

Road Identification using Convolutional Neural Network on Autonomous Electric Vehicle

Markus Hermawan

Department of Electrical Engineering
Faculty of Engineering, Sriwijaya University
Ogan Ilir 30662 Sumatera Selatan, Indonesia
markusherawan104@gmail.com

Zaenal Husin

Electrical Engineering
Faculty of Engineering, Sriwijaya University
Ogan Ilir 30662 Sumatera Selatan, Indonesia
zaenalhusin@gmail.com

Hera Hikmarika

Department of Electrical Engineering
Faculty of Engineering, Sriwijaya University
Ogan Ilir 30662 Sumatera Selatan, Indonesia
herahikmarika@gmail.com

Suci Dwijayanti

Department of Electrical Engineering
Faculty of Engineering, Sriwijaya University
Ogan Ilir 30662 Sumatera Selatan, Indonesia
sucidwiyanti@ft.unsri.ac.id

Bhakti Yudho Suprpto

Department of Electrical Engineering
Faculty of Engineering, Sriwijaya University
Ogan Ilir 30662 Sumatera Selatan, Indonesia
bhakti@ft.unsri.ac.id

Abstract — *Research in the field of autonomous electric vehicle has growth rapidly since they can overcome traffic accidents due to human error. Currently, the method used to identify the road for an autonomous electric vehicle is not in real-time. Thus, this study proposed a method for the autonomous electric vehicle to follow a predetermined route by identifying the road using the Convolutional Neural Network (CNN) as input of the steering control system. The optimal CNN model was obtained using an optimizer of Stochastic Gradient Descent with 150 epoch optimizer that was then used in simulation testing and real-time testing. In simulation testing, from 15 trials conducted, the percentage of success was 93.333%. The success rate to transmit the data from the system to the tool in a real-time manner is 100%. In real-time testing, the autonomous electric vehicle was successfully able to follow the predetermined route accurately. However, the autonomous electric vehicle has not succeeded in avoiding the object in front of it due to the lack of precise steering mechanics and the lack of variation in training data from various conditions that may be passed by the autonomous electric vehicle.*

Keywords: *Autonomous Electric Vehicle, CNN, Road Identification.*

I. INTRODUCTION

Human error is the main factor causing an accident, especially in traffic accidents which have caused various losses, such as material losses, injuries, to the loss of human life. This encourages continuous development in autonomous electric vehicles to overcome these problems [1]. The autonomous electric vehicle is a vehicle that refers to automation without human intervention that is integrated from automobile science, electricity and electronics, geography, computers, IT. It is equipped with a self-driving system [2] that can be interpreted as a system to facilitate human's work in terms of driving, such as moving the gas throttle, gear shift, brakes, and steering automatically. The development of this self-driving system has also been integrated with various research methods, such as using ultrasonic sensors, radar, lidar, GPS, or cameras to substitute human's sense [3].

The autonomous electric vehicle must have an ability to recognize the existing environmental conditions so that the vehicle can operate automatically without human assistance on a predetermined route [4]. Environmental conditions can be visualized with a navigation system by using cameras as vision which works as eyes to detect roads and applying ultrasonic proximity sensors to detect obstacles to prevent accidents.

Several studies regarding road detection on autonomous electric vehicles have been carried out by applying various

methods, such as the Hue Saturation Value (HSV) color filtering method [3], Hough Transform [5], Support Vector Machine (SVM) [6], mask R-CNN [7], and Convolutional Neural Network (CNN) [8]. Those methods have shown good results in road detection. However, [3] and [5] were only able to detect roads with marking lines. Meanwhile, the SVM method was used to detect unstructured roads [6]. Then, the mask R-CNN method was utilized to detect roads from various conditions, such as snow, rain, fog, and others [7]. However, these methods can't be performed in real-time. The research conducted by Jihun Kim and Minh Lee using the CNN method was carried out in real-time and obtained quite accurate results on complex road conditions. Nevertheless, these roads had road markings [8].

To overcome the problems in the previous studies, this work is conducted to detect roads in real-time using CNN. The work is implemented for the roads which do not have any line markings. The CNN has the advantage to recognize objects in the image at various possible positions (translation invariance). In addition, it has a more powerful computational process, not limited to large-scale datasets, and techniques for training deeper networks [9]. In this study, the out of CNN become an input that provides a signal to the controller in the form of coordinates to the microcontroller through serial communication. These coordinates determine the direction of the steering so that the autonomous electric vehicle can operate automatically on a predetermined route.

