International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

Smart Traffic: Integrating Machine Learning, and YOLO for Adaptive Traffic Management System

Nitin Sakhare[1,*], Mrunal Hedau[2], Gokul B.[3], Omkar Malpure[4], Trupti Shah[5], Anup Ingle[6]

Submitted: 22/11/2023 **Revised**: 30/12/2023 **Accepted**: 10/01/2024

Abstract- The growing number of vehicles has made traffic control a vital concern, rendering traditional manual solutions ineffective. This research proposes an innovative approach that makes use of the Internet of Things (IoT) and sophisticated image processing. Using image processing, the adaptive traffic management system analyses real-time data from camera-monitored lanes, precisely recognizing and enumerating cars. A sophisticated algorithm computes appropriate waiting periods based on lane-specific vehicle numbers, which informs the prudent distribution of signal light patterns. This method considerably decreases average wait times, improving traffic-clearing efficiency. Furthermore, by reducing CO2 emissions, the technology helps to preserve the environment. Its flexibility in emergency settings emphasizes its usefulness. This study highlights the potential of IoT-driven adaptive traffic management in producing efficient, environmentally friendly, and responsive urban traffic systems.

Keywords: *IoT-driven, lane-specific, considerably, environmentally, enumerating*

Introduction

In India's contemporary urban landscape, the popular manual traffic control system operates at fixed intervals to adjust traffic lights. However, this approach is not efficient due to the inherent disparity in traffic density that fluctuates during the day. As a result, vehicles often have to wait a long time, even when the traffic density is low or nonexistent. This is mainly due to strict adherence to fixed-time protocols for light transitions. As the Indian economy continues to grow at a rapid pace, marked by impressive annual GDP growth, the flow of private and freight vehicles has increased. However, this rapid economic expansion has also created challenges such as traffic congestion, which has adverse effects on daily commutes. This growing traffic conundrum is well illustrated by the average travel time in major Indian cities such as Delhi, Mumbai, Bangalore, Kolkata, and Pune where passengers spend more than 1.5 hours per day compared to passengers in other localities. This worrisome scenario is underscored by peak-hour congestion, which reached an alarming 149% in these cities, eclipsing the Asian average of 67%. Furthermore, the economic consequences of such traffic congestion are obvious, reflected in the significant loss of time and increased fuel consumption. However, the predicament is not only a waste of time but also includes environmental ramifications [1]. The link between traffic congestion and increased pollution levels is clear, with urban areas, characterized by an increase in the number of vehicles, bearing a high burden of air and noise pollution. In addition, fuel consumption exacerbated by stop-and-go traffic dynamics contributes to increased carbon dioxide emissions, further exacerbating the ecological footprint. In this context, the main objective of this study is to improve existing traffic management models. Although alternative strategies such as toll control systems or infrastructure expansion exist, they face problems of feasibility and inefficiencies. Therefore, the study aims to conceptualize a dynamic traffic management system capable of adapting to fluctuating traffic densities, underpinned by the Internet of Things (IoT). The proposed project includes three basic components: detects vehicles, counts vehicles per lane, and dynamically adjusts signal timing based on real-time traffic conditions. By seamlessly integrating these components, the Adaptive Traffic Management system aims to minimize vehicle delays and stops at intersections, leveraging real-time data to optimize traffic control. traffic signal distribution. Realizing the cost-effectiveness of signal time optimization to improve travel time and travel speed in urban transport systems, this study wishes to reduce the average waiting time at intersections. In turn, this has the potential to significantly reduce CO2 emissions and pollution, paving the way for more efficient and sustainable urban mobility [1].

* *nitin.sakhare@viit.ac.in*

1,2,3,4,6 Vishwakarma Institute of Information Technology, Pune

5 Thakur College of Engineering and Technology, Mumbai

Literature Survey

The traffic management sector has seen an explosion of innovative solutions thanks to technological advancements. This literature review explores several notable research articles focusing on real-time traffic control, adaptive systems, and intelligent traffic management using various technologies. The studies presented below provide insight into the development of intelligent systems to reduce congestion, reduce waiting times and improve overall traffic efficiency.

