Integration of Cloud Computing and Artificial Intelligence in Driverless Car Using High Performance Behavioural Cloning

Prabhat Kumar
M. Tech.Scholar
Dept.of CSE
Arya college of Engg. & I.T.
mishra.anupam.wit@gmail.com

Prof.(Dr.) Vishal Shrivastava
M.Tech co-ordinator
Dept.of CSE
Arya college of Engg. & I.T.
vishalshrivastava.cs@aryacollege.in

Abstract:- Research on autonomous vehicle system has risen in the past couple of years. It has posed challenges to the researchers to develop an understanding about the real time scenarios. Deep Learning has demonstrated its extraordinary computational potential by transcending its abilities into more complex areas, where pattern matching, image recognition and behavioral cloning plays a vital role.

The system consists of Image Processing and analyzing of the training data into behavioral cloning of the vehicle in a simulated environment. A control algorithm responsible for consolidating the sub systems calculations of the correct steering angle is used to keep the vehicle within the lane markings of the road. The solution proposed requires a better data collection and data interpretation. Addition of cloud computing fastens the data calculation and hence improves the performance of the system.

Keywords:- *Image Processing, Cloud Computing, Control Algorithm, Behavioral Cloning, Pattern Matching*

*****

I. INTRODUCTION

Artificial intelligence [1] and machine learning are at the forefront of autonomous technologies. Deep learning is being applied as a mean of enabling autonomy in vehicles. For this purpose Udacity has released the simulator as open source software for the enthusiasts to teach a car how to drive using only camera images and deep learning. Driving a car in an autonomous manner requires learning to control steering angle, throttle and brakes. Behavioral cloning technique is used to mimic human driving behavior in the training mode on the track. That means a dataset is generated in the simulator by user driven car in training mode, and the deep neural network model then drives the car in autonomous mode. A large amount of data hence been generated can be processed with the help of cloud computing in the Google's Colaboratory Project. Though the models performed well for the track it was trained with, the real challenge was to generalize this behavior on a second track available on the simulator.

II. BACKGROUND STUDY

A. Machine Learning [2] is a dominant field within artificial intelligence and it's commonly defined as a "field of study that gives computers the ability to learn without being explicitly programmed". Machine learning can be divided into two primary categories:

- Supervised Learning: The goal of supervised learning is to learn a mapping between inputs to outputs, which yields the desired output for all inputs used in a set of labeled input/output-pairs. It includes reinforcement learning [3].
- Unsupervised Learning: Unsupervised learning is the training of an artificial intelligence algorithm using information that is neither classified nor labeled and allowing the algorithm to act on that information without guidance [4].

B. An Artificial Neural Network (ANN) [5] is a network composed of several simple processing units, referred to as neurons that are interconnected between each other. Each neuron in the ANN produces a sequence of activations that propagates through the network, activating neurons by weighted connections to itself and other previously activated neurons in the network. The output neurons of the ANN may influence the surrounding environment by triggering actions, where learning of the network is perceived when the weights of the network are adjusted in such a way that the ANN approaches the desired behavior when these actions are triggered.

- Feed-forward Neural Network: A feed-forward neural network is defined as a neural network where the connections only go in one direction. The current layer to the adjacent forward layer, but not other way around [6]. It distinguishes itself from recurrent neural networks by not having any feedback loops in its architecture.
- Fully-connected Neural Network: A fully-connected layer is defined as a net- work where every unit is connected to every unit of its adjacent forward layer.[6]
- Loss function: A loss function is used in supervised learning for calculating the difference between the output of an ANN and the correct output, specified by the supervisor.
- Gradient Optimization: The value of the loss function defines a search direction based on the ANN's weights which are shifted towards. The size of the shift is determined by a parameter known as the learning rate.[7]
- Back propagation: Back propagation applies the gradient optimization by propagating backwards through the network, adjusting the weight of the network to compensate for the error calculated by the loss function.[8]

C. A Convolutional Neural Network (CNN) have many similar features of regular neural networks, it consist of processing units that apply a process of learning that uses weights and biases1 and process data by adjusting its connections. In some cases the network thereafter sends the processed data through a squashing non-linearity that maps the output value to a specified range, more commonly known as a squashing function. It also employs a loss function as a measurement for observing and optimizing the learning rate of the network [9].

D. Augmentation and Image processing: In a real-life situation, we can never train a self-driving car model for every track possible, as the data will be too huge to process. Also, it is not possible to gather the dataset for all the weather conditions and roads. Thus, there is a need to come up with an idea of generalizing the behavior on different tracks. This problem is solved using image preprocessing and augmentation techniques, which will be discussed in the following section:

- Crop: The images in the dataset have relevant features in the lower part where the road is visible. The external environment above a certain image portion will never be used to determine the output and thus can be cropped. Approximately, 30% of the top portion of the image is cut and passed in the training set.
- Flip (horizontal): The image is flipped horizontally (i.e. a mirror image of the original image is passed to the dataset). The motive behind this is that the model gets trained for similar kinds of turns on opposite sides too.
- Brightness: To generalize to the weather conditions with bright sunny day or cloudy, lowlight conditions, the brightness augmentation can prove to be very useful.
- Shadow: Even after taking into considerations the light conditions, there are still chances that there are shadows on the road. This will give an instance of half lit and half lowlight scenes in the image.

