Short-Term Traffic Congestion Prediction Using Machine Learning: XGBoost vs. Deep Neural Networks

George S. Theodoropoulos[1], Zoi Nikolarakis[2] and Yannis Theodoridis[1]

[1]University of Piraeus, Karaoli ke Dimitriou 80, Piraeus, 185 34, Greece
[2]University of Muenster, Leonardo-Campus 3, Muenster, 48149, Germany

Abstract

Recurring large-scale traffic congestion is one of the most important challenges that modern urban cities face, necessitating accurate prediction models to enable timely interventions. While Deep Learning approaches such as Graph Convolutional Networks (GCNs) and Long Short-Term Memory networks (LSTMs) have been widely adopted for capturing spatio-temporal dependencies, they are often accompanied by high computational costs and lack interpretability. This paper presents a comprehensive comparison of these deep learning methods against a Gradient Boosting approach (XGBoost) enhanced by systematic feature engineering for short-term traffic congestion prediction, formulated as a binary classification problem. We evaluate the models on two distinct datasets, the public PEMS-BAY and the proprietary Dutch NDW, across prediction horizons ranging up to 1 hour (in 10-minute intervals). Our results demonstrate that the XGBoost-based model significantly outperforms GCN and LSTM in terms of prediction accuracy (F_1 score) while requiring substantially less computational resources, achieving up to 3 orders of magnitude savings in inference time. Additionally, we highlight the inherent explainability of the tree-based approach, which provides actionable insights into congestion propagation patterns, offering a practical and transparent solution for real-world urban traffic management systems.

Keywords

traffic congestion prediction, XGBoost, graph convolutional networks, LSTM, urban traffic management

1. Introduction

Urban traffic congestion has become an increasingly pressing issue in cities around the world, affecting millions of commuters daily and costing economies billions of dollars each year [1]. As cities continue to grow and more people move into urban areas, the problem of gridlock during peak hours has only gotten worse, leading to longer commute times, higher fuel consumption, and increased air pollution. For city planners and traffic engineers, finding ways to predict when and where congestion will occur has become essential for developing effective traffic management strategies. Being able to anticipate traffic jams before they happen allows for timely interventions such as adjusting traffic signal timings, implementing dynamic lane management, or providing real-time navigation alternatives to drivers through mobile applications. However, accurately forecasting congestion remains a challenging task because traffic patterns are influenced by numerous factors including time of day, day of week, weather conditions, special events, and the interconnected nature of road networks where a problem in one location often quickly spreads to others. Figure 1 visually presents such congestion events in an urban traffic network, where a sensor is experiencing congestion (i.e., the average speed is below the congestion threshold that is represented by the red dotted line) while others are not.

In recent years, researchers have explored various machine learning approaches to tackle this complex prediction problem, with many studies focusing on deep learning techniques that can capture both spatial relationships between road segments and temporal patterns in traffic flow [2]. Despite their demonstrated potential, these methods often require substantial computational resources and can be difficult to interpret, making it hard for transportation professionals to understand why certain predictions are made.

Proceedings of the Workshops of the EDBT/ICDT 2026 Joint Conference, March 24–27, 2026, Tampere, Finland
gstheo@unipi.gr (G. S. Theodoropoulos); znikolar@uni-muenster.de (Z. Nikolarakis); ytheod@unipi.gr (Y. Theodoridis)
0000-0003-4547-6646 (G. S. Theodoropoulos); 0009-0001-3414-2117 (Z. Nikolarakis); 0000-0003-2589-7881 (Y. Theodoridis)

Our work adopts a different approach by employing an efficient machine-learning solution (XGBoost) and comparing it with deep learning models, namely Graph Convolutional Networks (GCNs) and Long Short-Term Memory networks (LSTMs), to determine which provides the most effective solution for real-world traffic management applications. In this paper, we present a comprehensive evaluation of these three approaches for short-term traffic congestion prediction, defined as forecasting whether specific road segments will experience significant slowdowns within the next 10–60 minutes. We formalize this as a binary classification problem where congestion is identified based on speed measurements falling below a certain threshold relative to historical patterns at each location. Our experiments on two different datasets show that systematic feature engineering leads to an XGBoost model that is more accurate, efficient and interpretable, turning what would otherwise be a black-box system into a valuable diagnostic tool for improving urban mobility.

The remainder of this paper is structured as follows: Section 2 reviews related work on traffic state prediction, contrasting traditional statistical methods with modern deep learning architectures. Section 3 formally defines the short-term traffic congestion prediction problem as a binary classification task. Section 4 describes the deep learning baselines used in this study, detailing the GCN and LSTM approaches. Section 5 presents the proposed XGBoost-based methodology, emphasizing the systematic five-phase feature engineering pipeline. Section 6 reports the experimental results on the PEMS-BAY and NDW datasets, evaluating prediction accuracy and computational efficiency. Finally, Section 7 concludes the paper and discusses the implications for practical urban traffic management systems.

2. Related Work

Early efforts to understand and mitigate the problems that urban traffic congestion introduces focused on broad conceptual frameworks, as seen in Afrin & Yodo's [3] comprehensive survey. They emphasized the need for holistic, sustainable solutions that integrate infrastructure, traffic management, and policy, highlighting how data-driven approaches and smart mobility concepts are becoming essential for building resilient transportation systems.

