

Traveller Safety Web Application using License Plate Recognition, GPS Tracking, Route Deviation Detection, and Emergency Alert System

Dilshith P

Department of Computer Applications

Nehru Institute of Information Technology and Management, Coimbatore 641105, Tamil Nadu, India

Anna University, Chennai, India

Email: dilshith890@gmail.com

Dr. S. Menaka, MCA, M.Phil., Ph.D.

Head of the Department, Department of Computer Applications

Nehru Institute of Information Technology and Management, Coimbatore 641105, Tamil Nadu, India

Anna University, Chennai, India

Abstract

Ensuring traveller safety in public transportation remains a critical challenge, particularly in densely populated urban environments where vehicular crime, route deviations, and inadequate emergency response mechanisms contribute significantly to passenger risk. This paper presents a comprehensive Traveller Safety Web Application that integrates multiple intelligent subsystems — License Plate Recognition (LPR) using Detectron2 and EasyOCR, real-time GPS tracking, A*-based optimal route computation, Hidden Markov Model (HMM)-driven route deviation detection, and an automated Emergency Short Message Service (SMS) Alert System — into a unified, scalable web-based platform.

The proposed architecture leverages a Python-Flask backend integrated with a PostgreSQL relational database and a React.js front-end interface. Vehicle identification is performed through a two-stage pipeline wherein Detectron2 detects license plate regions with a mean Average Precision (mAP) of 94.7%, followed by EasyOCR character recognition achieving an accuracy of 96.2% under varied illumination and occlusion conditions. The GPS subsystem employs the Haversine formula for geodesic distance computation and maintains a 3-second polling interval to provide near real-time location updates. Route deviation detection is implemented using a six-state HMM that models legitimate trip trajectories, triggering alerts when the Viterbi-decoded path probability falls below a defined threshold of 0.15. Emergency alerts are dispatched via the Twilio SMS API with an average latency of 1.8 seconds.

Experimental evaluations conducted on a dataset of 1,200 annotated vehicle images and 340 simulated GPS trip trajectories demonstrate that the integrated system achieves an overall safety event detection accuracy of 93.4%, with a false positive rate of 4.1% and a system response time of 2.3 seconds. The results confirm the viability of the proposed framework as a deployable public safety solution for urban transportation networks. Future work will explore federated learning-based plate recognition and edge-deployed IoT integration for offline functionality.

Keywords: *License Plate Recognition, Detectron2, EasyOCR, GPS Tracking, Route Deviation Detection, Hidden Markov Model, A* Algorithm, Emergency Alert System, Traveller Safety, Intelligent Transportation Systems*

1. Introduction

The rapid urbanisation of developing nations has placed unprecedented strain on public transportation infrastructure, creating conditions in which passenger safety is increasingly difficult to guarantee. According to the National Crime Records Bureau (NCRB) of India, crimes against women in public conveyances increased by 23.7% between 2019 and 2023, with a significant proportion of incidents occurring in taxis, shared cabs, and intercity buses [1]. Similar trends have been documented globally, prompting regulatory bodies to mandate the implementation of vehicle tracking and passenger safety mechanisms in fleet management systems.

The proliferation of smartphones, inexpensive GPS modules, and cloud computing infrastructure has created an enabling technological environment for the development of real-time safety systems that can monitor, detect, and respond to potential threats without requiring dedicated hardware installations in every vehicle. However, existing solutions remain fragmented — commercial ride-hailing platforms such as Ola and Uber incorporate basic GPS sharing features, but lack intelligent anomaly detection capabilities that can differentiate between deliberate route deviation and legitimate traffic diversions [2].

Artificial intelligence and computer vision have matured significantly over the past five years, with object detection frameworks such as Detectron2 [3] and optical character recognition libraries such as EasyOCR [4] offering near-human accuracy in controlled and semi-controlled environments. Simultaneously, graph-based pathfinding algorithms such as A* [5] and probabilistic sequence models such as Hidden Markov Models [6] have demonstrated strong applicability in trajectory analysis and anomaly detection tasks. The convergence of these technologies presents an unprecedented opportunity to design a cohesive, intelligent traveller safety system.

This paper contributes the following to the body of knowledge in intelligent transportation systems and public safety computing:

- A novel multi-modal safety architecture that integrates LPR, GPS tracking, route optimisation, and probabilistic anomaly detection into a single deployable web application.
- An adapted Detectron2-EasyOCR pipeline specifically tuned for Indian license plate morphology, including multi-line and high-speed blur scenarios.
- A six-state Hidden Markov Model formulation for route deviation detection that accounts for natural route variability due to traffic conditions.
- A formal mathematical framework governing the probabilistic safety scoring of trip trajectories.
- Rigorous experimental evaluation over a comprehensive dataset with comparative analysis against existing commercial systems.

1.1 Research Objectives

1. To design and implement an integrated web-based traveller safety system that combines License Plate Recognition, real-time GPS tracking, A*-based route optimisation, and HMM-based deviation detection into a unified platform with sub-3-second emergency response capability.
2. To evaluate the accuracy, precision, recall, and F1-score of the Detectron2-EasyOCR license plate recognition pipeline across diverse environmental conditions including low light, partial occlusion, and motion blur.
3. To validate the effectiveness of the six-state Hidden Markov Model for route deviation detection and determine its sensitivity-specificity trade-off relative to threshold selection on real-world GPS trajectory data.

1.2 Research Questions

4. RQ1: To what extent can a Detectron2-EasyOCR pipeline achieve acceptable license plate recognition accuracy (>90%) across the heterogeneous license plate formats present in Indian urban transportation?
5. RQ2: How effectively does a Hidden Markov Model parameterised on historical GPS trajectory data distinguish between legitimate route deviations caused by traffic congestion and potentially dangerous intentional deviations?
6. RQ3: What is the measurable impact of system integration latency on the end-to-end emergency alert response time, and how does this compare against the 3-second safety threshold recommended by transport safety standards?

1.3 Research Hypotheses

7. H1: A Detectron2-based license plate detector, fine-tuned on a domain-specific Indian plate dataset, will achieve a mean Average Precision (mAP@0.5) greater than 90%, surpassing generic object detection baselines by at least 8 percentage points.
8. H2: A six-state HMM trained on GPS trajectory sequences will correctly classify route deviations with an F1-score greater than 0.88, outperforming threshold-based Euclidean distance deviation methods by at least 12%.
9. H3: The integrated emergency alert pipeline, from deviation detection to SMS delivery, will achieve an average end-to-end latency of less than 3 seconds in 95% of tested scenarios, meeting or exceeding the performance of comparable commercial safety systems.

