Self-Driving Car Navigation, Implementation and Comparison using Machine Learning Algorithm and Deep Learning Algorithm

Shruti Godghate[1], P. G. Chilveri[2], S. S. Gaikwad[3], Harshvardhan shende[4],

E&TC Department, SKNCOE, SPPU Pune, India

[1]pgchilveri.skncoe@sinhgad.edu
[2]suchita.gaikwad_skncoe@singhad.edu
[3]godghateshruti6@gmail.com
[4]harshubs4@gmail.com

***Abstract-* Vehicles are the most preferable means of transportation for both work and household living in today's society. Automobiles are used for a wide range of purposes, resulting in a rising number of road accidents. By building a self-driving car prototype and installing machine learning algorithms such as CNN and SVM, we will be able to determine which machine learning method has the best accuracy in this suggested system. This will require the use of a processor on which the car will operate. The Raspberry Pi is the CPU we're using. Images collected by the camera will be the system's input database. We will detect the obstacles and the path's road lines using the provided database. This data will then be given to the processor as an input, where it will be processed using machine learning techniques such as CNN and SVM, and the output will be sent to the self-driving car's motor driver and motor. Then, based on the results of our proposed theory, we can calculate the model's accuracy as well as monitor obstructions and lane detection using some parameters. Then we'll have to repeat the operation with a different algorithm, which we'll have to implement the same way we did before. As a result, we can determine which method (Algorithm) has the better accuracy, and this will be the project's final conclusion.**

Keywords- Self driving car, Machine learning algorithms, safety measures, object detection.

I.INTRODUCTION

An autonomous car is a vehicle that is able to sense its surroundings and move safely without or minimal human intervention. Self-driving cars rely on powerful sensors, algorithms and processors to run software. By installing machine learning & deep learning algorithms like SVM and CNN, we'll find out which one is more efficient. Our processor will use input from the camera video and that input will be processed further for lane & object detection process, after that will be used to navigate. The input is the images collected by the camera/video that serves as the database. This enables the objects and the lines of the road to be recognized. The input is further processed with machine learning & deep learning algorithms such as SVM and CNN to make an appropriate decision. Here we will compare both algorithms through implementation. With this proposed theory, we can further compute with some parameters. By the results, we can determine which algorithm is more efficient by comparing with accuracy, Time required to process, objects detected, capacity to process data, etc. and that will be our final conclusion.

II. LITERATURE SURVEY

Wuttichai Vijitkunsawat presented a comparison of 3 famous algorithms of machine learning: SVM, ANN-MLP, and CNN-LSTM on different scenarios and different speed levels. From the experiment, it can be seen that the percentage of the accuracy rate of the CNN-LSTM algorithm is the highest efficiency. T. Banerjee, S. Bose, A. et. al. presented a unique embedded controller design of a driverless, green energy powered, collision protected and GSM destination guided vehicle. A GPS module accurately tracks the location of the car, source and destination, and mapping the coordinates provides navigation Speeds of the vehicle is automatically controlled by keeping a safe distance, which is a function of speed, with the vehicle in front [5]. Nischal Sanil, Pasumarthy Ankith Naga Venkat et. al. proposed that for a driverless car, Accuracy and efficiency are directly proportional to the amount of training data and the quality of the object detection model. This concept calls for an advanced and safe future for all citizens in the field of transportation due to its ability to relay passengers or goods without human intervention [6]. Mostafa Zaman, Nasibeh Zohrabi, Sherif Abdelwahed presented that (CNN) uses the images with the specified lane and vehicle detection as an input. It predicts the intention of going right, left, or driving straight based on the relative car distance on the video or the image. The main idea is to integrate CNN with lane and vehicle detection modules to estimate a safety path progression for a specific amount of time from the video or image based on the relative distance from other vehicles. Simulation results are given to illustrate the effectiveness of the proposed approach [11]. Ittikon Thammachantuek, Mahasak Ketcham presented the performance comparison of popular supervised learning algorithms: SVM, MLP, CNN, DT,and RF. These algorithms are used for road images recognition. All images are collected by our model car. They are labelled with four different classes: left, right, forward and stop. We use 90% of them for training and use 10% for testing each model [4]. Yevgeniy Byeloborodov, Sherif Rashad proposed that Computer vision library OpenCV, image processing, machine learning techniques and neural networks will be combined and used to build this algorithm. This paper presents the current progress of this research project to design machine learning algorithms for behavioral prediction of objects for self-driving cars [12]. Divya Arora, Mehak Garg, Megha Gupta proved that, CNN is so vastly used, as, unlike traditional Neural Nets, it reduces thnumber of parameters and focuses more on domain specific features. There are various CNN architectures proposed, such as LeNet, AlexNet, GoogleNet. In this paper, we talk about the structure of CNN and all the models of CNN which are proposed till date [13].

