

Network and layer experiment using convolutional neural network for content based image retrieval work

Fachruddin¹, Saparudin², Errissya Rasywir³, Yovi Pratama³

¹Doctoral Student Program, Faculty of Engineering, Universitas Sriwijaya, Palembang, Indonesia

²Informatics Department, Telkom University, Bandung, Indonesia

³Department of Informatics Engineering, Faculty of Computer Science, Universitas Dinamika Bangsa, Jambi, Indonesia

Article Info

Article history:

Received Jan 23, 2021

Revised Dec 19, 2021

Accepted Dec 27, 2021

Keywords:

Content based image retrieval

Convolutional neural network

Image

Neural network

System

ABSTRACT

In this study, a test will be conducted to find out how the results of experiments on the network and layer used on the convolutional neural network algorithm. The performance and accuracy of the retrieval process method that was tested using the algorithm approach to do an object image retrieval. The expected results of this study are the techniques offered can provide relatively better results compared to previous studies. The results of the classification of object images with different levels of confusion on the Caltech 101 database resulted an average accuracy value. From the experiments conducted in the study, content based image retrieval work (CBIR) work using convolutional neural network (CNN) algorithm in terms of execution time, loss testing and accuracy testing. From several experiments on layers and networks shows that, the more hidden layers used, then the result is better. The graph of validation loss decreases at fewer epochs, slightly fluctuating at more epochs. Likewise, validation accuracy increases insignificantly on epochs with small amounts, but tends to be stable on more epochs.

This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



Corresponding Author:

Fachruddin

Department of Informatics Engineering, Universitas Dinamika Bangsa

Jendral Sudirman St., Thehok. South Jambi, Jambi City, Indonesia

Email: fachruddin.stikom@gmail.com

1. INTRODUCTION

Increasing the efficiency of digital image search and indexing of images remains a major challenge, various studies have been conducted using various content based image retrieval (CBIR) approaches [1]–[3]. In conducting CBIR, it is necessary to do a preprocessing stage to get better performance. For better image retrieval results, an image enhancement process is needed to improve the quality of inputs for further image processing. Also in CBIR, image capture is based on content or features, but other alternative features still need to be developed to improve accuracy in CBIR [4]. However, it has been stated that image processing using deep learning does not require the extraction of image features [5], [6].

The deep learning method in CBIR enables better learning capability so that performance and precision are higher [7]. Because deep learning is included in the field of machine learning computing and is similar to artificial neural network (ANN)[8], [9]. However, deep learning has a deeper neural network that provides hierarchical representations of data through various convolutions [10]. Deep learning is the latest, modern and robust technique [8], [11]–[13], while the progress and application of deep learning in other domains shows great potential [14]–[17]. The fact, that currently there are at least 40 studies that use deep learning to overcome various agricultural problems with excellent results[18], encourage the writer to prepare this research [19], [20].

Deep learning (DL) is used in core research to conduct testing in object detection work using CNN. The image object dataset (Caltech 101) has 101 categories. The number of images is around 40 to 800 images per category [14]. Most categories have around 50 images, gathered in September 2003 by Fei-Fei Li, Marco Andreetto, and Marc 'Aurelio Ranzato. The size of each image is approximately 300×200 pixels [21], [22]. The use of the Caltech 101 dataset is done to test the deep learning algorithm of CNN. This test is carried out to test the reliability of the CNN to test whether it is proportional to the results of tests conducted previously using compound images. In training and testing on the number of images of the Caltech 101 dataset, the experiments were carried out randomly. When the test results are evaluated the value of the confusion matrix is to determine which images are right or wrong.

The deep learning method in CBIR enables greater learning capability so that performance and precision are higher [4], [15], [23], [24]. Because deep learning is included in the field of machine learning computing and is similar to ANN [24]. However, deep learning has a deeper neural network that provides hierarchical representations of data through various convolutions [20], [25], [26]. Moreover, deep learning is the latest, modern and promising technique with increasing popularity, while the progress and application of deep learning in other domains show great potential [27].

In this study, a test will be conducted to find out how the results of experiments on the network and layer used on the CNN algorithm, the performance, and accuracy. The retrieval process method that was tested using this algorithm approach in the process of object image retrieval. The expected results from this study are the best techniques from the experiments on neural networks (NN) to be applied in CBIR offered can provide relatively better results than previous studies.

2. RESEARCH METHOD

In this study, several steps were taken to carry out the network and layer experimentation process on the Caltech 101 dataset using the CNN algorithm. The first step is to build a network architecture from CNN to be tested on CBIR work. The Figure 1, in general, displays the results of the image label classification process. Figure 1 explains an example of CNN architecture. As neural networks in general, CNN has several hidden layers of input in the form of a single vector. The neurons of each layer are connected, and so on. The last layer that is connected to the previous hidden layers is called the output layer and represents the final class classification.

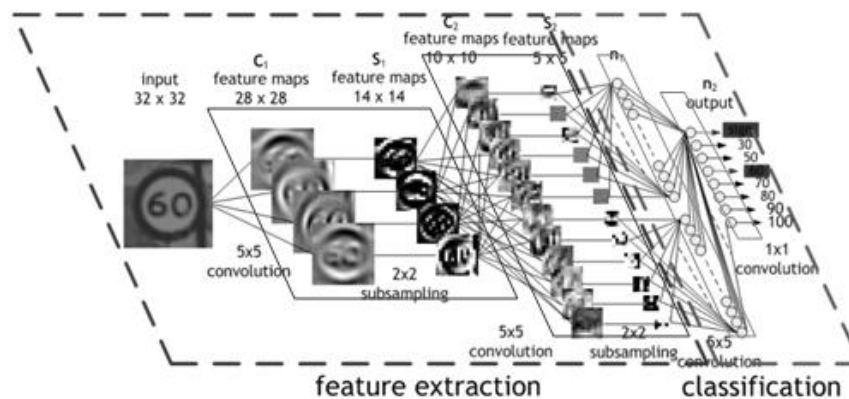


Figure 1. Architecture of CNN

Figure 2 is to experiment the network and layer architecture. The Figure 2 is quite capable of displaying steps on CNN. In general, there are 2 main steps in carrying out classification images in this study:

a. Feature learning

- The layers contained in feature learning are useful for translating an input into features based on the characteristics of the input in the form of numbers in vectors. This feature extraction layer consists of convolutional layer and pooling layer.
- Convolutional layer will calculate the output of neurons connected to the local area in the input, each calculating the product of the point between their weights and the small region connected to the input volume.

