International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 07 Issue: 10 | Oct 2020 www.irjet.net p-ISSN: 2395-0072

A Deep Learning Approach to Real-Time Driving Assistant System

Ameya Morajkar[1], Siddhant Medar[2]

[1]Student, Dept. of Computer Engineering, Vidyalankar Institute of Technology, Mumbai, India
[2]Student, Dept. of Electronics & Telecommunication Engineering, Vidyalankar Institute of Technology, Mumbai, India

---***---

Abstract - *Driving Assist System (DAS) is a real-time system since it reacts quickly to video feed input and prioritize the incoming information to prevent accidents. In this paper, our DAS particularly focuses on detecting traffic sign, vehicles with their relative distance and provide human-like general voice feedback. In the recent advancement of object detection, algorithms like Faster R-CNN, R-FCN, SSD, FPN, have provided result efficient enough to be implemented in real-time. This system can help to decrease the number of road accidents happening every year, as it would contribute to the safety of the drivers, pedestrians and vehicles.*

Key Words: Driving Assist Systems, Image Recognition, Mobilenet, SSD, Traffic Sign Detection, Vehicle Detection, Approximate distance predictor, Forward Collision Warning.

1. INTRODUCTION

Artificial intelligence (AI) and self-driving cars are bilateral topics in technology. Traffic sign & vehicle detection and recognition plays a crucial role in such expert systems, such as advanced driving systems as it instantly assists drivers or automatic driving systems to detect and recognize traffic signs effectively thus avoiding accidents. Usually its driver's responsibility to read traffic signs and make sure he doesn't collide into other cars. However, many adverse factors, such as bad weather, viewpoint variation, human distraction, wore out traffic signs, physical damage, etc., might affect the ability of driver to driver properly. The problems we are trying to address in this paper, are as follows.

1. Humans can only focus on one thing, so the chances that driver might miss a small road sign is quite high.

2. A fatigued driver might misjudge the relative distance between his car and the car in front of him.

3. Traffic sign discoloration, traffic sign damage, rain, snow, fog, and other problems, are also given challenges for driver in identifying them.

4. Traffic signs on some road often gets blocked by buildings, trees, and other vehicles; therefore, we needed to recognize the traffic signs with incomplete information.

All this problem can be addressed by introducing a system with a combination of Forward Collision Warning (FCW) system and Traffic sign detection system. A well-trained ML model can perform this task of multiple object detection more efficiently than humans.

In this research, we evaluated the performance of real time traffic sign detection with 40 different classes of German traffic signs by using the concept of transfer learning to detect custom objects. Transfer learning is a machine learning design methodology where we reuse a model developed for a task as a starting point for another task, this allows us to speed up training and improve the performance of our deep learning model. We created a custom dataset consisting of vehicles and traffic signs by dividing traffic video into numerous frames such that each frame contains at least one vehicle and/or traffic sign. Then the model training was performed on a Google Colab. Finally, we enhanced the ability of the model to adapt to real-world environment by using ultra realistic game like GTA 5.

2. BACKGROUND & RELATED WORK

In this section, we portray the different work carried out by others in areas which are relevant to our research. The subparts below are the most important key aspects in our research work.

Object recognition is a general term to describe a collection of related computer vision tasks that involve identifying objects in digital photographs.Image classification involves predicting the class of one object in an image. Object localization refers to identifying the location of one or more objects in an image and drawing a bounding box around their extent. Object detection combines these two tasks and localizes and classifies one or more objects in an image.

Traffic Sign Recognition: Almost all of the traffic sign recognition algorithms followed a two-stage convolution neural network architecture. The first stage consisted of detecting a traffic sign from the image frame using image processing principles such as shape detection, contour detection, etc. and the following stage classified the detected traffic sign on the basis of the model trained on the traffic signs dataset.

Distance Estimation & Collision Warning: Distance estimation is an important aspect of driving, without which fatal accidents may occur. In order to tackle the same, we have implemented a basic Forward Collision Warning (FCW) system, wherein we determine the relative distance of vehicle in front of the subject by comparing the relative area of object boundary box to the area of our entire screen [7].