This paper is organized as follows. Section 2 describes the convolutional neural network. Then, Section 3 presents the methods used in this work. The results and discussion are shown in Section 4. This work is then concluded in Section 5.

II. CONVOLUTIONAL NEURAL NETWORK (CNN)

Convolutional Neural Network (CNN) is one of the machine learning method developed from Multi-Layer Perceptron (MLP) which is generally used to process or create data. CNN is also a type of deep neural network because it has a high level of network depth which is suitable for processing images as input. In CNN, the input layer or data propagated on the network is in two dimensions which is different from MLP that only uses the form of one dimension [10]. Therefore, the linear operations are convolution and weight parameters on CNN are no longer one-dimensional, but they are in the form of four dimensions which are a collection of various convolution kernels. Fig.1. Shows the network architecture of CNN.

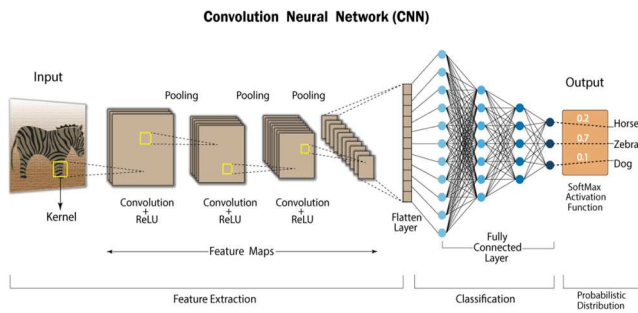


Fig. 1. CNN Architecture [11]

Based on the CNN network architecture, as shown in Fig.1, the CNN structure consists of input, the feature extraction process, classification process, and output. In the CNN, feature extraction consists of several hidden layers, namely convolution, activation function (ReLU), and pooling. While the classification process consists of a fully connected layer and an activation function (softmax) which will produce an output in the form of classification results [12].

In the CNN, feature extraction process is performed in the convolution layer that will use a filter to extract objects from the input image that contains weights to detect the character of the object to produce a linear transformation of the input image that matches the spatial information in the data. In the convolution layer, there are three parameters that can be changed in the filter, namely filter size, stride, and padding. Stride is a parameter that determines the number of shifts in the input data filter results. While, the padding is a parameter for adding pixel sizes (containing a value of 0) on each side of the input data which aims to manipulate the output dimensions of the convolution layer. After performing the convolution process, an activation function (ReLU) is performed. This is an operation to introduce nonlinearity and to improve the representation of the model. If the input is negative, the output value of the neuron can be expressed as 0. Meanwhile, if the input value is positive, then the output value of the neuron is expressed by the value of the activation input itself [13]. After the activation function (ReLU), there is pooling or subsampling that reduces the size of the matrix aimed to overcome overfitting and speed up computing with fewer parameters. In the CNN classification process, the fully-connected layer is a layer with neurons that are entirely connected between the neurons of the previous layer and the next layer. This fully connected layer aims to transform the data dimensions so that data can be classified linearly. Next, the activation function softmax will generate a probability that has not been normalized based on the interpreted value for each class.

III. RESEARCH METHODS

A. System Design

The design is divided into two, namely software and hardware design. In software design, the process of detecting roads based on CNN is programmed using PyCharm software with Python programming language. The output of CNN is then sent to Arduino through serial communication. The data are in the form of class as results of road segment detection. The output signal is then used as the input in the steering control on the autonomous electric vehicle so the vehicle can move and follow the predetermined route. The other design is hardware design that includes the layout and type of hardware

used in the autonomous electric vehicle which can be seen in Fig. 2.

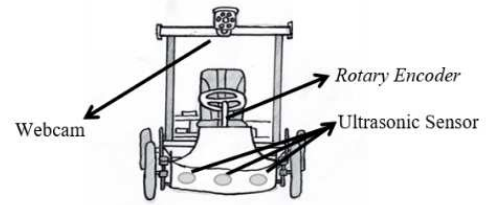


Fig. 2. Sketch of the Position of the Webcam, Rotary Encoder, and Ultrasonic Sensor

B. Testing

Testing is carried out to determine the error rate and also the accuracy of the system which has been previously designed with the CNN to identify the road. In this study, testing was carried out in 2 stages, namely a simulation and testing in real-time.

