Zaka Capstone Project

Behavioral Cloning for Self-Driving Cars Using AI Technology

Rana Jasser, Maryam AlAmri

Abstract

A project that relies on the latest techniques of artificial intelligence to reproduce and mimic human behavior, and therefore to apply it to a self-driving car, as well as determine and detect the speed based on the environment surrounding the car.

Introduction | Problem Definition

Many inventions have appeared with the increase and development of technology and artificial intelligence, including self-driving cars. These cars work by scanning the road, taking some pictures, and analyzing them, and based on these analyzes the car simulates human behavior in driving.

Self-driving cars are defined as those that operate independently of a driver. The first truly autonomous vehicles were supported by DARPA (Defense Advanced Research Project Agency) in the 1980s. The field has undergone a fundamental shift as a result of developments in computer vision and machine learning over time. Planning the best driving route to take and cloning human behavior are the keys part of self-driving technology. It has been demonstrated that planning the driving route using behavioral reproduction is a successful method for training and continuously improving the route.

Using machine learning algorithms, behavioral cloning replicates human behavior based on visual kinetic policies. The concept of transfer learning has been applied to behavioral reproduction in the context of self-driving cars in recent years.

It has become very important to train self-driving cars to drive independently, with high efficiency, and determine the speed and route themselves based on the information that has been trained, to maintain the safety of their users.

The solution to the safety problem in self-driving cars is to train a model to drive a self-driving car on a simulated track. The ability of the model to drive a car is recognized by replicating the behavior of a human driver. Training data is obtained from a human driving simulation and then fed into a deep learning network, which learns to respond to each frame encountered in the simulation.

Overview | Previous Solution | Objective

In our project, we cloned the correct human behavior while driving, and analyzed the resulting data for use in training the device on self-driving. We also added an important feature, which is to analyze and determine the speed based on the images that the device is trained on, a model that feeds the 3 images to detect the speed and improve the model.

The goal is to simulate human driving behavior on a simulator using a model developed using deep neural networks. To mimic how a human drives, the technique is known as Behavioral Cloning.

Many engineers solved the problem, they get a simulator that has two tracks as well as two modes: training mode and autonomous mode. The dataset is generated by the user when driving the car in training mode in the simulator. They tried to achieve the same precision and accuracy on real-time data and they mimicked the steering angle.

Project Workflow | Experimental setup

The collab:

https://colab.research.google.com/drive/1zjI7qlWqEcUfvMPr5eUi3vprXXS1kKrM?usp=sharing

For this project, we used it as inputs: The driving aspects of the simulator have been designed in such a way that they imitate the presence of three cameras on the car. The three cameras on the front of the car are in the center, right, and left, and they capture continually while we record in training mode.

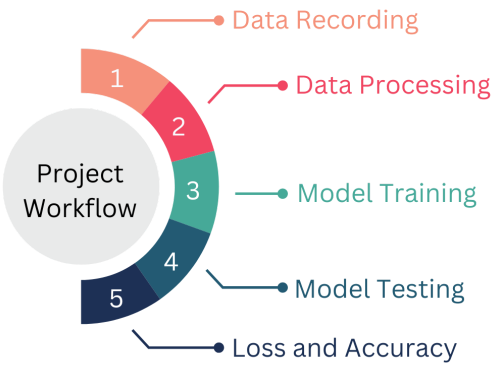
The features are the image paths with corresponding steering angle (0 depicts a straight, positive value is a right turn and a negative value is a left turn), throttle (acceleration), brakes (declaration), and speed of the car.

And the desired outputs: We aim to achieve the same precision and accuracy on real-time data in the autonomous car (tested data) as in trained data and to predict the best speed based on the 3 images, from left, right, and center.

The objective of this dataset-based project is to train a neural network to operate an autonomous driving agent on the tracks of Udacity's Car Simulator environment. First, we take care of the steering angle, throttle, and brakes when driving an autonomous vehicle. The behavioral cloning approach is used in the track's training mode to mimic human driving behavior. In the simulator, a user-driven car creates a dataset while in training mode, and the deep neural network model then uses that dataset to drive the car on its own.

