IJCRT.ORG

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

# AI-Powered Smart Navigation System for Urban Route Optimization: A Multi- Objective Approach Integrating Safety, Traffic, and Flood Risk Intelligence

**SIDDARTH RK, RENGESH PSR, AND ARUN SATHYAMURTHY**

[1]Department of Computer Science and Engineering, [Sri Venketeswara College of Engineering], Chennai 600001, India[2]Department of computer science and engineering, [Sri Venketeswara College of Engineering], Chennai 600025, India

Doi: https://doi.org/10.56975/ijcrt.v14i7.308991

**ABSTRACT**

Urban navigation in megacities such as Chennai, India, presents multifaceted challenges arising from high traffic density, seasonal flooding, accident-prone zones, and the inadequacy of existing systems that optimize solely for travel time and distance. This paper proposes a comprehensive AI-powered Smart Navigation System (SNS) for urban route optimization that integrates multi-objective routing, machine learning-based risk prediction, and real-time data fusion. The proposed architecture combines graph-theoretic routing on the OpenStreetMap road network via OSMnx and NetworkX, an XGBoost-based accident risk classifier, a time-series congestion predictor, and a flood-zone awareness layer specifically calibrated for Chennai's monsoon environment. The system computes four distinct route classes— fastest, shortest, safest, and traffic-optimized — scored by a composite safety metric: Safety = 100 − (AccidentRisk + FloodRisk + CongestionPenalty). Evaluation on a real-world Chennai road network demonstrates a route safety improvement of 31.4% over shortest-path baselines, an 18.7% reduction in average travel time under congested conditions compared to static routing, and a flood-zone avoidance accuracy of 94.2%. A full multi-modal transportation planner integrating metro, bus, and driving modes is additionally provided. Results confirm the proposed SNS significantly outperforms conventional navigation approaches and offers a scalable framework for other complex urban environments.

**Index Terms—** *Urban route optimization, intelligent navigation, multi-objective routing, accident risk prediction, flood-aware navigation, machine learning, graph-based pathfinding, XGBoost, OpenStreetMap, Chennai, ITS.*

# I. INTRODUCTION

## *A. Background and Motivation*

The rapid expansion of urban populations worldwide has placed unprecedented pressure on city transportation infrastructures. Chennai, India—with a population exceeding 10 million—experiences daily gridlock, unpredictable monsoon flooding, and chronic road safety incidents that collectively degrade quality of life and economic productivity. The global urban navigation market, dominated by applications such as Google Maps and Waze, primarily optimizes routing based on historical travel times and real-time congestion feeds, with limited consideration of localized safety hazards, flood-prone corridors, or integrated multi-modal planning [1].

The emergence of intelligent transportation systems (ITS), powered by graph-based algorithms, machine learning, and real- time sensor fusion, has opened new avenues for context-aware, safety-first urban navigation [2]. Graph-theoretic approaches model road networks as weighted directed graphs, allowing the application of shortest-path algorithms and their multi-criteria extensions [3]. Simultaneously, predictive ML models have demonstrated strong performance in forecasting traffic congestion and accident risk [4],[5]. However, very few systems synthesize all of these dimensions into a single coherent, deployable navigation framework targeted at the operational realities of a complex Indian metropolis.

Chennai's geography further compounds navigation difficulty. The city lies on the Coromandel Coast, and its flat topography with inadequate stormwater drainage makes it highly susceptible to flash flooding during both the northeast and southwest monsoon seasons. The 2015 Chennai floods—among the worst urban flood disasters in South Asia in a century—rendered vast sections of the road network impassable for weeks, caused over 500 fatalities, and inflicted economic losses estimated at Rs. 20,000 crore [6]. Generic navigation platforms, not calibrated to local hydrology, proved inadequate in routing commuters away from flooded arteries. This failure underscores the life-critical importance of city-specific, flood-aware navigation intelligence.

## *B. Problem Statement*

Despite advances in navigation technology, three critical gaps persist. First, existing systems treat safety, traffic, and geographic hazards as independent concerns, providing no unified multi-objective routing framework. Second, city-specific hazards such as Chennai's documented susceptibility to catastrophic flooding [6] are largely absent from generic navigation platforms. Third, multi-modal route planning combining driving, metro, and bus options in a single decision framework remains underdeveloped for Indian metropolitan contexts.

Additionally, most commercial navigation platforms do not expose their algorithmic internals, precluding academic validation, custom risk model integration, or adaptation for specific urban hazard profiles. The proposed Smart Navigation System (SNS) directly addresses all three gaps through an open, modular, full-stack architecture validated on real-world Chennai data.

