Training a Convolutional Neural Network for Driverless Car

Mrs. Neha N. Gharat
EXTC
Vidyavardhini's College of Engineering and Technology
Mumbai, India

Ajaya Kudva
EXTC
Vidyavardhini's College of Engineering and Technology
Mumbai, India

Pritesh Chavan
EXTC
Vidyavardhini's College of Engineering and Technology
Mumbai, India

Swapnesh Karle
EXTC
Vidyavardhini's College of Engineering and Technology
Mumbai, India

Akhil Rajan
EXTC
Vidyavardhini's College of Engineering and Technology
Mumbai, India

***Abstract*—For as far back as decade, there has been a rise of intrigue in driverless cars. This is because of leaps forward in the field of deep learning where deep neural networks are trained to perform errands that require human mediation. CNN's apply models to distinguish patterns and highlights in images, making them helpful in the field of Computer Vision. Instances of these are object recognition, image classification, picture captioning, etc. In this paper, training of a CNN is done utilizing images captured by a simulated vehicle so as to drive the vehicle self-sufficiently. The CNN takes in novel highlights from the images and generates predictions permitting the vehicle to drive without a human. For testing purposes and setting up the dataset the Unity based simulator gave by Udacity was utilized.**

Keywords—Self-driving, convolutional neural networks, deep learning

I. INTRODUCTION

Recently, self- driving algorithms utilizing minimal effort vehicle-mounted cameras have attracted expanding research attempts from both, the scholarly community and industry. Different degrees of automation have been characterized in self-driving domain. There's no automation in level 0. A human driver controls the vehicle. Level 1 and 2 are propelled by driver assistance systems where a human driver still controls the system but a couple of features like brake, steadiness control, and so on are automated. Level 3 vehicles are self-sufficient, but a human driver is as yet expected to mediate at whatever point important. Level 4 vehicles are fully autonomous however the automation is restricted to the operational design of the vehicle i.e. it doesn't cover every driving situation. Level 5 vehicles are required to be completely independent and their performance ought to be proportionate to that of a human driver. The progress though is exceptionally a long way from accomplishing level 5 driverless vehicles sooner rather than later. Moreover, level - 3/4 autonomous vehicles are possibly turning into a reality sooner rather than later. Essential reasons behind radical accomplishments are due to breakthroughs in the field of computer vision, artificial intelligence and furthermore the ease in vehicle-mounted cameras which can either autonomously provide noteworthy data or supplement different sensors. In this paper, the primary target is autonomous steering, which is a moderately unexplored undertaking in the field of computer vision and deep learning.

In this paper, a convolutional neural network (CNN) is implemented to map raw pixels from the captured images to steer the autonomous car. With least training data from the humans, the framework learns how to steer and about, with or without the lane markings.

II. RELATED WORK

The DAVE system made by DARPA [3] utilized images from two cameras in left and right directions to prepare a model to drive. It shows that the procedure of end-to-end learning can be applied to self-driving vehicles. This implies that the intermediate road highlights such as the stop signs and lane markings don't have to be labeled for the system to learn. DAVE is an early venture in the field of self-driving vehicles. With regards to current innovation, a huge portion depended on remote data exchange as the vehicle couldn't communicate to the computer which contrasts lighter gear that exists today. The design of this model was a CNN made up of completely associated layers that originated from networks utilized in object recognition.

The ALVINN system [5] is a 3-layer back-propagation system built by a group at CMU to complete the undertaking of lane following. It trains on images from a camera and a distance measure from a laser range discoverer to yield the direction the vehicle should move. ALVINN's model uses a single hidden layer back-propagation network.

Attempts are made to replicate a study by NVIDIA [5]. The framework utilizes a end-to-end approach where the data is first gathered in various diverse ecological conditions. The data is then augmented to make the system strong to driving off center and to various potential environments. The following stage is training the network on this data. The network architecture is of 9 layers beginning with convolutional layers also, trailed by completely connected layers. This is the network that was endeavored to reproduce.

As of late, a paper by a few IEEE researchers presented a significant diverse neural network design that likewise takes the temporal data into account [5]. They accomplished this in practice by a mix of standard vector-based Long Short-Term Memory (LSTM) and convolutional LSTM at various layers of the proposed deep network. Sequential frames often have a similar visual appearance, yet subtle per pixel movements can be seen when the optical stream is computed. Traditional image convolutions, as those embraced by cutting edge image classification models, can move along both spatial dimensions in a image, which infers that they are basically 2-D. Since these convolutions work on static images or multi-channel response maps, they are unequipped for catching temporal elements in recordings. The creators adopted a spatio temporal convolution (ST-Conv) that shifts in both spatial and temporal dimensions thus applying the convolution in 3 dimensions as opposed to the conventional 2-D process.