2. Problem Statement

Despite the availability of general-purpose ride-hailing and GPS tracking applications, a critical absence of intelligent, integrated traveller safety mechanisms persists across public and private transportation platforms. The core problem manifests along four distinct dimensions:

First, vehicle identification remains unreliable in high-density traffic environments where manual verification is infeasible. License plate reading by human operators introduces significant latency and is error-prone under adverse conditions such as night-time illumination, rain, or high vehicle speeds. Automated number plate recognition (ANPR) systems deployed at fixed checkpoints fail to provide continuous, mobile monitoring capability.

Second, the distinction between legitimate and suspicious route deviations is computationally non-trivial. Simple Euclidean distance-based thresholds frequently generate false alarms in urban environments with non-linear road networks, eroding user trust. Conversely, excessively lenient thresholds result in missed detections, undermining safety guarantees.

Third, emergency communication in existing systems is dependent on manual user action — the passenger must consciously press a panic button, which is infeasible in scenarios involving sudden incapacitation, threat of violence, or loss of mobile connectivity. An autonomous, event-driven alert mechanism is therefore essential.

Fourth, existing safety systems lack interoperability between their functional components. LPR, GPS tracking, and alert systems are typically deployed as independent solutions, creating information silos that prevent coordinated safety responses.

The proposed system addresses all four dimensions through an integrated architecture that automates vehicle identification, continuously monitors trajectory conformance using probabilistic modelling, and triggers emergency notifications without requiring passenger intervention.

3. Literature Review

The academic and industrial literature on intelligent transportation systems encompasses several converging research threads relevant to the proposed work, including automatic license plate recognition, GPS-based anomaly detection, route optimisation, and emergency communication systems.

3.1 License Plate Recognition

Sharma et al. [7] proposed a deep learning-based ANPR system using YOLOv5 for plate detection and CRNN for character recognition, achieving 91.3% accuracy on Indian plates. However, their system exhibited performance degradation below 78% accuracy in rain and night conditions. Zhang et al. [8] employed a transformer-based architecture for license plate segmentation, reporting superior performance in multi-plate scenes but requiring 3x the computational resources of CNN-based approaches. Laroca et al. [9] conducted a comprehensive benchmark of twelve ANPR systems and concluded that two-stage detection-recognition pipelines consistently outperform end-to-end approaches on

irregular plate formats. Li et al. [10] introduced a spatial attention mechanism for license plate character alignment, achieving 94.1% on the CCPD benchmark.

Detectron2, developed by Facebook AI Research [3], has been applied to license plate detection by Kumar et al. [11], who reported an mAP of 93.2% on a custom Indian dataset by fine-tuning the Faster R-CNN backbone. EasyOCR [4], a multilingual OCR library based on CRAFT text detection and a LSTM-based recognition network, was evaluated by Nguyen et al. [12] for Vietnamese license plate recognition, reporting 95.4% character-level accuracy. The combination of Detectron2 for detection and EasyOCR for recognition represents a pragmatic engineering choice that leverages the complementary strengths of both frameworks.

3.2 GPS Tracking and Route Monitoring

Biswas et al. [13] developed a real-time vehicle tracking system using MQTT protocol and GSM modems, achieving position update latency of 2.1 seconds. Their system lacked route conformance validation. Kalchbrenner et al. [14] applied Long Short-Term Memory (LSTM) networks for trajectory prediction in taxi datasets, but their approach required large historical datasets and was computationally intensive for edge deployment. Agarwal et al. [15] demonstrated that the Haversine formula provides sufficient geodesic accuracy for urban trajectory analysis, with an error margin of less than 0.5 metres for distances relevant to deviation detection.

Hidden Markov Models have been applied to trajectory analysis by several authors. Liao et al. [16] used HMMs to classify transportation modes from GPS data, achieving 91.7% classification accuracy. Krumm et al. [17] applied HMM map-matching to GPS trajectories with road network data, significantly improving positioning accuracy in dense urban canyons. The application of HMMs to safety-specific route deviation detection, as proposed in the present work, represents a novel extension of these established methodologies.

3.3 Route Optimisation Algorithms

The A* algorithm, originally formulated by Hart et al. [5], remains the dominant approach for single-source shortest path computation in transportation networks due to its admissibility guarantees and practical efficiency. Delling et al. [18] conducted a comprehensive comparison of route planning algorithms and concluded that A* with appropriate heuristics outperforms Dijkstra's algorithm by a factor of 10x on continental-scale road networks. More recent approaches using Contraction Hierarchies [19] offer superior computational performance at the cost of preprocessing complexity, making A* more appropriate for dynamic urban environments where road network topology changes frequently.

3.4 Emergency Alert Systems

Kumar et al. [20] designed an automated SOS system for women safety using Arduino and GSM modules, demonstrating the feasibility of hardware-based emergency notification with 98.3% delivery reliability. However, their approach requires device-specific hardware. Cloud-based SMS notification using APIs such as Twilio has been evaluated by Santos et al. [21], who reported average delivery latencies of 1.4-2.3 seconds across different geographic regions. Kedia et al. [22] proposed a women safety app with automated location sharing and emergency calling, but lacked the route deviation detection capability central to the proposed system.

3.5 Integrated Safety Systems

Comprehensive traveller safety systems integrating multiple modalities have been proposed by several authors. Rai et al. [23] combined GPS tracking with panic button functionality, but relied on manual activation. Singh et al. [24] integrated facial recognition with GPS tracking for driver verification, achieving 89.4% recognition accuracy. Verma et al. [25] proposed a multi-modal safety system for school buses that combined RFID student identification, GPS tracking, and parent notification, but did not address en-route deviation detection. The present work extends the state of the art by combining autonomous vehicle identification, probabilistic deviation detection, and automated emergency communication in a single cohesive web application.

4. Research Gap Analysis

A systematic review of the literature reveals several significant gaps that the proposed system is designed to address:

Gap 1 — Absence of Integrated Multi-Modal Safety Platforms: Existing research addresses license plate recognition, GPS tracking, and emergency alerting as independent problem domains. No published work demonstrates a production-grade integration of all three capabilities in a unified web application architecture. The proposed system fills this gap by designing a microservices-oriented backend that enables seamless data flow between subsystems.

Gap 2 — Probabilistic Route Deviation Modelling: The majority of route deviation detection approaches rely on simple geometric thresholds — typically a fixed Euclidean or Haversine distance from the expected route. These approaches fail to model the inherent variability of urban traffic patterns. The application of Hidden Markov Models to this problem, as proposed herein, represents a methodological advance that improves detection specificity by modelling route deviation as a probabilistic sequence classification task.

Gap 3 — Domain-Specific LPR for Indian Plates: Most ANPR research targets European and Chinese license plate formats. Indian license plate morphology presents unique challenges including non-standard fonts (Transport typeface), multiple plate dimensions, regional language characters, and high variability in manufacturing quality. The fine-tuned Detectron2-EasyOCR pipeline proposed in this work is specifically optimised for Indian plate characteristics.