In order to create a machine learning model, we used deep neural networks. The CNN model and Keras were used to build these models. Once trained, the model provides the server with the steering and throttle inputs necessary for autonomous driving. These modules, or inputs, are sent back to the server where they are used to control the car autonomously in the simulator and keep it from deviating from the path. By analyzing the speed based on the images that the device is trained on, we have made a significant contribution.

A model is being trained to operate a vehicle autonomously on a simulated track. To teach the model how to operate the car, a simulation of a human driver is used.



Data recording:

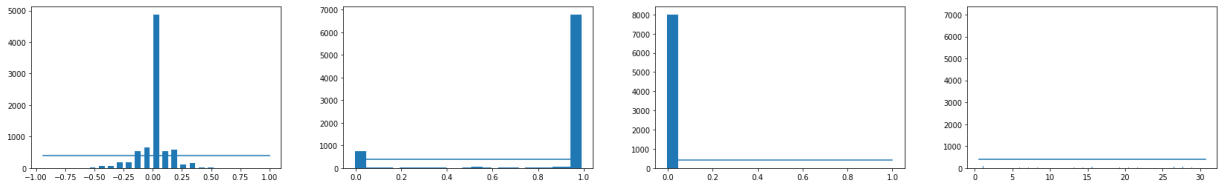
Both training mode and autonomous mode are available. Driving through the tracks while in training mode and saving the driving data in a folder is how training data is gathered. A trained model is tested in the autonomous mode.

Our dataset contains the following features (left, center, right, steering, throttle, speed) stored in an excel file and 24108 images.

https://drive.google.com/file/d/1wmzk3dJnhJ99yhR8eInuQFq0VZWk5ewv/view?usp=share_link

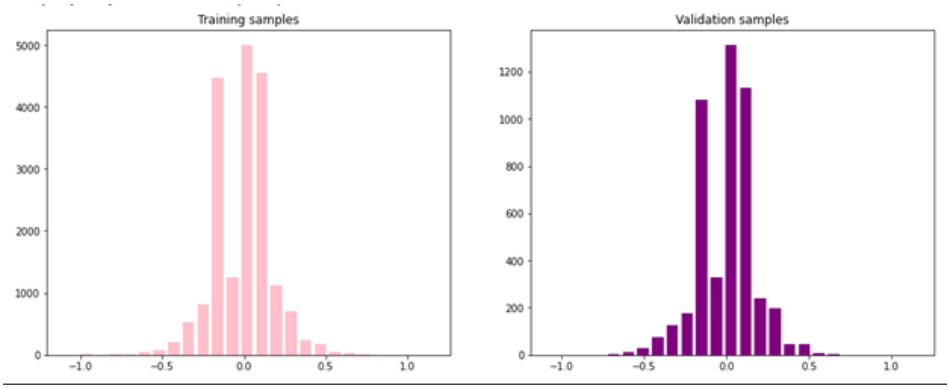
https://drive.google.com/drive/u/0/folders/15ZV5uh9ykGXw0fYf2e6YNNvCPhhCLXYt?lfhs=2

Firstly we import the libraries and connect our collab to our dataset, then we will divide the number of values to 25, odd number, to get center distribution we will use 400 samples to plot the histogram for the steering angle, throttle, brake, and speed.



Then we load the image into an array using locd_img_steering function and we take the image path as an empty list and steering as an empty list and we loop through the image path and steering.

Then we split the samples for the built-in function and we plot the histogram of training and validation samples.



