Expert Systems With Applications 327 (2026) 132713

Contents lists available at ScienceDirect

Expert Systems With Applications

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

Graph neural network for continuous traffic density estimation on unmonitored roads from very few scattered measurements

Nathan Acciai [a], Stefano Bilotta [b], Marco Fanfani [a], Paolo Nesi [a,*]

[a] Department of Information Engineering, University of Florence, Via S. Marta, 3, Firenze, FI, 50139, Italy
[b] Department of Mathematics and Computer Science, University of Florence, Viale Morgagni, 67/a, Firenze, FI, 50134, Italy

ARTICLE INFO

Keywords:
Intelligent transport system
Traffic flow reconstruction
Graph neural network
Feature encoding
Temporal gating
Real-time continuous estimation
Road networks
Urban mobility

ABSTRACT

Traffic control and management play a crucial role within intelligent transportation systems. However, comprehensive traffic monitoring is often constrained by the limited availability of traffic sensors across urban road networks. This paper addresses the problem of Traffic Flow Reconstruction (TFR), which aims to estimate traffic conditions over an entire road network starting from a few sparse traffic sensor measurements. This approach does not require full road coverage by sensors, thereby reducing deployment costs and supporting sustainable urban mobility. In the present research, we propose a lightweight Graph Neural Network (GNN) architecture for real-time TFR using data from a few number of fixed sensors. The model exploits a carefully designed road graph, explicitly augmented to capture traffic propagation and backpropagation effects at junctions, and well-studied node features augmented though a learned encoder. Compared to classical Partial Differential Equations-based TFR solutions, the proposed method achieves satisfactory estimation accuracy while significantly reducing setup times by avoiding scenario-dependent optimizations and precomputation of traffic flow distributions at junctions. Moreover, it overcomes the limitations of existing GNN-based approaches, which often rely on complex architectures that are difficult to adapt to different urban contexts without reprocessing them. Extensive experiments have been conducted and presented in this paper taking into account different moments of the year, various road graph representations and sensors' positioning, and multiple city environments. Results demonstrate the robustness, generalization capability, and computational efficiency of the proposed approach, making it suitable for large-scale and real-world mobility and transport applications.

1. Introduction

Urban mobility is one of the main criticism cities have to face nowadays. Indeed, achieving sustainable mobility solutions is one of the UN Sustainable Development Goals (United Nations, 2026). It is required to provide municipalities and decision makers with tools and solutions to address such problems. Intelligent Transportation Systems (Elassy et al., 2024) and, more recently, Smart City Digital Twins (Adreani et al., 2024b) can exploit the ever-increasing amount of data to implement decision support systems helping to design, test, and deploy improved mobility solutions (Bellini et al., 2023). Among the different mobility domains - e.g., public transport planning, security and safety, freight logistics, shared mobility - traffic control and management are probably the most relevant since they can impact on all the other domains.

Traffic flow estimation is related to compute the current traffic conditions across all segments of a road network. This real-time information is essential for a wide range of intelligent transportation services, including adaptive route planning (Liu et al., 2022; Wang et al., 2024), traffic light optimization (Bilotta et al., 2025; Bouktif et al., 2023; Kumar et al., 2024), congestion management, traffic reduction strategies (Adreani et al., 2024a; Anciaes et al., 2025; Cheng et al., 2024), and monitoring of fuel usage and emission levels (Alam et al., 2025; Bilotta & Nesi, 2022; Caraveo Mena et al., 2025; Linardatos et al., 2023).

Within the scope of traffic flow theory, it is important to distinguish between Traffic Flow Reconstruction (TFR), Traffic Flow Prediction (TFP), Missing Data Imputation (MDI), and Traffic State Estimation (TSE).

TFR has the primary objective of estimating the complete traffic state across an entire road network, starting from a sparse set of measurements. In particular, this paper focuses on urban traffic, a more complex scenario with respect to extra-urban or highway cases. Traffic measurements are typically collected from a very limited number of sensors placed at fixed locations, such as video cameras or inductive loop detectors (Gheorghiu et al., 2021; Lee et al., 2023; Thabit et al., 2024). These tools can provide data on vehicle count, traffic density, and speed. However, due to environmental and budget constraints, no city is able to de-

* Corresponding author.
E-mail addresses: nathan.acciai@edu.unifi.it (N. Acciai), stefano.bilotta@unifi.it (S. Bilotta), marco.fanfani@unifi.it (M. Fanfani), paolo.nesi@unifi.it (P. Nesi).

https://doi.org/10.1016/j.eswa.2026.132713
Received 10 February 2026; Received in revised form 13 April 2026; Accepted 1 May 2026
Available online 11 May 2026

ploy sensors on every road segment. To overcome this limitation, TFR algorithms are employed to estimate conditions in unmonitored areas based on available data. TFR is formulated over a road graph, where sensors provide data only at certain nodes, i.e., roads, and the goal is to infer the traffic flow across the remaining unmeasured nodes. In this context, the estimation of the traffic distribution at junctions plays a crucial role in the effectiveness of the solution in real context for large networks (Bilotta et al., 2024; Bilotta & Nesi, 2021; Cardaliaguet & Souganidis, 2024; Khan et al., 2022; Saliev & Damyanov, 2024).

TFP focuses on forecasting future traffic conditions (over short- or long-term horizons) at specific locations. Traffic prediction has been extensively studied using Machine Learning and Deep Learning models (Bilotta et al., 2022; Khan et al., 2023; Li & Wang, 2025; Zheng et al., 2024). These methods aim to forecast traffic volumes or congestion levels at specific points where sensors are available.

MDI problem is a general issue that focuses on recovering incomplete information in datasets (Adhikari et al., 2023; Sun et al., 2022), and it is sometimes confused with TFR. However, there are important differences. Missing data imputation methods typically assume that most of the data is available, and only a small percentage of values are missing. Then, these methods are designed for scenarios where the underlying data matrix or graph is mostly complete. This does not appear in TFR approaches, since an extrapolation approach passing from few sensors' data to thousands traffic flow segments is generally assumed.

TSE is another related concept, which shares some goals with TFR but differs in methodology and assumptions. The aim of TSE is to estimate traffic conditions using partial observations combined with data from multiple heterogeneous sources. These sources can include fixed sensors, GPS data from probe vehicles, mobile apps, crowdsourced data, etc. (Xing et al., 2022). Note that, except for sensor data, other considered sources are usually difficult to acquire due to high costs or privacy concerns. TSE typically relies on the integration of physical models with observed data to produce accurate traffic estimates (Raissi et al., 2019; Syum Gebre et al., 2025). However, this approach assumes the availability of large quantities of data from various sources, which may not always be feasible in practice. Moreover, TSE often involves very complex and heavy model architectures, such as black (GNNs), Transformed-Based models, Physics-Informed Neural Networks (PINNs), and Large Language Models (LLMs) for capturing spatial and temporal dependencies. In particular, most existing approaches are evaluated using standard datasets for motorway scenarios, such as the Caltrans Performance Measurement System (PeMS). However, these datasets are not directly comparable to urban traffic scenario involving complex interactions at junctions, traffic congestion, different types of roads and the associated variability in traffic conditions. As noted in Xing et al. (2022), traffic estimation in the complex environment of urban road network is more difficult than in highways.

TFR methods can be generally classified as model-driven and data-driven. Model-driven methods rely on physical traffic models, which treat traffic flow as fluid dynamics and require complex handling of traffic distribution at junctions. Model-driven also includes agent-based simulations, such as SUMO (Lopez et al., 2018; Santana et al., 2018), where individual vehicles are modelled as agents. On the other hand, data-driven approaches either simply use statistical and machine learning techniques to infer traffic states directly from observed data or integrated physical models with advanced machine learning methods. In general, model-driven approaches require significant efforts to determine traffic distribution at junctions, often by using nonlinear partial differential equations (PDEs). On the other hand, data-driven methods are rarely used since, usually, exploits complex and heavy architectures and requires high data volumes (not easy to acquire), thus limiting their applicability on unseen scenarios without continuous calibration or training.