The simulation test is conducted to determine whether the designed system can be used on an autonomous electric vehicle in real-time. This test was carried out using the model from the previous training using video that consists of road in Sriwijaya University, Indralaya. The performance of a model can be seen in an accuracy. To calculate the accuracy value, the confusion metric evaluation method is used as can be seen in Table I.

TABLE I. CONFUSION METRIC

		Actual Value	
		True Positive (TP)	False Positive(FP)
Predicted Values	True Positive (TP)		
	False Negative(FN)		True Negative(TN)

From the table There are two variables in the method, Confusion Metric namely predicted values and actual values. The predicted value is the predicted value given by the CNN model, while the actual value is a predetermined target value. Based on the table Confusion Metric above, the accuracy of the model can be determined using the following equation:

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

Meanwhile, in this test real-time, the model used in the simulation test is validated with the real situation using the autonomous electric vehicle. This vehicle is implemented to the road within the Sriwijaya University, Indralaya with a route that has been determined. The results of the detection with the CNN algorithm is later sent to Arduino in the form of road identification class data so that it can be processed to determine the steering direction. The standard of success from this real-time test is that autonomous electric vehicles can operate following a predetermined road route automatically.

IV. RESULTS AND DISCUSSION

A. Autonomous Electric Vehicle Design

Before collecting training data, the design of the autonomous electric vehicle is carried out by providing various necessary components, such as sensors and microcontrollers. The design of this autonomous electric vehicle consists of mechanical design and wiring can be seen in Fig. 3.



Fig. 3. Autonomous Electric Vehicle Front View (a), Side View (b), and Steering Control and Camera Location (c)

B. Training Data Collection

The data used is primary data in the form of road images as input for training the CNN. Data collection was taken using an external Logitech C925e camera that was installed on the autonomous electric vehicle. The data were in video format (.mp4) of the roads around campus of Universitas Sriwijaya, Indralaya during the day. Then, each frame in the video was converted into images using the Free Video to JPG Converter application and saved in .JPG format. Those image data were grouped into 7 classes. Fig. 4 shows some sample images of the training data that have been collected.

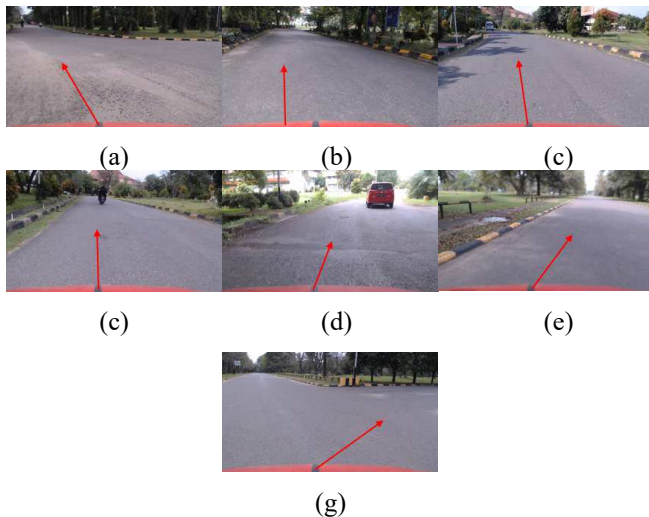


Fig. 4. Sample Image Training Data : Left Broken (a), Left Medium (b), Left (c), Straight (d) , Right (e), Medium Right (f), and Right Broken (g)

Finally, there are 2382 images converted from videos used as training data. Then, the data was processed by grouping the data according to their respective classes.

C. Training Data Processing

The collected training data images were labelled by grouping the images based on the perspective observation according to their respective classes. In this study, 7 classes are used, namely left fracture (0), moderate left (1), left (2), straight (3), right (4), right moderate (5), and right fracture (6). The results of this training data processing can be seen in Fig. 5.

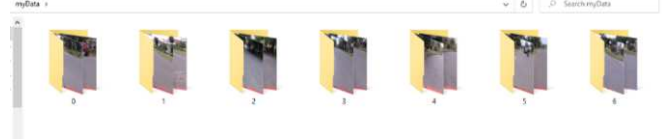


Fig. 5. Training Data Processing Results

D. CNN Training

The training process to identify roads was carried out using the CNN. In the training process, the pickle library in python was used to get the training result model. The resulted training model was then saved in format .p. The architecture of CNN used in this study can be seen in Table II.