- Random Blur: To take care of the distortion effect in the camera while capturing the images, this augmentation is used as an image captured is not clear every time. Sometimes, the camera goes out of focus, but the car still needs to fit that condition and keep the car steady. This random blur augmentation can take such scenarios into consideration.

E. For the purpose of implementation some important libraries were used as follows:

- Tensorflow: An open-source library for dataflow programming. It is widely used for machine learning applications. It is also used as both a math library and for large computation. Keras: A high-level API that uses Tensorflow as the backend is used.
- Keras facilitate in building the models easily as it more user friendly. Different libraries are available in Python that helps in machine learning projects.
- Numpy: Provides with high-level math function collection to support multi-dimensional matrices and arrays. This is used for faster computations over the weights (gradients) in neural networks.
- scikit-learn: A machine learning library for Python which features different algorithms and Machine Learning function packages.
- OpenCV (Open Source Computer Vision Library): It is designed for computational efficiency with focus on real-time applications. In this project, OpenCV is used for image preprocessing and augmentation techniques.

F. For the purpose of cloud computing following platform were used:

- MiniConda Environment: It is an open source distribution for Python which simplifies package management and deployment. It is best for large scale data processing.
- Google Colaboratory: Colaboratory is a Google research project created to help disseminate machine learning education and research. It's a Jupyter notebook environment that requires no setup to use and runs entirely in the cloud. This means that as long as you have a Google account, you can freely train your models on a K80 GPU.
- Git: It is a distributed version-control system for tracking changes in source code during software development. It can be used to track changes in any set of files. Its goals include speed data integrity and support for distributed, non-linear workflows.

III. METHOD FOR DATA COLLECTION AND BEHAVIORAL CLONING

A. Experimental Configuration

- The sequential models built on Keras with deep neural network layers are used to train the data.
- Models are only trained using the dataset from Track_1.
- Epochs = 10, i.e. number of iterations or passes through the complete dataset.
- Batch-size = 300, i.e. number of image samples propagated through the network, like a subset of data as complete dataset is too big to be passed all at once.
- Learning rate = 0.0001, i.e. how the coefficients of the weights or gradients change in the network.
- ModelCheckpoint() is the function provided in Keras to save checkpoints and to save the best epoch according to the validation loss.

B. Model

There are different combinations of Convolution layer, Time-Distributed layer, MaxPooling layer, Flatten, Dropout, dense and so on, that can be used to implement the Neural Network models. The model used here is a tuned and optimized in order to suit our simulator and learn well. The model uses five convolutional 2D layers followed by the flatten layer and four dense layers.

- 2D convolution layer: This layer creates a convolution kernel that is convolved with the layer input to produce a tensor of outputs.
- Flatten Layer: A “flatten” operation on a tensor reshapes the tensor to have a shape that is equal to the number of elements contained in the tensor.
  - Dense Layer: A dense layer is just a regular layer of neurons in a neural network which receives input from all the neurons in the previous layer, thus densely connected.

C. Data Collection

The simulator was started and the car in simulator was driven by using keyboard keys so as to train the car and generate the training data. Convolution neural network works behind as to monitor the controlled operation and movement of the vehicle. Images where captured through three camera mounted on the car. Each set of image depicts left, right and centre. A corresponding steering angle is also recorded.

D. Training Process

The data collected from the supervised mode is uploaded to the github by creating a repository. This data is then called for analysis and interpretation by the Google’s Colaboratory Project where the data passes through the model to create a training file. This file is then been used to predict the motion of car while in autonomous mode.

IV. RESULT AND ANALYSIS

While collecting the training samples from the training data we find that the car drives in a straight direction for the most of the time which is similar to the real world scenario. So the number of 0's in steering data is huge. In order to avoid this, the data should be shuffled and rearranged. Also the repeated data should not be considered for training purposes. Please refer figure 1and figure 2.

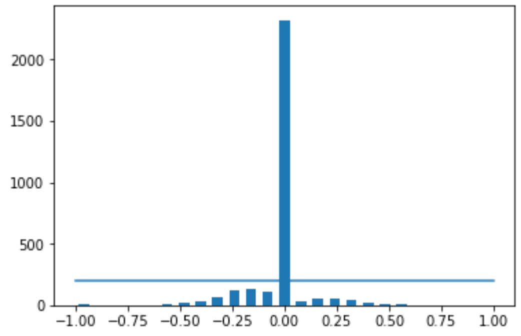

Fig.1. Repeated Data (0’s)

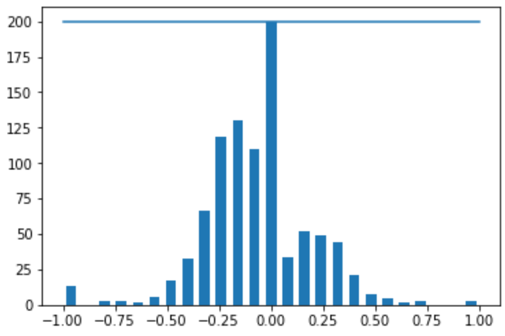

Fig.2. Uniform Balanced Data

Another observation was loss over each epoch. Keras provides “val_loss” as value loss, which is the average loss after that epoch. The loss observed during the initial epochs at the beginning of training phase is high, but it falls gradually, and that is evident by the table 4.1.

