Real-Time Road Traffic Image Streaming for Travel Planning and Commuter Safety

Abhishikta Panja[1], Harshit Agrawal[2]

Student, Department of Electronics and Communication Engineering, SRM Institute of Science and Technology Chennai, India[1]

Student, Department of Electronics and Communication Engineering, SRM Institute of Science and Technology Chennai, India[2]

abhishiktapanja@gmail.com

harshitagrawal025@gmail.com

***Abstract*—The advent of the Intelligent Traffic Management and Monitoring System has made the commute system very efficient enabling people to cope up with this remarkable increase in traffic on a daily basis. The huge amount of optimized data offered to the public over user-friendly applications play a significant role in enhancing the ease of transportation. This paper proposes robust algorithms based on Image Processing and Deep Learning algorithm YOLO-CA, that will not only provide the most convenient route of the journey but also the real-time images along the entire route, directly to the user, for better decision-making purposes and clarity on a mobile application. High-resolution cameras are setup for capturing images every second for complete monitoring of roadways. These continuous images are analysed for traffic density, speed, and accident detection. Moreover, it incorporates an Automatic alert system to nearby hospitals in case of accident detection so as to ensure treatment in time for the victims. We use Automatic License Plate Detection for booking drivers with Challans for rash driving and exceeding the speed limit. The implementation is done with camera modules, Deep Learning algorithms, GPUs, Azure Cloud, and REST APIs. The paper emphasizes the benefits of this improved transport monitoring system in terms of complexity, efficiency, throughput, safety, and security.**

***Index Terms*—Assistive Technology, Image Processing, Mobile Application, REST API, YOLO-CA**

1. INTRODUCTION

There are millions of people commuting at every hour in a day. Along with an increase in population and urbanisation, people travelling in their personal vehicles have escalated significantly, in turn, contributing to the increase in traffic. However, the smart traffic management system has enabled people to choose the best-suited route for themselves before even starting their commute, using the analysed data and features provided to them over mobile applications. In this paper, we propose a mobile application which makes commute smoother and safer. We contribute to make better, the existing Intelligent traffic Monitoring system by increasing the accuracy of traffic density calculation, suggestion of optimum routes, faster

response times, flexible installation and low maintenance costs, enhancing clarity at user's end, post-accident measures and penalizing traffic rule defaulters. Many existing traffic management systems use inductive loops, expensive sensor packages, GPS, etc. These approaches are not only cost-intensive but also incorporate lengthy installation processes. GPS system navigation applications make use of GPS modules in mobile phones or vehicles itself, wherein, an area where the number of pedestrians is high would erroneously result in high vehicular traffic density zone. Such single parameter based methods cannot match the required accuracy and real-time requirements. Thus, we use an Image Processing algorithm for monitoring traffic. Poles equipped with cameras and motion sensors are laid at definite intervals on the roads and highways. Motion sensors help the cameras to operate only when motion is detected, so as to, reduce the data clutter on the GPU. These images are processed to determine the traffic density and suggest the best route using Dijkstra's Algorithm. Trained Image classifiers are deployed to detect accidents, and over-speeding vehicles. This application has interface ends for commuters, Hospitals and Police Stations. On selection of a source and destination, the commuter will get the best route suggestions supported by real-time updates regarding road blockages and accidents occurring in their route. Re-routing suggestions are provided automatically wherever necessary. Given a situation in a highly congested city, an existing navigation application might suggest routes with similar traffic density and ETAs. In such a case, decision making becomes difficult due to the lack of real-time data. Thus, we intend to provide the general public with continuous live images of the the roads to enhance traffic planning efficiency. On the user's interface, each pole location is pinpointed by a marker that can be tapped for easy access to real-time images of the chosen route. In case of an accident detection, images and location of the mishap will be forwarded to the nearest hospital and police station for immediate action at ground zero. Thus, timely treatment of the victims is ensured even in sparsely populated areas where easy help in case of accidents is not possible. With the help of Automatic License Plate Detection, vehicles exceeding the speed limit will be charged on the basis of a Central Challan System. The Challan notification will be sent to the application immediately, along with sharing of necessary data with the respective police stations. The architecture of the proposed idea involves high-resolution cameras, GPUs, Deep learning algorithms, Azure cloud, and REST APIs. Azure Cloud inherits Shared Access Signatures (SAS) which encrypts a large amount of data processed and ensures its all-round security and systematic delivery.