## *C. Scope and Objectives*

The primary objective of this research is to design, implement, and evaluate an AI-powered navigation system for Chennai that simultaneously optimizes four routing objectives: travel time, physical distance, safety risk, and flood exposure. Secondary
objectives include: (i) training a city-calibrated ML model for accident risk prediction using historical incident data; (ii) building a real-time data pipeline integrating traffic APIs, rain gauges, and incident databases; (iii) developing a multi-modal planner that considers bus, metro, and driving alternatives holistically; and (iv) providing an open-source, extensible framework applicable to other flood-prone Indian cities such as Mumbai,

Kochi, and Kolkata.

### *D. Main Contributions*

The main contributions of this paper are:

- A multi-objective routing framework simultaneously optimizing travel time, distance, safety risk, and traffic via a composite weighted cost function over a directed road graph;
- An XGBoost-based accident hotspot prediction model trained on 14,823 Chennai-specific historical incident records (2018–2023), embedded as dynamic edge weights in the routing graph;
- A flood-zone awareness layer with severity-graded road exclusion (four severity levels), calibrated for Chennai's monsoon hydrology using GCC GIS atlas and SDMA rain gauge data;
- A rolling re-optimization mechanism adapted from predictive control theory [8], enabling dynamic route switching during active navigation based on real-time speed deviations;
- A real-time multi-modal transportation planner integrating driving, MTC bus, and CMRL metro networks with live ETA prediction and transfer penalty modeling;
- A full-stack validated implementation on the OpenStreetMap Chennai road network (~42K nodes, ~98K edges) with statistically significant performance improvements over multiple baselines;
- A comprehensive system latency evaluation confirming sub-500 ms route computation at the 95th percentile, meeting interactive navigation usability standards.

## II. LITERATURE REVIEW

### *A. Traffic Forecasting and Dynamic Routing*

Wang and Wang [8] introduced a real-time dynamic route optimization model based on the predictive control principle, treating driving time as a controlled variable and road speed as a manipulated variable. While algorithmically elegant and avoiding dependence on traffic flow prediction, the model is restricted to single-objective (time-minimal) routing and does not address safety or flood hazards. The rolling optimization philosophy of [8] directly informs the feedback correction mechanism adopted here.

Vlahogianni et al. [1] surveyed short-term traffic forecasting approaches, categorizing statistical, ML, and deep learning methods. Their work confirmed the superiority of data-driven forecasting but noted a persistent disconnect between prediction quality and actionable route optimization—a gap the proposed system bridges. Zhao et al. [9] proposed an LSTM network for short-term traffic speed forecasting, achieving mean absolute errors below 3.2 km/h on urban expressways. The work focuses exclusively on speed prediction, providing no route planning mechanism.

Ma et al. [10] employed deep CNNs to model traffic as spatial images for large-scale network speed prediction. Though scalable, the approach is computationally expensive and not adapted for safety-aware or flood-aware routing. Duan et al. [13] introduced deep hybrid CNN-LSTM networks for urban traffic flow prediction, showing strong predictive performance but requiring large training datasets with no route planning integration. These deep learning approaches, while powerful, introduce significant inference latency incompatible with real-time interactive navigation.

### *B. Safety-Aware and Multi-Criteria Routing*

Zhou et al. [11] developed a two-level hierarchical MPC architecture for large-scale urban traffic signal networks, demonstrating predictive control effectiveness in traffic management at the infrastructure level. Their framework does not address individual vehicle routing. Rahman et al. [12] proposed an adaptive

route selection system using real-time road traffic information for Bangladeshi urban roads. Their system lacks ML-based risk assessment and flood intelligence, and has not been validated on large-scale road networks.

Zhi et al. [14] addressed vehicle routing using travel time reliability metrics, improving schedule adherence but not incorporating accident risk, flood hazard, or multi-modal options. Khanda et al. [15] studied efficient route selection for drone delivery under time-varying dynamics using graph-based methods with temporal edge weights— an approach that informs the temporal weighting strategy of the present system. None of these works, however, address the compound challenge of accident risk, flood hazard, congestion, and multi-modal integration simultaneously.

### *C. Flood-Aware Navigation and Urban Resilience*

Urban flood routing has received limited attention in the navigation literature despite its practical importance in monsoon- affected cities. Adinarayana et al. [6] provided a GIS-based flood vulnerability assessment for Chennai, identifying high-risk zones and drainage bottlenecks. While this work provides critical geospatial intelligence, it does not translate vulnerability maps into routing decisions. The proposed system operationalizes this vulnerability data by converting GCC GIS flood polygons into dynamic edge weights, enabling real-time flood-avoidance routing that activates automatically based on rain gauge thresholds.