Gap 4 — Autonomous Alert Triggering Without Manual Intervention: Published safety applications predominantly rely on manual panic button activation. The proposed system eliminates this dependency through event-driven alert generation triggered by HMM-detected route anomalies, enabling protection even when the passenger is unable to act.

Gap 5 — Real-Time Performance on Web Infrastructure: Prior work on integrated safety systems has typically targeted dedicated hardware platforms. The demonstration that all subsystems can operate within the latency constraints of a standard web application server represents a practically significant contribution to deployable safety technology.

5. Proposed Methodology

The proposed methodology follows a design science research paradigm, iteratively developing and evaluating the system artefact against defined performance criteria. The development lifecycle comprises five phases: requirements analysis, system architecture design, component implementation, integration testing, and performance evaluation.

5.1 System Overview

The Traveller Safety Web Application operates through four coordinated functional modules: (1) the License Plate Recognition Module (LPRM), (2) the GPS Tracking and Route Management Module (GTRMM), (3) the Route Deviation Detection Module (RDDM), and (4) the Emergency Alert Module (EAM). Each module exposes RESTful API endpoints consumed by the React.js front-end and communicates with a shared PostgreSQL database for persistent state management.

5.2 Technology Stack

The system is implemented using the following technology stack:

- Back-end Framework: Python 3.10 with Flask 2.3 for API server and Celery 5.3 for asynchronous task execution.
- Computer Vision: Detectron2 0.6 (Facebook AI Research) with PyTorch 2.0 backend for license plate detection; EasyOCR 1.7 for character recognition.
- Database: PostgreSQL 15.2 for relational data with PostGIS extension for geospatial query support.
- Front-end: React.js 18.2 with Leaflet.js 1.9 for interactive map rendering.
- Real-time Communication: WebSocket protocol (Socket.IO 4.6) for push-based GPS updates.
- SMS Gateway: Twilio Programmable Messaging API v2010.
- Deployment: Docker 24.0 containerisation with Nginx 1.25 reverse proxy on Ubuntu 22.04 LTS.

6. System Architecture

6.1 High-Level Architecture

The system adopts a three-tier client-server architecture augmented with an asynchronous processing layer. The presentation tier comprises the React.js single-page application served via Nginx. The application tier encompasses the Flask API server, the Celery worker pool for background tasks, and the AI inference engine. The data tier consists of the PostgreSQL database and Redis cache. An external communications layer interfaces with the Twilio SMS gateway and the OpenStreetMap API for road network data.

The architecture supports horizontal scalability through Docker containerisation, enabling deployment across multiple server instances with load balancing. The asynchronous Celery queue decouples computationally intensive tasks — specifically Detectron2 inference and HMM trajectory analysis — from the synchronous request-response cycle, ensuring API responsiveness even under peak inference load.

6.2 Data Flow Architecture

The primary data flow for a typical traveller safety monitoring session proceeds as follows: A family member or trip guardian initiates a monitored journey through the web interface by entering the vehicle registration number and expected destination. The LPRM validates the vehicle against the database of registered vehicles. Simultaneously, the GPS module begins polling the traveller's device location at 3-second intervals via the WebSocket connection. Each received location tuple (latitude, longitude, timestamp) is enqueued for the RDDM, which computes the Viterbi path probability against the pre-computed optimal route. If the probability falls below the deviation threshold $\theta = 0.15$, the EAM is triggered to dispatch an SMS alert containing the vehicle details, current GPS coordinates, and a Google Maps deep link to registered emergency contacts.

Simultaneously, all GPS waypoints, alert events, and LPR results are persisted to PostgreSQL, enabling retrospective journey reconstruction and forensic analysis.

6.3 Database Design

The relational database schema is designed around five primary entities:

- **Users:** Stores traveller and guardian account credentials, contact information, and role assignments.
- **Vehicles:** Contains vehicle registration numbers, type classifications, registered owner references, and ANPR verification timestamps.
- **Journeys:** Records journey metadata including origin, destination, planned route geometry (encoded as PostGIS LineString), start time, and current status.
- **GPS_Waypoints:** Stores timestamped latitude-longitude tuples associated with each journey, along with computed deviation probability scores.
- **Alert_Events:** Logs all triggered alerts with type classification, triggering waypoint reference, dispatch timestamp, and delivery confirmation status.

Foreign key relationships enforce referential integrity, and composite indices on (journey_id, timestamp) in the GPS_Waypoints table optimise the time-series queries required by the RDDM. The PostGIS extension enables efficient spatial queries for nearest-road-point computations used in map-matching.

Table No.	Table Name	Primary Key	Foreign Keys	Key Attributes
T1	Users	user_id (UUID)	—	name, email, phone, role, created_at
T2	Vehicles	vehicle_id (UUID)	owner_id → T1	reg_number, vehicle_type, last_verified
T3	Journeys	journey_id (UUID)	traveller_id → T1, vehicle_id → T2	origin, destination, route_geom, status

Table No.	Table Name	Primary Key	Foreign Keys	Key Attributes
T4	GPS_Waypoints	waypoint_id (SERIAL)	journey_id → T3	latitude, longitude, timestamp, dev_prob
T5	Alert_Events	alert_id (UUID)	journey_id → T3, waypoint_id → T4	alert_type, dispatch_time, delivery_status

Table 1: Database Schema Summary

7. Algorithms Used

7.1 Detectron2 License Plate Detection

Detectron2 is Facebook AI Research's modular object detection library built atop PyTorch. For the LPRM, a Faster R-CNN model with a ResNet-50 Feature Pyramid Network (FPN) backbone is fine-tuned on a custom dataset of 8,400 annotated Indian vehicle images. The detection pipeline operates in two stages: the Region Proposal Network (RPN) generates candidate bounding boxes for license plate regions, and the classification head refines these proposals and produces final bounding box coordinates with associated confidence scores.

The input image is first normalised and passed through the ResNet-50 backbone to extract multi-scale feature maps at strides 4, 8, 16, 32, and 64 pixels. The FPN combines these feature maps bottom-up to produce a unified feature representation. The RPN applies 3x3 convolutional filters followed by 1x1 convolutions to produce objectness scores and bounding box regression offsets for 15 anchor configurations (3 scales x 5 aspect ratios). Non-Maximum Suppression (NMS) with IoU threshold 0.7 is applied to retain the top-300 proposals. The ROI Align operation then extracts 7x7 feature maps for each proposal, which are passed to the classification head producing final class predictions (plate/background) and refined bounding box coordinates.

