Steering Angle Prediction for Autonomous Vehicles Using Deep Transfer Learning

Hoang Tran Ngoc *, Phuc Phan Hong, Anh Nguyen Quoc, and Luyl-Da Quach

Department of Information Technology, FPT University, Can Tho, VietNam
Email: PhucPHCE171166@fpt.edu.vn (P.P.H.); Hoang2531992@gmail.com (H.T.N.); AnhNQCE170483@fpt.edu.vn (A.N.Q.); Luyldaquach@gmail.com (L.D.Q)
*Corresponding author

Abstract—**Self-driving cars are anticipated to revolutionize future transportation due to their reliability, safety, and continuous learning capabilities. Researchers are actively engaged in developing autonomous driving systems, employing techniques like behavioral cloning and reinforcement learning. This study introduces a distinctive perspective by employing an end-to-end approach, using camera inputs to predict steering angles based on a model learned from human driving expertise. The model demonstrates rapid training and achieves over 90.1% accuracy in Mean Percentage of Prediction (MPP). In this context, the study aims to replicate driver behavior by applying transfer learning from a pre-trained VGG19 model with various activation functions. The proposed model is trained to analyze road images as input, predicting optimal steering adjustments. Evaluation encompasses a dataset from the ROS2 simulation environment, comparing results with several Convolutional Neural Networks (CNNs) models including Nvidia's, MobileNet-V2, ResNet50, VGG16, and VGG19. The impact of activation functions like Exponential Linear Unit (ELU), Rectified Linear Unit (ReLU), and Leaky ReLU on the transfer learning model is also explored. This research contributes to advancing autonomous driving systems by addressing real-world driving complexities and facilitating their integration into everyday transportation. The novel approach of utilizing transfer learning and comprehensive evaluation underscores its significance in optimizing self-driving technology.**

Keywords—**autonomous vehicle, residual net, MobileNetv2, VGG16, VGG19, Convolutional Neural Networks (CNNs), activation function**

I. Introduction

Autonomous vehicles are created as a solution to the significant number of vehicle accidents resulting from human mistakes, which make up 90% of all accidents [1, 2]. Although significant progress has been made in the field of autonomous vehicles, there is still ongoing research to improve the technology for predicting steering angle control. Self-driving cars have been introduced in several cities, including Mountain View, San Francisco, and Los Angeles. Many car manufacturers have pledged to begin large-scale production to meet the demands of commercial markets, with major companies such as Tesla, Ford, Toyota, General Motors, Google, and Mercedes investing heavily in self-driving cars [3]. The potential success of commercializing autonomous vehicles is immense, as it could lead to the prevention of millions of fatalities. The safety and overall performance of autonomous vehicles are heavily dependent on their ability to accurately predict steering angles.

This involves the use of machine learning algorithms that process road images and adjust steering angles based on road markings, traffic signs, and other objects on the road to avoid obstacles [4–11]. Using supervised learning training data and input from image pixels, an artificial neural network model is trained to predict steering angles in autonomous driving, enabling autonomous steering angle predictions without human intervention [12]. In recent years, Convolutional Neural Networks (CNNs) can be used to process vectorized images for autonomous vehicle systems, and with the advancement of computational power, CNNs have shown promising results in image classification [13]. In addition to cameras, other sensors such as lidar, radar, proximity sensors, and infrared cameras are also used. The expected output from the CNN is the steering wheel angle. Therefore, accurate steering angle prediction is a critical component in autonomous vehicle control systems, as it directly affects the vehicle's ability to navigate and respond to complex driving scenarios. By improving the accuracy of steering angle prediction, we can enhance the vehicle's ability to stay within the designated lane, negotiate curves, and avoid obstacles, thereby reducing the risk of accidents.

This research addresses the need for efficient training processes in the development of autonomous vehicles. The utilization of transfer learning techniques derived from the image network classification challenge allows us to leverage pre-existing knowledge and models, reducing the reliance on large amounts of labeled training data and significantly improving training efficiency. By incorporating transformation learning techniques within a ROS2 simulation environment, our objective is to enhance the accuracy and efficiency of the training process using a transfer deep learning model. The aim of this paper is to create a model that can predict the appropriate steering angle for a lane-keeping vehicle based on a given road image. By comparing the performance of our model with

Manuscript received July 31, 2023; revised August 14, 2023; accepted September 11, 2023; published January 25, 2024.

others and incorporating multiple activation functions, we can demonstrate the optimality of our proposed approach. The successful implementation of an effective steering angle prediction model based on deep transfer learning has the potential to contribute to the widespread adoption of autonomous vehicles. This can revolutionize transportation systems, leading to enhanced safety and potentially saving countless lives by minimizing human errors on the road.

The remainder of this paper is structured as follows: Section II provides a comprehensive examination of the related research. Section III and Section IV describe the data processing and proposed transfer deep learning model for steering angle prediction, including the data collection, model architecture, and evaluation metrics. Section V presents the experimental system and results along with a discussion. Finally, Section VI concludes the paper.

II. Related Work

Pomerleau *et al.* [14] credited with pioneering research on autonomous vehicle navigation through the development of the Autonomous Land Vehicle in a Neural Network (ALVINN). This fully-connected neural network, while basic compared to modern models, provided a foundation for end-to-end autonomous navigation. Although the model was only capable of handling simple scenarios with few obstacles, it was a significant advancement in self-driving car technology.

Research conducted by Testa *et al.* [15] at Nvidia centered on the creation of an autonomous vehicle through end-to-end learning. The approach that is taken involves training a Convolutional Neural Network (CNN) to directly correlate raw pixel data from a single front-facing camera to steering commands. The team’s findings revealed that this technique was remarkably precise, as the model learned how to drive on local roads with or without lane markings and highways with minimal human involvement. A few further studies were performed on the basic CNN network as shown in [16, 17], they used the LeNet network architecture, which complements the activation function and some convolution and fully connected layers to optimize performance.