Alghamdi et al. [4] demonstrated the utility of ARIMA models for short-term traffic forecasting, showing they can effectively capture linear, stationary patterns in historical traffic data with reasonable computational efficiency. However, they also noted a key limitation: these models often struggle with the complex, non-linear dynamics inherent in real-world traffic flow. This observation spurred significant interest in machine learning and deep learning techniques, which could better handle the spatio-temporal complexity of urban networks.

Early deep learning approaches focused on modeling temporal dependencies. Liu et al. [5] were among the first to successfully integrate spatial awareness directly into temporal modeling with their Conv-LSTM architecture. By embedding convolutional operations within LSTM units, their model learned spatial correlations alongside temporal dynamics, significantly outperforming standard LSTMs and traditional time-series methods for short-term traffic flow prediction. This established a crucial principle: effective traffic prediction requires joint modeling of space and time.

Subsequent research refined this spatio-temporal modeling. Zhao et al. [6] took a significant step forward by introducing the Temporal Graph Convolutional Network (T-GCN), which explicitly leveraged the underlying road network topology using GCNs combined with Gated Recurrent Units (GRUs). This approach proved highly effective, outperforming both statistical models and earlier deep learning baselines by more accurately representing how traffic states propagate through connected road segments.

Further advancements focused on optimizing the representation of traffic data. Guo et al. [7] developed an Optimized Graph Convolutional Recurrent Neural Network (OGCRNN), incorporating an optimization strategy to improve feature propagation and reduce noise within the graph structure, leading to even higher prediction accuracy. Ranjan et al. [8] combined CNN, LSTM, and Transpose CNN layers to generate high-resolution congestion maps for city-wide networks, demonstrating strong performance for large-scale real-time monitoring. Nagy & Simon [9] made a particularly important contribution by emphasizing the spatial propagation of congestion itself; they showed that explicitly

modeling how congestion spreads across the network, rather than just predicting isolated points, significantly boosts forecasting accuracy. This highlighted the need for models to understand traffic as a connected system where a jam in one area directly impacts neighboring segments.

A critical challenge in applying these models to congestion prediction specifically—often framed as a binary classification problem (congested vs. non-congested)—is the inherent class imbalance, as congestion events are typically rare (e.g., less than 15–20% of observations). Chen et al. [10] directly addressed this with their Periodic Convolutional Neural Network (PCNN). They explicitly modeled traffic's strong daily and weekly periodicity using a time-series folding technique to create periodic input matrices, making the model robust to non-stationary patterns. Crucially, PCNN treated congestion as a binary state defined by a speed threshold (e.g., below 70% of historical baselines), similar to many operational definitions. By prioritizing recall for congested states through weighted loss functions and processing multi-grained temporal windows, PCNN achieved strong F_1-scores (87.3%) for short-term (30-min) congestion detection on NYC data, outperforming LSTMs and GCNs.

More recently, research has increasingly tackled the practical constraint of sparse sensor coverage, a common reality where only a fraction of road segments have fixed sensors. Li et al. [11] proposed a Multi-Task Graph Neural Network (MT-GNN) specifically designed for this scenario. Their key insight was using multi-task learning: the model jointly predicted continuous speed (the auxiliary task) and binary congestion states (the primary task). This approach enriched the features used for congestion classification, helping overcome the data scarcity at any single point.

Liu et al. [12] further advanced solutions for partial sensing with their Spatio-Temporal Partial Sensing (STPS) framework. Recognizing that unsensed locations exhibit different traffic distributions than observed ones, they introduced a learned spatial transfer matrix based on graph attention. This matrix quantifies how congestion propagates from sensed to unsensed segments, effectively modeling spatial dependencies even with incomplete data. To handle non-stationarity (like sudden incidents), they used rank-based speed features relative to historical distributions (e.g., "slower than 95% of past observations") instead of raw speeds.

The importance of carefully selecting evaluation metrics for these tasks cannot be overstated. Naidu, Zuva, & Sibanda [13] provided a vital systematic review, stressing that inappropriate metrics (e.g., relying solely on overall accuracy for imbalanced congestion data) can lead to misleading conclusions. They underscored the need to align metrics like F_1-score or precision-recall curves with the specific problem objectives and data characteristics, which is essential for fairly comparing models like PCNN, MT-GNN, and STPS.

3. Problem Definition

This work addresses the critical challenge of **short-term traffic congestion prediction** in urban road networks, specifically formulated as a spatio-temporal binary classification problem. Given an urban network of fixed-point traffic sensors monitoring real-time speed measurements, we define congestion at location s and time t as a significant deviation from typical traffic conditions, formalised through the following condition:

$$\mathcal{C}(s,t) = \begin{cases} 1 & \text{if } v(s,t) < \tau \cdot \text{quantile}_\alpha(v(s,\cdot)) \\ 0 & \text{otherwise} \end{cases}$$

where $v(s,t)$ denotes the traffic speed at sensor s, τ is the congestion severity threshold, α represents the high-speed quantile reference, and $\text{quantile}_\alpha(v(s,\cdot))$ is the α-quantile of historical speed distribution for sensor s. Without loss of generality and according to domain experts' feedback, we set $\tau = 0.65$ and $\alpha = 0.9$, which, in fact, considers acute congestion events as speed reductions exceeding 35% of the typical (90th percentile) flow conditions at each location, as shown in the sample time series in Figure 1.

