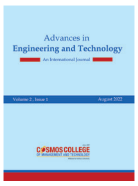
Advances in Engineering and Technology
An International Journal



Implementing Autonomous Driving System in a Virtual Environment

Chitran Pokhrel,[*] Aayush Khatiwada

Department of Computer Engineering, Advanced College of Engineering and Management, Nepal

Abstract

The automation in the current times has become the growing hot topic for discussion around the globe. The growth of the machine learning and the development of new techniques have made the automation one of the major topics of research among the scientist around the world. It might be too costly to observe the accuracy and precision of the algorithm in the real world; hence the virtual environments are used to test the algorithms before the real world tests with the technique are done. This work focuses on one of the various ways of developing the autonomous driving system using Convolutional Neural Networks (CNN) in a gaming environment.

Keywords: AlexNet, ANN, Back-propagation, CNN, OpenCV, Q-learning, Reinforcement learning, ROI

1. Introduction

The automation in the current time is one of the major topics of focus. The growth of the AI has leaded the automation to dominate every single sector of the industrial world. The use of AI has made it easy to perform the task considered as risky for the humans. With the growth of research in the fields of machine learning, it has become more evident that the world is changing from a manual system to an autonomous system.

The approach towards automation has now made its move for the work that was considered to be only done by humans, such as flying aircrafts, surgery, painting, etc. The necessity of automating for works that a human can do good comes out of the necessity for the accuracy and factors that affects the work done by human. It is true that a machine can work more efficiently and accurately in several cases compared to a human. The plus point in having a machine work as effectively as human actually reduces the amount of time at which the works could be done [1–3]. It also address to the human concerns of risk. A human suffers from emotions, fatigue and stress that directly impact on their capabilities of doing work and sometimes cause accidents which can be lethal. Looking at the possible risk factors, accuracy and effectiveness, today the industrial world prefer using machines [4]. The Autonomous Driving System is also being developed because of same reasons.

[*]Corresponding author
Email: chitranpokhrel@gmail.com(C.Pokhrel)

Every year around 1.35 million people lose their lives in road accidents and around 20 to 50 million more people suffer non-fatal injuries but a lot of them cause disabilities. More than 90% of those road deaths occur in low and middle-income countries [5]. Based on a survey conducted in England, among 4600 male drivers it was found out that 9% -10% of road accidents were due to fatigue out of which 20% of them were on motorways [6]. These studies clearly showed the vehicle manufacturers the necessity and the profits they could make by introducing an autonomous driving system in their products. However, building an algorithm that could work on the real world traffic would need lots of effort as well as working on a system in the real world might be costly and any sorts of mistake could lead to fatal injuries.

In this work, a gaming environment was taken to build the autonomous driving system such that the gaming environment can be used to see how the system would react to the real world scenario with the same algorithms. Since it was to mimic the real world scenario the environment of the selected game should be realistic and support open world driving, in this case it was NFS: most wanted. The algorithm and the Model used may not be perfectly fit for a driving system but having a simulated or virtual environment that can mimic the real case can easily show us the areas where we need to work as well as stops any sorts of possibilities of risk and losses that might happen if the experiment is to be done with robots or any kinds of vehicle.

2. Related Works

There are some mathematical, neural networks, and reinforcement learning as a proposed solution for the problem. Out of these, for the gaming environment reinforcement learning and the mathematical approaches are more frequently used.