### *D. Research Gap and Positioning*

In summary, existing works address individual sub-problems in isolation. No prior work comprehensively addresses multi-
objective routing with integrated safety, flood, and multi-modal intelligence for an Indian megacity context. Commercial platforms such as Google Maps include limited flood alerts in some markets but do not expose algorithmic safety scoring, do not integrate local government hydrology data, and do not provide a unified multi-modal optimization framework. This paper fills that gap with a validated, open-architecture system.

## III. PROPOSED SYSTEM AND METHODOLOGY

### *A. Overall System Architecture*

The proposed SNS follows a modular three-tier architecture as shown in Figure 1. The three tiers are: (1) a Frontend Layer implemented in React 18 with TypeScript, Leaflet.js 1.9 for interactive mapping, and TanStack Query for real-time state management; (2) a Backend API Layer in Python FastAPI handling route computation, ML inference, real-time data aggregation, and RESTful endpoint exposure; and (3) a Data and Intelligence Layer comprising OSMnx/NetworkX for graph routing, Firebase Realtime Database for incident management, TomTom API for live traffic feeds, SDMA rain gauge data for flood activation, and trained ML models for risk prediction.

The separation of concerns enables independent scaling of each tier. The Backend API layer is horizontally scalable behind a load balancer, the graph data is preloaded into memory at server startup for low-latency path queries, and the Firebase layer provides sub-second incident propagation to all connected clients. This architecture supports concurrent multi-user navigation sessions with consistent route quality.

*[ FIGURE 1: Three-Tier System Architecture— Frontend / Backend API / Data-Intelligence Layer ] Fig. 1. Proposed Smart Navigation System architecture showing the three-tier modular design.*

### *B. Road Network Graph Model*

The road network is modeled as a weighted directed graph G = (V, E, W), where V is the set of intersections (nodes), E is the set of road segments (directed edges), and W is a multi-dimensional weight function. For each directed edge $e^{Ij}$, the composite edge weight is defined as:

$$w^{Ij} = \alpha \cdot t^{Ij} + \beta \cdot d^{Ij} + \gamma \cdot r^{Ij} + \delta \cdot f^{Ij} \quad (1)$$

where $t^{Ij}$ is normalized travel time, $d^{Ij}$ is normalized distance, $r^{Ij}$ is the accident risk score, and $f^{Ij}$ is the flood severity index for edge (i,j). Coefficients $\alpha, \beta, \gamma, \delta \in [0,1]$ with $\alpha+\beta+\gamma+\delta = 1$ are configured per route optimization mode. All component weights are min-max normalized to [0,1] prior to combination, ensuring dimensional consistency across heterogeneous data sources.

The Chennai road network graph contains approximately 42,000 nodes and 98,000 directed edges extracted via OSMnx from OpenStreetMap. The graph is restricted to drivable road types (motorway, trunk, primary, secondary, tertiary, residential) and projected to UTM zone 44N for accurate metric distance computation. Graph loading is performed as a background process at server startup and cached in memory, reducing per-request latency to under 100 ms for typical urban route queries.

### *C. Multi-Objective Route Optimization*

Four optimization modes are defined by varying the weight vector ($\alpha, \beta, \gamma, \delta$): (i) Fastest: (0.7, 0.1, 0.1, 0.1); (ii) Shortest: (0.1, 0.8, 0.05, 0.05); (iii) Safest: (0.1, 0.1, 0.7, 0.1); (iv) Traffic-Optimized: (0.5, 0.1, 0.2, 0.2). All four modes employ Dijkstra's algorithm on the configured weighted graph, extended to support multi-dimensional edge weights. The weight configuration for each mode reflects its primary optimization objective while retaining partial weighting for secondary objectives, preventing pathological routing behavior such as choosing an extremely dangerous fast route.

Users can also define custom weight vectors within the constraint $\alpha+\beta+\gamma+\delta = 1$, enabling personalized optimization profiles. For example, a user commuting during monsoon season may prefer a flood-weighted custom profile (0.2, 0.1, 0.3, 0.4) that strongly penalizes flood exposure while maintaining reasonable time performance.

### *D. Composite Safety Score*

A segment-level safety score $S^{Ij} \in [0, 100]$ is assigned to each road edge, aggregating three independent risk dimensions:

$$S^{Ij} = 100 - (\rho \cdot R_a^{cc} + \sigma \cdot R\check{E}l^{ood} + \tau \cdot R^{cong}) \quad (2)$$

where $R_a^{cc}$ is the XGBoost classifier output probability scaled to [0,100], $R\check{E}l^{ood} \in \{0, 25, 50, 75, 100\}$ is the mapped flood severity index, and $R^{cong}$ is the real-time congestion penalty. The empirically tuned weights $\rho = 0.40, \sigma = 0.35, \tau = 0.25$ reflect the relative impact of each risk type on overall route safety in Chennai's context, derived from analysis of historical incident severity distributions. A route's aggregate safety score is the harmonic mean of its constituent edge safety scores, giving appropriate downward weight to the most dangerous segments.

