WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

Self-Driving Carusing Deep Learning

Tushar Garg[1], Saksham Aggarwal[2], Shivam Goel[3], Aarti Chaudhary[4]

[1,2,3,] Student (Final Year), Department of Computer Engineering, Raj Kumar Goel Institute of Technology & Management, Ghaziabad, UP, India

[4]Professor, Department of Computer Engineering, Raj Kumar Goel Institute of Technology & Management,Ghaziabad,UP India

A B S T R A C T

The rapid development of Artificial Intelligence has revolutionized the area of autonomous vehicles by incorporating complex models and algorithms. Self-driving cars are always one of the biggest inventions in computer science and robotic intelligence. Highly robust algorithms that facilitate the functioning of these vehicles will reduce many problems associated with driving such as the drunken driver problem. In this paper our aim is to build a Deep Learning model that can drive the car autonomously which can adapt well to the real-time tracks and does not require any manual feature extraction. This research work proposes a computer vision model that learns from video data. It involves image processing, image augmentation, behavioural cloning and convolutional neural network model.The Model achieved better performance when it is provided even more dataset. Here, we USE many Convolutional neural network architectures to obtain better performance with lesser load.

Keywords:CNN, Deep Learning, Image Processing,Image Flip, Lower Resolution

1. INTRODUCTION

Autonomous cars are the cars that require no interaction with driver to run on the track. Autonomous car relies on central processing unit, Graphics processing unit and sensors to run car without the occurrence of faults. If a person wants to go hospital or to some other place but he or she not able to drive or don't know how to drive, in this case the person will have to hire a driver which may be costly for person to afford. That situation might be unpleasant for the car owner. To avoid such type of unwanted situations autonomous car can play a key role and can be affordable for the person.

Automated vehicles are technological development in thefield of automobiles. Although the automated vehicles are forease of humankind yet they are the most expensive vehicles in the paper considering the different features and the cost, ona small scale a three-wheel Vehicular Robotic prototype hasbeen designed that will automatically reach the destination ofanother vehicle to which it is supposed to follow.We have focused on two applications of an AutomatedVehicles here and designed a prototype vehicle for that the one major issue is during heavy traffic a driver hasto continuously push brake, accelerator and clutch to moveto destination slowly. We have proposed a solution to relax the driver in that situation by making vehicle smart enoughto make decisions automatically and move by maintaining aspecified distance from vehicles and obstacles around.The second issue is when two vehicles have the samedestination but one of the drivers doesn't know its route. Thedriver can make his vehicle follow the front vehicle if they areknown and share their location to reach the same destination.

The purpose of the paper is to reduce the loss function between the actual and predicted values of the target value so that that the accuracy of the model can be increased on the given large dataset generated by simulator. By generating the large amount of dataset, we can use it to train the CNN model to predicted steering angle. Our primary goal is studying the three different CNN architecture to choose well sophisticated architecture that captures and mimic the driving scenario of a driver in real life.

2. RELATED WORKS

In past few years Deep learning networks are very helpful in automation filed. Convolutional neural networks (CNN) are the most important for image processing. CNN is used in enormous number of technical fields like handwriting recognition, face recognition and various application in computer vision. They are introduced by Yann LeCun, a postdoctoral computer science researcher, to create a very basic of image recognition network. CNN is very important neural network. These are used recognition, segmentation and prediction of target value. The DARPA (Defence Advance Research Project

* *Corresponding author.*

E-mail address: tushargarg388@gmail.com

Agent) created an autonomous driving system known as DAVE which used images captured from the two front cameras along with steering angle to train the system to drive. The Autonomous Land Vehicle in a neural network (ALVINN) is multilayer backpropagation network developed by researchers at Carnegie Melion University. It uses images captured by camera and distance measure by laser to train the machine to predict the direction in which car should move to keep the car on track. There are many pre-trained networks for image recognition. Among from them VGG16 have show very good results but it contains huge number of parameters, thus it requires enormous number of resources to train the machine and can lead to overfitting of data which may predict the wrong steering angles and can led the car run into an accident.

