An amalgamation of YOLOv4 and XGBoost for next-gen smart traffic management system

Pritul M Dave[1], **Arjun B Chandarana**[1], **Parth N Goel**[1], and **Amit P Ganatra**[2]

[1]Computer Science & Engineering Department, Devang Patel Institute of Advance Technology and Research, Charotar University of Science and Technology (CHARUSAT), CHARUSAT campus, Changa 388421, Gujarat, India
[2]Computer Engineering Department, Devang Patel Institute of Advance Technology and Research, Charotar University of Science and Technology (CHARUSAT), CHARUSAT campus, Changa 388421, Gujarat, India

Corresponding author:
Pritul M Dave[1]

Email address: pritul.dave@gmail.com

ABSTRACT

The traffic congestion and the rise in the number of vehicles have become a grievous issue, and it is focused worldwide. One of the issues with traffic management is that the traffic light's timer is not dynamic. As a result, one has to remain longer even if there are no or some number vehicles on a roadway, causing unnecessary waiting time, fuel consumption, and leads to pollution. Prior work on smart traffic management systems repurposes the use of Internet of things, Time Series Forecasting, and Digital Image Processing. Computer Vision-based smart traffic management is an emerging area of research. Therefore a real-time traffic light optimization algorithm that uses Machine Learning and Deep Learning Techniques to predict the optimal time required by the vehicles to clear the lane is presented. This paper concentrates on a two-step approach. The first step is to obtain the count of the independent category of the class of vehicles. For this, the You Only Look Once version 4 (YOLOv4) object detection technique is employed. In the second step, an ensemble technique named eXtreme Gradient Boosting (XGBoost) for predicting the optimal time of the green light window is implemented. Furthermore, the different implemented versions of YOLO and different prediction algorithms are compared with the proposed approach. The experimental analysis signifies that YOLOv4 with the XGBoost algorithm produces the most precise outcomes with a balance of accuracy and inference time. The proposed approach elegantly reduces an average of 32.3% of waiting time with usual traffic on the road.

Keywords Object detection, YOLOv4, Machine Learning, Deep Learning, Computer Vision, Regression Analysis, eXtreme Gradient Boosting (XGBoost)

INTRODUCTION

Traffic management and pollution arising from the traffic are significant issues in India's metropolitan areas, resulting in unnecessary waiting time and congestion. The transportation is the third predominant cause of air pollution, according to the official data of Environmental Statistics 2019 (IndianMinistryOfStatistics, 2019). Although India's government began to encourage public transport and escalated the private vehicle taxes, the effect is minimal. In recent times different methods for the regulation and management of the traffic, including Internet of Things (Greengard, 2015), Time Series Forecasting (Jenkins, 1994) and Digital Image processing (Rafael, 1992) are proposed. Although these approaches achieve impressive results, they are still not cost-effective, fast, accurate, and far from satisfactory to work in real-time scenarios. To overcome these, a method using the Convolutional Neural Network (KrizhevskyA, 2012) is bestowed. The advancements and development of the effective algorithms in Computer Vision (Szeliski, 2010) enables the computers to perform the task without the use of sensors or human intervention and convey

real-time information. As a result, an Artificial Intelligence-based (Long et al., 2020) Computer Vision solution is proposed to detect the traffic density on the lanes and anticipate the optimum time required to clear the traffic. Moreover, the computer vision-based approaches serves the visual interpretation of the scenarios, resulting in model transparency. The analyzed results validates that the proposed approach is faster at inference, cost-effective, accurate, and operates over CPU (Central Processing Unit) with minimal processing power.

At the intersection, regardless of the number of vehicles present over the lane, the traffic lights timer is in the constant round-robin phase. For instance, during the peak hours, the stream of traffic towards the south is significant, but decidedly less traffic flow in the other direction. However, the timer allocated to each lane remains constant, and as a repercussion, extraneous waiting time emerges. The Indian government has already established the CCTV (Closed-Circuit Television) cameras at the intersections and began using electronic format-based paperwork commonly known as e-challan, having a beneficial impact, but yet the dynamic traffic light not implemented. Additionally, in certain areas, the traffic police regulate traffic through hand-held traffic movements, which is arduous and cumbersome. Therefore, a simple and effective solution is bestowed to change the timer of the green light window based on the different classes of vehicles present over the lanes. Moreover, the precipitation details are taken into consideration. as it broadly affects the lane clearance time. State of the art, You Only Look Once (YOLOv4) (Bochkovskiy et al., 2020) algorithm is employed for detecting the vehicles and counting different classes of the vehicles since it is one stage detector and having higher accuracy along with lower inference time. The robust ensemble technique, eXtreme Gradient Boosting Algorithm (XGBoost) (Chen and Guestrin, 2016) is proposed for predicting the optimal time of the green light window as it is fast, efficient, accurate, and prune to overfitting. The prediction model is constructed based on analyzing the traffic patterns from the city of Vadodara during rush hours. Furthermore, Microsoft Common Object in Context (MS COCO) (Lin et al., 2014) dataset is being considered for detecting the different classes of vehicles. After constructing and fine-tuning the prediction and the detection model, the experimental results shows a reduction of 32.3% in the average waiting time of vehicles during the time interval of regular traffic.

Moreover, in this paper, different objection detection algorithms and regression-based prediction models are explored. Among them YOLOv4 (You Look Only Once version 4)(Bochkovskiy et al., 2020) and XGBoost (Extreme Gradient Boosting) (Chen and Guestrin, 2016) outperformed in all the constrained scenarios.

The remainder of the paper is organized as follows: Section 2 presents the related work addressing the Smart Traffic Management framework using different techniques. Section 3 describes the materials and proposed methodology along with algorithm for building the system's architecture. The models' results are discussed in Section 4, along with that the models' output is addressed. Section 5 concludes the paper by discussing possible future work for smart traffic management systems.

LITERATURE REVIEW

Preliminary approaches for smart traffic management systems are IoT-based approaches (Greengard, 2015), Time-Series based approaches (Jenkins, 1994), and Computer Vision-based approaches (Rafael, 1992). Although these approaches yield plausible results, they are expensive, not repurposing, less trustworthy, and less interpretable.

IoT Based Approach

(Kalaiselvi et al. (2017),Sharma et al. (2018)) have proposed a technology-driven solution, Light Fidelity (LiFi), that transmits signals to the traffic control system signaling an emergency vehicle's arrival. (Akhil and Parvatha, 2017) discussed the Sound Navigation and Ranging (SONAR) technique which measures the density of vehicles with an array of UltraSonic Sensors. The timer of the green light and red light is derived from SONAR readings. (Bui et al., 2017) proposed an IoT-based sensor network combined with the game theory. Every IoT-based entities such as vehicles, sensors, and traffic lights exchange the information in that proposed approach. The Cournot competition model for non-priority vehicles and the Stackelberg competition model for priority vehicles are adopted. Based on this game model, the required time to allocate for a traffic light is determined. (Atta et al., 2020) has discussed methodology using RFID sensor modules for sensing the density of vehicles and minimizing the congestion. For every incoming vehicle, the signal is sent to the RFID receiver, and accordingly, the count of vehicles is incremented.

98 Following that, the fuzzy inference is drawn to predict the estimated time. (Zambrano-Martinez et al.,
99 2019) proposed a load balancing algorithm in which traffic is routed to the particular lane where there is
100 lesser traffic. A route server is implemented to handle all of the city's traffic. It includes the SUMO and
101 DFROUTER tools, which produce data from real-world traffic traces. The ABATIS connection interface
102 connects two simulators, the SUMO traffic, and the OMNET++ network simulator. Furthermore, the
103 results are validated by injecting 34,065 vehicles. An 8% increase in travel time and a 16% improvement
104 under heavy loads is observed in their proposed approach.