For these reasons, this paper proposes an novel learning-based approach to solve the TFR problem in the challenging urban context. The proposed method implements a simple, lightweight, and robust architecture, based on GNN, able to provide accurate reconstructions using only very limited data in few fixed locations provided by municipal sources. Such a result has been achieved by introducing carefully designed feature encoding layers that enable the network to learn the importance of road characteristics avoiding the need for convoluted architectures and models, like recurrent network or transformers, to explicitly model temporal aspects, thus allowing easier training and deployment. In addition, a specific graph structure design is proposed to better capture traffic propagation. In this way, the presented method can learn how traffic conditions evolve over time and propagate through space, accurately reflecting the real-world scenarios. Indeed, traffic condition evolution is typically modeled through fluid dynamics approaches, and those methods require more computationally intensive training and simulation processes in model-driven fields. In contrast, the proposed GNN is optimized to minimize computational time, offering a general and efficient solution for real-time TFR.

This paper provides the following main contributions:

- A lightweight neural network to solve the TFR problem in urban contexts. The network is composed by GNN and fully connected network (FCN) and exploits a carefully designed feature definition and learned encoding together with a specific road graph representation.
- A specific training strategy exploiting a *leave-one-out* schema in order to train the network using only data coming from very few traffic sensors, thus not requiring a ground truth density for all the road segments.
- A continuous traffic density estimation strategy to apply the proposed solution on real scenarios, made robust to missing sensor observations with the use of typical time trends computed by averaging historical data.
- A detailed ablation study showing the relevance of various components of the architecture, the impact of the road graph modelling, and that the continuous estimation strategy can be reliably initialized with zero values, without relying on costly or complex initialization schemes.
- An extensive experimental analysis demonstrating the capability of the proposed solution to work on different time periods, different sensors' positions, and diverse urban contexts, highlighting its accuracy and robustness on unseen scenario without retraining or fine-tuning.

The paper is organized as follows. In the next section (Section 2) related works are presented and discussed, highlighting current limits. In Section 3, the proposed method is described in detail, including input data modelling, training algorithm and loss function, and neural architecture. In Section 4, the test setup is illustrated, presenting the different datasets, the metric used, and the recursive estimation strategy. Experiments are reported in Section 5, including ablation studies and results on unseen scenarios. Finally, in Section 6, conclusions are drawn.

This work has been carried out in the context of the Italian National Center for Sustainable Mobility (CN MOST) Spoke 9 (Alberti et al., 2023) and Spoke 8 flagship project OPTIFaaS and scalability project SASUAM.

2. Related works

In recent years, GNNs have been widely applied across various domains due to their ability to analyze graph-structured data, and their use in transportation has grown significantly. Generally, in the context of GNNs, traffic flow issues are investigated by means of studies aimed at improving the efficiency of existing Graph Convolutional Network (GCN) models. A survey can be found in Rahmani et al. (2023).

Most current approaches primarily address challenges related to MDI, which differ from a typical TFR approach. Unlike these tasks, which operate in dense datasets and primarly aim to fill gaps in observations, TFR provides the reconstruction of the entire traffic state across a network from extremely low number of measurements. In this

way, TFR is a highly underdetermined problem, requiring both information about the propagation across the graph and structural and physical constraints, particularly at the network junctions where the flow distribution is unknown. In the following, we discuss some related works involving different approaches, such as classical GNNs, LLM-based methods and phisics-informed models, highlighting their limitations in the context of TFR.

In GNN framework, much research is related to the task of reconstructing missing data at the nodes of a graph. For example, the work in S and Zacharias (2024) focuses on the reconstruction of missing data using a GCN architecture. The graph dataset is first partitioned into clusters in an unsupervised manner using the K-means clustering algorithm. Then, a Gaussian Mixture Model (GMM) is employed to initialize the missing values within each cluster. For each cluster, a GCN combined with Matrix Factorization (GCN-MF) is applied. Finally, the reconstructed matrices from all clusters are merged and passed through a standard GCN model for further processing. In Xia et al. (2024) a MDI approach using autoencoders has been developed, in particular a dual GCN encoder to encode structural information and attribute information and a multiscale decoder with masked nodes to extract node reconstructions. The work in Chen and Chen (2022) deals with missing traffic data: a reinforced dynamic GCN is proposed to simultaneously conduct both missing data imputation and traffic prediction. In Liu et al. (2019), a graph based-theory neural network is applied to characterize traffic correlations from taxi probes, where, on average, 17,716 road segments (30.4% of all) are traveled by taxis in a timeslot.

The above-mentioned approaches address a general topic which is related to the reconstruction of missing data within graphs and the strategies for recovering of such data. However, when such approaches are applied to the specific problem of TFR from a limited number of control points, then they present several limitations. More precisely, the architectural complexity appears not appropriate to the objective of the present work. A key limitation of such approaches is that, as the amount of missing data increases, the reconstruction error rises and the accuracy drops significantly. Moreover, even in the most challenging setting, for example those considered in Chen and Chen (2022), Liu et al. (2019), where up to 70% of the data is missing, at least 30% of the observations still remains available. This is completely different from TFR scenarios, where only 1–3% of the data is accessible, making the reconstruction problem heavily undermined.

Given the rapid development of LLMs, several recent techniques have incorporated the LLM to enhance performance in the context of GNN and a possible application in traffic flow problems. Some examples can be found in Chen et al. (2024) and Ren et al. (2024), where an in-depth analysis of the role of LLMs in graph learning is provided. Nevertheless, LLMs are not directly applicable to the problem of TFR, since such approaches are primarily designed for scenarios where graphs incorporate textual or semantic information. In the context of TFR, the problem addressed relies only on spatio-temporal numerical data collected from a limited number of control sensors. In general, the application of direct LLMs lacks sufficient information to operate effectively in this setting.

In the context of physics-informed models, a work in Syum Gebre et al. (2025) presented a novel approach for real-time TSE on sparse sensor data and inflexible physical models. The proposed framework, PINN-TSE (Physics-Informed Neural Network-Based Traffic State Estimator), integrates the classical (Aw-Rascle) traffic model with machine learning techniques and LLM. In this case, LLMs are utilized to generate contextualized and user-friendly insights through a chat-based web application and not directly used for traffic computation. The effectiveness of the approach is validated using real-world traffic data measured on 640 m of the US-101 highway. Consequently, the optimized parameterization is scenario-specific and the related architecture cannot be directly generalized to other contexts. For instance, the application to different traffic scenarios, such as urban networks, is not suitable. In contrast, the proposed approach is designed to enable immediate adaptability to a wider range of scenarios, encompassing different datasets and network configurations, thereby enhancing its potential for general real-world applicability. Often, in PIDL (physical-informed deep learning), no road crossing modeling has been considered to address the indeterminacy at the junctions, and the possible application in real-case city-road network is then limited (Huang & Agarwal, 2022; Rempe et al., 2021; Shi et al., 2021).

GNNs for traffic related problems have been exploited in Kaiser et al. (2026) with the aim of interpolating traffic flow in road segments. Evaluation is carried out using two datasets. A first dataset considers cycling paths in Berlin, which is of limited interest for analyzing urban vehicle traffic. The second dataset focuses on road network in New York and has been developed starting from taxi pick-up and drop-off positions. Then using a routing algorithm, trajectories has been reconstructed and used to compute the number of taxis passing per street segment, hour by hour, obtaining a sort of traffic flow. However, such dataset has some relevant drawbacks. First, limiting to taxis, only a small fraction of the total vehicular traffic is taken into account. Moreover, computing traffic flows exploiting trajectories obtained with routing algorithm without using more accurate simulations considering traffic, can lead to erroneous routes and travel times, thus affecting the flow values. Nonetheless, even under the assumption of dataset validity, the method still has limitations: when reducing the number of roads with known traffic flow from 90% to 1%, error on flow estimates on unmonitored roads increases of 76% on the New York taxi dataset. This hampers the applicability of the method in real scenarios in which sensor coverage can be inferior to 1% of the roads. Moreover, no tests have been carried out on scenarios not seen during training. Finally, graph node features includes social, economic, and environmental data: even if including such data is reasonable, they requires substantial work to be acquired, particularly for small municipalities with limited access to weather sensors and well structured social and economic data.

