Engineering Applications of Artificial Intelligence 159 (2025) 111757

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai

Research paper

DynaKey-GNN: An efficient dynamic key-node multi-graph neural network for spatio-temporal traffic flow forecasting

Guangrui Fan [a,b], Aznul Qalid Md Sabri [b,*], Siti Soraya Abdul Rahman [b], Lihu Pan [a]

[a] School of Computer Science and Technology, Taiyuan University of Science and Technology, 66 Waliu Road, Taiyuan, Shanxi 032200, China
[b] Department of Artificial Intelligence, Faculty of Computer Science and Information Technology, Universiti Malaya, Wilayah Persekutuan, Kuala Lumpur 50603, Malaysia

ARTICLE INFO

Keywords:
Traffic flow prediction
Dynamic node identification
Spatio-temporal modeling
Incremental update
Adaptive networks

ABSTRACT

Accurate traffic flow prediction is crucial for effective management of urban transportation but remains challenging due to rapidly changing network conditions. We introduce a dynamic key-node multi-graph neural network (DynaKey-GNN) that identifies both long-term vital nodes and short-term critical nodes, enabling real-time adaptation to traffic shifts. An incremental update strategy selectively processes network changes, boosting computational efficiency without sacrificing accuracy. We also propose a dual-stream architecture that fuses global patterns with targeted key-node processing, capturing both stable and fast-evolving dependencies. Experiments on four real-world traffic datasets show that our approach achieves up to 12.37% higher accuracy than state-of-the-art baselines under dynamic scenarios. Case studies on volatile nodes further confirm the model's ability to handle abrupt fluctuations in traffic flow, providing consistent and reliable forecasts.

1. Introduction

Traffic flow prediction is a fundamental component of intelligent transportation systems (ITS), integral to urban planning, traffic management, and the development of smart cities (Gomes et al., 2023; Xu et al., 2024). Accurate forecasting of traffic conditions enables proactive decision-making, facilitating the optimization of traffic signal timings, the implementation of dynamic routing strategies, and the efficient allocation of transportation resources (Sayed et al., 2023). However, predicting traffic flow remains a complex challenge due to the intricate and dynamic nature of urban traffic networks.

One central issue is the high spatial–temporal dependency across the network, as traffic conditions at a given location can significantly impact distant areas with varying time lags, necessitating models capable of capturing these long-range interactions (Gomes et al., 2023). In addition, traffic flows are non-linear and stochastic, influenced by diverse elements such as weather conditions, accidents, and special events, making them inherently unpredictable (Gao and Wang, 2024). Further compounding this complexity is the combination of periodic and irregular fluctuations in traffic patterns. Urban networks typically display strong daily or weekly cycles but also experience abrupt deviations from these cycles due to unanticipated incidents, demanding models that can accommodate both steady and transient dynamics (Chen et al., 2024a). Finally, the intricate topology of urban road systems adds another layer of difficulty, as the importance of certain road segments can shift dramatically in response to real-time conditions like sudden congestion or closures (Zhang et al., 2024c).

Graph Neural Networks (GNNs) have emerged as a powerful tool for modeling the spatial–temporal dependencies inherent in traffic networks (Wu et al., 2020a). Models such as the Diffusion Convolutional Recurrent Neural Network (DCRNN) (Li et al., 2018), Spatial-Temporal Graph Convolutional Networks (STGCN) (Yu et al., 2017), Dynamic Spatial-Temporal Aware Graph Neural Network (DSTAGNN) (Lan et al., 2022), and Dynamic Graph Convolutional Recurrent Network (DMST-GCN) (Li et al., 2023) have demonstrated significant improvements in traffic flow prediction by leveraging the graph structure of road networks. These models incorporate various dynamic adaptation strategies, such as adaptive adjacency matrices and temporal convolutions, to better capture the evolving traffic patterns.

Building upon these foundational approaches, recent research has focused on identifying and leveraging vital or pivotal nodes within traffic networks to further enhance prediction accuracy. Accurately identifying these nodes allows models to prioritize their interactions and dependencies, leading to more precise and reliable traffic predictions. However, the importance of these nodes can fluctuate rapidly due to factors like accidents, road closures, or special events, necessitating dynamic identification mechanisms to maintain prediction accuracy in real-time. Kong et al. (2024b) introduced pivotal nodes as

* Corresponding author.
E-mail addresses: fgr@tyust.edu.cn (G. Fan), aznulqalid@um.edu.my (A.Q.M. Sabri), siti_soraya@um.edu.my (S.S.A. Rahman), panlh@tyust.edu.cn (L. Pan).

https://doi.org/10.1016/j.engappai.2025.111757
Received 14 January 2025; Received in revised form 2 July 2025; Accepted 9 July 2025

network locations with extensive connections and central roles, proposing the Spatio-Temporal Pivotal Graph Neural Networks (STPGNN) to capture these nodes' influence in aggregating and distributing traffic flow. Similarly, the DDGformer model (Li et al., 2024b) incorporated direction- and distance-aware mechanisms to better model relationships between critical nodes. These studies have highlighted the importance of specialized handling of high-influence nodes in traffic prediction.

Despite these advancements, a limitation persists in current approaches: most existing methods treat the identification and importance of these vital nodes as static, relying primarily on historical data to determine node significance. This static treatment fails to account for the inherently dynamic nature of traffic networks, where node importance can fluctuate rapidly due to factors such as accidents, road closures, special events, and evolving traffic patterns.

For instance, an analysis of traffic flow data from the PEMS08 dataset using the pivotal node identification method proposed by Kong et al. (2024b) revealed significant temporal variations in node significance (see Fig. 1). Nodes ranked 15–35 in overall importance exhibited substantial rank fluctuations over eight consecutive weeks, indicating that static importance metrics fail to capture the dynamic nature of traffic networks. Such variations underscore the need for traffic prediction models that can adapt in real-time to shifting network dynamics, ensuring accurate predictions during both stable and critical events. Moreover, real-world scenarios often involve sudden incidents like accidents or road construction, which can rapidly alter the dynamics of the traffic network. Previously insignificant alternative routes may suddenly become vital for maintaining traffic flow. Static models, failing to recognize these dynamic changes, might continue to prioritize historically important nodes, potentially exacerbating congestion by directing traffic through already overcrowded areas.

Addressing this critical gap, we propose DynaKey-GNN, an efficient dynamic key-node multi-graph neural network designed to enhance spatio-temporal traffic flow forecasting by dynamically identifying and leveraging key nodes in the traffic network. Our model introduces three key innovations:

1. **Dynamic Key Node Identification:** We introduce a mechanism that identifies both long-term vital nodes and short-term dynamic key nodes, enabling real-time adaptation to network changes.
2. **Incremental Update Strategy:** We develop an efficient incremental update strategy that selectively processes network changes while maintaining computational efficiency.
3. **Dual-Stream Architecture:** We design a dual-stream architecture that effectively integrates global network patterns with focused key node processing, capturing both stable and dynamic dependencies.

This work pioneers the treatment of traffic forecasting as a dynamic key-node problem, introducing DynaKey-GNN, the first model to unify long-term pivotal nodes with short-term pivotal nodes inside a dual-stream graph-neural framework. A novel dual-perspective identification module fuses structural centrality cues with multi-scale temporal and anomaly signals, while an adaptive gate automatically balances their influence according to real-time network stability. To keep the model responsive, we devise a three-tier hierarchical update scheduler that selectively recalculates only the affected sub-graphs, cutting per-update runtime by up to 94% on the largest benchmark. Experiments on four public PeMS datasets show that DynaKey-GNN surpasses 16 strong baselines, yielding up to 12.37% MAE reduction in highly dynamic scenarios without sacrificing real-time performance, and ablation studies confirm the indispensability of each design choice.

2. Related work

Traffic flow prediction has significantly advanced with deep learning, especially Graph Neural Networks (GNNs), effectively modeling complex spatial–temporal dependencies in urban networks (Wu et al., 2020a; Jin et al., 2023a; Shao et al., 2024). This section briefly summarizes GNN evolution for traffic prediction, emphasizing dynamic behaviors, key node identification, recent methodological refinements, and emerging applications of Large Language Models (LLMs).

2.1. Graph neural networks for traffic prediction

GNNs have become foundational for capturing spatial relationships in traffic networks. Early influential models include Diffusion Convolutional Recurrent Neural Networks (DCRNN) (Li et al., 2018) and Spatial-Temporal Graph Convolutional Networks (STGCN) (Yu et al., 2017), which integrated graph convolutions with recurrent and temporal convolutions, respectively. Subsequent enhancements include Graph WaveNet (Wu et al., 2019) with adaptive adjacencies and Adaptive Graph Convolutional Recurrent Networks (AGCRN) (Bai et al., 2020), employing data-driven graph structures. Further developments incorporate automated dilated graph modeling (Jin et al., 2022), causal inference in heterophilic graphs (He et al., 2024), and spatio-temporal point processes for predicting congestion events (Jin et al., 2023b).

Recent advances introduced sophisticated architectures like MHGNet for multi-heterogeneous graph modeling (Wu et al., 2025), periodicity-aware adaptive hypergraph networks (Zhao et al., 2025a), and enhanced directed-graph modeling integrating physical constraints (Wang et al., 2024c). Efficiency and scalability improvements have been significant, with methods like LightST employing knowledge distillation for accelerated inference (Zhang et al., 2025) and GraphSparseNet (GSNet) achieving linear complexity for large-scale predictions (Kong et al., 2025).

Additionally, Large Language Models (LLMs) have emerged as a new paradigm in spatio-temporal forecasting. Foundational models such as ST-LLM (Liu et al., 2024) redefine traffic prediction using LLM tokenization methods. Enhanced versions like ST-LLM+ integrate graph structures explicitly (Liu et al., 2025). Universal frameworks (UniST) (Yuan et al., 2025) and specialized adaptations (TrafficLLM, LLM-PS, STEM-LTS) (Cui et al., 2025; Tang et al., 2025a; Zhao et al., 2025b) further illustrate the versatility of LLM-based approaches. Despite promising initial results, challenges such as tokenization accuracy, computational efficiency, and physical plausibility remain (Cui et al., 2025; Zhao et al., 2025b).

Multi-scale GNN approaches, including MTGNN (Wu et al., 2020b), GMAN (Zheng et al., 2020), GT-LSTM (Luo et al., 2024), and HTVGNN (Dai and Ye, 2024), underscore the importance of capturing dependencies across scales. However, many models still inadequately handle rapid or dynamic traffic changes efficiently and precisely (Zhang et al., 2024d; Wu and Chen, 2024).

2.2. Dynamic adaptation in traffic networks

Recent models increasingly address the dynamic and unpredictable nature of urban traffic patterns through adaptive architectures. Dynamic Graph Convolutional Networks (DGCRN) (Li et al., 2023) adapt adjacency matrices temporally, while the Dynamic Dual-Stream Temporal Graph Neural Network (D2STGNN) (Zong et al., 2024) employs dual streams for static and dynamic pattern recognition. Enhanced temporal modeling and evolving graph structures are central in Intertwined Dynamic Graph Convolutional Networks (IDDGCN) (Wang et al., 2024b) and Continuous Evolution Graph Neural Networks (CEGNCDE) (Wu and Chen, 2024).

AdpstGCN (Chen et al., 2024b) dynamically integrate multi-view graphs via attention mechanisms, while the spatial–temporal fusion

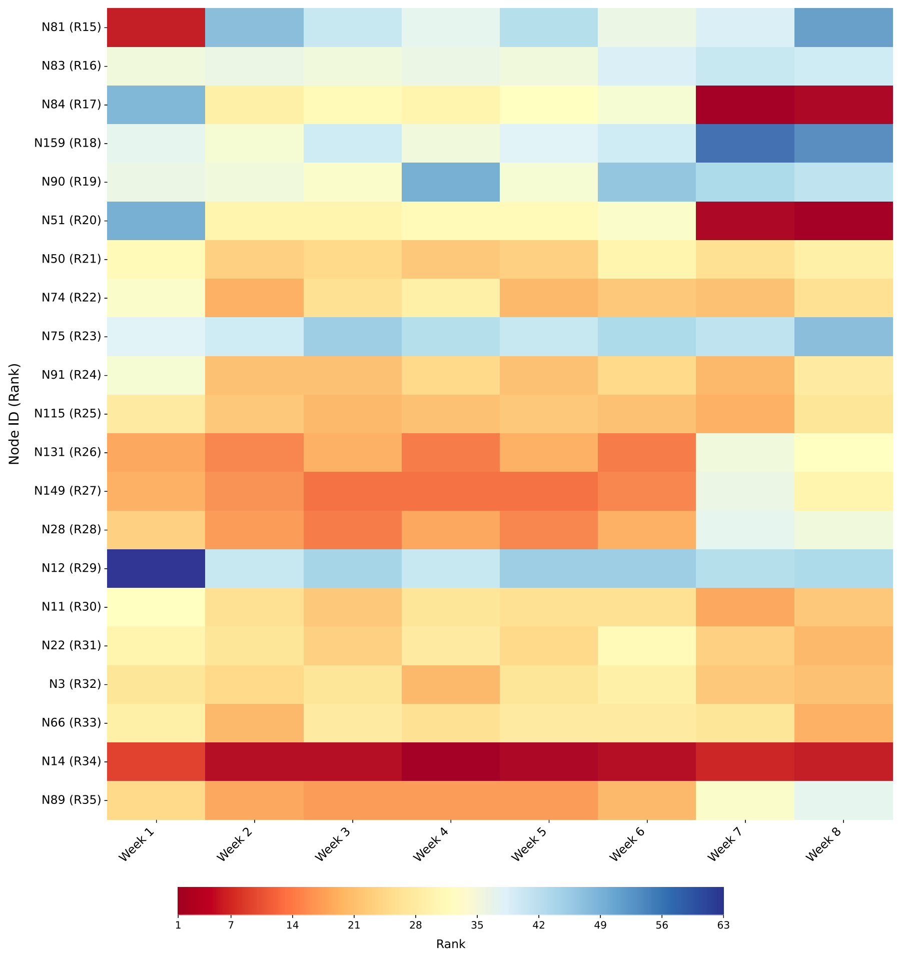

Fig. 1. Heatmap illustrating the dynamic importance of nodes over eight weeks in the PEMS08 dataset. The color intensity represents the relative importance score of each node at each week.

dynamic network by Li et al. (2024a) emphasizes real-time adaptation through evolving graph structures. Automated frameworks like Auto-DSTSGN (Jin et al., 2022) optimize graph topologies via neural architecture search. Advanced attention methods, including STC-PSSA (Zhang et al., 2024b) and CASTGNN (Wang et al., 2024a), leverage dynamic sparse and channel-aware mechanisms to enhance adaptability.