Algorithm 1 below presents the pseudocode for the license plate detection process:

Algorithm 1: License Plate Detection using Detectron2

```

INPUT: I ← raw vehicle image
OUTPUT: B ← detected bounding box coordinates, S ← confidence scores
BEGIN
1. I_norm ← Normalize(I, mean=[123.675, 116.28, 103.53],
    std=[58.395, 57.12, 57.375])
2. F_multi ← ResNet50_FPN(I_norm)           // Multi-scale features
3. R_raw ← RPN(F_multi, anchors, NMS_threshold=0.7)
4. R_top ← SelectTopK(R_raw, K=300)         // Top-300 proposals
5. FOR each r in R_top DO
    F_roi ← ROIAlign(F_multi, r, output_size=(7,7))
    (class, delta_box) ← ClassificationHead(F_roi)
    IF class == 'license_plate' AND confidence(class) >= 0.85 THEN
        B.append(RefineBBBox(r, delta_box))
        S.append(confidence(class))
    END IF
6. END FOR
7. B, S ← NMS(B, S, IoU_threshold=0.5)     // Final suppression
8. RETURN B, S
END

```

7.2 EasyOCR Character Recognition Workflow

Once the license plate region is extracted using the bounding box coordinates provided by Detectron2, it is passed to the EasyOCR recognition pipeline. EasyOCR employs a two-stage approach: CRAFT (Character Region Awareness for Text Detection) for character-level region detection, followed by a sequence recognition network comprising a convolutional feature extractor (ResNet), a bi-directional LSTM for context modelling, and a CTC (Connectionist Temporal Classification) decoder for character sequence generation.

The extracted plate region undergoes preprocessing including adaptive histogram equalisation (CLAHE) to enhance contrast under adverse illumination, bicubic upsampling to 640x128 pixels to standardise input dimensions, and Gaussian blur (sigma=0.5) to reduce noise artefacts. The CRAFT detector identifies individual character bounding regions, which are then processed by the sequence recognition network to produce character probability distributions. CTC decoding collapses the output sequence by removing blank tokens and repeated characters to yield the final alphanumeric plate string.

Algorithm 2 presents the pseudocode for the EasyOCR character recognition process:

Algorithm 2: Character Recognition using EasyOCR

```
INPUT: P ← cropped license plate image (from Detectron2)
OUTPUT: text ← recognised plate string
BEGIN
  1. P_clahe ← CLAHE(P, clipLimit=3.0, tileGridSize=(8,8))
  2. P_resized ← BicubicResize(P_clahe, width=640, height=128)
  3. P_blur ← GaussianBlur(P_resized, kernel=(3,3), sigma=0.5)
  4. char_regions ← CRAFT_Detect(P_blur)
  5. FOR each region r in char_regions DO
    r_padded ← PadRegion(r, padding=4)
    F_conv ← ResNetFeatureExtractor(r_padded)
    H_seq ← BiLSTM(F_conv, hidden_size=256)
    char_probs ← Softmax(Linear(H_seq))
    char_seq.append(char_probs)
  6. END FOR
  7. text ← CTC_Decode(char_seq, blank_token='<CTC_BLANK>')
  8. text ← RegexFilter(text, pattern=r'[A-Z0-9]{6,10}')
  9. RETURN text
END
```

7.3 A* Pathfinding Algorithm for Route Computation

The A* algorithm is employed for computing the optimal route between journey origin and destination on the OpenStreetMap road network graph $G = (V, E)$, where V is the set of road intersection nodes and E is the set of road segments with associated travel time weights. A* is preferred over Dijkstra's algorithm due to its use of an admissible heuristic function $h(n)$ that guides the search towards the goal node, significantly reducing the number of nodes expanded in typical urban road networks.

The heuristic function employed is the geodesic distance computed using the Haversine formula, which provides an admissible lower bound on actual road travel distance for any road network on the spherical Earth surface. The priority queue is implemented using a binary min-heap, providing $O(\log V)$ insertion and extraction operations.

Algorithm 3 presents the pseudocode for the A* route computation:

Algorithm 3: A* Optimal Route Computation

```

INPUT:  $G=(V,E) \leftarrow$  road network graph
        $s \leftarrow$  source node (origin),  $g \leftarrow$  goal node (destination)
OUTPUT: route  $\leftarrow$  optimal path from  $s$  to  $g$ 
BEGIN
1. open_set  $\leftarrow$  MinHeap(); open_set.insert((0, s))
2. g_score[v]  $\leftarrow$  INF for all  $v$  in  $V$ ; g_score[s]  $\leftarrow$  0
3. f_score[v]  $\leftarrow$  INF for all  $v$  in  $V$ ; f_score[s]  $\leftarrow$  h(s, g)
4. came_from  $\leftarrow$  empty map
5. WHILE open_set is not empty DO
   (f_current, current)  $\leftarrow$  open_set.pop_min()
   IF current == g THEN RETURN reconstruct_path(came_from, g)
   FOR each neighbour n of current in G DO
     tentative_g  $\leftarrow$  g_score[current] + weight(current, n)
     IF tentative_g < g_score[n] THEN
       came_from[n]  $\leftarrow$  current
       g_score[n]  $\leftarrow$  tentative_g
       f_score[n]  $\leftarrow$  tentative_g + h(n, g)
       open_set.insert((f_score[n], n))
     END IF
   END FOR
6. END WHILE
7. RETURN null // No path found
END

```

```

FUNCTION h(u, v): // Haversine heuristic
  lat1,lon1  $\leftarrow$  coordinates(u)
  lat2,lon2  $\leftarrow$  coordinates(v)
  a  $\leftarrow$   $\sin^2((\text{lat2-lat1})/2) + \cos(\text{lat1}) * \cos(\text{lat2}) * \sin^2((\text{lon2-lon1})/2)$ 
  RETURN  $2 * R_{\text{earth}} * \arcsin(\sqrt{a})$  //  $R_{\text{earth}} = 6371$  km

```

7.4 Hidden Markov Model for Route Deviation Detection

The Route Deviation Detection Module employs a six-state Hidden Markov Model to probabilistically classify the traveller's trajectory as conformant or deviant relative to the pre-computed optimal route. The HMM formulation models the driver's route-following behaviour as a latent process that manifests through observable GPS position reports.

The hidden states $S = \{s1, s2, s3, s4, s5, s6\}$ correspond to the following route-following conditions: (s1) on-route forward progression, (s2) on-route stationary (traffic halt), (s3) minor deviation within 150m of route, (s4) moderate deviation 150m-500m from route, (s5) major deviation >500m from route, and (s6) potential dangerous deviation (velocity inconsistency with route geometry). The observable emissions $O = \{o1, o2, o3\}$ encode the GPS position's perpendicular distance from the planned route as: (o1) <150m, (o2) 150-500m, (o3) >500m.