The main limitations of the above cited approaches, in the context of TFR issue, are summarized in Table 1.

Among the studies of the literature attempting to address and partially overcome the above limitations, a notable example is (Bilotta & Nesi, 2021). That study is directly situated within the same application domain considered here, focusing on the TFR problem in urban contexts. Nevertheless, the methodology proposed in Bilotta and Nesi (2021) remains constrained by a computationally demanding framework, both in terms of processing time and resource usage. The accuracy of that solution primarily depends on a stochastic relaxation approach for estimating the Traffic Distribution Matrices (TDMs), which describe the percentage of vehicles exiting each outgoing link relative to those entering from each incoming link at a junction. TDMs are computed for each junction in the network and at each time step. During each iteration, if the local error decreases compared to the previous step, the corresponding weights are retained. The process continues by testing new TDM configurations until the overall system error is minimized, following a mechanism similar to Simulated Annealing. Although this approach can yield accurate results, it is computationally intensive and therefore typically applied only sporadically to update the TDMs at each hour of the day. In contrast, the present work introduces a more efficient approach in terms of computational speed and cost, enabling more frequent and scalable model updates across different traffic scenarios. To achieve accurate TFR in large-scale road networks, the following key challenges have to be considered: (i) generate dense traffic flow estimations across graph-based city-road networks via general approach (while, PDE-based model-driven approaches can provide such reconstructions, they are computationally expensive); (ii) solve the indeterminacy of traffic distribution at junctions, including backpropagation effects, which remains a complex problem for most methods (e.g., the stochastic relaxation approach proposed in Bilotta and Nesi (2021)); (iii) address high computational complexity and execution time, which can be better handled through machine learning or deep learning tech-

Table 1
Comparison of related works and their limitations in solving TFR.

Reference	Main Focus	Limitations in TFR
S & Zacharias (S & Zacharias, 2024)	Missing data reconstruction in graphs	• High architectural complexity unsuitable for real-time TFR • Poor scalability when only few sensors (1–3%) available • Accuracy drops significantly as missing data increases
Xia et al. (2024)	Missing data imputation using GCN autoencoders	• Overly complex for efficient real-time estimation • Limited robustness with highly sparse input data
Chen & Chen (Chen & Chen, 2022)	Joint missing data imputation and traffic prediction	• High computational cost • Limited to dense datasets • Accuracy declines under sparse sensor availability
Chen et al. (2024), Ren et al. (2024)	Integration of LLMs with GNNs for graph learning	• High architectural complexity • Require textual/semantic context not available in TFR • Ineffective for direct traffic computation
Syum Gebre et al. (2025)	Real-time traffic estimation with physics-informed ML and LLMs	• Scenario-specific tuning (US-101 only) • Poor generalization to other networks • LLMs not directly used for traffic computation
Huang and Agarwal (2022), Shi et al. (2021), Rempe et al. (2021)	Physics-Informed Deep Learning (PIDL) models	• Lack of road junction modeling • Limited applicability to complex city networks • Indeterminacy issues at intersections
Kaiser et al. (2026)	GNN for traffic interpolation in urban scenarios	• Evaluated on derived datasets with partial vehicle coverage • Decreased performance when limiting monitored roads to 1% • No evaluation on scenarios not seen in training • Requires a large number of node features not easily accessible

niques. The approach presented in this paper leverages a GNN architecture specifically designed to tackle these challenges.

3. Method description

The TFR problem consists of reconstructing the traffic flow or densities for the whole considered road graph using few measurements taken from a very limited set of control points (i.e., traffic sensors) arranged over the road network. More formally, the TFR problem can be defined as follows: let $G = \{N, E\}$ be a graph modelling the road network, where N is the set of nodes representing the roads and E the set of edges representing the road connections. Nodes are bipartite in two disjoint sub-sets S and $\bar{S}$, with $N = \{S \cup \bar{S}\}$ and $|S| \ll |\bar{S}|$, where S is the set of roads monitored by a traffic sensor and $\bar{S}$ the set of unmonitored roads. Nodes in S are labeled with traffic density measures, and the objective of the TFR problem is to infer traffic densities for all the nodes in $\bar{S}$.

In this section, the proposed solution is described. The proposed method exploits GNN, careful modelling of the road graph, and accurate feature engineering to model spatio-temporal interaction among the roads and learn how traffic flow spreads out in the road network. In the next section, the road graph modelling and the nodes' features are presented. Then, the training strategy and the implemented loss function are discussed. Finally, the adopted artificial intelligence architecture composed of graph convolutional layers and dense feed-forward layers is outlined, illustrating also the hyperparameters found though optimization.

3.1. Data modelling

In this section, data preprocessing is described. The road graph is obtained from Snap4City digital twin platform (https://www.snap4city.org) (Adreani et al., 2024c) in which corrected and updated OpenStreetMap (OpenStreetMap, 2026) data have been ingested, including road attributes as number of lanes, travel directions, road type, etc. First, the road graph modelling is presented, then features used to describe each graph nodes are introduced. Finally, two different road graph representations, used in the following section to assess the impact of the number of nodes in the TFR resolution, are described.

3.1.1. Road graph modelling

A complex road network structure is necessary to ensure applicability to real-world urban traffic scenarios and to address the limitations of single-road models as in Huang and Agarwal (2022), Rempe et al. (2021), Shi et al. (2021). Starting from Snap4City data, the road network has been defined as a directed graph $G = \{N, E\}$ where N is the set of nodes representing the roads and E the set of edges representing the road connections (i.e., the road junctions). Such a definition has been taken since GNNs are more effective in solving node-level tasks instead of edge-level tasks. Modelling roads as nodes, the TFR problem is framed as a node labelling task in which a few nodes (roads with sensors) have a label (the traffic flow measure) and the neural network must infer the labels on all the other nodes.

The direction of the edges in E follows the road travel direction, i.e., an edge $e_{AB} \in E$ connects nodes $n_A, n_B \in N$ if and only if vehicles move from road A to road B. In the case of bidirectional roads, each road direction is instantiated as a separate node with new connecting edges representing different travel directions.

When vehicles slow down due to congestion, a wave propagating backward through the stream of vehicles is created, even though vehicles are moving forward. To consider such a backward propagation, dummy edges $\hat{e}$ are introduced with an inverse orientation with respect to the travel direction. In this way, we model the fact that vehicle queues develop downstream when they form and then dissipate upstream. The bidirectionality of the edges implements the concept of information feedback, providing the network with greater awareness of how traffic moves within the graph. In Fig. 1, an example is reported, showing how the road network is modelled in graph G.