Mathematical approach needs explicit decomposition of the problem. The technique uses either monocular stereo vision. The approach comes up with the detection of lane marker which requires the baseline to be horizontal i.e. horizon in the frame is parallel to the x-axis [7]. The other approach is reinforcement learning. In this approach, the agent (vehicle) interacts with the environment to reach an optimal state or objective. To interact with the environment the agent performs an action. Based on the action performed in the environment the agent receives rewards. If the action performed by the agent is positive, then the reward is positive, but if the action performed by the agent is negative such as hitting some other vehicle, people or driving through sidewalk the reward is negative [8–11].The learning approach has an environment, with which the agent reacts, states at which the agent might remain, actions performed by the agent which brings changes in the states of an agent. As the agent performs action, the performed action derives some rewards and based on the reward the agent changes its states in the environment [12]. The goal of an agent is to perform right actions to achieve the maximum rewards. However, both the approaches have their cons, the mathematical approaches needs the defined flat surface, focal length and optical center and cannot exactly determine whether it is on the right path in the lane or not whereas with the reinforcement learning it takes several trials and errors for the agent to perform well. The solution purposed in this paper aims to provide a more effective solution to the problem and provide an efficient automation to the driving system without explicit high level mathematical analysis and modeling of the problem, in a virtual environment.

3. Methodology

The method purposed for achieving the efficient autonomous driving system is the use of a Convolu- tional Neural Network (CNN) with lane detection capabilities. For this approach the most important thing is to collect valid training data. It is done by recording frames and saving commands the player gives during the gameplay in which a player drives the vehicle in a virtual environment, in the real world scenarios it would be the recordings of video frames from an onboard camera and steering command. CNNs have revolutionized pattern recognition and they are capable of learning features automatically from training data [13].

3.1 Data collection

Training data were collected manually by driving the vehicle in the virtual environment (game). For collecting data from the gameplay screen, it is important that we could grab the screenplay by defining our Region of Interest (ROI) this was done with OpenCV. Data were collected through one hour of gameplay recording. Since the game environment provides users with its changeable camera angle any kinds of extra shots were unnecessary. In a game environment, it is easy to change the lighting effects and different conditions to mimic the real scenario. The training data collected in manual mode using game camera was stored as pair of values. The first parameter was the command which was being sent to the vehicle (key input for movement of vehicle) and the second parameter was the frame at which those commands were being sent.

3.2 Neural Network

Artificial Neural Networks (ANN) is highly simplified mathematical models of biological neural net- works having ability to learn and provide meaningful solution to the problems with high complexity and non-linearity [14]. It is faster than the conventional techniques available to solve any given problem. The development of ANN is motivated to imitate the human brain [15–17]. Here in this case, the AlexNet has been used to solve our problems.

AlexNet famously won the 2012 ImageNet LSVRC-2012 competition by a large margin 15.3% VS 26.2% (second place) error rates [18–20]. In this competition, data is not a problem; there are

about 1.2 million training images, 50 thousand validation images, and 150 thousand testing images. AlexNet forged a new landscape in the world of computer vision. AlexNet maximizes the multinomial logistic regression objective, which is equivalent to maximizing the average across training cases of the log-probability of the correct label under the prediction distribution [20]. The model used had 5 convolutional layers and 3 fully connected layers. ReLU function is applied after very convolutional and fully connected layer.

The architecture used here consisted of four neurons in output layers representing the possible actions that network can perform (forward right, forward left, stop, forward). The neuron utilizes the ReLU activation function. The number of neurons is determined by the size of the image used as the input for the network. The recorded individual frames were converted into a grayscale format to optimize the algorithm because the color information is unnecessary. Frames were then converted into NumPy arrays so that they could be used as input to the network. These arrays are paired with labels which are the actual user commands during those frames and this paired data is saved as .npy file for training purposes. The weight generated after the training were saved into .xml file for the prediction. The format at which the label values were recorded is shown in table 1.

S.N	Movement	Value format
1	Left	[1,0,0,0]
2	Right	[0,1,0,0]
3	Forward	[0,0,1,0]
4	Stop	[0,0,0,1]

Table 1: Format of Label Values

3.3 Algorithm

3.3.1 Back Propagation

Back-propagation is a method used in Artificial Neural Network (ANN) to calculate a gradient that is needed in the calculation of the weights to be used in the network, commonly used to train deep neural networks, a term referring to neural networks with more than one hidden layer [21]. Back- propagation may be a special case of a way called automatic differentiation. Back-propagation is usually employed by the gradient descent optimization algorithm to regulate the weight of neurons by calculating the gradient of the loss function also called backward propagation of errors, because the error is calculated at the output and distributed back through the network layers [22]. Forward propagation, backpropagation, and weight update are the three consecutive procedures used for neural network training. Each neuron calculates a weighted total of the inputs from the associated neurons in its preceding layer, and then adds them with a bias, as part of the forward propagation process which can be mathematically expressed as the equation: $O^l_j = \sum_{i=1}^{N} w_{ij}{}^l + b^l$.

