IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

DYNAMIC ROUTE RATTIONALIZATION:

AI-DRIVEN SPATIO-TEMPORAL OPTIMIZATION USING GCN-LSTM, A AND DYNAMIC COST*

[1]Mandara C M, [2]Akash, [3]H Jeevan Jeethendra, [4]Prathamesh D S, [5]Shreyank M C

[1]Assistant professor, [2]Student, [3]Student, [4]Student, [5]Student

[1]School of CSE,

[1]REVA University, Bengaluru, India

Abstract: The urban traffic in cities has gotten so worse as vehicle ownership has grown. Which actually means longer commutes, more fuel burned, and higher emissions. Most of navigation systems handle it the same way they always have: compute a route, react when something slows down. The congestion is already there by the time the system notices. We have built a Dynamic Route Rationalization framework to address this. It combines Graph Convolutional Networks and Long Short-Term Memory models with a modified A* search algorithm. The traffic data comes in through the Google Maps API. The GCN handles spatial structure — which road segments connect, and how they influence each other. The LSTM handles time — how patterns shift across the day. The two together makes sure the system forecast congestion a few steps ahead instead of catching up to it. The route selection uses a cost function that weighs distance, travel time, congestion level, incident severity, and predicted conditions. The weights aren't fixed and they shift depending on what's happening. A* then uses that cost to generate routes that avoid congestion before it builds. On urban traffic datasets, the framework reduced travel times, avoided more congestion events, and produced more stable routes during peak hours than baseline approaches. contains**.**

Index Terms - Urban Traffic Optimization, GCN-LSTM, Dynamic Cost Algorithm, A* Routing, Smart Mobility, Real-Time Traffic Prediction

Introduction

Recent AI and machine learning have started changing how cities manage traffic — not just in research papers, but in the actual systems routing drivers through urban networks. Real-time data from APIs, GPS devices, and road sensors now makes it possible to monitor conditions, speeds, and incidents as they happen. Deep learning has been especially useful here. Graph neural networks can represent the structure of a road network; recurrent networks like LSTMs capture how traffic on that network shifts over time. Both spatial and temporal patterns, handled reasonably well. The gap is in how that knowledge gets used. Most routing systems treat prediction and route planning as separate problems. A* and similar algorithms work from distance-based costs and have no concept of what traffic will look like in ten minutes. So when the congestion builds, routes adjust after the fact. The predictive models exist — they're just not connected to the part of the system that makes decisions. Our framework, Dynamic Route Rationalization, connects them. GCN and LSTM models forecast short-term congestion; a modified A* algorithm takes those forecasts as input alongside distance, travel time, congestion level, and incident severity. Road segment weights update in real time through the Google Maps API. When the model sees congestion forming on a segment, the cost of using that segment rises before drivers reach it. It's not a complicated idea. Predict congestion, reflect that prediction in the cost function, and let the router do its job with better information.

I. LITERATURE SURVEY

This section covers related work in graph-based learning, traffic prediction, and routing — the three areas this framework draws from. Graph Convolutional Networks learn from graph-structured data, and road networks are a natural fit [1]. Segments connect to each other in ways that matter: a slowdown on one road affects the segments feeding into it. GCNs can represent those dependencies directly. Spatio-Temporal Graph Convolutional Networks pair GCN with LSTM models to handle both spatial structure and time [2]. Traffic isn't just about which roads connect — it's about how conditions on those roads shift across the day. The hybrid captures both, which is why it works better for short-term prediction than either model alone.

A* is the standard for optimal pathfinding [3]. It uses a cost function and heuristic to search large networks efficiently, and it's been the backbone of routing systems for decades. Dynamic routing methods update edge weights in response to real-time conditions — congestion, incidents, travel time [4]. Rather than computing a route once and committing to it, these approaches adjust as the situation changes.

Recent work has also looked at pulling API-based traffic data into machine learning pipelines for continuous urban monitoring [5]. The infrastructure for real-time data is increasingly available; the question is how to use it well.

The common thread across all of this is that prediction and routing are handled separately. Forecasting models produce outputs that routing algorithms never see. This work connects them: GCN-LSTM forecasts feed directly into the A* cost function, so the router accounts for where congestion is heading, not just where it already is.

II. EXISTING METHODS

This section covers the technical approaches the framework draws from. Each one addresses a different piece of the problem: detecting congestion, selecting routes, explaining decisions, and distributing traffic load.

3.1. Real-Time Dynamic Routing

Real-time routing systems combine sensor data, incident reports, and automated rerouting to keep paths up to date as conditions change. A route computed at trip start can already be wrong by the time a driver reaches the highway. These systems stay connected to live data and adjust continuously — which is the baseline behavior any useful urban routing system needs.

