1 Introduction

The last decade has seen massive growth in the field of Autonomous Driving, primarily due to a) proliferation of graphical processing unit(GPU), b) several projects like Google(Waymo) [1], Berkeley-DeepDrive [2], Apollo [3], have made their datasets open-source making it easier for people to work on these data and achieve better performance gains.

Training a deep neural network(DNN) forms the core of making a car autonomous. By using supervised learning, one can achieve reliable results as it gives greater control at each stage of training. The data-driven approach collects data in advance and labels it appropriately. It can then be fed to the DNN using supervised learning algorithms to train the best model possible.

Ever since the discovery of Alexnet in 2012 [4], the convolutional neural network(CNN) and deep learning(DL) are preferred choices to analyse images. However, it is well known that the camera sensors are susceptible even to a slight change in weather conditions. Sensors like radar [5], LIDAR [6], ultrasonic[7], depth camera give additional depth information for obstacle detection. These values then are fused with the camera images to make data fusion possible.

Even though there are some public data available, it is still not enough to reliably train a DNN. Then there is the cost of building an autonomous car. Fortunately, the last years have seen growth in reliable simulators which helped massively to collect data to help explore this field of research. To name a few simulators that are being actively used – LGSVL [8], Nvidia Drive [9], Carla [10], CarMaker [11]. In this thesis, the LGSVL simulator is used.

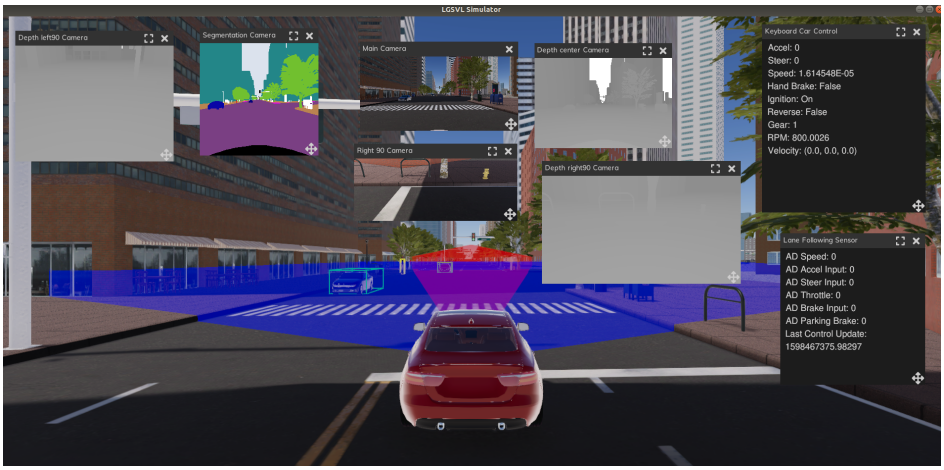

Figure 1.1: LGSVL[8] simulator active with all sensors

The LGSVL simulator allows the use of different sensors with minimal effort. The data from different sensors are published through websocket. So to capture these data, we need an interface/protocol which can understand the sent data's message types and enable the receiving node/programme to store them. However, the data from each sensor arrives at different rates. Hence it is necessary to collect and synchronise them in the order of their arrival before storing, to not lose their integrity and

thereby prevent corrupting the dataset. Robotic operating system(ROS) [12] and its functionalities fulfil this purpose. It allows seamless transfer of simulator's data by subscribing to it in the form of topics. Then the subscribing node synchronises it as necessary for storage.

So, now the data that resembles real-world is stored locally for later analysis and research.

1.1 Motivation

Though autonomous driving is one of the favourite research areas in mobility, a significant challenge is still the cost associated with integrating all the necessary sensors. Representing the environment around the vehicle(ego vehicle) requires information from all in-car sensors. The resources demanded to make an optimal decision are also a challenge. The motivation for this thesis is to use a simulator, do the required tests and determine whether using a simulator does indeed help in perceiving the environment and accomplish the goal of driving in the real-world.