III. PROPOSED METHODOLOGY

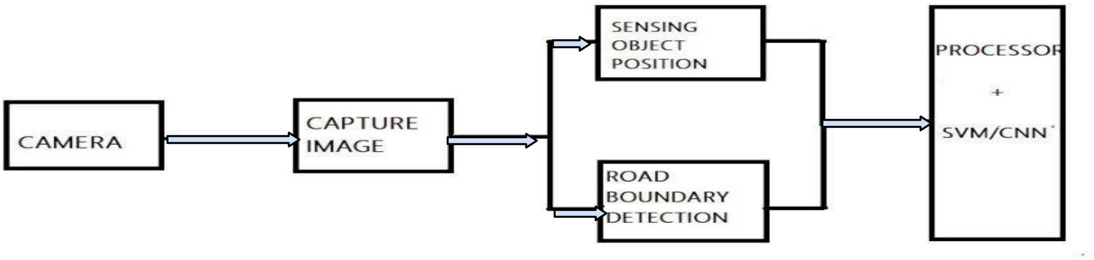


Fig.1. Block diagram of the proposed system

The workflow begins with the Images captured by the camera/ video will be given to the processor as an input, where it will be processed using machine learning deep learning techniques like SVM and CNN for IMAGE PROCESSING purposes. Mini camera /video is a small module for the processor to detect road marks and object detections. It is capable of 3280x2464 pixels, 8 Megapixels, and 160o diagonal fields of view. Relative Algorithms- machine Learning and Deep Learning algorithms like SVM, CNN etc.

IV. IMPLEMENTATION

A. *Database Creation:*

We have created a database for our system which is in the form of images taken with the help of code which we have implemented in our system. The created database will be helpful to train models and will be given as an input to the processor for image processing purposes. For which we have tested for both video as an input and input via camera.

B. *Object Detection:*

Object detection will be reflected by the red square around them in the result. For this we have used libraries such as open CV, NumPy and a few packages of them. Also, we have used mobile net SSD (single shot detection) to capture and (detect) the objects. With the help of all these packages from libraries we have succeeded to be able to detect objects from our given input.

C. *Training/Testing (SVM):*

Here, our algorithm will reshape the images (database), combine and hog features into a single array, Test the dataset, train the dataset and classify the database using SVC. After this it will also generate predictions in its environment by training. The Training of the model will take place in this step. Also, we are calculating its (Model training) accuracy and saving this dataset on the local disk.

D. *Navigation using SVM:*

Here we have used our System to navigate the car with the help of SVM algorithm, and it will show the output in the rectangular box with the appropriate predictions for navigation purposes by which the car will get to know how to react in the upcoming conditions. Here first it will first resize the frames as done earlier, before producing output from the processor.

