (SOCC), 2017, pp. 158-162, doi: 10.1109/SOCC.2017.8226028.

[14] P. Thanapol, K. Lavangnananda, P. Bouvry, F. Pinel and F. Leprévost, "Reducing Overfitting and Improving Generalization in Training Convolutional Neural Network (CNN) under Limited Sample Sizes in Image Recognition," 2020 - 5th International Conference on Information Technology (InCIT), 2020, pp. 300-305, doi: 10.1109/InCIT50588.2020.9310787.

[15] A. Gavrilov, A. Jordache, M. Vasdani and J. Deng, "Convolutional Neural Networks: estimating relations in the ising model on overfitting," 2018 IEEE 17th International Conference on Cognitive Informatics & Cognitive Computing (ICCI*CC), 2018, pp. 154-158, doi: 10.1109/ICCI-CC.2018.8482067.