A similar paper also proposed the plan to consolidate temporal data in the model to get familiar with the steering data [3]. In this paper, the authors exhibit quantitatively that a Convolutional Long Short-Term Memory Recurrent Neural Networks (C-LSTM) can fundamentally improve end-to-end learning execution in self-driving vehicles steering dependent on camera images. Motivated by the ampleness of CNN in visual feature extraction and the efficiency of Long Short-Term Memory (LSTM) Recurrent Neural Networks in managing long range temporal dependencies our methodology permits to model dynamic temporal conditions with regards to steering angle prediction based on camera input.

III. DATA COLLECTION

Fig. 1. Udacity Simulator

Udacity's self-driving car simulator is utilized for gathering the data. This simulator is built in Unity and was used by Udacity for the Self-Driving Nanodegree program which recently released [6]. It reproduces what NVIDIA did in the simulation. Simulator is source for gathering all the data. Utilizing our keyboard to drive the vehicle, options to instruct the simulated vehicle to turn left, right, accelerate and slow down were available. Another significant perspective is that this simulator can be used for training as well as testing the model. Hence, it has two modes: (i) Training mode, and (ii) Autonomous mode as shown in Fig. 1.

The training mode is utilized to collect the data and the self-driving mode is used to test the model. Furthermore, there are two kinds of tracks in the test system - the lake track and the jungle track. The lake track is smaller and simpler to deal with the handling of the vehicle in contrast with the jungle track as shown in Fig. 2 and Fig. 3. The simulator captures data when the vehicle is driven around the track utilizing left and right keys to control the steering angles and here and keyword arrow keys control the speed.

Fig. 2. The lake track

From this, the simulator creates a folder containing images and one CSV document. The image folder contains three images for every frame captured by the left, center and right camera and every row in the CSV document contains four measurements – steering angle, speed, throttle and brake. Fig. 4, Fig. 5. also, Fig. 6 show the left, center and right picture, for one frame.

Fig. 3. The jungle track

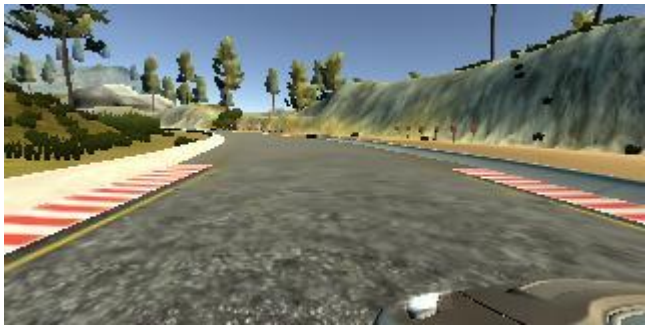

Fig. 4. Left image

Fig. 5. Center image

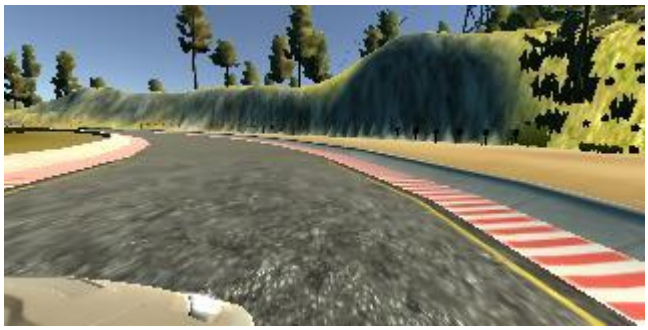

Fig. 6. Right image

IV. DATA PREPROCESSING

The data that is gathered i.e. the captured images are preprocessed before training the model. While preprocessing, the pictures are trimmed to remove the sky and front part of the vehicle. The pictures are then converted from RGB to YUV and resized to the input shape utilized by the model. This is done since RGB isn't the best mapping for visual perception. YUV color-spaces is a considerably more effective coding and diminishes the transmission capacity more than RGB capture can.