As briefly mentioned, the high cost of associated sensors such as LIDAR [13], has put off many smaller research groups from implementing into their work. By using a simulator, again at a low cost, we can conduct adequate tests on how different constellation of sensors work, how different modalities interact with each other and what impact these factors have on the overall performance of the DNN.

Finally, implement an end-to-end system which simulates real-world behaviour which can then be applied to future research and make it more robust.

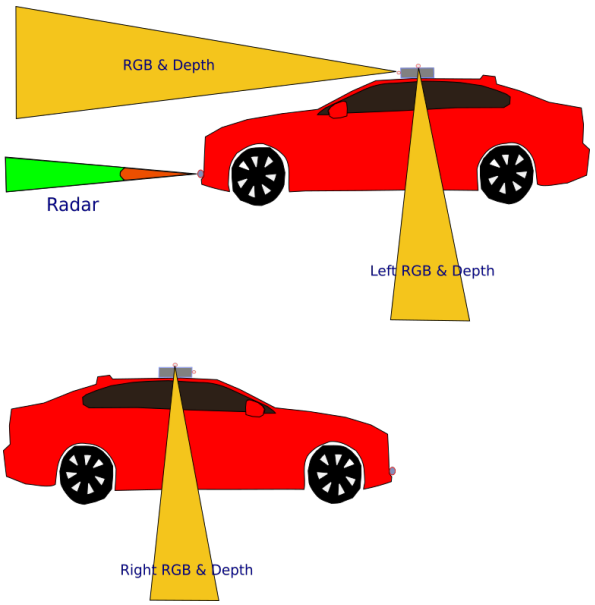

Figure 1.2: LGSVL[8] pictorial representation of sensors

1.2 Goal

The desired goals of this thesis are listed below:

- Building an autonomous driving framework -
  - ROS - use ROS2 to synchronise the data received from the simulator through a rosbrige, use functionalities such as slop and cache, to sort the data according to their received time

in order not to scramble the information. During the evaluation, use the same functionalities to send command controls to the simulator.

  - Rosbridge - use a bridge transport protocol that connects the ros on the receiving end to the simulator on the sending end or vice versa.
  - Docker - set up a work environment that is independent of hardware or operating system which allows easy running of the commands for data collection and evaluation.

- Implement an end-to-end neural network architecture which learns to drive by training from image pixels to steering commands. Also, apply state of the art DL techniques to it.
- Implement a system that can efficiently collect and label data.
- Implement and analyse different constellation of sensors with different data fusion techniques.

1.3 Related Work

In 2012, Alexnet [4] used CNNs to do object classification which, then in Computer Vision became the dominated approach for classification. Both Chen *et al.* [14] and Bojarski *et al.* [15] extended [4]'s approach of using CNN and detailed that in addition to classification, CNN can extract features from images. Then they went on to demonstrate through an end-to-end network(which self-optimises itself based on its inputs) steering angles can be predicted that keep the car in the lane of a road.

In a different field, but using CNN, Sergey Levine *et al.* [16] in 2016 corroborated that it was indeed possible to extract features with CNN and predict motor control actions in *object picking robots*.

Then, Xu *et al.* [17] in the same year with CNN-LSTM architecture showed that using the previous ego-motion events helped predict future ego-motion events. Using CNNs in an end-to-end architecture raised some questions on how it reached its decisions. So in 2017, both [18], [19] did visual analysis after the CNN layers to better understand the module's functionality. Vehicle control is more than just steering control. For smoother control, acceleration and braking are necessary besides steering. Both acceleration and deacceleration are dependent on the user's driving style, lane speed limit and traffic etc. Yand *et al.* [20] used CNN-LSTM architecture and provided the LSTM with feedback speed to determine the velocity of the ego vehicle.