Time-Series Forecasting

106 The time-series methods are based on historical data. (Natafgi et al., 2018) proposed an adaptive
107 reinforcement learning approach. In which the multiple agents are assigned to a crossroad and learn
108 optimal time and distance to travel based on reward and penalty of their actions. (Ata et al., 2019)
109 proposed Artificial Backpropagation Neural Network for the Smart Road Traffic Congestion, which
110 predicts time delay based on traffic speed, humidity, wind speed, and air temperature. Furthermore, (Chen
111 et al., 2016) refined this method using Genetic Algorithms. The number of vehicles heading towards the
112 green light and the vehicles halted at the red light are used as parameters. However, historical data is less
113 accurate when compared with real-time scenarios and can yield erroneous results.

Computer Vision and Machine Learning Based Approach

115 In the Digital Image Processing-based approach, the image is subtracted using foreground-background
116 subtraction with the reference image, and the blob of the object is obtained. Thereafter the traffic light
117 is controlled by counting the number of detected blobs (Choudekar et al. (2011),Frank et al. (2019)).
118 However, conventional image processing approaches are not much robust in terms of the changing light
119 conditions, and become skewed against dense traffic. Other proposed methods involve detecting and
120 tracking the rearmost vehicle in the frame (Asha and Narasimhadhan, 2018). However, since determining
121 the last vehicle is difficult, this approach is limited to a local solution. (Castaño et al., 2017) and
122 (Castaño et al., 2018) introduced the obstacle detection method where the object is recognized using
123 Support Vector Machine. Collision avoidance is accomplished using a Multi-Layer Perceptron and a
124 Self-Organizing Map. Along with that for collision detection, the 3DLiDAR technique is presented and
125 minimized using Reinforcement Learning. In order to recognize objects, the Support vector machine
126 (SVM) classifier is used. Nonetheless, the complexity of SVM escalates as the training sample size
127 increases, necessitating further computation and resulting in overfitting (Caruana and Niculescu-Mizil,
128 2006). However, Convolutional Neural Networks (CNN) are more advanced than Multi-Layer Perceptrons
129 because filters and kernels extract only the essential features from the image. (Harrou et al., 2020)
130 proposed a piecewise linear traffic (PWSL) and Kalman filter as a virtual sensor for estimating traffic
131 densities. The residuals from the actual and virtual sensors are fed as an input to the unsupervised
132 K-nearest neighbor (KNN) algorithm. The temporal clustering optical flow features based on Temporal
133 Unknown Incremental Clustering are proposed by (Kumaran et al., 2019). In that approach, the moving
134 cluster of objects are tracked based on the region of interest. These moving clusters determines the
135 density of vehicles on the lane. The signal time is estimated based on the type of vehicle rather than
136 the number of vehicles. Throughput and Average Waiting Time Optimization Model, which is based on
137 Gaussian regression, has been trained for the departure and arrival rates of different clusters. (Lv et al.,
138 2014) predicted the density of traffic at different roads. The 15,000 data points are gathered and analyzed
139 from different sensors. The stacked autoencoder is used to make the prediction. It consists of multiple
140 autoencoders stacked together followed by a logistic regression layer at the top. Furthermore, the results
141 of stacked autoencoders are compared with different neural networks.
142   This paper focuses on efficiently reducing excessive waiting time in real-time scenarios by using
143 computer vision-based object detection and machine learning-based prediction models. As a result, two
144 distinct studies are discussed in subsequent sections.

Overview Object detection Algorithms

146 Prior to advancing the deep Convolutional Neural Network (KrizhevskyA, 2012), Image Processing-based
147 methods for object detections were used. The most widely used methods were Scale-Invariant Feature
148 Transformation (SIFT) (Lowe, 2004), and the Histogram of Oriented Gradients (HOG) method (Dalal and
149 Triggs, 2005). Thereafter, a sliding window approach is introduced for object detection. Region-based
150 Convolutional Neural Network (R-CNN) (Girshick et al., 2014) has modified this approach by proposing

151 the "Region Proposals," which consist of obtaining a subset of the image and then classifying the object
152 using Convolutional Neural Network (Fukushima et al., 1983). Hence R-CNN is a two-stage detector,
153 and it requires more time for the inference.

154   You Only Look Once (YOLO) (Redmon and Farhadi, 2018) is a one-stage detector in which the
155 image is divided into the SxS grids, each of which acts as a classifier cell predicting the bounding box
156 and confidence ratio. The Fast R-CNN (Girshick, 2015), a variant of R-CNN, has an mAP of 70% with
157 0.5 FPS, while the YOLO-based model has 63.4% mAP with 155 FPS on Pascal VOC 2007 dataset
158 (Everingham, 2012). The mAP value is the mean of average precision, which combines the value of
159 precision and recall values given as $\frac{\sum_r P@r}{R}$ (Zhang and Zhang, 2009).

160   YOLOv2 (Redmon and Farhadi, 2017) employs the Batch Normalization, which improved accuracy
161 by 2%. Furthermore, the principle of anchor box prediction is introduced. The bounding box's dimensions
162 are predicted using KMeans Clustering. The mAP value of YOLOv2 is 76.8%, which is 3.6% higher than
163 Faster RCNN over Pascal VOC 2007 dataset (Everingham et al., 2010).

164   YOLOv3 improved the loss function, which is exhibited in Equation (1). The whole loss function of
165 YOLOv3 is formulated on regression loss, confidence loss, and classification loss. Moreover, the concept
166 of different layers for the different sizes of the object is introduced. In Equation, $1 1_{ij}^{obj}$ states that the
167 output value will be 1 if the box and cell value matches; otherwise, it will be 0. When their is no entity
168 1_{ij}^{noobj} reverses the output value.

$$
\begin{aligned}
\lambda_{\text{coord}} &\sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{\text{obj}} \ (xi - \hat{x}i)^2 + (yi - \hat{y}_i)^2 \\
+\lambda_{\text{coord}} &\sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{\text{obj}} \ \left(\sqrt{wi} - \sqrt{\hat{w}i}\right)^2 + \left(\sqrt{hi} - \sqrt{\hat{h}_i}\right)^2 \\
&+\sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{\text{obj}} \ \left(Ci - \hat{C}_i\right)^2 \\
&+\lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{\text{noobj}} \ \left(Ci - \hat{C}_i\right)^2 \\
&+\sum_{i=0}^{S^2} 1_{i}^{\text{obj}} \sum c \in \ \text{dasses} \ (p_i(c) - \hat{p}_i(c))^2
\end{aligned}
\tag{1}
$$

169   The YOLOv4 achieves an mAP value of 43% with 43 FPS, whereas FasterRCNN achieves 39.8%
170 mAP with 9.4 FPS over Titan X Pascal GPU.

171   The Paddle-Paddle YOLO (PP-YOLO) (Long et al., 2020), is the recent advancement based on Paddle
172 Paddle Detector Framework. It employs ResNet50-vd as Backbone, Feature Pyramid Network (FPN)
173 with DropBlock regularization as Neck and YOLOv3 as Head. The overall AP value is increased to
174 1.5% by introducing Intersection over Union (IoU) Loss and Grid Sensitive modules. Furthermore, the
175 Matrix-based Non-Max Suppression and the CordConv increases mAP value by 0.8%.

176   The YOLO-based object detection framework outperformed since it detects the objects in a single
177 shot. In this paper, the different novel YOLO architectures and their implementations are compared. The
178 metrics AP_{50}, the average precision at IoU of 50, and AP_M, the average precision with the object size
179 medium are considered for comparison.