- Rectified linear Unit (ReLU) to eliminate vanishing gradients by applying the activation function to the element activation element will be carried out when it is at the threshold 0.
- Pooling layer to reduce the dimensions of the feature map or better known as a ledge for down sampling, thus speeding up computing because fewer parameters need to be updated and overfitting to overcome. Pooling commonly used is max pooling and average pooling. Max pooling to determine the maximum value per filter shift, while average pooling will determine the average value.

b. Classification

This step is to classify each neuron that has been extracted features:

- Flatten as reshape the feature (map) into a vector so that we can use it as input from the fully-connected layer.
- Fully-connected will calculate class scores. Like a normal neural network, each neuron in this layer will be connected to all the numbers in the volume.

Softmax to calculate the probability of each target class for all possible target classes and will help to determine the target class for the given input.

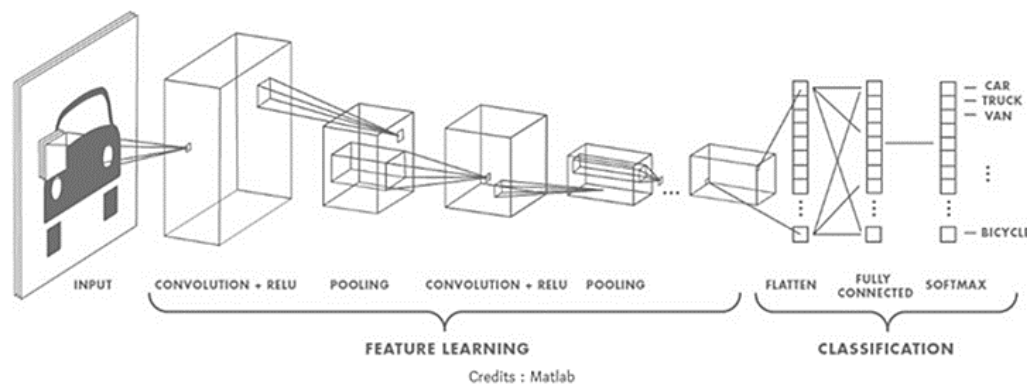


Figure. 2. Coverage available on CNN

3. RESULTS AND ANALYSIS

This section describes the results and analysis of the experiments carried out. In this part, it is explained the analysis of compound object image testing with deep learning and the analysis of dataset testing (caltech 101) with deep learning. The sections contained in this session include analysis of compound object image testing with deep learning and analysis of dataset testing (Caltech 101) with deep learning (CNN). This section describes the network model in testing with deep learning algorithms.

3.1. Analysis of compound object image testing with deep learning

Deep learning is used in core research to carry out testing in object detection work using CNN. The Python library is used to empower developers to independently build applications and systems with computer vision capabilities. In Figure 3, the image used for object detection experiments from various kinds of compound objects contained in one image file. By using the CNN algorithm, object detection is carried out in the image. Object categories that will be categorized after detection are person (P), truck (T), bicycle (B), Car (C) and Bus (B). The class of categories will be classified by the algorithm for comparison with human perception as an initial comparison in the object detection experiment in this study.

Figure 4 is the output image generated by the system in categorizing object types using the convolutional neural network algorithm. In the output of the detection results, it is known the percentage of accuracy of object classification and the type of class detected at once. The following table describes the value of detection accuracy and object categorization along with the types of objects carried out with CNN, there are object variables, n (number of objects detected), recognition (human and DL), and accuracy per object.

The object detection process in Figure 4 uses the Python programming language and uses the CNN algorithm by utilizing a number of libraries for deep learning functions such as Pandas. Pandas is a machine learning library in Python that consists of high-level data structures and various tools for performing data analysis. One of the great features of this library is its ability to translate complex analyzes relying on data from just one or two commands. Panda also has a lot of interesting capabilities when it comes to grouping, aggregating, and filtering data. The Pandas library is also constantly being updated, including hundreds of

new features, fixes, improvements and API changes. In python, which we implement, image processing is also done using the python library with OpenCV. The results of these experiments using CNN as shown in Table 1. From Table 1 we can see that the comparison of between the human recognition as match as and DL (deep Learning) recognition. The column "accuracy per object" is the value of recognition accuracy in every object. In average, the value is in range 57- 96%. So, the value is so much has insignificantly consistent.