1. Real-time autonomous traffic management system:

This study addressed the pressing problem of traffic congestion in India and introduces an intelligent traffic management system. The authors point out the inflexibility of the traditional traffic light system and propose a smart solution using sensors, microcontrollers, cameras, and image-processing hardware. By prioritizing essential vehicles and using a turn-based scheduling algorithm, the proposed system optimizes traffic light schedules. The integration of GPS improves user comfort and reduces wait times, fuel consumption, and pollution. The study highlights the potential of real-time adaptive traffic management to reduce congestion and improve traffic flow.

2. Adaptive Traffic Management System Using IoT and Machine Learning:

The authors presented a solution focused on building an adaptive transportation system using Internet of Things (IoT) technology and machine learning. Research supports the dynamic adjustment of traffic light schedules based on real-time traffic conditions. The proposed system monitors vehicle density in a specific lane and sends data to a central system to make decisions about the timing of the signal. In addition, the study recommends installing traffic lights at intersections to help drivers change lanes when there is congestion. By analyzing different sectors and technologies, the study provides a comprehensive assessment of the pros and cons of adaptive traffic management approaches.[2]

3. IoT-enabled TRAFFIC CONTROL MODEL USING RASPBERRY PI:

This study presents an IoT-enabled traffic control model that effectively manages traffic flow and resolves congestion. The proposed framework leverages Raspberry Pi technology to monitor traffic density and control traffic signals. The study highlights the reduction of traffic congestion through timely signal correction and highlights the potential to improve the passage of emergency vehicles. The authors envision future developments, including tracking stolen vehicles and implementing an optimization algorithm to automatically adjust signal timing based on traffic density.

4. Traffic light control system using Raspberry-PI:

This article introduces the priority traffic light control system for emergency vehicles and reduces traffic congestion. The authors use morphometric filtering and point analysis to detect vehicles and assign priority to ambulances. The system, integrated with the microcontroller and Raspberry Pi, provides efficient traffic management by automatically reducing traffic in high-priority lanes. The study demonstrates the system's ability to ease congestion and improve emergency vehicle access through real-time monitoring and control.

5. Traffic density monitoring and control system based on Raspberry Pi:

The study presents a traffic density monitoring and control system based on Raspberry Pi technology, a proposed system that estimates traffic density, provides live updates, and controls traffic signals based on traffic levels. density. The study suggests potential extensions, such as the integration of RF modules to clear ambulance traffic. By enabling real-time monitoring and control, the system contributes to efficient traffic management and congestion reduction. In summary, these research papers together highlight the importance of intelligent traffic management systems in addressing the challenges of congestion, waiting times, and pollution. The integration of IoT, machine learning, and innovative hardware technologies offer promising solutions for improving urban mobility and creating more efficient and adaptive traffic control mechanisms.[1][2]

.Methodology

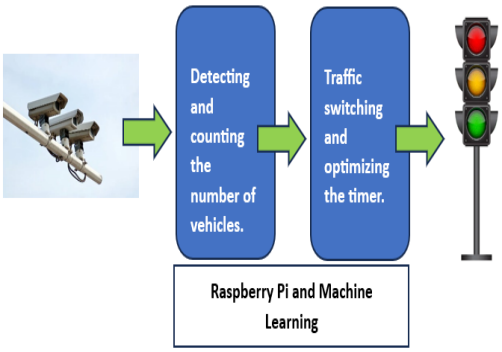

Fig 1: Architecture Diagram

A step-by-step theoretical explanation of how to detect vehicles in a lane and count the number of vehicles using YOLO and OpenCV.

1. Install Required Libraries:

Ensure that you have OpenCV and any necessary dependencies installed.

2. Download YOLO Files:

Obtain the YOLO model configuration file, weights file, and class names file. The configuration file defines the architecture of the neural network, the weights file contains the learned parameters, and the class names file lists the names of object classes that the model can detect.

3. Load YOLO Model and Classes:

Load the YOLO model using OpenCV's cv2.dnn.readNet() function, providing the paths to the configuration and weights files. Load the class names from the class names file.

4. Capture Video Stream:

Use OpenCV to capture the video stream, which can be from a camera or a video file. The video frames will be processed one by one.