180   In this paper, for the Smart Traffic Management System, the YOLOv4 based architecture is employed
181 to detect the vehicles.

182 **Overview of Machine Learning Prediction Models**
183 Machine learning-based regression models are categorized into two types: polynomial curve-based
184 approach and decision tree-based approach.

185   The Elastic Net (Zou and Hastie, 2008) and Support Vector Machine regressor (Scholkopf, 1998)
186 are based on polynomial approach. While the Random Forest (Breiman, 1996) and eXtreme Gradient
187 Boosting (Chen and Guestrin, 2016) are the decision tree-based approach.

The Elastic Net is based on the Lasso and Ridge regularization methods. The regularization allows to train only the essential features of the model while penalizing the rest and prevent from overfitting. Equation 2, represents the polynomial regression, in which the corresponding constants b are multiplied for each polynomial input x. These values optimized using Gradient Descent Optimizer. Equation 3, represents the Gradient Descent cost function. The y_i represents the actual feature value and $\beta' x_i$ is the predicted value and λ_1 and λ_2 are the regularization constants.

$$\operatorname{argmin}_\beta y = b_0 + b_1 x_1 + b_2 x_1^2 + \ldots + b_n x_1^n \tag{2}$$

$$\operatorname{argmin}_\beta \sum_i \left(y_i - \beta' x_i \right)^2 + \lambda_1 \sum_{k=1} |\beta_k| + \lambda_2 \sum_{k=1} \beta_k^2 \tag{3}$$

In the Support Vector Machine, the polynomial kernel computes the optimal fit hyperplane by scaling down the maximum margin classifier's error rate within a threshold ε. Thereafter, the hyperplane function is computed by optimizing the primal function (Vapnik, 1995). The support vectors are the data points that are near to the boundary of hyperplane and at ε distance far. Equations (4) and (5) represent the kernel equation of the Support Vector Machine.

$$y = \sum_{i=1}^N (\alpha_i - \alpha_i^*) \cdot \langle \varphi(x_i), \varphi(x) \rangle + b \tag{4}$$

$$y = \sum_{i=1}^N (\alpha_i - \alpha_i^*) \cdot K(x_i, x) + b \tag{5}$$

Random forest is a bagging technique based on decision trees, whereas the Gradient Boosting tree (GBT) is an ensemble boosting technique. Bagging is a technique that determines the output by considering all the values of the decision tree. The Information Gain based on Gini Index and Entropy determines the split of the decision tree. The Gini Index and Entropy are the two methods used to calculate the Information Gain, depicted in Equations 6 and 7. The boosting technique involves weight adjustment based on the previous decision tree prediction. The residual error $\bar{Y} = Y | Fo(x)$ is determined in both the bagging and boosting techniques. This residual error is added to the initial value and fine-tuned until the value approaches close to the ground truth values. However, the eXtreme Gradient Boosting (XGBoost) is an advanced Gradient Boosting Tree Algorithm. The built-in cross-validation capability, efficient handling of missing data, regularization to avoid overfitting, catch awareness, tree pruning, and parallelized tree building are all features that contribute to XGBoost's robustness. Equation 8, shows the objective function of XGBoost, which is solved using the Second-order Taylor polynomial approximation.

$$\text{Entropy} = \sum_{i=1}^C -p_i * \log_2(p_i) \tag{6}$$

$$\text{GiniIndex} = 1 - \sum_{i=1}^C (p_i)^2 \tag{7}$$

$$\mathcal{L}^{(t)} = \sum_{i=1}^n l\left(y_i, \hat{y}_i^{(t-1)} + f_t(\mathbf{x}_i)\right) + \Omega(f_t) \tag{8}$$

MATERIALS AND METHODOLOGIES

Proposed Methodology for Smart Traffic Management System

This section discusses the proposed methodology for Smart Traffic Management System. The proposed method is simple, robust, accurate, and applicable to traffic circles, cross-roads, and fly-overs. Figure 1
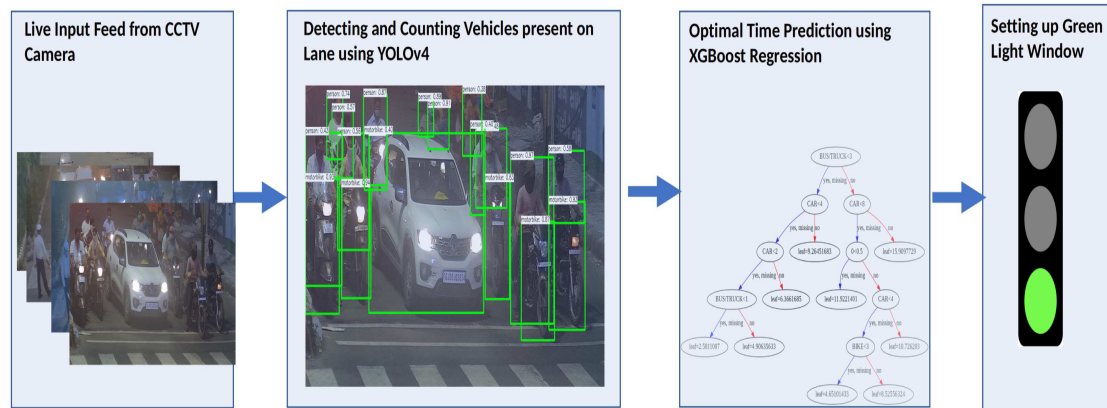
Figure 1. Flowchart for the Smart Traffic Management System

presents the framework of the proposed system. The architecture comprises two phases. The first phase involves object detection, and the second phase involves Machine-Learning based regression. Object Detection detects the different vehicle classes, whereas Machine Learning based regression predicts the optimal time of the green light window.

The initial step of the proposed approach is to fetch the IP address of the CCTV camera from each lane. After that, the region of interest is determined. The Perspective Transformation (Shakunaga and Kaneko, 1989) Algorithm is utilized for determining the region of interest.

The obtained frame is processed by the State-of-the-Art YOLOv4 object detector, which counts the number of cars, buses, trucks, and bikes on the road. Furthermore, the precipitation information is fetched from the Open Weather Map API (OlgaUkolova, 2017) based on the latitude and longitude of the CCTV Camera's location since precipitation has a significant impact on lane clearance time. Every hour, the precipitation status is retrieved.

The count of different vehicles' categories and the precipitation information passed as an input to the Machine Learning based Prediction model. This paper proposes the eXtreme Gradient Boosting Algorithm (XGBoost) (Chen and Guestrin, 2016), which is fast, efficient, accurate, and prune to overfit. The XGBoost predicts the optimal lane clearance time for the green light window once the assigned previous lane's timer is completed.

The entire process works seamlessly without creating any buffer zone in processing time. The processing of the next lane is completed in the last 5 seconds when there is a yellow light. As a result, lane processing and prediction are performed once in a complete sprint. This flow of the whole system is depicted in Algorithm 1.

Our algorithm's main modules are 1) Obtaining the vehicles' density using the YOLOv4 Object Detection Model and 2) Predicting the optimal time for the green light window using the eXtreme Gradient Boost Prediction Model, which is briefly described in the subsequent sections.

Object Detection

The OpenCV-based Leaky YOLOv4 is proposed for analyzing the density of vehicles presented over the lane. The classes that are taken into consideration for vehicles' density are Car, Heavy Loaded Vehicles, and Bike. Fixed confidence of 25% while 5% for the Tiny model, NMS threshold of 50% are configured as hyper-parameter for Object Detection. After that, the count of vehicles is passed to determine the prediction time.