3.1.2. Node features

Each node of G is associated with a set of static and dynamic features. Static features include the number of lanes, whether there is a traffic sensor on the road, and the road type (e.g., primary, secondary, residential, etc.). Dynamic features encode time information and traffic

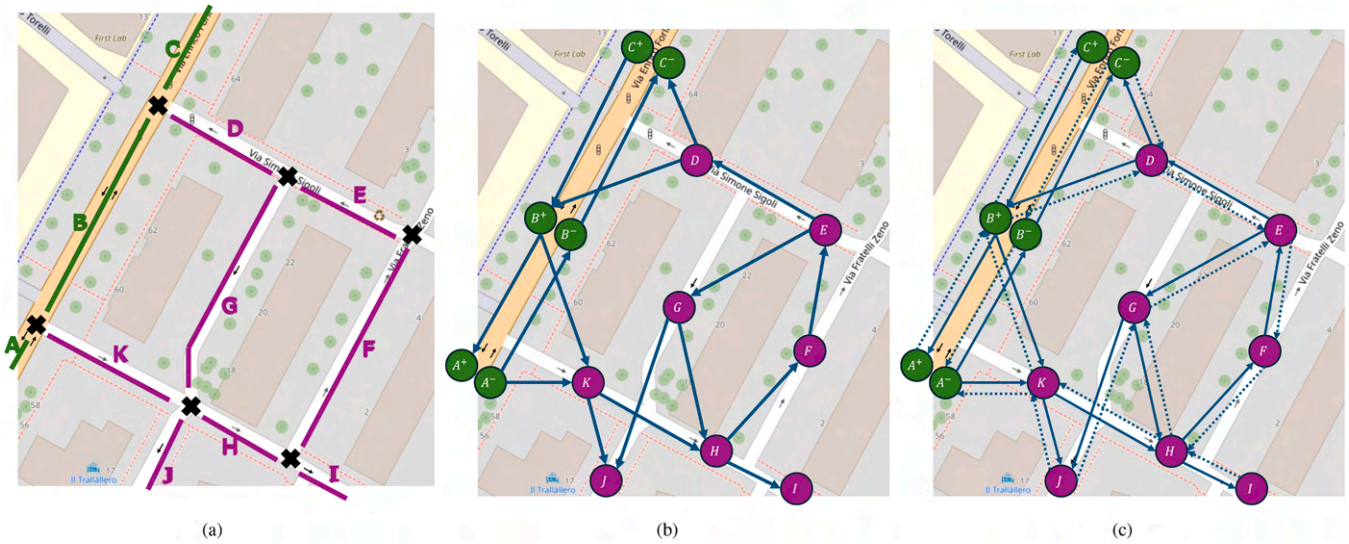

Fig. 1. Example of road network modelling. In (a) the road network is shown: junctions are highlighted as black crosses, one-way roads as purple segments, and two-way roads as green segments (travel direction is indicated with small arrows in the underling map). In (b) the graph is defined by associating nodes to segments and oriented edges to junctions. In case of two-way roads, i.e., A, B, and C green segments in (a), different nodes are instantiated for each travel direction (A^+, A^-, B^+, B^-, C^+, C^-). Edges follow the allowed travel direction connecting the roads: for example node E has two exiting edges toward D and G, and one entering edge from F. In (c) the graph is updated by introducing dummy edges (shown as dashed arrows) used to consider backward traffic propagation.

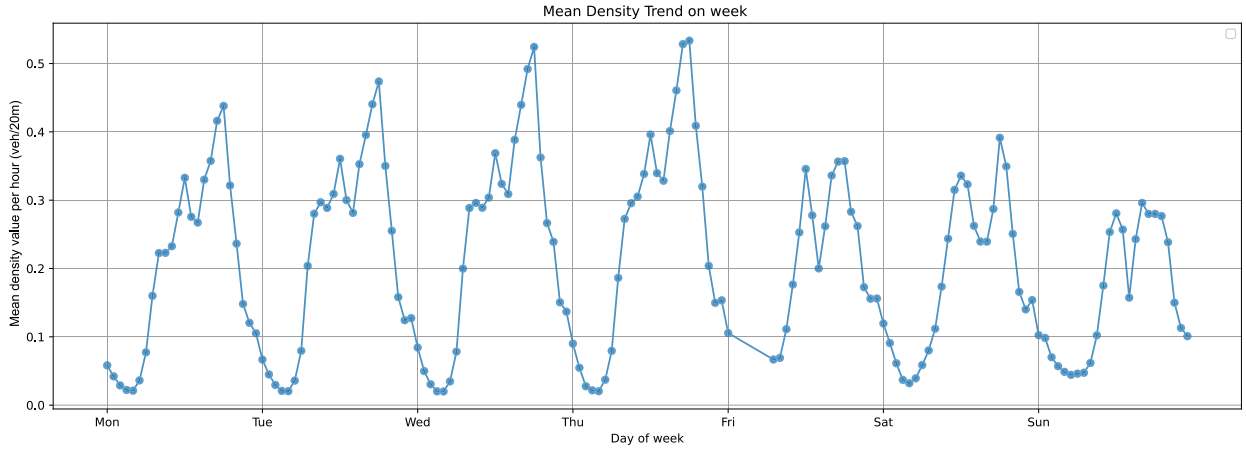

Fig. 2. Plot of the weekly variation of mean traffic density normalized with respect to a single lane (expressed as vehicles per 20m) considering all sensors on the road network by averaging hourly measurements over a month. The plot shows evident periodicity. Note that, being an average, this plot does not show strong traffic density peaks that, instead, in some positions, can reach values greater than 3 veh/20m.

density as described in the following. Urban traffic exhibits a periodicity that must be considered when modelling dynamic features of a node. Fig. 2 shows an example of the weekly variation of mean traffic density over the whole road network averaging hourly measurements over a month. It can be observed that, the level of densities repeats quite regularly from Monday to Thursday with a lower peak in the morning and a higher peak in the evening. On Friday and Saturday peaks are less pronounced, and on Sunday inferior traffic densities are measured. Since time is highly correlated with the traffic density it has been modelled into the node features, and to consider its periodicity, sine and cosine functions have been exploited. A naïve encoding by simply using the hours would have led to discontinuities when passing from one day to the next. Sinusoidal transformations allow to preserve cyclicity and provides smooth continuities.

A series of conversions are required to encode the times. Since in our experiments we consider time slots of 15 min, time is converted in quarters q according to equation

$$q = 4h + \frac{m}{15} \tag{1}$$

where $h \in [0, 23]$ indicates hours, and $m \in \{0, 15, 30, 45\}$ min. Since in a day there are 96 slots of 15 min, then we normalize the q values with

$$\theta_q = \frac{2\pi q}{96} \tag{2}$$

obtaining angle θ_q. Finally, sine and cosine of θ_q are computed. A similar approach has been used to encode the day of the week. Let $d = [0, 6]$ be the representation of the day of the week, with 0 for Monday and 6 for Sunday, the angle θ_d is computed as follows:

$$\theta_d = \frac{2\pi d}{7} \tag{3}$$

Then, sine and cosine of θ_d are computed. Dynamic temporal features also include an encoding of the parts of the day, i.e., morning, afternoon, evening, night, and a binary attribute indicating whether the day is a workday or a holiday.

Dynamic traffic features have been used instead to label each node with traffic densities. Density at time t, i.e., δ_t, and four previous density measurements, δ_{t-15}, δ_{t-30}, δ_{t-45}, and δ_{t-60}, have been used to model the density evolution in steps of 15 min. Note that traffic densities have

Table 2
Static and dynamic features used to describe each road segment.

Kind	Feature	Description	Type
Static	lanes	Number of the lanes in the road segment	Integer
	sensor	Indicates whether the segment has a traffic sensor	Binary
	road-type	Encodes the type of road (e.g., primary, secondary)	Integer
Dynamic (time)	$\cos(\theta_q)$	Cosine of the time, see (1) and (2)	Float
	$\sin(\theta_q)$	Sine of the time, see (1) and (2)	Float
	$\cos(\theta_d)$	Cosine of the weekday, see (3)	Float
	$\sin(\theta_d)$	Sine of the weekday, see (3)	Float
	day-part	Encodes the part of the day (morning, afternoon, evening, night)	Float
	holiday	Indicates whether the day is a workday or holiday	Binary
Dynamic (density)	δ_t	Traffic density in the segment (node) at time t	Float
	δ_{t-15}	Traffic density in the segment (node) 15 min before t	Float
	δ_{t-30}	Traffic density in the segment (node) 30 min before t	Float
	δ_{t-45}	Traffic density in the segment (node) 45 min before t	Float
	δ_{t-60}	Traffic density in the segment (node) 60 min before t	Float

been set in input only for the roads monitored by a traffic sensor and are inferred by the neural network for all others unmonitored road segments. During training, all the densities, usually provided as number of vehicles per meter, have been normalized to $[0, 1]$ using a min-max normalization. Table 2 summarizes the node features, showing the division between static and dynamic attributes.