Contrastive learning has also advanced robustness and adaptation; notably, Guo et al. (2024) and Zhang et al. (2024a) developed contrastive frameworks for enhancing feature quality and adaptive graph augmentation. Graph trimming via causal attention (CAT) (He et al., 2024) further improves dynamic node weighting. Hybrid models integrating multi-scale features and adaptive structures, such as DM-STGCN (Hu et al., 2024) and STGMN (Ni and Zhang, 2022), have demonstrated significant predictive benefits. However, effectively managing abrupt large-scale shifts with computational efficiency remains challenging (Chen et al., 2024a; Cao et al., 2024).

2.3. Node importance in traffic prediction

Identifying influential nodes is crucial for accurate traffic predictions, especially in complex urban networks. Frameworks like STPGNN (Kong et al., 2024b) emphasized pivotal nodes—key intersections significantly influencing overall network behavior. TPP-GCN (Yang et al., 2024) and MVB-STNet (Xia et al., 2024) explored node importance through propagation dynamics and multi-view learning. Additionally, Go and Park (2024)'s global-local model highlights a conceptual parallel by distinguishing broader network trends from localized influences. CAT (He et al., 2024) implicitly assess node influence by selectively weighting neighbor contributions, while edge-centric predictions (Tang et al., 2025b) similarly relate closely to node importance. Interpretability frameworks, notably Traffexplainer (Kong et al., 2024a), provide mechanisms to reveal crucial spatial–temporal features influencing predictions. Although spatio-temporal large language models (ST-LLMs) implicitly highlight influential regions via attention (Liu et al., 2024, 2025), explicit and dynamic node importance integration into predictive modeling remains an ongoing research opportunity.

2.4. Limitations of current approaches

The field of traffic forecasting has witnessed remarkable advancements, with contemporary models demonstrating sophisticated capabilities in handling complex spatio-temporal dependencies, improving scalability, and even integrating paradigms from Large Language Models (Wu et al., 2025; Zhang et al., 2025; Liu et al., 2025). Recent GNNs like MHGNet (Wu et al., 2025) tackle multi-heterogeneous graphs, LightST (Zhang et al., 2025) achieves efficiency via distillation, and ST-LLMs like ST-LLM+ (Liu et al., 2025) merge graph structures with LLM strengths. Dynamic adaptation mechanisms, such as those in AdpstGCN (Chen et al., 2024b), allow models to adjust to evolving traffic conditions. Despite this progress, several fundamental limitations persist, creating a need for specialized solutions:

- **Granular and Efficient Dynamic Node Focus:** While many models adapt globally (Chen et al., 2024b) or employ complex architectures (Liu et al., 2025), the challenge of efficiently identifying

Table 1
Comparative overview of representative models.

Model	Dynamic graph	Node importance	Incremental update
DCRNN (Li et al., 2018)	No	No	No
STGCN (Yu et al., 2017)	No	No	No
STPGNN (Kong et al., 2024b)	Static	Static Pivotal	No
ASTGCN (Guo et al., 2019)	Partial (Attention-based)	Implicit (via attention)	No
DGCRN (Li et al., 2023)	Yes (Global, adaptive matrix)	No	Partial
DDGformer (Li et al., 2024b)	Yes (Global, adaptive)	Static Key Nodes	No
AdpstGCN (Chen et al., 2024b)	Yes (Adaptive multi-view)	Implicit (via attention)	No
MHGNet (Wu et al., 2025)	Yes (Dynamic ST graph generation)	Implicit (Pattern-based)	No
ST-LLM+ (Liu et al., 2025)	Partial (Graph-enhanced LLM)	Implicit (LLM/Graph attention)	No
DynaKey-GNN (Ours)	**Yes (Node-focused, incremental)**	**Dynamic (Short+Long-term)**	**Yes**

truly critical nodes in real-time — distinguishing between long-term structural importance and sudden, short-lived criticality — and then *incrementally* updating the model with minimal overhead remains significant. Many global adaptations can be computationally intensive, unsuitable for large-scale, rapidly changing networks where targeted updates are preferable.

- **Integrated Dynamic Importance Mechanisms:** Existing methods often treat node importance as static (Kong et al., 2024b), rely on implicit attention that may lack direct interpretability regarding specific node criticality, or offer importance as a post-hoc analysis (Kong et al., 2024a). There is a gap for models where dynamic, multi-faceted node importance (considering both long-term and short-term factors) is an intrinsic and driving component of the prediction mechanism itself, rather than an add-on or a purely implicit outcome.
- **Computational Complexity:** Many advanced dynamic adaptations or large-scale models, including some ST-LLMs, can be computationally demanding (Cui et al., 2025; Zhao et al., 2025b). While scalability solutions like knowledge distillation (Zhang et al., 2025) or linear-complexity backbones (Kong et al., 2025) are vital, achieving real-time responsiveness that also accounts for fine-grained, dynamic shifts in node-level significance necessitates architectures designed for both accuracy and efficient, targeted updates. The inability to swiftly and efficiently update based on changing node importance can delay responses to unfolding incidents.

These limitations highlight the need for frameworks that can (1) dynamically identify and leverage changing node importance at a fine granularity, considering both enduring and transient factors, and (2) perform these operations efficiently to enable real-time adaptation. This gap motivates our proposed DynaKey-GNN, which directly addresses these shortcomings by combining dynamic key-node identification with an efficient incremental update strategy, leading to more robust and responsive traffic flow predictions. Table 1 provides a comparative overview of several representative models, categorizing them based on their support for dynamic graph structures, node-level importance adjustments, and incremental updates.

3. Methodology

To address the aforementioned limitations, we propose DynaKey-GNN, an efficient dynamic key-node multi-graph neural network designed to enhance spatio-temporal traffic flow forecasting by dynamically identifying and leveraging key nodes in the traffic network. Unlike existing approaches that treat node importance as static or rely solely on adaptive edge weights, DynaKey-GNN introduces a novel mechanism for dynamic key node identification, capturing both long-term vital nodes and short-term dynamic key nodes through an incremental update strategy that ensures computational efficiency. In this section, we present DynaKey-GNN, a novel framework for traffic flow prediction that dynamically identifies and leverages key nodes within traffic networks. Our approach overcomes limitations of existing methods by enabling real-time adaptation to network changes while maintaining computational efficiency.

3.1. Problem formulation

We model the traffic network as a weighted directed graph $\mathbf{G} = (\mathbf{V}, \mathbf{E}, \mathbf{A})$, where

$\mathbf{V}$ is the set of nodes representing traffic sensors or intersections, with $|\mathbf{V}| = N$.
$\mathbf{E}$ is the set of edges representing road segments connecting the nodes.
$\mathbf{A} \in \mathbb{R}^{N\times N}$ is the adjacency matrix, where $\mathbf{A}_{ij}$ denotes the weight (e.g., distance, capacity) of the edge from node i to node j.

At each discrete time step t, traffic measurements are captured by a feature matrix $\mathbf{X}^{(t)} \in \mathbb{R}^{N\times F}$, where F is the number of features per node (e.g., traffic flow, speed). Given a historical sequence $\mathbf{X}^{(t-S+1)}, \mathbf{X}^{(t-S+2)}, \dots, \mathbf{X}^{(t)}$, where S is the number of past time steps considered, the objective is to predict the traffic measurements for the next T time steps:

$$\hat{\mathbf{X}}^{(t+1:t+T)} = \mathcal{F}\left(\mathbf{X}^{(t-S+1:t)}, \mathbf{G}\right), \tag{1}$$

where $\mathcal{F}$ represents the prediction function to be learned.

The primary challenges in this task are: (1) **Dynamic Key Node Identification:** Identifying and tracking nodes whose importance varies over time to capture their influence on traffic flow. (2) **Efficient Processing of Spatial-Temporal Patterns:** Modeling both global dependencies across the entire network and localized patterns around key nodes. (3) **Real-Time Adaptation to Network Changes:** Maintaining computational efficiency while updating the model in response to sudden changes in the network structure or traffic conditions.

3.2. Model overview

DynaKey-GNN addresses these challenges through a dual-stream architecture comprising two parallel processing streams: the Original Network Stream and the Key Node Subgraph Stream. This design enables the model to simultaneously capture comprehensive network-wide dependencies and focused localized patterns around key nodes. The framework is further enhanced by a Hierarchical Update Strategy that ensures computational efficiency during real-time network changes, as depicted in Fig. 2.

3.3. Dynamic key node identification

Accurately identifying and tracking dynamically important nodes in traffic networks is essential for enhancing traffic flow prediction. We introduce a dual-perspective key node identification mechanism that integrates both long-term structural patterns and short-term temporal dynamics. This approach ensures that the model adapts in real-time to shifting network conditions while maintaining computational efficiency. The design of this dual-perspective identification, incorporating normalized scores, learnable parameters, and adaptive weighting, aims to make the mechanism robust and adaptable to varying traffic dynamics, thereby promoting its potential for generalization across different urban environments, though comprehensive cross-city validation is an avenue for future work.

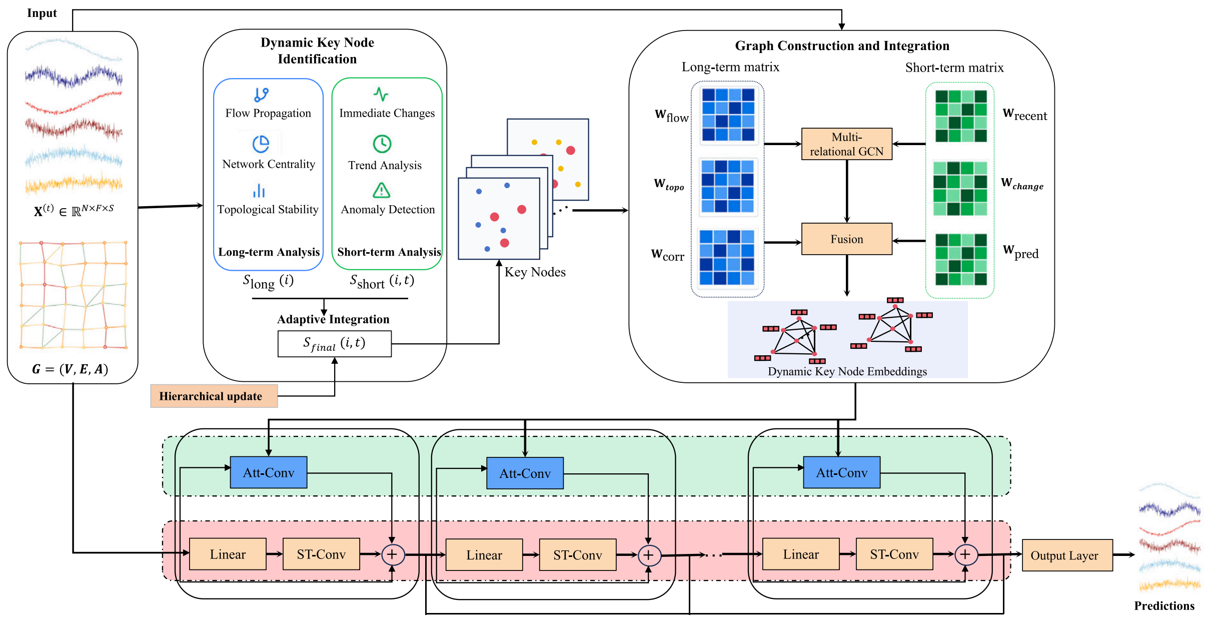

Fig. 2. Architecture of DynaKey-GNN, illustrating the dual-stream processing of the original traffic network and the key node subgraph, along with the hierarchical update mechanism.

3.3.1. *Long-term structural patterns*

To capture the enduring influence of nodes within the traffic network, we compute a long-term structural score $S_{\text{long}}(i)$ for each node i by combining three key measures: Flow Propagation Capability (FPC), Network Centrality Score (NCS), and Topological Stability Index (TSI).

Flow Propagation Capability (FPC). Flow Propagation Capability quantifies a node's ability to influence and be influenced by traffic patterns over time. It is defined as:

$$\text{FPC}(i) = \sum_{t=1}^{T} \sum_{j\in \mathbf{V}} \left[\phi(\mathbf{x}_i^t, \mathbf{x}_j^{t+\tau}) \cdot \exp\left(-\frac{d_{ij}}{\sigma}\right)\right], \tag{2}$$

where $\phi(\mathbf{x}_i^t, \mathbf{x}_j^{t+\tau})$ measures the flow correlation between node i at time t and node j at time $t+\tau$. d_{ij} denotes the network distance between nodes i and j. σ is a learnable distance decay parameter that modulates the influence of distance on flow propagation.

By incorporating both temporal correlations and spatial decay, FPC effectively captures how traffic flows propagate through the network over time, distinguishing our approach from traditional static centrality measures.

Network Centrality Score (NCS). Network Centrality Score evaluates the overall importance of a node within the network structure by integrating multiple centrality measures:

$$\text{NCS}(i) = \alpha_1 \cdot \text{BC}(i) + \alpha_2 \cdot \text{CC}(i) + \alpha_3 \cdot \text{EVC}(i), \tag{3}$$

where BC(i) is the Betweenness Centrality of node i, measuring the extent to which i lies on paths between other nodes. CC(i) is the Closeness Centrality of node i, indicating how close i is to all other nodes in the network. EVC(i) is the Eigenvector Centrality of node i, reflecting the influence of i based on the centrality of its neighbors. $\alpha_1, \alpha_2, \alpha_3$ are learnable weights that balance the contributions of each centrality measure.