S.No.	Epoch	Training	Validation
1	1	0.1126	0.06
2	2	0.078	0.0626
3	3	0.0709	0.0643

4	4	0.0608	0.0572
5	5	0.0542	0.0571
6	6	0.0461	0.0537
7	7	0.0418	0.0414
8	8	0.0384	0.0359
9	9	0.0348	0.036
10	10	0.0317	0.0317

TABLE 4.1.
Training Data

A graph was plotted to Loss over Epoch from the training data where the training loss and validation data are converging.

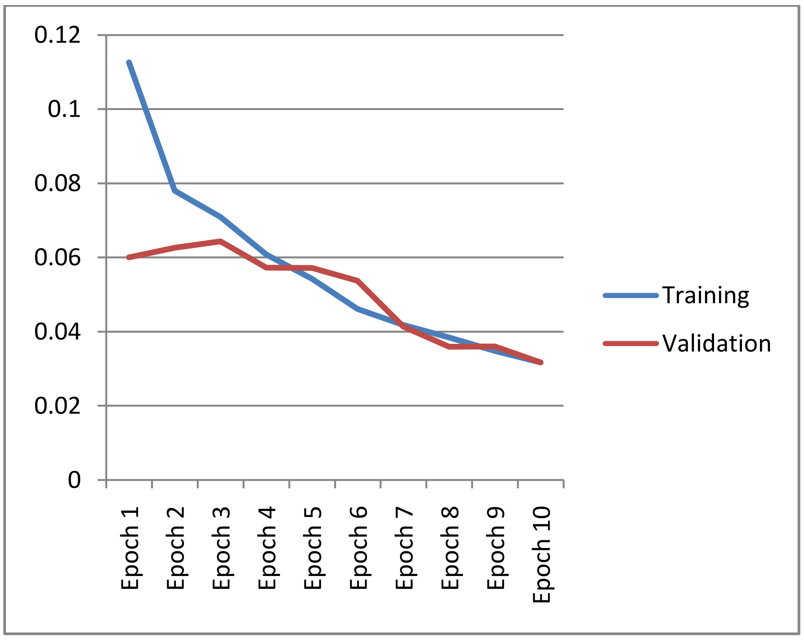

Fig.4.3. Graph plotting Loss over Epoch from the training data

III. CONCLUSION

This project started with training the models and tweaking parameters to get the best performance on the tracks and then trying to generalize the same performance on different tracks. The model performed best on 1 track did average on Track_2; hence there was a need for more image augmentation and processing to achieve real time generalization. The use of CNN for getting the spatial features in the image dataset makes this combination a great fit for building fast and lesser computation required neural networks.

REFERENCES

[1] S. Russell and P. Norvig, Artificial Intelligence: A Modern Approach, 3rd ed.Upper Saddle River, NJ, USA: Pearson Prentice Hall, 2009.

[2] A. Munoz, "Machine Learning and Optimization", Courant Institute of Mathematical Sciences, 2014[Online].Available: https://www.cims.nyu.edu/~munoz/files/ml_optimization.pdf

[3] K. P. Murphy, "Machine learning: A probabilistic perspective" Cambridge, MA, USA: MIT Press, 2012

[4] E. Alpaydin, “Introduction to machine learning”, 3rd ed. Cambridge, MA, USA: MIT Press, 2014.

[5] J. Schmidhuber, “Deep learning in neural networks: An overview”, Neural Networks, vol. 61, pp. 85–117, Jan. 2015, doi:10.1016/j.neunet.2014.09.003. [Online]. Available: http://www.sciencedirect.com/science/article/pii/ S0893608014002135

[6] S. Haykin, “Neural Networks and Learning Machines”, 3rd ed. Upper Saddle River, NJ, USA: Pearson Prentice Hall, 2008

[7] C. Asplund, “Object classification and localization using machine learning techniques: Designing and training models for use in limited hardware- applications,” M.S thesis, Department of Physics, Chalmers University of Technology, Gothenburg, Sweden, Jun. 2016.

[8] M. Riedmiller, “Advanced Supervised Learning in Multi-layer Perceptrons From Backpropagation to Adaptive Learning Algorithms,” *Int. Journal of Computer Standards and Interfaces*, vol. 16, no. 3, pp. 265–278, Jul.1994, doi:10.1016/0920-5489(94)90017-5.[Online]Available: http://www.sciencedirect.com/science/article/pii/0920548994900175?via%3Dihub