Ambulance detection using image processing and neural networks

K Agrawal¹, M K Nigam¹, S Bhattacharya¹, Sumathi G¹,*

¹SENSE School, Vellore Institute of Technology, Vellore, Tamil Nadu, India - 632014

*sumathi.g@vit.ac.in

Abstract. Ambulance Detection using Image Processing and Neural Network is a vehicle detection and tracking system, which recognizes the vehicle (i.e., Ambulance in this case) amidst the traffic congestion. Due to the fact from past few years, the range of vehicles usage of the road is growing each day that results in traffic congestion, for better management of this traffic this system is useful. Traffic Congestion, as mentioned above, can be observed at an ever-growing pace in countries like India and Thailand, where the roads’ width and length make it impossible to make a separate lane for the emergency vehicle (like that of ambulance); Hence making it hard for the vehicle to pass through the traffic at the earliest possible time. The Ambulance tracking system is activated at the mapped junctions and that program detects the ambulance coming close to it and turns the traffic light to Green for the next 15 seconds. Geocoding is the practice of transforming addresses (like a physical address) to location information (like longitude and latitude) that can be used to locate a label on a map or to mark a grid. They plan to provide ambulances with this software to make it easy to transform addresses into a programmable format for review and retrieval. This data is converted to a system that shows all the crossings it must pass to meet the endpoint.

Keywords— Neural Network, Image Processing, Vehicle Detection, Tracking system, Geocoding

1. Introduction

We propose a method that offers much more reliable vehicle detection with less time required compared to other deep convolutional approaches. The device is designed to track vehicles in traffic through video surveillance. Unlike other convolutional neural network approaches that identify specified objects, the modified convolutional deep learning approach increases the accuracy of detecting ambulances [2]. The time taken is, therefore, smaller than the other processes, where YOLOv3 is an exception in the case of time. The updated deep convolutional approach often requires fewer parameters, making the device more effective without overfitting [1]. In the process, the features are extracted from different layers of CNN (convolutional neural network) from the given input image after the pre-set stage. The proposals are divided into equal-sized regions which track small-scale and large-scale proposals [4]. At the end of the day, the global average pooling and the predicted bounding boxes for the detection outcome are carried out. The framework uses traffic
representations as inputs that involve various types of cars and other road items. The resulting output is the ambulances observed from the traffic data. The plans are based on the CNN network [7].

After the image detection, just to specify the detection process and narrow it down to the ambulance, we will do an audio detection. In case of occurrence of any extra sounds, it can be quickly determined for the elimination. This can be considered as a future expansion and will not be shown in this demonstration [5]. The next step of the process is to do bilateral filtering, where using this weighted intensity of the neighbouring pixels obtained from the previous step, the intensity value of each pixel is saved [6]. The pixel values of the edges are stored in this step. Next step after saving high-intensity edge values, the unnecessary irregular signals are deleted here in the next stage of complete variation. Highly variable signals with, which are vulnerable to, suspicion is eliminated [8]. Consequently, unnecessary specifics are omitted, and edges are avoided. The data from this pre-set module is forwarded to the next CNN layer, where further filtering is done here. The regional proposal network is a specialized and specific architecture in which proposals for a specific area are created by sliding a small network over a convolutionary feature map [3]. The target is in this line. This idea is chosen if there is a high likelihood of the existence of the event. These features are derived from these ideas using the ROI pooling process before being fed into the convolutional network. Given the 5×5 function map and the ROI filter, this is added. The filter output size is 2×2. The suggested area is (0,0) and (5,4).

The use of RFIDs and Bluetooth in the existing work has been used in vehicle detection for a long time. These devices however have a few drawbacks when being used for detecting vehicles:

- The number of devices required to install for vehicle detection is increased.
- Since more devices are bought, the cost is high.
- Due to the low response speed, the connection takes time to be established.
- Have a low range, hence it requires to be placed at every diametrical distance from the last device, for greater accuracy.