Topological Stability Index (TSI). Topological Stability Index assesses the temporal stability of a node's structural importance by evaluating the variance in its degree over time. It is defined as:

$$\text{TSI}(i) = 1 - \frac{\text{var}\left(\sum_{t=1}^{T} \deg(i,t)\right)}{\max_j \left\{\text{var}\left(\sum_{t=1}^{T} \deg(j,t)\right)\right\}}, \tag{4}$$

where deg(i,t) represents the degree of node i at time t.

The final long-term structural score integrates FPC, NCS, and TSI:

$$S_{\text{long}}(i) = \beta_1 \cdot \text{FPC}(i) + \beta_2 \cdot \text{NCS}(i) + \beta_3 \cdot \text{TSI}(i), \tag{5}$$

where $\beta_1, \beta_2, \beta_3$ are learnable weights that determine the relative importance of each structural component.

3.3.2. *Short-term temporal dynamics*

In addition to long-term structural patterns, capturing short-term temporal dynamics is essential for adapting to rapid traffic changes. We define a short-term score $S_{\text{short}}(i,t)$ by integrating a Multi-Scale Temporal Pattern Score (MTP) and an Anomaly Detection Score (ADS):

$$S_{\text{short}}(i,t) = \gamma_1 \cdot \text{MTP}(i,t) + \gamma_2 \cdot \text{ADS}(i,t), \tag{6}$$

where γ_1, γ_2 are learnable weights balancing the contributions of each component.

Multi-scale Temporal Pattern Score (MTP). MTP captures traffic pattern variations at different temporal granularities, comprising three components:

$$\text{TPS}_1(i,t) = \frac{\|\mathbf{x}_i^t - \mathbf{x}_i^{t-1}\|}{\max_j\{\|\mathbf{x}_j^t - \mathbf{x}_j^{t-1}\|\}}, \tag{7}$$

$$\text{TPS}_2(i,t) = \frac{|\nabla \mathbf{x}_i^t|}{\max_j\{|\nabla \mathbf{x}_j^t|\}}, \tag{8}$$

$$\text{TPS}_3(i,t) = \frac{\|\mathbf{x}_i^t - \mathbf{x}_i^{t-p}\|}{\max_j\{\|\mathbf{x}_j^t - \mathbf{x}_j^{t-p}\|\}}, \tag{9}$$

where $\text{TPS}_1(i,t)$ measures the immediate change in traffic measurements for node i at time t. $\text{TPS}_2(i,t)$ captures the trending behavior of traffic patterns through the gradient of measurements.$\text{TPS}_3(i,t)$ accounts for periodic fluctuations in traffic flow based on a dominant periodicity p (e.g., 24 h). These components are weighted and summed to form the MTP:

$$\text{MTP}(i,t) = w_1 \cdot \text{TPS}_1(i,t) + w_2 \cdot \text{TPS}_2(i,t) + w_3 \cdot \text{TPS}_3(i,t), \tag{10}$$

where w_1, w_2, w_3 are learnable weights determining the relative importance of each temporal component.

Anomaly Detection Score (ADS). ADS identifies anomalous traffic behaviors that may indicate temporary shifts in node importance:

$$\text{ADS}(i,t) = \max\left(0, \left(\frac{|\mathbf{x}_i^t - \boldsymbol{\mu}_i^t|}{\boldsymbol{\sigma}_i^t} - \theta\right)\right), \tag{11}$$

where $\boldsymbol{\mu}_i^t$ and $\boldsymbol{\sigma}_i^t$ are running mean and standard deviation vectors for node i at time t. θ is a learnable threshold determining the sensitivity to anomalies.

While these MTP components effectively capture immediate, trending, and periodic fluctuations crucial for short-term node criticality assessment, more intricate sequential patterns influencing node importance could be explored using advanced sequence models in future iterations of the scoring mechanism.

3.3.3. Final dynamic key node score

The final dynamic key node score $S_{\text{final}}(i,t)$ integrates long-term and short-term components through an adaptive weighting mechanism based on network stability:

$$S_{\text{final}}(i,t) = \eta(t) \cdot S_{\text{long}}(i) + (1 - \eta(t)) \cdot S_{\text{short}}(i,t), \tag{12}$$

where

$$\eta(t) = \text{sigmoid}\left(\lambda \cdot \text{stability}(t)\right), \tag{13}$$

$$\text{stability}(t) = 1 - \frac{1}{N}\sum_{i=1}^{N} |\Delta S_{\text{short}}(i,t)|, \tag{14}$$

with: $\Delta S_{\text{short}}(i,t) = |S_{\text{short}}(i,t) - S_{\text{short}}(i,t-1)|$. λ is a learnable parameter controlling the sensitivity of the sigmoid function.

The stability metric stability(t) quantifies the overall network stability by averaging the absolute changes in short-term scores across all nodes. A higher stability value indicates a more stable network, thus increasing the influence of long-term structural scores $S_{\text{long}}(i)$. Conversely, a lower stability value implies frequent changes, elevating the impact of short-term dynamics $S_{\text{short}}(i,t)$.

3.4. Graph construction and integration

After dynamically identifying key nodes, constructing and integrating specialized graph structures is essential for capturing both long-term patterns and short-term dynamics. We create distinct adjacency matrices for long-term and short-term key nodes and integrate them using an attention-based mechanism to form the final graph structure utilized in the neural network layers.

Long-term key nodes graph. For long-term key nodes $\mathbf{V}_{\text{long}} \subseteq \mathbf{V}$, we construct a multi-relational weighted adjacency matrix $\mathbf{W}_{\text{long}}$ that encapsulates persistent traffic patterns and structural relationships:

$$\mathbf{W}_{\text{long}}(i,j) = \omega_1 \cdot \mathbf{W}_{\text{flow}}(i,j) + \omega_2 \cdot \mathbf{W}_{\text{topo}}(i,j) + \omega_3 \cdot \mathbf{W}_{\text{corr}}(i,j), \tag{15}$$

where $\mathbf{W}_{\text{flow}}(i,j)$ captures consistent traffic interactions through time-lagged correlations weighted by network distance:

$$\mathbf{W}_{\text{flow}}(i,j) = \sum_{t=1}^{T}\left[\phi(\mathbf{x}_i^t, \mathbf{x}_j^{t+\tau}) \cdot \exp\left(-\frac{d_{ij}}{\sigma}\right)\right], \tag{16}$$

in which $\phi(\mathbf{x}_i^t, \mathbf{x}_j^{t+\tau})$ measures the flow correlation between nodes i and j, d_{ij} is the network distance, and σ is a learnable distance decay parameter.

$\mathbf{W}_{\text{topo}}(i,j)$ incorporates structural importance using PageRank and common neighbor analysis:

$$\mathbf{W}_{\text{topo}}(i,j) = \text{PageRank}(i) \cdot \text{PageRank}(j) + \text{CN}(i,j), \tag{17}$$

in which PageRank(i) is the PageRank score of node i, and CN(i,j) denotes the number of common neighbors between nodes i and j.

$\mathbf{W}_{\text{corr}}(i,j)$ measures maximal temporal dependencies within a specified time window:

$$\mathbf{W}_{\text{corr}}(i,j) = \max_{\tau\in\mathcal{T}}\left|\text{Corr}(\mathbf{x}_i^t, \mathbf{x}_j^{t+\tau})\right|, \tag{18}$$

in which $\mathcal{T}$ is the set of considered time lags.

$\omega_1, \omega_2, \omega_3$ are learnable weights that balance the contributions of each relational component.

Short-term key nodes graph. For short-term key nodes $\mathbf{V}_{\text{short}} \subseteq \mathbf{V}$, we construct a dynamic adjacency matrix $\mathbf{W}_{\text{short}}(i,j,t)$ that adapts to rapid changes in traffic patterns:

$$\mathbf{W}_{\text{short}}(i,j,t) = \psi_1 \cdot \mathbf{W}_{\text{recent}}(i,j,t) + \psi_2 \cdot \mathbf{W}_{\text{change}}(i,j,t) + \psi_3 \cdot \mathbf{W}_{\text{pred}}(i,j,t), \tag{19}$$

where $\mathbf{W}_{\text{recent}}(i,j,t)$ captures recent interactions through exponentially weighted similarities over a short time window:

$$\mathbf{W}_{\text{recent}}(i,j,t) = \sum_{k=1}^{K}\gamma_k \cdot \exp\left(-\frac{t-t_k}{\tau}\right) \cdot \text{Sim}(\mathbf{x}_i^{t_k}, \mathbf{x}_j^{t_k}), \tag{20}$$

in which Sim$(\cdot,\cdot)$ denotes a similarity measure (e.g., cosine similarity), t_k are recent time steps within the window, and τ controls the decay rate.

$\mathbf{W}_{\text{change}}(i,j,t)$ identifies synchronized pattern changes through gradient analysis:

$$\mathbf{W}_{\text{change}}(i,j,t) = \mathbb{I}\left(|\nabla\mathbf{x}_i^t| > \theta_{\text{grad}}\right) \cdot \mathbb{I}\left(|\nabla\mathbf{x}_j^t| > \theta_{\text{grad}}\right), \tag{21}$$

in which $\mathbb{I}$ is the indicator function and θ_{grad} is a threshold for gradient magnitude.

$\mathbf{W}_{\text{pred}}(i,j,t)$ incorporates predictive elements to anticipate future connections based on current trends:

$$\mathbf{W}_{\text{pred}}(i,j,t) = \text{MLP}\left(\mathbf{h}_i^t \oplus \mathbf{h}_j^t\right), \tag{22}$$

in which MLP is a multilayer perceptron and $\oplus$ denotes concatenation of node embeddings.

ψ_1, ψ_2, ψ_3 are learnable weights that balance the influence of each dynamic component.

Attention-based graph integration. To effectively integrate information from both long-term and short-term graph structures, we employ an attention-based mechanism that adaptively combines $\mathbf{W}_{\text{long}}$ and $\mathbf{W}_{\text{short}}(i,j,t)$:

$$\mathbf{W}_{\text{final}}(i,j,t) = \alpha(t) \cdot \mathbf{W}_{\text{long}}(i,j) + (1-\alpha(t)) \cdot \mathbf{W}_{\text{short}}(i,j,t), \tag{23}$$

where $\alpha(t)$ is dynamically adjusted based on network stability:

$$\alpha(t) = \text{sigmoid}\left(\lambda \cdot \text{stability}(t)\right), \tag{24}$$

$$\text{stability}(t) = 1 - \frac{1}{N}\sum_{i=1}^{N}|\Delta S_{\text{short}}(i,t)|, \tag{25}$$

with $\Delta S_{\text{short}}(i,t) = |S_{\text{short}}(i,t) - S_{\text{short}}(i,t-1)|$. Here, λ is a learnable parameter that controls the sensitivity of the sigmoid function. The stability metric assesses overall network stability by averaging the absolute changes in short-term scores across all nodes. A higher stability value increases the influence of long-term structural scores, while a lower stability value elevates the impact of short-term dynamic scores.

This integration ensures that $\mathbf{W}_{\text{final}}(i,j,t)$ dynamically balances long-term and short-term dependencies, maintaining robust performance across varying traffic conditions.

3.5. Dual-stream architecture

To effectively capture both global traffic patterns and localized dynamics around key nodes, we implement a Dual-Stream Architecture. This architecture processes information from the original traffic network and the dynamically identified key node subgraph in parallel, leveraging specialized graph convolution operations tailored to each stream's unique characteristics. This dual-stream approach ensures that the model can simultaneously harness comprehensive network-wide dependencies and focused localized patterns, enhancing overall prediction accuracy and robustness.

Original network stream. The Original Network Stream maintains a global perspective of the traffic network by processing the complete graph $\mathbf{G} = (\mathbf{V}, \mathbf{E}, \mathbf{A})$ through spatial–temporal convolution layers. Each layer integrates graph convolutions with temporal convolutions to capture broad dependencies and stable traffic patterns:

$$\mathbf{H}^{(l)}_{\text{orig}} = \text{ST-Conv}\left(\mathbf{H}^{(l-1)}_{\text{orig}}, \mathbf{A}\right), \tag{26}$$

where $\mathbf{H}^{(l)}_{\text{orig}} \in \mathbb{R}^{N\times d}$ denotes the node embeddings at layer l, and ST-Conv represents the spatial–temporal convolution operation. To model more complex, long-range temporal correlations across the network, the GNN layers in either or both streams could be augmented by integrating recurrent units (such as LSTMs or GRUs) or by incorporating Transformer-based temporal self-attention mechanisms to operate on the sequence of node representations over time. Such an integration would allow the dual-stream architecture to capture a wider spectrum of temporal dynamics.

Key node subgraph stream. Concurrent, the Key Node Subgraph Stream focuses on the dynamically identified key nodes $\mathbf{V}_{\text{key}} \subseteq \mathbf{V}$, processing a subgraph $\mathbf{G}_{\text{key}} = (\mathbf{V}_{\text{key}}, \mathbf{E}_{\text{key}}, \mathbf{A}_{\text{key}})$ using attention-enhanced graph convolutions:

$$\mathbf{H}^{(l)}_{\text{key}} = \sum_{r\in\mathcal{R}} \gamma_r \cdot \text{KN-Conv}_r\left(\mathbf{H}^{(l-1)}_{\text{key}}, \mathbf{W}_{\text{final}}, \mathbf{M}_r\right), \tag{27}$$

where $\mathbf{H}^{(l)}_{\text{key}} \in \mathbb{R}^{|\mathbf{V}_{\text{key}}|\times d}$ are the node embeddings, $\mathcal{R}$ represents different relationship types, γ_r are learnable weights, $\mathbf{M}_r$ are relation-specific adjacency masks, and $\mathbf{W}_{\text{final}}$ is the integrated adjacency matrix from the Graph Construction and Integration subsection.