The ouput goes through an activation function expressed as: ylj =$f(O^l_j)$ which detemines the data to pass through next layer. In this case ReLU has been used as the activation function. Due to its ability to significantly lower calculation costs, the ReLU is very popular in ANNs. When the input is negative, the ReLU propagates zero to the following layer; otherwise, it skips the input value. The weights are adjusted using the backpropagation method by calculating derivatives [23].

Algorithm 1 Algorithm For Back-propagation

```
initialize network weights (often small random values)
do
For Each training example named ex
predict ⟵ neural − net − outputs(network, ex) Forward pass
actual ⟵ teacher − output(ex)
error ⟵ (predict − actual)at output units
For all weights from hidden layer to output layer compute $\Delta w(h)$
Compute $\Delta w(i)$ for all weights from input to hidden layer update network weights
Until all example classified correctly or another stopping criterion satisfied
return network
```

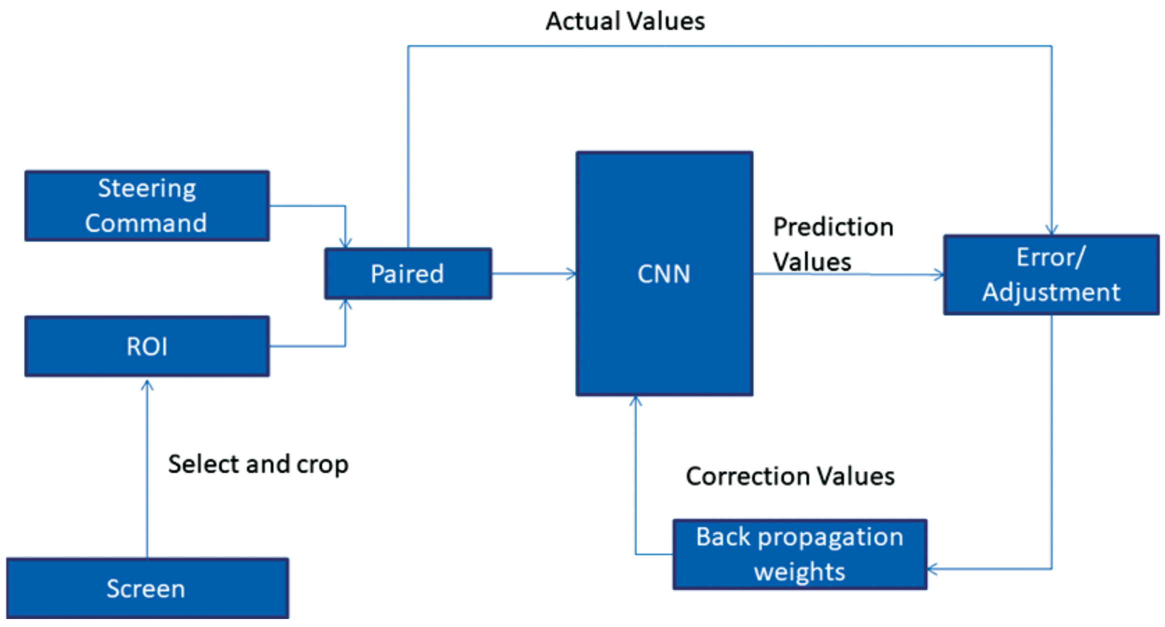


Figure 1: Back propagation with CNN

3.4 Lane Detection

For the gaming environment, the detection of the lane is done through the polygon edge handling and detection technique. Generally, the detection technique uses vehicle as a reference point that we know is or isn't in the polygon, then draw a line between the tested point and this reference point, then count how many times and in which direction it intersects the polygon segments. Mathematical approach uses vehicle as the reference points and through edge detection detects the lane, as the vehicle moves the reference point i.e. vehicle in this case also starts to shift towards one lane and away from another. The main objective of the approach is to keep vehicle in equidistant between lanes. And with the help of trigonometric calculation steering angles could be calculated.