3.1.3. Graph representations

Considering a fixed number of nodes labeled with traffic densities, corresponding to the road monitored with a traffic sensor, the finer the road representation is, the higher the number of graph nodes is, and, as consequence, more difficult is the solution of the TFR. To assess the impact of the number of nodes on the prediction accuracy, two different graph models have been defined starting from G, Granular graph (G_G) and Backbone graph (G_B). Road segments of G_G are split into sub-segments of 20m, following the representation used in Bilotta and Nesi (2021). Such modelling allowed a detailed view of the traffic flow and a finer representation of the traffic evolution over time, at the cost of a higher number of nodes. Differently, G_B has been obtained starting from G by merging consecutive road segments with identical static features along the travel direction (i.e., the e edges) until is reached a node with different features or with more than one incoming or outgoing edge. This representation requires less nodes to be instantiated, thus an easier prediction task is addressed, yet it offers a high-level view of the road network in which long segments are labeled with a single traffic estimate.

3.2. Training algorithm and loss function

Care has been devoted to designing a training algorithm that enables the GNN to learn how to propagate the traffic densities measured from the road on which a sensor is present.

Let $S \subset N$ be a subset of the graph nodes including all the road segments having a traffic sensor. Note that, the number of roads with sensors is extremely inferior to the total number of roads, i.e., $|S| \ll |N|$, which is consistent with reconstruction problems characterized by severe data sparsity where only 1-3% of the data is available. Roads with sensors are divided into two disjoint sets, named S_R and S_C, where S_R is the reference set including the sensors located in the most strategic positions, while S_C is the control set in which are put all the other sensors scattered over the road graph. Moreover, $|S_R| \ll |S_C|$ and $S = S_R \cup S_C$. In this case, missing-data imputation methods as in Chen and Chen (2022), S and Zacharias (2024), Xia et al. (2024) suffer from several limitations when they are applied to solve TFR from a very small number of control points. In this paper, we focus on the case $|S| \ll |N|$, as it corresponds to scenarios with reduced infrastructure costs enabled by a limited number of sensors. Taking into account few scattered traffic measurements, a *leave-one-out* approach has been applied to train the network: for each epoch, a sensor $s_O \in S_R$ has been randomly selected and its δ_t and previous densities values δ_{t-15}, δ_{t-30}, δ_{t-45}, and δ_{t-60} have been blanked out (transforming it in an unmonitored road). For all the other sensors, the normalized densities are kept and used in input.

Once the predicted densities $\hat{\delta}_t$ have been obtained, they are exploited in the loss function to train the network. The training loss function is composed of two parts: the first evaluates the errors committed on the sensors for which δ_t is provided in input, while the second focuses on the left-out sensor s_0. This formulation assigns more relevance to the error on s_0 (which requires an actual prediction), while preventing the network from diverging from the provided inputs of the other sensors. The implemented loss is defined as follows:

$$\mathcal{L}_{total} = \mathcal{L}_{S\setminus\{s_O\}} + \mathcal{L}_{\{s_O\}} \tag{4}$$

where $\mathcal{L}_\Omega$ is defined as

$$\mathcal{L}_\Omega = \frac{1}{\mathfrak{N}_\Omega} \sum_{i\in\Omega} w_t^i \ell(\hat{\delta}_t^i - \delta_t^i) \tag{5}$$

in which Ω can be the set of all the nodes in S without the left-out sensor ($S \setminus \{s_o\}$) or the set with only the left-out sensor $\{s_O\}$. δ_t^i and $\hat{\delta}_t^i$ are measured (ground-truth) and predicted densities at time t of node $i \in \Omega$, respectively. $\mathfrak{N}_\Omega$ indicates the cardinality of Ω.

Function ℓ is a smooth L1 loss with a $\beta \in (0, 1]$ parameter to select between the linear or quadratic cost, similar to the Huber loss with a higher penalization for small errors:

$$\ell(\epsilon) = \begin{cases} \frac{1}{2\beta}\epsilon^2 & \text{if } |\epsilon| < \beta \\ |\epsilon| - \frac{\beta}{2} & \text{otherwise} \end{cases} \tag{6}$$

where $|\epsilon|$ is the absolute value of ϵ. By using the quadratic term to penalize small errors and the linear term for greater errors leads to more stable training by avoiding excessively large gradients in the presence of large errors and maintaining sensitivity to small errors, which are important for the average prediction quality. Finally, w_t^i is computed as

$$w_t^i = \frac{|\hat{\delta}_t^i - \delta_t^i|}{|\mu_{h(t)}^i - 1|} \tag{7}$$

where $\mu_{h(t)}^i$ is a factor that model the typical density of hour $h(t)$, with $t \in h(t)$, obtained as an average of historical density measurements and normalized in $[0, 1]$. Such a factor is used to give more importance to the errors made when the typical density is high. Indeed, when $\mu_{h(t)}^i$ is close to 1, then $|\mu_{h(t)}^i - 1| \to 0$, thus amplifying the error contribution in the loss. After each training epoch, a different s_O is sampled from S_R. This approach has been chosen to force the network to predict $\hat{\delta}_t$ in different positions, avoiding overfitting on specific sensor locations, something that may happen using a fixed sub-set of sensors with the density features blanked out. Moreover, s_O is selected from the reference sensor set S_R since it represents sensors located in the most central positions of the road graph which are affected by a larger number of other sensors, unlike those placed at the border of the city road graph.

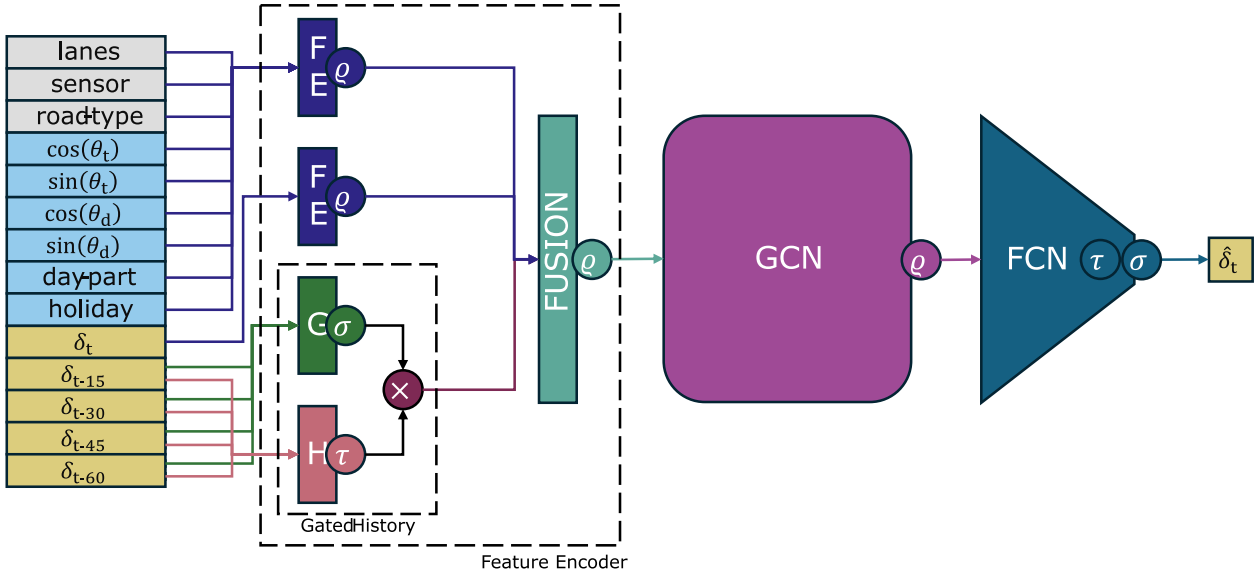

Fig. 3. Diagram of the implemented architecture. On the left the node features sent in input are reported. The static features (gray blocks) together with the dynamic temporal features (light blue blocks) are sent into a feature encoding (FE) layer. Similarly, the δ_t is sent into a different FE layer. The historic density features δ_{t-15}, δ_{t-30}, δ_{t-45}, and δ_{t-60} are sent into a gate layer (G) and an history encoding layer (H). The output of G acts as a learned weight to determine the importance of the past measurements and is multiplied with the output of H. Then all the outputs of these encodings are concatenated and sent into a fusion layer to obtain the input for the GCN. Finally, a series of FCN are used to obtain the predicted density $\hat{\delta}_t$. Activation functions are shown in round shapes with ϱ for ReLU, σ for Sigmoid, and τ for Tanh. The $\times$ symbol represent the element-wise vector product.