In the wake of choosing the final set of frames, the data is expanded by including artificial movements and rotations to show the system how to recuperate from a poor position or direction. While augmenting, we arbitrarily pick right, left or center images, flip the images left/right and change the steering angle. The controlling point is balanced by +0.2 for the left image and -0.2 for the right image. Utilizing the left/right flipped images is helpful to train the recovery driving situation. Additionally random translation of image on a horizontal plane or vertical plane with steering angle alteration is done. The horizontal translation can be helpful for taking care of situations with troublesome bends. The size of these changes is picked randomly from a normal distribution. The distribution has a zero mean. Application of these augmentations utilizing a script from the Udacity repository is performed.

Augmented images are included into the present set of images and their comparing steering angles are additionally balanced regarding the augmentation performed. The essential reason behind this augmentation is to make the framework progressively powerful, consequently, taking in however much as could be expected from the environment by utilizing diverse perspectives with various settings.

V. DEEP LEARNING MODEL

The deep learning model which is arranged depends on the study done by NVIDIA for their self-driving vehicle [5]. The model contains the following layers.

A. Convolutional Layer

Convolutional layer applies the convolution function on the input image and creates a 3D yield activation of neurons. This layer serves to find different highlights of the image which is utilized for classification. The number of convolutional layers in the system relies on the application. The underlying convolutional layers in the systems help recognize the low level features of the image which are basic and the convolutional layers further assist in identifying the elevated level features of the image.

B. Max Pooling Layer

Down sampling layer helps in lessening the number of parameters/loads in the system and helps in diminishing the training time without losing any particular feature information in the image. This layer creates a smaller image than the input image by downsampling the picture utilizing pooling of neurons. There are various kinds of pooling like max pooling, average pooling, L2-norm pooling and so forth in any case, max pooling is the one which is most generally used.

C. Dense Layer

Dense layer or Fully associated layer is equivalent to normal neural network layer in which all the neurons in this layer is associated with every neuron from past layer. This layer is generally planned as the last layer in the convolutional neural network.

The whole architecture diagram is shown in Fig. 7 There are 5 convolutional layers with differing number of filters and sizes and a dropout after that to deal with overfitting. In the end, 3 associated layers were included trailed by the output layer. Adam optimizer was utilized for parameter optimization with a fixed learning rate. Batch size of 100 was picked and the quantity of epochs of 50-60 was explored. On a machine without GPU, 8 GB ram Core i5 it took about 8 hours of training.

VI. SYSTEM ARCHITECTURE

Fig. 8 shows a High level architecture of the framework. In the wake of performing data augmentation on the input images, batches are made from them and fed to the CNN model for training. After the training is finished, the model is utilized to perform prediction on the steering angle and send the predictions to the Udacity Simulator to drive the vehicle.