The lane detection for this work was done with the help of openCV libraries. OpenCV provides with different functions that supports complicated image processing. With openCV we can find lane in the given frame. It started with the preprocessing of the image. The image or frame is converted into grayscale, such that the procedures required further could be made easy as grayscale image is better for feature extraction and optimizes the process. We applied Gaussian smoothing of the image to support Canny edge detection. Next, we detected edges for lane detection since the contrast between the lane and the surrounding road surface provides us with useful information on detecting the lane lines. Canny edge detection is an operator that uses the horizontal and vertical gradients of the pixel values of an image to detect edges [24]. This gives us edge detection for the entire vision (entire scenario of the screen) but for the lane detection we need only region that shows us the lane this was achieved by masking other region to define ROI.

We applied Hough Transformation to indicate the lines in the lanes. The Hough transformation converts an "x vs. y" line to a point in "gradient vs. intercept" space. In Hough space, lines will match up with the points in the picture. An intersection of lines in Hough space will thus correspond to a line in Cartesian space [25]. Using this method, we found lines from the pixel outputs of the canny edge detection output. The behavior of the vehicle when it detects the lane is coded within in such a way that it tries keeping the vehicle between those lanes(hard-coding approach).

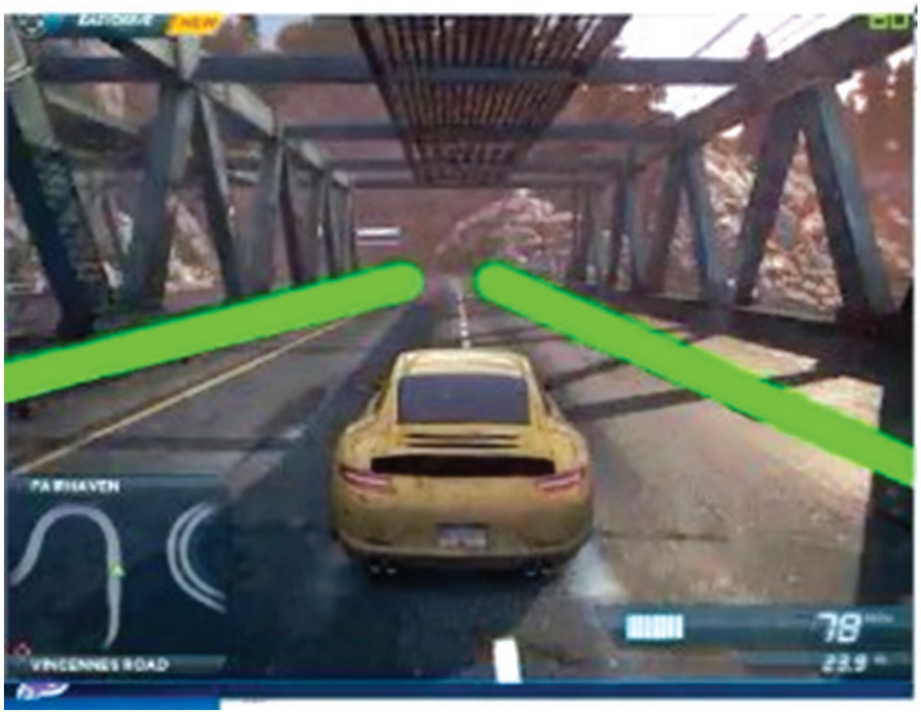


Figure 2: Lane Detection in the virtual environment

4. Discussion and Conclusion

After the model is constructed we train and test the model. In this work, the model performed well despite of some issues and the decision taking capacity of the model was satisfactory. The model was able to take decisions in a way to keep the vehicle within the lane and drive it through the road. The performance of the model can be observed on the basis of accuracy and loss graphs as shown in the figures 3.