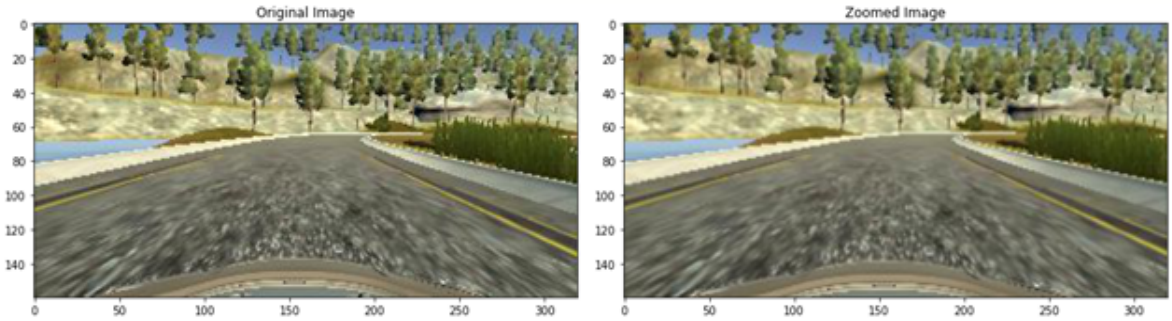
Data Augmentation and image preprocessing

In practice, we can never train a self-driving car model for every possible route since the data would be too large to analyze. Furthermore, it is not possible to collect the dataset for all weather

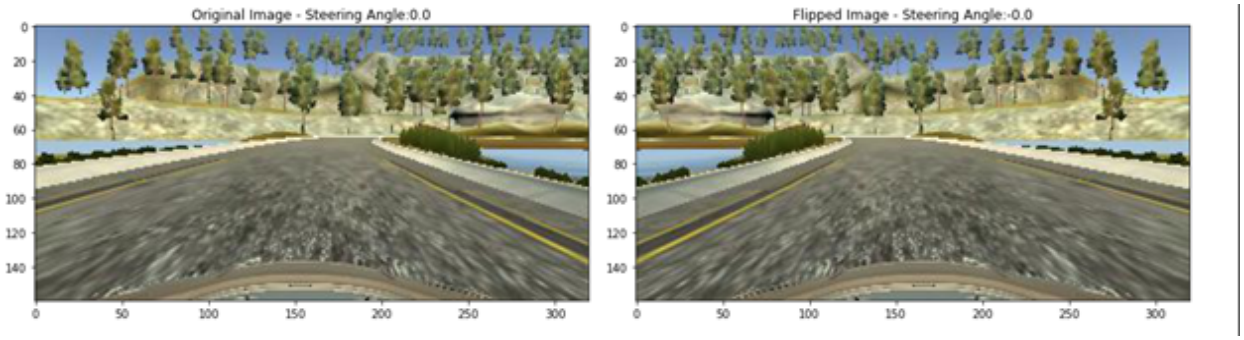
conditions and roads. As a result, an idea for generalizing the behavior on other tracks is required: Data augmentation and image preprocessing.

Data processing is carried out to make it simple for our model to work with raw data for training. The data processing in this project is integrated into a generator (keras fit generator) to enable real-time data processing.

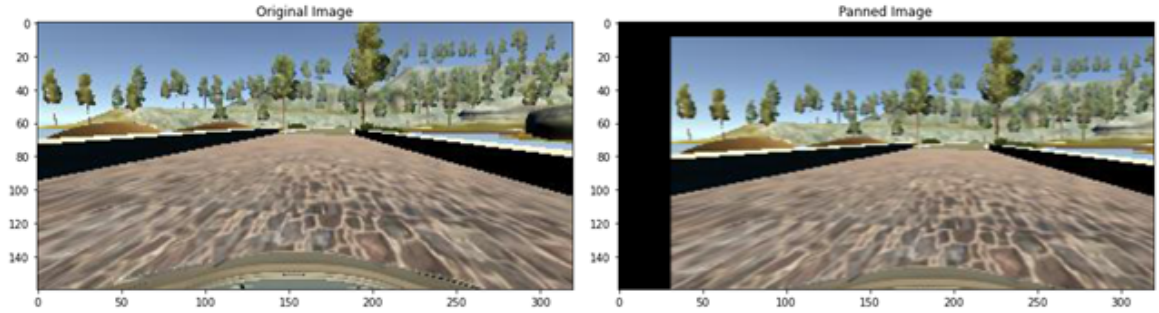
From our images, we need the route: 70% from the lower part of the image, so we will cut 30% of the top portion of the image.