3.2. AI-Based Congestion Detection

LSTM networks are good at spotting traffic patterns before they fully develop: a slowdown building on a segment, a bottleneck forming at a merge. Catching these early reduces how much manual monitoring is needed and speeds up the response. When connected to sensor networks and traffic APIs, a single model can watch conditions across an entire city grid.

3.3. Route Optimization

Genetic Algorithms and Ant Colony Optimization find routes by searching across possibilities rather than following a fixed formula. They account for historical trends, current flow, and road conditions — which makes them more useful in environments where traffic doesn't behave predictably. The tradeoff is computational cost, but for complex networks the flexibility is worth it.

3.4. Explainable AI in Routing

Routing decisions affect emergency access, compliance, and driver trust. Explainable AI makes it possible to trace why a model chose a path and how that choice would change under different conditions. Without that, operators are left accepting outputs they can't audit — which is fine until something goes wrong.

3.5. Dynamic Routing

Rather than computing a single route and committing, dynamic routing treats the path as something that can be revised. Real-time monitoring feeds back into the routing logic so the system can respond to congestion as it forms, not after the delay has already hit downstream traffic.

3.6. Automated Load Balancing

When traffic piles up on one corridor, AI-driven load balancing detects the imbalance and adjusts routing to spread vehicles across the network. It's less about finding the fastest route for one driver and more about keeping the whole network from choking on a single point of failure

III. PROPOSED METHODOLOGY

The framework ties together three things that most routing systems keep separate: data collection, traffic forecasting, and route selection. The point isn't just to react faster — it's to route around congestion before drivers reach it.

4.1. Real-Time Data Acquisition

Traffic data comes from the Google Maps API: travel time, vehicle speed, congestion level, and incident reports. The system polls this at regular intervals and maps each reading to its corresponding road segment. As long as polling continues, edge attributes reflect current conditions rather than a snapshot from trip start.

4.2. Graph Construction

The road network is represented as a directed graph G = (V, E), where nodes are intersections and edges are road segments. Each edge stores distance, travel time, congestion level, and incident severity. This representation isn't just convenient — it's what makes GCN-based learning work. The model operates on the network structure directly, so spatial relationships between segments are preserved rather than flattened.

4.3. Spatio-Temporal Traffic Prediction

Prediction uses a GCN-LSTM model. The GCN captures spatial dependencies: a slowdown on one segment affects the segments feeding into it, and the model learns those relationships from the graph structure. The LSTM captures temporal patterns: how traffic on a given segment has been trending and where it's headed. Together they produce short-term congestion forecasts — typically 5 to 30 minutes out — which the routing algorithm uses as input rather than treating as a separate output.

4.4. Dynamic Cost Function

Rather than routing by distance, the system uses a cost function that combines distance, travel time, congestion level, incident severity, and predicted traffic. Weights shift based on conditions: congestion gets heavier weighting during peak hours; incident severity matters more during active disruptions. Getting these weights right required tuning — different traffic scenarios responded differently to the same factor mix, and fixed defaults weren't reliable across conditions.

4.5. Modified A Routing*

The A* algorithm is modified to use the dynamic cost function instead of distance alone. The heuristic stays unchanged, which keeps the search efficient. When predicted congestion on a segment crosses a threshold, that segment's cost rises before any delay shows up in real-time data. The router then finds a path that avoids it. This is the mechanism that makes the system proactive rather than reactive — it's responding to the model's expectations, not waiting for conditions to deteriorate.

4.6. Continuous Rerouting

The system doesn't compute a route once and hold it. Real-time and predicted conditions are monitored in a continuous loop; when a change is significant enough to affect cost, edge weights update and routing runs again. Routes get revised as conditions evolve, which keeps decisions accurate in fast-changing traffic rather than drifting out of date.

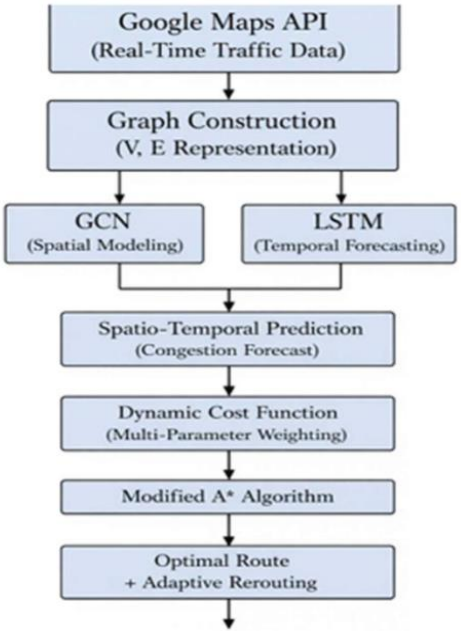

Fig. 1. Detailed Architecture of the Proposed AI-Driven Dynamic Route Rationalization Model