### *E. Accident Risk Prediction Model*

An XGBoost gradient boosted decision tree classifier [5] is trained on 14,823 historical Chennai accident records obtained from the Tamil Nadu Police Department (TNPD) and the Ministry of Road Transport and Highways (MoRTH) for the period 2018–2023. The feature set includes: hour-of-day (cyclic-encoded), day-of-week (one-hot encoded), GPS coordinates (latitude, longitude), road category (motorway to residential, one-hot), binary rain flag, traffic density decile, road lighting status (binary), and nearest intersection type (one-hot:

signalized/unsignalized/roundabout).

Class imbalance— with accident events constituting approximately 25% of training samples— is addressed using Synthetic Minority Oversampling Technique (SMOTE) at a 3:1 negative-to-positive ratio. Training employs 5-fold stratified cross- validation to preserve class distribution across folds. The XGBoost model uses 500 estimators, maximum tree depth of 6, learning rate of 0.05, and L2 regularization coefficient of 1.0, tuned via Bayesian optimization on the validation fold. The classification threshold is set at P = 0.42 via precision-recall analysis to minimize safety-critical false negatives (missed high- risk zones).

For runtime integration, the trained model is serialized using joblib and loaded into the FastAPI backend. At each routing request, the model performs batch inference over all candidate edge segments, with the output probability linearly mapped to $R_a^{cc} \in [0, 100]$ and stored as an edge attribute in the NetworkX graph. Model inference across all ~98K edges takes approximately 340 ms on a standard server CPU, and results are cached for 15-minute intervals to balance freshness with computational cost.

### *F. Traffic Congestion Prediction*

Congestion prediction employs a gradient boosting regressor with input features: hour-of-day, day-of-week, weather (binary rain), historical speed percentiles (25th, 50th, 75th, 90th) for the road segment, and 15-minute rolling average speed from the TomTom Traffic API (refreshed every 30 seconds). The congestion penalty is computed consistently with the predictive control formulation of [8]:

$$R^{cong} = 100 \cdot (1 - V^e / V^h) \quad (3)$$

where $V^e$ is the real-time enabled speed and $V^h$ is the posted speed limit for the road segment. This formulation yields $R^{cong} = 0$ under free-flow conditions and approaches 100 under near-standstill congestion. The gradient boosting regressor achieves a MAPE of 7.3% on holdout test data, outperforming LSTM, ARIMA, and historical average baselines (see Table III). For road segments with insufficient historical data (typically minor residential roads), congestion is imputed from the median of neighboring same-category edges within a 500-meter spatial buffer.

### *G. Flood Zone Awareness Layer*

Flood risk data is sourced from the Greater Chennai Corporation (GCC) GIS flood vulnerability atlas, which provides city- wide polygon layers delineating flood-prone zones based on 50-year return period rainfall scenarios and historical inundation extents. These polygons are augmented with live readings from 39 SDMA rain gauge stations distributed across Chennai's 15 zones. Road segments receive a flood severity index $R_{flood} \in \{0, 1, 2, 3\}$ corresponding to no risk, low inundation risk (passable with caution), moderate inundation (risk of vehicle damage), and severe inundation (impassable). During active rainfall events exceeding 30 mm/hr at the nearest gauge, severity-3 segments are excluded from the routing graph ($f^{r_{ij}} \to \infty$) and severity-2 segments receive maximum flood penalty ($f^{r_{ij}} = 1.0$).

The GCC flood polygon layers are rasterized at 10-meter resolution and intersected with road segment centroids to assign per- edge severity indices. Flood activation is governed by a tiered trigger protocol: light rain (<15 mm/hr) produces no routing change; moderate rain (15–30 mm/hr) upgrades severity indices by one level; heavy rain (>30 mm/hr) activates full severity exclusion. This tiered protocol prevents unnecessary route disruption during minor precipitation while ensuring aggressive avoidance during dangerous flooding conditions.

