Data Extraction from Traffic Videos Using Machine Learning Approach

Anshul Mittal, Mridul Gupta and Indrajit Ghosh

Abstract Traffic safety has become one of the major concerns in most of the countries with extensive road networks. With the ever-increasing traffic and its various types, it has become increasingly difficult to check if the road network can sustain the surge. To evaluate the efficiency and safety of a network, several factors such as speed, vehicular composition, traffic volume are required. Data collection for calculating each of these factors is time-consuming. Most of the current activities in the area of intelligent transportation systems involve the collection of data through various sources such as surveillance cameras, but the collection alone is not sufficient. It requires a lot of time to process this data and determine the safety level of the road network which becomes manifold for a country with a vast network like India. It is a necessity to expand the use of intelligent transportation for the processing of the data. To achieve this initially, it is required to have traffic flow data. Therefore, high-resolution video cameras were placed at vantage points approximately 100–150 m away from the center of intersection locations. Two such intersections were selected from the National Capital Region (NCR) of India. Traffic flow-related data was recorded from 10 am to 4 pm during good weather condition. The obtained videos were then processed to segregate different types of vehicles. The proposed algorithm deals with the vehicles which are up to 70% occluded. A CNN-LSTM (Krizhevsky et al. in Advances in neural information processing systems, pp 1097–1105, 2012 [6]) model is trained for the recognition of a vehicle. Following this, a minimal cover volume algorithm is developed using bi-grid mapping for classifying vehicles and evaluating various parameters such as base center, orientation, and minimizing error due to occlusion. The proposed algorithm is based on machine learning, and it can estimate the required parameters with minimal human assistance and accuracy of 95.6% on test video and 87.6% on cifar-100 for object detection.

A. Mittal (✉) · M. Gupta · I. Ghosh
Department of Civil Engineering, Indian Institute of Technology Roorkee, Roorkee, India
e-mail: anshulmittal71@gmail.com

M. Gupta
e-mail: mridulgupta9@gmail.com

I. Ghosh
e-mail: indrafce@iitr.ac.in

J. C. Bansal et al. (eds.), *Soft Computing for Problem Solving*,
Advances in Intelligent Systems and Computing 816,
https://doi.org/10.1007/978-981-13-1592-3_16

Keywords Machine learning · Intelligent transportation system · ITS
Computer vision · Safety performance indicators

1 Introduction

Road transport system is one the fastest growing sectors increasing from 200 billion tonne km in 1980 to 700 billion tonne km in 2012 in India. This hike in the vehicles, on the one hand, represents the development of a country, and on the other hand, it leads to an increase in the road accidents. Over 1.2 million people worldwide are killed in road accidents each year, and many more are injured. These estimates are expected to increase by about 65% over the next 20 years unless we commit ourselves to its prevention. Safety performance indicators which are causally related to the number of crashes or to the injury consequences of a crash are increasing used. However, confirming the safety of a road network on the basis of crashes and injuries only is insufficient, and more information about the safety performance indicators (SPIs) such as speed, vehicular composition, maximum flow is needed. With rapidly expanding road networks and increasing vehicles, it is necessary to monitor these roads for safety standards at a similar pace. Thus, it is not preferable to waste time on manually extracting the SPIs from the data, which could very well be used in designing better road network. What we need is a software that can analyze the video of the traffic flow and supply us with all the required SPIs.

In this paper, we have presented an algorithm that can detect and accurately categorize the vehicles in three different weight classes, namely light, medium, and heavy to track the movement of the vehicles with reasonable accuracy. The categorization of vehicles' accuracy has been calculated on cifar-10 dataset and also on our own video (due to the absence of standard dataset). The remainder of the paper is structured as follows. In Sect. 3, we have explained the proposed algorithm in detail and the corresponding results are then presented in Sect. 4. Finally, we conclude the paper in Sect. 5.