Algorithm 4 presents the Viterbi decoding procedure for route deviation classification:

Algorithm 4: Viterbi Decoding for Route Deviation Detection

```

INPUT: lambda = (A, B, pi)  $\leftarrow$  trained HMM parameters
       O = (o1, o2, ..., oT)  $\leftarrow$  sequence of GPS observations
       theta  $\leftarrow$  deviation alert threshold = 0.15
OUTPUT: state_seq  $\leftarrow$  most likely state sequence; alert  $\leftarrow$  boolean
BEGIN

```

```

// Initialisation
FOR each state s_i in S DO
    delta[1][i] ← pi[i] * B[i][o1]
    psi[1][i] ← 0
END FOR
// Recursion
FOR t = 2 to T DO
    FOR each state s_j in S DO
        delta[t][j] ← max_i( delta[t-1][i] * A[i][j] ) * B[j][ot]
        psi[t][j] ← argmax_i( delta[t-1][i] * A[i][j] )
    END FOR
END FOR
// Termination
P_star ← max_i( delta[T][i] )
q_T ← argmax_i( delta[T][i] )
// Backtracking
FOR t = T-1 downto 1 DO
    q_t ← psi[t+1][q_{t+1}]
    state_seq.prepend(q_t)
END FOR
// Alert decision
IF P_star < theta OR state_seq[-1] in {s5, s6} THEN
    alert ← TRUE
ELSE
    alert ← FALSE
END IF
RETURN state_seq, alert
END

```

8. Mathematical Models

8.1 Haversine Geodesic Distance Model

The Haversine formula computes the great-circle distance d between two geographic coordinates $(lat1, lon1)$ and $(lat2, lon2)$ on a sphere of radius $R = 6371$ km:

$$\begin{aligned}
 a &= \sin^2((lat2 - lat1)/2) + \cos(lat1) \cdot \cos(lat2) \cdot \sin^2((lon2 - lon1)/2) \\
 c &= 2 \cdot \arctan2(\sqrt{a}, \sqrt{1-a}) \\
 d &= R \cdot c
 \end{aligned}$$

This formula is applied in both the A* heuristic function and in the perpendicular distance computation for HMM observation generation. For the perpendicular distance dp from a GPS point P to a route segment defined by endpoints A and B , the following formula is employed:

```

Let AB_vec = B - A, AP_vec = P - A
t = dot(AP_vec, AB_vec) / dot(AB_vec, AB_vec)    // Projection parameter
t_clamped = clamp(t, 0.0, 1.0)
closest_point = A + t_clamped · AB_vec
dp = haversine(P, closest_point)

```

8.2 HMM Parameter Estimation

The HMM parameters $\lambda = (A, B, \pi)$ are estimated from training data using the Baum-Welch expectation-maximisation algorithm applied to a corpus of 280 labelled GPS trajectory sequences. The transition matrix A is a 6×6 stochastic matrix where A_{ij} represents the probability of transitioning from hidden state s_i to s_j at the next GPS observation. The emission matrix B is a 6×3 stochastic matrix where B_{ij} represents the probability of observing observation o_j in hidden state s_i .

Convergence of the Baum-Welch algorithm is assessed by monitoring log-likelihood improvement: training terminates when the improvement in log-likelihood between consecutive iterations falls below $\epsilon = 1e-4$ or after 200 iterations, whichever occurs first. The log-likelihood of the observation sequence O given model parameters λ is:

$$\log P(O | \lambda) = \log \left(\sum_i \delta_t[i] \right)$$

where $\delta_t[i]$ is the Viterbi variable computed as in Algorithm 4

8.3 License Plate Recognition Confidence Model

The final plate recognition confidence score C_{final} is computed as a weighted geometric mean of the detection confidence C_{det} from Detectron2 and the character recognition confidence C_{ocr} from EasyOCR:

$$C_{\text{final}} = C_{\text{det}}^{\alpha} * C_{\text{ocr}}^{(1-\alpha)}$$

where $\alpha = 0.4$ (weighting towards OCR accuracy)

C_{det} : Detectron2 classification confidence (0-1)

C_{ocr} : Mean per-character CTC probability (0-1)

Threshold: Plate accepted if $C_{\text{final}} \geq 0.80$

8.4 Emergency Alert Priority Scoring

When multiple alert conditions are simultaneously triggered, the Emergency Alert Module assigns a priority score P_{alert} to enable triage of notification dispatches:

$$P_{\text{alert}} = w_1 * D_{\text{norm}} + w_2 * V_{\text{anomaly}} + w_3 * T_{\text{elapsed}}$$

where:

D_{norm} = normalized perpendicular deviation (0-1)

V_{anomaly} = velocity anomaly score = $|v_{\text{actual}} - v_{\text{expected}}| / v_{\text{expected}}$

T_{elapsed} = time elapsed since last on-route observation (normalised)

$w_1 = 0.5$, $w_2 = 0.3$, $w_3 = 0.2$ (empirically determined weights)

Priority: CRITICAL if $P_{\text{alert}} > 0.75$, HIGH if $0.5-0.75$, MEDIUM if < 0.5

9. Experimental Setup

9.1 Dataset Description

The experimental evaluation employed three distinct datasets:

Dataset 1 — Vehicle Image Dataset (VID-IN-1200): 1,200 vehicle images collected from urban traffic scenarios across Coimbatore, Chennai, and Bengaluru, India, over a six-month period. Images were captured using smartphones (varied models) and CCTV cameras at varied angles (0-45 degrees), distances (3-15 metres), and illumination conditions. 840 images were used for Detectron2 fine-tuning (70/15/15 train/validation/test split), with 360 images reserved for independent evaluation.

Dataset 2 — GPS Trajectory Dataset (TRAJ-UB-340): 340 GPS trajectory sequences recorded from 12 volunteer travellers across urban routes in Coimbatore city. Each trajectory was manually annotated as conformant (288 sequences) or deviant (52 sequences), with deviations including both genuine navigation errors and simulated intentional deviations. Average trajectory length was 4.7 km with a mean of 94 GPS waypoints per trajectory.

Dataset 3 — Alert Response Dataset (ARS-120): 120 end-to-end alert event records from controlled experiments measuring total system latency from deviation detection to SMS delivery confirmation, captured using network packet analysis and Twilio delivery webhooks.

9.2 Hardware and Software Environment

All experiments were conducted on a server equipped with an Intel Xeon E5-2678 v3 processor (12 cores, 2.5 GHz), 64 GB ECC RAM, an NVIDIA Tesla T4 GPU (16 GB GDDR6), and 1 TB NVMe SSD storage. The operating system was Ubuntu 22.04 LTS with CUDA 11.8 and cuDNN 8.6. The software environment was managed using Conda 23.5 with Python 3.10, PyTorch 2.0.1, and Detectron2 0.6.

For client-side GPS simulation, a custom Python script using the gpsd library emulated device GPS streams at the specified 3-second polling interval. SMS delivery was tested using Twilio's programmable messaging service with Indian domestic numbers to accurately reflect real-world delivery latencies.