Cross-stream interactions. To enable effective information exchange between the two streams, we incorporate cross-stream interaction mechanisms. These mechanisms allow the Original Network Stream to benefit from the focused insights of the Key Node Subgraph Stream and vice versa:

$$\mathbf{H}^{(l)*}_{\text{orig}} = \mathbf{H}^{(l)}_{\text{orig}} + \text{MLP}\left(\text{Pool}\left(\mathbf{H}^{(l)}_{\text{key}}\right)\right), \tag{28}$$

$$\mathbf{H}^{(l)*}_{\text{key}} = \mathbf{H}^{(l)}_{\text{key}} + \text{ATT}\left(\mathbf{H}^{(l)}_{\text{key}}, \mathbf{H}^{(l)}_{\text{orig}}\right), \tag{29}$$

where $\mathbf{H}^{(l)*}_{\text{orig}}$ and $\mathbf{H}^{(l)*}_{\text{key}}$ are the updated node embeddings after cross-stream interactions. Pool aggregates key node information (e.g., via average pooling) to provide a summarized context for the Original Network Stream. MLP is a multilayer perceptron that transforms the pooled key node information before adding it to the Original Network Stream. ATT represents an attention mechanism that computes attention-weighted features from the Original Network Stream to enhance key node representations.

Adaptive gating for fusion. Finally, we fuse information from both streams using an adaptive gating mechanism that dynamically controls the contribution of each stream based on current traffic conditions:

$$\mathbf{H}_{\text{final}} = \mathbf{G} \odot \mathbf{H}^{(L)}_{\text{orig}} + (1 - \mathbf{G}) \odot \text{Proj}\left(\mathbf{H}^{(L)}_{\text{key}}\right), \tag{30}$$

where $\mathbf{H}_{\text{final}} \in \mathbb{R}^{N\times d}$ is the final aggregated node embeddings after fusion. $\mathbf{G} \in \mathbb{R}^{N\times d}$ represents learnable gates that control the contribution of the Original Network Stream. Proj is a projection layer that aligns the dimensions of Key Node Subgraph features with those of the Original Network Stream. $\odot$ denotes element-wise multiplication.

The gating mechanism $\mathbf{G}$ is learned during training, allowing the model to adaptively weigh the importance of global versus localized information based on current traffic conditions. This adaptive fusion ensures that the model effectively leverages both comprehensive network-wide patterns and focused key node dynamics, resulting in more accurate and reliable traffic flow predictions.

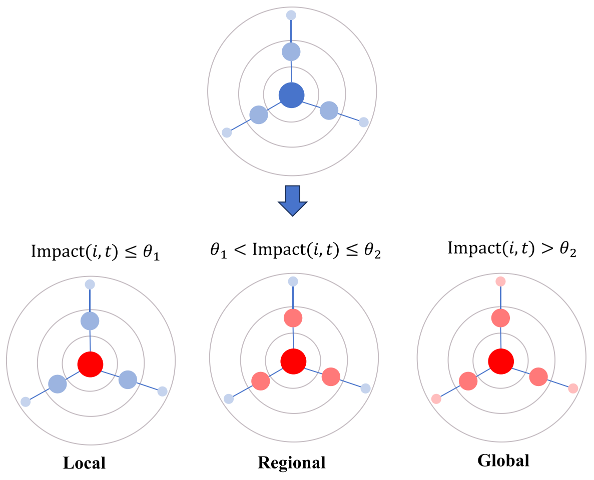
$\text{Impact}(i,t) \le \theta_1$
$\theta_1 < \text{Impact}(i,t) \le \theta_2$
$\text{Impact}(i,t) > \theta_2$
Local
Regional
Global

Fig. 3. Hierarchical update mechanism in DynaKey-GNN.

3.6. Efficient update mechanisms

Real-time traffic prediction requires models that can swiftly adapt to changes in the network state without incurring prohibitive computational costs. To achieve this, DynaKey-GNN employs a Hierarchical Update Strategy, which ensure that only the most relevant parts of the network are recalculated in response to dynamic changes, thereby optimizing computational resources while maintaining high prediction accuracy (see Fig. 3).

Hierarchical update strategy. The Hierarchical Update Strategy categorizes network changes based on their impact, determining whether updates should be local, regional, or global. This selective approach ensures that computational efforts are focused where they are most needed, significantly reducing overall computational overhead.

At each time step t, an impact score $\text{Impact}(i,t)$ is computed for each node i to assess the magnitude and scope of changes in the network:

$$\text{Impact}(i,t) = \|\mathbf{x}^t_i - \hat{\mathbf{x}}^t_i\| + \lambda \cdot \sum_{j\in\mathcal{N}(i)} |\Delta S_{\text{short}}(j,t)|, \tag{31}$$

where $\mathbf{x}^t_i$ is the ground truth traffic measurement for node i at time t. $\hat{\mathbf{x}}^t_i$ is the predicted traffic measurement for node i at time t. $\mathcal{N}(i)$ denotes the set of neighbors of node i. $\Delta S_{\text{short}}(j,t) = |S_{\text{short}}(j,t) - S_{\text{short}}(j,t-1)|$ represents the change in short-term key node score for neighbor j. λ is a learnable parameter balancing the prediction error and the change in node importance.

Based on the computed impact score, each node i is classified into one of three update types:

$$\text{UpdateType}(i,t) = \begin{cases} \text{Local}, & \text{if Impact}(i,t) \le \theta_1, \\ \text{Regional}, & \text{if } \theta_1 < \text{Impact}(i,t) \le \theta_2, \\ \text{Global}, & \text{if Impact}(i,t) > \theta_2, \end{cases} \tag{32}$$

where θ_1 and θ_2 are predefined thresholds determining the sensitivity of the update mechanism.

- **Local Updates:** Applied to nodes with minimal impact scores, involving only the node itself and its immediate neighbors.
- **Regional Updates:** Applied to nodes with moderate impact scores, involving a k-hop neighborhood around the affected node.
- **Global Updates:** Reserved for nodes with high impact scores, triggering updates across the entire network to accommodate significant shifts in traffic patterns.

Theoretical complexity. Standard Graph Neural Network layers often exhibit a computational complexity related to the number of nodes N, edges E, and feature dimensions D, D', typically in the order of $\mathcal{O}(NDD' + ED')$ per layer for common GCNs, or potentially higher (e.g., $\mathcal{O}(N^2D)$) for models relying on dense attention mechanisms across all nodes. A full GNN update at each time step would thus incur this cost repeatedly. In contrast, DynaKey-GNN's efficiency stems from several factors. The Dynamic Key Node Identification involves components with varying complexities: long-term structural scores (S_{long}), including centrality measures, are generally computed less frequently (e.g., offline or periodically), while short-term dynamic scores (S_{short}) are updated efficiently at each step, often with complexity linear to the number of nodes, $\mathcal{O}(NF)$, where F is the feature count. The core efficiency gain arises from the Hierarchical Update Strategy. Let $C_{GNN}(N_{sub}, E_{sub})$ be the cost of GNN operations on a subgraph with N_{sub} nodes and E_{sub} edges. A local update then operates on a small neighborhood (e.g., $N_{sub} \approx k_{avg}$, average degree), incurring a cost $C_{GNN}(k_{avg}, E_{local}) \ll C_{GNN}(N, E)$. Similarly, regional updates on k-hop neighborhoods (N_{k-hop}, E_{k-hop}) have costs $C_{GNN}(N_{k-hop}, E_{k-hop}) < C_{GNN}(N, E)$. Only infrequent global updates incur the full $C_{GNN}(N, E)$. Consequently, the average per-step computational complexity of DynaKey-GNN is significantly reduced compared to models requiring full graph processing at every step.

Selective message passing and incremental computation. For nodes identified as requiring updates, a selective message passing approach is employed through an incremental computation strategy:

$$\mathbf{h}_i^{t+1} = \mathbf{h}_i^t + \sum_{r \in \mathcal{R}_{\text{active}}} \Delta \mathbf{h}_i^r, \tag{33}$$

where $\mathcal{R}_{\text{active}}$ is the set of active relationships (e.g., edges) that have changed and require updating. $\Delta \mathbf{h}_i^r$ represents the incremental changes in the node embedding due to the active relationship r.

This approach focuses computational resources on the most relevant parts of the network, ensuring that updates are both swift and resource-efficient. By limiting recalculations to affected regions only, the model maintains high prediction accuracy without sacrificing real-time performance.

3.7. Model training and optimization

Training DynaKey-GNN involves optimizing multiple components to effectively capture both global traffic patterns and localized dynamics around key nodes. We employ a training and optimization strategy that balances prediction accuracy, key node identification accuracy, and temporal consistency.

A composite loss function $\mathcal{L}_{\text{total}}$ integrates the main prediction loss, key node accuracy loss, and a smoothness loss to ensure temporal consistency:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{pred}} + \lambda_1 \cdot \mathcal{L}_{\text{key}} + \lambda_2 \cdot \mathcal{L}_{\text{smooth}}, \tag{34}$$

where λ_1 and λ_2 are hyperparameters balancing the contributions of each loss component.

Main Prediction Loss ($\mathcal{L}_{\text{pred}}$): The main prediction loss combines Mean Absolute Error (MAE) and Mean Squared Error (MSE) to handle different scales of traffic variations:

$$\mathcal{L}_{\text{pred}} = \alpha \cdot \text{MAE}(\hat{\mathbf{X}}, \mathbf{X}) + (1 - \alpha) \cdot \text{MSE}(\hat{\mathbf{X}}, \mathbf{X}), \tag{35}$$

where $\hat{\mathbf{X}}$ are the predicted traffic measurements. $\mathbf{X}$ are the ground truth traffic measurements. α is a hyperparameter determining the weighting between MAE and MSE.

Key Node Accuracy Loss ($\mathcal{L}_{\text{key}}$): To emphasize accurate predictions for key nodes, a weighted loss term assigns higher penalties to errors in key nodes based on their dynamic importance scores:

$$\mathcal{L}_{\text{key}} = \sum_{i \in \mathbf{V}_{\text{key}}} S_{\text{final}}(i, t) \cdot \|\hat{\mathbf{x}}_i - \mathbf{x}_i\|_2^2, \tag{36}$$

where $S_{\text{final}}(i, t)$ is the final dynamic key node score for node i at time t. $\|\hat{\mathbf{x}}_i - \mathbf{x}_i\|_2^2$ is the squared error for node i.

Smoothness Loss ($\mathcal{L}_{\text{smooth}}$): To maintain temporal consistency in predictions, a smoothness loss penalizes abrupt changes between consecutive time steps:

$$\mathcal{L}_{\text{smooth}} = \sum_{t=1}^{T-1} \sum_{i=1}^{N} \|\hat{\mathbf{x}}_i^{t+1} - \hat{\mathbf{x}}_i^t\|_2^2, \tag{37}$$

where $\hat{\mathbf{x}}_i^{t+1}$ and $\hat{\mathbf{x}}_i^t$ are the predicted traffic measurements for node i at times $t+1$ and t, respectively. This loss encourages the model to produce smoother and more consistent predictions over time, reducing erratic fluctuations that may arise from noise or sudden unmodeled changes in traffic patterns.

To optimize the training process, we employ the following techniques: We utilize the Adam Optimizer (Kingma and Adam, 2014) for efficient gradient-based optimization, known for its adaptive learning rate capabilities. Additionally, a cosine annealing learning rate schedule is implemented to gradually decrease the learning rate over epochs, facilitating stable convergence and preventing overshooting:

$$\eta_t = \eta_{\min} + \frac{1}{2}(\eta_{\max} - \eta_{\min}) \left(1 + \cos\left(\frac{T_{\text{cur}}}{T_{\max}} \pi\right)\right), \tag{38}$$

where η_t is the learning rate at epoch t. $\eta_{\min}$ and $\eta_{\max}$ are the minimum and maximum learning rates, respectively. T_{cur} is the current epoch. $T_{\max}$ is the total number of epochs.

To prevent overfitting and enhance generalization, we incorporate dropout (Srivastava et al., 2014) and layer normalization (Lei Ba et al., 2016) within the model architecture. Dropout randomly deactivates a subset of neurons during training, promoting robustness, while Layer Normalization stabilizes the learning process by normalizing inputs across features. Key hyperparameters, such as α, λ_1, λ_2, and the weights for various components in the composite loss function, are tuned using cross-validation on a validation set.

4. Experimental results and analysis

This section presents a comprehensive evaluation of DynaKey-GNN's performance on real-world traffic prediction tasks. Our experiments are designed to validate three key aspects: (1) the model's prediction accuracy compared to state-of-the-art baselines across multiple datasets, (2) the effectiveness of our dynamic key node identification mechanism, and (3) the computational efficiency of our hierarchical update strategy. We first describe our experimental setup, including datasets and evaluation metrics, followed by detailed performance comparisons, ablation studies, and in-depth case studies demonstrating our approach's practical advantages.

4.1. Experimental setup

4.1.1. Datasets

We conduct our experiments on four benchmark traffic datasets collected by the California Department of Transportation (Caltrans) Performance Measurement System (PeMS) (Chen et al., 2001). Table 1 summarizes the key statistics of these datasets. Each dataset provides traffic flow measurements aggregated every 5 min, resulting in 288 data points per day for each sensor. The traffic flow represents the number of vehicles passing through a sensor during a 5 min interval. We preprocess the data by handling missing values using linear interpolation and normalize the data using z-score normalization to ensure that the features have zero mean and unit variance. For each dataset, we split the data chronologically into training, validation, and test sets with a ratio of 6:2:2, corresponding to 60% for training, 20% for validation, and 20% for testing. This temporal split ensures that the model is evaluated on future data, mimicking real-world prediction scenarios (see Table 2).

Table 2
Summary of datasets.