Voice feedback: After successful detection of any traffic sign, the next step is to provide aid to the driver and the same is carried out by providing audio feedback. However, there are few challenges to do so such as playing the audio which is associated to the same frame, moreover, if the frame containing traffic sign is detected then the frame may wait for the voice feedback to completely speak out and then proceed to the next frame for processing which may cause a significant delay resulting in asynchronization with real time. Also, one more problem one could face is when there are multiple traffic signs in the same frame which may result in echo or irritation to the driver.

3. PROPOSED SYSTEM (IMPLEMENTATION)

The diagram below shows the overall flow the system.

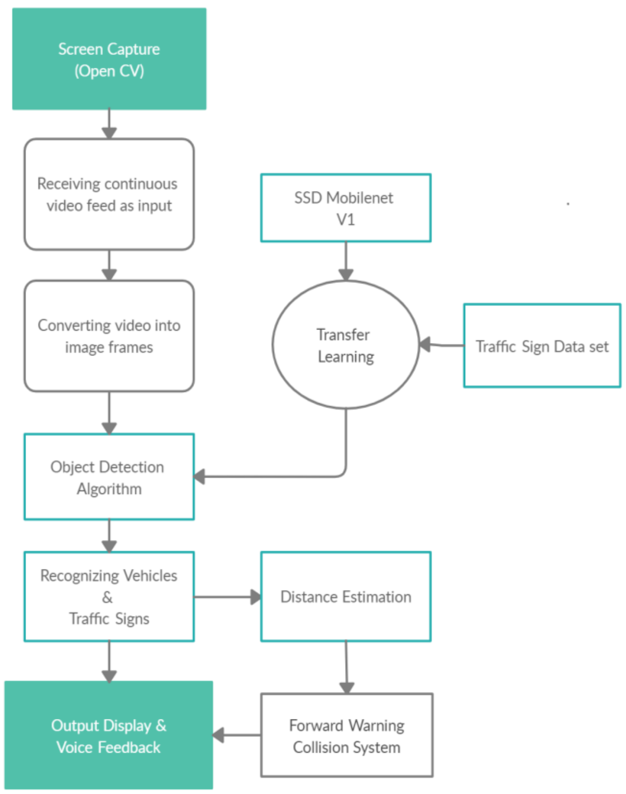

Fig.1. Workflow Diagram

Step 1: Grabbing the Screen to get the video input feed from virtual driving game.

Step 2: Video feed is then converted into individual image frames

Step 3: Object Detection is now applied on these frames to detect other vehicles and traffic signs

Step 4: Traffic signs and vehicles are recognized and appropriate labels are displayed.

Step 5: For vehicles, the area of the bounding box is used to determine the relative distance of the other vehicles.

Step 6: A output video feed with detected vehicles, traffic signs is displayed with audio feedback.

4. DATASET COLLECTION

The main part of creating an efficient Neural Network Model begins with collecting relevant and then preprocessing it.

The dataset was constructed by collecting suitable video games from virtual driving games like GTA 5 and Watch Dogs as our project mainly focuses on vehicle and traffic sign detection. Along with that, we also used German Traffic sign dataset [5].

Fig.2. Traffic Signs

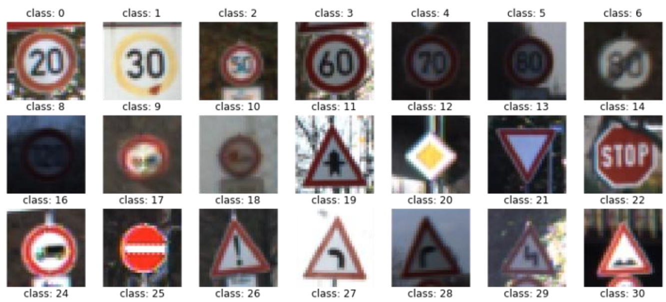

Fig.3. Traffic Signs (Training Data)

5. PREPROCESSING

We preprocessed the data by removing the frames which didn't contain any vehicles or traffic sign in it so as to reduce the make the dataset clean. Additionally, we prepared a .csv file which contained the coordinates of objects location in a given frame.