Besides vehicle control, perceiving the environment is necessary for collision avoidance. The RGB colour camera sensors don't provide the depth information which is critical for collision avoidance. Hence, it is essential to fuse other sensors with diverse modalities with RGB to predict an optimal output. Liu *et al.* [21] provided rules in fusing data. They said that it was essential to pick out only vital information and discard other noisy data. They also described the techniques involved in data fusion – early/late fusion, hybrid fusion, model ensemble and joint training. Park *et al.* [22] gave us methods to enhance the features by using feature amplification or multiplicative fusion. Zhou *et al.* [23] detailed how fusing data into CNN affects the overall performance.

Even though the fused dataset gives a performance boost, it performs worse compared to individual modality. The combined fused model overfit more than its counterparts. The fundamental drawback of *gradient descent* in backpropagation causes the networks to overfit. This paper [24] introduced a technique called *gradient blending* to counteract this problem.

Xiao *et al.*[25] applied all the fusion techniques mentioned above with an imitation based end-to-end network[26]. RGB images with depth information(obtained through a different modality) could indeed result in better performing end-to-end network model was their conclusion.

1.4 Contribution

2 Fundamentals

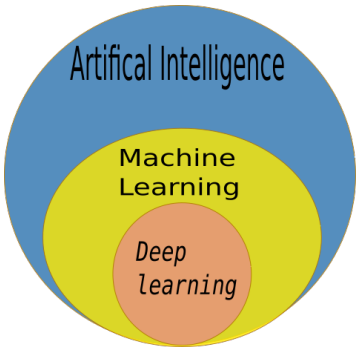

Figure 2.1: Representing Artificial Intelligence, Machine Learning and Deep Learning as a subset of one another.

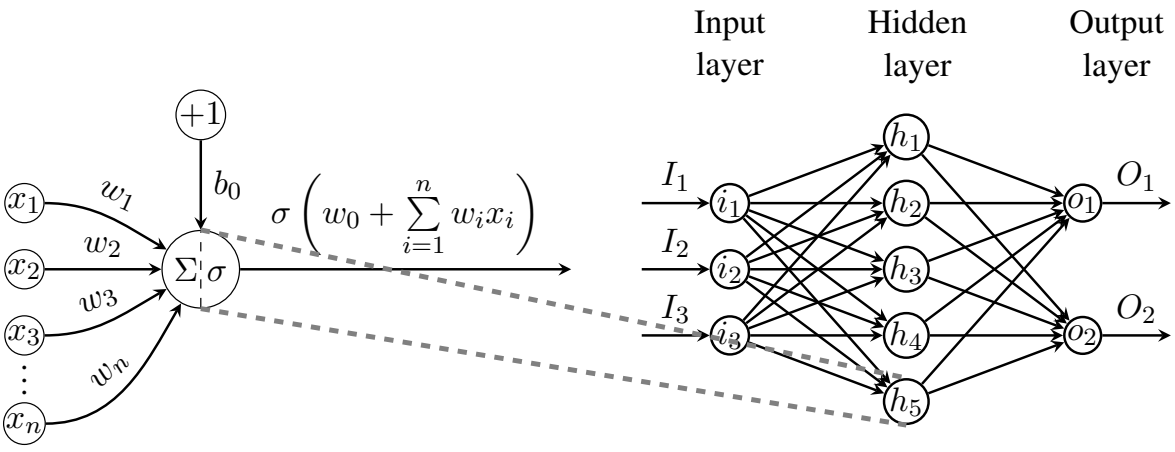

Figure 2.2: Multilayer Perceptron.