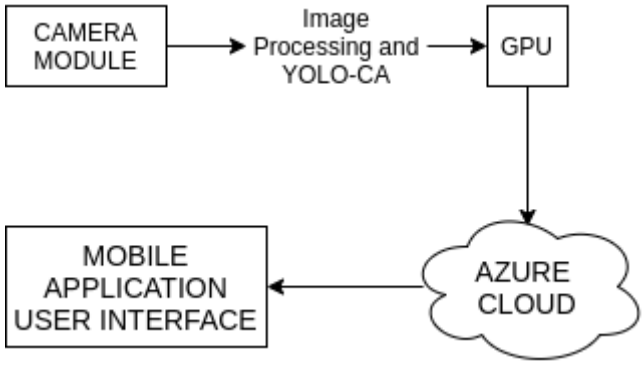

Fig. 1. Application Work-Flow

2. RELATED WORKS

The increasing traffic on the road calls for development in the field of traffic monitoring, management, reduction and control. Many researchers have given way to many technological advancements in detection and prediction of traffic density, accident detection methods etc. Prediction, Optimization and Calculation of commute time have a variety of algorithms with varying efficiencies. Some researchers use motion parameters of the vehicle like acceleration and velocity, behaviour of the engine or steering to predict abnormalities, for which sensor networks are established [11]. This requires external hardware installation to a number of vehicles or infrastructure resulting in high maintenance costs. Although these methods may have high efficiency, but, relying entirely on vehicular monitoring and communication equipment may be damaging in case of some extreme circumstances, such as heavy canopy, underground tunnel, or severe accidents. Google Maps [2] uses the GPS system of our mobile phones to determine traffic density in a particular area and thus suggest the best route. In crowded areas with high pedestrian traffic the vehicular density calculation might give wrong figures. With advancements in video processing, Deep learning and neural networks researchers have started to adopt these methods in Intelligent Traffic System Management. Reference [3] proposed a Dynamic-Spatial Attention Recurrent Neural Network (RNN) for predicting accidents in captured videos around 2 seconds beforehand with 80% recall and 56.14% precision. This precision percentage may lead to false alerts and erroneous predictions.

3. SYSTEM MODEL

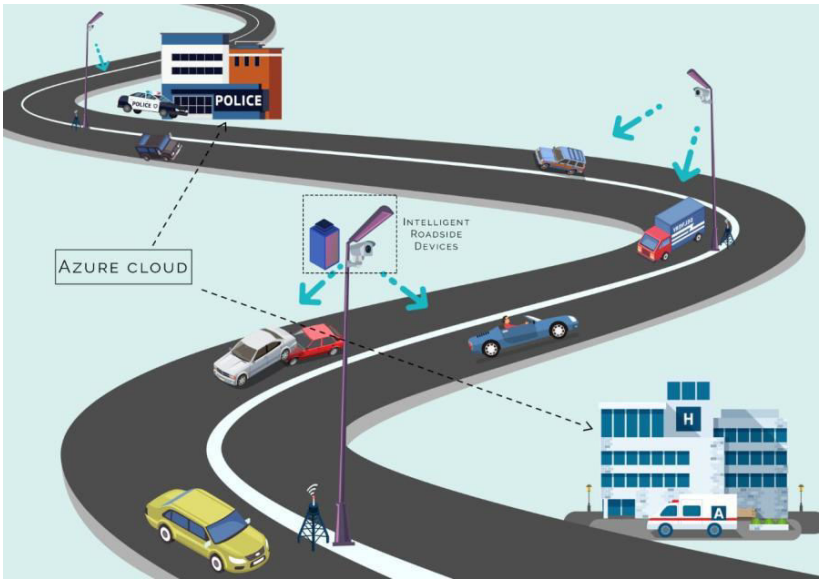

Fig. 2. Implementation Illustration

A. TRAFFIC DENSITY MONITORING SYSTEM

This methodology implements image processing to determine the vehicle density on the road in a particular area and time. A camera module placed at an optimum angle, controlled by a motion sensor, captures real-time images to extract traffic density. Most of the existing