Figure 3. Sample initial test image

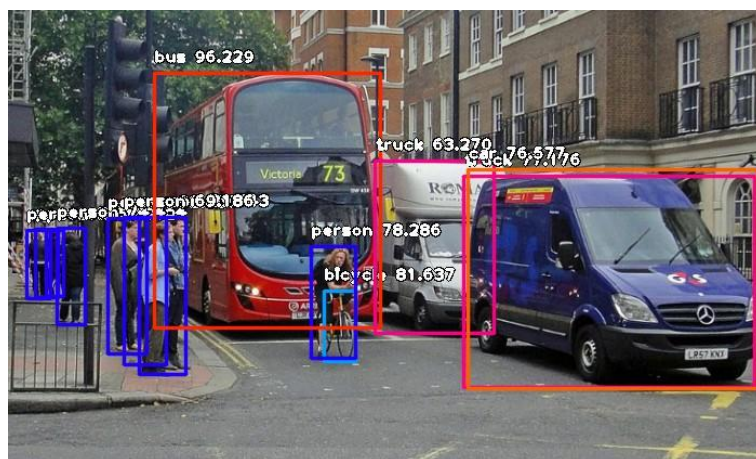


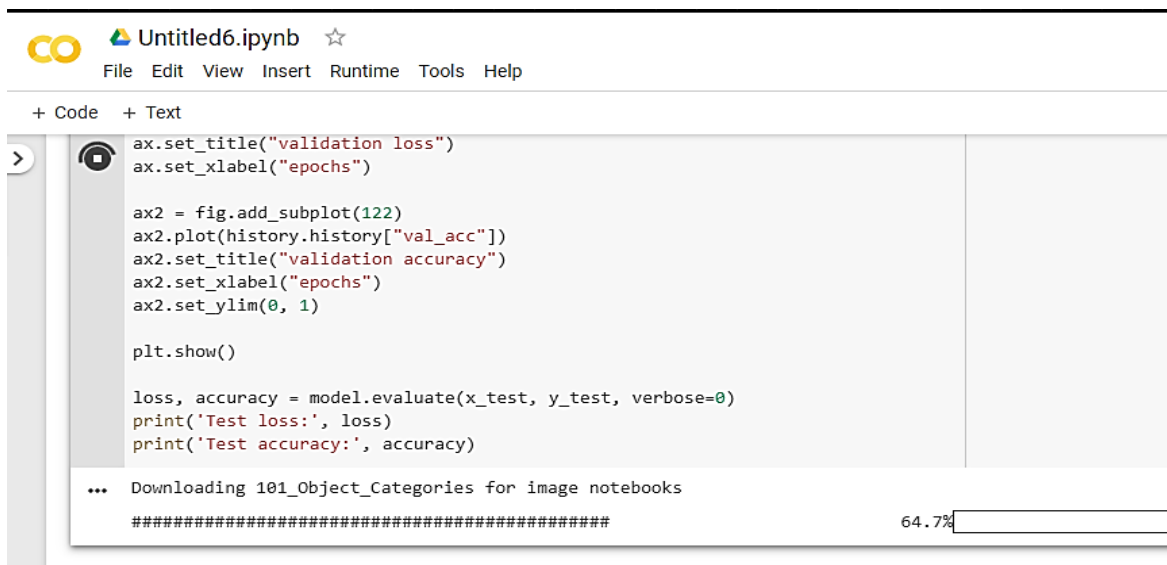
Figure 4. Object detection results using convolutional neural network

No	Object	n	Recognition		Accuracy Per Object
			Human	DL	
1	Person (P)	8	Person	Person	P1: 57.649195 P2: 50.662142 P3: 72.156518 P4: 74.453651 P5: 82.836878 P6: 89.743179 P7: 69.186091 P8: 78.285849
2	Truck (T)	2	Truck	Truck	T1: 63.270264 T2: 77.175962
3	Bicycle	1	Bicycle	Bicycle	81.636720
4	Car	1	Car	Car	76.576918
5	Bus	1	Bus	Bus	96.228921

3.2. Analysis of dataset testing (Caltech 101) with deep learning (CNN)

The dataset object image (Caltech 101) has 101 categories. The number of images is about 40 to 800 images per category. Most of the categories have about 50 images. Collected in September 2003 by Fei-Fei Li, Marco Andreetto, and Marc 'Aurelio Ranzato. Each image is approximately 300x200 pixels in size. The use of the Caltech 101 dataset was carried out to test the deep learning algorithm (CNN). This test is conducted to test the reliability of the CNN to test whether it is comparable to the results of previous tests using multiple images. In training and testing, the experiment was carried out randomly. When the test results will be evaluated the confusion matrix value is to find out which image is correctly and incorrectly classified. The following is Experiment I with 10 Epochs.

Figure 5 is the program code used to conduct testing and training using the CNN to test the Caltech 101 dataset. With object-oriented programming and Python programming language, we conduct training and testing to get the results of the Caltech 101 dataset testing on the deep learning method. In the Figure 5 program code an evaluation of accuracy, validation accuracy, number of epochs, a test loss is a data testing package which is split between the testing data and training data.



```

ax.set_title("validation loss")
ax.set_xlabel("epochs")

ax2 = fig.add_subplot(122)
ax2.plot(history.history["val_acc"])
ax2.set_title("validation accuracy")
ax2.set_xlabel("epochs")
ax2.set_ylim(0, 1)

plt.show()

loss, accuracy = model.evaluate(x_test, y_test, verbose=0)
print('Test loss:', loss)
print('Test accuracy:', accuracy)

```

... Downloading 101_Object_Categories for image notebooks
64.7%

Figure 5. Source code for testing and training

Furthermore, the Figure 5 shows the results of running the process of taking image categories from the dataset used to be tested in this study. The following part is the list of object category of Caltech 101 dataset:

Downloading 101_Object_Categories for image notebooks

100.0%

101_ObjectCategories sample_data

['101_ObjectCategories/garfield', '101_ObjectCategories/elephant', '101_ObjectCategories/accordion', '101_ObjectCategories/crab', '101_ObjectCategories/butterfly', '101_ObjectCategories/beaver', '101_ObjectCategories/binocular', '101_ObjectCategories/buddha', '101_ObjectCategories/wrench', '101_ObjectCategories/flamingo', '101_ObjectCategories/anchor', '101_ObjectCategories/soccer_ball', '101_ObjectCategories/camera', '101_ObjectCategories/grand_piano', '101_ObjectCategories/ewer', '101_ObjectCategories/bonsai', '101_ObjectCategories/hawkbill', '101_ObjectCategories/hedgehog', '101_ObjectCategories/dragonfly', '101_ObjectCategories/kangaroo', '101_ObjectCategories/windsor_chair', '101_ObjectCategories/sea_horse', '101_ObjectCategories/stapler', '101_ObjectCategories/laptop', '101_ObjectCategories/joshua_tree', '101_ObjectCategories/gerenuk', '101_ObjectCategories/ibis', '101_ObjectCategories/ferry', '101_ObjectCategories/umbrella', '101_ObjectCategories/gramophone', '101_ObjectCategories/nautilus', '101_ObjectCategories/inline_skate', '101_ObjectCategories/menorah', '101_ObjectCategories/pagoda', '101_ObjectCategories/metronome', '101_ObjectCategories/lamp', '101_ObjectCategories/pigeon', '101_ObjectCategories/water_lilly', '101_ObjectCategories/electric_guitar', '101_ObjectCategories/ceiling_fan', '101_ObjectCategories/pyramid', '101_ObjectCategories/platypus', '101_ObjectCategories/brain', '101_ObjectCategories/revolver', '101_ObjectCategories/dalmatian', '101_ObjectCategories/llama', '101_ObjectCategories/headphone', '101_ObjectCategories/pagoda', '101_ObjectCategories/okapi', '101_ObjectCategories/car_side', '101_ObjectCategories/crocodile', '101_ObjectCategories/mayfly', '101_ObjectCategories/yin_yang', '101_ObjectCategories/cougar_body', '101_ObjectCategories/emu', '101_ObjectCategories/lotus', '101_ObjectCategories/scissors', '101_ObjectCategories/stop_sign', '101_ObjectCategories/minaret', '101_ObjectCategories/dollar_bill', '101_ObjectCategories/barrel', '101_ObjectCategories/cougar_face', '101_ObjectCategories/ketch', '101_ObjectCategories/helicopter', '101_ObjectCategories/watch', '101_ObjectCategories/crayfish', '101_ObjectCategories/strawberry',