Prediction Model

Polynomial curve fitting and tree-based regression models are proposed to predict the green light window timer. The methodology followed for training the model is to split the dataset into the training part and testing part. Furthermore, the 10-fold cross-validation technique (Browne, 2000) is applied, which divides the training data into the training set and validation set. The model is first trained on training data and then fine-tuned over validation data. Finally, the model's unbiased evaluation is performed on the unseen testing dataset.

Algorithm 1: Algorithm for Green Light Prediction

input : Set of IP Addresses (N)
        Set of Region of Interest (N)

output : Predicted time of Green Signal

begin

    $LANE \longleftarrow 0$;

    $TIMER \longleftarrow 0$;

    Loop

        $LANE \longleftarrow (LANE + 1)modN$;

        $VEHICLES \longleftarrow YOLOv4.Detect(frame)$;

        foreach *objects obj in VEHICLES* **do** $COUNT_{VEHICLES}[obj] \longleftarrow obj.value$;

        $RAIN \longleftarrow 0$;

        $isPrecipitation \longleftarrow WEATHER\_API(LAT, LON)$;

        if *isPrecipitation* **then**

            $RAIN \longleftarrow 1$;

        $PREDICTED\_TIME \longleftarrow XGBoost.Predict(COUNT_{VEHICLES})$;

        while *TIMER not* 0 **do**

            WAIT;

        $TIMER \longleftarrow PREDICTED\_TIME$;

        $GREEN\_SIGNAL \longleftarrow TRUE$;

        if *TIMER is* 5 **then**

            $YELLOW\_SIGNAL \longleftarrow TRUE$;

Along with the count of vehicles, the precipitation details are fetched from Open Weather Map API (OlgaUkolova, 2017) in one hot encoded form and conveyed to the model.

The predicted time is set to the green light window once the previous lane's timer is completed. When the last 5 seconds remain, the light of the current lane switches from green to yellow, and the next lane's processing is performed.

This process is repeated for each lane. Therefore in our proposed algorithm, the processing of lanes and predictions are performed only once per cycle.

Datasets

The proposed architecture comprises of two models Object Detection Model and Prediction Model. Hence two distinct datasets are used.

1) The Microsoft Common Objects in Context (MS COCO) dataset (Lin et al., 2014) is considered for object detection. This dataset is often used for performing classification, detection, and segmentation. The dataset consists of 91 different types of objects with around 328K images. The Car, Bus, Truck, and Bike classes are segregated, and the vehicle's count is passed to the prediction model.

2) A dataset of 1128 sample points from six different crossroads in Vadodara City, Gujarat, India, is acquired for training the prediction model. These particular crossroads are illustrated in Figure 2. The total dataset is partitioned into 90% of the training set and 10% of the testing set, which comprises 1015 as the sample training data points and 113 testing data points. For the validation, the 10-Fold cross-validation (Browne, 2000) technique is utilized, which further split the training dataset into the validation dataset; with the equal partition of 10 blocks.

RESULTS AND DISCUSSION

The proposed Algorithm is a two-step approach. In the first step, the count of the independent category of vehicles is obtained, which serves as input to the next step for predicting the optimal time of the green light window. Consequently, the whole approach is the amalgamation of detection using the YOLOv4 (You Only Look Once version4) object detection Algorithm and prediction using the XGBoost (eXtreme Gradient Boosting) Algorithm; As a result, it is an amalgamation of YOLOv4 and XGBoost for regulating the Traffic present at the lane.
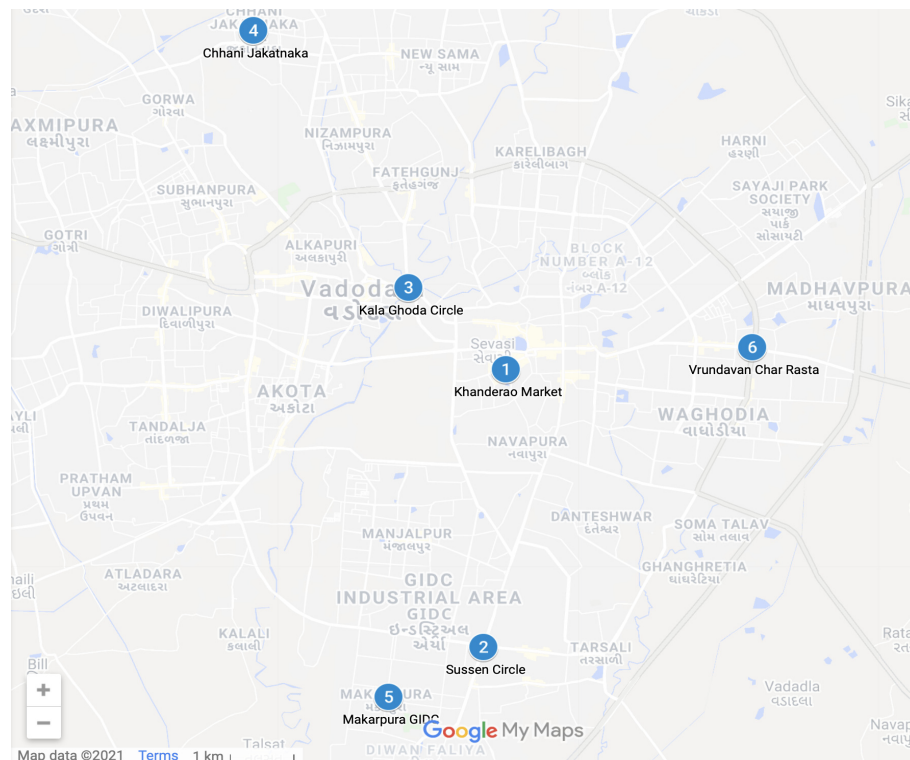
Figure 2. Cross-Roads undertaken for analytics from Vadodara City - (Map data ©2021 Google)

Analysis of Object Detection Algorithm

The inference time is essential in traffic management; hence, it is reasonable to construct a better accuracy model with lesser inference time. Comparative performance analysis of different YOLO-based detectors is performed to evaluate the accuracy and inference time over CPU in a constrained environment. The six different models are proposed and evaluated including OpenCV DNN based leaky and Mish YOLOv4 (Bradski, 2000), Open Neural Network Exchange (ONNX) based YOLOv4 (Bai et al., 2019), PP-YOLO (Long et al., 2020), Darknet YOLOv4, and Darknet YOLOv4 lite (Bochkovskiy et al., 2020) from MS COCO dataset. A fixed confidence level of 5% is defined for the tiny model in the experiment, and a threshold of 25% is defined for the remaining models. The NMS threshold is set to 50%, with an input image size of 1359x720x3 for Day Time and 1366x768x3 for Evening Time. The network resolution is set to 416x416x3 pixels. Figures 3 and 4, depict the study of the different object detection algorithms during the day and at night under low - light conditions.