TABLE II. CNN ARCHITECTURE

Layer (Type)	Output Shape	Parameter
conv2d_1 (Conv2D)	(None, 28, 28, 60)	1560
conv2d_2 (Conv2D)	(None, 24, 24, 60)	90060
max_pooling2d_1 (MaxPooling2)	(None, 12, 12, 60)	0
conv2d_3 (Conv2D)	(None, 10, 10, 30)	16230
conv2d_4 (Conv2D)	(None, 8, 8, 30)	8130
max_pooling2d_2 (MaxPooling2)	(None, 4, 4, 30)	0
dropout_1 (Dropout)	(None, 4, 4, 30)	0
flatten_1 (Flatten)	(None, 480)	0
dense_1 (Dense)	(None, 500)	240500
dropout_2 (Dropout)	(None, 500)	0
dense_2 (Dense)	(None, 7)	3507

The hardware specifications used during the training are NVIDIA GEFORCE 930MX GPU, Intel i5 Processor, 8GB RAM. This hardware is also supported by several programs including NVIDIA GPU drivers, CUDA 10, cuDNN SDK 7.4, python 3.6, TensorFlow-GPU 1.15, OpenCV 4.5.1. hard 2.2.3, etc.

In training, 2 types of optimizers were used, namely the Stochastic Gradient Descent (SGD) optimizer and the Adam optimizer. The optimizer SGD works by reducing the value of the function by changing the parameter values step by step, while the Adam works by calculating individual learning rates for separate parameters. This training aims to get the best model. The parameters used in the training with the optimizer SGD and Adam had three different number of epochs, 100, 150, and 200 epochs, respectively. Both optimizers used a learning rates of 0.001. The performance of a model for each optimizer can be seen from the accuracy and loss values.

CNN training process using the optimizer SGD and Adam will be carried out with several epochs of 100, 150, and 200 epochs with learning rates the same of 0.001. The parameters

used to determine the best model to be used for the whole system are based on the level of accuracy and the resulting loss value. Comparison of the value loss and accuracy of results training CNN using the optimizers SGD and Adam can be seen in Table III.

The loss values and accuracy using the optimizers of SGD and Adam can be seen in Table III

TABLE III. COMPARISON OF LOSS AND ACCURACY

Epoch	Optimizer SGD		Optimizer Adam	
	Loss	Accuracy	Loss	Accuracy
100	0.6389	0.7533	1.7930	0.7323
150	0.6133	0.7743	1.8498	0.7087
200	0.7881	0.7428	2.0672	0.6877

Based on Table III above, using SGD optimizer and epoch of 100, the loss and accuracy were 0.5970 and 0.7533, respectively. Furthermore, in the 150 epoch training, the value loss became smaller than the 100 epochs, which is 0.6133. The accuracy value has improved into 0.7743. In 200epoch, the loss values had the greatest improvement compared to the two previous epochs, which was 0.7881. Nevertheless the accuracy became 0.7428 which is smaller than the result of 150 epochs.

When Adam was used as the optimizer, the loss values and accuracy obtained in 100 epochs were 1.7930 and 0.7323, respectively. Furthermore, in the 150 epochs, the value loss was greater than the previous training, which was 1.8498. The accuracy value decreased to 0.7087. In 200 epochs, the loss value was 2.0672 which is larger than the results obtained from 100 and 150 epochs. The accuracy value was 0.6877 which is smaller compared to 100 and 200 epochs. The graph of the training data is shown in Fig. 6.

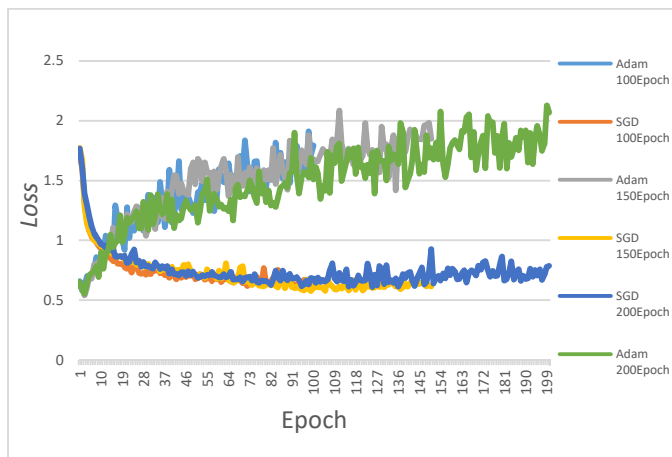


Fig. 6. Comparison Graph of Training Results Optimizer SGD and Adam

The best training model is the model with the highest accuracy level and the smallest loss value [14][15]. Based on this, it can be concluded that the training model obtained by SGD with 150 epochs is the best CNN model to be applied to the system as a whole. Thus, this model was then implemented to the testing data to see its robustness in identifying the data that were not included in the training process.