Regarding the prediction horizon, the objective of this work is to forecast future congestion states $\mathcal{C}(s, t_{now} + h)$ for arbitrary sensor locations s and prediction horizons h, where $h = 10, 20, \ldots, 60$

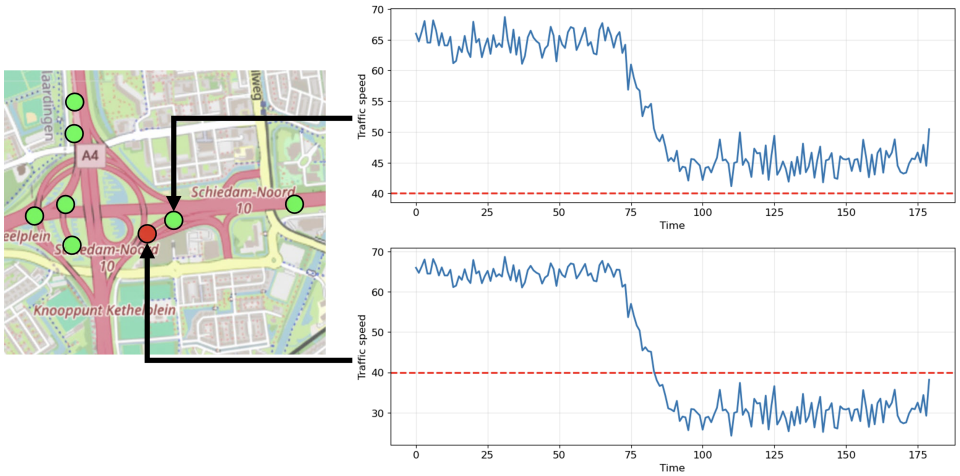

Figure 1: An example of congestion in a road network.

minutes. This requires modeling the complex spatio-temporal dynamics of traffic flow, where congestion emerges through spatial propagation, temporal patterns, non-stationarity, and class imbalance (congestion events typically constitute less than 15% of observations).

The above formulation extends beyond conventional traffic state prediction (which typically estimates continuous speed or flow values) by explicitly framing congestion as a binary classification problem with an *operational threshold* that directly corresponds to traffic management interventions. This binary formulation enables direct alignment with real-world decision-making: when congestion is predicted with sufficient confidence, traffic engineers can implement specific interventions such as dynamic signal timing, ramp metering, or incident response deployment.

4. Deep Learning Models for Traffic Congestion Prediction

The sections that follow describe two distinct deep learning modeling strategies that are used to forecast congestion at individual sensor locations. The first strategy employs a GCN, which first aggregates information from neighboring sensors based on their spatial proximity and then applies a temporal processing step to capture the dynamics of traffic flow. The second strategy builds on this spatial foundation by adding a recurrent LSTM architecture that further refines the representation of traffic over successive time steps. Together, these approaches illustrate the effectiveness of deep learning models in predicting congestion across different horizon values.

4.1. The GCN Approach

The GCN model is specifically engineered to leverage the spatial relationships between traffic sensors while capturing temporal traffic patterns. The methodology exploits both dynamic traffic measurements and static geographical information through a specialized graph convolution layer that incorporates the precomputed weighted adjacency matrix. This design allows the model to identify how traffic conditions propagate through the road network based on physical proximity between sensors.

The network structure begins with the input layer that receives the spatiotemporal feature matrix, followed by the graph convolution operation that applies spatial filtering across the sensor network. The output from the graph convolution layer is then processed through feature aggregation mechanisms that combine information across both spatial and temporal dimensions before passing to the final prediction layers. This formulation conceptually follows the proposed GCN architecture of Yu et al. [14].

Spatial Dependency Modeling The spatial relationships between sensors are encoded in a weighted adjacency matrix derived from geographical distances between sensor locations. This matrix is trans-

formed using a Gaussian Radial Basis Function (RBF) kernel to emphasize closer relationships while diminishing the influence of distant sensors. The kernel function is defined as $A_{ij} = \exp(-d_{ij}^2/\sigma^2)$, where d_{ij} represents the geographical distance between sensors i and j, and σ is a scaling parameter.

The adjacency matrix is row-normalized to create a weighted representation where the influence of neighboring sensors is proportional to their spatial proximity. This normalized matrix is then used in the graph convolution operation to ensure that each sensor's representation is updated based on its immediate neighbors with appropriate weighting, effectively modeling traffic flow propagation across the network.

Feature Processing The model incorporates various feature types processed through the graph convolution layer. Dynamic features include current speed measurements for each sensor and neighbor-aggregated speed values weighted by spatial distance. The temporal component is captured through cyclical features (sin/cos transforms) for minute-of-day and day-of-week, which encode the periodic nature of traffic patterns.

Static geographical coordinates (latitude and longitude) are processed separately and then fused with the dynamic features after graph convolution. This fusion occurs at a dedicated feature combination layer, allowing the model to leverage both the spatial relationships captured through the graph structure and the absolute geographical positioning of sensors.