'101_ObjectCategories/dolphin', '101_ObjectCategories/brontosaurus', '101_ObjectCategories/rhino', '101_ObjectCategories/bass', '101_ObjectCategories/pizza', '101_ObjectCategories/lobster', '101_ObjectCategories/wild_cat', '101_ObjectCategories/saxophone', '101_ObjectCategories/crocodile_head', '101_ObjectCategories/stegosaurus', '101_ObjectCategories/mandolin', '101_ObjectCategories/panda', '101_ObjectCategories/snoopy', '101_ObjectCategories/wheelchair', '101_ObjectCategories/chair', '101_ObjectCategories/rooster', '101_ObjectCategories/chandelier', '101_ObjectCategories/trilobite', '101_ObjectCategories/ant', '101_ObjectCategories/starfish', '101_ObjectCategories/cannon', '101_ObjectCategories/cellphone', '101_ObjectCategories/sunflower', '101_ObjectCategories/tick', '101_ObjectCategories/scorpion', '101_ObjectCategories/euphonium', '101_ObjectCategories/octopus', '101_ObjectCategories/cup', '101_ObjectCategories/Leopards', '101_ObjectCategories/schooner']

The above part is a log of retrieving process result in images category from the datasets used for testing in this study. There are 97 (Ninety-seven) types of categories available in this Caltech 101 dataset. The category labels for each image appear after "101_ObjectCategories/*". This category label is used for the process of labeling used in data training. Then, the following part is the object code of one category:

[82, 37, 43, 89, 55, 87, 58, 18, 58, 90, 36, 62, 58, 96, 95, 24, 62, 15, 35, 70, 9, 86, 47, 63, 4, 16, 39, 74, 38, 59, 61, 3, 57, 69, 92, 30, 13, 76, 42, 64, 72, 61, 91, 88, 24, 50, 54, 9, 6, 21, 74, 67, 37, 71, 7, 37, 63, 65, 45, 86, 20, 25, 1, 82, 45, 71, 51, 32, 24, 83, 27, 36, 38, 11, 18, 7, 15, 24, 62, 16, 6, 62, 87, 42, 57, 2, 28, 95, 64, 9, 46, 49, 18, 88, 13, 22, 12, 86, 38, 60, 80, 13, 63, 13, 27, 88, 24, 85, 5, 64, 52, 91, 19, 40, 59, 30, 42, 17, 14, 29, 27, 37, 20, 4, 89, 63, 70, 20, 50, 64, 71, 81, 19, 62, 58, 16, 34, 64, 67, 57, 4, 4, 54, 32, 65, 10, 89, 89, 44, 17, 72, 76, 42, 89, 83, 70, 54, 95, 13, 43, 96, 87, 30, 92, 42, 66, 64, 42, 44, 58, 74, 0, 88, 27, 58, 75, 23, 33, 3, 18, 57, 63, 24, 7, 82, 62, 58, 87, 44, 92, 52, 26, 23, 79, 35, 66, 23, 14, 7, 50, 88, 95, 78, 24, 8, 59, 81, 80, 26, 60, 28, 26, 42, 60, 64, 14, 80, 32, 8, 91, 43, 91, 54, 60, 35, 54, 80, 62, 89, 14, 13, 95, 37, 54, 79, 45, 30, 36, 64, 54, 15, 17, 25, 3, 64, 32, 12, 16, 68, 59, 42, 43, 26, 11, 18, 58, 83, 60, 2, 49, 72, 74, 22, 64, 75, 88, 32, 23, 83, 82, 65, 82, 65, 56, 4, 82, 64, 16, 14, 10, 45, 86, 67, 90, 3, 35, 11, 0, 89, 68, 42, 73, 71, 61, 51, 55, 17, 63, 15, 8, 80, 84, 54, 18, 31, 17, 92, 48, 79, 32, 16, 25, 5, 13, 77, 15, 91, 74, 81, 93, 95, 53, 10, 16, 80, 65, 16, 38, 42, 23, 18, 24, 2, 36, 71, 64, 3, 16, 14, 52, 55, 76, 83, 70, 85, 28, 87, 19, 90, 93, 74, 95, 43, 52, 94, 38, 43, 16, 54, 96, 18, 65, 18, 96, 34, 35, 63, 65, 86, 62, 52, 35, 8, 23, 11, 40, 96, 39, 69, 49, 93, 33, 25, 64, 52, 14, 51, 43, 5, 82, 62, 9, 67, 32, 59, 18, 64, 27, 11, 13, 64, 32, 63, 49, 45, 64, 20, 0, 86, 43, 28, 27, 90, 84, 14, 95, 15, 95, 95, 64, 86, 78, 56, 74, 69, 32, 22, 95, 63, 26, 30, 89, 46, 75, 63, 4, 39, 49, 91, 49, 93, 32, 64, 40, 54, 15, 49, 70, 32, 42, 9, 13, 64, 6, 59, 62, 62, 64, 83, 86, 64, 32, 16, 30, 91, 95, 92, 76, 1, 21, 91, 16, 71, 32, 3, 39, 71, 95, 92, 62, 4, 62, 95, 42, 64, 40, 95, 95, 3, 32, 95, 15, 4, 32, 61, 59, 49, 74, 59, 13, 89, 64, 49, 4, 13, 68, 90, 62, 35, 78, 83, 24, 37, 49, 3, 63, 4, 9, 32, 15, 69, 72, 64, 11, 4, 71, 58, 82, 41, 39, 95, 39, 30, 35, 95, 58, 4, 95, 7, 3, 84, 27, 40, 44, 21, 45, 31, 24, 62, 61, 57, 12, 15, 65, 1, 7, 94, 22, 7, 84, 54, 18, 60, 30, 31, 86, 19, 96, 21, 43, 52, 21, 48, 15, 59, 21, 19, 49, 57, 14, 45, 77, 25, 18, 95, 31, 64, 20, 59, 29, 0, 16, 13, 60, 95, 87, 65, 83, 9, 56, 39, 95, 4, 15, 95, 13, 52, 16, 15, 73, 64, 87, 50, 95, 53, 28, 71, 32, 22, 51, 41, 85, 44, 26, 69, 16, 65, 7, 89, 17, 8, 62, 88, 63, 91, 62, 96, 7, 3, 42, 81, 45, 5, 64, 15, 83, 86, 65, 61, 9, 83, 64, 3, 86, 64, 46, 79, 71, 47, 78, 15, 18, 30, 86, 43, 24, 81, 4, 54, 63, 7, 23, 87, 23, 35, 91, 91, 64, 64, 7, 92, 68, 80, 57, 59, 35, 8, 7, 64, 49, 36, 33, 62, 16, 64, 7, 62, 62, 49, 28, 77, 38, 26, 52, 14, 95, 86, 21, 78, 92, 55, 95, 1, 55, 13, 62, 83, 6, 40, 16, 91, 57, 27, 10, 9, 52, 78, 96, 84, 84, 75, 51, 71, 92, 35, 15, 15, 67, 75, 47, 86, 80, 13, 13, 87, 42, 62, 49, 15, 31, 62, 11, 86, 62, 50, 44, 95, 32, 45, 49, 62, 29, 89, 61, 47, 65, 16, 71, 91, 71, 1, 37, 11, 90, 32, 48, 74, 49, 83, 79, 19, 14, 85, 70, 49, 91, 4, 17, 12, 36, 8, 91, 59, 4, 7, 64, 95, 3, 62, 39, 50, 76, 64, 75, 74, 13, 69, 91, 49, 36, 82, 55, 96, 66, 96, 39, 43, 22, 64, 33, 83, 95, 8, 29, 43, 49, 29, 29, 47, 12, 37, 35, 3, 56, 3, 82, 83, 49, 53, 53, 77, 4, 63, 77, 76, 44, 16, 82, 15, 4, 19, 89, 42, 3, 95, 1, 96, 49, 15, 84, 26, 37, 96, 20, 14, 86, 4, 7, 35, 61, 26, 79, 77, 64, 14, 32, 62, 19, 49, 90, 63, 75, 37, 85, 9, 61, 58, 16, 62, 76, 52, 64, 44, 86, 3, 0, 70, 75, 78, 57, 15, 74, 53, 30, 58, 41, 46, 94, 62, 4, 76, 64, 53, 14, 67, 28, 91, 70, 91, 63, 23, 90, 12, 9, 20, 64, 89, 80, 40, 64, 89, 42]