Table 3
Hyperparameters of the neural network architecture. For each parameter the search space is reported together with the selected optimal value found using Optuna.

Hyperparameter	Search space	Selected hyperparameter value
G–H layers width	[64, 128]	128
G activation function	[ReLU, Tanh, Sigmoid]	Sigmoid
H activation function	[ReLU, Tanh, Sigmoid]	Tanh
FE layer width	[64, 128]	128
FE activation function	[ReLU, Tanh, Sigmoid]	ReLU
Fusion layer width	[64, 128]	128
Fusion activation function	[ReLU, Tanh, Sigmoid]	ReLU
GCN number of layers	[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]	9
GCN layer width	[128, 256, 512, 1024]	512
GCN activation function	[ReLU, Tanh, Gelu]	ReLU
FCN number of layers	[2, 3, 4, 5, 6]	4
FCN activation function	[ReLU, Tanh, Gelu]	Tanh
FCN output function	[Sigmoid, No Sigmoid, Gelu, ReLU]	Sigmoid
Smooth L1 loss β	[0.05, 0.15, 0.2, 0.3]	0.15
Dropout probability	[0.3, 0.4, 0.5, 0.6]	0.4
Learning rate	[0.00005, 0.0001, 0.0005, 0.001]	0.0001
Weight decay	[0.000005, 0.0001, 0.0005]	0.00001
Batch size	[8, 16, 32]	16
Patience	[10, 15, 20, 50, 100, 150]	100

Note that, since the sensors lack direct connections and are instead interconnected through a network of road segments, being capable of correctly predicting $\hat{\delta}_t$ in the sensor positions requires the neural network to be able to properly propagate the traffic on the whole road graph.

3.3. Architecture and hyperparameter optimization

The neural architecture used to solve the TFR problem has been composed of four main parts: a gated history block, included in a feature encoder block, followed by a GCN, and a FCN. The whole network is trained end-to-end minimizing the loss function defined in the previous section. In Fig. 3, a sketch of the implemented architecture is shown. Network hyperparameters, including network depth, width, activation functions, etc., have been found thought a systematic optimization process by exploiting Optuna (2026). In Table 3, hyperparameters are reported indicating the search spaces and the found optimal values.

The input features are initially sent into the feature encoder block. Such a block is composed of two feature encoder (FE) layers, that are linear layers with ReLU activations with a width of 128 neurons, a gate layer G and an history layer H, both with 128 output dimension and using Sigmoid and Tanh activations, respectively. The first FE layer takes in input the static features together with the dynamic temporal features. The second FE layer is devoted to the encoding of traffic density δ_t. The G and H are used to learn the importance of the past density estimates. While the H layer performs an encoding of the past densities, the G layer, which output is point-wise multiplied with the output of H, is used to weigh the contribution of the encoded past densities. Then, the outputs of the two FE layers and of the product of the G and H layers are concatenated and sent into a fusion layer (an additional linear layer with output dimension 128 and ReLU activation) to obtain a single unified and rich representation. After applying feature encoding, the resulting latent representation is passed to a GCN to learn the spatial interaction

among the nodes of the road graph and compute embeddings to be exploited in the FCN to infer the missing traffic densities, solving a graph node labelling problem.

The GCN is composed of 9 layers, all using ReLU activations to compute the node embeddings. The FCN has 4 layers with Tanh activation. All GCN layers have a fixed width of 512 neurons, while the FCN layers have a decreasing width of 256, 128, 64, 32 neurons. An additional layer of 512 neurons is used to facilitate the feature remapping between the GCN and the FCN. Finally, the last layer with output dimension 1 has been used to produce the predicted density $\hat{\delta}_t$.

According to the hyperparameter values identified after the optimization (see Table 3), during training a dropout of 0.4 has been used. Learning rate has been set to 0.0001 and weight decay, used as regularization technique to penalize large weights, has been set to 0.00001. A batch size of 16 samples has been used, and 1000 maximum training epochs have been utilized, exploiting early stopping with a patience of 100 successive epochs without improvement (on the validation set) to prevent overfitting and unnecessary computations.

In the following sections details on data used and experiments carried out to validate the proposed approach are presented.

4. Datasets, test setup, and evaluation metrics

In this section the datasets used to train and test the proposed solution are introduced. Then an approach used to conduct the network tests is described. Such a strategy, in which previous estimates are used as input for the successive estimations, demonstrates how the proposed solution can be implemented in real scenarios. Finally, the method and metrics used to evaluate prediction accuracy are presented.

4.1. Datasets

In order to train and test the proposed approach real data, acquired with the Snap4City platform, from the cities of Florence and Pisa (Italy) have been used. The considered road network in Florence has been modelled in the granular G_G^F and backbone G_B^F graphs, including a total of 20,775 and 2209 road segments (i.e., nodes), respectively. Measurements from 96 sensors, producing traffic density observations every 15 min, expressed as number of vehicles per 20m ($veh/20m$), have been used. Out of the 96 sensors, 15 have been selected as reference sensors, S_R, and 81 as control sensors, S_C, as described in Section 3.2. Observations collected by the sensors over years 2022 and 2023 have been used to build different datasets to train and test the deep network. In Fig. 4, the road graph of the city of Florence is depicted and the main roads considered in the experiments are highlighted together with the available sensors, divided into reference and control sets.

The first dataset Ja2023 included 2318 samples (i.e., road graph with real sensor observations) for the whole month of January 2023. Note that, due to missing measurements, samples are fewer with respect to the total number of possible observations (i.e., 2976, that are 96 time slots per day for 31 days). Ja2023 dataset has been used to train and test the network using the following partition. First, a week composed of 672 continuous observations without missing measurements, from Thursday 19/01/2023 at 00:00 to Wednesday 25/01/2023 at 23:45, has been singled out to realize the test set, corresponding to about 29% of Ja2023 samples. Then, the remaining 71% of samples have been randomly shuffled and split into 90% training (1481 samples) and 10% validation (165 samples), equal to about 64% and 7% of Ja2023, respectively. Details on Ja2023 usage and the results achieved are reported in Section 5.1.

Three additional datasets Ju2023, Se2023, and PI2022 have been used to validate the trained network on completely unseen real road graph conditions collected at different times of the year and in different cities without retraining or fine-tuning the network weights. Ju2023 and Se2023 have been obtained from measurements acquired in Florence during June and September 2023 respectively. PI2022 has been instead collected in the city of Pisa, Italy, during January 2022. Ju2023 counts

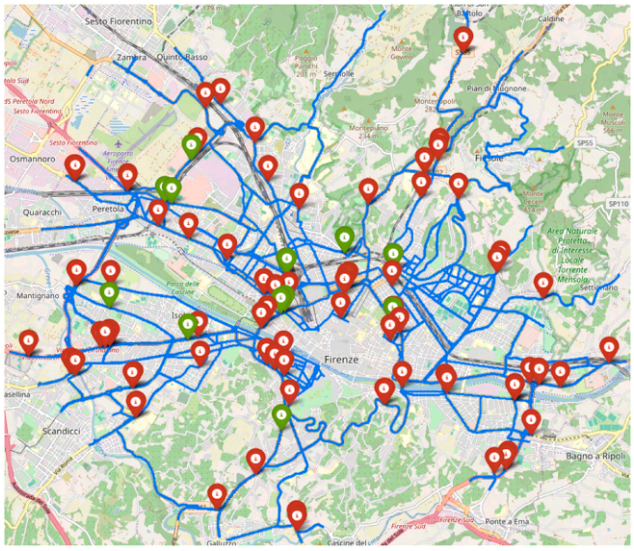

Fig. 4. Florence city road graph. Considered roads are highlighted in blue, while sensors' positions are indicated with pins. Reference sensors S_R are represented with green pins. Control sensors S_C are indicated with red pins. Note that some sensors are placed very close each other, thus some pins are occluded in the map.