3. METHODOLOGY AND IMPLEMENTATION

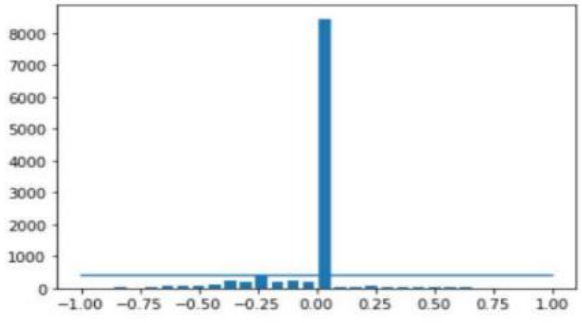
A. Data gathering and simulator

We used the term 1 Udacity simulator to generate the data. This Udacity simulator is built with the help of Unity and it is open sourced for everyone to use it. we can easily generate the data by using simulator. The data is generated in form of csv log file containing state of vehicle (steer angle, brake, throttle and name of the images (Centre) captured by cameras (Centre) at an instance) and it also contain folder containing those images mention in CSV log file. We collected nearly 30000 images collectively from all three cameras (Centre) in 1920X1050 resolution along with vehicle data when the car is driven around the track.

Fig. 3.1.1 Centre camera image

B. Avoiding overfitting

To avoid the overfitting of the model on any value during the training we need to remove some of the data on that value. In the dataset, there are enormous number of data entries of the steering angle are very close to zero. So, we deleted some of the entries in dataset to spread the values across the range and to avoid the overfitting.



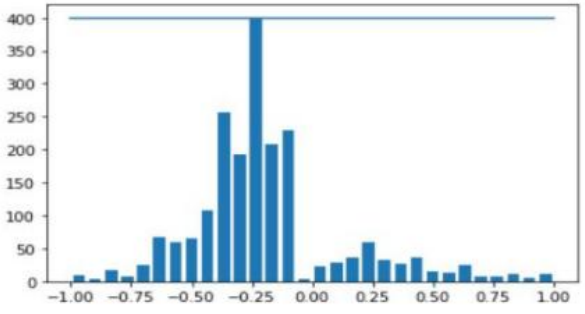
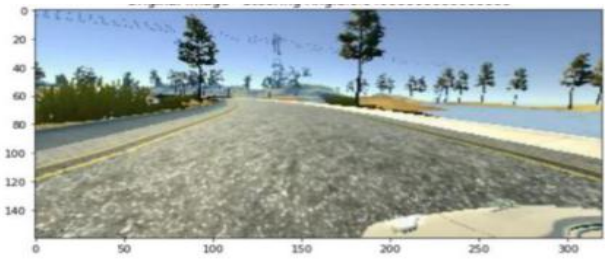


Fig. 3.2.1 Data beforeFig. 3.2.2 Data after

C. Image Flip

The chosen random image is flipped horizontally in the process along with its steer angle by multiplying it by - 1. After the image is chosen then it is decided with the constant probability to whether to flip the image or not. This helps to add new data to the dataset if the image is flipped so that model can be trained in every possible steering angle more accurately.



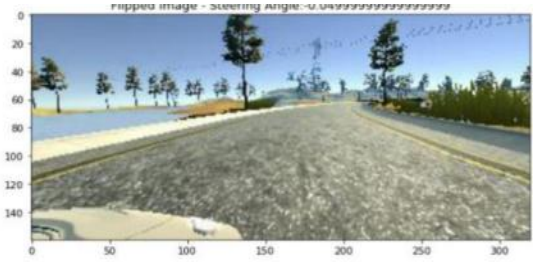


Fig. 3.3.1 Original imageFig. 3.3.2Flipped image

D. Image processing

After the downsizing and cropping of the image, it is converted from RGB image to YUV image. YUV is a color coding system that is commonly used in color image pipeline. The main reason is that brightness is important than the image color, so you can lower the resolution of V and U while keeping Y resolution same. Therefore, the u and v resolutions can be lowered, they are compatible with CNNs, and CNNs can be trained with the same precision many times faster than traditional RGB models. After the conversion we normalize the pixel values of whole image by dividing it by 255 and then store it into multidimensional array to feed it to network.