### *H. Rolling Re-Optimization Mechanism*

Drawing from the predictive control paradigm of Wang and Wang [8], the system performs rolling re-optimization after each completed road segment during active navigation. If the real-time speed $V^e$ falls below a

threshold $V_{\in}$ — the theoretical speed required to meet the desired arrival time $t^{y}$ — a route switch is evaluated. The desired arrival time is computed from the initial reference route:

$$t^{y} = S_o / \bar{V} \quad (4)$$

where $S_o$ is the initial static reference route length and $\bar{V}$ is the estimated average speed. Route switching follows a five- condition protocol adapted from [8]: (C1) current speed below threshold; (C2) alternative route exists with lower composite cost; (C3) alternative route does not re-enter a recently penalized congestion zone; (C4) time saving exceeds 90 seconds (to prevent oscillatory switching); and (C5) flood severity on alternative route is equal or lower. When all five conditions are satisfied, the system dynamically reroutes and notifies the user through the frontend interface.

### *I. Multi-Modal Transportation Integration*

CMRL metro schedules (107 stations across the Blue and Green lines) and MTC bus routes (820+ routes covering the metropolitan area) are integrated into a unified multi-modal graph augmenting the base driving graph. Inter-modal transfer nodes are defined at metro stations and major bus terminals. Transfer edges carry a time penalty reflecting average walking time to the platform plus average waiting time based on headway schedules. A modified Dijkstra's algorithm with transfer time penalties at modal switch edges solves the multi-modal shortest path problem for any combination of driving, bus, and metro segments. Live bus ETA predictions are provided through integration with the MTC real-time tracking API where available, with schedule-based estimates as fallback.

## IV. IMPLEMENTATION DETAILS

### *A. Technology Stack*

Frontend: React 18, TypeScript 5.0, Vite 5.0, Tailwind CSS 3.4, Leaflet.js 1.9, Shadcn UI component library, TanStack Query 5.0 for server state management, React Hook Form 7.0.

Backend: Python 3.11, FastAPI 0.110, Uvicorn ASGI server, OSMnx 1.9, NetworkX 3.2, Firebase Admin SDK 6.2, BeautifulSoup 4.12 for live news scraping, Scikit-learn 1.4, XGBoost 2.0, Joblib 1.3.

Infrastructure: Firebase Realtime Database for incident management and real-time push; TomTom Traffic Flow API (30-second refresh); SDMA rain gauge API (60-second refresh); OpenStreetMap via Overpass API for map tiles; LiveChennai.com scraping for local incident news.

### *B. Datasets*

*TABLE I — Datasets Used in the Proposed System*

| Dataset | Source | Size / Coverage | Usage |
|---|---|---|---|
| Chennai Road Network | OpenStreetMap (OSMnx) | ~42K nodes, ~98K edges | Graph routing |
| Historical Accidents | TNPD / MoRTH | 14,823 records (2018–23) | Risk model training |
| Flood Vulnerability | GCC GIS Atlas | City-wide polygon layers | Flood zone tagging |
| Real-Time Traffic | TomTom API | Live, 30-sec refresh | Congestion penalty |

| | | | |
|---|---|---|---|
| Metro/Bus Schedules | CMRL / MTC Open Data | 107 stations, 820+ routes | Multi-modal planning |
| Rain / Weather | SDMA Rain Gauges | 39 stations, hourly | Flood trigger activation |
| Incident News | LiveChennai.com (scraped) | Continuous, ~30 articles/day | Incident layer overlay |

### *C. Preprocessing Pipeline*

The OSMnx graph is simplified to drivable road types and projected to UTM zone 44N for metric accuracy. Missing speed values are imputed using road-category medians (e.g., 50 km/h for primary roads, 30 km/h for residential). Accident records are spatially joined to the nearest road segment within a 25-meter tolerance. Flood polygons are rasterized at 10-meter resolution and intersected with road segment centroids to assign severity indices. For ML training, categorical features are one-hot encoded, numerical features are normalized via min-max scaling, and class imbalance is addressed with SMOTE at a 3:1 negative-to-positive ratio before model training.

### *D. Frontend Architecture*

The React frontend is organized into five principal views: the main Navigation Dashboard (route input, live map with layer toggles), the Traffic Dashboard (area-level congestion heat map, peak hour forecasts), the Flood Dashboard (severity zone
overlay, rain gauge readings, alert status), the Route Comparison View (side-by-side comparison of all four route modes with time, distance, and safety score), and the User Account view (login, saved routes, incident reporting). The Leaflet.js map supports three base tile layers—Street, Satellite, and Hybrid—and seven optional overlay layers (flood zones, metro network, bus routes, accident hotspots, traffic congestion, weather, and real-time incidents).

TanStack Query provides intelligent caching and background refetching: traffic data is invalidated every 30 seconds, weather data every 60 seconds, and incident data every 15 seconds. This approach minimizes unnecessary API calls while ensuring data freshness critical for real-time navigation safety.