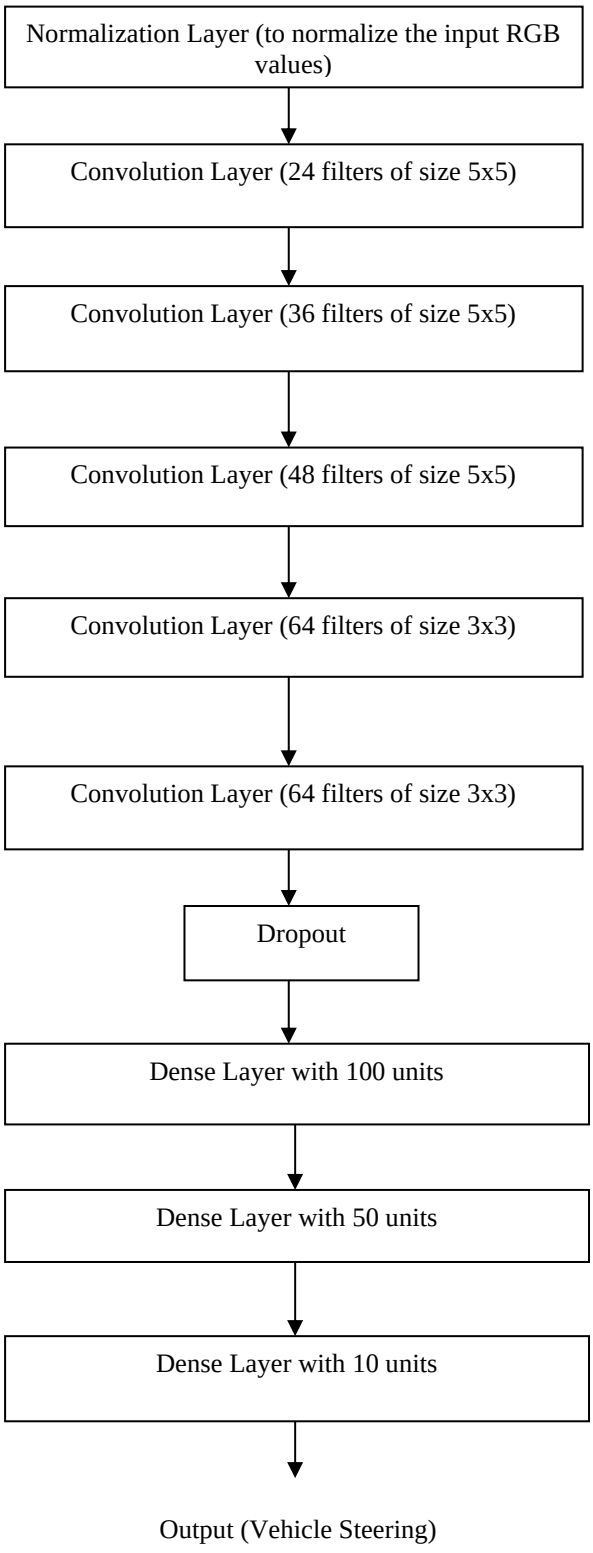

Fig. 7. Deep Neural Network Architecture

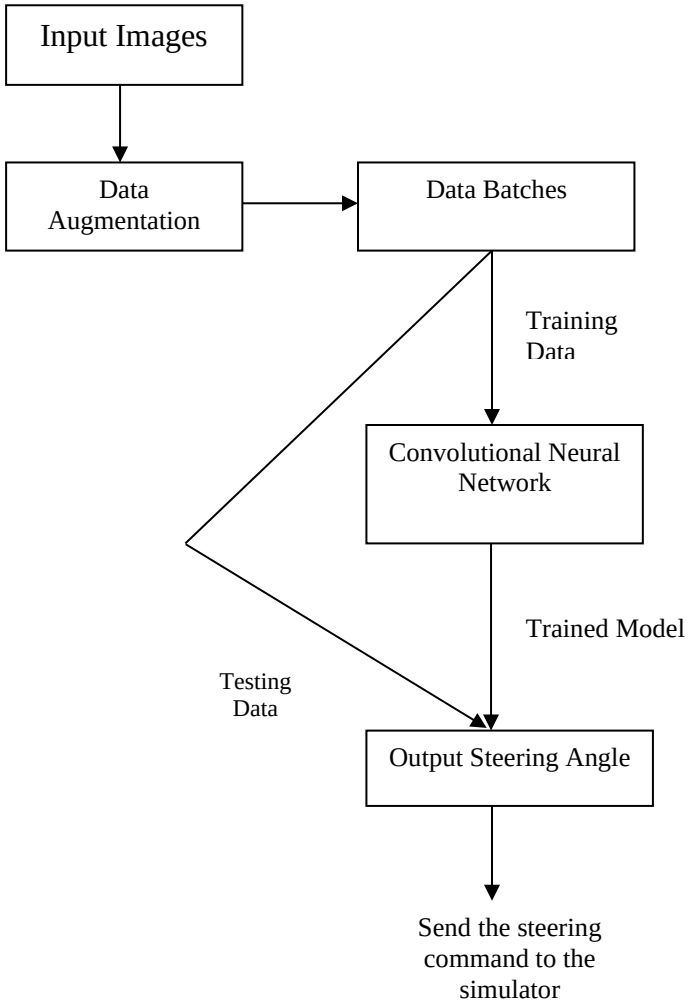

Fig. 8. High-level System Architecture

VII. PERFORMANCE EVALUATION

For evaluation of the performance of the model, mean squared error is utilized as a loss function.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y - y')^2 \quad (1)$$

In real life situations, while driving on street, the following metric has been proposed in [5]. It is determined by computing the quantity of mediations, multiplying by 6 seconds, dividing by the slipped amount of time of the simulated test, and afterwards subtracting the result from 1.

$$Autonomy = \frac{1 - mediations * 6\ seconds * 100}{elapsed\ time} \quad (2)$$

To assess our system, the autonomous part of the simulator on both the jungle and lake track was executed. In the jungle track, the vehicle drove off the track after 10 seconds of driving. By the above mentioned expression of autonomy, this would imply that the vehicle was 60% autonomous. Re-training of the model with more information was done and the vehicle never drove off the driving track making it 100%

autonomous. The overall behavior doesn't seem reasonable. It shows up to fluctuate between the edges of the path.