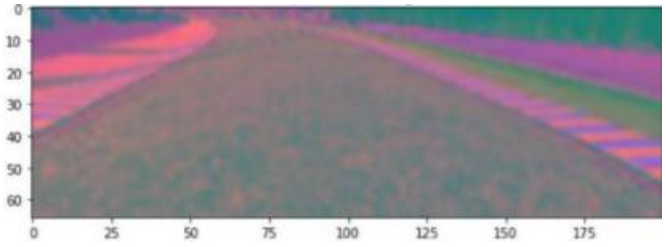
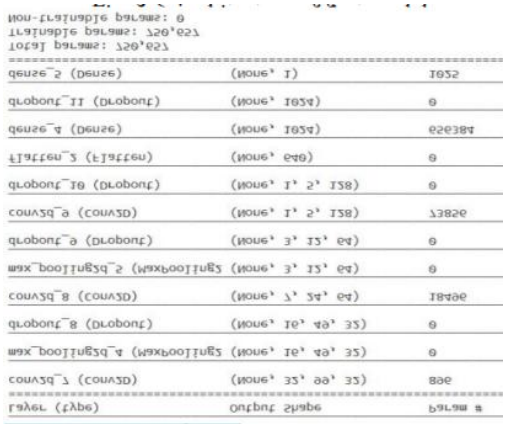


Fig 3.4 Processed image ready for input.

E. Convolutional neural network architectures

We have designed three multilayer CNN architecture with the aim of comparing these architectural models to get maximum accuracy model and to minimize the loss while training the model. The first architecture consists of five Convolutional layers along with varying filter layers like Maxpooling, Dropout, Flatten, and Dense layer. In this last Dense layer output only one value which is considered the predicted steering angle for processed input image passed to first convolutional layer as shown below in Fig. 3.5

In this architectural (3.5) model we use small kernel size of (3,3) because it reduces computational costs and Weight sharing that ultimately leads to lesser weights for back-propagation. And in this architecture, we use the RELU activation function.



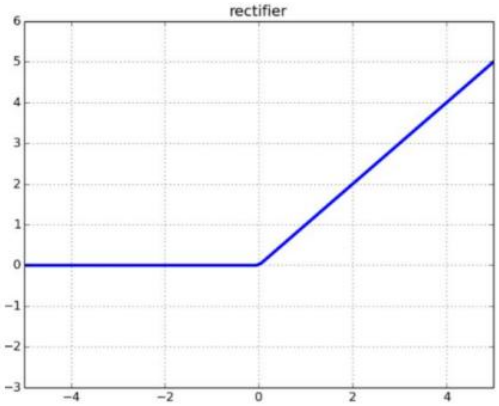


Fig. 3.5 Architecture of first model.Fig 3.6 The graph of RELU activation function.

In the second architecture, there are four convolutional layers and also along with varying no. of filter layers Maxpooling, Dropout, Flatten, and Dense layer. The output of last Dense layer is the steering angle calculated for the input image. The architecture second of the CNN model is shown in Fig. 3.7.

we use the ELU activation function. And also, we took set the strides to (2,2) and kernel size to (5,5) for the first three convolutional layers and for the rest convolutional layers we set kernel size to (3,3) to increase the performance of the architecture. Andexpectedly the training parameter were decreased to train the model in less time.