E. *Navigation using SVM (using different environment):*

After completing the program testing using a given video, we have also experimented with 2 other different roads with different scenarios for live navigation testing purposes. First, we have trained our system with the help of a dataset created by the system itself. then trained the system and checked whether it's working or not (in terms of different surroundings). Afterwards we experimented with the unknown (i.e., un trained environment) to the system observed the results. In these experimentations we didn't provide the dataset to train for those specific surroundings. We did it on the basis of previous surroundings to check (test) its generality results. Here we have used our System to navigate the car with the help of SVM algorithm, and it will show the output in the rectangular box with the appropriate predictions for navigation purposes. by which the car will get to know how to react to the upcoming scenarios. Here first it will first resize the frames as done earlier, before producing output from the processor.

F. *Training/Testing (CNN):*

Here, our algorithm will be able to reshape the images (database), combine and hog features into a single array, test the dataset, train the dataset and classify the database using CNN. After this it will also generate predictions in its environment by training. We have done labelling the dataset, creating training data and processing the given test data. Also, we have created Neural network using

Tensorflow and finally setting up the features and labelling. Additionally, we are calculating its training accuracy of the model and saving this dataset on the local disk.

G. *Navigation using CNN:*

After experimenting with the SVM algorithm, we chose to implement our system through its paces with the CNN algorithm. We followed the same processes as before.

1) Creating Database (from given video in the form of images with code).

2) Training the system.

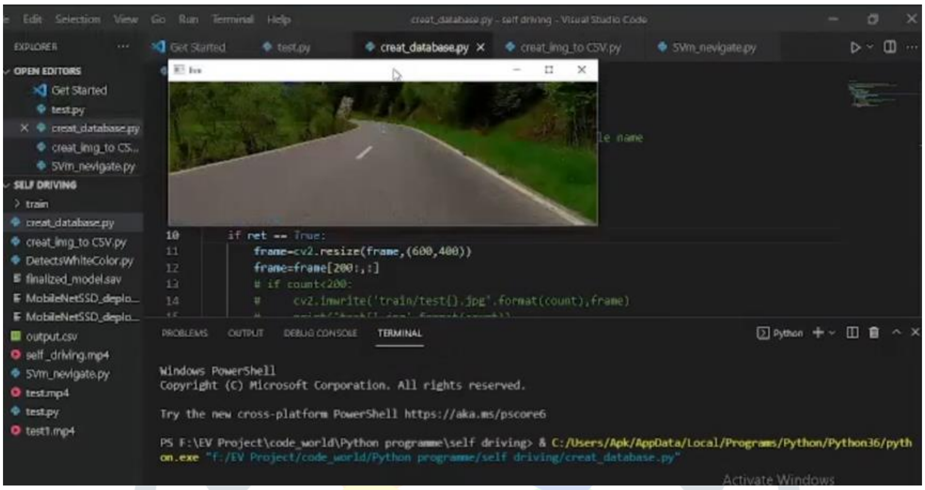
3) Experimenting.

4) and last but not the least, Observing Results (outputs of the program).

V. EXPERIMENTATION

A. *Database Creation:*

We have implemented and tested the algorithm for database creation. We have successfully developed an algorithm for creation of a database for our system. In which we used Python IDE and imported a few packages and libraries.



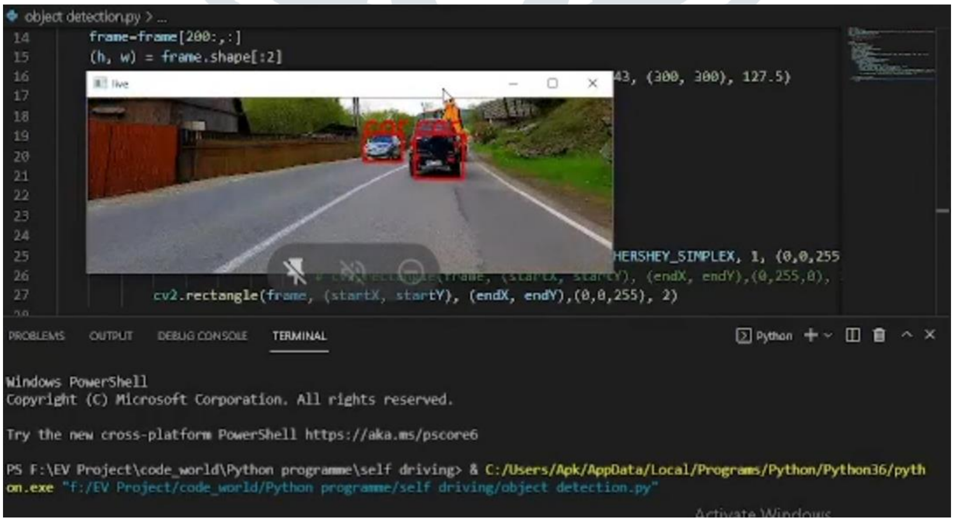
```
le name
        if ret == True:
            frame=cv2.resize(frame,(600,400))
            frame=frame[200:,:]
            # if count<200:
            #     cv2.imwrite('train/test{}.jpg'.format(count),frame)
```