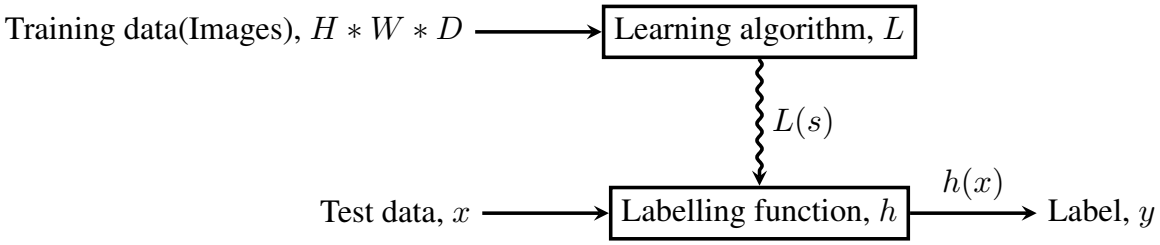

Figure 2.3: Supervised Learning set up

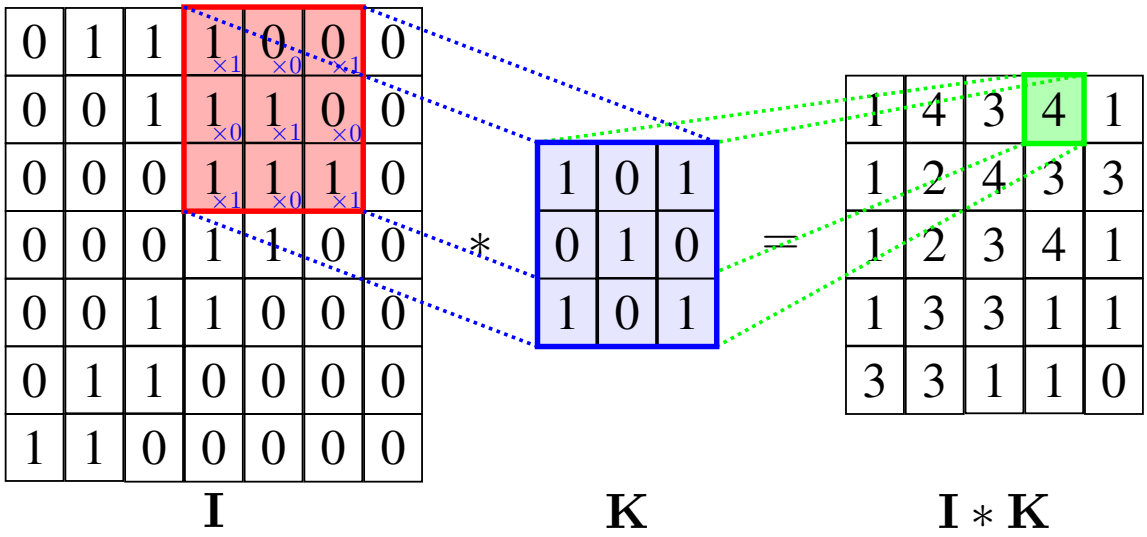

Figure 2.4: Two dimensional convolution

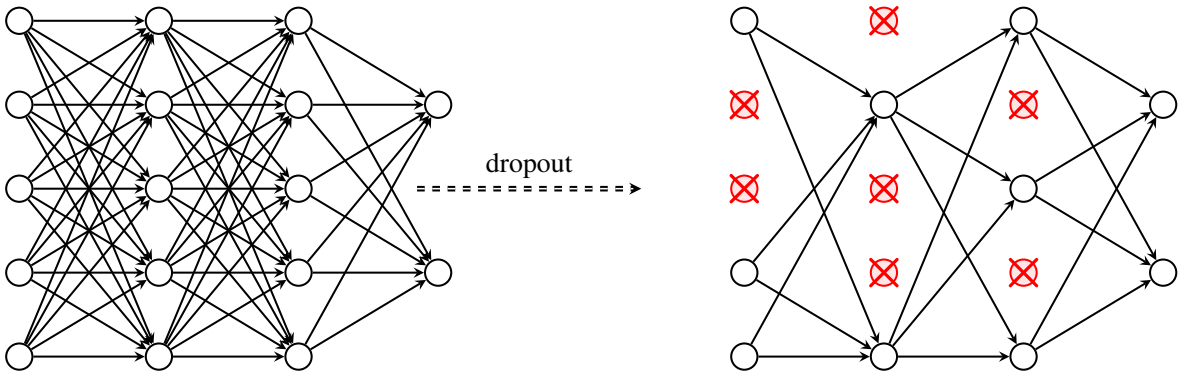

Figure 2.5: Illustrating dropout functionality

References

[1] "Waymo." https://waymo.com/

[2] "Berkeley-deepdrive." https://bdd-data.berkeley.edu/

[3] "Apollo." http://apolloscape.auto/

[4] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 1097–1105. http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