Recently, many deep learning models have been born and applied to replace the basic CNN to increase the accuracy of the predicted angle. To minimize the mean square error between the true and predicted steering angles, Gupta *et al.* [18] employed the MobileNetV2 model for lane-keeping. Du and Gao *et al.* [19] utilized transfer learning for dataset learning and implemented convolution layer modeling within the Long Short-Term Memory (LSTM) recurrent layer to develop their model. Several studies have utilized Recurrent Neural Networks (RNNs) [20] to predict the next likely action of a pedestrian or driver based on road images captured at different time intervals, in order to navigate the vehicle in the correct direction. Some of these studies have used RNNs such as Long Short-Term Memory (LSTM) to predict steering angles based on historical driving data [21] or to predict trajectories based on past spatial-temporal features [22, 23]. Oussama and Mohamed [24] found that incorporating event cameras with ResNet 50 architecture can result in improved steering angle prediction. Meanwhile, Oussama *et al.* [25] developed two models, a CNN based on the VGG16 architecture as the transfer deep learning model, to address the steering angle problem. Reinforcement Learning (RL) involves the autonomous vehicle functioning as an intelligent agent that strives to maximize cumulative rewards by making correct driving decisions. Researchers of Refs. [26–29] have utilized RL for prediction purposes. However, one of the main obstacles to using RL is its difficulty in solving problems that have sparse reward signals, as noted in [30]. Table I summarizes the existing methods with their limitations.

TABLE I. Steering Angle Prediction Approaches

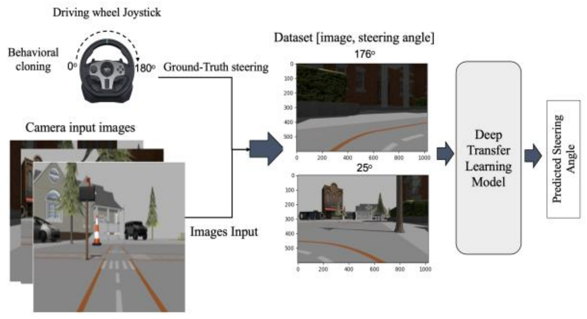
Approach	Method	Limitation
Pomerleau *et al.* [14]	Pomerleau’s study focused on developing a neural network (NN) model to process input data from sensors and navigate the autonomous vehicle based on that.	The model’s capability was limited to handling simple situations with minimal obstacles. This constraint restricts its applicability in complex and diverse traffic scenarios.
Testa *et al.* [15]	The methodology employed in this study utilized the CNN’s ability to extract relevant features from the raw pixel data, enabling the model to effectively learn the mapping between visual input and corresponding steering commands.	The CNN-based end-to-end learning method heavily relies on the quality and diversity of the training data. Variations in lighting conditions, weather, road types, and traffic situations may pose challenges to the model’s generalization capability.
Gupta *et al.* [18]	The approach taken in the study is to utilize the MobileNetV2 model for lane-keeping, which involves predicting steering angles based on the input data. This approach aims to minimize the mean square error between the predicted and true steering angles.	The study’s limitation is its narrow focus on lane keeping and the relatively low accuracy of the estimated steering angle. It does not account for more complex driving situations and unpredictable obstacles.
Gao *et al.* [21]	Transfer learning was employed to develop their model, where convolution layers were incorporated within the LSTM recurrent layer. RNNs, specifically LSTM, were utilized to predict pedestrian/driver actions and steering angles based on road images.	The study’s main limitation is its focus on a specific environment, limiting the applicability of the model to diverse and rapidly changing traffic situations.
Oussama *et al.* [24] and Sokipriala [25]	These two deep convolutional network models, ResNet50 and VGG16, have been proposed for steering angle estimation based on the transfer deep learning model.	These methods have not achieved high accuracy, which is crucial for ensuring traffic safety. The simulation system also does not consider the dynamics of the vehicle model.
Abdur *et al.* [26–28]	Reinforcement Learning (RL) involves the functioning of the autonomous vehicle as an intelligent agent, aiming to maximize cumulative rewards through the execution of correct driving decisions.	The main limitation of RL is the difficulty it faces in solving problems characterized by sparse reward signals.

To address these limitations, we recommend employing the enhanced VGG19 model to construct a predictive steering angle model for autonomous vehicles. Image data will be gathered and augmented to diversify the dataset, facilitating the resolution of various scenarios. Leveraging the improved VGG19 model, characterized by high accuracy, we will identify a suitable activation function for the steering angle prediction task. We will then use the transfer learning method, where the VGG19 model will be trained on the large dataset first and save the weights, and then we will train on our dataset. After training to increase the accuracy, the model will be applied to the donkey car in the ROS2 simulation environment to evaluate the accuracy. In addition, we will also compare with the methods just listed above such as CNN of Nvidia, MobileNet-V2, ResNet50, and VGG16 with various Activation Functions.

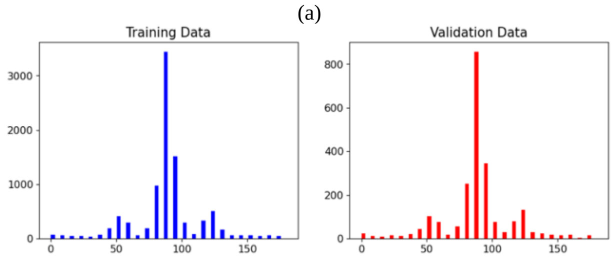
III. DATA PROCESSING

A. Data Collection

The Donkey self-driving car is built in the Gazebo-ROS2 3D simulation environment. We employ a driving wheel joystick to keep the lane and operate the car, which has a steering angle limit of 0° to 180°. The car has a front-facing camera that captures images, and the ROS2 controller records the joystick's steering angle and the camera's images as training data at a rate of 30 frames per second. In this study, the dataset includes 11,000 images that continue to be collected along with the steering angle from human perception (https://www.kaggle.com/datasets/ngochoangtran1992/steering-angle-prediction). Fig. 1 shows the method of collection and distribution of the dataset for the steering angle prediction (image_size=1024×600).



(a)



(b)

Fig. 1. The method of collection (a) and distribution (b) of the dataset for the steering angle prediction with 9,000 training images and 2,000 testing images.

The utilization of behavioral cloning was necessary because it is not feasible to manually encode every conceivable driving scenario. Consequently, we resorted to employing behavioral cloning which involves replicating the driving behavior of a human driver using a Gazebo/ROS2 donkey car simulator to gather training data. The aim of this approach is to facilitate the learning of our model in terms of how to drive autonomously.

B. Data Augmentation and Normalization

1) Randomly zoom

Zoom is a data augmentation technique used in machine learning to increase the amount of training data available for a model. We applied this method to randomly zoom images from 100% to 125% in our dataset. Fig. 2 shows the image with the zoom technique.

