THINK INDIA JOURNAL

ISSN: 0971-1260
Vol-22-Issue-3-July-2019

Car Speed and Number Plate Detection

Pardeep Kumar
Assistant Professor
Lovely Professional University
Pardeep.25237@lpu.coin

Ankit Anand
Computer Science Engineering
LPU
ankanand219@gmail.com

Piyush Kumar
Computer Science Engineering
LPU
Piyush.imp.kp@gmail.com

Aman Kumar
Computer Science Engineering
LPU
amankumar99.ak@gmail.com

Abstract - Detect cars, their speed and their number plate in a real time environment. In this we have used convolution neural network(CNN) to detect the cars, the speed of the car and also number plate of the car. For Number plate detection we have used the Optical character recognition(OCR). We have trained a CNN model and also used the OpenCV library to solve the problem.

I. BACKGROUND

Vehicle detection and speed detection systems have become more common in the computer vision community in recent years. The most important processing steps in these systems are the extraction of image sources in video frames and their classification using classifier of trained components to determine the class of an object.

The vehicle speed detection was previously measured by taking the distance between camera and the vehicle to measure the vehicle speed across the frame. Specifically, for speed cameras, they use the concept Doppler technology to calculate changes in microphones reflected by vehicles for speed.

Number plate recognition is one of the computer vision technology for licensing number of vehicle images. It is a system that contains several applications and challenge. This system can be used in many fields like parking areas, tolls, public entrances, private entrances and theft control etc. This mainly done by Neural Network and using tools like OpenCV.

II. DEFINITION OF PROJECT GOALS

The goals of our project have been defined as follows:

- Vehicle Detection: Being able to identify and locate vehicles in a video sequence.
- Speed Detection: Detecting the speed of the identified vehicles (km/hr) in a video sequence.
- Number Plate Detection: Being able to detect the number plate of cars through photograph/video frames.

III. SCOPE OF PROJECT

In this project we aimed to detect the cars using deep convolutional networks (CNN) and detecting the speed of the detected cars. We have also implemented the detection of Number plate of the cars using python tesseract on video frames screenshots.

The project focuses on identifying cars/vehicles in real time environment using video, by training a neural network to spot vehicles and by placing a bounding box over the vehicles detected in the frame. Two other features for our project are,

ISSN: 0971-1260
Vol-22-Issue-3-July-2019

one is calculating the speed of the vehicle using the relative pixel co-ordinates of the vehicles detected in respective frames divided by the focal length and the other is detecting the number plate of the vehicle in which we click the image from camera then load it on a system after that we use the OpenCV library. Then we used tesseract library by python to recognize different characters from images. When the vehicle's number plate is detected.

IV. PROBLEM DECOMPOSITION

This project was decomposed into the following sub-tasks:

A. Vehicle Detection

In this section, vehicle boundaries (primarily Cars) were identified automatically using Deep Learning via convolutional neural networks. We trained CNN models to identify the car's boundaries on frames of the video. This took training and optimizing the model specifically for vehicle classification.

B. Speed Detection

Keeping the track of relative pixel positions in every single frame and identifying the changes in positions of cars detected by using pre-trained classifier and then converting them to real object parameter, the speed of vehicle was detected.
To generalize the direction in which the vehicle is subjecting to change in pixel values, Manhattan distance iscalculated between the resulting pixel value and the centroids of the vehicles in the frames.Then the value is converted to object plane.

C. Number Plate Detection

Number Plate Detection uses the method of Optical Character Recognition (OCR) to read the characters as strings on a vehicle license plate. It takes theimage of a vehicle as the input and gives the output as the characters written on the plate. We are using machine learning for the character recognition aspect i.e map a character image to its actual character.