Model Training The GCN model is trained using binary cross-entropy loss, with early stopping implemented to prevent overfitting. During training, the model processes fixed-length historical windows of traffic data to predict congestion states at multiple future time horizons. The training procedure maintains the chronological order of data, with the 80-10-10 split ensuring that no future information leaks into the training process.

The model is trained independently for each sensor location, allowing it to capture the unique traffic patterns at each specific location. The spatial relationships encoded in the adjacency matrix enable the model to leverage information from neighboring sensors while maintaining sensor-specific prediction capabilities.

Decision Threshold Optimisation Since the classifier outputs a vector of estimated congestion probabilities, an appropriate decision threshold must be selected. This threshold is identified by computing the F_1 score on the test set for each candidate threshold and choosing the value that maximizes the F_1 metric. This process results in a model that exhibits greater robustness and generalization across unseen data, while also striking a balance between precision and recall scores.

4.2. The LSTM Approach

The LSTM approach that draws inspiration from the work present in Zhao et al. [6] extends the spatial modeling foundation established by the GCN framework with enhanced temporal processing capabilities. While leveraging the same graph-based spatial dependency modeling as the GCN (detailed in Section 4.1), this architecture introduces sequential temporal processing to capture the dynamic evolution of traffic patterns. The hybrid design processes spatial relationships through graph convolution before feeding spatially-enhanced representations into an LSTM layer, creating a unified spatiotemporal model.

Spatial-Temporal Integration The spatial dependency modeling follows the GCN implementation described above, utilizing the same distance-based adjacency matrix transformed via Gaussian RBF kernel and row-normalized for spatial weighting. However, the LSTM architecture introduces a critical modification: the graph convolution layer processes input features at each timestamp independently, generating spatially-aware representations that are then sequentially fed into the LSTM. This two-stage processing (spatial $\rightarrow$ temporal) enables the model to first contextualize traffic conditions within the road network before analyzing their temporal evolution, effectively capturing how congestion propagates through both space and time.

Sequence Processing The LSTM layer processes the sequence of spatially-enhanced representations from historical time windows, maintaining a hidden state that evolves with each time step to capture temporal dependencies. Unlike the GCN's static spatial aggregation, the LSTM explicitly models the progression of traffic states through its memory cells, making it particularly effective at capturing congestion formation patterns, propagation speeds, and dissipation dynamics. The model processes fixed-length historical sequences to predict congestion states at multiple future horizons, with its stateful architecture allowing it to retain relevant information from earlier time steps while discarding transient fluctuations.

Model Training Training follows the same fundamental protocol as the GCN approach (binary cross-entropy loss, early stopping, and chronological 80-10-10 data split), but with sequence-specific adaptations. The model processes sliding windows of historical traffic data where each window contains multiple time steps of spatially-processed features. Speed features are normalized using the same StandardScaler methodology described in Section 4.1, ensuring consistent scaling across sensors and time periods. While maintaining per-sensor training to capture location-specific patterns, the LSTM benefits from the spatial information propagated through the graph convolution layer, creating a balanced approach that leverages both local sensor characteristics and network-wide traffic dynamics.

Decision Threshold Optimisation The optimal decision threshold is identified in the same way as with GCN, ensuring robustness and comparability between the two methods.

5. XGBoost for Traffic Congestion Prediction

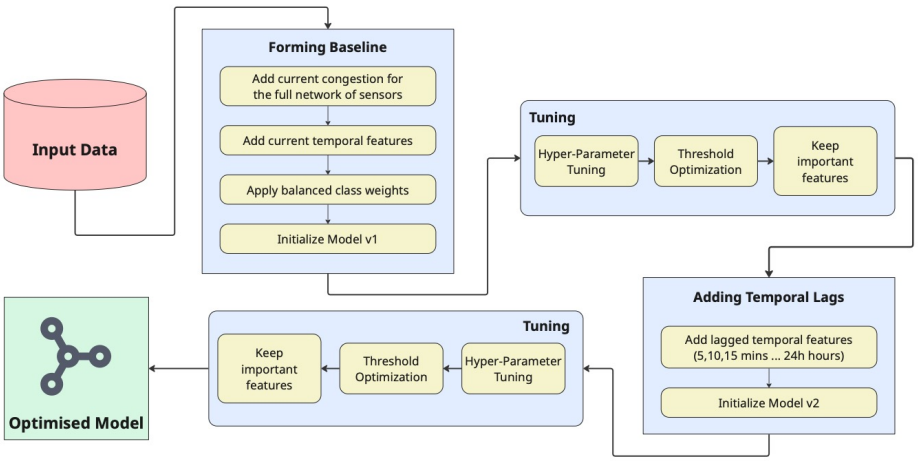

Figure 2: The architecture of the proposed XGBoost-based congestion prediction methodology.

In this section, we present our XGBoost-based methodology for congestion prediction. Unlike the GCN and LSTM approaches (detailed in Sections 4.1 and 4.2, respectively), which inherently capture spatial dependencies through graph-based architectures and temporal dynamics via sequence modeling, this tree-based approach relies on explicit feature engineering to encode spatiotemporal patterns. While deep learning models automatically learn representations from raw sensor data, XGBoost requires careful construction of lag features and interaction terms to model traffic propagation effects—highlighting a fundamental methodological contrast between representation learning and feature engineering paradigms.