The inference and mAP analysis is described in Table 1. The accuracy and inference time of object detection models are evaluated under the constrained environment with 1xsingle core hyperthreaded Xeon Processors @2.3GHz (1 core, 2 threads) with 12.6 GB RAM. From this analysis, it is concluded that lite models are remarkably fast, but the error rate is high, resulting in a significant impact over predicting the optimal time. Darknet YOLOv4 is highly accurate, but the inference time exceeds 5 seconds, making it infeasible. Although the PP-YOLO using conventional greedy NMS is more accurate and faster than Darknet YOLOv4, its inference time is closer to 5 seconds. The AP value of YOLOv4 ONNX and OpenCV DNN Mish is the same; however, since OpenCV is substantially optimized for CPU, OpenCV DNN has a lesser inference time than ONNX. Equations 9 and 10 represent the Mish Activation Function and Leaky Activation Function respectively.

$$
\begin{aligned}
f(x) &= x \cdot \tanh(softplus(x)) \\
&= x \cdot \tanh\left(\ln\left(1 + e^x\right)\right)
\end{aligned}
\tag{9}
$$

$$
\text{Leaky ReLU } \max(0.1x, x)
\tag{10}
$$
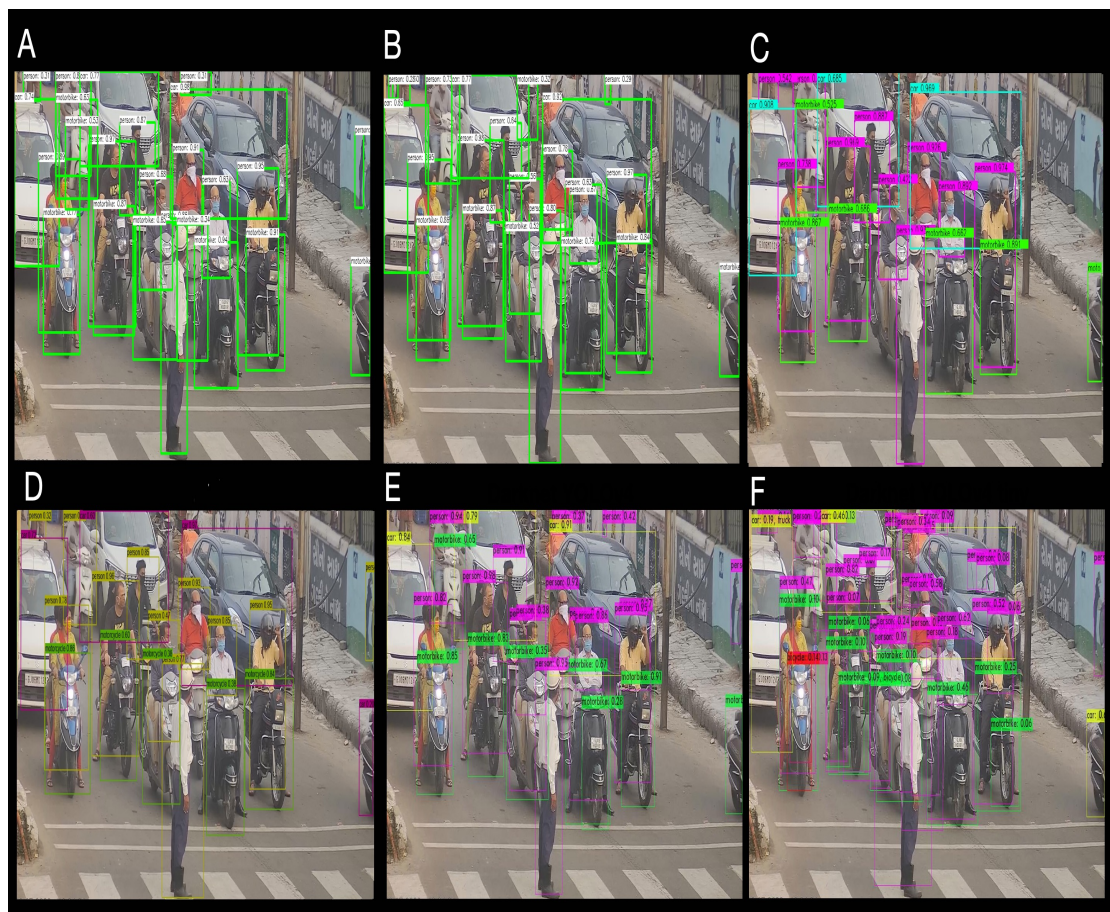
Figure 3. Different object detection algorithm analysis for traffic at cross-road in the day time
A) OpenCV DNN Leaky YOLOv4 B) OpenCV DNN Mish YOLOv4 C) ONNX YOLOv4 D) PP-YOLO
E) Darknet YOLOv4 F) Darknet YOLOv4 Tiny

In YOLOv4, a combination of Mish function + Cross Stage Partial Network (CSPDarknet53) is utilized, which, although slightly expensive in inference time but significantly improves detection accuracy. The Mish and Leaky YOLOv4 are quite close. As a result, OpenCV-based Leaky YOLOv4 is proposed for smart traffic management system (Bradski, 2000).

Analysis of Prediction Model

Elastic Net, Support Vector Regressor, Random Forest Regressor, and Extreme Gradient Boosting algorithms are analyzed to determine the prediction model. Elastic Net and Support Vector Regressors are polynomial-based regressors, while Random Forest and Extreme Gradient Boosting are decision tree-based regressors. Table 2 details the fine-tuned hyper-parameters and regression analysis of each model across the training, validation, and test datasets. The Bayesian parameter optimization technique from the hyperparameter framework "Optuna" is considered for parameter optimization and function subset selection (Akiba et al., 2019). The r-squared coefficient of determination and the mean squared error (MSE) is used as an evaluation metrics, depicted in Equations 11,12 and 13. Based on the analysis findings, it is concluded that the decision tree-based models are best fit than polynomial-based models. Furthermore, studies shows that Elastic Net performed poorly with a r2 score of 0.638 and an MSE value of 21.79 over unseen samples, whereas Extreme Gradient Boosting outperformed and shows promising results with a r2 score of 0.92 and an MSE value of 5.53.

$$r^2 = 1 - \frac{Sum\ of\ Square\ of\ Residual\ Error\ (ss_{res})}{Sum\ of\ Suare\ of\ Total\ Variation\ (ss_{tot})} \tag{11}$$
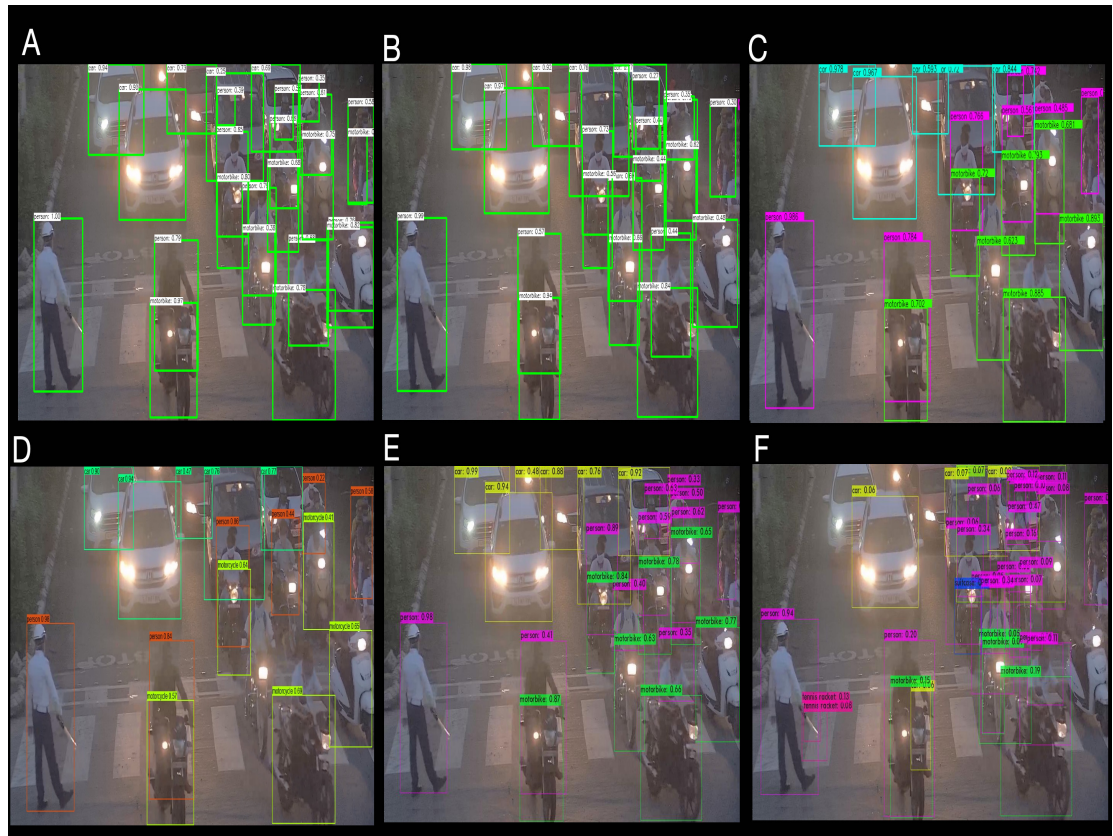
Figure 4. Different object detection algorithm analysis for traffic at cross-road in the night time
A) OpenCV DNN Leaky YOLOv4 B) OpenCV DNN Mish YOLOv4 C) ONNX YOLOv4 D) PP-YOLO
E) Darknet YOLOv4 F) Darknet YOLOv4 Tiny