E. System Simulation Testing

Testing is carried out in the simulation data to see the ability of the model to generalize the new data that do not exist in the training data. This test also examines the process of sending data from the system as the class data which was obtained from road identification using the best training model. The process of sending class data from the road identification was compiled in the form of a number code, namely the number "0" to the number "6" which is then sent from the system to the device. Later, this number code is used as an input to the steering control system of autonomous electric vehicle in real-time test.

This simulation test was done using the video data which are different from the training data. The results of the simulation test can be seen in Table IV.

TABLE IV. SIMULATION TESTING RESULT

No	Identification Image Results	Class Data Received	Description
1.			Success

Table IV shows that the system has been able to perform class classification, and was able to send information in the form of class data from the system to the autonomous electric vehicle. From 15 tests, the percentage of success was 93.333% with only one class error and 100% for sending data from the system to the main microcontroller in autonomous electric vehicle. This error occurs because the data used for classes "0" (sharp turns) and "6" (left or right fractures) are slightly similar so it is quite difficult to distinguish them. In addition, the lack of variation in the training data used also causes the error.

F. Real-time System Testing

Real-time test is carried out to see if the autonomous electric vehicle can move following a predetermined route on the Sriwijaya University, Indralaya. The test route for real-time implementation of autonomous electric vehicles can be seen in Fig. 7. The distance for this test is 1.7 km.



Fig. 7. Autonomous Electric Vehicle Real-time Test Route

This test was carried out by taking as 12 test route points and a condition when an object was blocked. Data was retrieved automatically as the movement of autonomous electric vehicles for each frame in each point of the test route with an average speed of 14 km/hour. The results of this test can be seen in Table V.

TABLE V. TEST RESULT REAL-TIME SYSTEM

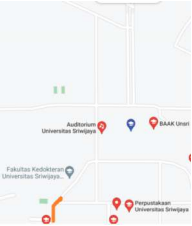

No	Testing Route	Autonomous Electric Vehicle Movement	Description
1.			Successfully Following the Path

Table V shows that the movement of the autonomous electric vehicle was less stable and sometimes out of the line. In the 13th test, when there was an object/obstacle in front of it, the autonomous electric vehicle did not manage to pass the obstacle properly. At first, the autonomous electric vehicle managed to turn to avoid the object in front, but its movement was too tight on the object and then moved towards the object so that it did not manage to avoid the object properly. This can happen due to the lack of precision steering mechanics. Thus, the accuracy of the steering wheel degrees when turning becomes inaccurate. Besides, the lack of variation in training data from various possible conditions passed by an autonomous electric vehicle may also cause this failure.

However, from the results of system test that have been carried out, it shows that the system is able to identify classes of various road conditions that are passed as input to the steering control system so that the autonomous electric vehicle is able to follow a predetermined route.

V. CONCLUSION

Based on the research that has been done, it can be concluded that the best model of CNN was in an optimizer of stochastic gradient descent and epoch of 150 because it had a smaller loss value of 0.6133 and a higher accuracy rate of 0.7743 compared to the training model using the Adam as the optimizer. This study has shown that the Convolutional Neural Network (CNN) can identify roads which then became inputs to the autonomous electric vehicle steering control system with 93.333% success in simulation test and

100% for data transmission. The autonomous electric vehicle was successfully able to follow the route which has been specified.

For future work, the mechanical of autonomous electric vehicle must be repaired again especially inaccurate steering wheel degrees and asymmetrical front tire position. More training data is needed with a variety of conditions especially the road condition that is passed by autonomous electric vehicles. And increase the accuracy of CNN's model by using K-Fold Cross-Validation and LSTM algorithm for the CNN classification.

ACKNOWLEDGEMENTS

The research/publication of this article was funded by DIPA of Public Service Agency of University Sriwijaya 2021. SP DIPA-023.17.2.677515 /2021, On November 23, 2020. In accordance with the Rector's Decree Number: 0010/UN9/SK.LP2M.PT/2021, On April 28, 2021.