For the XGBoost model, we implement a systematic five-phase ablation methodology designed to iteratively refine feature engineering and model optimization. This pipeline consists of a sequential series of feature-engineering and hyper-parameter tuning steps that progressively narrow the predictor

from all sensors toward a minimal but high-performing configuration. Each stage builds on the results of the previous one, resulting in a final model that provides the most accurate predictions. A visual illustration of this workflow is presented in Figure 2.

Phase 1: Baseline Model Construction The first phase establishes a reference model using all available sensor locations as features, augmented with 5 basic temporal indicators, namely day, hour, minute, weekend/rush hour status. Sample weights address class imbalance by inversely scaling with class frequency. This baseline provides initial feature importance rankings and performance metrics that guide subsequent ablation steps, while establishing a critical comparison point for evaluating the value of iterative refinement.

Phase 2: Primary Feature Selection Leveraging importance scores from Phase 1, this phase identifies the most predictive features by training models with reduced sets (3, 5, or 10 features). The optimal feature count is determined by maximizing F_1 score, significantly reducing dimensionality while preserving predictive power. This step isolates key sensor locations and temporal patterns critical for congestion prediction at each specific site, contrasting with the GCN/LSTM's continuous spatial aggregation.

Phase 3: Advanced Feature Engineering This phase introduces lagged features at multiple intervals (5–720 minutes) to explicitly model temporal dependencies that deep learning architectures capture implicitly. The expanded set includes:

- Short-term dynamics (5–60 minute lags)
- Mid-term patterns (720-minute lag)
- Daily/weekly periodicity

These engineered features compensate for XGBoost's lack of inherent temporal modeling, directly encoding how congestion evolves from free-flow to peak conditions—addressing a key limitation of non-sequential tree-based models.

Phase 4: Fine-Grained Feature Selection Systematic selection across 5–50 features identifies the minimal optimal set by evaluating performance at each cardinality. The optimal count is determined at the point of diminishing returns, where additional features no longer yield significant accuracy improvements (represented in this work by the more robust F_1 score metric that takes both precision and recall into account). This phase ensures computational efficiency while eliminating redundant signals, contrasting with GCN/LSTM's fixed architectural constraints.

Phase 5: Final Model and Threshold Optimization A comprehensive grid search optimizes hyperparameters across:

- Learning rate [0.01, 0.1]
- Tree depth [3, 5, 7, 9]
- Boosting rounds [100, 300, 500, 700]

The configuration maximizing validation F_1 score is selected, completing the ablation pipeline. Lastly, the optimal decision threshold is established using the same procedure employed for the aforementioned deep-learning models.

Overall, this multiphase approach demonstrates that sequential refinement creates synergistic performance gains that cannot be easily obtained through isolated optimizations.

6. Experimental Study

This section presents the comprehensive evaluation of the proposed traffic congestion prediction models. All experiments were conducted on an Intel-Xeon based server equipped with an NVIDIA A-100 GPU with 40GB of VRAM, with the three methods under consideration utilising the GPU for both training and inference. Regarding data, the publicly available PEMS-BAY dataset and the proprietary Dutch National Data Warehouse (NDW) dataset were used, each providing unique spatio-temporal characteristics and allowing for comprehensive validation under diverse traffic conditions.

Our evaluation protocol employs a static model approach. The model is trained once on the training set (80% of chronologically ordered data), with the test set (remaining 20%) being used only for final evaluation without any model updates. This protocol reflects a practical scenario where the model is trained offline once and deployed for inference. In a real-time deployment scenario, the model could be periodically retrained (e.g., weekly) to incorporate new data, but this is beyond the scope of the current evaluation.

6.1. Datasets Used

PEMS-BAY[1] is a widely adopted open-access dataset sourced from the California Department of Transportation's Performance Measurement System (PeMS). It comprises 325 loop-detector sensors deployed across the San Francisco Bay Area highway network, recording vehicular speed at 5-minute intervals from January 2017 to June 2017 (6 months total). This yields 52,115 samples per sensor (16,937,375 samples network-wide) with an average speed of 62.62 mph. Critically, the dataset provides precise geocoordinates and sensor IDs, facilitating the construction of a directional adjacency matrix that captures spatial dependencies between sensors. For our experiments, 18 target sensors were randomly selected to ensure unbiased evaluation of the model's generalization capability. As a benchmark dataset in traffic forecasting literature, PEMS-BAY's open availability enables reproducibility and direct comparison with state-of-the-art methods.

NDW[2] is a proprietary dataset provided by the Dutch National Data Warehouse, containing high-resolution traffic measurements from 204 loop detectors along two major highways (A20 and A4) in Rotterdam, the Netherlands. Collected at 1-minute intervals over 13 months (April 2022 – May 2023), it delivers 544,320 samples per sensor (112,129,920 samples network-wide) with an average speed of 94.31 kph. The dataset uniquely captures complex traffic dynamics across critical infrastructure including merging zones, bottlenecks, and free-flow segments, while encompassing seasonal variations, weather-related disruptions (e.g., rain/snow events), and incident-induced anomalies. Unlike PEMS, NDW's target sensors were strategically defined by local traffic management experts as four *Locations of Interest* (LOIs)—nodes identified as congestion initiators where early intervention can prevent network-wide spillback. This expert-curated selection presents a more challenging prediction scenario but offers higher real-world utility for traffic management. Due to its proprietary nature, NDW access is restricted to approved research collaborations with Dutch transportation authorities.