Table 4
Datasets used to train and evaluate the proposed GNN-based solution.

Dataset	Number of examples	City	Road graph type	Used in train
Ja2023	2318	Florence	G_G^F and G_B^F	64%
Ju2023	2490	Florence	G_G^F	No
Se2023	2850	Florence	G_G^F	No
PI2022	2663	Pisa	G_G^P	No
JaOBU2024	672	Florence	G_G^F	No

2490 examples, and Se2023 has 2850 examples. PI2022 includes 2663 examples.

Note that PI2022 is defined on a different road graph on which its granular representation G_G^P counts 9953 road segments with only 7 traffic sensors. Since roads in Pisa are arranged in a topology completely different from the one used during training, such tests demonstrate the wide applicability of the proposed solution, able to generalize on unseen scenarios without either retraining or fine-tuning.

In order to evaluate the traffic reconstruction performance achieved on the whole road graph, a traffic dataset obtained by analyzing On-Board Units (OBU) installed on private vehicles running in Florence during the last week of January 2024 (from the 19/01/2024 at 00:00 to 25/01/2024 at 23:30, for a total of 672 examples) has been acquired from a mapping company. Such a dataset has been named JaOBU2024 and used in Section 5.2 to evaluate TFR on any road segment.

Table 4 reports a summary of the datasets.

4.2. Continuous prediction test setup

During training, samples are provided to the network randomly, while during tests samples are provided in chronological order simulating a real scenario within continuous prediction process. For such continuous estimations, traffic density features δ_{t-15}, δ_{t-30}, δ_{t-45}, and δ_{t-60} have been set to ground truth values for the sensors, while predictions made by the neural network itself at times $t-15$, $t-30$, $t-45$, and $t-60$ have been used to set the features for the unmonitored road segments. As graphically shown in Fig. 5, the historic dynamic features of each unmonitored road are set with $\hat{\delta}_t$ predictions obtained in previous iterations. By implementing such an iterative strategy for predicting $\hat{\delta}_t$ allowed the proposed method to be deployed in real scenarios.

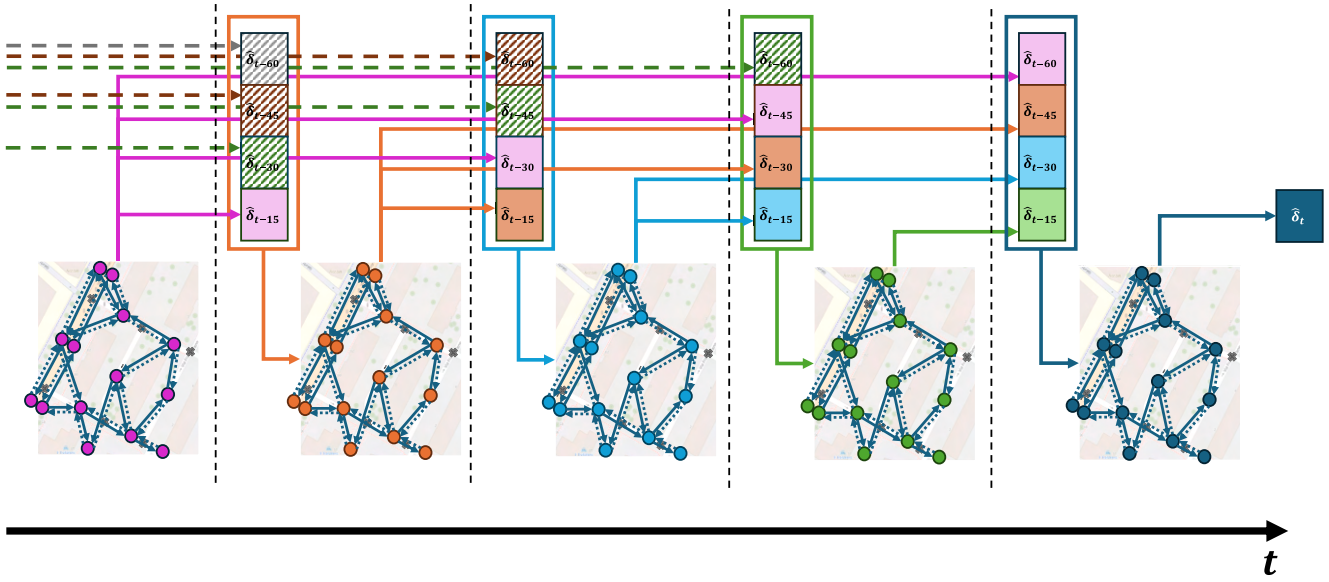

Fig. 5. Example of continuous estimation. Previous predictions are used to fill the past densities in the node features. (Best viewed in color).

To perform the cold-start of the process, different strategies have been evaluated and presented in Section 5.1 to initialize the δ values in the first iterations.

First a continuous week from each of the datasets has been used to evaluate the case in which all the sensors' measurements are available (see Sections 5.1 and 5.2). However, to deploy the proposed solution in actual scenarios, the possibility of missing observations must be considered. To fill possible temporal gaps in the sensor measurements and be able to continuously compute the TFR, typical time trends (TTTs) computed from sensors' measurements recorded in previous years have been used. The TTT of a sensor is a sequence of traffic density measurements from 00:00 to 23:45 with steps of 15 min for any day of the week of a specific month. To compute the TTT of sensor s for a specific weekday w (e.g., Monday) during a month m, all the measurements recorded during any w in m in the previous year are selected, and for each timestep, the average of the measured densities is computed. Then, to fill possible gaps in observations the value at the missing timesteps is sampled from the TTT.

By using TTTs to fill temporal gaps, further tests have been carried out on Ju2023, Se2023, and PI2022 considering the whole months as continuous sequences and reported in Section 5.2. Note that TTTs have also been used to estimate the factor $\mu^i_{h(t)}$ in Eq. (7).

4.3. Evaluation method and metrics

Ja2023, Ju2023, Se2023, and PI2022 datasets only provide ground truth densities for the road segments monitored by a traffic sensor. Following the validation approach used in Bilotta and Nesi (2021), errors have been evaluated only on the sensors' locations, bearing in mind that to obtain correct predictions in the sensors' locations requires the network to properly propagate traffic over the whole road network. A *leave-one-out* approach has been used to evaluate $\hat{\delta}_t$ predictions on sensors' locations. Let S be the set of the graph nodes having a sensor. A single node $s_k \in S$ is selected, and all its dynamic density features are blanked out, as it was representing a normal unmonitored road. The $\hat{\delta}^k_t$ prediction on s_k is obtained for all timesteps t included in the test samples and compared with the ground truth values δ^k_t. Then, this leave-one-out approach is repeated for each sensor in S.

In order to summarize the errors obtained, the Root Mean Square Error (RMSE) has been used and computed as

$$\mathrm{RMSE} = \frac{1}{\mathfrak{N}_S} \sum_{k \in S} \mathrm{RMSE}_{LOO}(k) \tag{8}$$

where $\mathfrak{N}_S$ is the cardinality of set S. The $\mathrm{RMSE}_{LOO}(k)$, that is the RMSE on the specific left-out sensor s_k for all the timesteps considered, is defined as

$$\mathrm{RMSE}_{LOO}(k) = \sqrt{\frac{1}{\mathfrak{N}_T} \sum_{t \in T} (\hat{\delta}^k_t - \delta^k_t)^2} \tag{9}$$

where T is the set of the timesteps considered (excluding eventual timesteps filled by TTT) and $\mathfrak{N}_T$ the cardinality of the set. Note that, to better understand the method performance and comply on the error reported in Bilotta and Nesi (2021), when computing the RMSE, all $\hat{\delta}^k_t$ and δ^k_t are denormalized back to actual traffic densities as number of vehicles per 20 m ($veh/20m$).