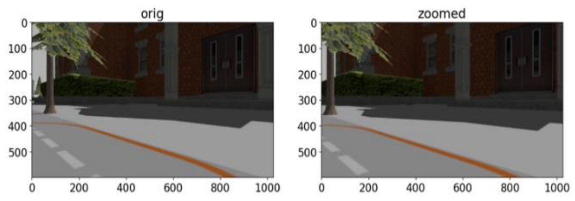


Fig. 2. Original and Zoomed image.

2) Randomly pan

Pan is a technique to shift the image along the x and y axes with random displacement values. The images in our dataset will then be randomly selected and shifted left, right, up, and down a random value. Fig. 3 shows the Pan technique.

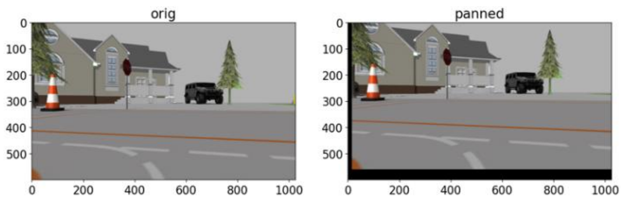


Fig. 3. Original and Panned image.

3) Randomly adjust brightness

To ensure that the vehicle can steer properly in adverse conditions such as poor weather or shadows that were not present in the training dataset, Fig. 4 shows the brightness technique.

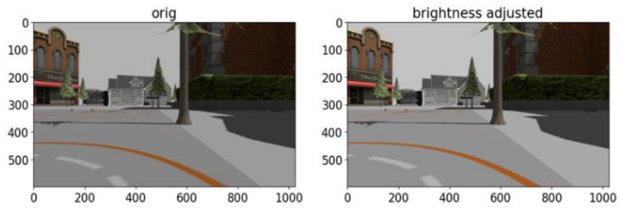


Fig. 4. Original and Brightness adjusted image.

4) Randomly blur

It involves applying a blur filter to an image, which can help to reduce noise and smooth out any irregularities or sharp edges. This technique can help to create new training data that is more robust and better able to handle noise or other types of distortion that may be present in real-world situations. Fig. 5 shows the Blur technique.

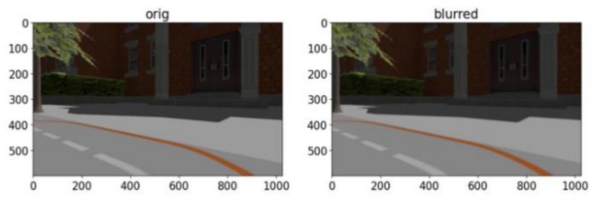


Fig. 5. Original and Blurred image.

5) *Randomly flip*

It involves flipping an image horizontally or vertically to create a mirror image. This technique can help to create more training data and improve the performance of a model by reducing the effects of bias caused by the orientation of objects in the images. In this paper, we randomly flipped the photo horizontally. The angle value after flipping will be calculated according to the following Eq. (1). Fig. 6 shows the Flip technique.

$$Steering\ Angle_{Flipped} = 180^0 - Steering\ Angle \quad (1)$$

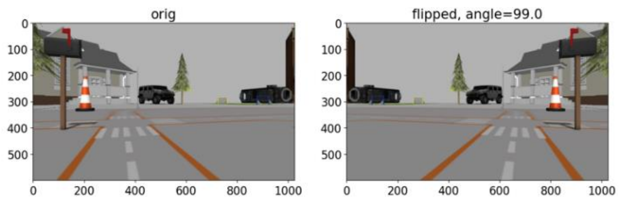


Fig. 6. Original and Flipped image.

6) *Normalization*

Normalization is a data preprocessing technique that is commonly used before training a machine learning model. This is done to ensure that each input feature contributes equally to the model and to prevent any one feature from dominating the others. We have utilized a technique that involves converting the data from RGB to YUV color space, and then resizing the images to a standardized size of 224×224. According to the results of Ref. [15], the YUV color space gives more accurate predictive results. Fig. 7 shows the Normalization technique. Fig. 8 shows the data statistics after the data augmentation process. Steering angle data is more evenly distributed before augmentation data.

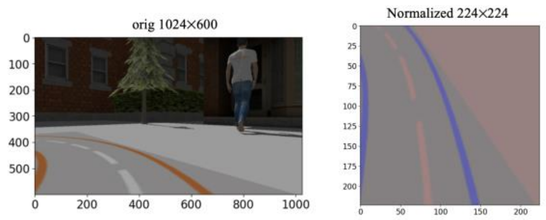


Fig. 7. Original and Normalized image.

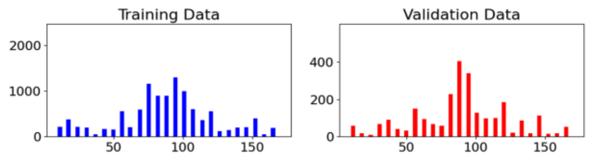


Fig. 8. Data statistics after the data augmentation process.

IV. Transfer Deep Learning Models

A. *Steering Angle Prediction System Architecture Based on Deep Transfer Learning VGG19 Model*

Our proposed model for steering angle estimation is based on the VGG19 CNN architecture [31–34]. It is an extension of the VGG16 model, incorporating three additional convolutional layers at the end. The final classification layer of the pre-trained model was replaced with a series of dense layers, including one Flatten layer, one Dropout layer, and three Dense layers, as illustrated in Fig. 9. The architecture and parameters of our steering angle prediction model, which utilizes transfer learning with VGG19, are depicted in Fig. 10. To improve the accuracy of our approach, we employ transfer learning by leveraging pre-trained weights from models trained on related tasks, such as the ImageNet challenge for image recognition. By utilizing these pre-trained weights, our model benefits from important learned features from a large dataset. During training, the focus is primarily on training the dense layers, while keeping the pre-trained weights fixed.