Both datasets are visually presented in Figure 3. Their comparative characteristics are summarized in Table 1.

6.2. Experimental Results

This section presents an analysis of the comparative performance of the three proposed approaches for short-term traffic congestion prediction. We evaluate each model across the two distinct datasets (NDW and PEMS-BAY) at multiple prediction horizons (10–60 minutes), with performance measured using precision, recall, and F_1 score after decision threshold tuning. The results reveal significant differences in predictive capabilities, computational efficiency and robustness across horizons.

[1] https://pems.dot.ca.gov
[2] https://english.ndw.nu

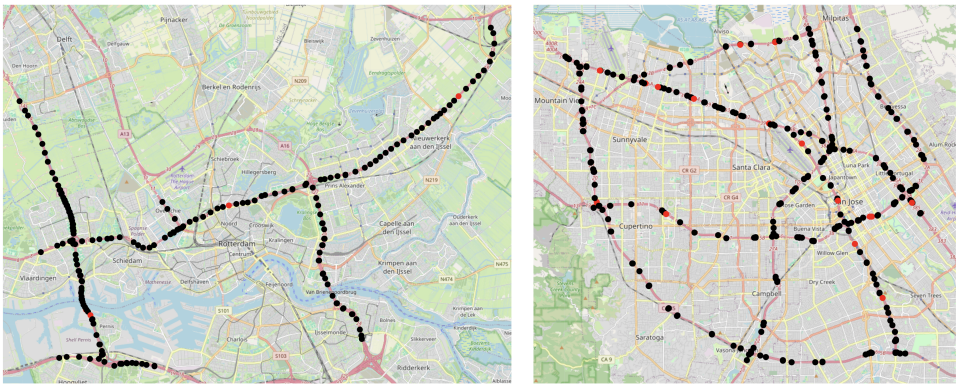

Figure 3: The NDW (left) and PeMS (right) datasets. Target traffic speed sensors are colored red, with the rest colored black.

Table 1
Summary of datasets used in congestion prediction experiments.

Attribute	PEMS-BAY	NDW
Location	San Francisco, USA	Rotterdam, NL
No. of Sensors	325	204
Total Time	6 months	13 months
Sampling Rate	5 minutes	1 minute
Total Samples	≈17M	≈112M
Average Speed	62.62 mph	94.31 kph
Target Sensors	18 (random)	4 (expert-picked)
Data Availability	Open	Proprietary

Table 2 presents the complete experimental results, including the optimal decision thresholds, performance metrics, and computational characteristics for all configurations. The most striking observation is the consistent superiority of the XGBoost approach across all prediction horizons and both datasets.

Specifically, the results show that the XGBoost approach achieves the highest F_1 scores across all prediction horizons on both datasets. On the NDW dataset, XGBoost attains F_1 scores ranging from 0.70 (10-minute horizon) to 0.44 (60-minute horizon), significantly outperforming GCN (0.50 to 0.43) and LSTM (0.51 to 0.43). Similarly, on the PEMS-BAY dataset, XGBoost maintains F_1 scores from 0.86 down to 0.74, while GCN and LSTM fluctuate in the 0.72–0.74 range. This consistent superiority validates the effectiveness of the systematic feature engineering approach, particularly the incorporation of multi-scale temporal lag features that capture both short-term dynamics and periodic patterns. Additionally, XGBoost appears to be more robust overall, with the standard deviation of its F_1 scores being consistently lower than those of its deep-learning competitors, especially for lower prediction horizons.

Regarding optimal threshold identification, XGBoost consistently employs high decision thresholds (0.76–0.86) across all horizons, indicating its preference for high-confidence predictions that maximize precision (0.39–0.84). In contrast, GCN and LSTM use significantly lower thresholds (0.15–0.45), prioritizing recall (0.55–0.88) at the expense of precision. This trade-off aligns with their architectural designs: the deep learning models' spatial propagation mechanisms inherently favor capturing congestion spread (high recall), while XGBoost's feature-based approach enables more discriminative decision boundaries. Notably, the XGBoost models maintain recall above 0.55 even at the 60-minute horizon on NDW, demonstrating its ability to balance both metrics effectively.

The computational characteristics, however, reveal a big contrast between the approaches. XGBoost demonstrates significantly improved efficiency in the inference phase, as it processes individual predictions in 0.10–1.10 μs/sample, while GCN and LSTM require 57.5–210.1 μs/sample. This three-orders-of-magnitude advantage in inference speed is critical for real-time traffic management systems where prediction latency directly impacts intervention effectiveness.

Table 2
Experimental results comparing GCN, LSTM, and XGBoost across NDW and PEMS-BAY datasets. Metrics include the optimal decision threshold, Precision, Recall, F_1-score and inference time per sample.