The accuracy of the model was measured to be at 97% with the training data and the accuracy was found to be same at nearly 97% for the validation data. The loss also gradually decreased in both the training and validation with some spikes going up at some moment but had an overall decrease. The loss for both training and validation is at around just about 1%. If we compare both training accuracy and loss with the validation accuracy and loss, respectively we can see that they are fairly close, however there is a fluctuation in both the graphs regarding both loss and the accuracy this is due to the fact that the sample used for training was not large enough for the network to learn enough from it. AlexNet is a large network and the network was fed with a small dataset of only 200000 samples with 4000 training epochs, and 50000 validation set due to the limitation of the system. The training set is relatively small and if we put the complexity of the problems in mind it was more likely that there were fewer samples which might have ended the model wandering around rather than locking on good local minima.

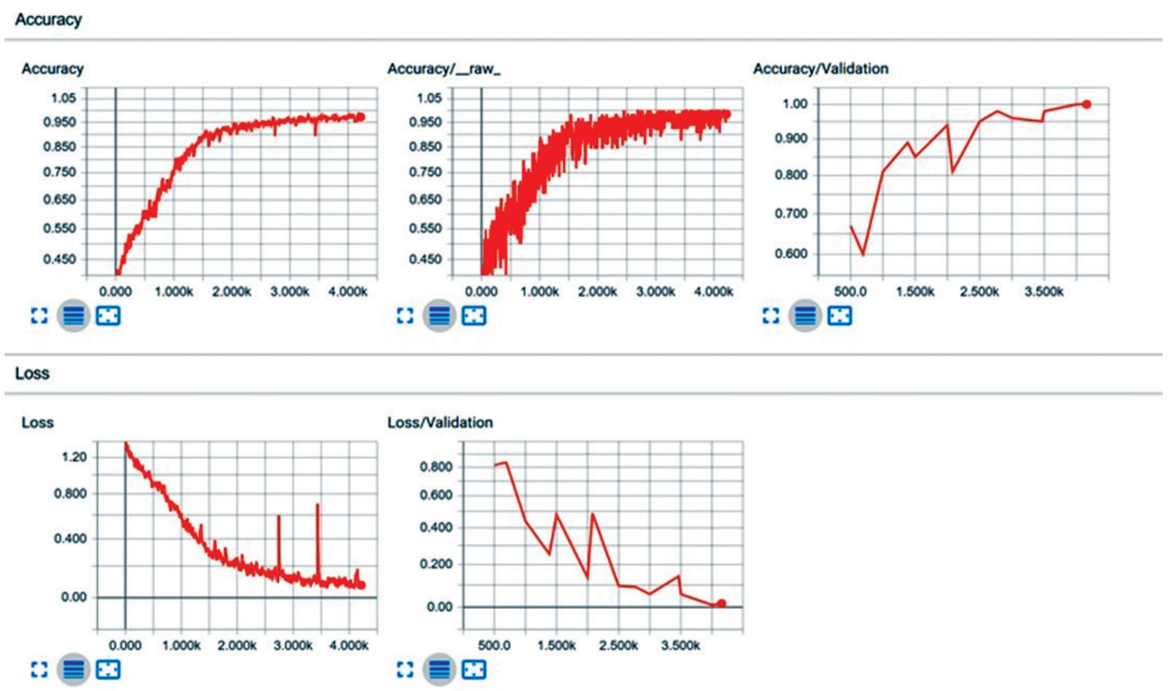


Figure 3: Accuracy and Loss graph for training and validation set generated by tensorboard for the model

From the figure 3 it can also be observed that the validation loss is slightly lower than that of training loss there are two major explanation to this; it is either because training set had many 'hard' cases to learn or validation set had mostly 'easy' cases to predict.