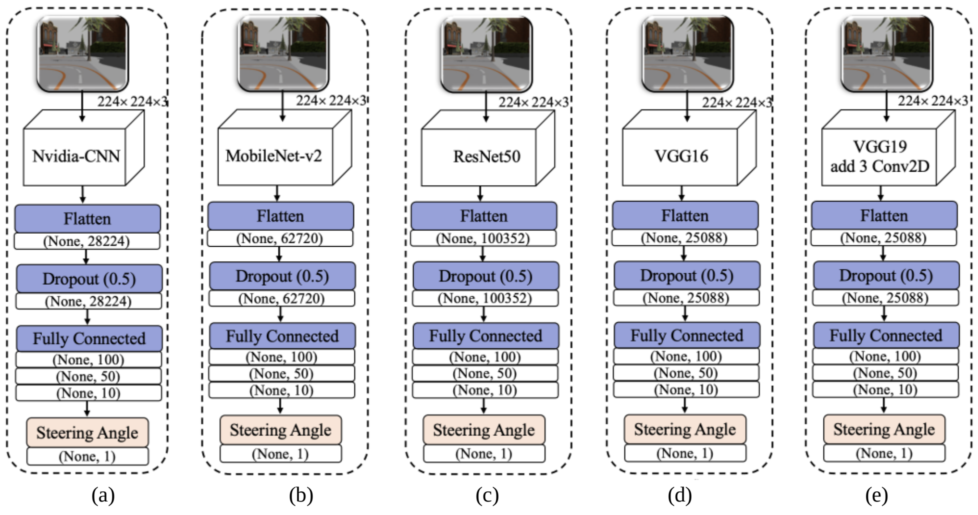
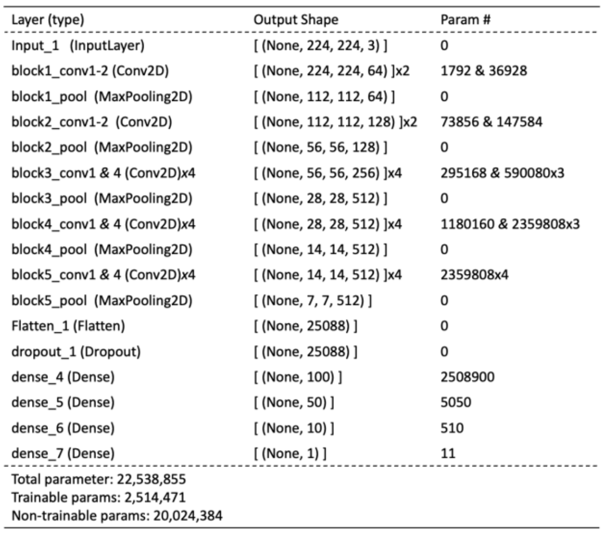


Fig. 9. The structure of the proposed fully connected layer in our model incorporates. (a) Nvidia-CNN, (b) MobileNet-v2, (c) ResNet50, (d) VGG16, and (e) VGG19 models.



Layer (type)	Output Shape	Param #
Input_1 (InputLayer)	[(None, 224, 224, 3)]	0
block1_conv1-2 (Conv2D)	[(None, 224, 224, 64)]x2	1792 & 36928
block1_pool (MaxPooling2D)	[(None, 112, 112, 64)]	0
block2_conv1-2 (Conv2D)	[(None, 112, 112, 128)]x2	73856 & 147584
block2_pool (MaxPooling2D)	[(None, 56, 56, 128)]	0
block3_conv1 & 4 (Conv2D)x4	[(None, 56, 56, 256)]x4	295168 & 590080x3
block3_pool (MaxPooling2D)	[(None, 28, 28, 512)]	0
block4_conv1 & 4 (Conv2D)x4	[(None, 28, 28, 512)]x4	1180160 & 2359808x3
block4_pool (MaxPooling2D)	[(None, 14, 14, 512)]	0
block5_conv1 & 4 (Conv2D)x4	[(None, 14, 14, 512)]x4	2359808x4
block5_pool (MaxPooling2D)	[(None, 7, 7, 512)]	0
Flatten_1 (Flatten)	[(None, 25088)]	0
dropout_1 (Dropout)	[(None, 25088)]	0
dense_4 (Dense)	[(None, 100)]	2508900
dense_5 (Dense)	[(None, 50)]	5050
dense_6 (Dense)	[(None, 10)]	510
dense_7 (Dense)	[(None, 1)]	11

Total parameter: 22,538,855
Trainable params: 2,514,471
Non-trainable params: 20,024,384

Fig. 10. Steering angle prediction model architecture based on VGG19.

Specifically, the model will be presented as follows, we will first initialize the VGG19 model with an input shape of (224, 224, 3) and use pre-trained weights (weights=ImageNet). Next, we freeze the layers of the base model to prevent them from being trained. Then, we add a new fully connected output layer with a single neuron (Predicted steering angle). This process involves taking the output of the base model and passing it through transformation layers. Specifically, we use the Flatten layer to flatten the output, the Dropout layer to randomly drop some connections to prevent overfitting, and the Dense layers to create fully connected layers with the corresponding number of neurons. Finally, we use the Adam optimizer with a learning rate of 0.001 to update the weights during training. The model is compiled with a Mean Squared Error (MSE) loss function and the accuracy metric. The backend math in the VGG19 algorithm involves specific mathematical operations such as convolution, pooling, activation functions, flattening, dropout, and computations related to fully connected layers (Dense). These operations are performed within the layers and optimized using the Adam optimization algorithm.

By leveraging the knowledge gained from pre-trained models, our approach enables faster learning and higher accuracy without the need for extensive training time and resources from scratch. To evaluate the performance of our method, we combine transfer learning models, including Nvidia-CNN, MobileNet-v2, ResNet50, VGG16, and VGG19, with the newly designed fully connected layers. Table II provides details of the redesigned layers and the number of parameters in each network.