In the part above is the number value of images in each category. The number 82 in the first digit indicates the number of images in the first category, which is represented by the label "101_ObjectCategories/Garfield". Then, in the category '101_ObjectCategories/elephant' there are 37 images, as well as further in the category '101_ObjectCategories/accordion' and other categories. Next, the log data of testing and training data process of experiment:

```
(932, 97)
finished loading 6209 images from 97 categories
train / validation / test split: 4346, 931, 932
training data shape: (4346, 224, 224, 3)
training labels shape: (4346, 97)
Input dimensions: (224, 224, 3)
Instructions for updating:
Please use `rate` instead of `keep_prob`. Rate should be set to `rate = 1 - keep_prob`.
Model: "sequential_1"
```

The part above are logs of the process data tested and trained on the number of images in each category. There are 6029 total images from 97 categories. The data is divided into 4346 images for training data, 931 images for data validation and 932 images for data testing. The image used has size of 224 x 224 pixels with 3 RGB color channels. The following part is the log of CNN network tested for Caltech 101 dataset classification in experiment:

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 222, 222, 32)	896
activation_1 (Activation)	(None, 222, 222, 32)	0

```

max_pooling2d_1 (MaxPooling2 (None, 111, 111, 32) 0
-----
conv2d_2 (Conv2D) (None, 109, 109, 32) 9248
-----
activation_2 (Activation) (None, 109, 109, 32) 0
-----
max_pooling2d_2 (MaxPooling2 (None, 54, 54, 32) 0
-----
dropout_1 (Dropout) (None, 54, 54, 32) 0
-----
conv2d_3 (Conv2D) (None, 52, 52, 32) 9248
-----
activation_3 (Activation) (None, 52, 52, 32) 0
-----
max_pooling2d_3 (MaxPooling2 (None, 26, 26, 32) 0
-----
conv2d_4 (Conv2D) (None, 24, 24, 32) 9248
-----
activation_4 (Activation) (None, 24, 24, 32) 0
-----
max_pooling2d_4 (MaxPooling2 (None, 12, 12, 32) 0
-----
dropout_2 (Dropout) (None, 12, 12, 32) 0
-----
flatten_1 (Flatten) (None, 4608) 0
-----
dense_1 (Dense) (None, 256) 1179904
-----
activation_5 (Activation) (None, 256) 0
-----
dropout_3 (Dropout) (None, 256) 0
-----
dense_2 (Dense) (None, 97) 24929
-----
activation_6 (Activation) (None, 97) 0
=====
Total params: 1,233,473
Trainable params: 1,233,473
Non-trainable params: 0

```

The part above is a network model or neural network used in testing with a deep learning algorithm. neural network is a model as neurons in the human brain work. Each neuron in the human brain is interconnected and information flows from each of these neurons. The type layer "conv2d_1 (Conv2D)" is the input layer with the number of output shapes (None, 222, 222, 32) and the number of parameters as many as 896 pieces.

Each neuron receives input and performs a dot operation with a weight, weighted sum and adds bias. The results of this operation will be used as a parameter of the activation function that will be used as the output of the neuron. The layer type "activation_1 (Activation)" is an activation function with the number of output shapes (None, 222, 222, 32) and the number of parameters is 0 units.

Activation function to determine whether the neuron must be active or not based on the weighted sum of the input. In general, there are 2 types of activation functions, namely linear and non-linear. The linear function is the default activation function of a neuron. If a neuron uses a linear function, then the output of the neuron is the weighted sum of the input + bias.

Sigmoid function (Non-Linear) has a range between 0 to 1 while the range of and tanh function (non-linear) is -1 to 1. Both of these functions are usually used for the classification of 2 classes or groups of data. But there are weaknesses of both functions. The layer type "max_pooling2d_1 (MaxPooling2)" is the output layer with the number of output shapes (None, 111, 111, 32) and the number of parameters is 0.