2 Related Work

Recently, several papers have proposed methods of using deep networks for classification of different objects. In the VGG method [1], deep networks (CNN) are trained with increasing depth using an architecture with small convolutional filter. With increasing depth, the accuracy of classification also improves and achieves maxima at 16–19 weight layers. The Maxout method [2] is the same as convolutional neural networks (CNN) or multi-layer perceptron, i.e., a feed-forward architecture, but explores the effect on accuracy by using a new activation function, the maxout unit. In the OverFeat method [3], a single object is assumed and a fully connected layer is trained to predict a box around it. In the multi-box algorithm [4, 5], regions are predicted on the basis of a fully connected network that predicts multiple boxes which are used for regional convolutional neural network object detection. A comparison of our algorithm with VGG and Maxout methods has been provided in Table 5.

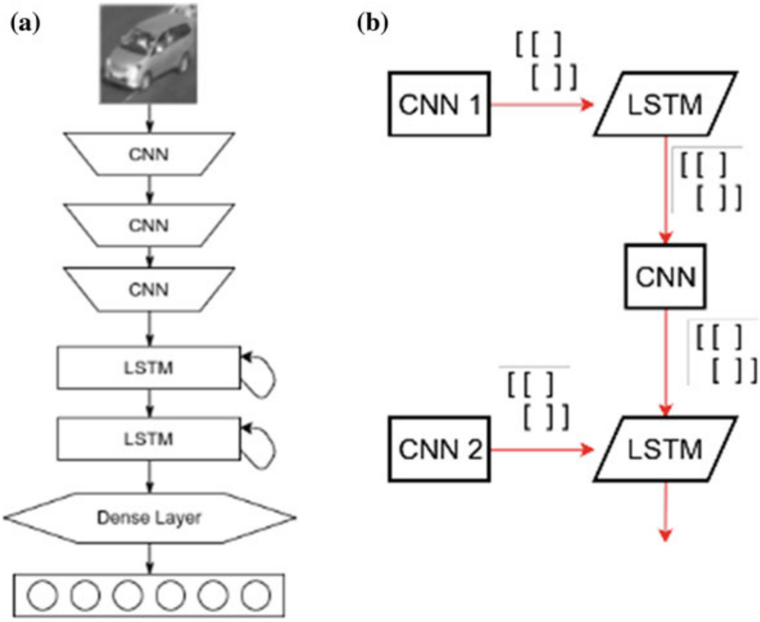

Fig. 1 **a** CNN-LSTM architecture for detecting class of vehicle and road surface. **b** Modified LSTM used in the network

3 Methodology

A. Vehicle and road detection

In this paper, we have used a deep CNN-LSTM architecture for recognizing vehicles and their classes, namely light, medium, and heavy vehicles as described by Krizhevsky et al. [6] and Sainath et al. [7]. We have used three CNN layers and two LSTM layers with a single dense layer in the end followed by softmax classification at the learning rate of 0.0001 with cross-validation. Memory cell comprises of CNN unit and a storage unit which convolves memory each time a new input arrives at input gate, and as a result, we get a convolving memory LSTM (see Fig. 1b) (CNN i for $i = 1$ to N where N is equal to number of features). Another contribution of this paper is to develop a mathematical model for detection of vehicles which are as much as 70% occluded. Results are compiled in the results section.

We segment the vehicle from its background using algorithm 1. Output sequence at each step of algorithm is shown in Fig. 2. Coordinates of the bounding box so obtained (Fig. 2d) are used for calculating minimum area rectangle to cover the segmented vehicle.