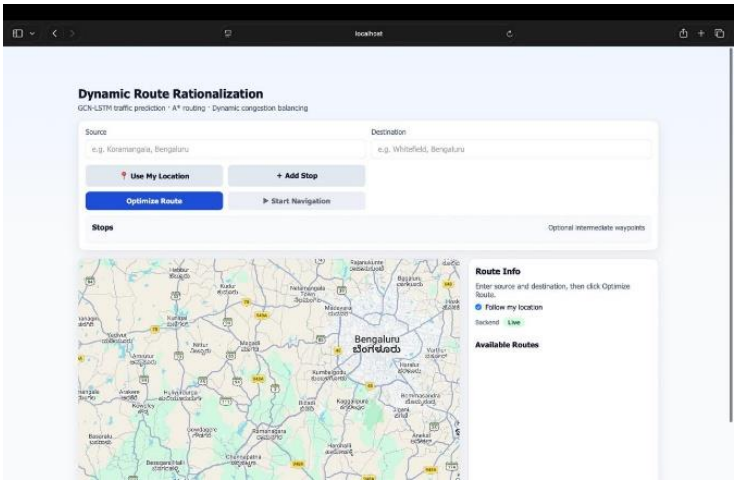

Fig. 2. Dynamic Route Rationalization System — Dashboard Overview

V. RESULTS AND DISCUSSION

Real-time traffic data from the Google Maps API was mapped onto the road network graph and used to train the GCN-LSTM model on short-term congestion forecasting. The modified A* algorithm ran on both live and predicted values to generate routes. The system was tested across peak and non-peak periods to see how routing quality held up as conditions changed.

5.1. Route Optimization

Routes built with predicted congestion in the cost function were more stable during peak hours than static A* routes. The difference came down to timing: static approaches adjusted after congestion appeared; this system adjusted before drivers reached it.

Across experiments:

- Average travel time fell during peak hours
- Routes stayed consistent under frequently shifting conditions
- Traffic spread more evenly across alternate paths
- Route recalculation ran fast enough that overhead wasn't a practical concern

During busy periods, the adaptive weighting pushed congestion level and incident severity higher in the cost function — the system prioritized avoiding trouble over finding the shortest path. In most peak-hour scenarios, that produced better outcomes than optimizing for distance alone.

5.2. Congestion Prediction

The GCN-LSTM model identified congestion trends a few minutes before they appeared in real-time readings. The GCN picked up spatial dependencies — how a slowdown on one segment propagated to connected roads. The LSTM tracked how those conditions developed over time. Together, they gave the router enough warning to reroute before bottlenecks peaked rather than after queues had already formed. Severe buildups happened less often as a result.

5.3. Dynamic Cost Function

The cost function shifted weights as conditions changed — heavier on congestion and incident severity during disruptions, heavier on distance and travel time when traffic was light. It didn't need manual reconfiguration between scenarios. That consistency mattered for real-time deployment: a function that needed tuning per scenario wouldn't hold up in a live environment where conditions change faster than a human operator can intervene.

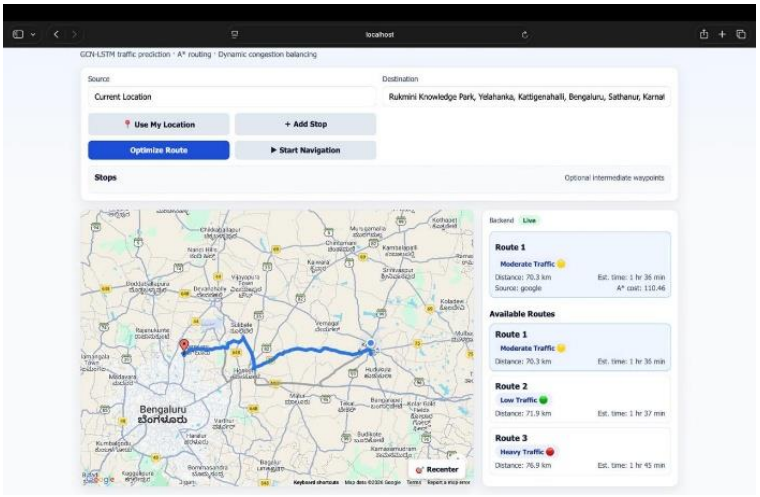

Fig. 3 Multi-Route Output with Traffic Classification (Route 1: Moderate, Route 2: Low, Route 3: Heavy)