Then, next part is the accuracy of each epoch iteration in the first try with epoch of 10 iterations:

```

Instructions for updating:
Use tf.where in 2.0, which has the same broadcast rule as np.where Train on 4346 samples, validate on 931 samples
Epoch 1/10
4346/4346 [=====]
- 255s 59ms/step - loss: 4.4912 - acc: 0.0520 - val_loss: 4.3379 - val_acc: 0.0730
Epoch 2/10
4346/4346 [=====]
- 251s 58ms/step - loss: 4.2142 - acc: 0.1029 - val_loss: 4.0124 - val_acc: 0.1353
Epoch 3/10
4346/4346 [=====]

```

```

- 251s 58ms/step - loss: 3.8286 - acc: 0.1740 - val_loss: 3.6576 - val_acc: 0.2030
Epoch 4/10
4346/4346
[=====]
- 251s 58ms/step - loss: 3.4403 - acc: 0.2437 - val_loss: 3.3470 - val_acc: 0.2782
Epoch 5/10.
4346/4346 [=====]
- 252s 58ms/step - loss: 3.0868 - acc: 0.3063 - val_loss: 3.0708 - val_acc: 0.3029
Epoch 6/10
4346/4346 [=====]
- 254s 59ms/step - loss: 2.7656 - acc: 0.3463 - val_loss: 2.8816 - val_acc: 0.3394
Epoch 7/10
4346/4346 [=====]
- 252s 58ms/step - loss: 2.4836 - acc: 0.4100 - val_loss: 2.7161 - val_acc: 0.3749
Epoch 8/10
4346/4346 [=====]
- 252s 58ms/step - loss: 2.1591 - acc: 0.4678 - val_loss: 2.6562 - val_acc: 0.3824
Epoch 9/10
4346/4346 [=====]
- 250s 58ms/step - loss: 1.9337 - acc: 0.5131 - val_loss: 2.6221 - val_acc: 0.3899
Epoch 10/10
4346/4346 [=====]
- 250s 58ms/step - loss: 1.7234 - acc: 0.5541 - val_loss: 2.5892 - val_acc: 0.3996
Test loss: 2.52228434454218
Test accuracy: 0.43776824034334766
    
```

The part above is the result of the accuracy of each epoch iteration in the first experiment with an epoch of 10 iterations. In average, from all 10 epochs the test loss results were obtained by 2.52228434454218 and the results of the test accuracy 0.43776824034334766 with an average execution time of 58.01 ms/step.

Figure 6 is the result of the 1st graph with the number of epochs of 10. Validation loss is a matrix that is almost the same as training loss, but validation loss is not used for update weights. Validation loss is calculated by running a forward search network against inputs and compared to outputs based on loss functions with individuals that differ between predicted values and targets. The validation loss graph in this one experiment has decreased. While the validation accuracy graph goes up. This proves that in the first experiment the results tended to increase insignificantly with a low achievement value of accuracy.

Figure 7 is the result of the 2nd graph with an epoch of 100. Validation loss is a matrix that is almost the same as training loss, but validation loss is not used for update weights. Validation loss is calculated by running a forward search network against inputs and compared to outputs based on loss functions with individuals that differ between predicted values and targets. The validation loss graph in experiment one has fluctuated. While the validation accuracy graph goes up. This proves that in experiment II the results are consistent with low accuracy achievement values, however, slightly better than experiments with epoch value of 10.

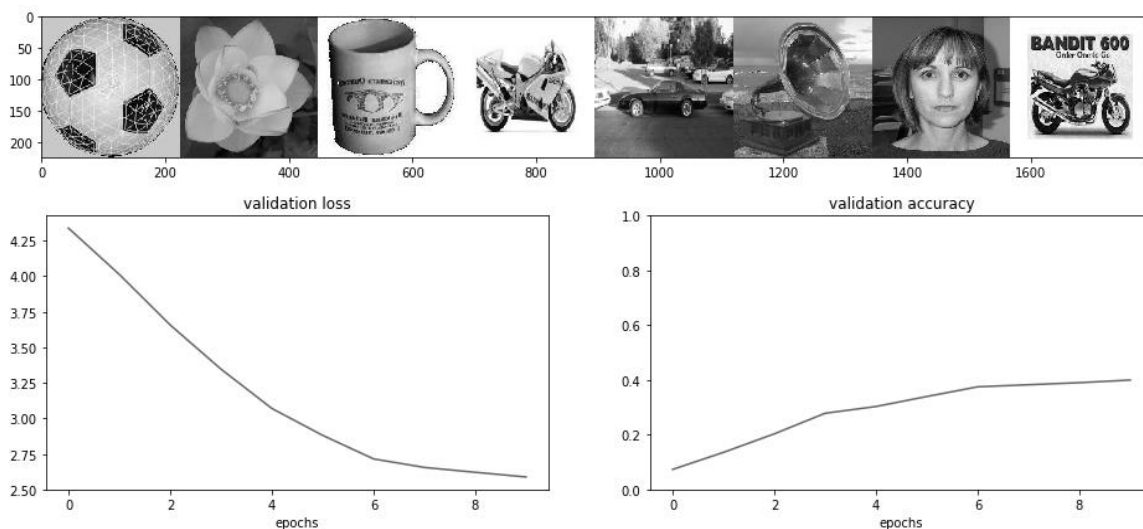


Figure 6. Results of experiment graph 1 with the number of epoch=10

The results of the epoch comparison in the caltech dataset shown in Table 2, it can be seen that the higher the epoch value, then the better the object classification. From the execution time, loss testing and testing accuracy, more iterations (epoch=100) also produce good values compared to values with epoch of 10, although the results obtained are still not optimal. When viewed from the graph as a whole validation loss decreases at epoch 10 and increases but, slightly fluctuating at epoch 100. Likewise, validation accuracy increases insignificantly at epoch 10, but tends to be stable at epoch 100 shows a network model that is good enough to be tested can then use an epoch of 100.