The model of CNN can perfectly work for the autonomous driving systems in a virtual environment and for the environment was acting as a specimen of the real world the model should do well in the real case scenario too. However, there needs to be some considerable change in the model for the real use.

For the model to be used in the real world there needs to be some add up to the factors like movement. The model here had limited movements like forward, forward right, forward left, stop, but in the real case there are more functions and decisions the model is supposed to be making such as reverse, reverse right, reverse left, gear shift, object and signal detections etc. The model will also be required to be trained on much larger datasets to achieve a smooth accuracy and precision. If the things done are right then with the add ups and some changes the model can do perfectly well for autonomous driving system in the real world too. Even though the model needs changes for implementation in the real world, it should currently be fine as the game bots for the games with racing genre.

5. Limitations

Although the model seemed working perfectly it had some limitations. The first limitation to the model was that the model was not consistent enough on performing the brake actions when needed. The model had limitations on identifying the objects such as other vehicles and traffic signals therefore, failed to perform the necessary maneuver operations because the model was not introduced with the capability of object detection and maneuvering over object to perform a smooth driving. Furthermore, the model was not able to maintain the speed limits which would be necessary in the real world scenario. Lastly, model had limited set of movements and couldn't perform reverse driving. Hence, if the model is to be introduced into the real-world cases the limitations needs to be addressed in a proper and efficient way.

6. Future Work

The method implemented would best suit for the condition when there is not enough risk of losing life or significantly damaging the property around unlike in the real-world because of the limitations the model had due to the limitations in capacity of the system we used to build the model. However, the limitations we had with the model are not something that couldn't be overcome.

The limitations of the model can be solved in the upcoming works. The model could be introduced for object detection, traffic signal recognition and relatively good maneuver operations with a combination of hard coding approach for maneuvering and by utilizing pre-trained neural networks for object detection and signal recognition which would reduce the efforts of training a network while increasing the performance of the model. The brake actions and speed limits is planned to be introduced more efficiently by increasing the training samples that would have more of the movements and operations. Furthermore, the reverse driving could be added with a little modification while training the neural network.

Conflict of interest

Not declared by the author(s).

Acknowledgement

The paper is based on our minor project conducted in 2018 named 'Game-bot using AI' under the requirement of Institute of Engineering, Tribhuwan University for the Bachelors' degree pro- gram. Therefore we would like to thank the Department of Computer Engineering, Advanced college of Engineering and Management, Nepal and then faculty members of the college. We would like to thank our then supervisors Er. Anku Jaiswal and Prof.Dr.Subarna Shakya for their valuable advices while we were doing our projects. Futhermore, we would like to extend our gratitude to Dr. Baburam Dawadi for his extremely valuable guidance in writing this paper. Without their help the paper would have been impossible.

References

[1] R. J. Schonberger, World class manufacturing. Simon and Schuster, 2008.

[2] N. B. Sarter, D. D. Woods, C. E. Billings, et al., “Automation surprises,” Handbook of human factors and ergonomics, vol. 2, pp. 1926–1943, 1997.

[3] B. M. Muir and N. Moray, “Trust in automation. part ii. experimental studies of trust and human intervention in a process control simulation,” Ergonomics, vol. 39, no. 3, pp. 429–460, 1996.

[4] A. Panesar, Machine learning and AI for healthcare. Springer, 2019.

[5] M. Peden and L. Sminkey, “World health organization dedicates world health day to road safety,” Injury Prevention, vol. 10, no. 2, pp. 67–67, 2004.

[6] P. Jackson, C. Hilditch, A. Holmes, N. Reed, N. Merat, and L. Smith, “Fatigue and road safety: a critical analysis of recent evidence,” Department for Transport, Road Safety Web Publication, vol. 21, 2011.

[7] I. Ciganovic, A. Pluškoski, and M. Jovanovic, “Autonomous car driving-one possible implementation using machine learning algorithm,” in Proceedings of the 5th International Conference on Electrical, Electronic and Computing Engineering, vol. 1, pp. 1016–1021, 2018.