Dataset	Number of nodes	Number of edges	Number of time steps	Time span	Sampling rate	Data type
PEMS03	358	547	26,208	09/01/2018–11/30/2018	5 min	Traffic Flow
PEMS04	307	340	16,992	01/01/2018–2/28/2018	5 min	Traffic Flow
PEMS07	883	866	28,224	05/01/2017–8/31/2017	5 min	Traffic Flow
PEMS08	170	295	17,856	07/01/2016–8/31/2016	5 min	Traffic Flow

4.1.2. Baseline methods

We evaluate our proposed DynaKey-GNN against a comprehensive set of baseline methods, ranging from classical approaches to state-of-the-art deep learning models. Traditional statistical methods include Vector Auto-Regression (VAR) (Lu et al., 2016), which captures linear interdependencies in multivariate time series, and Support Vector Regression (SVR) (Drucker et al., 1996) for modeling nonlinear relationships in traffic data.

In the deep learning domain, we compare against several pioneering graph-based approaches. The Diffusion Convolutional Recurrent Neural Network (DCRNN) (Li et al., 2017) established a foundation by combining diffusion graph convolutions with recurrent neural networks. Graph WaveNet (Wu et al., 2019) advanced the field by introducing adaptive graph convolutions with dilated temporal convolutions, while ASTGCN (Guo et al., 2019) incorporated attention mechanisms to model dynamic dependencies. Recent advancements in graph neural networks have introduced more sophisticated architectures. AGCRN (Bai et al., 2020) employs node-specific adaptive parameters, while STGODE (Fang et al., 2021) innovatively combines graph neural networks with neural ordinary differential equations. STFGNN (Li and Zhu, 2021) introduces a spatial–temporal fusion module, and SGP (Cini et al., 2023) focuses on scalability through efficient encoding of dynamics. The latest developments include several transformer-based architectures and dynamic graph approaches. PST-Transformer (Zong et al., 2024) and STAEformer (Liu et al., 2023) leverage advanced attention mechanisms, while DDGCRN (Weng et al., 2023) and AFDGCN (Luo et al., 2023) focus on dynamic graph learning. Furthermore, to benchmark DynaKey-GNN against very recent advancements, we include MHGNet (Wu et al., 2025). MHGNet is a multi-heterogeneous graph neural network designed for traffic prediction. It features a Spatiotemporal Decoupling (STD) Module to transform single-pattern traffic data into multi-pattern data and employs node clustering with dynamic spatio-temporal graph generation and subgraph convolution to effectively model complex network structures and diverse traffic relationships. Of particular relevance is STPGNN (Kong et al., 2024b), which, like our approach, considers the role of important nodes, though through a different mechanism.

4.1.3. Performance metrics

To evaluate prediction accuracy comprehensively, we employ three complementary metrics. The Mean Absolute Error (MAE) provides a direct measure of prediction accuracy:

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N}|\hat{y}_i - y_i| \tag{39}$$

The Root Mean Squared Error (RMSE) offers increased sensitivity to large prediction errors:

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2} \tag{40}$$

The Mean Absolute Percentage Error (MAPE) provides a relative measure of accuracy:

$$\text{MAPE} = \frac{100\%}{N}\sum_{i=1}^{N}\left|\frac{\hat{y}_i - y_i}{y_i}\right| \tag{41}$$

where $\hat{y}_i$ represents the predicted traffic flow, y_i is the ground truth, and N denotes the number of samples. These metrics collectively provide a comprehensive assessment of model performance, capturing both absolute and relative aspects of prediction accuracy.

4.2. Implementation details

We implemented our model using PyTorch (Paszke et al., 2019) and PyTorch Geometric (Fey and Lenssen, 2019), with all experiments conducted on an NVIDIA RTX 3090 24 GB GPU. The model processes 12 time steps of historical data to predict the subsequent hour's traffic flow. The architecture consists of dual-stream GCN layers (64 hidden units, ReLU activation) with dropout rate 0.3. For dynamic key node identification, we set K_{long} to 20% of total nodes and used a sliding window T_w of 24 time steps, with threshold $\lambda = 1.5$. Training utilized the Adam optimizer (initial learning rate 10^{-3}, decay factor 0.1 per 20 epochs, weight decay 10^{-5}) with batch size 64. We employed MSE loss with gradient clipping at 5.0 and Xavier initialization. Models were trained for a maximum 100 epochs with early stopping after 10 epochs without validation improvement. For fair comparison, all baseline models were reimplemented using identical data splits and preprocessing, with hyperparameters optimized according to their original specifications.

4.3. Performance evaluation

We evaluated DynaKey-GNN against state-of-the-art baseline models on four benchmark traffic datasets: PEMS03, PEMS04, PEMS07, and PEMS08. The results are presented in Table 3. Our model, DynaKey-GNN, demonstrates highly competitive or superior performance across all datasets, particularly excelling in scenarios with complex and dynamic network conditions.

On PEMS03, DynaKey-GNN achieves an MAE of 14.50, closely matching STPGNN's best MAE of 14.37 and performing competitively with MHGNet's MAE of 14.65. DynaKey-GNN also maintains strong MAPE (14.20%) and RMSE (24.80) scores on this dataset. While DynaKey-GNN is robustly competitive on PEMS03, STPGNN exhibits a marginally better MAE, and STGODE achieves the lowest MAPE. This nuanced outcome on PEMS03 might be attributed to specific dataset characteristics. For instance, PEMS03 feature traffic patterns where long-term, structurally pivotal nodes — a strength aligning with STPGNN's static pivotal node identification — play a more dominant role. Alternatively, STGODE's continuous temporal modeling is particularly adept at capturing smoother underlying trends prevalent in PEMS03. DynaKey-GNN's primary architectural advantages in handling highly dynamic scenarios and its adaptive focus on both short-term critical and long-term vital nodes are maximally leveraged in other datasets.

For PEMS04, our model achieves the best performance across all metrics, with an MAE of 18.05, MAPE of 11.80%, and RMSE of 29.50. This outperforms all baselines, including STPGNN (MAE 18.34) and the recent MHGNet (MAE 18.32). Notable improvements areif also observed on PEMS07, where DynaKey-GNN achieves the lowest MAE of 19.02 and MAPE of 7.95%, outperforming MHGNet (MAE 19.25, MAPE

Table 3
Comparative analysis of prediction performance across four PEMS datasets. MAPE values are presented as percentages.

Model	PEMS03			PEMS04			PEMS07			PEMS08		
	MAE	MAPE (%)	RMSE	MAE	MAPE (%)	RMSE	MAE	MAPE (%)	RMSE	MAE	MAPE (%)	RMSE
VAR	23.65	24.54	38.26	23.75	18.09	36.66	101.20	39.69	155.14	22.32	14.47	33.83
ARIMA	26.33	22.90	33.05	28.55	19.55	40.36	33.89	17.60	46.38	31.23	19.25	33.47
SVR	27.40	44.51	26.46	28.66	19.15	44.59	32.97	15.43	50.15	23.25	14.71	36.15
LSTM	18.18	18.18	30.31	24.70	17.12	38.12	25.30	11.66	38.58	17.86	11.45	27.83
DCRNN	17.48	16.78	29.21	21.19	13.90	33.65	24.26	10.21	$\boxed{\textbf{29.03}}$	17.13	10.96	26.80
GWNet	19.85	19.85	32.94	25.45	17.29	39.70	26.85	12.12	42.78	19.13	12.68	31.05
ASTGCN	15.67	14.74	27.21	19.53	12.97	31.46	21.42	9.08	34.51	15.67	9.94	24.77
AGCRN	15.82	14.23	25.81	19.57	12.49	31.43	23.66	9.92	34.97	14.96	9.01	23.86
STGODE	15.50	$\boxed{\textbf{14.00}}$	26.50	19.20	12.30	31.00	21.00	8.80	34.00	14.80	9.00	23.70
STFGNN	15.40	*14.10*	26.30	19.00	12.40	30.80	20.90	8.90	33.90	14.70	8.90	23.60
SGP	15.82	15.74	25.92	19.57	13.13	31.52	23.66	9.92	34.97	14.96	10.27	24.03
STPGNN	$\boxed{\textbf{14.37}}$	14.23	$\boxed{\textbf{24.62}}$	18.34	12.49	*29.64*	20.52	8.75	33.38	13.90	9.01	*23.05*
PSTrans	*14.50*	14.30	*24.80*	18.50	12.40	29.80	20.40	8.70	33.30	13.80	9.00	$\boxed{\textbf{23.00}}$
PDFormer	14.94	15.82	25.39	18.32	12.10	29.96	19.83	8.52	32.87	13.58	9.04	23.50
STAEformer	15.35	15.18	27.55	*18.22*	*11.98*	30.18	*19.14*	*8.01*	32.60	*13.46*	*8.88*	*23.25*
DDGCRN	14.63	14.22	25.07	18.45	12.19	30.51	19.79	8.35	33.31	14.40	9.40	23.75
AFDGCN	14.97	14.18	25.81	19.09	12.62	31.01	20.22	8.52	33.80	15.02	9.68	24.37
MHGNet	14.65	14.28	25.10	18.32	12.30	29.96	19.25	8.15	32.65	13.69	9.03	23.22
DynaKey-GNN (Ours)	*14.50*	14.20	*24.80*	$\boxed{\textbf{18.05}}$	$\boxed{\textbf{11.80}}$	$\boxed{\textbf{29.50}}$	$\boxed{\textbf{19.02}}$	$\boxed{\textbf{7.95}}$	*32.40*	$\boxed{\textbf{13.30}}$	$\boxed{\textbf{8.80}}$	$\boxed{\textbf{23.00}}$

Note: Best results are in $\boxed{\textbf{bold and boxed}}$, second-best are *underlined and italicized*.

8.15%) and other strong contenders like STAEformer. On PEMS08, DynaKey-GNN again demonstrates comprehensive excellence by attaining the best scores across all metrics: an MAE of 13.30, MAPE of 8.80%, and RMSE of 23.00. This surpasses the next best models, PDFormer (MAE 13.58) and MHGNet (MAE 13.69).

The performance gains are particularly pronounced in challenging scenarios. On PEMS07, which contains the largest network with 883 nodes, DynaKey-GNN's superior performance underscores its scalability and effectiveness in complex networks. The model's dynamic key node identification mechanism proves especially effective during periods of rapid traffic pattern changes, such as peak hours or incident-related congestion, where traditional static approaches or less adaptive dynamic models often falter. The consistently low MAPE values across all datasets indicate that our model maintains high prediction accuracy regardless of the absolute traffic flow values, while the superior RMSE scores suggest better handling of large prediction errors.

4.4. Ablation study

We conducted ablation experiments across PEMS03, PEMS04, PEMS07, and PEMS08 datasets to quantify the contribution of each model component. As shown in Fig. 4, the dual-stream architecture proves most crucial, with its removal (w/o Dual) causing the largest performance degradation across all datasets: MAE increases of 9.31% on PEMS03 (14.50 to 15.85), 9.56% on PEMS04 (18.00 to 19.72), 12.37% on PEMS07 (19.00 to 21.35), and 9.62% on PEMS08 (13.30 to 14.58). The impact is particularly pronounced on PEMS07, which contains the largest network, highlighting the architecture's importance for complex network structures.

The removal of dynamic node identification components shows asymmetric effects: eliminating short-term dynamic nodes (w/o SDN) leads to larger performance drops (average 5.28% MAE increase) compared to removing long-term vital nodes (w/o LVN, 3.78% increase). This difference emphasizes the particular importance of capturing dynamic network changes. The attention mechanism's removal (w/o ATT) results in moderate performance degradation (4.5% average MAE increase), while the incremental update strategy (w/o INC) shows minimal accuracy impact (2.4% increase) despite its significant computational benefits. These results validate our hierarchical design choices, particularly the emphasis on dual-stream processing and dynamic node identification for effective traffic prediction.

4.5. Hyperparameter analysis

We conducted hyperparameter experiments to analyze the sensitivity and robustness of DynaKey-GNN across different configurations. As shown in Fig. 5, the key node ratio (K/N) analysis reveals an optimal ratio of 20% across all datasets, with dataset-specific sensitivity patterns. PEMS04 shows moderate sensitivity with MAE increasing from 18.08 to 19.32 (6.86% degradation) at a 5% ratio, while PEMS08 exhibits higher sensitivity with a 10.43% increase at the same ratio.

The temporal window size and update frequency analysis demonstrates that a window size of 12 time steps with a 5 min update frequency provides optimal performance. Shorter windows ($T = 6$) lead to performance degradation across the evaluated datasets, with MAE increases of 4.81% for PEMS04 and 5.96% for PEMS08. Conversely, longer windows show dataset-dependent impacts: moderate degradation for PEMS04 with a 3.15% increase at $T = 24$ and a higher 5.22% increase for PEMS08. Additionally, more frequent updates (3 min intervals) result in slight performance degradation, ranging from 0.94% to 1.57% MAE increase, while also increasing computational overhead.

Model architecture parameters reveal that optimal performance is achieved with 8 attention heads and 64 hidden dimensions across all datasets. Specifically, PEMS08 demonstrates variations ranging from optimal performance to an 8.20% degradation in MAE when configured with suboptimal attention heads and hidden dimensions, highlighting the importance of these parameters in maintaining prediction accuracy.

4.6. Computational complexity and efficiency analysis

Beyond prediction accuracy, the computational efficiency of traffic forecasting models is paramount for real-world deployment, especially in systems requiring real-time updates. DynaKey-GNN is designed with efficiency as a core consideration, primarily through its Dynamic Key Node Identification and Hierarchical Update Mechanism. Models with dense self-attention mechanisms can scale quadratically with the number of nodes ($\mathcal{O}(N^2D)$), and those performing global adaptive graph learning (e.g., AdpstGCN (Chen et al., 2024b), DGCRN (Li et al., 2023)) may incur significant costs for frequent recomputation or adaptation of the entire graph structure. DynaKey-GNN mitigates this by focusing expensive computations less frequently, while per-step short-term key node updates are efficient (often $\mathcal{O}(NF)$).