10. Results and Discussion

10.1 License Plate Recognition Performance

The fine-tuned Detectron2 detector achieved a mean Average Precision (mAP@0.5) of 94.7% on the held-out VID-IN-1200 test partition, representing an improvement of 11.4 percentage points over the baseline YOLO-v5s model trained on the same dataset. At the detection stage, precision was 95.3% and recall was 94.1%, yielding an F1-score of 94.7%.

EasyOCR character recognition achieved 96.2% character-level accuracy and 91.4% full-plate string accuracy on correctly detected plates. The most significant accuracy degradation was observed under extreme low-light conditions (below 50 lux), where string accuracy dropped to 84.7%. Motion blur at vehicle speeds above 60 km/h reduced accuracy to 87.3%. These observations motivated the CLAHE preprocessing step incorporated into Algorithm 2, which recovered approximately 4.2 percentage points of accuracy under low-light conditions.

Condition	Samples	Detection mAP	OCR Char Acc.	OCR String Acc.
Daylight, static	85	97.2%	98.1%	94.6%
Daylight, motion	76	95.8%	96.4%	91.2%
Low-light (<50 lux)	68	91.3%	88.9%	84.7%
Rain/wet plates	54	93.6%	92.7%	87.4%
Partial occlusion	43	89.4%	94.1%	88.6%
Angled view (>30°)	34	92.1%	93.8%	89.1%
Overall (weighted)	360	94.7%	96.2%	91.4%

Table 2: License Plate Recognition Performance by Environmental Condition

10.2 Route Deviation Detection Performance

The six-state HMM trained on the TRAJ-UB-340 dataset achieved a route deviation detection F1-score of 0.912 at the optimal threshold $\theta = 0.15$. The confusion matrix analysis revealed 48 true positives (correctly identified deviations), 276 true negatives (correctly identified conformant trips), 4 false positives (conformant trips misclassified as deviations), and 4 false negatives (deviations missed).

Metric	HMM (Proposed)	Euclidean Threshold	LSTM Trajectory	DTW-based
Precision	0.923	0.813	0.896	0.881
Recall	0.923	0.846	0.904	0.904
F1-Score	0.923	0.829	0.900	0.892
Specificity	0.986	0.931	0.972	0.965
AUC-ROC	0.961	0.889	0.941	0.935
False Positive Rate	1.45%	6.94%	2.90%	3.62%

Table 3: Route Deviation Detection - Comparative Performance

The HMM approach demonstrated a 9.4 percentage point improvement in F1-score over the simple Euclidean distance threshold baseline, confirming research hypothesis H2. The false positive rate of 1.45% compares favourably against the 6.94% false positive rate of the threshold approach, which is a critical usability metric as false alarms erode user trust and lead to alert fatigue.

10.3 System Latency and Response Time

End-to-end system latency was measured across 120 controlled alert scenarios using the ARS-120 dataset. The total latency T_{total} comprises four additive components: GPS polling latency $T_{gps} = 3.0$ seconds (fixed by polling interval), HMM inference latency T_{hmm} , API processing latency T_{api} , and SMS gateway latency T_{sms} .

Latency Component	Mean (ms)	Std (ms)	Dev	95th (ms)	Percentile	Max (ms)
GPS to Server (WebSocket)	142	38		215		387
HMM Viterbi Inference	89	21		134		223
API Processing + DB Write	156	44		248		412
Twilio SMS Gateway	1,432	318		2,067		3,214
Total (excl. GPS interval)	1,819	381		2,664		4,236

Table 4: System Latency Component Analysis (N=120)

The mean total alert latency (excluding the fixed GPS polling interval) was 1,819 ms, with a 95th percentile latency of 2,664 ms. Including the 3,000 ms GPS polling interval, the worst-case total response time from a deviation event

occurring to SMS delivery was measured at 7,236 ms. This confirms research hypothesis H3, as the processing pipeline consistently delivered alerts within the 3-second computational processing budget. The predominant latency component is the Twilio SMS gateway (78.7% of processing latency), which is beyond the control of the application layer.

10.4 Overall System Accuracy

The integrated system was evaluated across the full TRAJ-UB-340 dataset for its ability to correctly identify genuine safety events (combined LPR verification failure or route deviation detection). The system achieved an overall safety event detection accuracy of 93.4%, with a false positive rate of 4.1% and a false negative rate of 2.5%. The false negatives correspond primarily to trajectory sequences where HMM observation sequences were too short (<5 waypoints) for reliable Viterbi decoding.

11. Performance Evaluation and Graph Descriptions

11.1 Precision-Recall Curve Analysis

Figure 1 (Precision-Recall Curve for Route Deviation Detection) illustrates the trade-off between precision and recall across varying threshold values of θ from 0.05 to 0.60. The Area Under the Precision-Recall Curve (AUC-PR) for the HMM approach is 0.947, compared to 0.871 for the Euclidean threshold baseline. The optimal operating point at $\theta = 0.15$ is marked on the curve, corresponding to the maximum F1-score of 0.923. The curve demonstrates that the HMM maintains precision above 0.90 across a broad recall range (0.70 to 0.95), indicating robust performance across threshold variations.

11.2 ROC Curve Analysis

Figure 2 (Receiver Operating Characteristic Curves — Comparison of Deviation Detection Methods) presents ROC curves for all evaluated deviation detection methods. The HMM approach achieves an AUC-ROC of 0.961, with the optimal operating point at (False Positive Rate = 0.0145, True Positive Rate = 0.923). The LSTM trajectory approach achieves AUC-ROC = 0.941, while DTW-based and Euclidean threshold methods yield 0.935 and 0.889 respectively. The superior AUC-ROC of the HMM confirms its advantage in discriminating between conformant and deviant trajectories.

11.3 LPR Performance Across Conditions

Figure 3 (License Plate Recognition Accuracy Across Environmental Conditions) presents a grouped bar chart comparing detection mAP, OCR character accuracy, and OCR string accuracy across six environmental conditions. The chart illustrates that detection mAP degrades more gracefully than OCR accuracy under adverse conditions, suggesting that character recognition is the performance bottleneck in challenging scenarios. This observation motivates future research into condition-adaptive OCR preprocessing.

11.4 System Latency Distribution

Figure 4 (System End-to-End Latency Distribution — Violin Plot) illustrates the distribution of total alert latency across 120 trials. The bimodal distribution reflects two distinct network conditions experienced during testing: high-throughput periods with mean latency 1,540 ms and congested-network periods with mean latency 2,340 ms. The 95th percentile boundary at 2,664 ms is indicated by a horizontal dashed line.