Layer (type)	Output Shape	Param #
conv2d_6 (Conv2D)	(None, 31, 98, 24)	1824
conv2d_7 (Conv2D)	(None, 14, 47, 36)	21636
conv2d_8 (Conv2D)	(None, 5, 22, 48)	43248
conv2d_9 (Conv2D)	(None, 3, 20, 64)	27712
conv2d_10 (Conv2D)	(None, 1, 18, 64)	36928
dropout_1 (Dropout)	(None, 1, 18, 64)	0
flatten_1 (Flatten)	(None, 1152)	0
dense_4 (Dense)	(None, 100)	115300
dense_5 (Dense)	(None, 50)	5050
dense_6 (Dense)	(None, 10)	510
dense_7 (Dense)	(None, 1)	11

Total params: 252,219
Trainable params: 252,219
Non-trainable params: 0

Fig. 3.7 Architecture of second model

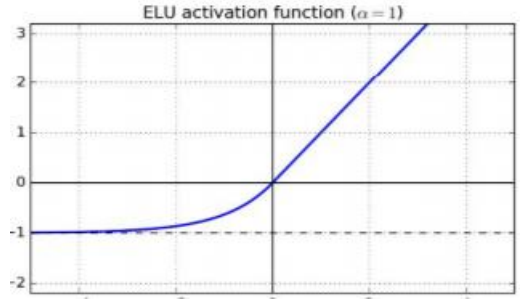


Fig 3.8 the graph of ELU function.

$$f(\alpha, x) = \begin{cases} \alpha(e^x - 1) & for\ x < 0 \\ x & for\ x \geq 0 \end{cases}$$

F. Training and validation

By using the traintestsplit function from the sklearn we split the total image dataset of 31845 images between training set and the validation set like 80%goes to training set and remaining 20% goes to validation set randomly. We set the batch size to 300 images for training and to 150 images for validating. We set the stepperepoch to 300 and validationstep to 150 and epoch to 30.

$$MSE = \frac{1}{n}\sum(Yi - \hat{Y}i)^2$$

Where Yi is the actual value,
$\hat{Y}i$ is the predicted value,
n defines the number of training samples.

We use Adam optimizer with learning rate of 10-4 and loss as MSE (mean squared error) which compute the loss on predicted target value and actual target value.

4 FINAL RESULTS

The above fig. 4.1 shows the loss computed by MSE loss function during every epoch for the first CNN architecture of. We see the drastic decrement in the loss over the first few epochs during the training and validation. And for the rest of the remaining epochs loss decrease very slowly. Here loss at the 30th epoch is 0.045 for validation.The fig 4.2 demonstrate the accuracy for the first CNN architecture over epochs during training and validation. This graph shows the average accuracy of 85% for training and validation.

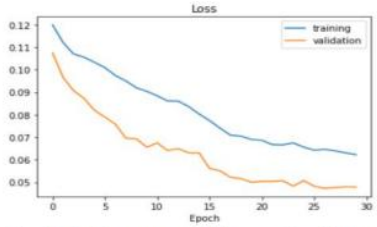


Fig. 4.3 The loss over epoch for second architecture.

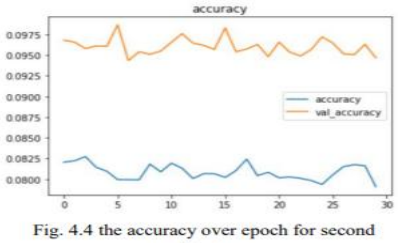


Fig. 4.4 the accuracy over epoch for second architecture.

accuracy

The Fig 4.3 demonstrate the loss function value over the 30 epochs during the training and validation. We see the slight decrease in the loss over every epoch. The observed loss during the training is 0.062 and during validation is 0.047. As shown in the Fig. 4.4 the validation accuracy 94.7% which is very high. And training accuracy is 79.1% which is comparatively very low than validation accuracy.

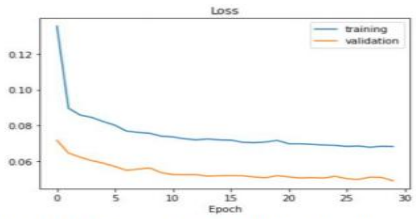


Fig. 4.5 The loss over epoch for third architecture.

