Design Thinking Approach for SMART TRAFFIC CONTROL SYSTEM USING AI

R Bhuvaneswari
Department of Computer Science and Engineering,
Rajalakshmi Engineering College,
Chennai, India
bhuvaneswari.r@rajalakshmi.edu.in

A. Sangamithran
Department of Computer Science and Engineering
Rajalakshmi Engineering College,
Chennai, India
230701283@rajalakshmi.edu.in

D. Sanjay Kishan
Department of Computer Science and Engineering
Rajalakshmi Engineering College,
Chennai, India
230701287@rajalakshmi.edu.in

Abstract **-** Traffic congestion remains a critical urban challenge, often causing delays for emergency services, escorted vehicles, and high-priority transportation. Studies reveal that such congestion results in billions of dollars in economic losses and an average of 54 hours of lost productivity per commuter annually. To address this, we propose an AI-powered traffic management system that leverages computer vision and deep learning—specifically Convolutional Neural Networks (CNNs)—for real-time vehicle classification and dynamic signal control at intersections.

Our approach integrates two core components: real-time video-based data acquisition from traffic cameras and a CNN-based classification model trained different classification of vehicles using MobileNetV2 architecture. The system performs object detection and classification to assess vehicle density and dynamically adjusts traffic signal timings based on computed passage requirements. Furthermore, continuous model refinement through ongoing data collection supports the development of a predictive traffic flow framework for anticipatory control.

The trained CNN model demonstrates its utility through accuracy evaluations, confusion matrix analysis, and classification reports, achieving a classification accuracy of up to 87% on a custom dataset. The model's efficacy is validated both through simulation and prototype deployment, affirming the feasibility and potential of AI-based traffic control systems to enhance urban mobility.

I. INTRODUCTION

Traffic congestion is a persistent and critical issue in cities across the globe, irrespective of their geographic scale or economic development. Intersections are particularly vulnerable zones where inefficiencies in traffic signal timing result in long vehicle queues, unnecessary fuel consumption, increased noise from idling engines, and elevated levels of air pollution. These complications are further intensified during emergencies or high-priority transit scenarios where delays can have serious consequences. Therefore, the need for an intelligent, data-driven traffic management system has become more urgent than ever.

Traditional traffic control systems follow pre-set timing cycles, which do not adapt in real-time to the dynamic traffic conditions. This creates bottlenecks and delays that affect both urban mobility and environmental sustainability. To address this, modern traffic systems must integrate real-time data processing, adaptive control mechanisms, and automated vehicle detection using advanced technologies.

In this study, we propose a Convolutional Neural Network (CNN)-based traffic classification model to optimize signal timing at intersections using visual inputs from traffic cameras. The model is trained on different types of ongoing vehicles. By deploying a lightweight but powerful MobileNetV2 architecture, we ensure that the system is both accurate and efficient in real-time deployments.

II. LITERATURE REVIEW

Traffic management and vehicle classification have been widely researched topics due to their significant impact on urban mobility and safety. Traditional machine learning and deep learning approaches have been extensively explored to develop robust traffic analysis systems.

Several classification algorithms have been tested for vehicle detection and categorization. Convolutional Neural Networks (CNNs), due to their superior ability to automatically extract hierarchical image features, have consistently outperformed classical machine learning models. For example, a CNN-based model achieved an accuracy of **87.78%**, demonstrating its effectiveness in complex vehicle image classification tasks.

Other popular machine learning models have also been evaluated for vehicle classification. The k-Nearest Neighbors (KNN) algorithm, known for its simplicity and effectiveness in small datasets, achieved an accuracy of **76%**. However, KNN's performance typically degrades with larger and more varied image datasets due to its reliance on distance metrics in high-dimensional spaces.