TABLE II. PARAMETERS OF MODELS

Network	Depth	Parameters (Millions)	Trainable params (Millions)	Input Size
Nvidia-CNN	13	2.9	2.9	224×224
MobileNet-V2	159	8.5	6.2	224×224
ResNet50	56	33.6	10	224×224
VGG16	22	17.2	2.5	224×224
VGG19	25	22.5	2.5	224×224

B. Testing with Activation Functions

We use the three most common activation functions, Exponential Linear Unit (ELU), Rectified Linear Unit (ReLU), and Leak ReLU to evaluate the accuracy of the above transfer learning network models. ELU, ReLU, and Leaky ReLU have commonly used activation functions in deep learning. ELU has faster learning than traditional activation functions, can prevent vanishing gradient problems, and produce negative output values. However, it is computationally more expensive than ReLU and may produce NaN values. ReLU is computationally efficient and works well in deep neural networks. However, it may cause the dying ReLU problem and only produces non-negative output values. Leaky ReLU addresses the dying ReLU problem, and is computationally efficient, but may produce inconsistent outputs when the slope value is not optimized and only produces non-negative output values. The equations of the above three functions are shown in Eqs. (2)–(4).

$$ELU = \begin{cases} x\,, if\ x \geq 0 \\ \delta(e^{x} - 1)\,, if\ x < 0 \end{cases} \tag{2}$$

$$ReLU = \begin{cases} 0\,, if\ x < 0 \\ x\,, if\ x \geq 0 \end{cases} \tag{3}$$

$$Leaky\ ReLU = \begin{cases} x\,, if\ x > 0 \\ 0.01x\ otherwise \end{cases} \tag{4}$$

where *x* is the steering angle; δ is a positive constant.

The choice of activation function depends on the specific task and performance requirements. ELU is a good choice when accuracy is crucial. ReLU is efficient and useful for deep networks. Leaky ReLU is ideal for tackling the dying ReLU problem. Ultimately, the chosen activation function must match the task's requirements. The comparison results will be presented in the next section.

C. Evaluation Metrics

1) Loss function

To accurately predict the steering angle, we solved it as a regression problem. To evaluate the difference between the predicted steering angle and the true steering angle, we use the Mean Square Error (MSE) as the *loss function* (5). Since MSE calculates the squared difference, it is highly effective in adapting any variation between the predicted steering angle and the true steering angle.

$$MSE(x, y) = \frac{1}{N}\sum_{i=1}^{N}(x_i - y_i)^2 \tag{5}$$

where x_i and y_i are the predicted and true steering angles

2) Accuracy

The output prediction result of this research is an angular value within the range of 0 to 180 degrees. Therefore, we established an accuracy function to assess the precision of different models. This function determines the percentage of accuracy by measuring the total predicted angular deviation against the total correct angle on each epoch. The accuracy equation is shown in Eq. (6).

$$Accuracy_{epoch} = 100 - \frac{|\sum_{i=1}^{N}(x_i - y_i)|}{\sum_{i=1}^{N}(y_i)} \times 100 \tag{6}$$

where *epoch* is a number of times the entire data is trained. N is the total of trained data.

V. Experimental Results and Discussion

The virtual world is built and designed with Gazebe/ROS2 software as shown in Fig. 11. The simulation world contains the donkey car, a map with two lanes used for training mode and used for testing the self-driving mode. The control of the donkey car is carried out by human expertise via joystick control. The images obtained are then saved and used to train the proposed models. The training was performed using a dataset of 11,000 images on a computer equipped with Ubuntu 20.04, an Intel i7 3.4 GHz CPU, an Nvidia GTX 1650-8GB GPU, and 32 GB RAM. Our algorithm was implemented in Python 3.10, utilizing the Tensorflow 2.12.0 and Keras 2.12.0 libraries.

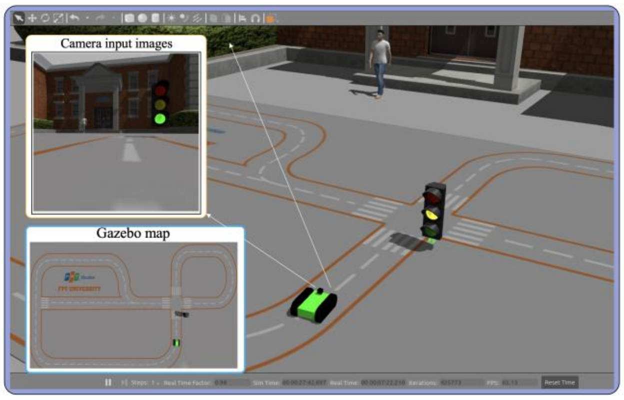


Fig. 11. Virtual world built on Gazebo/ROS2.

Upon completing the construction of our models, we performed an experiment to evaluate the accuracy of our proposed model for predicting steering angle, compared to other existing models. Our training parameter included a learning rate of 0.001 and 20 epochs, using the Adam optimizer for the loss function [19]. The error value was evaluated based on the MSE (5) loss function. In terms of training time, the models underwent 20 epochs with the following durations: Nvidia-CNN took 2,822 s, MobileNetV2 required 2,970 s, ResNet50 took 3,465 s, VGG16 consumed 5,519 s, and the proposed method using VGG19 took the longest time at 6,813 s.

Our comparison results between the proposed VGG19's steering angle prediction ability and other models are illustrated with data augmentation in Figs. 12 and 13. Fig. 12 shows the minimum loss values for each model, where Nvidia-CNN has a minimum loss value of 237.2, MobileNet-v2 has a value of 217.3, ResNet50 has a value of 646.9, VGG16 has a value of 227.2, and our suggested VGG19 had a value of 151.8. In Fig. 13, the corresponding percentage accuracy calculated using the numerical Eq. (6) is presented, where Nvidia-CNN is 84.5%, MobileNet-v2 is 87%, ResNet50 is 74.2%, VGG16 is 85.3%, VGG19 is 90.1%.

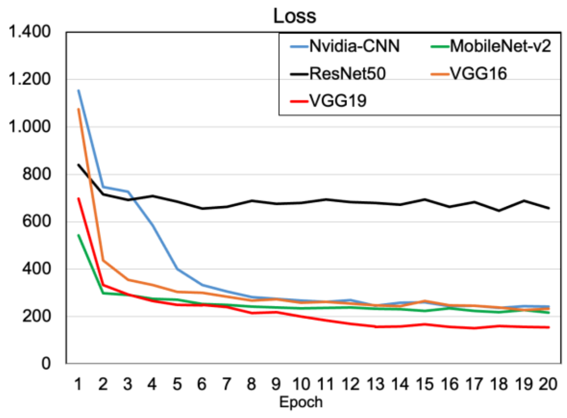


Fig. 12. Comparisons of loss value between Nvidia-CNN, MobileNet-v2 RasNet50, VGG16, and proposed VGG19.