$$SS_{res} = \sum_i ((y_i) - f_i)^2 = \sum_i e_i^2 \quad and \quad SS_{tot} = \sum_i ((y_i) - (\bar{y}))^2 \tag{12}$$

$$MSE = \frac{1}{n} \sum_{i=1}^{n} (y_i - \overline{y}_i)^2 \tag{13}$$

306   The randomly selected data points from the testing dataset are analyzed in Table 3. The XGBoost
307 prediction model considers the number of vehicles and precipitation details to determine the optimal time.
308 Thereafter, the predicted time is compared with the static time, and the calculation of reduced waiting
309 time is performed, which is depicted in Equation 14. The reduced average waiting time is 32.3% in
310 comparision with the static time allocated to the usual road traffic based on the test dataset. Hence the
311 proposed methodology reduces the intersection's waiting time and improves the traffic lights' efficiency.

$$\frac{1}{N} \sum_{i=1}^{N} \frac{(\text{Original time} - \text{Predicted time})}{\text{Original time}} * 100\% \tag{14}$$

   The histogram along with kernel density estimation (KDE) for prediction model is depicted in Figure
5. The data distribution is based on the density of vehicles against the time predicted by the model. The
probability density of continuous or non-parametric data variables is visualised using the Kernel Density
Estimate. It is calculated as the region in the interval between the density function (graph) and the x-axis
which is represented in the Equation 15. The b is representing the bandwidth of bin, the kernel density
function $K(.)$ is the chosen Kernel weight function and x_i is the observing data point. The KDE reflects
how far the individual data point is from the mean data point in the same bin. This plot clearly shows that
the distribution is not skewed. The average predicted value under normal traffic is 46.834 seconds while

Table 1. YOLO Object Detection Comparison between YOLOv4 ONNX,YOLOv4 Darknet,YOLOv4 Darknet Tiny YOLOv4,PP-YOLO,OpenCV Leaky YOLOv4 and OpenCV YOLOv4.
The inference time and accuracy is calculated by using fixed computational environment .

Architecture	Inference time inDay light	Inference time in Evening (low light)	AP50 (416x416)	APM (416x416)
YOLOv4 ONNX	∼3.1327 sec	∼3.168 sec	**63.3%**	44.4% AP
YOLOv4 Darknet	∼8.865 sec	∼8.965 sec	**63.3%**	44.4% AP
YOLOv4 Darknet Tiny	**∼0.44 sec**	**∼0.1 sec**	40.2%	-
PP-YOLO	∼4.489 sec	∼4.468 sec	62.8%	**45.2% AP**
OpenCV Leaky YOLOv4	∼1.40821 sec	∼1.4109 sec	62.7%	43.7% AP
OpenCV Mish YOLOv4	∼1.6733 sec	∼1.679 sec	**63.3%**	44.4% AP

Table 2. Regression Model Comparison between Elastic Net, Support Vector Machine Regressor (SVR),Random Forest regressor and Extreme Gradient Boosting tree based (XGBoost GBT).
The hyperparameters of each model are optimized for Training, Validation and Testing set.

Model	Hyper-parameters	Training set	Cross validation	Testing set
Elastic Net	Degree: 2 Interaction: True Learning Rate: 0.05 L1 ratio: 0.5	R2: 0.7891 MSE: 34.56	R2: 0.7936 MSE: 36.672	R2: 0.638 MSE: 21.79
SVR	Kernel: poly Degree: 3 C: 4.688	R2: 0.8207 MSE: 35.157	R2: 0.7964 MSE: 35.323	R2: 0.7321 MSE: 22.39
Random Forest regressor	N estimators: 120 Max depth: 58	R2: 0.98 MSE: 3.592	R2: 0.8932 MSE: 18.918	R2: 0.8933 MSE: 19.16
XGBoost GBT	N estimators: 76 max depth: 120 learning rate: 0.35 gamma: 0.018 base score: 0.578	**R2: 0.9944 MSE: 1.113**	**R2: 0.9384 MSE: 11.576**	**R2: 0.92179 MSE: 5.513**

Table 3. Waiting Time Reduced by XGBoost in comparison to Static Time.
The parameters used for prediction are number of Cars, Bus or Trucks, Bike and Precipitation.

CAR	BUS and TRUCK	BIKE	RAIN	XGBoost predicted (in sec)	Static Time (in sec)	Reduced Waiting Time
1	6	5	YES	48	60	20%
4	5	2	NO	48	60	20%
5	2	5	YES	40	60	33%
7	1	3	YES	44	60	27%
2	4	5	YES	35	60	42%
3	2	4	NO	32	60	47%
4	1	1	YES	27	60	55%

the median value is 45.01 seconds. These indicates that their is a symmetric distribution in predicted time with respect to density of traffic. The Minimum predicted time is 14.98 seconds and the maximum time predicted is 100.03 seconds.

$$\hat{f}^{\text{Kernel}}(x) = \frac{1}{Nb} \sum_{i=1}^{N} K\left(\frac{x_i - x}{b}\right) \tag{15}$$