6. MODEL TRAINING

We chose transfer learning design methodology to achieve efficient results in a short span of time. Specifically, the pre-trained model, we used was 'SSD MobileNet V1'. This model was originally trained on the COCO (Common Objects in Context) Dataset.

MobileNet model uses SSD (Single Shot Detector) algorithm, which uses a matching phase while training, to match the appropriate anchor box with the bounding boxes of each ground truth object within an image. Essentially, the anchor box with the highest degree of overlap with an object is responsible for predicting that object's class and its location in the image.

Table 1. MobileNet Body Architecture

Type / Stride	Filter Shape	Input Size
Conv / s2	$3 \times 3 \times 3 \times 32$	$224 \times 224 \times 3$
Conv dw / s1	$3 \times 3 \times 32$ dw	$112 \times 112 \times 32$
Conv / s1	$1 \times 1 \times 32 \times 64$	$112 \times 112 \times 32$
Conv dw / s2	$3 \times 3 \times 64$ dw	$112 \times 112 \times 64$
Conv / s1	$1 \times 1 \times 64 \times 128$	$56 \times 56 \times 64$
Conv dw / s1	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 128$	$56 \times 56 \times 128$
Conv dw / s2	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 256$	$28 \times 28 \times 128$
Conv dw / s1	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 256$	$28 \times 28 \times 256$
Conv dw / s2	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 512$	$14 \times 14 \times 256$
5× Conv dw / s1	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
5× Conv / s1	$1 \times 1 \times 512 \times 512$	$14 \times 14 \times 512$
Conv dw / s2	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
Conv / s1	$1 \times 1 \times 512 \times 1024$	$7 \times 7 \times 512$
Conv dw / s2	$3 \times 3 \times 1024$ dw	$7 \times 7 \times 1024$
Conv / s1	$1 \times 1 \times 1024 \times 1024$	$7 \times 7 \times 1024$
Avg Pool / s1	Pool 7×7	$7 \times 7 \times 1024$
FC / s1	1024×1000	$1 \times 1 \times 1024$
Softmax / s1	Classifier	$1 \times 1 \times 1000$

Fig.4. SSD MobileNet V1 Architecture.

Since MobileNet is a pretrained model which was originally trained on the COCO dataset. In our project, we have modified the same model to detect vehicles, their relative distance and traffic signs using customized dataset as mentioned above.

With about 6000 images in our dataset, each image we trained the existing pre-trained model on Google Colab platform as it provides a single 12GB NVIDIA Tesla K80 GPU for a runtime of about 10 hours. During training, the average loss, the model came down to was 0.32 at a learning rate of 0.001. After every multiple iteration, final model was saved as graph file.

7. FORWARD COLLISON WARNING

Normally Advanced Forward Collision Warning (FCW) System use lot of sensors but in this project, we focused only on images to make a simple version of FCW. Here we made use of the pixel width of the object detection boundary box of the car in front of the subject and position of the car in the image. Comparing this to the whole size of image we can give an approx. relative distance from the car.

$$rel\_apx_{distance} = round\left(\left(\left(1 - (boxes[0][i][3] - boxes[0][i][1])\right)\right), 1\right)$$

This formula [6] allows us to get a distance value in range 0 to 1 where 0 being closest and 1 being farthest. Now this relative approx. distance can be set as a threshold for warning system. We set a threshold of 0.3, i.e. when a car gets too close enough, distance < 0.3 and its position is in front of us, a warning message will be displayed. This message will be displayed on the bounding box of car itself.

Fig.5. Forward Collision Warning System

Fig.6. Vehicle Detection and Approximate Distance.

8. VOICE FEEDBACK

For voice feedback assistance, we have used Google Text-to-Speech (gTTS) python library [8] which generates sound

International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 07 Issue: 10 | Oct 2020 www.irjet.net p-ISSN: 2395-0072

based on the text. gTTS (*Google Text-to-Speech*), a Python library and CLI tool to interface with Google Translate's text-to-speech API. We have used gTTS to provide suitable audio output either if a traffic sign is detected or in event of a warning signal by FCW.