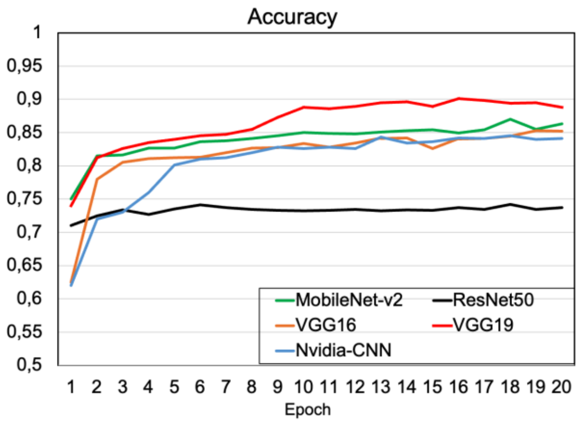


Fig. 13. Comparisons of accuracy value between Nvidia-CNN, MobileNet-v2 RasNet50, VGG16, and proposed VGG19.

The VGG19, being a profoundly deep convolutional neural network, surpasses other models like Nvidia-CNN, ResNet50, MobileNet-v2, and VGG16 in terms of sheer depth. These layers allow VGG19 to apprehend and dissect intricate data patterns and features comprehensively. The model's impressive depth equates to an exceptional capacity for feature extraction, albeit at the cost of increased computational complexity. As demonstrated, when we combined VGG19 with a redesigned fully connected set, the accuracy of our proposed model was higher than other methods. This will enhance the car's ability to adhere to the lane, as it learns from human knowledge in driving the vehicle.

Figs. 14 and 15 show a comparison of the proposed model activity before and after augmentation data, which accounts for 30% of the complete dataset consisting of 11,000. It is observed that after augmentation data during training and validation, the result of Loss is lower. It is noteworthy that the ELU activation function was employed in Figs. 12–15.

The results for the prediction of steering angle using the VGG19 model are presented in Fig. 16. It is evident that the accuracy of the angle prediction is around 90%.

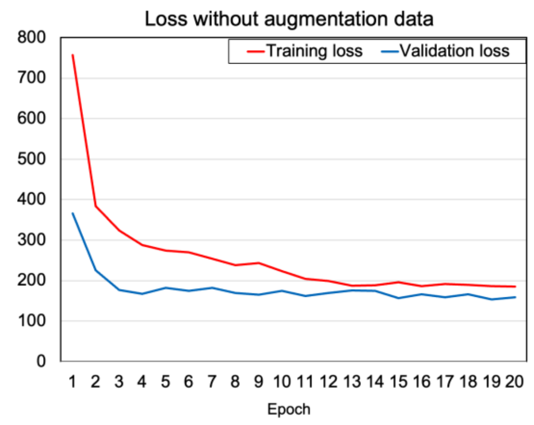


Fig. 14. Training and Validation loss of VGG19 model without augmentation data.

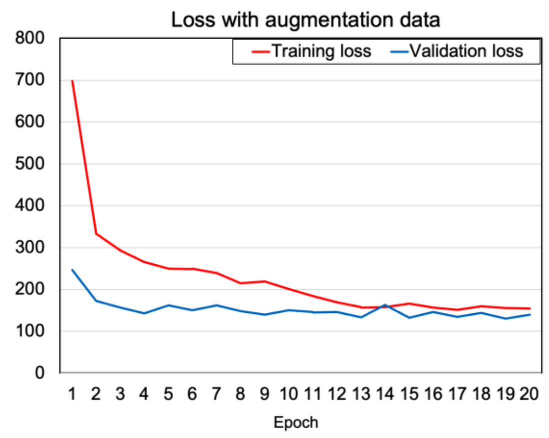


Fig. 15. Training and Validation loss of VGG19 model with augmentation data.

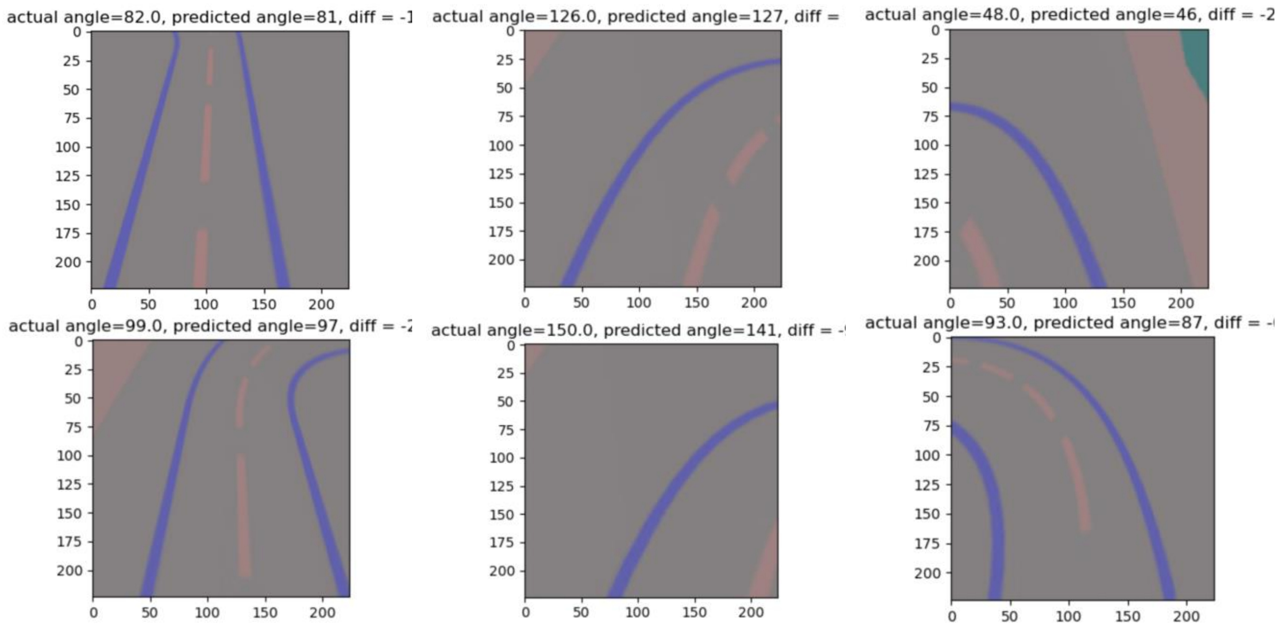

actual angle=82.0, predicted angle=81, diff = -1
actual angle=126.0, predicted angle=127, diff =
actual angle=48.0, predicted angle=46, diff = -2
actual angle=99.0, predicted angle=97, diff = -2
actual angle=150.0, predicted angle=141, diff = -
actual angle=93.0, predicted angle=87, diff = -

Fig. 16. Prediction results of steering angle using the VGG19 model.