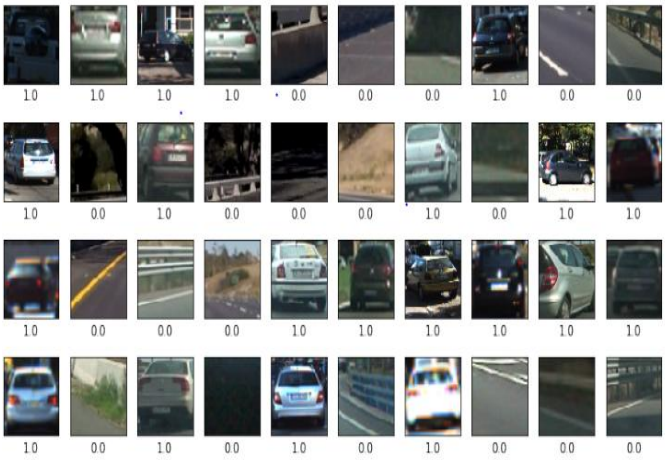
V. IMPLEMENTATION

In our approach to solving the problems mentioned in the problem decomposition, we have utilized a combination of neural networks and traditional computer vision processing techniques in order to develop our prototype.

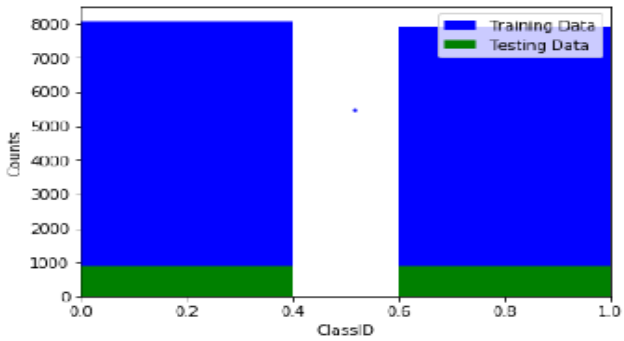
A. Design & Algorithms

Vehicles and cars are detected using the concept of deep learning and with a full convolution network (CNN), details are explained below:

The data set is explored here. It comprises of images has been taken from the KITTI vision benchmark suite, GTI vehicle image database, and the examples were extracted from the test video itself. The dataset is labelled with two classes i.e. cars and non-cars.
Cars have a label of **one**.0, whereas non-cars have a label of 0.0, as can be seen from the following figure:



A total of 17760 samples were acquired from the above mentioned dataset, each image is colored and has a resolution of 64x64 pixels. The dataset was then further split into the training set i.e. 90% and validation set i.e. 10%.
From the following distribution, it can be seen that the dataset is nicely balanced, which is important for training the neural network. This balance ensures that the neural network is not biased towardseither of the classes. The distribution looks as follows:



The Neural Network's architecture is explained here. This is essentially a binary classification problem, where the neural network has to decide whether the given image is a vehicle or not. The model has 10 layers excluding the input and output layers. The architecture of the model is as follows:

ISSN: 0971-1260
Vol-22-Issue-3-July-2019

- Input layer, here the 64x64 pixels were fed with all the 3 color channels
- Convolutional Layer 1 & dropout, a 2-dimensional conv layer with a 128 filter, 50% dropout and ReLU activation
- Convolutional layer 2 & dropout, a 2D conv layer with a 128 filter, 3x3 each, 50% dropout and ReLU activation.
- Convolutional layer 3, dropout & maxpool, a 2D conv layer with 8x8 max-pooling, 50% dropout and ReLU activation.
- Dense convolutional layer, 128 dense neuron layer with a ReLU activation
- Dense neuron layer, with hyperbolic tangent (tanh) activation