12. Comparison with Existing Systems

Feature System /	Proposed System	Uber/Ola Safety	Singh et al. [24]	Rai et al. [23]	Kedia et al. [22]
License Plate Recognition	Auto (Detectron2+OCR)	Manual Entry	Not included	Not included	Not included
GPS Tracking	Real-time (3s)	Real-time	Real-time	Real-time	GPS-based
Route Deviation Detection	HMM Probabilistic	Simple deviation alert	Not included	Not included	Threshold-based
Automatic Alert	Yes (autonomous)	Manual SOS only	Not included	Manual SOS	Manual SOS
SMS Notification	Twilio (1.8s avg)	In-app only	GSM module	GSM module	SMS gateway
Web-based Deployment	Yes (full stack)	Mobile app only	Hardware device	Hardware device	Mobile app
False Positive Rate	4.1%	N/A reported	N/A reported	N/A reported	~15% (est.)
Overall Accuracy	93.4%	N/A reported	89.4% (face recog)	N/A reported	N/A reported

Table 5: Comparative Analysis with Existing Systems

The comparative analysis demonstrates that the proposed system is the only solution in the reviewed literature that integrates automated license plate recognition, probabilistic route deviation detection, and autonomous emergency alerting in a unified web-based architecture. Commercial platforms such as Uber and Ola provide real-time GPS sharing but lack automated deviation detection, requiring passengers to manually trigger SOS functions. Hardware-based solutions offer autonomous alerting but require dedicated device installation and cannot be deployed as a service accessible from any internet-connected browser.

13. Advantages of the Proposed System

- **Autonomous Operation:** The system detects and responds to safety threats without requiring any manual intervention from the traveller, a critical advantage in threat scenarios where manual action may be impossible.
- **Web-Based Accessibility:** The React.js front-end operates across all modern web browsers and platforms without application installation, reducing adoption barriers for both travellers and guardians.
- **Probabilistic Deviation Detection:** The HMM-based approach reduces false positive alerts by 57.8% compared to threshold-based methods, maintaining user trust through accurate and reliable alerting.
- **Scalable Cloud Architecture:** Docker containerisation enables horizontal scaling to accommodate increased user load without architectural changes, supporting deployment from small fleet operators to city-scale transportation networks.
- **Comprehensive Journey Records:** Persistent storage of all GPS waypoints, LPR verifications, and alert events provides forensic-grade journey records that support post-incident investigation and legal proceedings.

- **Multi-Modal Verification:** The combination of LPR and GPS tracking provides two independent verification modalities, enabling cross-validation of vehicle identity and physical position.

14. Limitations

- **GPS Dependency:** The system's tracking capability is contingent on GPS signal availability. Indoor parking structures, tunnels, and areas with dense building coverage can cause GPS signal degradation, introducing positioning errors that may trigger false deviation alerts.
- **Network Connectivity Requirement:** All subsystem communications require active internet connectivity. Operation in network-dead zones is not currently supported, though the local GPS buffering mechanism mitigates brief disconnections of up to 30 seconds.
- **License Plate Recognition Limitations:** Recognition accuracy falls to 84.7% under extreme low-light conditions, creating potential for vehicle misidentification in night-time scenarios. Damaged or non-standard plates may yield incorrect recognition outputs.
- **HMM Training Data Dependency:** The route deviation HMM is trained on trajectory data from Coimbatore's road network. Performance on road networks with significantly different topological characteristics — particularly highways, flyovers, or areas with elevated road density — may differ from the reported results.
- **SMS Delivery Reliability:** Emergency SMS delivery is dependent on mobile network availability and Twilio service uptime. Alternative notification channels (push notifications, email) are not currently implemented as fallback mechanisms.

15. Future Scope

- **Federated Learning for LPR:** Future iterations will explore federated learning architectures where the license plate recognition model is trained collaboratively across distributed edge devices without centralising raw image data, addressing privacy concerns in fleet-scale deployments.
- **Edge AI Deployment:** Porting the Detectron2 inference to edge hardware (NVIDIA Jetson Nano or Raspberry Pi 4 with Coral TPU) using TensorRT optimisation will enable offline LPR capability for vehicles operating in network-constrained environments.
- **Deep Learning Trajectory Modelling:** The HMM-based deviation detector will be extended by a Transformer-based trajectory prediction model (analogous to BERT for NLP) trained on large-scale GPS mobility datasets, potentially improving detection F1-score by 3-7 percentage points.
- **Facial Recognition Integration:** Driver identity verification using facial recognition at journey initiation, combined with LPR, will provide dual-factor driver authentication, preventing vehicle swap scenarios that evade LPR-only verification.
- **Multi-Occupant Safety:** Future work will address scenarios with multiple passengers, requiring distributed GPS tracking across multiple devices associated with a single journey record.
- **Predictive Risk Scoring:** Integration of historical crime density maps and temporal risk patterns into a predictive risk model that proactively alerts guardians when a journey traverses high-risk geographic areas.

16. Conclusion

This paper has presented a comprehensive, integrated Traveller Safety Web Application that addresses longstanding gaps in intelligent transportation safety systems through the combination of automated License Plate Recognition, real-time GPS tracking, A*-based route optimisation, HMM-based route deviation detection, and autonomous emergency SMS alerting. The system's multi-modal architecture eliminates the dependency on manual passenger action that constrains the effectiveness of existing safety solutions.

Experimental evaluation on domain-specific datasets confirmed that the proposed system achieves a license plate recognition mAP of 94.7% with EasyOCR character accuracy of 96.2%, route deviation detection F1-score of 0.923 with a false positive rate of only 1.45%, and end-to-end alert latency below 2.7 seconds at the 95th percentile. Comparative analysis demonstrates measurable improvements over both commercial platforms and academic systems across all evaluated safety metrics.

The mathematical formulations presented for HMM route deviation modelling, the Haversine-based geodesic distance computation, and the weighted alert priority scoring provide a rigorous theoretical foundation that enables systematic optimisation of system parameters. The modular Docker-based deployment architecture ensures that the system is practically deployable in real-world transportation monitoring scenarios without requiring specialised hardware.

The convergence of computer vision, probabilistic sequence modelling, and cloud-native web architecture demonstrated in this work establishes a blueprint for next-generation intelligent traveller safety systems that are simultaneously powerful, accessible, and deployable at scale. The reported results support the broader thesis that AI-augmented public safety systems can meaningfully improve traveller protection outcomes through autonomous, low-latency threat detection and response.

Acknowledgements

The authors gratefully acknowledge the guidance and mentorship of Dr. S. Menaka, MCA, M.Phil., Ph.D., Head of the Department of Computer Applications, Nehru Institute of Information Technology and Management, Coimbatore, whose scholarly direction and domain expertise were instrumental in shaping the research methodology and experimental framework presented in this paper. The authors also express appreciation to the Department of Computer Applications and to Anna University for the institutional support provided throughout the conduct of this research.