The ResNet50 model, a deeper convolutional neural network architecture designed to address vanishing gradient problems through residual connections, achieved a classification accuracy of **72.68%** in similar contexts. Although powerful, ResNet50's heavier computational requirements may limit its

real-time applicability in resource-constrained environments

S.NO	ALOGORITHM COMPARISON		
	ALGORITHM	ACCURACY	IMPLEMENTATION
1.	KNN	76%	NO
2.	CNN	87.78%	YES
3.	RESNET50	72.68%	NO
4.	SVM	80.22%	NO
5.	RANDOM FOREST	78%	NO

such as embedded traffic cameras.

Support Vector Machines (SVM), widely regarded for their effectiveness in high-dimensional classification tasks, yielded an accuracy of **80.22%**. SVMs perform well when the data is well-separated, but require significant feature engineering when applied to raw image data, which can limit scalability.

Decision Tree and Random Forest models, known for their interpretability and robustness against overfitting, reported accuracies around **78%**. While these ensemble learning methods can handle heterogeneous data, they depend heavily on the quality of manually extracted features and are generally less suited for image classification compared to deep learning models.

In summary, the literature indicates that while classical machine learning algorithms like KNN, SVM, and Random Forest provide reasonable accuracy, CNN-based models—especially lightweight architectures such as MobileNetV2—offer superior performance and scalability for vehicle classification in real-time traffic management systems.

III. PROPOSED SYSTEM

1. Front-End Development

The front-end interface of the system was developed using react.js and Tailwind CSS. This web-based dashboard enables users—such as traffic control authorities—to upload live video feeds or CCTV footage from intersections. It provides real-time visualization of detected vehicles. Additionally, it dynamically updates the calculated green-light durations based on real-time vehicle densities. The interface is designed to be lightweight, intuitive, and responsive for fast decision-making in a traffic control environment.

2. Back-End Development

The back-end was developed using Python with the Flask web framework. This module acts as the communication bridge between the front-end and the deep learning model. It receives video frames via HTTP requests or real-time video streams, processes these frames, and forwards them to the deep learning module for vehicle detection and classification. Once the vehicle counts are returned, the back-end computes optimal signal durations and transmits the result back to the front-end. For real-world deployment, it can also interface with traffic signal controllers using MQTT or REST APIs.

3. Deep Learning Module

The deep learning module is the core of the system, responsible for detecting and classifying vehicles using a CNN model based on the MobileNetV2 architecture. Several machine learning and deep learning models were evaluated on the filtered vehicle dataset for performance comparison:

Table 1.Comparison of algorithms

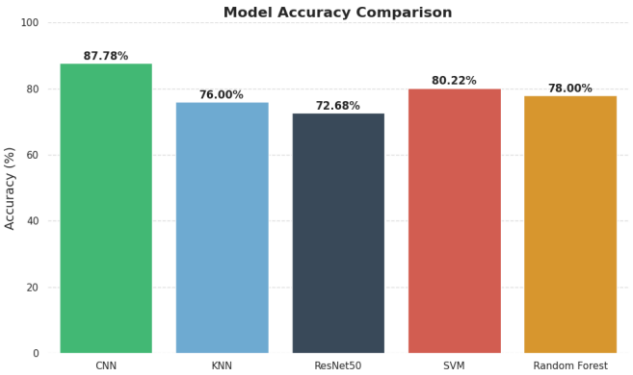

Fig 1. Comparison bar graph

The MobileNetV2-based CNN was selected for its balance of speed and accuracy. The architecture includes a global average pooling layer, dropout (to prevent overfitting), a fully connected dense layer with 128 units, and a SoftMax output layer for five-class classification. The model performs real-time classification, enabling efficient signal adjustment at intersections.

4. Database Design

A PostgreSQL-based database was integrated into the system to store structured traffic data. It logs vehicle counts per frame, signal timings per phase, and historical performance data for analysis. This data aids in future model retraining and the generation of traffic reports. An ORM layer using SQL Alchemy ensures seamless access to database records from the back-end service.

5. Dataset Preprocessing

The dataset used for model training was obtained from the Kaggle "Vehicles Image Dataset" and filtered to include only five relevant classes: bike, bus, car, truck, and ambulance. All images were resized to 128 × 128 pixels, normalized, and labelled accordingly. Data augmentation techniques such as random rotation, flipping, zooming, and shifting were applied to artificially expand the dataset and reduce overfitting. The processed dataset was split into an 80:20 ratio for training and validation, ensuring class balance across both subsets.