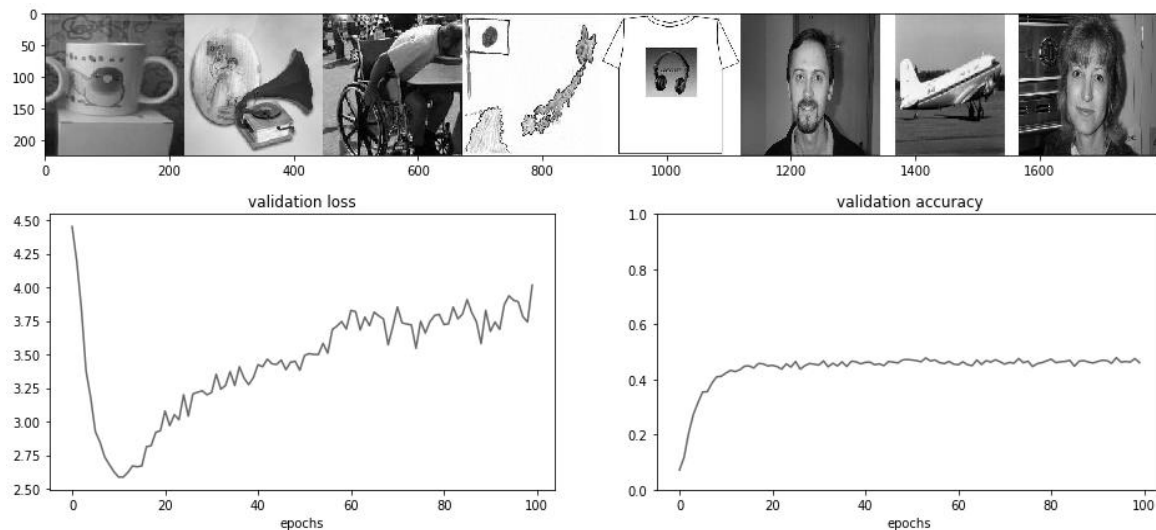


Figure 7. The results of the 2nd experiment chart with the number of epoch=100

Table 2. Trial epoch algorithm deep learning on caltech dataset

Testing Parameter	Epoch=10	Epoch=100
Execution Time	58.01 ms/step	32 ms/step
Loss Testing	2.52228434454218	3.968608917596514
Testing Accuracy	0.43776824034334766	0.44635193133047213
Validation Loss	decreased	increased and slightly fluctuated
Validation Accuracy	increased insignificantly	tend to be stable

4. CONCLUSION

From the results of the epoch comparison experiment using deep learning algorithm on the caltech 101 dataset, shown that, a higher epoch value can result a better object classification. From the execution time, loss testing and testing accuracy, more iterations (epoch=100) produce good values compared to values with epoch of 10, although the results obtained are still not optimal. When viewed from the graph as a whole validation loss decreases at epoch 10, but slightly fluctuating at epoch 100. Likewise, validation accuracy increases insignificantly at epoch 10, but tends to be stable at epoch 100. It shows a network model with epoch of 100 is good enough to be tested.

ACKNOWLEDGEMENTS

Thank you to the Dinamika Bangsa Foundation for providing material and moral support in carrying out this research.




REFERENCES

- [1] M. Rehman, M. Iqbal, M. Sharif, and M. Raza, "Content based image retrieval: survey," *World Appl. Sci. Journal, ISSN 1818-4952*, vol. 19, no. 3, pp. 404–412, 2012, doi: 10.5829/idosi.wasj.2012.19.03.1506.
- [2] L. Haldurai and V. Vinodhini, "A study on content based image retrieval systems," *Int. J. Innov. Res. Comput. Commun. Eng.*, vol. 3, no. 3, pp. 2554–2559, 2015, doi: 10.15680/ijirccce.2015.0303186.
- [3] B. P. S. S. Ashish Mohan Yadav, "A survey on: 'content based image retrieval systems,'" *Int. J. Emerg. Technol. Adv. Eng.*, vol. 4, no. 6, pp. 22–26, 2014, doi: 10.5120/802-1139.