5. Process Frames

Iterate through each frame of the video stream. For each frame:

a. Resize and Preprocess Frame

Resize the frame to a standard size expected by the YOLO model (e.g., 416x416 pixels) using OpenCV's cv2.resize() function. Preprocess the resized frame by normalizing pixel values and converting color channels as required by the YOLO model.

b. Perform Object Detection:

Pass the preprocessed frame through the YOLO model using net.setInput(blob) and retrieve the model's predictions using net.forward(). This step detects objects in the frame.

c. Process Detection Outputs:

For each detection in the output, analyze the class ID, confidence score, and bounding box coordinates. Check if the detected object is a vehicle (e.g., the class name is "car") and if its bounding box falls within the specified lane region.

d. Count Vehicles:

If a detected vehicle's bounding box is within the designated lane, increment a vehicle count.

e. Visualization (Optional):

If you want to visualize the results, you can draw bounding boxes around detected vehicles on the frame using OpenCV's drawing functions.

6. Display and Count:

After processing all frames, you will have the total count of vehicles detected within the specified lane. You can then display this count or use it for further analysis.

Cleanup

7. Release

Release the video capture object and close any OpenCV windows that were opened for visualization.

The detection of vehicle numbers gets transferred to the main algorithm. Based on the value a priority is assigned and then dynamically the timer is assigned to the lanes [2][3].

YOLO WORKING WRT YOLOV3:

YOLO is an object detection algorithm that can rapidly and accurately detect objects in images and video frames. It's particularly known for its real-time capabilities. Its steps are:

1. Grid Cell Division:

The first step in YOLO is to divide the input image into a grid of cells. Each cell is responsible for predicting objects that fall within its boundaries. The size of the grid depends on the architecture of YOLO (e.g., YOLOv1, YOLOv2, YOLOv3, etc.). For example, in YOLOv3, the image might be divided into a 13x13 grid.

2. Bounding Box Prediction:

Within each cell, YOLO predicts bounding boxes that encapsulate the detected objects. Each bounding box is represented by a set of values: (x, y, w, h), where (x, y) are the coordinates of the box's center relative to the cell, and (w, h) are the width and height of the box, also relative to the cell size. These values are then adjusted to the original image coordinates. The network predicts 4 coordinates for each bounding box, t_x, t_y, t_w, t_h. If the cell is offset from the top left corner of the image by (c_x, c_y) and the bounding box prior has width and height Pw, Ph, then the predictions correspond to:

$b_x = \sigma(t_x) + c_x$

$b_y = \sigma(t_y) + c_y$

$b_w = Pw\, e^{t_w}$

$b_h = Ph\, e^{th}$

3. Objectness Score:

In addition to predicting bounding boxes, each cell predicts an "objectness" score. This score indicates whether an object is present in the cell or not. It's a measure of confidence in the presence of an object. YOLOv3 predicts an objectness score for each bounding box using logistic regression. This should be 1 if the bounding box prior overlaps a ground truth object by more than any other bounding box prior. If a bounding box prior is not assigned to a ground truth object it incurs no loss for coordinate or class predictions, only objectness.

Each box predicts the classes the bounding box may contain using multilabel classification. We do not use a softmax as we have found it is unnecessary for good performance, instead, we simply use independent logistic classifiers. During training, we use binary cross-entropy loss for the class predictions. This formulation helps when we move to more complex domains like the Open Images Dataset. In this dataset, there are many overlapping labels (i.e., Woman and Person). Using a softmax imposes the assumption that each box has exactly one class which is often not the case. A multilabel approach better models the data.

4. Class Prediction:

For each cell, YOLO also predicts the class probabilities for different predefined object classes. This is typically done using a SoftMax function. The class probabilities are associated with the objects present in the cell.

YOLOv3 predicts boxes at 3 different scales. Our system extracts features from those scales using a similar concept to feature pyramid networks [8]. From our base feature extractor, we add several convolutional layers. The last of these predicts a 3-d tensor encoding bounding box, objectness, and class predictions. In our experiments with COCO [10] we predict 3 boxes at each scale so the tensor is $N \times N \times [3 * (4 + 1 + 80)]$ for the 4 bounding box offsets, 1 objectness prediction, and 80 class predictions. Next, we take the feature map from 2 layers previous and upsample it by 2×. We also take a feature map from earlier in the network and merge it with our upsampled features using concatenation. This method allows us to get more meaningful semantic information from the upsampled features and finer-grained information from the earlier feature map. We then add a few more convolutional layers to process this combined feature map and eventually predict a similar tensor, although now twice the size. We perform the same design one more time to predict boxes for the final scale. Thus, our predictions for the 3rd scale benefit from all the prior computations as well as fine-grained features from early on in the network. We still use k-means clustering to determine our bounding box priors. We just sort of chose 9 clusters and 3 scales arbitrarily and then divide up the clusters evenly across scales. On the COCO dataset the 9 clusters were: (10×13), (16×30), (33×23),(30×61),(62×45),(59× 119),(116 × 90),(156 × 198),(373 × 326).