[5] "Radar introduction." https://www.explainthatstuff.com/radar.html

[6] "Lidar introduction." https://www.geospatialworld.net/blogs/what-is-lidar-technology-and-how-does-it-work/

[7] "Ultrasonic - what is it?" https://www.microcontrollertips.com/principle-applications-limitations-ultrasonic-sensors-faq/

[8] "Lgsvl simulator." https://www.lgsvlsimulator.com/

[9] "Nvidia simulator." https://www.nvidia.com/en-us/self-driving-cars/drive-constellation/

[10] "Carla simulator." https://carla.org/

[11] "Carmaker simulator." https://ipg-automotive.com/products-services/simulation-software/carmaker/

[12] "Robotic operating system." https://index.ros.org/doc/ros2/

[13] "Verge report on lidar." https://www.theverge.com/2020/1/7/21055011/lidar-sensor-self-driving-mainstream-mass-market-velodyne-ces-2020

[14] Z. Chen and X. Huang, "End-to-end learning for lane keeping of self-driving cars," 06 2017, pp. 1856–1860.

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

[16] S. Levine, P. Pastor, A. Krizhevsky, and D. Quillen, "Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection," *The International Journal of Robotics Research*, 03 2016.

[17] H. Xu, Y. Gao, F. Yu, and T. Darrell, "End-to-end learning of driving models from large-scale video datasets," *CoRR*, vol. abs/1612.01079, 2016. http://arxiv.org/abs/1612.01079

[18] J. Kim and J. F. Canny, "Interpretable learning for self-driving cars by visualizing causal attention," *CoRR*, vol. abs/1703.10631, 2017. http://arxiv.org/abs/1703.10631

[19] M. Bojarski, P. Yeres, A. Choromanska, K. Choromanski, B. Firner, L. D. Jackel, and U. Muller, "Explaining how a deep neural network trained with end-to-end learning steers a car," *CoRR*, vol. abs/1704.07911, 2017. http://arxiv.org/abs/1704.07911

[20] Z. Yang, Y. Zhang, J. Yu, J. Cai, and J. Luo, "End-to-end multi-modal multi-task vehicle control for self-driving cars with visual perception," 01 2018.

[21] K. Liu, Y. Li, N. Xu, and P. Natarajan, "Learn to combine modalities in multimodal deep learning," 2018.

[22] E. Park, X. Han, T. L. Berg, and A. C. Berg, "Combining multiple sources of knowledge in deep cnns for action recognition." in *WACV*. IEEE Computer Society, 2016, pp. 1–8. http://dblp.uni-trier.de/db/conf/wacv/wacv2016.html#ParkHBB16

[23] Y. Zhou and K. Hauser, "Incorporating side-channel information into convolutional neural networks for robotic tasks," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 2177–2183.

[24] W. Wang, D. Tran, and M. Feiszli, "What makes training multi-modal classification networks hard?" in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 12 695–12 705.

[25] Y. Xiao, F. Codevilla, A. Gurram, O. Urfalioglu, and A. M. López, "Multimodal end-to-end autonomous driving," *CoRR*, vol. abs/1906.03199, 2019. http://arxiv.org/abs/1906.03199

[26] F. Codevilla, M. Mller, A. Lopez, V. Koltun, and A. Dosovitskiy, "End-to-end driving via conditional imitation learning," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 4693–4700.