2. Proposed Method

In our work, the only piece of hardware we propose to use is the surveillance camera itself. However, the reason for using YOLOv3 and CNN is that it makes our work much more compatible with countries such as India and Thailand, where road width is irrelevant for separate ambulances. The use of YOLOv3 and CNN in conjunction with surveillance cameras is useful because:

- The amount of hardware to be installed is reduced to a single WIFI module connected to a microcontroller connected to the camera.
- Costs are drastically reduced because the same microcontroller can be used to connect 45 cameras per connection.
- Since it is the camera that detects and transmits data remotely, the response speed is much higher than the actual response speed at the intersection.
- Since this is the surveillance camera we use, the range is very high, which allows us to see a large amount of traffic in a single scan.
Figure 1. Block Diagram of use of YOLOv3 and CNN

the given diagram displays how the images are going to the YOLO program, then it is detecting whether the vehicle is a truck or not. Since we are using CNN, it helps us to generate our dataset, wherein the filtered data from the YOLO program comes as input (i.e., trucks). With our dataset generated, using CNN, we detect whether the detected truck is an ambulance or not.

3. Implementation

Our work can be broadly divided into 3 major phases –
- Training
- Detection
- Tracking

The videos captured by the camera sensor are converted into images and these images are processed by various algorithms and finally, the program gives the output whether the vehicle detected is an ambulance or not. Based on the final output, the location of the Ambulance is tracked, and traffic lights are altered.

3.1 Training

Before the detection of the vehicle and giving output based on the result, the machine must be taught how an ambulance looks like. For this, we created a tiny dataset consisting of a thousand images, and a model must be trained which will be called in the confirmation segment.

TensorFlow 2.1.0 algorithm is particularly used to avoid compatibility issues and is installed. Validation and trained images generated after training are stored in their respective folders in google drive and the drive is linked to the training algorithm. Matplotlib is used to give the output graphs.

3.2 Pre-processing

More than 1000 images were downloaded from various internet sources and given as input for training to generate a trained dataset that can be used for detection. “ImageDataGenerator” function is used to manipulate the existing images and make more copies, to increase the dataset size. “Train_generator” and “Valid generator” functions are used to resize and segregate the images and save them in Train and Validation folders in the drive, both have 2 folders each - 0 and 1. Images that have ambulance are saved in 1 and others in 0. Batch size for training the data is set as 32 and the images are resized to 224x224 pixels.
3.3 Training
After the pre-processing, the standard TensorFlow algorithm runs to train the dataset. Using Keras open-source library, a sequential list is created. As we are working on CNN, 4 layers of 2D convolutions are set, and after each convolution – Max-pooling operation runs, and the least relevant results are dropped out. The whole process runs 10 times (epochs = 10) and takes a maximum of 15 minutes.

3.4 Combining
After training the whole data is combined and optimized using adam optimizer and losses are calculated to make the model more efficient. Using the Fit generator, the model is created, and later, it is saved in the drive folder. The whole model is based on the metrics accuracy and loss and the graphs for the same are plotted.

3.5 Detection
It takes place in several steps
- Camera Sensor captures small video clips of traffic, these Clips are taken as input and are sent to the processor.
- Clips are converted into pictures and are sent for processing. First, they are saved in the “overall” folder of the attached google drive.
- Images in the overall folder go to the YOLO program that detects if the vehicle is a truck. It is a predefined algorithm that uses the COCO dataset developed by Microsoft. YOLO algorithm detects that the image that is passed through it contains are truck or not. (figure 1)
- If the vehicle qualifies as a truck, then a copy of these images is saved in the “detected” folder after adding a bounding box around it.
- The images are cropped on their bounding boxes and the cropped image is saved in the “crops” folder.
- The image then passes to the main function, where our model is called, and the image is processed to check if it is an ambulance.
- If it qualifies as an ambulance then, yes is returned by the main function.
- On returning yes, a message is sent of the respective mobile no. for conveying that an ambulance is detected and then the images are saved in the “final” folder.