IV. IMPLEMENTATION

The implementation stage is the core of this project, where the AI-powered traffic management system is translated from concept to an operational platform. The system utilizes a

trained Convolutional Neural Network (CNN) model based on MobileNetV2 to classify five vehicle types—bike, bus, car, truck, and ambulance—and dynamically adjust signal timings accordingly. Real-world traffic videos are processed to test and demonstrate the model's effectiveness in live urban scenarios.

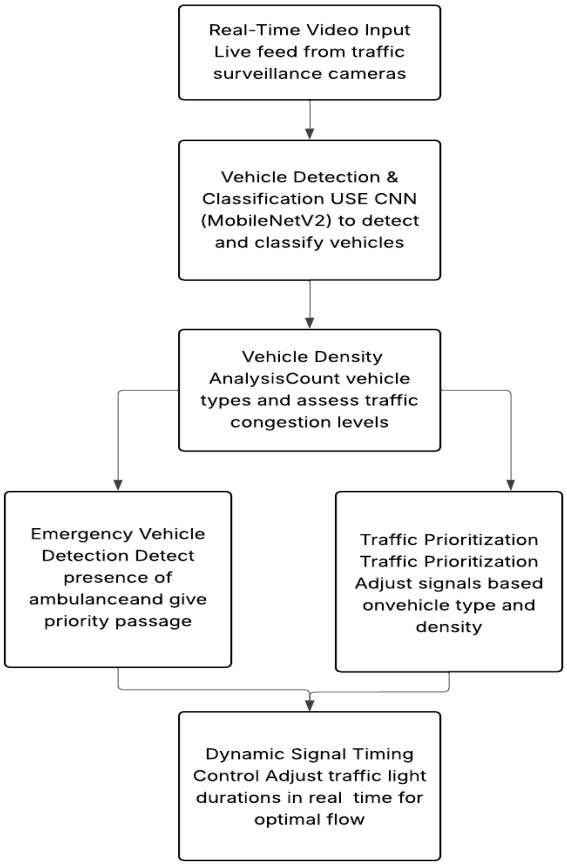

Fig.1 Block diagram of proposed system

1. *Smart Traffic Management Interface*

The main interface of the system is a web-based dashboard designed to be intuitive and functional. As shown in Fig. 2, users are presented with a titled interface, "Smart Traffic Management System," which enables real-time traffic monitoring and dynamic signal allocation. This dashboard acts as the control centre from which traffic officers can upload or stream videos from intersection cameras.

Fig 2. Dashboard Interface

2. *Upload Traffic Video Page*

The video upload page is central to the system's input pipeline. User scan upload videos for all four directions of an intersection—North, East, South, and West—as depicted in Fig. 3. The UI supports drag-and-drop functionality or click-to-upload, making the process user-friendly. Once videos are uploaded, they are automatically queued for processing by the deep learning module.

Fig 3. video upload page

3. *Process and Analysis Page*

After uploading the videos, users can initiate analysis by clicking the "Process Videos" button. This triggers frame-wise vehicle detection using the trained CNN model. Detected vehicles are categorized and counted per direction. The system then allocates signal timing based on traffic density, giving priority to emergency vehicles like ambulances if detected. The results are visualized using graphical traffic lights and percentage bars for each direction, as shown in Fig. 4. This real-time feedback enhances both monitoring and decision-making.