[8] W. Xia, H. Li, and B. Li, “A control strategy of autonomous vehicles based on deep reinforce- ment learning,” in 2016 9th International Symposium on Computational Intelligence and Design (ISCID), vol. 2, pp. 198–201, IEEE, 2016.

[9] R. R. O. Al-Nima, T. Han, and T. Chen, “Road tracking using deep reinforcement learning for self-driving car applications,” in International Conference on Computer Recognition Systems, pp. 106–116, Springer, 2019.

[10] R. Emuna, A. Borowsky, and A. Biess, “Deep reinforcement learning for human-like driving policies in collision avoidance tasks of self-driving cars,” arXiv preprint arXiv:2006.04218, 2020.

[11] Z. Cao, S. Xu, H. Peng, D. Yang, and R. Zidek, “Confidence-aware reinforcement learning for self-driving cars,” IEEE Transactions on Intelligent Transportation Systems, 2021.

[12] M. R. Mendonça, H. S. Bernardino, and R. F. Neto, “Simulating human behavior in fighting games using reinforcement learning and artificial neural networks,” in 2015 14th Brazilian sym- posium on computer games and digital entertainment (SBGames), pp. 152–159, IEEE, 2015.

[13] T. Agrawal, R. Gupta, and S. Narayanan, “On evaluating cnn representations for low resource medical image classification,” in ICASSP 2019-2019 IEEE International Conference on Acous- tics, Speech and Signal Processing (ICASSP), pp. 1363–1367, IEEE, 2019.

[14] M. R. Manne, D. Bisht, H. C. Sharma, and A. Kumar, “Development of artificial neural-network-based models for the simulation of spring discharge,” 2011.

[15] S. Agatonovic-Kustrin and R. Beresford, “Basic concepts of artificial neural network (ann) mod-

eling and its application in pharmaceutical research," Journal of pharmaceutical and biomedical analysis, vol. 22, no. 5, pp. 717–727, 2000.

[16] A. Dongare, R. Kharde, A. D. Kachare, et al., "Introduction to artificial neural network," International Journal of Engineering and Innovative Technology (IJEIT), vol. 2, no. 1, pp. 189– 194, 2012.

[17] A. Gupta, A. Anpalagan, L. Guan, and A. S. Khwaja, "Deep learning for object detection and scene perception in self-driving cars: Survey, challenges, and open issues," Array, vol. 10, p. 100057, 2021.

[18] M. Farag, M. El Din, and H. El Shenbary, "Deep learning versus traditional methods for park- ing lots occupancy classification," Indonesian Journal of Electrical Engineering and Computer Science, vol. 19, no. 2, pp. 964–973, 2020.

[19] Z. Zou, Z. Shi, Y. Guo, and J. Ye, "Object detection in 20 years: A survey," arXiv preprint arXiv:1905.05055, 2019.

[20] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," Communications of the ACM, vol. 60, no. 6, pp. 84–90, 2017.

[21] Y. LeCun, D. Touresky, G. Hinton, and T. Sejnowski, "A theoretical framework for back-propagation," in Proceedings of the 1988 connectionist models summer school, vol. 1, pp. 21–28, 1988.

[22] V. Ranganathan and S. Natarajan, "A new backpropagation algorithm without gradient de- scent," arXiv preprint arXiv:1802.00027, 2018.

[23] S. Park and T. Suh, "Speculative backpropagation for cnn parallel training," IEEE Access, vol. 8, pp. 215365–215374, 2020.

[24] P. M. Daigavane and P. R. Bajaj, "Road lane detection with improved canny edges using ant colony optimization," in Emerging Trends in Engineering & Technology, International Confer- ence on, pp. 76–80, IEEE Computer Society, 2010.

[25] X. Chen, R. Jia, H. Ren, and Y. Zhang, "A new vanishing point detection algorithm based on hough transform," in 2010 Third International Joint Conference on Computational Science and Optimization, vol. 2, pp. 440–443, IEEE, 2010.