9. RESULT

Below is the demonstration of the traffic sign recognition and vehicle detection inside virtual driving game.

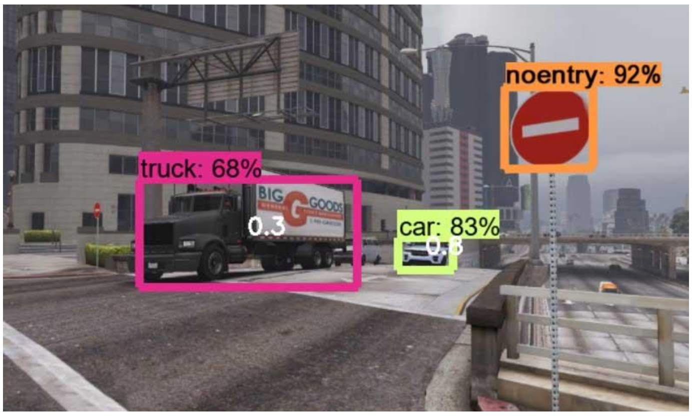

Fig.7. Forward Collision Warning System

In the above frame, we see that there are 2 category of vehicles – car & truck, which are detected with an accuracy of 68% and 83% respectively. Along with that the model also detects traffic signs on the street with an accuracy of 92%.

10. CONCLUSION

We have presented the idea of basic FCW system, traffic sign detection using pretrained object detection model. Our system provides audio feedback which gives the feel of an assistant to the driver. Our intent is to provide aid to the driver, so as to prevent accidents by means of basic FCW system. Moreover, the system also provides audio feedback for recognized traffic signs and warnings generated by FCW.

11. FUTURE SCOPE

The scope of the project can be further improved by incorporating detection for additional types of vehicles and obstacles such as pedestrians or trees. This driver assist system can be combined with a self-driving system to enhance both the efficiency and safety of the system.

REFERENCES

[1] End-to-End Deep Learning for Self-Driving Cars By Mariusz Bojarski, Ben Firner, Beat Flepp, Larry Jackel, Urs Muller, Karol Zieba and Davide Del Testa – Nvidia Developers https://images.nvidia.com/content/tegra/automotive/images/2016/solutions/pdf/end-to-end-dl-using-px.pdf

[2] Self-Driving RC Car – Zheng Wang https://zhengludwig.wordpress.com/projects/selfdriving-rc-car/

[3] Object Detection for Autonomous Vehicles. By Gene Lewis, Stanford University http://web.stanford.edu/class/cs231a/prev_projects_2016/object-detection-autonomous.pdf

[4] Chengji Liu ,Yufan Tao ,Jiawei Liang ,Kai Li ,Yihang Chen, “Object Detection Based on YOLO Network”, 2018 IEEE 4th Information Technology and Mechatronics Engineering Conference (ITOEC 2018)

[5] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel. The German Traffic Sign Recognition Benchmark: A multi-class classification competition. In Proceedings of the IEEE International Joint Conference on Neural Networks, pages 1453–1460. 2011..

[6] Priya Garg, Debapriyo Roy Chowdhury, Vidya N. More, “Traffic Sign Recognition and Classification Using YOLOv2, Faster RCNN and SSD 10th ICCCNT 2019 July 6-8, 2019, IIT - Kanpur,Kanpur, India. IEEE-45670..

[7] Sentdex Object detection and distance estimation https://pythonprogramming.net/detecting-distances-self-driving-car/

[8] “Google Text-to-Speech”, Available: https://gtts.readthedocs.io/en/latest/

[9] “OpenCV tutorials”, Available: https://docs.opencv.org/2.4/doc/tutorials/tutorials.html

[10] “TensorFlow Object Detection API tutorial”, Available: https://tensorflowobject-detection-api-tutorial.readthedocs.io/en/latest/