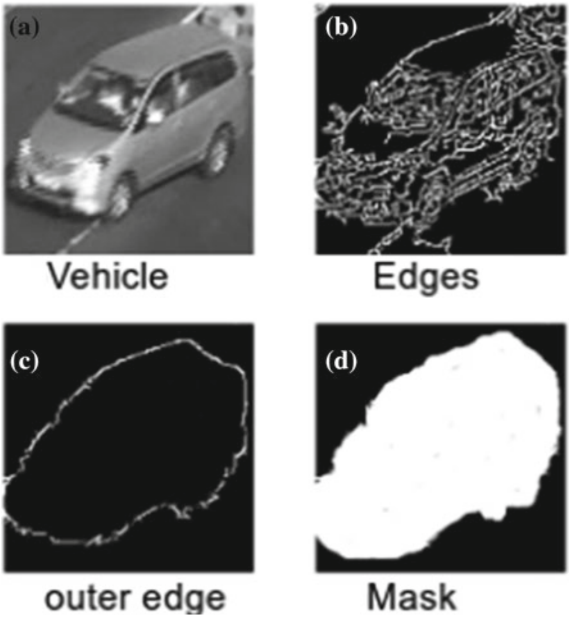

Fig. 2 Showing output of algorithm 1 as each step

Algorithm 1: Vehicle Extraction
//F(x,y) ← RGB Image
Edges ← edgeDetection(F(x,y))//Canny edge
outline ← outerMostCountour(Edges)
mask ← fillPolygon(outline)
return boundingBox(mask)

F(x, y) is the section of RGB image of road scene where detection of vehicle is positive Fig. 2a. The function edgeDetection accepts image as an input and returns edges as a vector and their hierarchy in the image Fig. 2b. The function outMostContour accepts the edges vector and returns the contour of which every other contour is a part of Fig. 2c. Function fillPolygon accepts the outermost contour and creates the binary image by filling inside the contour Fig. 2d. In the end, the algorithm returns the coordinates of the bounding rectangle of the mask Fig. 2d.

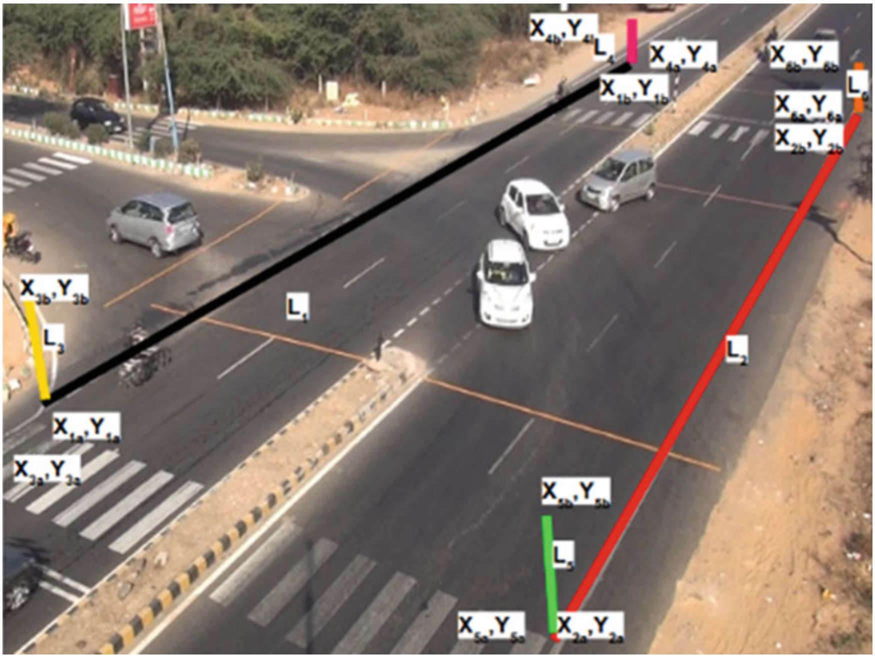

Fig. 3 Showing marking of line l_1, l_2, l_3, l_4, $l_{5,}$ and l_6

B. Scene parameters and cube fitting

Perspective images of road scene captured from the camera consist of information regarding depth (Y) as well its placement (X) with respect to the surface of road. For calculating parameters of perspective projection of the system, we follow the following procedure:

- We mark two line segments on the road scene l_1 and l_2 to meet at a point. For $l_{1,}$ we have two coordinates (x_{1a}, y_{1a}) and (x_{1b}, y_{1b}) similarly, for line $l_{2,}$ we have two coordinates (x_{2a}, y_{2a}) and (x_{2b}, y_{2b}) (Fig. 3).
- Next, we mark four vertical line segments for calculating boundary condition denoting projection of vertical objects l_3, l_4, $l_{5,}$ and l_6 whose coordinates are $[(x_{3a}, y_{3a}), (x_{3b}, y_{3b})]$, $[(x_{4a}, y_{4a}), (x_{4b}, y_{4b})]$, $[(x_{5a}, y_{5a}), (x_{5b}, y_{5b})]$, and $[(x_{6a}, y_{6a}), (x_{6b}, y_{6b})]$ for 2 m height (Fig. 3b). **Note**: for this algorithm to work always choose $(x_{3a}, y_{3a}) = (x_{1a}, y_{1a})$, $(x_{4a}, y_{4a}) = (x_{1b}, y_{1b})$, $(x_{5a}, y_{5a}) = (x_{2a}, y_{2a})$, and $(x_{6a}, y_{6a}) = (x_{2b}, y_{2b})$ (Fig. 3).
- Using minimal volume analysis as explained in the following section (Fig. 4), we calculate the orientation for the detected vehicle.

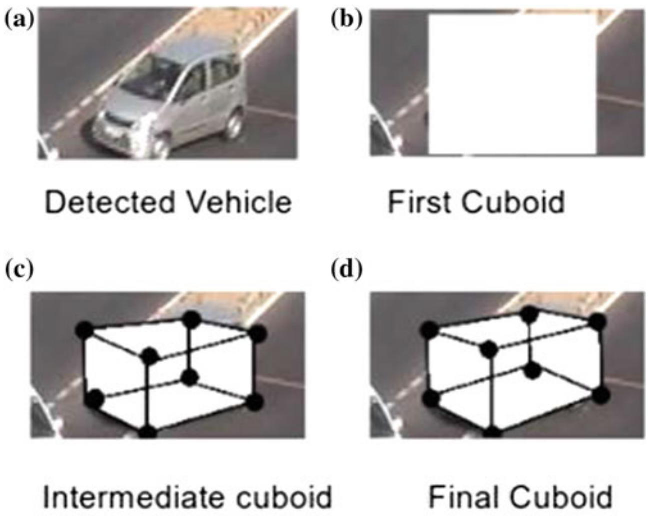

Fig. 4 Showing the time snaps for calculating intermediate cuboid

Table 1 Boundary condition to evaluate A and B

$[X', Y']$	$[X, Y]$
[0, 0]	$[x_{1a}, y_{1a}]$
[1, 0]	$[x_{1b}, y_{1b}]$
[0, 1]	$[x_{2a}, y_{2a}]$
[1]	$[x_{3a}, y_{3a}]$

C. Minimal volume cover

To evaluate the minimal cover, we have divided vehicles into three major classes, i.e., light, medium, and heavy. For each class, we have standardized the height of the vehicle to evaluate the minimal cover (Table 1). Using Eq. 1, we transform coordinates from perspective grid system to rectangular grid system, and using Eq. 2, we transform base of cuboid from rectangular grid system to perspective grid system, and using Eq. 3, we project height of cuboid from rectangular grid system to perspective grid system. For a given system, x, y represents coordinates in perspective distance and x', y' represents coordinates in transformed rectangular region.

$$X_0' = A * X_0 \tag{1}$$

$X_0{}' = [w'{*}x', w'{*}y', w]$ $w' = w$ for $w! = 0$ else infinity, $A' =$ is a 3×3 matrix, $X_0 = [x, y, 1]$. Using value from Table 2, we evaluate A.