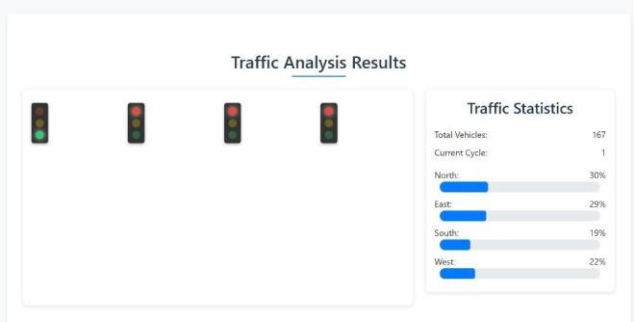

Fig 4. Traffic Analysis page

V. TOOLS AND TECHNOLOGIES USED

Component	Tool/Technology
Object Detection	Model detection Using CNN
Programming Language	Python
Real-Time Video Processing	OpenCV
Simulated Traffic Feed	Pre-recorded or synthetic traffic videos
Traffic Signal Logic	Custom Python logic + state machine
Model Training	Trained using Google collab CNN

INPUT

- A live or recorded video feed of a traffic intersection
- Vehicles including cars, buses, bikes, and emergency vehicles like ambulances and fire trucks

Processing Pipeline:

1. My Modal detects and classifies all vehicles in each frame
2. A Python script calculates the total number of vehicles in each lane
3. If an emergency vehicle is detected:
    a) The signal in its lane is prioritized (turned green)
    b) Other lanes receive red signal
4. If no emergency vehicle is present:
    a) Signals are controlled dynamically based on vehicle count density per lane
    b) A timer system simulates realistic signal transitions

OUTPUT:

a) A simulation of signal behaviour (visualized using OpenCV overlays or a simple GUI)
b) Emergency vehicles pass through intersections quickly

Normal traffic flows efficiently based on real-time data

VI. TESTING AND EVALUATION

The testing phase played a crucial role in evaluating the functionality, responsiveness, and accuracy of the Smart Traffic Management System under various real-world and simulated conditions. The goal was to validate the CNN-based vehicle classification model and assess the dynamic signal timing logic in different traffic scenarios, particularly in terms of emergency vehicle prioritization and user experience.

To assess the system's performance, several test scenarios were simulated:

- **Daytime traffic** with varied vehicle density and normal visibility conditions.
- **Night-time traffic** with low visibility and potential headlight glare.
- **Emergency vehicle entry** in congested lanes to test priority response.
- **Multi-lane intersections** with unequal traffic flow in different directions.
- **User testing of the interface**, focusing on ease of video upload, result interpretation, and feedback clarity.

These simulations revealed several practical insights. Even with a well-trained MobileNetV2 CNN model, real-world testing uncovered edge cases such as reduced accuracy in low-light frames or frames with motion blur. Feedback from users, including transportation engineers and field testers, emphasized the importance of clear visual feedback, signal timing transparency, and real-time responsiveness. Early user involvement in the testing phase proved valuable in identifying usability gaps and aligning the system closer to deployment standards.

VII. RE-DESIGN AND REFINEMENT

Based on the observations and feedback collected during the testing phase, the system underwent several rounds of re-design, both in terms of its user interface (UI/UX) and back-end algorithmic components. While the initial prototype demonstrated the essential features—vehicle detection using a CNN model, real-time signal adaptation, and basic emergency response logic—it lacked certain refinements required for practical deployment.

The improvements implemented as part of the re-design phase are summarized below:

Area	Issue Identified	Improvement Implemented
Detection Accuracy	Reduced classification accuracy in low-light or rainy conditions	Retrained MobileNetV2 model using augmented data including night-time, fog, and rain scenarios
Signal Switching Logic	Abrupt signal changes confused users	Introduced a 2-second yellow blinking buffer before each signal change
Emergency Vehicle Handling	Missed or delayed identification of ambulances	Integrated confidence threshold tuning and pattern recognition (e.g., flashing lights or vehicle Color)
User Interface	No visual indicator for signal status and vehicle load	Enhanced GUI with live traffic light visuals, vehicle count overlays, and emergency alerts
Traffic Insights Dashboard	No centralized monitoring of traffic trends	Added a lightweight analytics panel showing historical signal timings and direction-wise traffic density

The re-implementation focused on optimizing the model's performance while ensuring the system remains lightweight and scalable. Back-end improvements included modular restructuring of the CNN pipeline for faster inference and robustness. The user interface was redesigned for clarity, allowing seamless user interaction through upload prompts, status indicators, and real-time visualizations.