algorithms implement counting of the number of vehicles in the captured video. When two vehicles are in very close proximity they may be counted as one, leading to highly inaccurate results. Thus, we have adopted an algorithm that extracts the number of pixels, occupied by the vehicles, from a frame to inspect the traffic density. A video approximately gives output at the rate of 240 frames per second thus leading to expansive processing of data. We employ the cameras to capture images at tiny intervals to reduce the computational complexity and excessive load on the processor. We have adopted a combination of two methods namely, Gradient Magnitude method and the Direct Subtraction method to obtain the nearest accurate traffic congestion value [4]. In the first method, RGB images are converted to grayscale which is followed by Sobel edge detection operation [5], Weiner filtering [6] and Morphological closing [7] resulting in image G_{binary}. Further, in the second method the grayscale background image (picture of empty roads) is subtracted from the grayscale foreground image (subsequent images of the road) to obtain D_{object}. D_{object} undergoes filling, tuning and factoring operations [8] [9] to determine the total number of pixels giving rise to image D_{binary}. This combination prevents discrepancies due to color of the vehicle or non-formation of closed contour due to detected edges. Fig.3 and Fig.4 shows total area covered by vehicles in pixels.

$$I = 0.33 \times R + 0.33 \times G + 0.33 \times B \quad (1)$$

For Background Subtraction,

$$D_{object} = FG_{img} - BG_{img} \quad (2)$$

where,
BG_{img} - Background Image
FG_{img} - Foreground Image

Final Image after applying both methods,

$$I_{total} = G_{binary} + D_{binary} \quad (3)$$

$$I_{total} = \begin{cases} 1, & if\ pixelvalues \geq 1 \\ 0, & otherwise \end{cases} \quad (4)$$

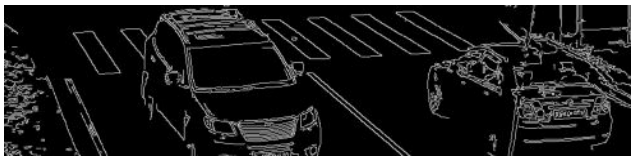

Fig. 3. Grayscale Image

Fig. 4. Combination of Gradient Magnitude Method and Direct Subtraction

B. BEST ROUTE ALGORITHM

Dijkstra's Algorithm is used to determine the shortest route from source to destination [10]. It is a stepwise iterative method in which weighted nodes with the lowest traffic density are identified and the fastest route is mapped by connecting the nodes. The starting point is known as the initial node. To improve the system this algorithm assigns some initial values by default. From the initial node, the distances to the next possible intersections are calculated and labelled. The intersections are compared and the one with lower weight (least traffic density) and lower distance is relabelled. This comparison is continued until the optimum node is marked. The marked nodes are termed as the visited nodes. The visited nodes calculated in all consequent iterations are connected up to the destination node to acquire the desired path.

C. IMAGE PROCESSING ALGORITHMS

1) Deep Learning Algorithm for Object Detection:

In the majority of cases, victims of a road accident suffer more due to untimely treatment and lack of communication between the hospital or the police. Also, the common persisting problem of rash driving should be brought into check. If the vehicle speed exceeds the limit, the data would be reported to a police station and a Challan will be issued automatically. A notification will be activated for a penalty that will be displayed on the mobile application of the user. Thus, we intend to enforce an algorithm that detects abnormalities from the real-time images captured by the camera installed on the roads and sends information to the required authorities. We have employed a Deep Learning Convolution Neural Network (CNN) model which is trained to classify these image frames to determine an accident. An input image is an array of pixels depending upon its resolution and will consider the parameters h x w x d. For example, an image of array 6 x 6 x 3 (where 3 stands for RGB). Each image is passed through a series of convolutional layers or filters (kernels) to train and test Deep Learning CNN. This is followed by Pooling of fully connected layers and application of Softmax function to classify an object with a probabilistic value between 0 and 1. Convolution is the first layer to extract features from an input image. CNN reduces the images to a form that is easier to process, without losing critical features for image prediction. This mathematical operation takes two inputs such as image matrix and a filter or kernel as shown in Fig.5 An image matrix with volume of dimension (h x w x d) and a filter with dimensions (f_h x f_w x d) Outputs a volume of dimension (h - f_h + 1) x (w - f_w + 1)

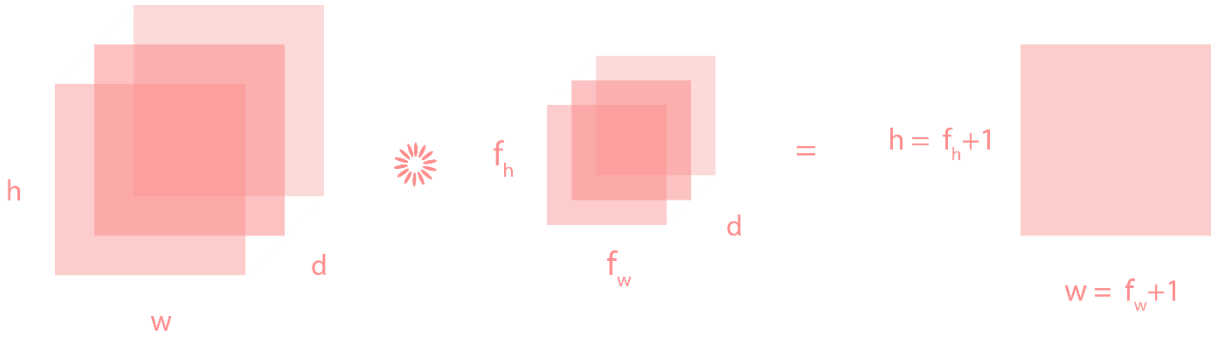

Fig. 5. Multiplication of Image Matrix with Kernel or Filter Matrix

The convolution of image matrix multiplied with filter matrix gives an output which is termed as the Feature Map. For desired results suitable parameters are chosen along with filters, strides and padding. Stride shifts the input sampling and downsamples the matrix. Padding is done when filter imperfectly fits the image and zeroes are added to the image to rectify it. ReLu (Rectified Linear Unit) activation is applied to the image to introduce nonlinearity to the convolution. Pooling layers section reduces the number of parameters when the images are too large. Spatial pooling reduces the dimensionality of each map which is called subsampling or downsampling and retains important information.

A YOLO-CA model is used for high reliability and fast response at various traffic conditions. It can differentiate images by identifying and assigning importance to various aspects in the frame. Along with the application of relevant filters, it captures the spatial and temporal dependencies in a frame. As compared to other classifications the pre-processing in CNN is much less and thus bears fast and accurate results in image classification. CNN is capable of giving results with an accuracy above 95% with smaller datasets as compared to other image classifiers.

2) Training of Model:

For the training of the YOLO algorithm, each image is divided into grids of dimension S x S. A grid cell enclosing the center of an object is responsible for detecting that object. An addition of Multi-Scale Feature Fusion (MSFF) with the YOLO algorithm dramatically improves the speed detection and accuracy of the system. This model is known as YOLO-CA [1]. Image upsampling and two different dimensional output tensors are obtained by utilizing 24 layers. YOLO-CA is composed of 228 neural network layers which constitutes of basic components, such as DBL and ResN. Convolution layer, Batch Normalization layer and Leaky ReLU layer constitute the DBL. ResN has the Zero Padding layer, DBL and N $\text{Resblock}_{\text{units}}$, which is designed to avoid neural network degradation caused by increased depth. Ups is known as the upsampling layer, which improves the performance of YOLO-CA in detecting small objects. Concat is used to concatenate the layer in Darknet-53 and upsampling layer.