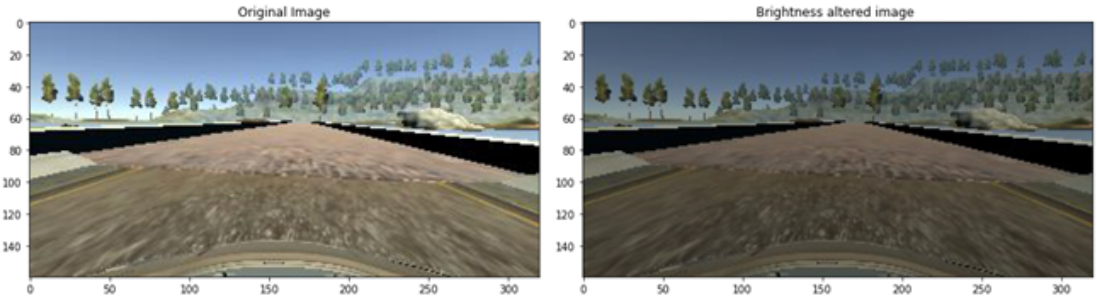
The picture is horizontally inverted (the dataset receives a mirror image of the original image). This serves to train the model for turns of a similar nature on opposite sides as well.



Slightly shift the image horizontally and vertically.

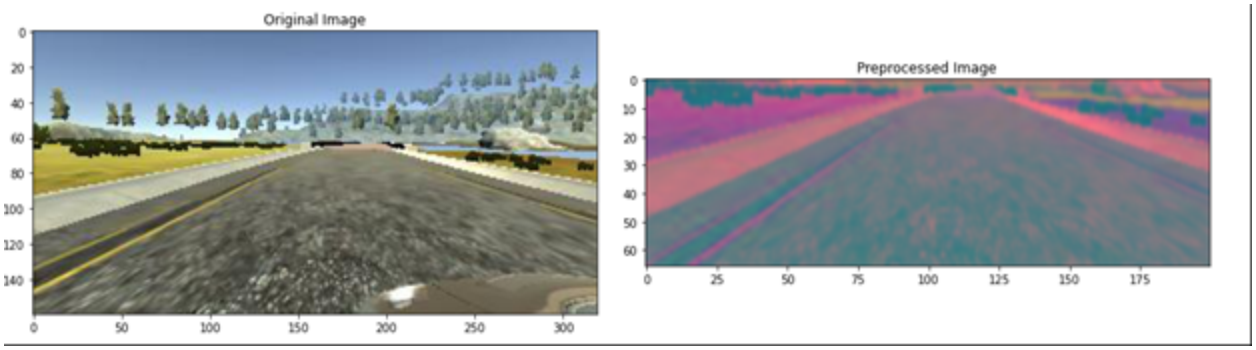


The brightness augmentation can be very beneficial when applied generally to weather conditions with bright sunny days or cloudy, low illumination situations. So, we make the



lighting brighter.

And now we compare original and preprocessed images



Speed detection

We import the libraries and we collect the training data. Then, we built a for loop to combine the left, center, and right images into one image.

Then, we split it to train and test data and we built a neural network architecture using relu activation function then we calculated the error on the tested data.

Neural Network | Model training | CNN

A normalization layer, five convolutional layers, and three fully connected layers are among the network's nine layers. Image normalization is carried out by the network's top layer. By performing network normalization, the normalization scheme can be modified to fit the network architecture and be accelerated using GPU processing.

In the first three convolutional layers, we use stridden convolutions with a 2x2 stride and a 5x5 kernel size, and in the final two layers, we use non-stridden convolutions with a 3x3 kernel size.