Overall, the iterative design, guided by structured testing and real-time feedback, significantly improved the system's reliability, user experience, and readiness for potential field deployment.

VIII. CONCLUSION

In this project, we explored the application of deep learning techniques—specifically Convolutional Neural Networks (CNN)—to address one of the most pressing urban challenges: traffic congestion and inefficient signal management. With increasing vehicle density and the urgent need to prioritize emergency response routes, traditional fixed-cycle traffic systems fall short in dynamic traffic environments. To overcome these limitations, we proposed and implemented a smart, AI-based traffic management system capable of detecting and classifying vehicles from live traffic footage and dynamically adjusting signal timings in real time.

The implementation followed the Design Thinking process, which allowed us to ground the system in real-world needs. During the **Empathize** and **Define** stages, we identified key challenges faced by commuters and traffic controllers, such as delayed emergency services and traffic signal inefficiencies. The **Ideation** phase involved evaluating multiple approaches, including traditional machine learning methods and deep learning models. Through comparative analysis, we selected CNN, and more specifically the MobileNetV2 architecture, for its balance of accuracy and efficiency—achieving a vehicle classification accuracy of **87.78%**.

The CNN model was trained on a curated dataset filtered from the Kaggle "Vehicles Image Dataset," focusing on five vehicle classes: bike, bus, car, truck, and ambulance. Python was used for implementation, along with TensorFlow and supporting libraries like NumPy and Matplotlib for model development and visualization. The model was deployed in a modular web platform that allows users to upload traffic videos, process directional inputs, and visualize signal status and vehicle statistics dynamically.

Our system not only demonstrates the capability of CNNs in real-time traffic analysis but also offers a scalable, practical solution for smart cities aiming to reduce congestion and enhance traffic flow. The visual dashboard, automated signal timing, and integration of AI ensure adaptability and user-friendliness.

In conclusion, this project showcases how artificial intelligence can transform traditional traffic management into a proactive, responsive, and intelligent system. In future iterations, integrating the system with IoT-based sensor networks, GPS data, and real-time emergency alerts could further enhance its responsiveness and utility in complex urban environments. This approach has strong potential to contribute toward safer, faster, and smarter transportation systems.

IX. REFERENCES

[1] Maura Mazzarello A traffic management system for real-time traffic optimisation

[2] Tomas Rodriguez 27 January 2009. An adaptive, real-time, traffic monitoring system

[3] Mohammed Sarrab Development of an IoT based real-time traffic monitoring system for city governance

[4] Neeraj kumar A Review on Traffic Monitoring System Techniques

[5] B.S Meghna, Santoshi kumari, and TP pushphavathi Comprehensive traffic management system: Real-time traffic data analysis using RFID

[6] Majid Ayoubi 2024 AI-Enabled Traffic Light Control System: An Efficient Model to Manage the Traffic at Intersections using Computer Vision

[7] DungLe Dona, Athanasios Ziliaskopoulos, and Hani Mahamassani On-Line Monitoring System for Real-Time Traffic Management Applications

[8] Diego Sandino, Luis M. Matey, and Gorka Vélez Design Thinking Methodology for the Design of Interactive Real-Time Applications

[8] Michael Osigbemeh, Michael Onuu, Olumuyiwa Asaolu, 2016.Design and development of an improved traffic light control system using hybrid lighting system

[9] Mamata Rath 2018 Smart Traffic Management System for Traffic Control using Automated Mechanical and Electronic Devices

[10] Moolchand Sharma, 2021.Intelligent Traffic Light Control System Based on Traffic Environment Using Deep Learning

[11] Bilal Ghazal, Khaled Chahine, 2016. Smart traffic light control system

[12] NAOKI KODAMA, TAKU HARADA AND KAZUTERU MIYAZAKI,2016. Traffic Signal Control System Using Deep Reinforcement Learning with Emphasis on Reinforcing Successful Experiences