3) Detection Principle

YOLO-CA contributes to extracting a feature map and bounding box prediction. The input image is divided into grids of 13 X 13 and 26 X 26. The first grid is responsible for detecting larger objects and the second grid is used for making up the inaccuracy in detecting smaller target objects. The two grids focus on extracting different features but follow a similar method of detection. The center cell predicts three different bounding boxes with six different parameters x, y, w, h, CS, p. (x,y) is the centre of bounding box; (w,h) is the width to height ratio of the bounding box to image; CS-confidence score; p-probability of a car accident. After batches of training, these parameters undergo losses

$$L(img) = L(xy) + L(wh) + L(cs) + L(p) \quad (5)$$

$$\text{Loss} = \frac{1}{b}\sum_{k-1}^{b} L(img)_{k} \quad (6)$$

where,

L(img)-Loss in entire image
L(xy)-Loss due to x and y parameters
L(wh)-Loss due to w and h parameters
L(p)-Loss due to probability

D. SPEED DETECTION ALGORITHM

The camera module monitors the vehicles as long as they are in their field of view over a short span of time. This algorithm determines the speed of the vehicle by using the generic formula of,

$$\text{Speed} = \frac{Distance}{Time} \quad (7)$$

The cameras have a wide-angle range and for every field of view, two markers are set. Consider one of them to be the starting point (say x1) and the other one to be the endpoint (say x2). Since, the same type of cameras are used for all roads, its range and field of view remains the same throughout. A fixed distance of (x1-x2) is taken as a reference standard. Since the camera continually captures images, the time taken for a vehicle to reach from x1 to x2 is easily calculated. Thus, the speed of a vehicle passing a camera pole is monitored and in case of detection of over-speeding vehicles, their license plate is automatically detected and a Challan is issued to the user.

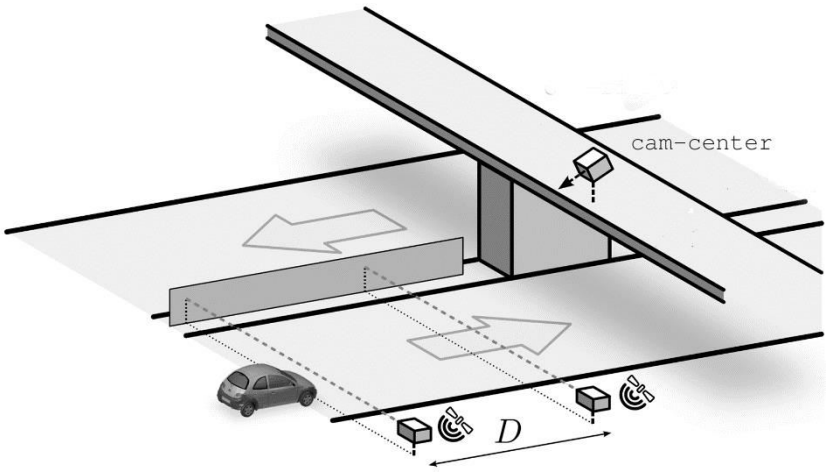

Fig. 6. Speed Detection Algorithm

E. AZURE CLOUD, REST API, AND MOBILE APPLICATION

We use Azure Cloud [12] to store the enormous amount of data collected from camera modules on a daily basis from the huge road network. The analysed and segregated data from the GPU is sent to the Azure cloud from where this data is distributed to respective interfaces. Microsoft secures Azure Cloud by highly advanced security tools like Threat Intelligence, Advanced Threat Analytics, Azure Information Protection, and Multi-Factor Authorization. In case of a breach of the red-flag threshold or vulnerabilities in security the cloud administrator is sent a ping immediately. It supports Advanced Message Queueing Protocol (AMQP) which enables one to build cross-platform, hybrid applications. Unlike most other cloud service providers Azure has high availability and scalability with guaranteed uptime of 99.95%. Its pay-as-you-go plan makes it extremely cost-efficient. REST is an acronym for REpresentational State Transfer. Three endpoints are created at the cloud end for Commuters, Hospitals and Police Station. The REST API fetches required data from these endpoints in the cloud in JSON format. JSON server supports to setup a REST API with CRUD.json server, available as a NPM package.

```
$json-server –watch DB.json
```