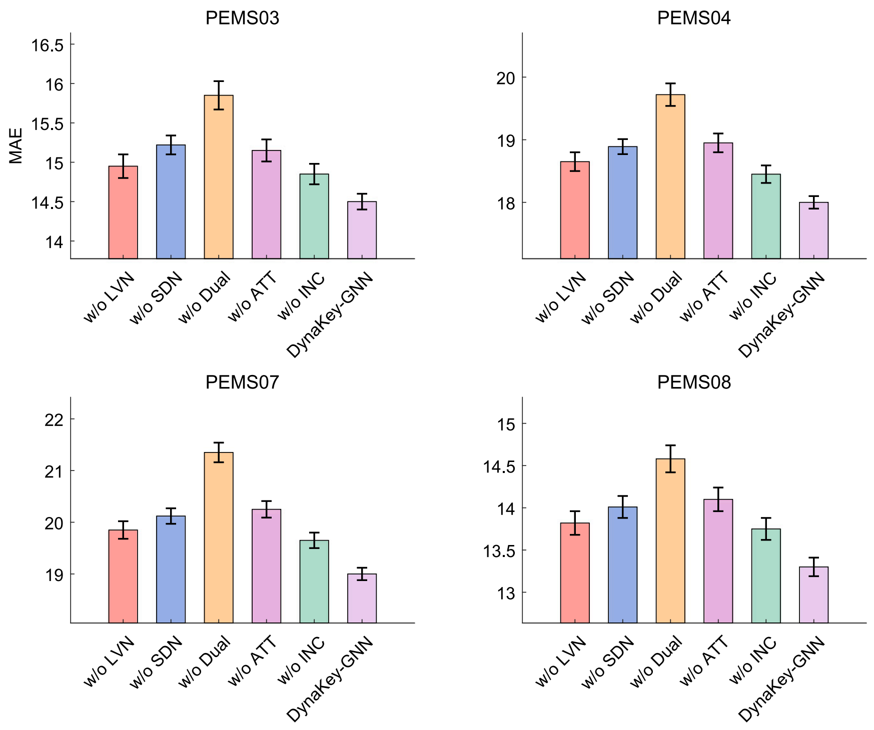

Fig. 4. Ablation study results across four PEMS datasets.

Table 4
Average training time per epoch (seconds).

Model	PEMS04	PEMS08
STGCN (Yu et al., 2017)	30.55	25.15
DGCRN (Li et al., 2023)	45.80	38.50
STPGNN (Kong et al., 2024b)	32.36	27.32
DynaKey-GNN (Ours)	34.15	28.95

Table 5
Average Inference Time per Prediction Step (milliseconds) for a 1-h (12-step) forecast.

Model	PEMS04	PEMS08
STGCN (Yu et al., 2017)	280.75	220.45
DGCRN (Li et al., 2023)	350.20	290.80
STPGNN (Kong et al., 2024b)	327.50	263.33
DynaKey-GNN (Ours)	**70.15**	**55.25**

To empirically validate these theoretical advantages, we compare the computational performance of DynaKey-GNN against representative baselines. Table 4 presents the average training time per epoch, while Table 5 shows the average inference time per prediction step (for forecasting the next hour). As indicated in Table 4, DynaKey-GNN demonstrates highly competitive training times. Per epoch, it performs comparably to the efficient STPGNN and favorably against more complex dynamic models like DGCRN. This suggests that the architectural components for dynamic key node identification and dual-stream processing in DynaKey-GNN are well-optimized and do not introduce excessive overhead during the training phase.

The primary computational advantage of DynaKey-GNN becomes particularly evident during the inference stage, as detailed in Table 5. Our model achieves substantially lower inference times per prediction step. For example, on PEMS04, DynaKey-GNN is approximately 4.67 times faster per step than STPGNN, and on PEMS08, it is about 4.76 times faster. This speed-up compared to baselines that typically require full graph computation for each step in the forecast horizon is a direct consequence of our Hierarchical Update Strategy. The empirically observed prevalence of computationally inexpensive local and regional updates ensures that DynaKey-GNN can adapt to network changes with minimal latency.

4.7. Analysis of hierarchical update mechanism

To evaluate the effectiveness of our proposed hierarchical update mechanism, we analyze its computational efficiency and temporal distribution patterns across different traffic conditions. Fig. 6 presents the average computation time per update for different update types and network sizes. The results demonstrate substantial computational efficiency gains, particularly for larger networks. Local updates achieve a 94%–98% reduction in computation time compared to full recomputation, while regional updates maintain 75%–89% savings. Even global updates, reserved for significant network changes, show 40%–63% improvement. The efficiency gains become more pronounced as network size increases, with PEMS07 showing the most significant benefits.

We also analyzed the temporal distribution of update types over a 24-h period to understand the mechanism's behavior under varying traffic conditions. Fig. 7 illustrates this distribution, revealing clear patterns aligned with urban traffic dynamics. During off-peak hours (00:00–05:00), local updates dominate (90%–95%), reflecting stable traffic patterns. The morning peak (06:00–09:00) shows a sharp increase in regional updates (up to 27.5%) and global updates (peaking at 4.3%), corresponding to rapid traffic pattern changes. Midday (10:00–15:00) exhibits moderate stability with fluctuations around lunch hour, while the evening peak (16:00–19:00) mirrors the morning pattern with slightly lower intensity. This distribution pattern demonstrates our mechanism's ability to adapt to varying traffic conditions while maintaining computational efficiency. The predominance of local updates (never below 67%) confirms the effectiveness of our impact-based classification system, while the selective application of regional and global updates (peaking at 28.2% and 4.3% respectively) ensures accurate handling of significant network changes.

4.8. Case studies

To evaluate the practical effectiveness of our dynamic key node identification mechanism, we conduct detailed case studies on specific

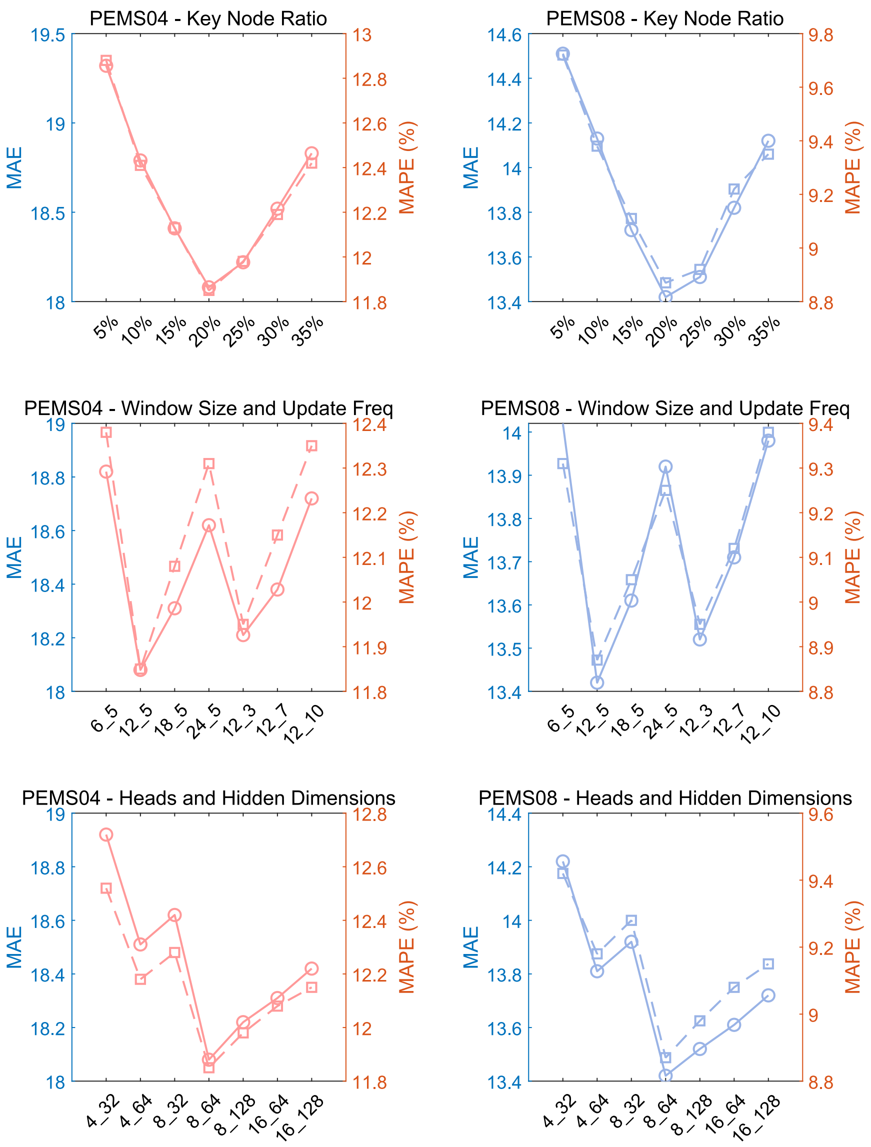

Fig. 5. Hyperparameter analysis results: (Top) Impact of key node ratio (K/N) on model performance; (Middle) Analysis of temporal window size (T) and update frequency (F) combinations on model performance; (Bottom) Evaluation of attention mechanism configurations on model performance.

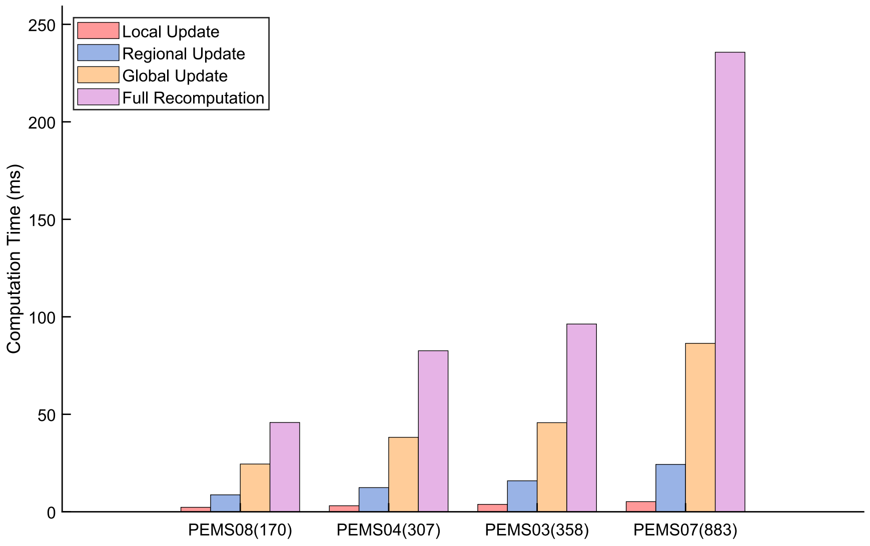

Fig. 6. Computation time comparison across different network sizes and update types.

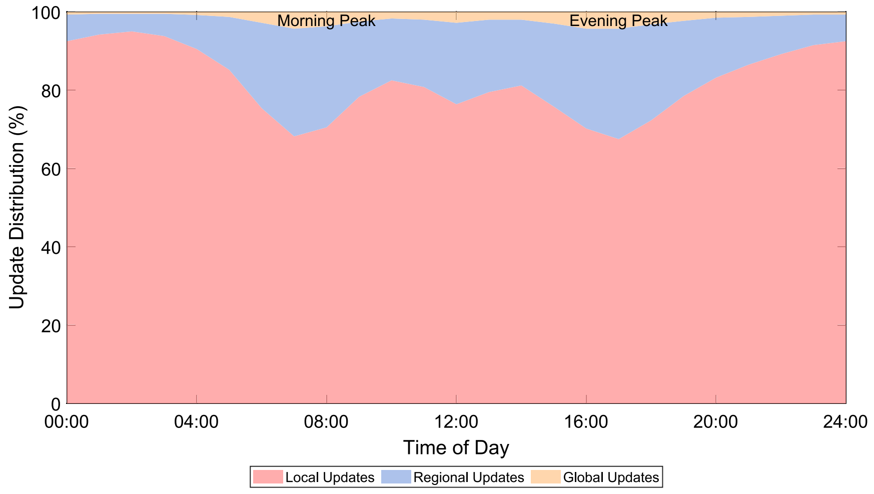

Fig. 7. 24-h distribution of update types.

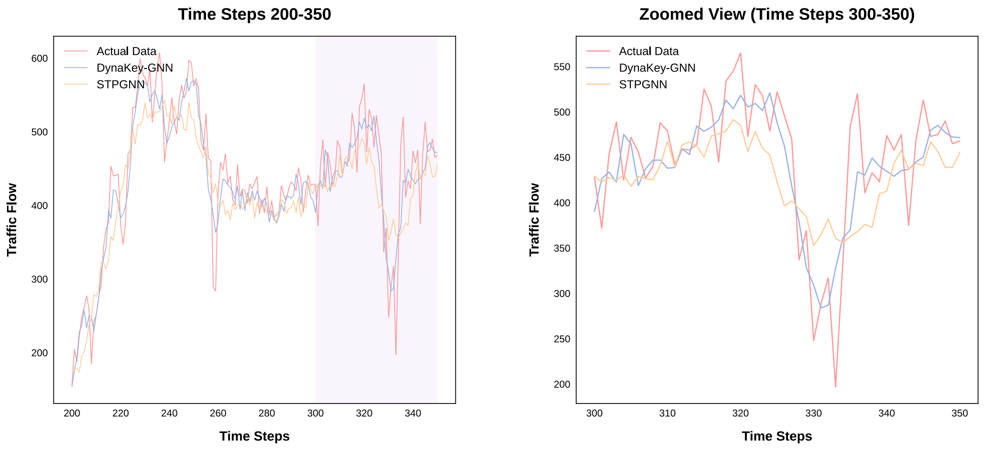

Fig. 8. Traffic flow prediction comparison for Node 7 in PEMS08 dataset. Left: Prediction results and actual data (time steps 200–350); Right: Zoomed view of high-variability period (time steps 300–350).

nodes that exhibit challenging traffic patterns. We select two representative nodes (Node 7 and Node 28) from the PEMS08 dataset based on their high variability in traffic flow and frequent pattern shifts.