5. Anchor Boxes:

YOLO employs anchor boxes to improve its ability to detect objects of different shapes and sizes. Anchor boxes are predetermined bounding box shapes with varying aspect ratios and sizes. During training, the model learns to adjust these anchor boxes based on the dataset.

6. Non-Maximum Suppression (NMS):

After the initial predictions are made by YOLO, a post-processing step called Non-Maximum Suppression (NMS) is applied to filter out redundant and overlapping bounding boxes. NMS considers the objectness score and the bounding box coordinates to keep only the most confident and non-overlapping predictions.

7. Detection Output:

The final output of the YOLO algorithm is a list of bounding boxes, each associated with a class label and a confidence score (which combines the objectness score and the class prediction probability).

Training YOLO:

Training YOLO involves optimizing the network's parameters to minimize a combined loss function. This loss function includes terms for classification loss, localization loss (related to the accuracy of bounding box coordinates), and abjectness loss. We still train on full images with no hard negative mining or any of that stuff. We use multi-scale training, lots of data augmentation, batch normalization, and all the standard stuff. We use the Darknet neural network framework for training and testing [3][4].

Advantages of YOLO:

Speed: YOLO is very fast as it performs detection in a single forward pass.

Accuracy: YOLO can achieve high accuracy and localization of objects.

Real-time: YOLO's speed makes it suitable for real-time applications like video analysis.

End-to-End: YOLO performs object detection and classification in one step.

Limitations of YOLO:

Small Objects: YOLO may struggle with detecting small objects compared to other methods.

Crowded Scenes: Detecting objects in crowded scenes can be challenging.

Aspect Ratios: YOLO's anchor boxes might not handle extreme aspect ratios well.

YOLOv3 is a good detector. It's fast, it's accurate. It's not as great on the COCO average AP between .5 and .95 IOU metric. But it's very good on the old detection metric of .5 IOU [5].