During the initial run on the lake track, the vehicle doesn't leave the track, however it seems to embrace the left side of the path and that is the direction that the bend in the track is turning towards. The vehicle drives to one side of the path and when it runs near the left path marking, the vehicle drives back toward the central point of the path until it is around somewhere between the central point of the path and the left lane marking, at that point it drives back to the left half of the path. This behavior repeats. After re-training the model, the vehicle's driving shows up considerably more smooth. It doesn't embrace the left half of the path, however it seems to head to the other side of the path and move towards the other end. This behavior also repeats. As far as autonomy is concerned, the two models show up to be 100% autonomous.

After the second time when the model was trained, the vehicle drives self-sufficiently, but the behavior is abnormal. This is something to remember when refining the system in future.

VIII. FUTURE WORK

We have a few plans to improve the performance of self-driving vehicle. The first thought is to add the component of speed to the CNN with the goal that when the simulator is in autonomous mode, it is utilizing the anticipated speed making so that the movement appears more realistic.

Another conceivable improvement would be to take into account every captured image from the cameras independently and make CNN models utilizing each surge of pictures to make a particular steering angle prediction originating from the left, center and right model. At that point, averaging the three of these to get a progressively precise prediction is possible. We would anticipate that most of the time, this model would have exact predictions however if one model predicts a steering angle that is dissimilar to the next two, at that point it turn the steering angle to an unforeseen course.

Also, we would like to include training data where the vehicle recovers from being off the track. In the expansion venture of this procedure, the system figures out how to recover from some little setbacks however never from bigger deviations from the track. In the event that the self-driving vehicle is driving totally off the track it won't have the option to recover. We want to include training data where the vehicle begins from being off the track and then advances back into it. This would require the data recording to be cut from the beginning when the vehicle is off the track which would remove the user driving the vehicle off the track since we don't need the model to learn how to drive the vehicle off the track, yet we do need it to get the hang of driving onto the track.

A chance to assess the heartiness of the system is to create another track other than the predefined jungle and the lake track and check whether the model trained would be able to drive the vehicle on that track. Likewise, in the event that it could drive moderately well, it would be fascinating to watch the distinction in how well it drives on the trained track versus the track that the model hadn't been trained on.

IX. CONCLUSIONS

In this paper, the steering angles were effectively predicted utilizing convolutional neural networks followed by the comprehension of the inner subtleties of convolutional neural networks alongside the manner in which they can be tuned. It is also additionally illustrated that CNN's can get familiar with the whole assignment of lane and track following without manual mediation into track. A small measure of training data from not less than hundred long stretches of driving hours was adequate to train the vehicle to work in assorted conditions, on roadways and on other tracks. A fascinating proviso to this is the system was able to effectively drive on the track that it had been trained on. Autonomous systems for vehicles that don't utilize the Udacity simulator require more noteworthy robustness as they need to take into account tracks that haven't been driven on. Moreover, the task defined in our paper was effectively achieved.

ACKNOWLEDGMENT

It has been a great experience working on this paper. However, it would not have been possible without the kind support and help of many individuals. We would like to extend our sincere gratitude to all of them.

We sincerely thank our paper guide Prof. Neha Gharat for her valuable guidance and encouragement during every stage of this paper.

We also owe a depth of gratitude to Dr. Vikas Gupta, HOD, Department of Electronics and Telecommunications Engineering for all the facilities provided during the course tenure.

REFERENCES

[1] Eraqi, Hesham M., Mohamed N. Moustafa, and Jens Honer. "End-to-End Deep Learning for Steering Autonomous Vehicles Considering Temporal Dependencies.", 2017.
[2] Bojarski, Mariusz, et al. "End to end learning for self-driving cars", 2016.
[3] LeCun, Y., et al. DAVE: Autonomous off-road vehicle control using end-to-end learning, 2004.
[4] Pomerleau, Dean A. "Alvinn: An autonomous land vehicle in a neural network." Advances in neural information processing systems, 1989.
[5] Chi, Lu, and Yadong Mu. "Deep Steering: Learning End-to-End Driving Model from Spatial and Temporal Visual Cues", 1989.
[6] Deep learning for Video classification and captioning by Zuxuan Wu, Ting Yao, Yanwei Fu, Yu-Gang Jiang
[7] Dropout: A Simple Way to Prevent Neural Networks from Overfitting by Nitish Srivastava, Geoffrey Hinton , Alex Krizhevsky, Ilya Sutskever, Ruslan Salakhutdinov.
[8] Visualizing and Understanding Convolutional Networks by Matthew D. Zeiler, Rob Fergus.
[9] www.github.com/udacity/self-driving-car-sim
[10] www.keras.io/backend