Dataset	Method	Prediction Horizon	Decision Threshold	Precision	Recall	F_1-Score $\mu(\sigma)$	Inference Time per sample (μs)
NDW	GCN	10	0.20	0.42	0.66	0.50 (0.18)	80.25
		20	0.21	0.42	0.64	0.49 (0.18)	86.50
		30	0.18	0.40	0.62	0.47 (0.19)	90.50
		40	0.20	0.39	0.58	0.46 (0.18)	86.75
		50	0.18	0.38	0.55	0.44 (0.19)	97.25
		60	0.20	0.37	0.53	0.43 (0.19)	90.00
	LSTM	10	0.24	0.44	0.65	0.51 (0.18)	75.50
		20	0.19	0.42	0.65	0.49 (0.19)	79.50
		30	0.21	0.40	0.62	0.48 (0.19)	57.50
		40	0.19	0.40	0.57	0.46 (0.19)	72.75
		50	0.18	0.37	0.58	0.44 (0.19)	59.75
		60	0.15	0.35	0.60	0.43 (0.20)	58.00
	XGBoost	10	0.86	0.67	0.74	**0.70 (0.10)**	**0.10**
		20	0.82	0.61	0.62	**0.61 (0.14)**	**0.12**
		30	0.85	0.50	0.62	**0.55 (0.15)**	**0.10**
		40	0.76	0.43	0.62	**0.50 (0.17)**	**0.10**
		50	0.76	0.38	0.59	**0.45 (0.21)**	**0.10**
		60	0.79	0.39	0.55	**0.44 (0.21)**	**0.12**
PEMS	GCN	10	0.42	0.68	0.80	0.73 (0.18)	210.11
		20	0.41	0.68	0.81	0.73 (0.18)	192.00
		30	0.40	0.68	0.81	0.73 (0.17)	178.61
		40	0.39	0.67	0.81	0.72 (0.18)	181.56
		50	0.40	0.67	0.81	0.72 (0.18)	185.06
		60	0.41	0.68	0.80	0.73 (0.18)	169.50
	LSTM	10	0.43	0.70	0.80	0.74 (0.17)	147.72
		20	0.43	0.68	0.81	0.73 (0.18)	145.67
		30	0.45	0.68	0.79	0.72 (0.18)	144.00
		40	0.42	0.68	0.80	0.72 (0.18)	170.17
		50	0.42	0.69	0.80	0.73 (0.17)	154.89
		60	0.40	0.67	0.81	0.73 (0.17)	138.89
	XGBoost	10	0.86	0.84	0.88	**0.86 (0.09)**	**1.09**
		20	0.84	0.79	0.82	**0.80 (0.12)**	**0.88**
		30	0.85	0.74	0.81	**0.77 (0.15)**	**0.93**
		40	0.82	0.73	0.79	**0.76 (0.15)**	**0.67**
		50	0.86	0.70	0.82	**0.75 (0.16)**	**0.91**
		60	0.85	0.69	0.80	**0.74 (0.18)**	**1.10**

6.3. A Note on Explainability

Another key advantage of the XGBoost model over black-box deep learning models is its inherent explainability. Through iterative feature ablation, it uses an identifiable and minimal sensor set to identify how congestion propagates between specific road segments at precise time intervals. For example, it can uncover patterns like morning rush-hour disturbances at arterial junctions causing downstream bottlenecks in 15–20 minutes, or evening patterns that spread from commercial to residential zones. This clarity lets traffic engineers directly identify critical relationships (e.g., how one interchange sequentially affects three segments) and distinguish congestion catalysts from buffers. The model's simplicity makes these insights immediately actionable without data science expertise, enabling targeted interventions

like timed signal adjustments.

In a more detailed example, sensors located at the A4 highway's merging zone near the Benelux tunnel and the A20 highway's interchange with the A13 consistently displayed high feature importance scores (ranked among the top 3–5 features) when predicting congestion at three of the four LOI target sensors. These sensors were consistently selected as important predictors regardless of the prediction horizon (10, 20, or 30 minutes), suggesting they represent genuine congestion initiation points rather than transient correlations. Critically, these high-importance sensors correspond precisely to locations that traffic engineering experts have identified as known bottlenecks, i.e., areas where traffic flow naturally degrades due to lane reductions, merging maneuvers, and geometric design. The feature importance analysis revealed that congestion at these upstream locations typically manifests as leading indicators for downstream LOI congestion with a time lag of approximately 15–25 minutes, which aligns closely with empirical observations of traffic wave propagation speeds on Dutch highways. This finding validates that the XGBoost model has learned physically meaningful relationships that mirror real-world traffic dynamics.

7. Conclusions & Future Work

This study shows that a well-engineered XGBoost model consistently outperforms deep learning architectures like GCNs and LSTMs in short-term traffic congestion forecasting, across both public and proprietary datasets. The model achieved higher F_1 scores at all prediction horizons while requiring far less computational resources, enabling deployment on standard hardware and frequent updates. For transportation agencies, this means advanced predictive capabilities are now accessible without costly GPU infrastructure. The model's transparency also supports collaboration between data scientists and traffic managers, and allows municipalities of all sizes to implement effective, timely interventions against congestion.

Future work will focus on validating generalizability across more large-scale urban datasets, exploring incremental retraining to handle shifting traffic patterns, and investigating deployment on edge hardware for decentralized, low-latency inference within traffic infrastructure.

Acknowledgments

This work was supported partially by the Horizon Europe R&I programme EMERALDS under the GA No. 101093051 and the University of Piraeus Research Center (UPRC). The authors also acknowledge Mr. Erik-Sander Smits (ARANE, NL) for his expert feedback.