## V. RESULTS AND DISCUSSION

### *A. Accident Risk Classifier Performance*

The XGBoost accident risk classifier was evaluated using 5-fold stratified cross-validation, ensuring consistent class distribution across train/validation splits. Results are compared against Random Forest and Logistic Regression baselines trained on identical features and preprocessing pipelines.

***TABLE II— Accident Risk Classifier Performance (5-Fold Stratified CV)***

| Metric | XGBoost (Proposed) | Random Forest | Logistic Regression |
|---|---|---|---|
| Accuracy | 91.4% | 88.9% | 79.3% |
| Precision | 88.7% | 85.1% | 73.6% |

| Metric | XGBoost (Proposed) | Random Forest | Logistic Regression |
|---|---|---|---|
| Recall | 90.2% | 86.8% | 77.4% |
| F1-Score | 89.4% | 85.9% | 75.4% |
| AUC-ROC | 0.963 | 0.941 | 0.847 |
| Training Time (s) | 18.4 | 11.7 | 2.1 |
| Inference (ms / edge) | 0.0035 | 0.0041 | 0.0008 |

The XGBoost classifier achieves an AUC-ROC of 0.963, indicating strong discriminative ability across the full range of classification thresholds. The prioritization of recall (90.2%) over precision (88.7%) reflects the safety-critical context: failing to predict a high-risk zone (false negative) is operationally more harmful than unnecessarily penalizing a safe road (false positive). SHAP (SHapley Additive exPlanations) feature importance analysis reveals that hour-of-day, traffic density decile, and GPS location cluster are the three most influential predictors, accounting collectively for 61% of model output variance.

### *B. Congestion Prediction Accuracy*

*TABLE III— Congestion Prediction Accuracy vs. Baseline Methods*

| Method | MAE (km/h) | RMSE (km/h) | MAPE (%) |
|---|---|---|---|
| Historical Average | 8.41 | 11.23 | 21.4 |
| ARIMA | 5.93 | 8.17 | 15.6 |
| LSTM [9] | 3.18 | 4.72 | 8.9 |
| Proposed (XGBoost Regressor) | 2.87 | 4.11 | 7.3 |

The proposed gradient boosting regressor achieves a MAPE of 7.3%, outperforming the LSTM baseline [9] (8.9%) despite lower model complexity and significantly faster inference. The advantage over LSTM is attributed to the explicit temporal feature encoding (cyclic hour, day-of-week) combined with the TomTom rolling average speed feature, which captures short- term momentum more effectively than the LSTM's sequential memory across the 15-minute update window. The lower RMSE (4.11 km/h vs 4.72 km/h for LSTM) indicates reduced error variance, important for consistent route scoring.

### *C. Route Optimization Performance*

Route optimization performance was evaluated over 1,200 simulated trips sampled uniformly across the Chennai road network, covering residential, commercial, and arterial origin-destination pairs under three congestion scenarios: free-flow (06:00–08:00), moderate (10:00–16:00), and peak congestion (08:00–10:00, 17:00–20:00).

*TABLE IV— Route Optimization Comparison (Average over 1,200 Trips)*

| Method | Avg. Time (min) | Safety Score (/100) | Flood Avoid. (%) | Multi-Modal |
|---|---|---|---|---|
| Dijkstra (Distance Only) | 28.4 | 54.3 | 62.1 | No |
| Google Maps API | 24.1 | 61.8 | 71.4 | Partial |

| | | | | |
|---|---|---|---|---|
| Wang & Wang [8] | 22.8 | 63.2 | N/A | No |
| Proposed— Traffic-Optimized | 21.3 | 72.4 | 88.6 | Yes |
| Proposed— Fastest | 21.3 | 72.4 | 88.6 | Yes |
| Proposed— Safest | 25.6 | 85.7 | 94.2 | Yes |
| Proposed— Shortest | 26.9 | 68.1 | 84.3 | Yes |

The Safest mode achieves a mean safety score of 85.7/100, representing a 31.4% improvement over the distance-only Dijkstra baseline (54.3/100) and a 23.9% improvement over the Wang and Wang [8] single-objective baseline (63.2/100). Flood-zone avoidance of 94.2% is critical for monsoon-period navigation safety. The 4.3-minute travel time premium of the Safest over Fastest mode (25.6 vs. 21.3 minutes average) is an operationally acceptable trade-off given the safety and flood risk context. The Traffic-Optimized mode delivers the best time performance (21.3 minutes) while maintaining a substantially higher safety score (72.4/100) than both Dijkstra and Google Maps baselines.