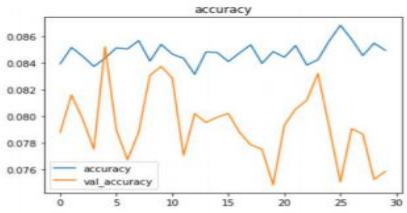


Fig. 4.6 the accuracy over epoch for third architecture.

In the Fig 4.5 for the first few epochs loss has rapidly decreased then after for remaining epochs less decrement is seen. The training loss is at 30th epoch is 0.068 and the validation loss is 0.049. In Fig. 4.6 it is seen that validation accuracy is less than training accuracy. The validation a accuracy is 75.9% and training accuracy is 84.2%. By analyzing the facts, we see loss over the epoch is decreasing continuously which mean the model is computing more accurate target value then before. So we can say that architecture with more training parameter will perform better. Hence, neural network architecture first and neural architecture third will give better performance than architecture second because that have large number of trainable parameters.

4. CONCLUSION

In this research paper, we explained the need of simulator to generate the data which turn out to be very effective for the model to train to compute the steering angle based on input images. The use of simulator save cost of physical equipment's required to collect the real-world data. And simulator turn out to be very well efficient for training the model for prediction steering angle. The prediction of the steering angle based on the processed captured images turn out to be fine for CNN network because it extracts the feature and find the necessary dependencies required for prediction. In this paper, we compared the three CNN architecture to know which architecture can give better performance. Here we find out that the more parameter architecture has the better the performance will be. The first and third architecture the show the better result than second architecture. The first architectureshows the accuracy of 85% and it is more than the other two architectures. The architecture with lesser trainable parameter should be trained over many more epochs than epochs in architecture have larger training parameter to give same performance.

5. FUTURE SCOPE

Future work involves theverification of the ensemble-M on more than two models for achieving the least RMS score and hencemaking end-to-end steering control system more feasible than what its status for implementation Istoday. There is a plethora of techniques to implement full-edged autonomous driving. Methodsincluding deep reinforcement learning, logic programming, deep learning (road segmentation, end-to-end)are a few of the methods to achieve the tasks. Irrespective of the technique used, there will always be at time-space trade-off. A deeper network tuned with correct parameters may lead to a better MSE score butat the same time, it may lead to a more time-consuming process. On the other hand, a shallow deepnetwork may not have a good MSE score but performs comparatively enough required by the application.Since autonomous driving involves the risk of life, this trade-off has to be balanced on some middleground. Since this project was implemented in the simulated environment, there will always be thepossibility of missing features in the simulator which may have affected the performance and decisionmaking of the model.

REFERENCES

[1] Baruch, J. (2016), "*Steer driverless cars towards full automation*", Nature, Vol. 536, p. 127.
[2] LeCun, Y., et al. DAVE: Autonomous off-road vehicle control using end-to-end learning. Technical Report DARPA-IPTO Final Report, Courant Institute/CBLL, 2004.
[3] Bojarski, Mariusz, "End to end learning for selfdriving cars." (2016).
[4] A. R. Diana Putri, Litasari and A. Susanto, "Comparison between colour models in automatic identification of cane sugar," 2013 IEEE International Conference on Computational Intelligence and Cybernetics Yogyakarta, 2013.
[5] Stefan Trommer, et al. (2016), Autonomous Driving: The Impact of Vehicle Automation on Mobility Behaviour, Institute of Transport Research (www.ifmo.de); at http://bit.ly/2kIAOOQ.
[6] Ackerman, E. (2016b), "*Lidar that will make self-driving cars affordable*", IEEE Spectrum, Vol. 53, p. 14.
[7] Wardrop, M., 2009, "Driverless Vehicles could be on Britain's Roads in 10 years" , Telegraph.

[8] S. Marra, M. A. Iachino and F. C. Morabito, "Tanh-like Activation Function Implementation for High-performance Digital Neural Systems," 2006 Ph.D. Research in Microelectronics and Electronics, Otranto, 2006.