Using aggressive Dropout (0.5) on all layers and L2 Regularization (0.001) on the top layer, overfitting was minimized. The dropout separates the convolutional neural network and the fully connected layers. The flatten convert the output from the previous convolution layers into a 1D array. Three fully connected layers came after the convolutional layers and then output fully connected layer with 1 neuron.

The optimization was carried out by an Adam optimizer. As a result of the learning rate being able to adapt, little to no tuning is necessary.

As this is a regression-type example, we used model.compile() to compile our architecture. The metrics we used are mean squared error and optimize as Adam. To improve accuracy, we'll be using a relatively slow learning rate. To prevent the data from being overfitting, we will use a dropout layer. During each update, the Dropout Layer sets the input of a random subset of nodes to "0." As it is forced to use various node combinations in order to learn from the same data during this, we will generate the training data. The convolution layer must be separated from the fully connected layer with a factor of 0.5 added so that it converts 50% of the input to 0.

Loss and Accuracy

The average loss following each epoch is provided by Keras as "val loss" to determine value loss over each epoch. We used 10 epochs, 100 for batch size, knowing that we split them into 80% for training and 20% for testing.

Discussion | Outcomes

Ultimately, training a model to operate a self-driving car on a simulated track will solve the safety problem with self-driving cars. By mimicking a human driver's actions, the model's capacity to operate a car is determined. A deep learning network is fed training data from human driving modeling, which the network then uses to train itself to react to each frame of the modeling.

The outcome we achieved from behavioral cloning is to use a model created using deep neural networks to imitate human driving patterns on a simulator. After data recording and augmentation, we built a CNN model to predict the steering angle and the speed of the car based on the images from the front of the car, then we calculated its accuracy.

Conclusion | Results, and Implications

In conclusion, our project is still under implementation and development, in order to give good results and high accuracy. With high performance and acceptable in the environment in which it will be implemented.

We created this website, when the user implements 3 images of the road, it predicts the speed based on the CNN model:

https://drive.google.com/drive/folders/1BEd4Zy_mXpZAhaRBf2wwv-Q6mVPR8mO8

Using data from a sample of human behavior, a deep learning model was trained to operate autonomously on a simulated track. The model, which has 3 additional fully connected layers in addition to 5 convolutional layers, was able to generalize response to a new test track and learn by mimicking human behavior.

Behavioral cloning accuracy in practice will ensure safety benefits for self-driving car users in different weather and health conditions. It will, quite literally, remove the driver from the road, reducing the risks associated with risky behavior such as impaired, drunk, or drugged driving, reckless speeding, and disturbed driving.

Elders and people with disabilities, such as those who are blind, paralyzed, or have intellectual impairments, will have more autonomy and independence thanks to autonomous vehicles.

fewer accidents as well as less ability for cars on the road, which are the two causes of traffic and congestion. Reduced traffic from lane closures will result in fewer auto accidents.

Potential Direction

In our project we predicted the speed and the steering angle, and we can also work on another project where we can predict the throttle, brake, and lane lines.

An important consideration to take while building autonomous car models is decision-making. We must decide what can be done in a situation while taking into account the current circumstances. The best option must then be selected after weighing the advantages and disadvantages of each scenario. The automated system must generate multiple decision-making

paths, and it selects the best one based on factors like viability, safety, legal implications, efficiency, and comfort,...

References

[1]*https://link.springer.com/referenceworkentry/10.1007/978-0-387-30164-8_69#:~:text=Behavioral%20cloning%20is%20a%20method,input%20to%20a%20learning%20program*.

[2]*https://www.geeksforgeeks.org/convolutional-neural-network-cnn-in-machine-learning/*

[3] *https://towardsdatascience.com/behavioural-cloning-applied-to-self-driving-car-on-a-simulated-track-5365e1082230*

[4]https://www.mosaic51.com/featured/11-benefits-of-self-driving-cars-how-will-your-life-improve/

[5]https://www.thinkautonomous.ai/blog/path-planning-for-self-driving-cars/