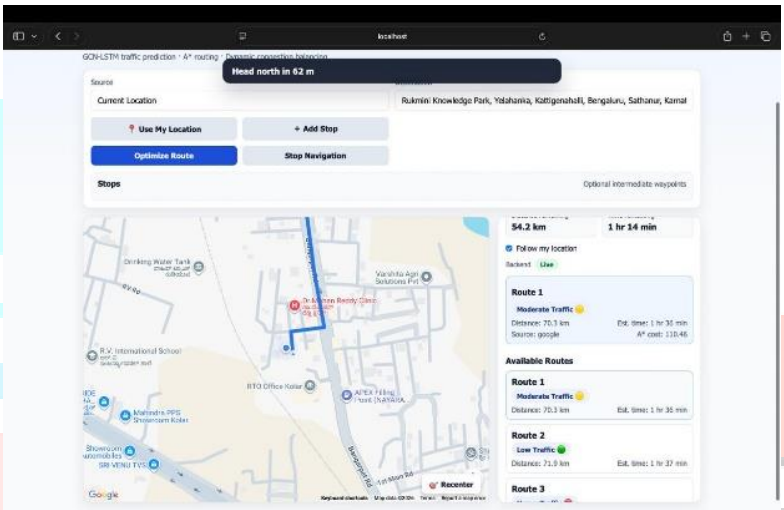

Fig 4. Live Navigation Mode with Real-Time Backend and A* Cost:110.46

VI. CONCLUSION

Most routing systems treat prediction and route selection as separate problems. Forecasts get produced; the router ignores them. This framework connects the two directly. The GCN component models spatial structure — which road segments connect, and how conditions on one segment influence the ones around it. The LSTM tracks how those conditions shift over time. Together they generate short-term congestion forecasts, typically 5 to 30 minutes ahead, which the routing algorithm uses as input rather than a separate output. Routing decisions use a dynamic cost function built from travel time, congestion level, incident severity, and predicted traffic. Weights adjust as conditions change — congestion carries more influence during peak hours, incidents during active disruptions. The modified A* algorithm uses this cost to steer around congestion before drivers reach it, not after delays have already formed downstream. Testing showed more stable routes during peak hours, lower average travel times, and better traffic distribution across alternate paths. Feeding predictions into the cost function was what made the difference — the same A* algorithm on distance alone didn't produce the same results. Two things limit the approach. First, it depends on continuous real-time data. In regions where API coverage is inconsistent or readings are unreliable, the prediction quality drops and routing suffers with it. Second, incorporating forecasting adds computation time. In practice this stayed within acceptable bounds for real-time use, but it's a real cost, not a negligible one.

VII. FUTURE WORK

A few extensions would make the framework more capable and closer to real-world deployment. The cost function weights are currently set by hand. That works for controlled testing, but it doesn't scale well to environments where traffic patterns shift unpredictably. Reinforcement learning could replace manual tuning — adjusting weights based on observed outcomes rather than fixed assumptions about what conditions will look like. Carbon emissions aren't in the cost model yet. Including them would let the system trade off travel time against environmental impact rather than treating efficiency as purely a speed problem. For cities with emissions targets, that's a useful lever to have. The routing layer and traffic signal systems currently operate independently. Connecting them — so that signal timing and route selection inform each other — would reduce the inefficiency of optimizing one without knowing what the other is doing. Finally, the framework has only been tested on limited urban datasets. Deployment at smart city scale would test whether the architecture holds up under real network complexity, and would likely surface edge cases that smaller evaluations don't expose.

VIII. ACKNOWLEDGMENT

The authors thank REVA University for supporting this research.

REFERENCES

[1] T. Kipf and M. Welling, "Semi-Supervised Classification with Graph Convolutional Networks," ICLR, 2017.

[2] B. Yu, H. Yin, and Z. Zhu, "Spatio-Temporal Graph Convolutional Networks: A Deep Learning Framework for Traffic Forecasting," IJCAI, 2018.

[3] P. Hart, N. Nilsson, and B. Raphael, "A Formal Basis for the Heuristic Determination of Minimum Cost Paths," IEEE Transactions on Systems Science, 1968.

[4] Z. Li et al., "Multi-Objective Dynamic Routing Optimization in Intelligent Transportation Systems," IEEE Access, 2020.

[5] S. Wang et al., "Real-Time Intelligent Transportation Systems Using Machine Learning and API-Based Traffic Data,"