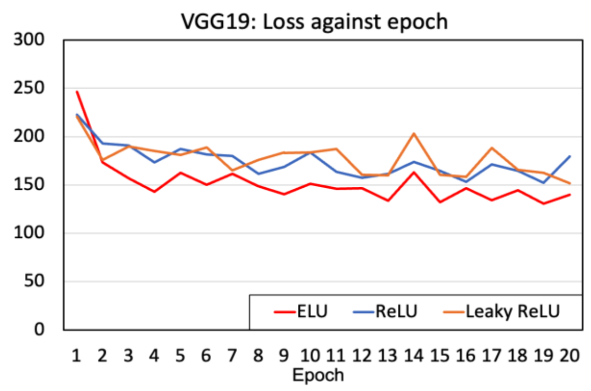


Fig. 17. Validation Loss value of VGG19 for different activation functions.

To identify the most appropriate activation function for the task of predicting the steering angle for lane-following vehicles, we performed tests using ELU, ReLU, and Leaky ReLU functions. The findings are depicted in Fig. 17, which shows the Validation Loss value of VGG19 for different activation functions. The results indicate that the ELU function produced the lowest Validation Loss value of 130.7, whereas ReLU and Leaky ReLU functions resulted in Validation Loss values of 152.2 and 151.57, respectively. Consequently, the ELU function is the most suitable choice for generating accurate predictions. Table III displays the Validation Loss outcomes for the models.

TABLE III. VALIDATION LOSS OF MODELS FOR DIFFERENT ACTIVATION FUNCTIONS

Network	Nvidia-CNN	MobileNet-V2	ResNet 50	VGG 16	VGG 19
ELU	167.5	156.3	635.3	160.4	130.7
ReLU	193.2	172.4	723.5	186.3	152.2
Leaky-ReLU	190.8	170.7	722.3	184.2	151.5

The proposed model's accuracy has been compared, and it has been observed that the accuracy has improved by over 10%.

Additionally, the most appropriate activation function has been identified for predicting the steering angle based on human perception. The vehicle has undergone experimental testing in a simulated environment, and it has been found that it produces the same level of accuracy as predicted. In our future research direction, we intend to explore scenarios where the vehicle needs to estimate steering angles while considering acceleration, deceleration, and avoiding other vehicles on the road. This will involve tackling problems related to lane changes, trajectory estimation, and integrating object detection methods using 3D lidar sensors to determine the positions of objects on the road.

VI. Conclusion

The issue of predicting the steering angle of autonomous vehicles has always been a fascinating subject and has attracted a lot of research interest. One of the main challenges is training deep learning models to accurately predict the steering angle in various traffic conditions. Developing an effective model requires more data and training time. To address this challenge, we conducted research on the VGG19 model and reconfigured it by integrating fully connected layers for predicting autonomous driving angles. Our redesigned VGG19 model incorporates three additional layers of CNN, which enhances its accuracy. In order to enhance the performance of the model even further, we implemented the ELU activation function along with the Adam optimization algorithm, which adjusts the learning rate and loss function Mean Squared Error (MSE). We also implemented the dropout function to reduce the number of duplicate parameters, thus avoiding overfitting. Furthermore, we utilized image augmentation techniques, which involved generating more images for the training dataset. This approach made the model more versatile and resulted in better prediction results when compared to the previous model. However, we acknowledge that our paper is limited by the fact that the predicted steering angle relies on individual images, while in actual driving scenarios, images in a sequence from the camera are interrelated. Recognizing this shortcoming, we intend to conduct further research in the future to incorporate models capable of capturing temporal dependencies, resulting in more accurate predictions.

Conflict of Interest

The authors declare no conflict of interest.

Author Contributions

Hoang Tran Ngoc and Luyl-Da Quach had the idea for the research and provided support for Phuc Phan Hong and Anh Nguyen Quoc. Phuc Phan Hong conducted the research, while Anh Nguyen Quoc wrote the paper and prepared the dataset; all authors approved the final version.

References

[1] Road Traffic Injuries. [Online]. Available: https://www.who.int/news-room/fact-sheets/detail/road-traffic-injuries

[2] M. G. Usman, C. Haruna, A. Nahla, A. Saidu, S. I. Popoola, and A. A.-G. Mohammed, "A survey on deep learning for steering angle prediction in autonomous vehicles," *IEEE Access,* vol. 8, pp. 163797–163817, 2020.

[3] K. J. Aditya, "Working model of self-driving car using convolutional neural network, raspberry Pi and arduino," in *Proc. International Conference on Electronics, Communication and Aerospace Technology (ICECA 2018*), 2018, vol. 66.

[4] S. Vaibhav, G. Snehal, A. Rohit, D. Pritam, and K. Urmila, "Steering angle prediction in autonomous vehicles using deep learning," in *Proc. 2019 5th International Conference on Computing, Communication, Control and Automation,* Pune, 2019, pp. 163797–163817.

[5] H. N. Tran and L. Quach, "Adaptive lane keeping assist for an autonomous vehicle based on steering fuzzy-PID control in ROS," *International Journal of Advanced Computer Science and Applications*, vol. 13, 2022.

[6] V. D. Nguyen, T. D. Trinh, and H. N. Tran, "A robust triangular sigmoid pattern-based obstacle detection algorithm in resource-limited devices," *IEEE Transactions on Intelligent Transportation Systems,* vol. 24, no. 6, pp. 5936–5945, June 2023.

[7] P. H. Phan, A. Q. Nguyen, L. Quach, and H. N. Tran, "Robust autonomous driving control using auto-encoder and end-to-end deep learning under rainy conditions," in *Proc. the 2023 8th International Conference on Intelligent Information Technology (ICIIT '23),* 2023, pp. 271–278.

[8] H. K. Hua, K. H. N., L. Quach, and H. N. Tran, "Traffic lights detection and recognition method using deep learning with improved YOLOv5 for autonomous vehicle in ROS2," in *Proc. 2023 8th International Conference on Intelligent Information Technology (ICIIT '23). Association for Computing Machinery,* 2023, pp. 117–122.