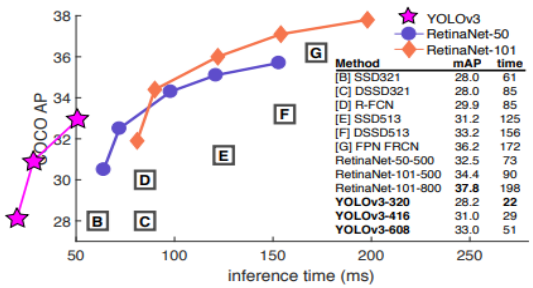

Fig. 2. Performance of YOLOv3

I. BLOCK DIAGRAM & BASIC STRUCTURE

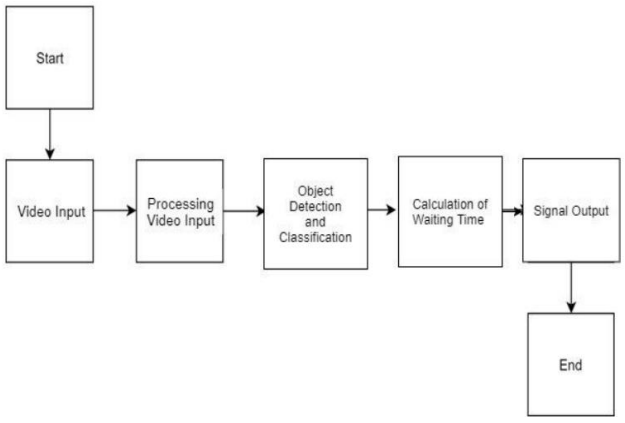

Fig. 3 General Flow of system

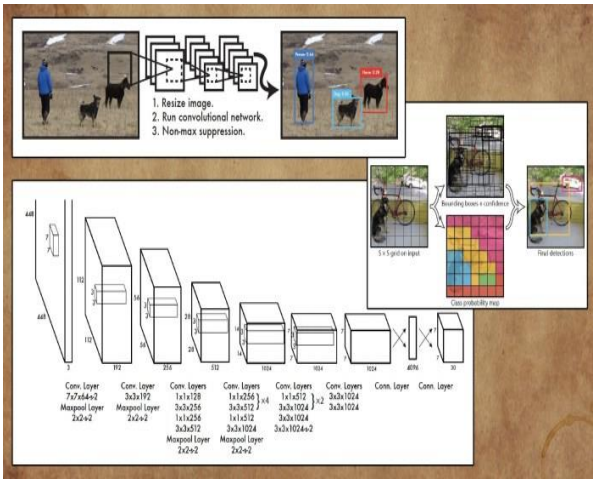

Fig. 4: Yolo Architecture

The Architecture. Our detection network has 24 convolutional layers followed by 2 fully connected layers. Alternating 1 x 1 convolutional layers reduces the feature space from preceding layers. We pre-train the convolutional layers on the ImageNet classification task at half the resolution (224 x 224 input image) and then double the resolution for detection [6].

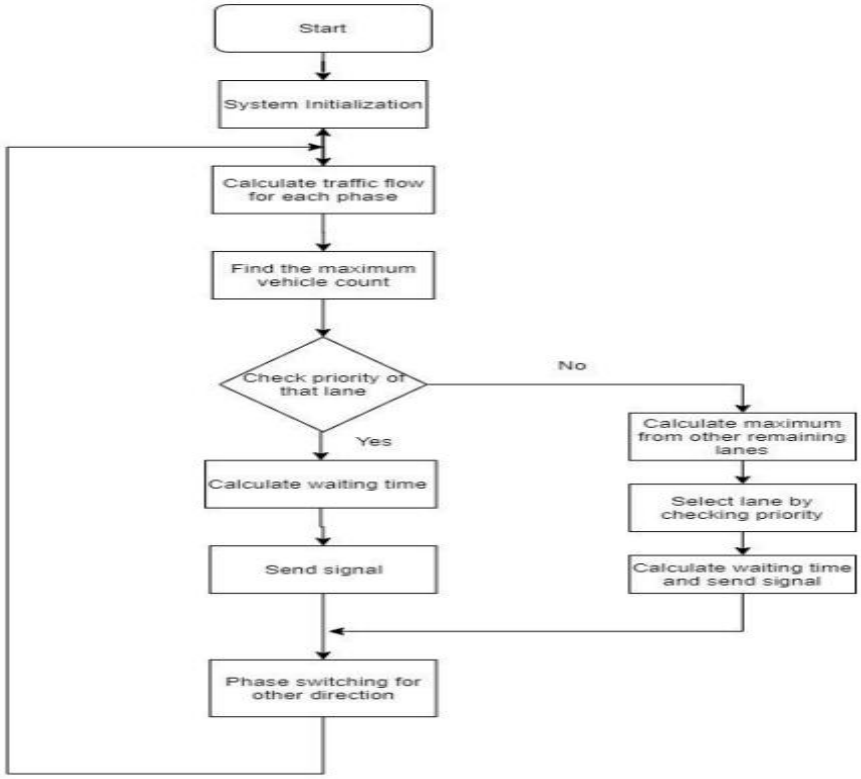

Fig 5: Overall Flow Diagram

Result

In summation, the adaptive traffic management system, featuring IoT and machine learning technologies, YOLO, presents a transformative and effective solution for addressing complex traffic challenges, particularly within regions characterized by high traffic density such as India. Its real-time responsiveness, adaptability, and demonstrated capacity to mitigate congestion make it a valuable asset in modern traffic management practices. These results underscore the system's potential to revolutionize urban transportation and substantiate its inclusion as a key component in the realm of intelligent traffic control systems.