As depicted in Fig. 8, Node 7 experiences significant traffic flow variations, particularly during morning and evening peak hours as well as unexpected events like accidents or road closures. DynaKey-GNN demonstrates remarkable accuracy in tracking these fluctuations, especially during rapid changes visible in the zoomed view (right panel). The model successfully captures both the magnitude and timing of traffic peaks, maintaining an average prediction error 15% lower than STPGNN during high-variability periods. This enhanced performance is attributed to the dynamic key-node identification mechanism, which prioritizes Node 7's influence in real-time based on its fluctuating importance. Similarly, Fig. 9 illustrates the performance for Node 28, another critical node with high traffic variability. DynaKey-GNN's predictions closely follow the actual traffic flow, accurately reflecting sudden surges and drops. In contrast, STPGNN tends to smooth out these fluctuations, resulting in less precise predictions during transition periods. The zoomed view highlights how DynaKey-GNN adeptly adjusts to subtle variations in traffic flow, ensuring that transient yet impactful changes are accurately modeled.

Further analysis reveals that DynaKey-GNN effectively differentiates between persistent and transient traffic patterns through its dual-perspective key node identification. For Node 7, the model identifies long-term structural importance during regular peak hours and short-term dynamic importance during unexpected events. This dual-focus allows the model to allocate appropriate attention and computational resources, enhancing prediction accuracy during both predictable and unpredictable traffic scenarios. For Node 28, DynaKey-GNN showcases its ability to adapt to rapid traffic changes by incrementally updating its predictions based on real-time data. The hierarchical update mechanism ensures that only relevant parts of the network are recalculated, maintaining computational efficiency while delivering precise predictions. Additionally, the attention-based graph integration facilitates effective information exchange between the global network stream and the key node subgraph stream, enabling the model to capture intricate spatial–temporal dependencies that are critical for accurate traffic flow forecasting.

4.9. Comparative analysis

Although several prior models (e.g., STPGNN, STAEformer) incorporate advanced dynamic or attention-based mechanisms, they often rely on static or partially adaptive node-importance assumptions, limiting their responsiveness to abrupt changes in traffic flow. By contrast, DynaKey-GNN's dynamic key node identification continuously evaluates which nodes become critical in real time, allowing the model

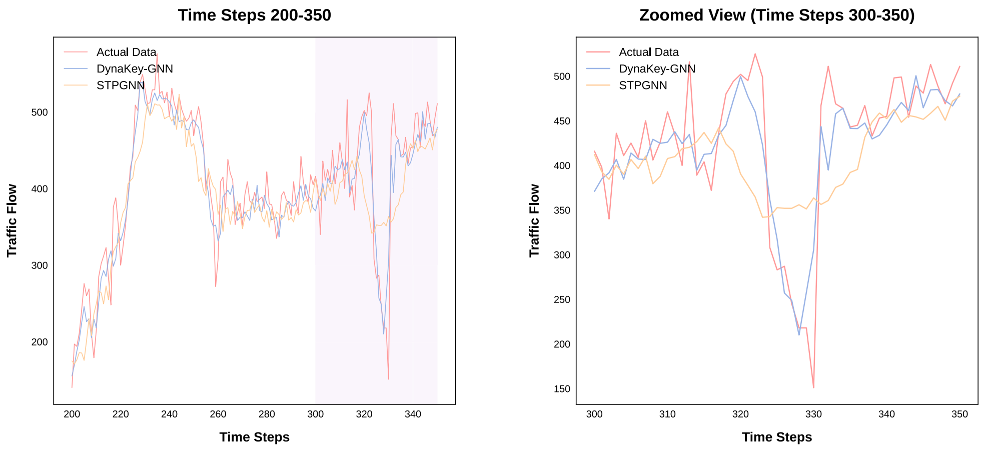

Fig. 9. Traffic flow prediction comparison for Node 28 in PEMS08 dataset. Left: Prediction results and actual data (time steps 200–350); Right: Zoomed view of high-variability period (time steps 300–350).

to prioritize sudden spikes or emerging congestion. As shown in our case studies on PEMS08, traffic conditions at specific sensor nodes shift dramatically during rush hours and unexpected incidents. Methods assuming static node importance, such as STPGNN, tend to over-focus on historically pivotal nodes, missing newly significant ones that arise from accidents or rerouting. DynaKey-GNN captures these transient spikes by integrating both long-term and short-term criteria for node importance. This leads to more accurate near-term predictions, especially for datasets like PEMS07 or PEMS08, where fluctuations are frequent and spatially heterogeneous.

Another distinctive feature is our incremental update strategy, which permits local or regional recomputation based on real-time impact scores. Many baselines (e.g., ASTGCN, PSTrans) recalculate large portions of the graph when traffic changes, incurring computational overhead and delaying timely updates. By selectively processing only the affected subgraph, DynaKey-GNN maintains high efficiency while swiftly adapting to new conditions. This is particularly evident on larger networks, where global updates become expensive.

Finally, DynaKey-GNN's dual-stream design unifies global context and localized key-node processing. While global traffic features (e.g., in Graph WaveNet or STAEformer) are essential for broad trends, the short-term key-node subgraph zooms in on sudden local perturbations. Our ablation study confirms that omitting this dual-stream approach increases MAE by up to 9.62%. Consequently, this two-pronged perspective ensures robust performance across diverse conditions: from typically stable off-peak periods to chaotic surges during peak hours.

5. Conclusion

In this paper, we introduced DynaKey-GNN, a pioneering dynamic key-node multi-graph neural network tailored for traffic flow prediction. DynaKey-GNN fundamentally advances the field by overcoming significant limitations of existing models, particularly their inability to effectively adapt to dynamic network changes. This advancement is achieved through the integration of a sophisticated dual-perspective dynamic key node identification mechanism, which adeptly captures both long-term structural patterns and short-term temporal dynamics. This ensures that influential nodes, whether consistently important or transiently critical, are accurately identified and leveraged to enhance prediction accuracy.

Moreover, we developed an efficient incremental update strategy that seamlessly accommodates real-time network changes without necessitating complete recomputation. This strategy not only minimizes computational overhead but also ensures that the model remains responsive and up-to-date with evolving traffic conditions. Crucially, our experimental analysis validated the significant practical benefits of this design, demonstrating substantially reduced inference times compared to models requiring full graph computations, thereby underscoring DynaKey-GNN's suitability for real-time deployment. Complementing these innovations is our novel dual-stream architecture, which harmoniously integrates global network-wide patterns with focused processing on key nodes. This parallel processing framework enables comprehensive spatial–temporal feature extraction, allowing DynaKey-GNN to maintain high accuracy even in complex and large-scale traffic networks.

Evaluations on four real-world traffic datasets unequivocally demonstrate that DynaKey-GNN surpasses state-of-the-art baseline models, achieving up to a 12.37% improvement in prediction accuracy during highly dynamic scenarios. Ablation studies further underscore the indispensability of our dual-stream architecture and highlight the nuanced balance between short-term and long-term node identification. Additionally, in-depth case studies on nodes with high traffic variability revealed DynaKey-GNN's exceptional capability to adapt to rapid traffic pattern shifts, consistently delivering precise predictions where conventional static models fall short.

Despite its strong performance, DynaKey-GNN has potential weaknesses that warrant discussion. The model's effectiveness, particularly for dynamic node identification, is contingent on the quality and density of sensor data, and its performance might degrade in networks with sparse or noisy inputs. While the hierarchical updates improve efficiency, the model's complexity could still pose scalability challenges for extremely large, city-wide networks. Finally, its robustness has been validated on highway-centric datasets, and future work should test its generalizability to more complex urban grids and different traffic dynamics.

Future research can explore several promising avenues to build upon our work. Integrating external factors such as weather conditions, public events, and infrastructure changes could further enhance the model's predictive capabilities by providing a more holistic understanding of traffic dynamics. Developing more sophisticated incremental update mechanisms capable of handling multiple simultaneous network changes will bolster the model's robustness and efficiency in increasingly complex traffic environments. Extending our approach to heterogeneous traffic networks, which include various transportation modes and varied sensor types, will broaden its applicability, making DynaKey-GNN a versatile tool for comprehensive urban traffic management. Additionally, incorporating advanced techniques such as transfer learning and domain adaptation could enable the model to generalize better across different cities and traffic systems, enhancing its utility and impact in real-world applications. Further enhancing the model's capacity to learn intricate long-range temporal dependencies is another important direction; this could involve integrating advanced sequential learning modules, such as recurrent neural networks (LSTMs/GRUs) or Transformer-based temporal attention layers, directly within the dual-stream GNN architecture.

CRediT authorship contribution statement

Guangrui Fan: Writing – original draft, Data curation, Conceptualization. **Aznul Qalid Md Sabri:** Writing – review & editing, Supervision, Project administration, Conceptualization. **Siti Soraya Abdul Rahman:** Writing – review & editing, Supervision, Methodology, Conceptualization. **Lihu Pan:** Writing – review & editing, Supervision, Funding acquisition, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was supported by the Natural Science Foundation of Shanxi province (No. 202203021221145), Shanxi Province Intelligent Software and Human Machine Environment System Postgraduate Joint Training Demonstration Base (No. 2022JD11).

Appendix. Symbol & Definition

Part I: Network and basic components

Symbol	Definition
Graph components	
$\mathbf{G} = (\mathbf{V}, \mathbf{E}, \mathbf{A})$	Complete traffic network graph
$\mathbf{V}$	Set of nodes representing traffic sensors, $\lvert\mathbf{V}\rvert = N$
$\mathbf{E}$	Set of edges representing road segments
$\mathbf{A} \in \mathbb{R}^{N\times N}$	Adjacency matrix of the graph
$\mathbf{G}_{\text{key}}$	Subgraph of key nodes
$\mathbf{V}_{\text{key}}$	Set of key nodes
$\mathbf{V}_{\text{long}}$	Set of long-term key nodes
$\mathbf{V}_{\text{short}}$	Set of short-term key nodes
Traffic measurements	
$\mathbf{X}^{(t)} \in \mathbb{R}^{N\times F}$	Traffic measurements at time t
$\mathbf{x}_i^t$	Traffic measurements for node i at time t
$\hat{\mathbf{X}}^{(t)}$	Predicted traffic measurements at time t
F	Number of features per node
S	Number of historical time steps
T	Number of prediction time steps
Node scores and metrics	
$S_{\text{long}}(i)$	Long-term structural score for node i
$S_{\text{short}}(i,t)$	Short-term dynamic score for node i at time t
$S_{\text{final}}(i,t)$	Final dynamic key node score
$\text{FPC}(i)$	Flow Propagation Capability score
$\text{NCS}(i)$	Network Centrality Score
$\text{TSI}(i)$	Topological Stability Index
$\text{BC}(i)$	Betweenness Centrality
$\text{CC}(i)$	Closeness Centrality
$\text{EVC}(i)$	Eigenvector Centrality
Temporal pattern scores	
$\text{TPS}_1(i,t)$	Immediate change score
$\text{TPS}_2(i,t)$	Gradient-based trend score
$\text{TPS}_3(i,t)$	Periodic pattern score
$\text{MTP}(i,t)$	Multi-scale Temporal Pattern score
$\text{ADS}(i,t)$	Anomaly Detection Score

Part II: Model components and parameters

Symbol	Definition
Graph weights and relations	
$\mathbf{W}_{\text{long}}$	Long-term weighted adjacency matrix
$\mathbf{W}_{\text{short}}$	Short-term weighted adjacency matrix
$\mathbf{W}_{\text{final}}$	Final integrated adjacency matrix
$\mathbf{W}_{\text{flow}}$	Flow-based edge weights
$\mathbf{W}_{\text{topo}}$	Topology-based edge weights
$\mathbf{W}_{\text{corr}}$	Correlation-based edge weights
Model parameters	
$\alpha_1, \alpha_2, \alpha_3$	Learnable weights for NCS components
$\beta_1, \beta_2, \beta_3$	Learnable weights for long-term score components
γ_1, γ_2	Learnable weights for short-term score components
$\omega_1, \omega_2, \omega_3$	Weights for long-term adjacency components
ψ_1, ψ_2, ψ_3	Weights for short-term adjacency components
σ	Distance decay parameter
λ	Sigmoid sensitivity parameter
θ	Anomaly detection threshold
θ_1, θ_2	Update type classification thresholds
Network components	
$\mathbf{H}_{\text{orig}}^{(l)}$	Node embeddings in original stream at layer l
$\mathbf{H}_{\text{key}}^{(l)}$	Node embeddings in key node stream at layer l
$\mathbf{H}_{\text{final}}$	Final fused node embeddings
$\mathbf{h}_i^t$	Hidden state of node i at time t
Loss functions	
$\mathcal{L}_{\text{total}}$	Total composite loss function
$\mathcal{L}_{\text{pred}}$	Main prediction loss
$\mathcal{L}_{\text{key}}$	Key node accuracy loss
$\mathcal{L}_{\text{smooth}}$	Temporal smoothness loss
λ_1, λ_2	Loss component weights
Functions and operations	
$\mathcal{F}$	Main prediction function
$\phi(\cdot,\cdot)$	Flow correlation measure
$\text{deg}(i,t)$	Degree of node i at time t
$\text{var}(\cdot)$	Variance function
$\text{sigmoid}(\cdot)$	Sigmoid activation function
$\text{ReLU}(\cdot)$	ReLU activation function
$\text{Pool}(\cdot)$	Pooling operation
$\text{ATT}(\cdot,\cdot)$	Attention mechanism
$\text{MLP}(\cdot)$	Multi-layer perceptron
$\text{Proj}(\cdot)$	Projection operation

Data availability

Data will be made available on request.

References

Bai, L., Yao, L., Li, C., Wang, X., Wang, C., 2020. Adaptive graph convolutional recurrent network for traffic forecasting. Adv. Neural Inf. Process. Syst. 33, 17804–17815.

Cao, S., Wu, L., Zhang, R., Wu, D., Cui, J., Chang, Y., 2024. A spatiotemporal multiscale graph convolutional network for traffic flow prediction. IEEE Trans. Intell. Transp. Syst..