[9] H. T. Vo, H. N. Tran, and L. Quach, "An approach to hyperparameter tuning in transfer learning for driver drowsiness detection based on bayesian optimization and random search," *International Journal of Advanced Computer Science and Applications(IJACSA),* vol. 14, no. 4, 2023.

[10] H. N. Tran, H. V. N. Nguyen, K. H. Nguyen, and L. D. Quach, "Lane road segmentation based on improved UNet architecture for autonomous driving," *International Journal of Advanced Computer Science and Applications(IJACSA),* vol. 14, no. 7, 2023.

[11] H. N. Tran, K. H. Nguyen, H. K. Hua, H. V. N. Nguyen, and L. D. Quach, "Optimizing YOLO performance for traffic light detection and end-to-end steering control for autonomous vehicles in gazebo-ROS2," *International Journal of Advanced Computer Science and Applications(IJACSA),* vol. 14, no. 7, 2023.

[12] X. Galorot and Y. Bengio, "Understanding difficulty of training feedforward neural networks," in *Proc. AISTATS*, 2010, vol. IX, pp. 249–256.

[13] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Communications of the ACM*, vol. I, no. 60, pp. 84–90, 2012.

[14] D. A. Pomerleau, "ALVINN: An autonomous land vehicle in a neural network," in *Proc. (NeurIPS) Neural Information Processing Systems*, December 1989, pp. 305–313.

[15] M. Bojarski, D. W. Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. J. Muller, X. Zhang, J. Zhao, and K. Zieba, "End to end learning for self-driving cars," arXiv preprint, arXiv:1604.07316, 2016.

[16] V. Rausch, A. Hansen, E. Solowjow, C. Liu, and E. Kreuzer, "Learning a deep neural net policy for end-to-end control of autonomous vehicles," in *Proc. American Control Conf. (ACC)* 2017, pp. 4914–4919,.

[17] R. Hazra, A. Kumar, and B. Baranidharan, "Effect of various activation function on steering angle prediction in CNN based autonomous vehicle system," *Int. Journal of Engineering and Advanced Technology (IJEAT)*, vol. 9, pp. 2249–8958, December 2019.

[18] M. Gupta, V. Upadhyay, P. Kumar, and F. Al-Ṭuman, "Deep learning implementation of autonomous driving using ensemble-M in simulated environment," *Research Square*, vol. 25, 2021.

[19] S. Du, H. Guo, and A. Simpson, "Self-driving car steering angle prediction based on image recognition," arXiv preprint, arXiv:1912.05440, 2019.

[20] L. Aoxue, J. Haobin, Z. Jie and Z. Xinchen, "Implementation of human-like driver model based on recurrent neural networks," *IEEE Access*, vol. 7, pp. 98094–98106, 2019.

[21] B. Abdelmoudjib, J. Vincent, C. Maaoui and B. Moussa, "Long-term prediction of vehicle trajectory using recurrent neural networks," in *Proc. IECON 2019 45th Annual Conference of the IEEE Industrial Electronics Society*, Lisbon, Portugal, 2019, vol. 7.

[22] V. Rodolfo, Z. Mahdi, O. Sedat, and P. F. Yaser, "Controlling steering angle for cooperative self-driving vehicles utilizing CNN and LSTM-based deep networks," in *Proc. 2019 IEEE Intelligent Vehicles Symposium (IV)*, 2019, vol. 1.

[23] J. Yonghwan, K. Seonwook, and Y. Kyongsu, "Surround vehicle motion prediction using LSTM-RNN for motion planning of autonomous vehicles at multi-lane turn intersections," *IEEE Access*, vol. 1, pp. 2–14, 2020.

[24] A. Oussama and T. Mohamed, "A literature review of steering angle prediction algorithms for self-driving cars," in *Proc. Int. Conf. on Advanced Intelligent Systems for Sustainable Development*, Feb. 2020, vol 1105.

[25] J. Sokipriala, "Prediction of steering angle for autonomous vehiclesusing pre-trained neural network," *European Journal of Engineering and Technology Research*, vol. 6, no. 5, 2021.

[26] R. Abdur, H. Sabir, O. Doukhi, and L. Deok-Jin, "Driverless car: Autonomous driving using deep reinforcement learning in urban environment," in *Proc. 2018 15th International Conference on Ubiquitous Robots (UR)*, Hawaii, 2018.

[27] K. Güçkıran and B. Bolat, "Autonomous car racing in simulation environment using deep reinforcement learning," in *Proc. 2019 Innovations in Intelligent Systems and Applications Conference (ASYU)*, Izmir, Turkey, 2019.

[28] S. Mo, X. Pei, and C. Wu, "Safe reinforcement learning for autonomous vehicle using monte carlo tree search," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 7, pp. 6766–6773, July 2022.

[29] J. Wu, Z. Huang, W. Huang, and C. Lv, "Prioritized experience-based reinforcement learning with human guidance for autonomous driving," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 476, 2022.

[30] J. Faith and D. Kristin, "Feudal steering: Hierarchical learning for steering angle prediction," in *Proc. 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2020, pp. 1–10.

[31] L. Wen, X. Li, X. Li, and L. Gao, "A new transfer learning based on VGG-19 network for fault diagnosis," in *Proc. 2019 IEEE 23rd International Conference on Computer Supported Cooperative Work in Design (CSCWD)*, Porto, Portugal, 2019, pp. 205–209.

[32] S. Kavitha, B. Dhanapriya, G. N. Vignesh, and K. R. Baskaran, "Neural style transfer using vgg19 and alexnet," in *Proc. 2021 International Conference on Advancements in Electrical, Electronics, Communication, Computing, and Automation (ICAECA)*, Coimbatore, 2021, pp. 1–6.

[33] S. Namani, L. S. Akkapeddi, and S. Bantu, "Performance analysis of vgg-19 deep Learning model for COVID-19 detection," in *Proc. 2022 9th International Conference on Computing for Sustainable Global Development (INDIACom)*, 2022, pp. 781−787.

[34] M. S. M. Khan, M. Ahmed, R. Z. Rasel, and M. M. Khan, "Cataract detection using convolutional neural network with vgg-19 model," in *Proc. 2021 IEEE World AI IoT Congress (AIIoT)*, 2021, pp. 0209–0212.