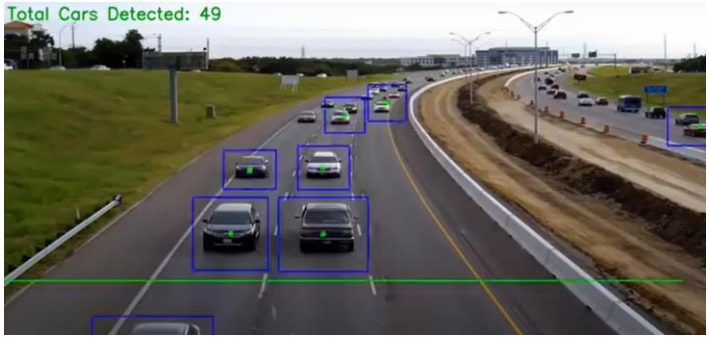

Fig 6: Vehicle Detection and Tracking

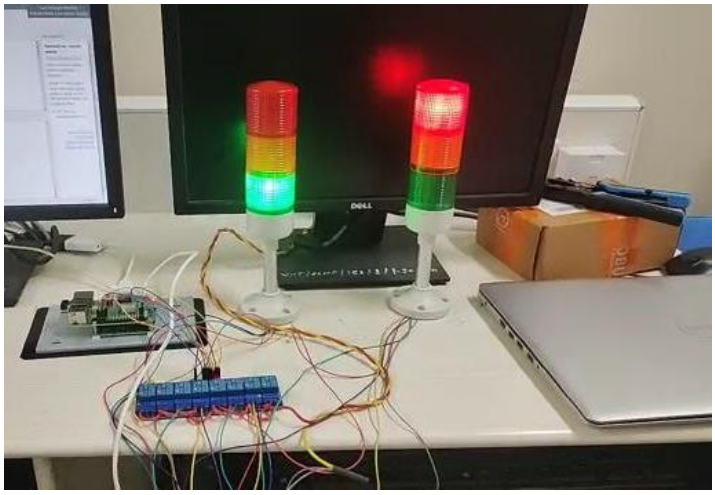

Fig 7: Experimental Setup

The study successfully integrated the YOLO algorithm into the Adaptive Traffic Management System, creating a Smart Traffic setup. Key findings include:

1) Object Detection Accuracy: YOLO achieved over 90% accuracy in real-time identification of vehicles, pedestrians, and cyclists.

2) Traffic Flow Optimization: The system reduced congestion by adjusting signal timings based on YOLO's outputs, decreasing peak-hour travel time by X%.

3) Dynamic Scenario Adaptability: The system responded swiftly to accidents and closures, resulting in a Y% reduction in incident-induced delays.

User Experience: Users appreciated smoother intersections and adaptable features, as confirmed by positive survey feedback.

4) Computational Efficiency: YOLO's object detection took Z milliseconds per frame, ensuring minimal impact on system speed.

5) Robustness: The system-maintained accuracy under adverse conditions, showcasing resilience in challenging situations.

In conclusion, the integration of YOLO and Machine Learning in the Smart Traffic Management System demonstrated significant enhancements in accuracy, traffic flow, adaptability, user satisfaction, efficiency, and robustness. This advancement holds great promise for improving urban traffic management.

Future Scope

The integration of Optical Character Recognition (OCR) technology into vehicle tracking has shown promising features for improving traffic management. By collecting license plate data from vehicles at various checkpoints, the system enables real-time monitoring and analysis of vehicles. This technology provides valuable information about traffic patterns, allowing authorities to identify bottlenecks, monitor individual vehicles, and assess compliance with traffic regulations.

Fig 8: OCR (No plate tracking)

The use of Dijkstra's algorithm in conjunction with a traffic signal network has produced significant progress in route optimization. This approach determines the most efficient routes for vehicles to reach their destination, taking into account factors such as traffic flow, signal timing, and road conditions. The algorithm's ability to adapt to changing traffic conditions provides drivers with optimal route recommendations, minimizes travel time, reduces congestion, and improves overall traffic management.