Watch function verifies that the server was started in watch mode i.e. file changes and updates are exposed to the API as per requirement. The following HTTP endpoints are created automatically by the server:

i. GET (GET Request 1: imports image from Cloud on pinpointing a particular pole)
ii. GET (GET Request 2: sends accident-site location and images to Hospital interface and Police Interface)
iii. POST (Authenticates the image and pushes it to the socket from the JSON Server)

iv. PUT (imports data to the user-interface)
v. PATCH (incorporates at the designated slot in the UI)
vi. DELETE (Deletes unnecessary images from the clutter).

This mobile application is accessible by all commuters, Hospitals and Police Stations. Fig. 7 shows the User Interface.

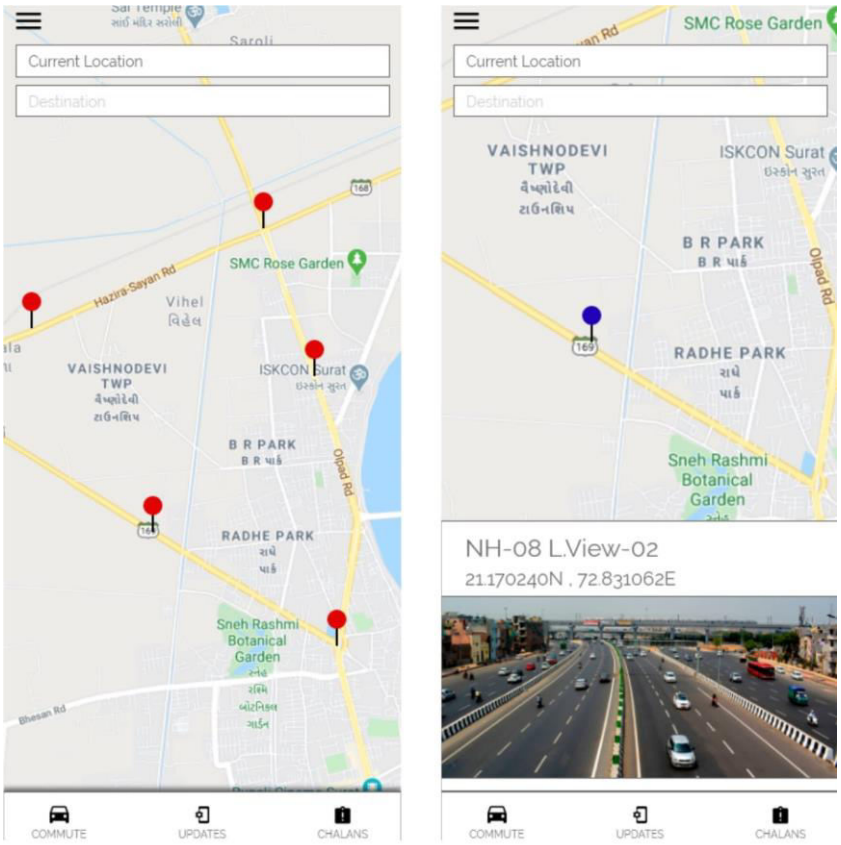

Fig. 7. User Interface

4. EXPERIMENT AND APPLICATION WORKFLOW

The primary hardware setup for our proposed model is the installation of poles equipped with camera modules and motion sensors at regular intervals covering the maximum length of the road. A preferred high-resolution camera module for prototyping is an 8MP Raspberry Pi NoIR Camera Module v2. These modules are set at an optimum height of 2.5m from the ground. Images captured at each second are processed through CUDA Toolkit which includes GPU-accelerated libraries and a compiler. A Convolution Neural Network model is trained to detect and classify real-time images of size 150 x 150 as Accident or Non-Accident or, Normal or Over-speeding.