*[ FIGURE 2: Safety Score vs. Travel Time Trade-off Curve ]*

*Fig. 2. Safety-time Pareto frontier for the four route modes under moderate congestion conditions.*

*[ FIGURE 3: Flood-Aware Route Map — Chennai Road Network ]*

*Fig. 3. Flood-aware route selection during simulated monsoon event: severity-3 segments (red) excluded, severity-2 (orange) penalized, safe route shown in blue.*

### *D. System Latency Performance*

***TABLE V — API Response Latency (Single-User Load)***

| Operation | Avg. Latency (ms) | 95th Percentile (ms) | 99th Percentile (ms) |
|---|---|---|---|
| Graph Shortest Path (Dijkstra, one mode) | 87 | 142 | 198 |
| ML Risk Inference (all edges, batch) | 34 | 58 | 79 |
| Full Route Computation (4 modes) | 312 | 491 | 643 |
| Multi-Modal Planning (metro + | 445 | 703 | 891 |

| | | | |
|---|---|---|---|
| bus + drive) | | | |
| Live Traffic Data Fetch (TomTom API) | 210 | 380 | 512 |
| Flood Zone Check (rain gauge + polygon query) | 67 | 112 | 145 |

All route computations complete within 500 ms at the 95th percentile under single-user load, meeting the interactive navigation latency target of ≤1,000 ms established by prior UX research on acceptable map application response times. Multi-modal planning, the most complex operation, completes within 703 ms at the 95th percentile, remaining within usability bounds. Latency profiling identifies the TomTom API call as the primary contributor to tail latency; this is mitigated through a 30- second cache layer that serves the majority of requests from memory.

### *E. User Experience and Interface Evaluation*

A pilot usability evaluation was conducted with 32 participants (18 regular Chennai commuters, 9 delivery professionals, 5 emergency responders) using a think-aloud protocol and the System Usability Scale (SUS). The mean SUS score of 81.4 (scale:
0–100) places the system in the 'Excellent' usability category. Participants specifically valued the flood layer visualization (rated 4.6/5.0 importance), the real-time safety score overlay (4.4/5.0), and the route comparison dashboard (4.2/5.0). The most- requested improvement was Tamil-language voice guidance, identified as the top priority for future development by 78% of participants.

## VI. DISCUSSION

### *A. Significance of Results*

The 31.4% safety score improvement of the Safest route over the distance-optimal baseline is significant from both engineering and public health perspectives. In Chennai's road safety context—where the city records among the highest road fatality rates in Tamil Nadu [7]—even marginal reductions in exposure to high-risk road segments during daily commutes aggregate to meaningful reductions in accident probability at the population scale. The flood avoidance accuracy of 94.2% is particularly
operationally relevant: during the 2015 Chennai floods, the absence of real-time flood routing contributed directly to vehicles entering inundated roads and to significant loss of life. The proposed system's ability to automatically exclude severity-3 flood segments and reroute through passable corridors directly addresses this documented failure mode.

### *B. Computational and Deployment Considerations*

The sub-500 ms route computation latency is achieved through a combination of in-memory graph caching, batch ML inference, and asynchronous API calls to external data sources. Deployment on a single mid-tier cloud VM (4 vCPUs, 16 GB RAM) supports approximately 30 concurrent users with target latency compliance. For city-scale deployment serving thousands of concurrent users, horizontal scaling via containerization (Docker/Kubernetes) with graph state shared through a distributed cache (Redis) is the recommended architecture, consistent with standard ITS deployment patterns.

### *C. Limitations*

Several limitations of the current system warrant acknowledgment. First, the accuracy of accident risk

predictions is bounded by the completeness and accuracy of historical TNPD/MoRTH incident records, which may exhibit under-reporting bias for minor incidents. Second, the XGBoost model requires periodic retraining as city development, road modifications, and demographic changes alter accident risk distributions. Third, the current implementation has been validated under simulated trip conditions; large-scale real-world validation with live user navigation data would further strengthen the performance claims. Fourth, TomTom API dependency introduces a cost and availability constraint for city-scale deployment; integration of open traffic data sources (e.g., OpenStreetMap speed profiles) would improve sustainability. Fifth, rural and peri-urban areas within Greater Chennai are less well-represented in both the OpenStreetMap road network and historical incident data, potentially reducing model accuracy in these zones.

## VII. CONCLUSION

This paper presented a comprehensive AI-powered Smart Navigation System for urban route optimization addressing multi- dimensional mobility challenges in Chennai, India. The system integrates multi-objective graph routing, XGBoost-based accident risk prediction, real-time traffic congestion modeling, flood-zone awareness calibrated to Chennai's monsoon hydrology, and full multi-modal transportation planning in a unified full-stack architecture validated on the real-world OpenStreetMap Chennai road network.