3.6 Tracking
Once the program knows that the ambulance is detected, then the tracking algorithm help to track where exactly and on which road the Ambulance has been found. Based on the location, in real-world scenarios, the traffic lights of that roadside can be turned to GREEN and the rest can be turned to red for 15 sec till the next detection. Here, for demonstration purposes, the 15-second rule is not applied, rather we check for the ambulance every 60 frames of the video, which is approximately every 2.55 seconds, and the whole process is repeated per intersection.
We use the matplotlib library for plotting the given detection and have collected 8 videos for the demo, only one with an ambulance (figure 2). For each point that detected the ambulance, we enable 8 points around that to start detection and we distribute 8 demo videos to those points simulating surveillance in the real world, and only one of them will detect the presence of an ambulance, and that point is plotted as a green mark in the grid. Using this, we have tracked the movement of the ambulance for five iterations for demonstration purposes.
4. Results and Discussions

Figure 2 – Depicts that our algorithm has detected the ambulance and makes a boundary box around it, as a result of object detection by image processing. It also shows the accuracy of our model, since it is not detecting the red lights of the truck coming from behind nor the white vehicle, which is standing on the left side of the road.

![Ambulance being scanned by the camera.](image)

Figure 2. Ambulance being scanned by the camera.

![Mapping of an ambulance in the grid.](image)

Figure 3. Mapping of an ambulance in the grid.
Figure 3 – Depicts the point mapping system of the object detected. The green points indicate the positive response of the algorithm of the object being detected and red being the negative one.

![Model Accuracy Graph](image1)

**Figure 4.** Model Accuracy Graph; Accuracy v/s epoch

Figure 4 – shows the accuracy of our model for test and train datasets. The average accuracy which can be derived from the graph is 78.00%, we can say that it has a fair level of accuracy.

![Model Loss Graph](image2)

**Figure 5.** Model Loss Graph; Loss v/s epoch

Figure 5 – shows the loss values of our graph, between test and train datasets. The average loss which can be observed from the graph is 46.00%, this loss needs to be improved in the future application of the idea.
5. Conclusion

In this proposed work, we were successfully able to:
- Prepare a dataset for the program to train and assess from it.
- The model so formed, was used by the program to identify the Ambulance amidst all the vehicles.
- Upon sensing that the captured vehicle is an ambulance, we successfully got the program to turn that traffic light into green, while others were turned red.
- The quick sensing of the camera sensor allows the ambulance, an emergency vehicle, to move quickly and reach its destination in much less time.
- Simulation for the entire program is successfully done, once the parts, Training, Detection, and Tracking were done separately.
- With Geocoding as an option in our work, there is still scope for improvement in our work, wherein we can trace the data of the ambulance to the shortest path of the nearest hospital.
- The future scope of the work can be implemented for other emergency vehicles as well.

6. Acknowledgment

We would like to thank our family for supporting us and our faculty for guiding us in this paper and for constant encouragement. We would also like to thank our University – Vellore Institute of Technology, Vellore.

References

[1] Visualizing Conical Intersection Passages via Vibrionic Coherence Maps Generated by Stimulated Ultrafast X-Ray Raman Signals (2020) Daniel Keefer, Thomas Schnappinger, Regina de Vivie-Riedle, Shaul Mukamel
[2] Vision-based vehicle detection and counting system using deep learning in highway scenes (2019) Huansheng Song, Haoxiang Liang, Huaiyu Li, Zhe Dai, and Xu Yun
[3] Fast car Crash Detection in Video (2018) V. Machacha, E. Laura Riveros
[4] Intersection Warning System for Occlusion Risks Using Relational Local Dynamic Maps (2017) Florian Damerow, Yuda Li, Tim Puphal, Benedict Flade, Julian Eggert
[5] Use of High-Resolution Google Earth Satellite Imagery in Landuse Map Applications (2016) K Malarvizhi, S. Vasantha Kumar, P. Porchelvan
[6] Image and map-based detection of vehicles at intersections (2013) Bunyo Okumura
[7] An Exploratory Cartographic Visualisation of London through the Google Maps API (2008) Maurizio Gibin, Alex Singleton, Richard Milton, Pablo Mateos & Paul longle
[8] A Google-Map-Based Arterial Traffic Information System (2007) Yao-Jan Wu; Yinhai Wang; Dalin Qian