Figure 6 highlights the variation of time based on the density of vehicles. The range that is considered is from 0% to 100% with an interval of 25% and the allocated times are 15 to 40, 25 to 60, 40 to 90, and 75 to 100 seconds for each interval. The average time predicted in 25% density is 35.84 seconds, for 50% density is 45.017 seconds, for 75% density is 54.91 seconds and 87.32% seconds in 75% to 100% density. The average difference between labelled time and model prediction is minimal, indicating that the model has a perfect fit over the Gaussian probability density curve. The comparison with the actual allocated static time over the Gaussian probability distribution is presented in Figure 7. This figure demonstrates that the proposed solution effectively decreases waiting time as opposed to the static allocated time.
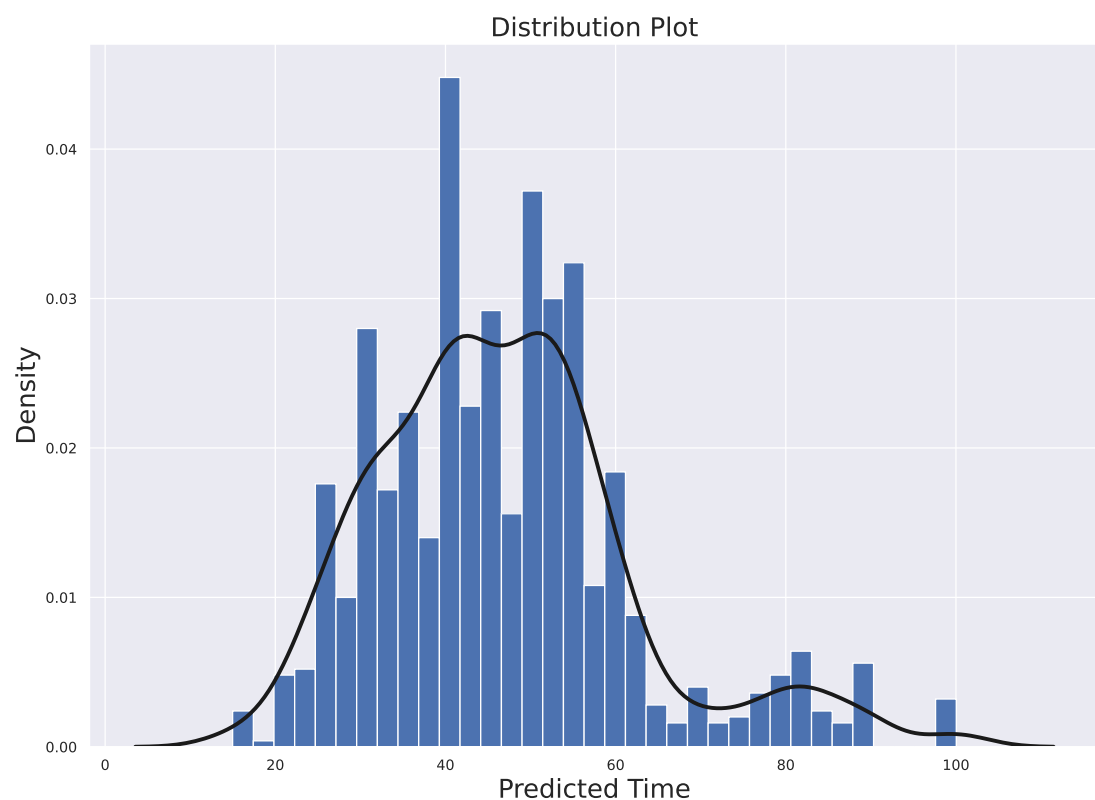
Figure 5. Histogram along with Kernel Density Estimation for the predictive model

CONCLUSION

In this work, a unified Algorithm for the Smart Traffic Management System is introduced to address traffic congestion by predicting the optimal time of the green light window. The proposed approach replaces the conventional method of allocating timer in a round-robin pattern with the dynamic time allocation. The research includes the use of incoming frames from the CCTV cameras. As a result, integrating it with traffic lights is simpler and efficient. The proposed approach is scalable as the system works appropriately during the high stream and in low traffic and it is cost-efficient as it does not require much maintenance or any sensors; instead, it is completely based on input frames from the CCTV cameras.

Furthermore, the proposed approach is robust, versatile, and works efficiently during the day and at night time with low street lights, and it works flawlessly under all weather conditions. An additional
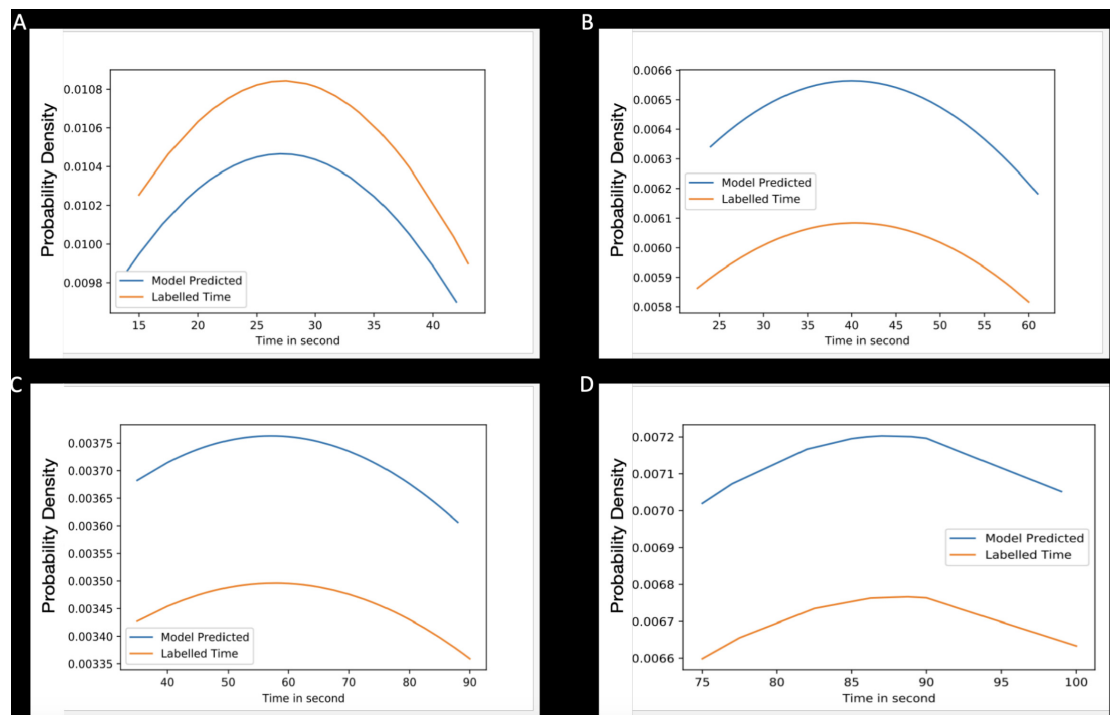
Figure 6. **Average value time between model predicted and labelled for training**
A) 25% of Vehicles B) 50% of Vehicles C) 75% of Vehicles D) 100% of Vehicles

monitoring system is provided to manage the failure, in which the notification sent to the control room, and traffic lights can be handled manually. The real-time traffic data is used and gives predictions based on the independent category of vehicles present on the lane. As a result, the unnecessary waiting time and fuel consumption are narrow down, resulting in lower air pollution. As the paper modernizes and renovates the conventional methodology of Traffic Management System with an Artificial Intelligence-based approach, it is theorized as a Smart Traffic Management System.

The future scope is to implement the lane clearance for an emergency vehicle, assessing the weather by classifying the frame, considering specific time events such as rush hour and peak hours into the effect.

ACKNOWLEDGEMENTS

This research is supported by Vadodara Municipal Corporation (VMC) and Charotar University of Science and Technology (CHARUSAT). The authors gratefully thank to Mr. Manish Bhatt, IT Director of VMC for providing the valuable video data and GPU workstation by CHARUSAT for our experiments. We also gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan X Pascal GPU used for this research.

REFERENCES

Akhil, J. and Parvatha, S. K. A. (2017). Dynamic traffic control system based on sonars. In *2017 IEEE International Conference on Smart Technologies and Management for Computing, Communication, Controls, Energy and Materials (ICSTM)*, pages 385–388. IEEE.