$$X_1 = B * X_1' \tag{2}$$

Table 2 Boundary condition to evaluate C

$[X', Y', Z']$	$[X, Y]$
[0,0,0]	$[x_{3a}, y_{3a}]$
[0,0,2]	$[x_{3b}, y_{3b}]$
[1,0,0]	$[x_{4a}, y_{4a}]$
[1,0,2]	$[x_{4b}, y_{4b}]$
[0,1,0]	$[x_{5a}, y_{5a}]$
[0,1,2]	$[x_{5b}, y_{5b}]$
[1,1,0]	$[x_{6a}, y_{6a}]$
[1,1,2]	$[x_{6b}, y_{6b}]$

Table 3 Showing vehicle parameters for different classes

Class	Width	Length	Height
Light	0.64	1.87	1.5
Medium	1.6	4	2
Heavy	2.43	10	3.5

$X_1{}' = [x', y', 1]$, $X_1 = [w'{*}x, w'{*}y, w]$ $w' = w$ for $w! = 0$ else infinity, $B =$ is a 3×3 matrix. Using value from Table 2, we evaluate B.

$$X_2 = C * X_2' \tag{3}$$

$X_2{}' = [x', y', z', 1]$, $X_2 = [w'{*}x, w'{*}y, w]$ $w' = w$ for $w! = 0$ else infinity, $C =$ is a 4×3 matrix using value from Table 2, we evaluate C.

Parameters A, B, C are calculated by solving system of linear equations.

Using Algorithm 2, we evaluate cuboid having minimal area (minCbids) covering the detected vehicle(s). Persp2Rect transforms bounding box of max evaluated at the end to algorithm 1 (Box) to rectangular grid system using Eq. 1 and stores in RectBox. MinX and MinY returns the minimum value of x', y' of the RectBox, respectively. Similarly, MaxX and MaxY returns the maximum value of x', y' of the RectBox, respectively. For a given class of vehicle, we use the value of width (w), length (l), and height (h) (Table 3, [8]) which is used by the function createBox to calculate coordinates of the cuboid rotated at the angle α along z' axis in clockwise direction. rect2Persp uses Eqs. 2 and 3 to project cube on the perspective grid. calcProjArea uses the projected area of the cuboid on the perspective grid and evaluates the vehicle region covered.

Algorithm 2: Minimal Cover
\\Box: Stores coordinates of bounding box of mask
Procedure MinimalCover(k,box) \\k stores the number of cubes required to fill the box
minCbids \\Stores k cuboids initialized with maximum volume
RectBox ← Presp2Rect(Box);
x`min ← minX (RectBox)
x`max ← maxX (RectBox)
y`min ← minY (RectBox)
y`max ← maxY (RectBox)
for **x`** ∈ (x`min , x`max):
for **y`** ∈ (y`min , y`max):
for **α** ∈ (0 , 90):
cbids ← createBox(x`,y`, α, w, l, h,k); \\returns the tensor of k cuboids
for each (cbid,minCbid) in (cbids, minCbids):
projCube ← rect2Presp(cbid);
Area ← calcProjArea(projCube);
if Area< minCbid.area:
minCbid ← cbid;
return minCbids;

After evaluating minimum cover cuboid, we can store its parameters, i.e., x', y', α, class of the vehicle for further data analysis. This procedure is repeated for every vehicle detected in the frame, and using mean shift algorithm, the vehicle's path is traced in a given video sequence.

D. Occlusion correction

Algorithm 3 deals with the occlusion present in the video of road. Procedure bounding cubes take unique classes detected in a box as input as well as box (section of image which is being analyzed). This procedure returns the tensor of cuboids which is used to segment box and each segment is again passed to the neural network for vehicle class prediction.