- [4] A. M. U. Ahamed, C. Eswaran, and R. Kannan, "CBIR system based on prediction errors," *J. Inf. Sci. Eng.*, vol. 33, no. 2, pp. 347–365, 2017, doi: 10.1688/JISE.2017.33.2.5.
- [5] F. Mayer and M. Steinebach, "Forensic image inspection assisted by deep learning," *Proc. 12th Int. Conf. Availability, Reliab. Secur. - ARES '17*, pp. 1–9, 2017, doi: 10.1145/3098954.3104051.
- [6] I. Supriana and Y. Pratama, "Face recognition new approach based on gradation contour of face color," vol. 9, no. 1, pp. 125–138, 2017, doi: 10.15676/ijeei.2017.9.1.8.
- [7] H. Pourghassem and S. Daneshvar, "A framework for medical image retrieval using merging-based classification with dependency probability-based relevance feedback," *Turkish J. Electr. Eng. Comput. Sci.*, vol. 21, no. 3, pp. 882–896, 2013, doi: 10.3906/elk-1010-876.
- [8] D. Z. Abidin, S. Nurmaini, Erwin, E. Rasywir, and Y. Pratama, "Indoor positioning system in learning approach experiments," *J. Electr. Comput. Eng.*, vol. 2021, 2021, doi: 10.1155/2021/6592562.
- [9] S. Kornblith, J. Shlens, and Q. V. Le, "Do better imagenet models transfer better?," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2019-June, pp. 2656–2666, 2019, doi: 10.1109/CVPR.2019.00277.
- [10] N. Rusk, "Deep learning," *Nat. Methods*, vol. 13, no. 1, p. 35, 2015, doi: 10.1038/nmeth.3707.
- [11] Fachruddin, Y. Pratama, E. Rasywir, D. Kisbianty, Hendrawan, and M. R. Borroek, "Real time detection on face side image with ear biometric imaging using integral image and haar-like feature," in *Proceedings of 2018 International Conference on Electrical Engineering and Computer Science, ICECOS 2018*, 2019, pp. 165–170, doi: 10.1109/ICECOS.2018.8605218.
- [12] Y. Pratama and E. Rasywir, "Automatic cost estimation analysis on datawarehouse project with modified analogy based method," *Proc. 2018 Int. Conf. Electr. Eng. Comput. Sci. ICECOS 2018*, pp. 171–176, 2019, doi: 10.1109/ICECOS.2018.8605195.
- [13] E. Rasywir, Y. Pratama, Hendrawan, and M. Istoningtyas, "Removal of modulo as hashing modification process in essay scoring system using rabin-karp," *Proc. 2018 Int. Conf. Electr. Eng. Comput. Sci. ICECOS 2018*, vol. 2019-Janua, pp. 159–164, 2019, doi: 10.1109/ICECOS.2018.8605211.
- [14] A. Alzu'bi, A. Amira, and N. Ramzan, "Content-based image retrieval with compact deep convolutional features," *Neurocomputing*, vol. 249, pp. 95–105, 2017, doi: 10.1016/j.neucom.2017.03.072.
- [15] C. Sun, A. Shrivastava, S. Singh, and A. Gupta, "Revisiting unreasonable effectiveness of data in deep learning era," *Proc. IEEE Int. Conf. Comput. Vis.*, vol. 2017-October, pp. 843–852, 2017, doi: 10.1109/ICCV.2017.97.
- [16] P. Liu, J. M. Guo, C. Y. Wu, and D. Cai, "Fusion of deep learning and compressed domain features for content-based image retrieval," *IEEE Trans. Image Process.*, vol. 26, no. 12, pp. 5706–5717, 2017, doi: 10.1109/TIP.2017.2736343.
- [17] F. Battistone and A. Petrosino, "TGLSTM: A time based graph deep learning approach to gait recognition," *Pattern Recognit. Lett.*, vol. 126, pp. 132–138, 2019, doi: 10.1016/j.patrec.2018.05.004.
- [18] F. Fachruddin, S. Saparudin, E. Rasywir, Y. Pratama, and B. Irawan, "Extraction of object image features with gradation contour," *TELKOMNIKA (Telecommunication Comput. Electron. Control.)*, vol. 19, no. 6, p. 1913, 2021, doi: 10.12928/telkomnika.v19i6.19491.
- [19] J. M. Alvarez and M. Salzmann, "Learning the Number of neurons in deep networks," in *NIPS 2016*, 2016, no. Nips.
- [20] J. Camacho-Collados and M. T. Pilehvar, "On the role of text preprocessing in neural network architectures: an evaluation study on text categorization and sentiment analysis," *arxiv Comput. Lang.*, 2017, doi: 10.1016/j.uology.2011.07.1392.
- [21] W. Liu, Z. Wang, X. Liu, N. Zeng, Y. Liu, and F. E. Alsaadi, "A survey of deep neural network architectures and their applications," *Neurocomputing*, vol. 234, no. November 2016, pp. 11–26, 2017, doi: 10.1016/j.neucom.2016.12.038.
- [22] K. T. Ahmed, S. Ummesafi, and A. Iqbal, "Content based image retrieval using image features information fusion," *Inf. Fusion*, vol. 51, pp. 76–99, 2019, doi: 10.1016/j.inffus.2018.11.004.
- [23] A. K. Dhara, S. Mukhopadhyay, A. Dutta, M. Garg, and N. Khandelwal, "Content-based image retrieval system for pulmonary nodules: assisting radiologists in self-learning and diagnosis of lung cancer," *J. Digit. Imaging*, vol. 30, no. 1, pp. 63–77, 2017, doi: 10.1007/s10278-016-9904-y.
- [24] K. T. Ahmed, S. A. H. Naqvi, A. Rehman, and T. Saba, "Convolution, approximation and spatial information based object and color signatures for content based image retrieval," *2019 Int. Conf. Comput. Inf. Sci. ICCIS 2019*, pp. 1–6, 2019, doi: 10.1109/ICCISci.2019.8716437.
- [25] Y. Wang *et al.*, "EV-gait: Event-based robust gait recognition using dynamic vision sensors," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2019-June, pp. 6351–6360, 2019, doi: 10.1109/CVPR.2019.00652.
- [26] A. Haider, Y. Wei, S. Liu, and S. H. Hwang, "Pre- and post-processing algorithms with deep learning classifier for Wi-Fi fingerprint-based indoor positioning," *Electron.*, vol. 8, no. 2, 2019, doi: 10.3390/electronics8020195.
- [27] L. Li, H. Feng, W. Zhuang, N. Meng, and B. Ryder, "CCLearner: A deep learning-based clone detection approach," *Proc. - 2017 IEEE Int. Conf. Softw. Maint. Evol. ICSME 2017*, pp. 249–260, 2017, doi: 10.1109/ICSME.2017.46.

BIOGRAPHIES OF AUTHORS






Fachruddin    received a Bachelor's degree (S.Pt) in Agriculture from Jambi University. He obtained a Master's degree (M.S.I) in Information Systems from the Dinamika Bangsa University. He is a student of the Doctoral Program, Faculty of Engineering, Sriwijaya University. He is a Lecturer in Computer Science, Informatics Engineering, Universitas Dinamika Bangsa (UNAMA). Artificial Intelligence (AI), and Information Systems. He can be contacted at email: fachruddin.stikom@gmail.com.






Saparudin    received a Bachelor (S1) in Mathematics Education, Sriwijaya University, graduated in 1993. Then continued with Master (S2) In Informatics Engineering, Bandung Institute of Technology, Software Engineering, Image Processing and Computer Vision, graduated in 2000. And took a Doctoral Degree in Philosophy (S3) in Computer Science, Universiti Teknologi Malaysia (UTM), expertise in Image Processing, Computer Vision and Pattern Recognition, graduated in 2012. He is currently a teaching professor at Telkom University, Indonesia. He can be contacted at this email: saparudin@telkomuniversity.ac.id.



Errissya Rasywir    received the Bachelor degree (S.Kom) in computer science from the Sriwijaya University. She received the Master degree (M.T) in Informatics Master STEI from the Institut Teknologi Bandung (ITB). She is a Lecture of Computer Science in the Informatics Engineering, Dinamika Bangsa University (UNAMA). In addition, she is serving as Head of the research group (LPPM) on UNAMA. Her research interests are in data mining, Artificial intelligent (AI), natural language processing (NLP), machine learning, deep learning. She can be contacted at email: errissya.rasywir@gmail.com.



Yovi Pratama    received the Bachelor degree (S.Kom) in computer science from the Sriwijaya University. He received the Master degree (M.T) in Informatics Master STEI from the Institut Teknologi Bandung (ITB). He is a Lecture of Computer Science in the Informatics Engineering, Dinamika Bangsa University (UNAMA). In addition, he is serving as Information Technology Division (IT Division) on UNAMA. His research interests are in data mining, Artificial intelligent (AI), natural language processing (NLP), machine learning, and deep learning. He can be contacted at email: yovi.pratama@gmail.com.