Fig. 8. No Accident Detected

Fig. 9. Accident Detected

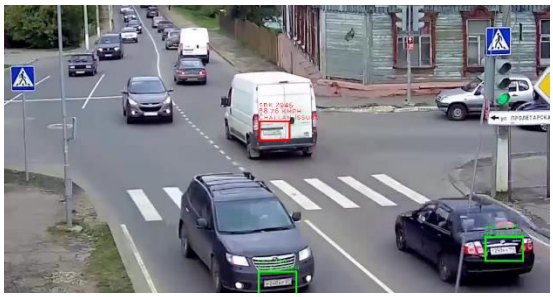

Fig. 10. Real-Time example of Speed Detection

Since we require high real-time response we have deployed a single-stage model YOLO-CA. Video clips from these roadside cameras are taken in mp4 format for the training of the YOLO model. 240 images are extracted frame by frame per second from the video for the learning model to reach high accuracy. We prepared a dataset for training in license plate detection which is followed by segmentation. To ensure proper segmentation we deploy certain preliminary processes as shown in Fig. 11.

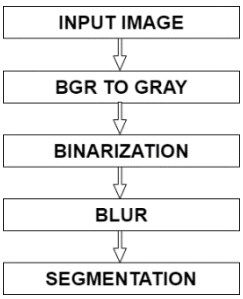

Fig. 11. Flowchart of Preliminary Processes

Once the model is properly trained, there is a shift from Video Processing to Image Processing to reduce computational complexity and dropout unnecessary load on the GPU. Images captured every second are analysed by the system model algorithms. The training of models is done using Keras 2.3.0 and implementation is done using Python 3.7. We used a batch size of 64, and trained the models for up to 30,000 iterations. The initial learning rate is set to 0.001, and updating with iteration parameter of 0.1/10,000 iterations. Moreover, we use a weight decay of 0.0005 to prevent model Overfitting.

		GROUND TRUTH	
		Positive	Negative
Prediction Result	Positive	TP	FP
	Negative	FN	TN

Table. 1. Ground Truth Prediction Result

As shown in the above table, the prediction results [14] can be divided into four parts:

(1) TP: Truth Positive

(2) FP: False Positive

(3) FN: False Negative

(4) TN: True Negative

The precision is defined and recall is defined as:

$$\text{Precision} = \frac{TP}{TP+FP} \tag{8}$$

$$\text{Recall} = \frac{TP}{TP+FN} \tag{9}$$

The precision of YOLO-CA is increasing gradually and converging over 90%. Recall eventually converges to more than 95% and the final convergence of loss is less than 0.2 (from equation 6). The segregated data is then dumped into the Azure Cloud and run on JSON servers. Further REST API fetches data from the endpoints and publishes it to the end users.

5. CONCLUSION

This paper proposes a mobile application for making commute easier and safer for the public and informs secure agencies immediately in case of accidents or breach of traffic rules. Poles equipped with cameras and motion sensors are setup at regular intervals wherein the camera captures images at the rate of 1 image/second. We determine the traffic density by calculating the area covered by vehicles in a frame, that is by calculating the number of pixels occupied by vehicles in an image. This is followed by application of the Djikstra's algorithm for suggesting the best route to the user. We provide real-time images of the chosen route to the user for better clarity and give them live-updates about the disruptions in the route. In case of accidents, the location of the mishap, severity of the accident along with on-spot images are sent to the nearest hospitals and the police. Furthermore, the speed of all the vehicles is continuously monitored. In any case of over-speeding the license plate of the vehicle is detected and automatic Challan is issued to the registered vehicle. We have implemented Image Processing and novel Deep Learning algorithm YOLO-CA giving an accuracy of upto 90% which is more efficient than many existing models for Traffic Monitoring and Management Systems. This mobile application handles enormous amount of backend data which is secured by Microsoft Azure's

best available 256-bit AES cipher for encryption and decryption Moreover, these techniques have a flexible low-cost installation and maintenance price along with precise real-time time data freely available to the public.