Table 4 Accuracy assessment for different models

S. no.	Model	Cifar-10 (%)	Scene specific (%)
1	1C-0L	75.1	72.3
2	1C-1L	85.7	76.8
3	2C-1L	90.4	82.5
4	2C-2L	97.3	85.6
5	3C-1L	96.1	84.9
6	3C-2L	99.5	95.6
7	3C-3L	99.4	95.2

Note xC-yL implies that x layers of CNN followed by y layers of LSTM network

```
Algorithm 3: Maximum number of bounding cubes
Procedure boundingCubes(classes,box):
    Flag = True
Volume = 0
k = classes
while(Flag):
    tempVolume = 0
    cuboids = MinimalCover(k,box)
    for each cuboid in cuboids:
        tempVolume += cuboid.volume
    if tempVolume <= 0.8*Volume:
        Volume = tempVolume
        k+=1
    else :
        return cuboids
```

4 Results

E. Accuracy of CNN-LSTM architecture for detecting class of vehicle and road surface.

We have experimented with multiple CNN-LSTM model, out of which we have chosen the one which is giving maximum accuracy. In Cifar-10 and scene-specific data, we have created ground truth manually for classification into light, medium, and heavy. For accuracy, we have given 1 for each correct and 0 for incorrect prediction. Results are summarized in Table 4. Results are evaluated by taking average of accuracy over 1000 images.

Table 5 Table for accuracy comparison for recognition on different datasets with recent papers

Dataset	VGG [1] (%)	Maxout [2] (%)	Our method (%)
Cifar-10	98.68	92.3	99.5
Image net	91.57	94.6	98.3

Table 6 Accuracy for different classes of vehicles

	2 Wheelers (Light) (%)	4 Wheelers (Medium) (%)	Trucks (Heavy) (%)
Our method	78.2	94.6	93.2

F. Comparative Studies

We have summarized results of comparison of our method with few current methods in Table 5. Our method outperforms all the mentioned neural networks.

G. Vehicle category recognition accuracy

In Table 6, we have summarized the accuracy of our algorithm on different classes of vehicles.

5 Conclusion

In this paper, we have presented a machine learning approach for faster vehicle data extraction with the overall accuracy of 95.6%. After detection using neural networks, minimal cover cuboid method was used to evaluate the orientation of the on-road vehicle. Since we have used standard widths and lengths of the vehicles from [8] and have estimated the height from the average height of the vehicles, we can estimate orientation of a vehicle even if it is occluded. Data collection and analysis which takes the major portion of the time of designing of a road network can now be done in considerably lesser time using this algorithm. It took only 2 h to analyze a 24 h video of an intersection and preparing a table for percentage occlusion, orientation, and vehicle classes. Therefore, the algorithm presented in this paper is faster, robust, and free from human intervention, thus saving large amount of money.

References

1. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: ICLR (2015)
2. Goodfellow, I.J., Warde-Farley, D., Mirza, M., Courville, A., Bengio, Y.: Maxout networks. arXiv:1302.4389 (2013)
3. Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., LeCun, Y.: Overfeat: integrated recognition, localization and detection using convolutional networks. In: ICLR (2014)
4. Erhan, D., Szegedy, C., Toshev, A., Anguelov, D.: Scalable object detection using deep neural networks. In: CVPR (2014)
5. Szegedy, C., Reed, S., Erhan, D., Anguelov, D.: Scalable, high-quality object detection. arXiv:1412.1441v2 (2015)
6. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems, pp. 1097–1105 (2012)
7. Sainath, T.N., Vinyals, O., Senior, A., Sak, H.: Convolutional, long short-term memory, fully connected deep neural networks. In: 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 4580–4584. IEEE (2015)
8. Nokandeh, M.M., Ghosh, I., Chandra, S.: Determination of passenger-car units on two-lane intercity highways under heterogeneous traffic conditions. J. Transp. Eng. **142**(2), 04015040 (2015)
9. World Health Organization (WHO): World Report on Road Traffic Injury Prevention, WHO, Geneva, http://www.who.int/violence_injury_prevention/publications/road_traffic/world_report/en/ (2004)
10. NTDPC ~ Vol-02 Part1 ~ Ch02.indd26 page30, http://planningcommission.nic.in/sectors/NTDPC/volume2_p1/trends_v2_p1.pdf