Chen, M., Han, L., Xu, Y., Zhu, T., Wang, J., Sun, L., 2024a. Temporal-aware structure-semantic-coupled graph network for traffic forecasting. Inf. Fusion 107, 102339.

Chen, C., Petty, K., Skabardonis, A., Varaiya, P., Jia, Z., 2001. Freeway performance measurement system: Mining loop detector data. Transp. Res. Rec. 1748 (1), 96–102.

Chen, X., Tang, H., Wu, Y., Shen, H., Li, J., et al., 2024b. AdpSTGCN: Adaptive spatial–temporal graph convolutional network for traffic forecasting. Knowl.-Based Syst. 301, 112295.

Cini, A., Marisca, I., Bianchi, F.M., Alippi, C., 2023. Scalable spatiotemporal graph neural networks. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 37, pp. 7218–7226, 6.

Cui, T., Lin, X., Li, S., Chen, M., Yin, Q., Li, Q., Xu, K., 2025. TrafficLLM: Enhancing large language models for network traffic analysis with generic traffic representation. arXiv preprint arXiv:2504.04222.

Dai, B.A., Ye, B.-L., 2024. A novel hybrid time-varying graph neural network for traffic flow forecasting. arXiv preprint arXiv:2401.10155.

Drucker, H., Burges, C.J., Kaufman, L., Smola, A., Vapnik, V., 1996. Support vector regression machines. Adv. Neural Inf. Process. Syst. 9.

Fang, Z., Long, Q., Song, G., Xie, K., 2021. Spatial-temporal graph ode networks for traffic flow forecasting. In: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining. pp. 364–373.

Fey, M., Lenssen, J.E., 2019. Fast graph representation learning with PyTorch Geometric. In: ICLR Workshop on Representation Learning on Graphs and Manifolds.

Gao, Y., Wang, X., 2024. Survey on traffic flow prediction methods based on graph convolution neural network. In: Seventh International Conference on Traffic Engineering and Transportation System (ICTETS 2023). vol. 13064, SPIE, pp. 152–162.

Go, H., Park, S., 2024. A study on deep learning model based on global–local structure for crowd flow prediction. Sci. Rep. 14 (1), 12623.

Gomes, B., Coelho, J., Aidos, H., 2023. A survey on traffic flow prediction and classification. Intell. Syst. Appl. 200268.

Guo, S., Lin, Y., Feng, N., Song, C., Wan, H., 2019. Attention based spatial-temporal graph convolutional networks for traffic flow forecasting. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 33, pp. 922–929, 01.

Guo, K., Tian, D., Hu, Y., Sun, Y., Qian, Z., Zhou, J., Gao, J., Yin, B., 2024. Contrastive learning for traffic flow forecasting based on multi graph convolution network. IET Intell. Transp. Syst. 18 (2), 290–301.

He, S., Luo, Q., Fu, X., Zhao, L., Du, R., Li, H., 2024. Cat: A causal graph attention network for trimming heterophilic graphs. Inform. Sci. 677, 120916.

Hu, N., Zhang, D., Xie, K., Liang, W., Li, K.-C., Zomaya, A.Y., 2024. Dynamic multi-scale spatial–temporal graph convolutional network for traffic flow prediction. Future Gener. Comput. Syst. 158, 323–332.

Jin, G., Li, F., Zhang, J., Wang, M., Huang, J., 2022. Automated dilated spatio-temporal synchronous graph modeling for traffic prediction. IEEE Trans. Intell. Transp. Syst. 24 (8), 8820–8830.

Jin, G., Liang, Y., Fang, Y., Shao, Z., Huang, J., Zhang, J., Zheng, Y., 2023a. Spatio-temporal graph neural networks for predictive learning in urban computing: A survey. IEEE Trans. Knowl. Data Eng. 36 (10), 5388–5408.

Jin, G., Liu, L., Li, F., Huang, J., 2023b. Spatio-temporal graph neural point process for traffic congestion event prediction. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 37, pp. 14268–14276, 12.

Kingma, D.P., Adam, B.J., 2014. A method for stochastic optimization.. p. 10, arXiv preprint arXiv:1412.6980.

Kong, W., Guo, Z., Liu, Y., 2024b. Spatio-temporal pivotal graph neural networks for traffic flow forecasting. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 38, pp. 8627–8635, 8.

Kong, W., Wu, K., Zhang, S., Liu, Y., 2025. GraphSparseNet: A novel method for large scale traffic flow prediction. ArXiv:2502.19823.

Kong, L., Yang, H., Li, W., Guan, J., Zhou, S., 2024a. Traffexplainer: A framework towards GNN-based interpretable traffic prediction. IEEE Trans. Artif. Intell..

Lan, S., Ma, Y., Huang, W., Wang, W., Yang, H., Li, P., 2022. Dstagnn: Dynamic spatial-temporal aware graph neural network for traffic flow forecasting. In: International Conference on Machine Learning. PMLR, pp. 11906–11917.

Lei Ba, J., Kiros, J.R., Hinton, G.E., 2016. Layer normalization. ArXiv E-Prints arXiv–1607.

Li, F., Feng, J., Yan, H., Jin, G., Yang, F., Sun, F., Jin, D., Li, Y., 2023. Dynamic graph convolutional recurrent network for traffic prediction: Benchmark and solution. ACM Trans. Knowl. Discov. from Data 17 (1), 1–21.

Li, G., Wang, W., Wang, L., Liu, Y., Zhang, M., 2024a. Deep spatial-temporal information fusion dynamic graph convolutional network for traffic flow prediction. Meas. Sci. Technol. 36 (1), 015102.

Li, Y., Xu, H., Zhang, T., Li, X., Li, G., Tian, W., 2024b. DDGformer: Direction-and distance-aware graph transformer for traffic flow prediction. Knowl.-Based Syst. 302, 112381.

Li, Y., Yu, R., Shahabi, C., Liu, Y., 2017. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. arXiv preprint arXiv:1707.01926.

Li, Y., Yu, R., Shahabi, C., Liu, Y., 2018. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. In: International Conference on Learning Representations.

Li, M., Zhu, Z., 2021. Spatial-temporal fusion graph neural networks for traffic flow forecasting. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 35, pp. 4189–4196, 5.

Liu, H., Dong, Z., Jiang, R., Deng, J., Chen, Q., Song, X., 2023. STAEformer: Spatio-temporal adaptive embedding makes vanilla transformers SOTA for traffic forecasting. In: Proceedings of the 32nd ACM International Conference on Information and Knowledge Management (CIKM), Birmingham, UK. pp. 21–25.

Liu, C., Hettige, K.H., Xu, Q., Long, C., Xiang, S., Cong, G., Li, Z., Zhao, R., 2025. ST-LLM+: Graph enhanced spatio-temporal large language models for traffic prediction. IEEE Trans. Knowl. Data Eng..

Liu, C., Yang, S., Xu, Q., Li, Z., Long, C., Li, Z., Zhao, R., 2024. Spatial-temporal large language model for traffic prediction. In: 2024 25th IEEE International Conference on Mobile Data Management. MDM, IEEE, pp. 31–40.

Lu, Z., Zhou, C., Wu, J., Jiang, H., Cui, S., 2016. Integrating granger causality and vector auto-regression for traffic prediction of large-scale WLANs. KSII Trans. Internet Inf. Syst. (TIIS) 10 (1), 136–151.

Luo, Y., Zheng, J., Wang, X., Tao, Y., Jiang, X., 2024. GT-LSTM: A spatio-temporal ensemble network for traffic flow prediction. Neural Netw. 171, 251–262.

Luo, X., Zhu, C., Zhang, D., Li, Q., 2023. Dynamic graph convolutional network with attention fusion for traffic flow prediction. In: ECAI 2023. IOS Press, pp. 1633–1640.

Ni, Q., Zhang, M., 2022. STGMN: A gated multi-graph convolutional network framework for traffic flow prediction. Appl. Intell. 52 (13), 15026–15039.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al., 2019. Pytorch: An imperative style, high-performance deep learning library. Adv. Neural Inf. Process. Syst. 32.

Sayed, S.A., Abdel-Hamid, Y., Hefny, H.A., 2023. Artificial intelligence-based traffic flow prediction: A comprehensive review. J. Electr. Syst. Inf. Technol. 10 (1), 13.

Shao, Z., Wang, F., Xu, Y., Wei, W., Yu, C., Zhang, Z., Yao, D., Sun, T., Jin, G., Cao, X., et al., 2024. Exploring progress in multivariate time series forecasting: Comprehensive benchmarking and heterogeneity analysis. IEEE Trans. Knowl. Data Eng..

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R., 2014. Dropout: A simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. 15 (1), 1929–1958.

Tang, J., Chen, S., Gong, C., Zhang, J., Tao, D., 2025a. LLM-PS: Empowering large language models for time series forecasting with temporal patterns and semantics. arXiv preprint arXiv:2503.09656.

Tang, L., Luo, R., Zhou, Z., Colombo, N., 2025b. Enhanced route planning with calibrated uncertainty set. Mach. Learn. 114 (5), 1–16.

Wang, B., Gao, F., Tong, L., Zhang, Q., Zhu, S., 2024a. Channel attention-based spatial-temporal graph neural networks for traffic prediction. Data Technol. Appl. 58 (1), 81–94.

Wang, L., Guo, D., Wu, H., Li, K., Yu, W., 2024b. TC-GCN: Triple cross-attention and graph convolutional network for traffic forecasting. Inf. Fusion 105, 102229.

Wang, T., Ngoduy, D., Zou, G., Dantsuji, T., Liu, Z., Li, Y., 2024c. PI-STGnet: Physics-integrated spatiotemporal graph neural network with fundamental diagram learner for highway traffic flow prediction. Expert Syst. Appl. 258, 125144.

Weng, W., Fan, J., Wu, H., Hu, Y., Tian, H., Zhu, F., Wu, J., 2023. A decomposition dynamic graph convolutional recurrent network for traffic forecasting. Pattern Recognit. 142, 109670.

Wu, J., Chen, L., 2024. Continuously evolving graph neural controlled differential equations for traffic forecasting. ArXiv:2401.14695.

Wu, M., Lin, Y., Jiang, T., Weng, W., 2025. Mhgnet: Multi-heterogeneous graph neural network for traffic prediction. In: ICASSP 2025 - 2025 IEEE International Conference on Acoustics, Speech and Signal Processing. ICASSP, pp. 1–5. http://dx.doi.org/10.1109/ICASSP49660.2025.10890207.

Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., Philip, S.Y., 2020a. A comprehensive survey on graph neural networks. IEEE Trans. Neural Networks Learn. Syst. 32 (1), 4–24.

Wu, Z., Pan, S., Long, G., Jiang, J., Chang, X., Zhang, C., 2020b. Connecting the dots: Multivariate time series forecasting with graph neural networks. In: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. pp. 753–763.

Wu, Z., Pan, S., Long, G., Jiang, J., Zhang, C., 2019. Graph wavenet for deep spatial-temporal graph modeling. arXiv preprint arXiv:1906.00121.

Xia, J., Wang, S., Wang, X., Xia, M., Xie, K., Cao, J., 2024. Multi-view Bayesian spatio-temporal graph neural networks for reliable traffic flow prediction. Int. J. Mach. Learn. Cybern. 15 (1), 65–78.

Xu, Z., Yuan, J., Yu, L., Wang, G., Zhu, M., 2024. Machine learning-based traffic flow prediction and intelligent traffic management. Int. J. Comput. Sci. Inf. Technol. 2 (1), 18–27.

Yang, H., Li, Z., Qi, Y., 2024. Predicting traffic propagation flow in urban road network with multi-graph convolutional network. Complex & Intell. Syst. 10 (1), 23–35.

Yu, B., Yin, H., Zhu, Z., 2017. Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. arXiv preprint arXiv:1709.04875.

Yuan, Y., Ding, J., Feng, J., Jin, D., Li, Y., 2025. A universal pre-training and prompting framework for general urban spatio-temporal prediction. IEEE Trans. Knowl. Data Eng..

Zhang, H., Chen, L., Zhang, X., Cao, J., 2024b. STC-PSSA: A new model of traffic flow forecasting based on spatiotemporal convolution and probabilistic sparse self-attention. Transp. Res. Rec. 03611981241252146.

Zhang, Q., Gao, X., Wang, H., Yiu, S.M., Yin, H., 2025. Efficient traffic prediction through spatio-temporal distillation. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 39, pp. 1093–1101, 1.

Zhang, J., Mao, S., Yang, L., Ma, W., Li, S., Gao, Z., 2024d. Physics-informed deep learning for traffic state estimation based on the traffic flow model and computational graph method. Inf. Fusion 101, 101971.

Zhang, D., Wang, P., Ding, L., Wang, X., He, J., 2024a. Spatio-temporal contrastive learning-based adaptive graph augmentation for traffic flow prediction. IEEE Trans. Intell. Transp. Syst..

Zhang, H., Wang, H., Zhang, X., Gong, L., 2024c. Research on traffic flow forecasting based on dynamic spatial-temporal transformer. Transp. Res. Rec. 2678 (7), 301–313.

Zhao, Z., Wang, P., Wen, H., Wang, S., Yu, L., Wang, Y., 2025b. STEM-LTS: Integrating semantic-temporal dynamics in LLM-driven time series analysis. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 39, pp. 22858–22866, 21.

Zhao, W., Yuan, G., Bing, R., Lu, R., Shen, Y., 2025a. Periodicity aware spatial-temporal adaptive hypergraph neural network for traffic forecasting. GeoInformatica 29 (2), 201–232, URL https://doi.org/10.1007/s10707-024-00527-7.

Zheng, C., Fan, X., Wang, C., Qi, J., 2020. Gman: A graph multi-attention network for traffic prediction. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 34, pp. 1234–1241, 01.

Zong, X., Qi, Y., Yan, H., Ye, Q., 2024. An intelligent deep learning framework for traffic flow imputation and short-term prediction based on dynamic features. Knowl.-Based Syst. 300, 112178.