6. REFERENCES

[1] D. Tian, C. Zhang, X. Duan and X. Wang, ”An Automatic Car Accident Detection Method Based on Cooperative Vehicle Infrastructure Systems,” in IEEE Access, vol. 7, pp. 127453-127463, 2019.

[2] https://www.businessinsider.in/business/news/an-artist-wheeled-99- smartphones-around-in-a-wagon-to-create-fake-traffic-jams-on-googlemaps/articleshow/73913493.cms

[3] F.-H. Chan, Y.-T. Chen, Y. Xiang, M. Sun,”Anticipating accidents in dashcam videos” in Computer Vision-ACCV 2016, Springer, pp. 136- 153, 2017.

[4] M. M. Hasan, G. Saha, A. Hoque and Md. Badruddoja Majumder, ”Smart traffic control system with application of image processing techniques,” 2014 International Conference on Informatics, Electronics & Vision (ICIEV), Dhaka, 2014, pp. 1-4.

[5] I. Sobel, An Isotropic 3x3 Gradient Operator, Machine Vision for Three Dimensional Scenes, Freeman H., Academic Pres, NY, pp. 376-379, 1990

[6] Lim, and S. Jae, Two-Dimensional Signal and Image Processing, Englewood Cliffs, NJ, Prentice Hall, 1990, p. 548, equations 9.44-9.46.

[7] C. Gonzalez, E. Woods, and L. Eddins, Digital Image Processing Using Matlab, Gatesmark Publishing, 2009, p. 347.

[8] P. Soille, Morphological Image Analysis: Principles and Applications, Springer-Verlag, 1999, pp. 173-174.

[9] N. Otsu,”A Threshold Selection Method from Gray-Level Histograms,” IEEE Transactions on Systems, Man, and Cybernetics, Vol. 9, No. 1, 1979, pp. 62-66.

[10] L. Rosyidi, H. P. Pradityo, D. Gunawan and R. F. Sari, ”Timebase dynamic weight for Dijkstra Algorithm implementation in route planning software,” 2014 International Conference on Intelligent Green Building and Smart Grid (IGBSG), Taipei, 2014, pp. 1-4.

[11] J. Zaldivar, C. T. Calafate, J. C. Cano, P. Manzoni, ”Providing accident detection in vehicular networks through OBD-II devices and Androidbased smartphones”, Proc. IEEE 36th Conf. Local Comput. Netw. (LCN), pp. 813-819, Oct. 2011.

[12] https://docs.microsoft.com/en-us/azure/cloud-adoption-framework/

[13] N. Dogru and A. Subasi,”Traffic accident detection using random forest classifier,” 2018 15th Learning and Technology Conference (L&T), Jeddah, 2018, pp. 40-45.

[14] N. D. Marom, L. Rokach and A. Shmilovici, ”Using the confusion matrix for improving ensemble classifiers,” 2010 IEEE 26-th Convention of Electrical and Electronics Engineers in Israel, Eliat, 2010, pp. 000555- 000559.

[15] S. Ghosh, S. J. Sunny and R. Roney, ”Accident Detection Using Convolutional Neural Networks,” 2019 International Conference on Data Science and Communication (IconDSC), Bangalore, India, 2019, pp. 1- 6.

[16] Jakub Sochor, Roman Juranek, Jakub ´ Spa ˇ nhel, Luk ˇ a´s Mar ˇ s ˇ´ık, Adam Sirok ˇ y, Adam Herout, Pavel Zem ´ c ˇ´ık, Brno University of Technology”Comprehensive Dataset for Automatic Single Camera Visual Speed Measurement”, 9 May 2018