Akiba, T., Sano, S., Yanase, T., Ohta, T., and Koyama, M. (2019). Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Asha, C. S. and Narasimhadhan, A. V. (2018). Vehicle counting for traffic management system using yolo and correlation filter. In *2018 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT)*, pages 1–6.
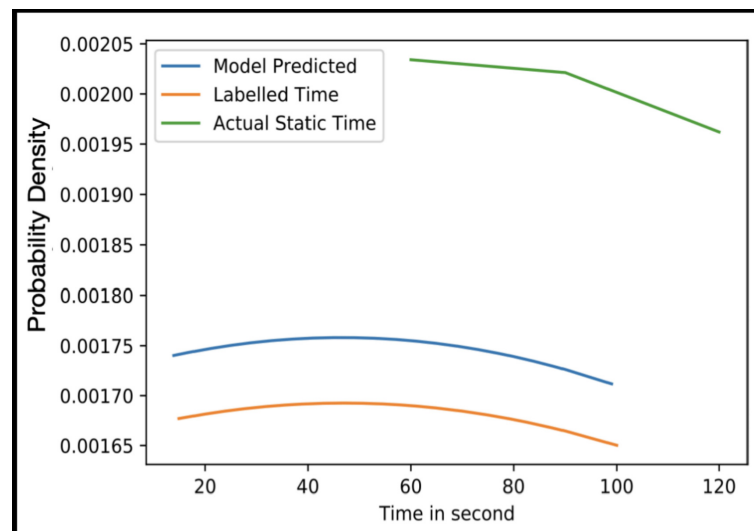
Figure 7. Average value time

355 Ata, A., Khan, M. A., Abbas, S., Ahmad, G., and Fatima, A. (2019). Modelling smart road traffic
356   congestion control system using machine learning techniques. *Neural Network World*, 29(2):99–110.
357 Atta, A., Abbas, S., Khan, M. A., Ahmed, G., and Farooq, U. (2020). An adaptive approach: Smart traffic
358   congestion control system. *Journal of King Saud University - Computer and Information Sciences*,
359   32(9):1012–1019.
360 Bai, J., Lu, F., Zhang, K., et al. (2019). Onnx: Open neural network exchange. url-
361   https://github.com/onnx/onnx.
362 Bochkovskiy, A., Wang, C.-Y., and Liao, H.-Y. M. (2020). Yolov4: Optimal speed and accuracy of object
363   detection. *arXiv preprint arXiv:2004.10934*.
364 Bradski, G. (2000). The opencv library. *Dr Dobb's J. Software Tools*, 25:120–125.
365 Breiman, L. (1996). Bagging predictors. *Machine learning*, 24(2):123–140.
366 Browne, M. W. (2000). Cross-validation methods. *Journal of mathematical psychology*, 44(1):108–132.
367 Bui, K.-H. N., Jung, J. E., and Camacho, D. (2017). Game theoretic approach on real-time decision
368   making for iot-based traffic light control. *Concurrency and Computation: Practice and Experience*,
369   29(11):e4077.
370 Caruana, R. and Niculescu-Mizil, A. (2006). An empirical comparison of supervised learning algorithms.
371   In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, page 161–168,
372   New York, NY, USA. Association for Computing Machinery.
373 Castaño, F., Beruvides, G., Haber, R. E., and Artuñedo, A. (2017). Obstacle recognition based on machine
374   learning for on-chip lidar sensors in a cyber-physical system. *Sensors*, 17(9).
375 Castaño, F., Beruvides, G., Villalonga, A., and Haber, R. E. (2018). Self-tuning method for increased
376   obstacle detection reliability based on internet of things lidar sensor models. *Sensors*, 18(5).
377 Chen, T. and Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd
378   acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794.
379 Chen, Y.-R., Chen, K.-P., and Hsiungy, P.-A. (2016). Dynamic traffic light optimization and control system
380   using model-predictive control method. In *2016 IEEE 19th International Conference on Intelligent
381   Transportation Systems (ITSC)*, pages 2366–2371. IEEE.
382 Choudekar, P., Banerjee, S., and Muju, M. K. (2011). Implementation of image processing in real
383   time traffic light control. In *2011 3rd International Conference on Electronics Computer Technology*,
384   volume 2, pages 94–98.
385 Dalal, N. and Triggs, B. (2005). Histograms of oriented gradients for human detection. In *2005 IEEE
386   computer society conference on computer vision and pattern recognition (CVPR'05)*, volume 1, pages
387   886–893. IEEE.
388 Everingham, M. (2012). Van gool, l. *Williams, CKI, Winn, J. and Zisserman, A. The PASCAL Visual
389   Object Classes Challenge*.

Everingham, M., Van Gool, L., Williams, C. K., Winn, J., and Zisserman, A. (2010). The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338.

Frank, A., Khamis Al Aamri, Y. S., and Zayegh, A. (2019). Iot based smart traffic density control using image processing. In *2019 4th MEC International Conference on Big Data and Smart City (ICBDSC)*, pages 1–4.

Fukushima, K., Miyake, S., and Ito, T. (1983). Neocognitron: A neural network model for a mechanism of visual pattern recognition. *IEEE transactions on systems, man, and cybernetics*, (5):826–834.

Girshick, R. (2015). Fast r-cnn. arxiv e-prints. *arXiv preprint arXiv:1504.08083*, 454.

Girshick, R., Donahue, J., Darrell, T., and Malik, J. (2014). *Rich feature hierarchies for accurate object detection and semantic segmentation*.

Greengard, S. (2015). *The internet of things*. MIT press.

Harrou, F., Zeroual, A., and Sun, Y. (2020). Traffic congestion monitoring using an improved knn strategy. *Measurement*, 156:107534.

IndianMinistryOfStatistics (2019). *Environmental Statistics Volume 1 2019*.

Jenkins, G. M. (1994). *Time Series Analysis*. Prentice Hall.

Kalaiselvi, V., Sangavi, A., and Dhivya (2017). Li-fi technology in traffic light. *2017 2nd International Conference on Computing and Communications Technologies (ICCCT)*, pages 404–407.

KrizhevskyA, S. (2012). Hintongeimagenetclassificationwithdeepconvolutionalneural networks. *Proceedingn IPS'12Proceedingsofthe 25thInternationalconferenceonneuralInformation ProcessingSystems*.

Kumaran, S. K., Mohapatra, S., Dogra, D. P., Roy, P. P., and Kim, B.-G. (2019). Computer vision-guided intelligent traffic signaling for isolated intersections. *Expert Systems with Applications*, 134:267–278.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. (2014). Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer.

Long, X., Deng, K., Wang, G., Zhang, Y., Dang, Q., Gao, Y., Shen, H., Ren, J., Han, S., Ding, E., et al. (2020). Pp-yolo: An effective and efficient implementation of object detector. *arXiv preprint arXiv:2007.12099*.

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110.

Lv, Y., Duan, Y., Kang, W., Li, Z., and Wang, F.-Y. (2014). Traffic flow prediction with big data: a deep learning approach. *IEEE Transactions on Intelligent Transportation Systems*, 16(2):865–873.

Natafgi, M. B., Osman, M., Haidar, A. S., and Hamandi, L. (2018). Smart traffic light system using machine learning. In *2018 IEEE International Multidisciplinary Conference on Engineering Technology (IMCET)*, pages 1–6.

OlgaUkolova (''2017''). *Open Weather Map*.

Rafael, C. (1992). Gonzalez, and richard e. woods. *Digital image processing*.

Redmon, J. and Farhadi, A. (2017). Yolo9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7263–7271.

Redmon, J. and Farhadi, A. (2018). Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*.

Scholkopf, B. (1998). Support vector machines: a practical consequence of learning theory. *IEEE Intelligent systems*, 13.

Sharma, P. K., Ryu, J. H., Park, K. Y., Park, J. H., and Park, J. H. (2018). Li-fi based on security cloud framework for future it environment. *Human-centric Computing and Information Sciences*, 8(1):1–13.

Szeliski, R. (2010). *Computer vision: algorithms and applications*. Springer Science & Business Media.

Vapnik, V. N. (1995). The nature of statistical learning. *Theory*.

Zambrano-Martinez, J. L., Calafate, C. T., Soler, D., Lemus-Zúñiga, L.-G., Cano, J.-C., Manzoni, P., and Gayraud, T. (2019). A centralized route-management solution for autonomous vehicles in urban areas. *Electronics*, 8(7):722.

Zhang, E. and Zhang, Y. (2009). *Average Precision*, pages 192–193. Springer US, Boston, MA.

Zou, H. and Hastie, T. (2008). Model building and feature selection with genomic data. *Computational Methods*, page 393.