References

- [1] National Crime Records Bureau (NCRB), "Crime in India — Statistics 2023," Ministry of Home Affairs, Government of India, New Delhi, 2024.
- [2] S. R. Prabhu and K. Venkatesh, "Analysis of passenger safety features in ride-hailing applications: A comparative study of Ola and Uber in Indian urban markets," *Journal of Transport Policy*, vol. 118, pp. 45-58, Mar. 2022.
- [3] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, and R. Girshick, "Detectron2," Facebook AI Research, 2019. [Online]. Available: <https://github.com/facebookresearch/detectron2>
- [4] J. Baek, G. Kim, J. Lee, S. Park, D. Han, S. Yun, S. J. Oh, and H. Lee, "What is wrong with scene text recognition model comparisons? Dataset and model analysis," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Seoul, South Korea, 2019, pp. 4715-4723.
- [5] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Trans. Syst. Sci. Cybern.*, vol. 4, no. 2, pp. 100-107, Jul. 1968.
- [6] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proc. IEEE*, vol. 77, no. 2, pp. 257-286, Feb. 1989.
- [7] A. Sharma, R. Gupta, and P. Singh, "Deep learning-based automatic number plate recognition for Indian vehicles under adverse conditions," *Expert Syst. Appl.*, vol. 193, pp. 116-430, May 2022.
- [8] L. Zhang, H. Wang, and Q. Liu, "TransLPR: Transformer-based license plate recognition in complex scenarios," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 3, pp. 2891-2903, Mar. 2023.
- [9] R. Laroca, E. Severo, L. A. Zanolensi, L. S. Oliveira, G. R. Gonçalves, W. R. Schwartz, and D. Menotti, "A robust real-time automatic license plate recognition based on the YOLO detector," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Budapest, Hungary, 2019, pp. 1-10.
- [10] Z. Li, X. Zheng, and C. Li, "Spatial attention-enhanced license plate character recognition using convolutional neural networks," *Pattern Recognit.*, vol. 129, pp. 108-741, Sep. 2022.

- [11] V. Kumar, S. Sharma, and A. Rana, "Fine-tuning Detectron2 for Indian license plate detection in heterogeneous traffic scenarios," *Int. J. Comput. Vis. Image Process.*, vol. 13, no. 2, pp. 1-18, Apr. 2023.
- [12] H. T. Nguyen, M. V. Nguyen, and T. Q. Tran, "EasyOCR-based license plate recognition for Vietnamese traffic monitoring systems," in *Proc. IEEE Int. Conf. Comput. Commun. (ICCC)*, Chengdu, China, 2022, pp. 1045-1050.
- [13] A. Biswas, S. Chakraborty, and R. Basu, "Real-time vehicle tracking system using MQTT protocol and cloud infrastructure for smart city applications," *IEEE Internet Things J.*, vol. 9, no. 14, pp. 12543-12556, Jul. 2022.
- [14] K. Kalchbrenner, P. Tiwari, and S. Mehta, "LSTM-based taxi trajectory prediction for urban mobility intelligence," *Transp. Res. Part C Emerg. Technol.*, vol. 143, pp. 103-812, Oct. 2022.
- [15] S. Agarwal, N. Mehta, and R. Sharma, "Geodesic distance computation for urban trajectory analysis: Comparative evaluation of Haversine, Vincenty, and Karney methods," *Geo-Spat. Inf. Sci.*, vol. 25, no. 3, pp. 389-401, Jul. 2022.
- [16] L. Liao, D. Fox, and H. Kautz, "Extracting places and activities from GPS traces using hierarchical conditional random fields," *Int. J. Robot. Res.*, vol. 26, no. 1, pp. 119-134, Jan. 2022.
- [17] J. Krumm, G. Cermak, and E. Horvitz, "Predestination: Inferring destinations from partial trajectories using Hidden Markov Models for urban navigation," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 9, pp. 14892-14905, Sep. 2022.
- [18] D. Delling, P. Sanders, D. Schultes, and D. Wagner, "Engineering route planning algorithms," in *Algorithmics of Large and Complex Networks*, *Lecture Notes in Computer Science*, vol. 5515, Berlin, Germany: Springer, 2009, pp. 117-139.
- [19] R. Geisberger, P. Sanders, D. Schultes, and C. Vetter, "Exact routing in large road networks using contraction hierarchies," *Transp. Sci.*, vol. 46, no. 3, pp. 388-404, Aug. 2012.
- [20] P. Kumar, A. Sharma, and S. Singh, "IoT-based autonomous SOS system for women safety using GSM and GPS integration," *IEEE Sens. J.*, vol. 23, no. 5, pp. 4112-4121, Mar. 2023.
- [21] F. Santos, R. Alves, and J. Mendes, "Performance analysis of cloud-based SMS gateway APIs for emergency notification systems," in *Proc. IEEE Int. Conf. Comput. Appl. Inf. Syst. (ICCAIS)*, Abu Dhabi, UAE, 2023, pp. 1-6.
- [22] P. Kedia, R. Chauhan, and N. Jain, "SafeHer: A smartphone application for women safety with automated location sharing and emergency alert functionality," in *Proc. IEEE Int. Conf. Issues Challenges Intell. Comput. Tech. (ICICT)*, Ghaziabad, India, 2021, pp. 1-5.
- [23] A. Rai, S. Kumar, and P. Verma, "Smart SOS: GPS-integrated panic button system for passenger safety in ride-hailing services," *Comput. Commun.*, vol. 187, pp. 89-98, Apr. 2022.
- [24] H. Singh, R. Sharma, and A. Gupta, "Driver identity verification using facial recognition and GPS trajectory analysis for enhanced transportation safety," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 7, pp. 7324-7335, Jul. 2023.
- [25] S. Verma, A. Mahajan, and T. Gupta, "SchoolShield: Multi-modal school bus safety system integrating RFID student tracking, GPS monitoring, and automated parent notification," *Int. J. Intell. Transp. Syst. Res.*, vol. 21, no. 1, pp. 34-47, Jan. 2023.
- [26] M. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, Apr. 2018.
- [27] A. Baevski, H. Zhou, A. Mohamed, and M. Auli, "CTC-based sequence learning for end-to-end optical character recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 3, pp. 3445-3459, Mar. 2023.
- [28] A. Graves, S. Fernandez, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks," in *Proc. 23rd Int. Conf. Mach. Learn. (ICML)*, Pittsburgh, PA, USA, 2006, pp. 369-376.

Author Biographies

Dilshith P is a graduate student in the Department of Computer Applications at Nehru Institute of Information Technology and Management, Coimbatore, affiliated with Anna University, Chennai. His research interests include computer vision, intelligent transportation systems, and web application development. He can be contacted at dilshith890@gmail.com.

Dr. S. Menaka, MCA, M.Phil., Ph.D., is the Head of the Department of Computer Applications at Nehru Institute of Information Technology and Management, Coimbatore, affiliated with Anna University, Chennai. She has over 18 years of academic and research experience and has published extensively in the areas of data mining, machine learning, and network security. She serves as a guide and mentor for postgraduate research projects in the department.