REFERENCE

- [1] W. Farag, "Recognition of traffic signs by convolutional neural nets for self-driving vehicles," *Int. J. Knowledge-Based Intell. Eng. Syst.*, vol. 22, no. 3, pp. 205–214, 2018, doi: 10.3233/KES-180385.
- [2] Y. G. Choi, K. Il Lim, and J. H. Kim, "Lane change and path planning of autonomous vehicles using GIS," *2015 12th Int. Conf. Ubiquitous Robot. Ambient Intell. URAI 2015*, pp. 163–166, 2015, doi: 10.1109/URAI.2015.7358855.
- [3] M. Taufiqurrahman, S. Sumardi, and M. A. Riyadi, "Perancangan Self Driving Dengan Metode Kontrol Pd Pada Sistem Tracking Autonomous Car," [Self Driving Design with PD Control on Autonomous Car Tracking System] (in Indonesian) *Transient J. Ilm. Tek. Elektro*, vol. 5, no. 2, pp. 173–179, 2016, [Online]. Available: <https://ejournal3.undip.ac.id/index.php/transient/article/view/13717>.
- [4] D. Neven, B. De Brabandere, S. Georgoulis, M. Proesmans, and L. Van Gool, "Towards End-to-End Lane Detection: An Instance Segmentation Approach," *IEEE Intell. Veh. Symp. Proc.*, vol. 2018-June, pp. 286–291, 2018, doi: 10.1109/IVS.2018.8500547.
- [5] J. He, H. Rong, J. Gong, and W. Huang, "A lane detection method for lane departure warning system," *Proc. - 2010 Int. Conf. Optoelectron. Image Process. ICOIP 2010*, vol. 1, pp. 28–31, 2010, doi: 10.1109/ICOIP.2010.307.
- [6] S. Zhou and K. Iagnemma, "Self-supervised learning method for unstructured road detection using fuzzy support vector machines," *IEEE/RSJ 2010 Int. Conf. Intell. Robot. Syst. IROS 2010 - Conf. Proc.*, pp. 1183–1189, 2010, doi: 10.1109/IROS.2010.5650300.
- [7] B. Liu, H. Liu, and J. Yuan, "Lane Line Detection based on Mask R-CNN," vol. 87, no. Icmcit, pp. 696–699, 2019, doi: 10.2991/icmicit-19.2019.111.
- [8] J. Kim and M. Lee, "Robust lane detection based on

- convolutional neural network and random sample consensus,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 8834, pp. 454–461, 2014, doi: 10.1007/978-3-319-12637-1_57.
- [9] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning (Adaptive Computation and Machine Learning Series)*. The IMT Press, 2016.
- [10] S. Ren, K. He, R. Girshick, X. Zhang, and J. Sun, “Object detection networks on convolutional feature maps,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 7, pp. 1476–1481, 2017, doi: 10.1109/TPAMI.2016.2601099.
- [11] D. Breach, “Convolution Neural Network Deep Learning.” <https://developersbreach.com/convolution-neural-network-deep-learning/> (accessed Jan. 19, 2021).
- [12] A. L. Katole, K. P. Yellapragada, A. K. Bedi, S. S. Kalra, and M. S. Chaitanya, “Hierarchical Deep Learning Architecture For 10k Objects Classification,” *Comput. Sci. Inf. Technol. (CS IT)*, pp. 77–93, 2015, doi: 10.5121/csit.2015.51408.
- [13] J. Kim, O. Sangjun, Y. Kim, and M. Lee, “Convolutional Neural Network with Biologically Inspired Retinal Structure,” *Procedia Comput. Sci.*, vol. 88, pp. 145–154, 2016, doi: 10.1016/j.procs.2016.07.418.
- [14] O. Nurima Putri, “Implementasi Metode CNN dalam Klasifikasi Gambar Jamur pada Analisis Image Processing (Studi Kasus: Gambar Jamur dengan Genus Agaricus dan Amanita),” [Implementation of CNN Method in Image Classification of Mushroom in Image Processing Analysis (Case Study: Mushroom Image with Genus Agaricus and Amanita)] (in Indonesian), unpublished.
- [15] M. R. Alhamdi, “Deteksi Jalan di sekitar Autonomous Electric Vehicle Berbasis Algoritma Convolutional Neural Network (CNN),” [Road Detection around Autonomous Electric Vehicle Based on Convolutional Neural Network Algorithm] (in Indonesian), 2020.