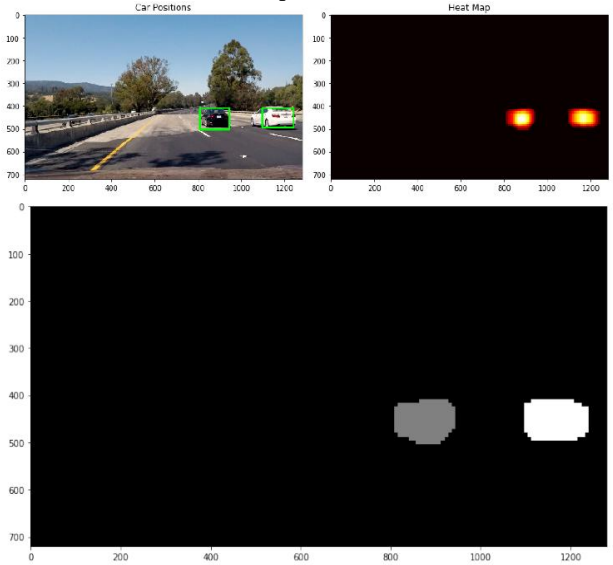
The full Neural network architecture is summarized below:

Layer (type)	Output Shape	Param #
lambda_1 (Lambda)	(None, 64, 64, 3)	0
conv1 (Conv2D)	(None, 64, 64, 128)	3584
dropout_1 (Dropout)	(None, 64, 64, 128)	0
conv2 (Conv2D)	(None, 64, 64, 128)	147584
dropout_2 (Dropout)	(None, 64, 64, 128)	0
conv3 (Conv2D)	(None, 64, 64, 128)	147584
max_pooling2d_1 (MaxPooling2	(None, 8, 8, 128)	0
dropout_3 (Dropout)	(None, 8, 8, 128)	0
dense1 (Conv2D)	(None, 1, 1, 128)	1048704
dropout_4 (Dropout)	(None, 1, 1, 128)	0
dense2 (Conv2D)	(None, 1, 1, 1)	129

Total params: 1,347,585
Trainable params: 1,347,585
Non-trainable params: 0

Every frame from the video was fed into the Neural Network, which was used to detect vehicles all over the frame. This network scales up to be compatible, no matter the input size is, as there is no fully-connected layer at the end, but just a conv layer with max pooling and dropout.
Since there is no single-neuron output, the neural network outputs an image like a virtual heat-map. The bounding boxes were then drawn on the hot positions as can be seen below:

A heatmap was created and a tiny threshold was added to avoid false positives. And a single bounding box can be drawn over each detected heat source, which essentially
Detects the cars within the picture, as can be seen below:



The network parameters (weights) were frozen and stored for future use, since the neural network has to be trained only once. These weights were then merged with Google Tensorflow pre-trained Object detection models and were later used in predictions.

Once the vehicle was detected, the remaining part that was yet to be solved was the speed of the vehicle. In order to achieve this, the video was read using Scikit video package and all video frames with vehicle being detected and bounding boxes drawn around with the help of the Cascade Classifier, were output to a folder. Subsequently, the video frames were iterated over one by one and their coordinates or absolute pixel values of the positions of the detected vehicles were obtained, using the HOG Classifier in order to track the ROI of the vehicle present in every frame.

ISSN: 0971-1260
Vol-22-Issue-3-July-2019

Once the regions of interest of all the vehicles in the frames were obtained, then midpoints of the bounding boxes which are surrounding the vehicles were obtained.This is held as theposition ofthe vehicle which was used as key for future frames. This process was repeated for all vehicles to create a list of midpoint values which were updated every frame.

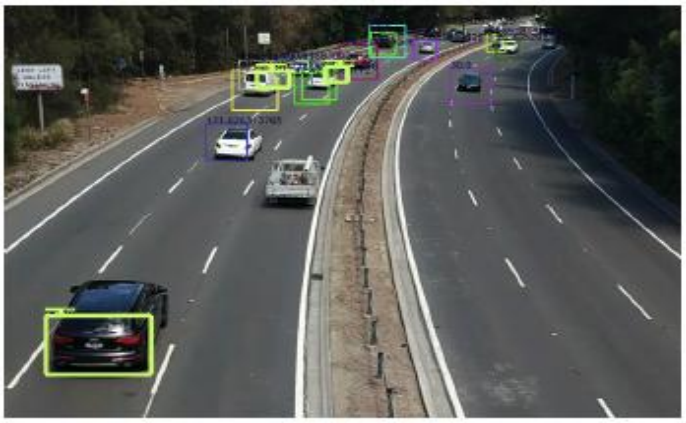
In order to find out the speed of any object the basic parameters that will be needed are distance and time. In this project, we instead are making all calculations in the image plane and then later projecting them in the object plane.In this case, the distance a particular vehicle has travelled in the image plane would be the relative differences in pixels at various vehicle positions in subsequent frames.Time would be the rate at which the frames were being processed, which we assumed to be 30 frames per second for the purposes of this project.