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS F:\EV Project\code_world\Python programme\self driving> & C:/Users/Apk/AppData/Local/Programs/Python/Python36/pyth
on.exe "f:/EV Project/code_world/Python programme/self driving/creat_database.py"
```

Fig. 2. Database creation

B. *Object Detection:*

After successfully creating a database, we have tried to detect an object with the help of algorithms. We have successfully implemented an algorithm by which we can detect objects from video as an input as well as with input as a camera.



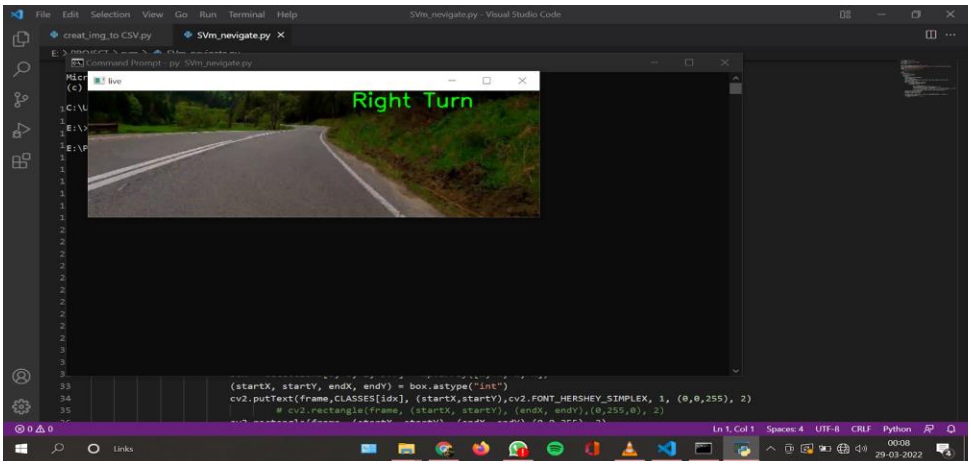
```
        frame=frame[200:,:]
        (h, w) = frame.shape[:2]
43, (300, 300), 127.5)
HERSHEY_SIMPLEX, 1, (0,0,255
            # cv2.rectangle(frame, (startX, startY), (endX, endY),(0,255,0),
            cv2.rectangle(frame, (startX, startY), (endX, endY),(0,0,255), 2)
```

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS F:\EV Project\code_world\Python programme\self driving> & C:/Users/Apk/AppData/Local/Programs/Python/Python36/pyth
on.exe "f:/EV Project/code_world/Python programme/self driving/object detection.py"
```

Fig. 3. Object Detection

C. *Navigation using SVM:*

While using Support Vector Machine (SVM) a Machine Learning algorithm, we have used a python IDE for program development, running as well as producing the outputs. This is how we have Implemented and Experimented the system with SVM.