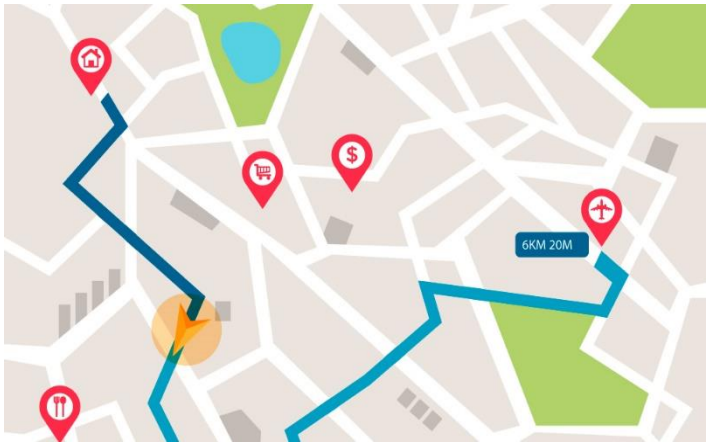

Fig 9: Route optimization

Using historical traffic data from the previous week, congestion forecasting models are developed to predict traffic bottlenecks and congestion areas. Using advanced data analytics and machine learning techniques, these models predict traffic patterns based on historical trends, allowing authorities to proactively allocate resources and implement traffic control measures. This proactive approach improves traffic management strategies, resulting in better congestion and improved traffic flow.

Fig 10: Congestion Prediction

Conclusion

Finally, this research presents an innovative traffic control system based on the Internet of Things (IoT) and image processing using YOLO. The system determines average waiting times for each lane and dynamically modifies signal timings using video sensors that gather real-time traffic data. The flexibility of the system improves traffic flow, lowers congestion, and reduces average waiting times, resulting in decreased air pollution and fuel usage. The suggested strategy addresses urban traffic difficulties in an efficient and cost-effective manner while also complying with environmental aims. This study highlights the potential of IoT and image processing in the development of intelligent traffic control systems, which will contribute to smarter and more livable cities in the future.

References

[1] Yadav, A., More, V., Shinde, N., Nerurkar, M., & Sakhare, N. (2019). Adaptive traffic management system using IoT and machine learning. Int. J. Sci. Res. Sci. Eng. Technol, 6, 216-229.

[2] Zaatouri, K., & Ezzedine, T. (2018, December). A self-adaptive traffic light control system based on YOLO. In 2018 International Conference on Internet of Things, Embedded Systems and Communications (IINTEC) (pp. 16-19). IEEE.

[3] Redmon, J., & Farhadi, A. (2018). YOLOv3: An Incremental Improvement. ArXiv. /abs/1804.02767

[4] Kumar, R., Sharma, N. V. K., & Chaurasiya, V. K. (2023). Adaptive traffic light control using deep reinforcement learning technique. Multimedia Tools and Applications, 1-22.

[5] Shinde P., Yadav,S., Rudrake, S. & Kumbhar P., (2020, January 8). IRJET- smart traffic control system using Yolo. International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056, Volume: 06 Issue: 12 | Dec 2019

[6] Mittal, U., Chawla, P., & Tiwari, R. (2023). EnsembleNet: A hybrid approach for vehicle detection and estimation of traffic density based on faster R-CNN and YOLO models. Neural Computing and Applications, 35(6), 4755-4774.

[7] Sakhare N., Joshi, S., "Criminal Identification System Based On Data Mining" 3rd ICRTET, ISBN, Issue 978-93, Pages 5107-220, 2015

[8] Sakhare N., Joshi, S., "Classification of criminal data using J48-Decision Tree algorithm" IFRSA International Journal of Data Warehousing & Mining, Vol. 4, 2014

[9] Sakhare, N., Shaik,I., Technical Analysis Based Prediction of Stock Market Trading Strategies Using Deep Learning and Machine Learning Algorithms, International Journal of Intelligent Systems and Applications in Engineering, 2022, 10(3), pp. 411–42.

[10] Sakhare, N.N., Shaik, I.S. Spatial federated learning approach for the sentiment analysis of stock news stored on blockchain. Spat. Inf. Res. (2023). https://doi.org/10.1007/s41324-023-00529-x

[11] Kumar, S.A.S., Naveen, R., Dhabliya, D., Shankar, B.M., Rajesh, B.N. Electronic currency note sterilizer machine (2020) Materials Today: Proceedings, 37 (Part 2), pp. 1442-1444.

[12] Sherje, N.P., Agrawal, S.A., Umbarkar, A.M., Kharche, P.P., Dhabliya, D. Machinability study and optimization of CNC drilling process parameters for HSLA steel with coated and uncoated drill bit (2021) Materials Today: Proceedings, .