From the above description, ***Time = No. of pixels moved/progressed byframes per second*** and ***Distance = squareroot(midd1*midd1+midd2*midd2)*** where midd1 and midd2 are the positions of the vehicle in frame 1 and frame2 respectively. If the difference is less than 10 pixels, then the car is assumed to be stationary/idol and gets neglected.

Once the speed of the car within the image plane is calculated, we then convert the pixel values which is obtained into km/sec which represents the object plane, using the distance of the object from the camera in the given formula exact **distance to object (mm) = real height of the object * focal length * image height (pixels)/sensor height * object height (pixels).**

However, the trained classifier is unable to find all the cars in each single frame and so the obtained speed values aren't continually correct and subject to deviate from original value depending on the following assumptions on the parameters. The height of the object in object plane is assumed to be 4meters in general approximating to median values for all cars which is not the same in every case. The distance between the camera and the road is approximated which is not accurate due to practical constraints and hence approximated.

The detected cars and their speeds are best demonstrated in the figure below:



B. Implementation issues

There were a few implementation issues when it came to developing the project. Firstly, training the neural network with the data-sets available took quite some time which interfered in achieving the deadlines.

When it came to implementing speed detection, we had issues with the system not having enough RAM and GPU processing power in order to process the video sufficiently. Unfortunately, we were not able to arrange for a better system so we had to cut short our project requirements.

In order to get correct speed detection readings, the distance parameter is manually modified for every video to replicate totally different approximate distances between vehicles and also the camera, which is obviously susceptible to inaccuracies. Also it is not able to detect cars in every single frame as the classifier needs to be more accurate.

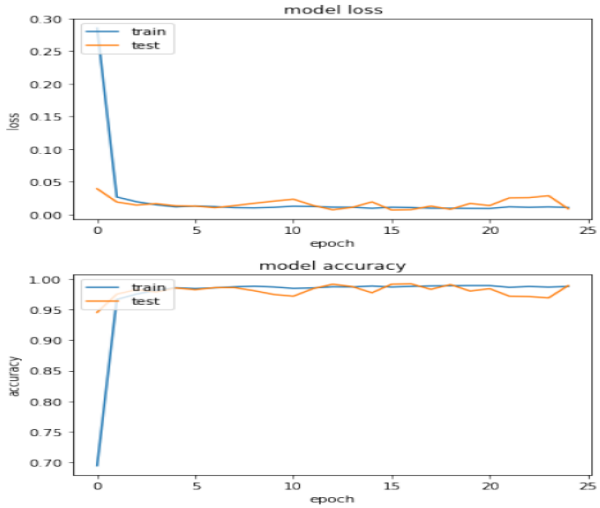
VI. DESIGN JUSTIFICATIONS, TESTING & RESULTS

Previous research has shown that Convolutional Neural Networks perform extremely well in Image classification problems because of their ability to exploit feature locality. They do it at different granularities, therefore being able to hierarchically model higher level features. CNN are not rotation-invariant, but they usually converge to filters which are rotated versions of the same filters, hence do support rotated inputs. Also it can be made into translation invariant with pooling units.