Declaration on Generative AI

During the preparation of this work, the authors used LLMs in order to conduct grammar and spelling checks. After using these tools, the authors reviewed and edited the content as needed and take full responsibility for the publication's content.

References

[1] M. Mokbel, M. Sakr, L. Xiong, A. Züfle, J. Almeida, T. Anderson, W. Aref, G. Andrienko, N. Andrienko, Y. Cao, S. Chawla, R. Cheng, P. Chrysanthis, X. Fei, G. Ghinita, A. Graser, D. Gunopulos, C. S. Jensen, J.-S. Kim, K.-S. Kim, P. Kröger, J. Krumm, J. Lauer, A. Magdy, M. Nascimento, S. Ravada, M. Renz, D. Sacharidis, F. Salim, M. Sarwat, M. Schoemans, C. Shahabi, B. Speckmann, E. Tanin, X. Teng, Y. Theodoridis, K. Torp, G. Trajcevski, M. van Kreveld, C. Wenk, M. Werner, R. Wong, S. Wu, J. Xu, M. Youssef, D. Zeinalipour, M. Zhang, E. Zimányi, Mobility data science: Perspectives and

challenges, ACM Trans. Spatial Algorithms Syst. 10 (2024). URL: https://doi.org/10.1145/3652158. doi:10.1145/3652158.

[2] C. Zheng, X. Fan, C. Wang, J. Qi, Gman: A graph multi-attention network for traffic prediction, Proceedings of the AAAI Conference on Artificial Intelligence 34 (2020) 1234–1241. URL: https://ojs.aaai.org/index.php/AAAI/article/view/5477. doi:10.1609/aaai.v34i01.5477.

[3] T. Afrin, N. Yodo, A survey of road traffic congestion measures towards a sustainable and resilient transportation system, Sustainability 12 (2020). URL: https://www.mdpi.com/2071-1050/12/11/4660. doi:10.3390/su12114660.

[4] T. Alghamdi, K. Elgazzar, M. Bayoumi, T. Sharaf, S. Shah, Forecasting traffic congestion using arima modeling, in: 2019 15th International Wireless Communications & Mobile Computing Conference (IWCMC), IEEE, 2019, pp. 1227–1232. doi:10.1109/IWCMC.2019.8766698.

[5] Y. Liu, H. Zheng, X. Feng, Z. Chen, Short-term traffic flow prediction with conv-lstm, in: 2017 9th International Conference on Wireless Communications and Signal Processing (WCSP), 2017, pp. 1–6. doi:10.1109/WCSP.2017.8171119.

[6] L. Zhao, Y. Song, C. Zhang, Y. Liu, P. Wang, T. Lin, M. Deng, H. Li, T-gcn: A temporal graph convolutional network for traffic prediction, IEEE Transactions on Intelligent Transportation Systems 21 (2019) 3848–3858. doi:10.1109/TITS.2019.2935152.

[7] K. Guo, Y. Hu, Z. Qian, H. Liu, K. Zhang, Y. Sun, J. Gao, B. Yin, Optimized graph convolution recurrent neural network for traffic prediction, IEEE Transactions on Intelligent Transportation Systems 22 (2019) 1138–1149. doi:10.1109/TITS.2019.2963722.

[8] N. Ranjan, S. Bhandari, H. P. Zhao, H. Kim, P. Khan, City-wide traffic congestion prediction based on cnn, lstm and transpose cnn, volume 8, 2020, pp. 81606–81620. doi:10.1109/ACCESS.2020.2991462.

[9] A. M. Nagy, V. Simon, Improving traffic prediction using congestion propagation patterns in smart cities, Advanced Engineering Informatics 50 (2021) 101343. URL: https://www.sciencedirect.com/science/article/pii/S1474034621000963. doi:https://doi.org/10.1016/j.aei.2021.101343.

[10] M. Chen, X. Yu, Y. Liu, Pcnn: Deep convolutional networks for short-term traffic congestion prediction, IEEE Transactions on Intelligent Transportation Systems 19 (2018) 3550–3559. doi:10.1109/TITS.2018.2835523.

[11] J. Li, J. Li, Y. jiao Gong, Multi-task learning for sparse traffic forecasting, ArXiv abs/2211.09984 (2022). URL: https://doi.org/10.48550/arXiv.2211.09984.

[12] Z. Liu, Z. Jiang, Z. Xu, T. Xiao, Z. Xiao, Y. zhang, H. Wang, S. Chen, Spatio-temporal partial sensing forecast for long-term traffic, 2025. URL: https://arxiv.org/abs/2408.02689. arXiv:2408.02689.

[13] G. Naidu, T. Zuva, E. Sibanda, A review of evaluation metrics in machine learning algorithms, in: R. Silhavy, P. Silhavy (Eds.), Artificial Intelligence Application in Networks and Systems, Springer International Publishing, 2023, pp. 15–25. doi:10.1007/978-3-031-35314-7_2.

[14] B. Yu, H. Yin, Z. Zhu, Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting, in: Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18, International Joint Conferences on Artificial Intelligence Organization, 2018, pp. 3634–3640. URL: https://doi.org/10.24963/ijcai.2018/505. doi:10.24963/ijcai.2018/505.