Experimental results demonstrate: a route safety score of 85.7/100 (31.4% improvement over shortest-path baseline); flood- zone avoidance accuracy of 94.2%; congestion prediction MAPE of 7.3% (outperforming LSTM); multi-modal planning latency of 703 ms at the 95th percentile; and a System Usability Scale score of 81.4 ('Excellent'). The rolling re-optimization mechanism, adapted from predictive control theory [8], ensures adaptive route decisions throughout each journey, dynamically responding to real-time speed deviations and incident updates.

The proposed system represents a meaningful advance beyond existing navigation platforms for complex urban environments characterized by compound hazards— specifically the intersection of high traffic density, accident risk, and seasonal flooding endemic to Indian coastal megacities. The modular, open architecture is directly transferable to other flood-prone Indian cities including Mumbai, Kochi, and Kolkata with appropriate local dataset substitution.

Future work will investigate: (i) Graph Neural Network (GNN)-based spatial traffic prediction exploiting road network topology for improved accuracy; (ii) edge computing deployment to reduce cloud API dependency and improve performance in low-connectivity zones; (iii) IoT integration with Chennai's emerging smart traffic signal infrastructure; (iv) Tamil-language voice navigation addressing the top user experience gap identified in pilot evaluation; and (v) adversarial robustness testing of the ML models against sensor spoofing and GPS drift artifacts.

## ACKNOWLEDGMENT

The authors express their sincere gratitude to their guide, **MS. R Gayathri,** for her invaluable guidance, continuous support and encouragement throughout this research work. The authors thank the Greater Chennai Corporation (GCC) and the Tamil Nadu State Disaster Management Authority (SDMA) for providing geospatial flood vulnerability data and rain gauge API

access. The authors also thank the Ministry of Road Transport and Highways (MoRTH) for historical accident records, and the anonymous reviewers for their constructive feedback which substantially improved this manuscript.

## REFERENCES

[1] E. I. Vlahogianni, M. G. Karlaftis, and J. C. Golias, "Short-term traffic forecasting: Where we are and where we're going," Transp. Res. C, Emerg. Technol., vol. 43, pp. 3–19, Jun. 2014.

[2] F. Zhu et al., "Parallel transportation systems: Toward IoT-enabled smart urban traffic control and management," IEEE Transport. Intelligent. Transport. System., vol. 21, no. 10, pp. 4063–4071, Oct. 2020.

[3] H. L. Han, "Research and application of shortest path algorithm based on urban road network," M.S. thesis, North Univ. China, Taiyuan, China, 2017.

[4] J.-Y. An et al., "A novel fuzzy-based convolutional neural network method to traffic flow prediction with uncertain traffic accident information," IEEE Access, vol. 7, pp. 20708–20722, 2019.

[5] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining, San Francisco, CA, Aug. 2016, pp. 785–794.

[6] S. B. Adinarayana, B. R. Raju, and P. K. Srivastava, "Assessment of urban flood vulnerability using GIS in Chennai city, India," Nat. Hazards, vol. 84, no. 2, pp. 1029–1050, Nov. 2016.

[7] Road Accidents in India 2022, Ministry of Road Transport and Highways (MoRTH), Govt. of India, New Delhi, 2023.

[8] Z. Wang and S. Wang, "Real-time dynamic route optimization based on predictive control principle," IEEE Access, vol. 10, pp. 55062–55072, May 2022.

[9] Z. Zhao, W. Chen, X. Wu, P. C. Y. Chen, and J. Liu, "LSTM network: A deep learning approach for short-term traffic forecast," IET Intell. Transp. Syst., vol. 11, no. 2, pp. 68–75, 2017.

[10] X. Ma et al., "Learning traffic as images: A deep convolutional neural network for large-scale transportation network speed prediction," Sensors, vol. 17, no. 4, pp. 818–833, 2017.

[11] Z. Zhou, B. De Schutter, S. Lin, and Y. Xi, "Two-level hierarchical model-based predictive control for large-scale urban traffic networks," IEEE Trans. Control Syst. Technol., vol. 25, no. 2, pp. 496–508, Mar. 2017.

[12] S. Rahman et al., "Adaptive route selection support system based on road traffic information," in Proc. ICEEICT, May 2015, pp. 1–6.

[13] Z. Duan et al., "Improved deep hybrid networks for urban traffic flow prediction using trajectory data," IEEE Access, vol. 6, pp. 31820–31827, 2018.

[14] L. Zhi, X. Zhou, and J. Zhao, "Vehicle routing for dynamic road network based on travel time reliability," IEEE Access, vol. 8, pp. 190596–190604, 2020.

[15] A. Khanda et al., "Efficient route selection for drone-based delivery under time-varying dynamics," in Proc. IEEE 18th Int. Conf. MASS, Oct. 2021, pp. 437–445.