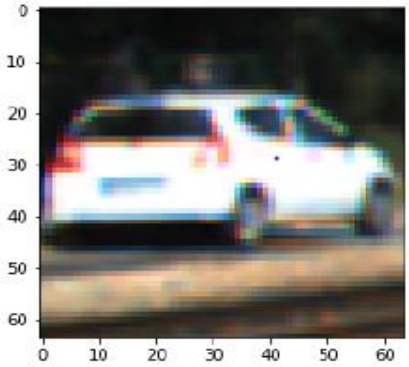
The Neural Network model was finalized because of its overall accuracy for the task, after training and testing various other models . The current model achieves Associate in Nursing accuracy of roughly ninety nine once twenty epochs of coaching.

ISSN: 0971-1260
Vol-22-Issue-3-July-2019

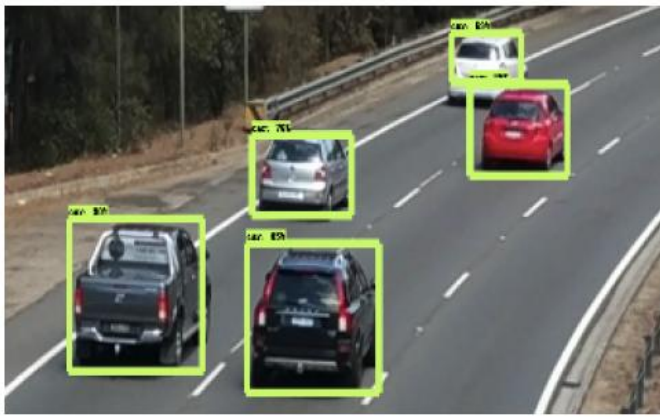
The loss and accuracy over time can be seen from the figure below:



A prediction made by the model on a random test image can be seen:



Next, a prediction on a full 1920 x 1080 real-world image/frame are often seen:



In order to test our prototype, samples of videos were collected across different locations around Sydney to capture vehicles with varying speeds and backgrounds.
The testing section allowed us to repair few bugs in our implementation that wants enhancements in potency of the Classifier to observe vehicles in each detected frame and work on removing false positives and subtracting stationary objects etc. One such case wherever false positives occur is incontestable in below figure.

The unskillfulness of the classifier not detection vehicles in each frame are often best incontestable in below figure within which the automotive isn't detected.

We are drawing a comparison between predicted vs obtained values, we are successful in most of the cases except some which are mentioned in the issues.
Implementation problems that area unit restricted by several real time constraints like FPS of camera, height of vehicles, focal length etc

VII. CONCLUSION & FUTURE EXTENSIONS

In conclusion, although there were few issues with false positives and the inability to detect cars, the prototype generally performed well during the lab demonstration.

ISSN: 0971-1260
Vol-22-Issue-3-July-2019

This project being one among the foremost topics in real time vehicle identification systems and can be extended to observe collisions. Another extension can be detecting over speeded cars and penalizing them for over-speeding.

VIII. INDIVIDUAL CONTRIBUTIONS

Out project contains 3 core modules, the tasks are split into three and assigned to the team as follows:

- Vehicle Identification – Ankit Anand
- Speed Detection– Piyush Kumar
- Number Plate Detection – Aman Kumar

REFERENCES

[1] Ask Your Neurons: A Neural-Based Approach to Answering Questions About Images Mateusz Malinowski, Marcus Rohrbach, Mario Fritzfor Object Detection and Identification.

[2] Aligning Books and Movies: Towards Story-Like Visual Explanations by Watching Movies and Reading Books
Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, Sanja Fidler

[3] Learning to See by Moving
Pulkit Agrawal, Joao Carreira, Jitendra Malik

[5] Poggio, T., Edelman, S.: A network that learns to recognize 3-D objects.
Nature 343, 163266 (1990)

[6] Y.Wang, E. K.Teoh and D. Shen, Lane Detection and Tracking Using
B-snake, Image and Vision Computing, vol. 22, pp. 269-280, 2004.

[7]ZF Net (2013)

[8] GoogLeNet (2015).

[9] A. Broggi and S. Berte, Vision-based Road Detection in Automotive