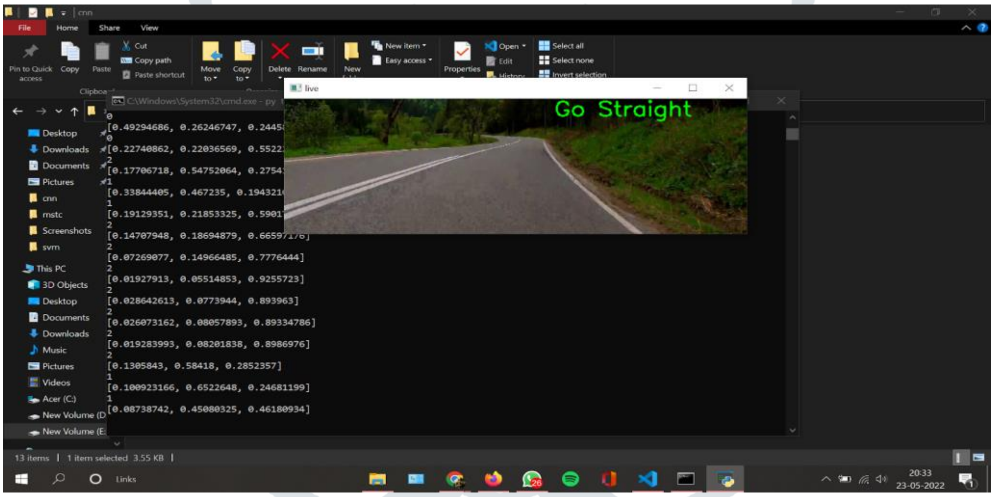

SVm_nevigate.py - Visual Studio Code
creat_img_to CSV.py
SVm_nevigate.py
Command Prompt - py SVm_nevigate.py
live
Right Turn
Ln 1, Col 1 Spaces: 4 UTF-8 CRLF Python
00:08
29-03-2022

```
(startX, startY, endX, endY) = box.astype("int")
cv2.putText(frame,CLASSES[idx], (startX,startY),cv2.FONT_HERSHEY_SIMPLEX, 1, (0,0,255), 2)
        # cv2.rectangle(frame, (startX, startY), (endX, endY),(0,255,0), 2)
```

Fig.4. SVM Experimentation

D. *Navigation using CNN:*

After experimenting with the SVM algorithm, we have constructed a code to train the module for Convolution Neural Network (CNN) and then experimented with this trained module for the CNN based navigation program that we have developed.



```
0
[0.49294686, 0.26246747, 0.2445
0
[0.22740862, 0.22036569, 0.5522
2
[0.17706718, 0.54752064, 0.2754
1
[0.33844405, 0.467235, 0.194321
1
[0.19129351, 0.21853325, 0.5901
2
[0.14707948, 0.18694879, 0.66597176]
2
[0.07269077, 0.14966485, 0.7776444]
2
[0.01927913, 0.05514853, 0.9255723]
2
[0.028642613, 0.0773944, 0.893963]
2
[0.026073162, 0.08057893, 0.89334786]
2
[0.019283993, 0.08201838, 0.8986976]
2
[0.1305843, 0.58418, 0.2852357]
1
[0.100923166, 0.6522648, 0.24681199]
1
[0.08738742, 0.45080325, 0.46180934]
```

Fig.5. Experimentation of navigation with CNN

VI.RESULT

TABLE I
Comparison of SVM and CNN

Parameters	**SVM**	**CNN**
Accuracy (as tested):	92.85%	98%
Time elapsed to process:	Took more time, 29.5 sec	Took less time, 18.92 sec
Ability to process	Low or medium database	Higher database
Object detection	7	7
Accuracy (with different surroundings without training module)	36.84 %	44.73 %

From our above results we can say that the CNN algorithm will be much effective to use as compared to SVM, as it is a deep learning algorithm which has many layers for processing and can predict the output more efficiently and accurately taking less time to process & even in case of huge data it works effortlessly, where SVM fails to process with such high margins of accuracy and timing.

CONCLUSION

Autonomous vehicles are both cost-effective and reliable, as well as speedy. It's also an innovative method that uses automation technology to require little or no human intervention and is effortless to use. In This system we compared machine learning/deep learning algorithms such as SVM and CNN in different scenarios. From our above implementation, experimentations and results we can say that the CNN algorithm will be much effective to use as compared to SVM, as it is a deep learning algorithm which has many layers for processing and can predict the output more efficiently and accurately taking less time to process & even in case of huge data it works effortlessly, where SVM fails to process with such high margins of accuracy and timing. Also, when we experimented the system with an unknown environment database without training the model with it, we observed the accuracy being dropped drastically.

REFERENCES

[1] Buehler, K. lagnemma, and S. Singh, “The 2005 DARPA Grand Challenge ", The great robot race, Vol. 36. Springer Science & Business Media, 2007.

[2] Juan Manuel Ramirez, Pilar Gomez-Gil and Filberto Lopez Larios, “A Robot-Vision System for Autonomous Vehicle Navigation with Fuzzy-logic Control using Lab-View”, Electronics, Robotics and Automotive Mechanics Conference, 2007

[3] Musashi Aoto, Yousuke Numata and Yasutaka Wada, “Development of an FPGA controlled Mini-Car toward Autonomous Driving”, International Conference on Field-Programmable Technology, pp.403-405, 2018

[4] Ittikon Thammachantuek; Somkiat Kosolsomnbat; Mahasak Ketcham “Comparison of Machine Learning Algorithm’s Performance Based on Decision Making in an Autonomous Car,” *2018 International Joint Symposium on Artificial Intelligence and Natural Language Processing (iSAI-NLP)*, 2018, pp. 1-6.

[5] T. Banerjee, S. Bose, A. Chakraborty, T. Samadder, B. Kumar and T. K. Rana, "Self driving cars: A peep into the future," *2017 8th Annual Industrial Automation and Electromechanical Engineering Conference (IEMECON)*, 2017, pp. 33-38.

[6] N. Sanil, P. A. N. venkat, V. Rakesh, R. Mallapur and M. R. Ahmed, "Deep Learning Techniques for Obstacle Detection and Avoidance in Driverless Cars," *2020 International Conference on Artificial Intelligence and Signal Processing (AISP)*, 2020, pp. 1-4.

[7] J. Jhung, I. Bae, J. Moon, T. Kim, J. Kim and S. Kim, "End-to-End Steering Controller with CNN-based Closed-loop Feedback for Autonomous Vehicles," *2018 IEEE Intelligent Vehicles Symposium (IV)*, 2018, pp. 617-622.

[8] W. -Y. Lin, W. -H. Hsu and Y. -Y. Chiang, "A Combination of Feedback Control and Vision-Based Deep Learning Mechanism for Guiding Self-Driving Cars," *2018 IEEE International Conference on Artificial Intelligence and Virtual Reality (AIVR)*, 2018, pp. 262-266.

[9] J. Newman, Z. Sun and D. -J. Lee, "Self-Driving Cars: A Platform for Learning and Research," *2020 Intermountain Engineering, Technology and Computing (IETC)*, 2020, pp. 1-5.

[10] N. M. Arshad and N. A. Razak, "Vision-based detection technique for effective line-tracking autonomus vehicle," *2012 IEEE 8th International Colloquium on Signal Processing and its Applications*, 2012, pp. 441-445.

[11] M. Zaman, N. Zohrabi and S. Abdelwahed, "A CNN-based Path Trajectory Prediction Approach with Safety Constraints," *2020 IEEE Transportation Electrification Conference & Expo (ITEC)*, 2020, pp. 267-272.

[12] Y. Byeloborodov and S. Rashad, "Design of Machine Learning Algorithms for Behavioral Prediction of Objects for Self-Driving Cars," *2020 11th IEEE Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*, 2020.

[13] D. Arora, M. Garg and M. Gupta, "Diving deep in Deep Convolutional Neural Network," *2020 2nd International Conference on Advances in Computing, Communication Control and Networking (ICACCCN)*, 2020, pp. 749-751.