A different assessment has been used for evaluating performance on JaOBU2024, since it provides density measures on all the road segments. Thus, the mean RMSE (mRMSE) has been computed as:

$$\mathrm{mRMSE} = \frac{1}{\mathfrak{N}_{\bar{S}}} \sum_{k \in \bar{S}} \sqrt{\frac{1}{\mathfrak{N}_T} \sum_{t \in T} (\hat{\delta}^k_t - \delta^k_t)^2} \tag{10}$$

where $\bar{S}$ indicate the set of road segments on which the density measure has been blacked out during the tests and predicted by the network.

5. Experimental results

In this section, experiments carried out to validate the proposed approach are described. In Section 5.1, ablation study shows the impact of different neural architectures, graph representations, and initializations of the continuous prediction approach. Section 5.2 is devoted to show performance of the proposed solution on data not used in training, conducting tests on different periods of the year, on a different city, and by changing the sensors' positions. Then, in Section 5.3 computational times for training and inference are reported, and, finally, a discussion on the obtained results is provided in Section 5.4.

5.1. Ablation study

To better understand the contribution of each component in the proposed model, we conduct an ablation study divided into three parts. In the first part, specific components of the neural network architecture have been systematically removed to assess their impact on the prediction performance. The full architecture, including the gated history (GH) and feature encoder (FE) blocks, has been named TFRGCN + GH + FE.

Table 5
Result of the ablation studies. Reported RMSE (according to Eq. (8)) of traffic flow densities are expressed in vehicles per 20 m, in bold the best results.

Experiment	TFR-0	TFR-E	TFR-TTT
TFRGCN + GH + FE	0.33	0.33	0.33
TFRGCN + GH	0.38	0.37	0.37
TFRGCN	0.72	0.72	0.72
TFRGCN + GH + FE-G_B	**0.21**	**0.21**	**0.21**
TFRGCN + GH + FE-$\hat{e}$	0.38	0.36	0.36

With TFRGCN + GH is indicated the architecture without the feature encoder layers FE and still using the GH layers, while TFRGCN indicates the architecture excluding both the GH and FE layers. All these three experiments used granular graph representation G_G. In the second part, the graph representation has been changed from granular to backbone to evaluate the influence of the number of nodes. This experiment has been indicated by TFRGCN + GH + FE-G_B. Finally, the impact of the graph modeling approach using dummy edges to better model traffic flow dynamics has been evaluated: the same architecture used in test TFRGCN + GH + FE has been evaluated using in input a graph without dummy edges; this test case has been named TFRGCN + GH + FE-$\hat{e}$. These experiments have been carried out on the test week of Ja2023 and evaluated according to the recursive approach described in Section 4.2.

Three different initialization procedures have been considered to set the historic dynamic density features of the unmonitored road and the left-out sensor during the first iterations. In the first case, named TFR-0, the past traffic flow densities have been set to zero in all segments, i.e., a cold start. In the second case, named TFR-E, traffic density values estimated with the method proposed in Bilotta and Nesi (2021) have been used to kick off the recursive estimation. Finally, TTTs have been exploited as starting traffic flow data, and the case has been named TFR-TTT. For all the cases, RMSEs have been reported according to Eq. (8).

Obtained results have been reported in Table 5. It is worth noting that, Bilotta and Nesi (2021) reported an RMSE of 0.5 as a small error being a fraction of a vehicle in a space of $20m$, and proposed a solution achieving a RMSE close to 0.30. In particular, this assumption does not take into account the normalization of the road values with respect to the number of lanes, as in Fig. 2. In an urban network, such as the city of Florence, several primary roads have more than one lane and they are characterized by heavy traffic conditions. In this context, a value of 2.5 cars per 20 m is quite typical during daytime measurements at the traffic sensor locations (with peaks in highly congested areas reaching up to 3.22 veh/20m on a single lane). Then, the results in Bilotta and Nesi (2021), and those in this paper, are not directly reflected in Fig. 2, where the data are averaged over the network, which includes many residential areas where traffic is typically lower, and lane normalization is applied.

Observing Table 5, results are practically independent from the initial set up, except for TFRGCN + GH and TFRGCN + GH + FE-$\hat{e}$ that achieve lower error using TFR-E and TFR-TTT. The full architecture (TFRGCN + GH + FE) outperforms all the other solutions (TRFGCN + GH, TFRGCN) demonstrating the importance of learning a mapping of node features in an expanded latent representation to allow the GCN to better understand the node interactions. In particular, the introduction of the gated history block discourages the network from relying on the historic densities and focus more on the node message passing avoiding diverging during recursive estimations. Moreover, regarding TFRGCN + GH + FE, TFR-0 can be used without loss of performance. This is a positive fact demonstrating that the recursive estimation can be initialized without requiring either external estimations or historical data.

As expected, working with simpler graphs and fewer nodes, as when using the backbone representation G_B, helps in obtaining lower errors. In G_G, the percentage of monitored roads with respect to the total number of roads is 0.46%, while in G_B this percentage rises to 5%. However, such a representation provides coarser results that could be insufficient to understand the traffic evolution over time, as it can be observed in Fig. 6 in which the TFR evolution of the same area at the same times using G_G and G_B is presented. Using G_G allows to obtain detailed reconstructions, better illustrating the evolution of traffic congestion. On the other hand, with G_B most of the traffic dynamic is lost due to the excessive length of the road segment. For example, in Fig. 6a, b, and c, it is possible to observe spatial and time-varying congestion of the road *Via della Torretta* since the road is split into subsegments of 20 m. Differently, the same area in Fig. 6d, e, and f presents constant congestion level. Note however that, the proposed solution is able to handle any kind of graph representation with road of arbitrary length, differently from Bilotta and Nesi (2021) that, relying on the resolution of a fluid dynamic model with finite difference, requires a fixed and proper setting related to the road lengths.

Finally, TFRGCN + GH + FE-$\hat{e}$, in which dummy edges have been removed from the graph, achieves a higher RMSE with respect to TFRGCN + GH + FE: the introduction of dummy edges allow to better model the traffic flow and to obtain lower errors. Moreover, as shown in Fig. 7, by inspecting the reconstructed maps, it is possible to observe that without dummy edges traffic reconstruction appears more static. Differently, when using dummy edges, more dynamic and realistic flows are observed. For example, note that in Fig. 7l, obtained without dummy edges, roads remain congested even during nighttime, while road congestion evidently decreases using dummy edges (see Fig. 7f).

Given the obtained results, in the successive experiments the TFRGCN + GH + FE architecture with the G_G representation (including dummy edges) and initialization at zero of the past densities (TFR-0) has been used.

5.2. Performance on unseen scenarios

In this section, results obtained by using Ju2023, Se2023, PI2022, and JaOBU2024 are presented. Since those datasets have not been seen by the network during training, and neither retraining nor fine-tuning has been used, these experiments allow us to demonstrate transferability capabilities of the proposed approach to work on unseen scenarios. Thus, the proposed work overcomes the limitations of state-of-the-art methods, as (Syum Gebre et al., 2025), where the effectiveness of the approach is validated only in a fixed real-world traffic scenario (e.g., the US-101 freeway).

In particular, using Ju2023 and Se2023, it has been possible to observe how the network is able to compute the TFR on different periods of the year, during which traffic can change, for example, due to different citizen behavior (e.g., schools close in June) and different weather. On the other hand, using PI2022 we want to show how the proposed approach is able to work without retraining even in new cities having a different road graph and unseen traffic patterns. JaOBU2024 has been used to quantify the reconstruction accuracy on every road segment and to evaluate the impact of sensors' positions. Finally, an additional test on a single day of January 2024 has been carried out to observe how TFR changes in presence of road closures, simulating a what-if scenario.

Considering experiments using Ju2023, Se2023, and PI2022, as anticipated in Section 4.2, first a single week without missing sensor measurements have been sampled from each dataset and used in the tests, similarly to what have been performed in Section 5 by using the test week of Ja2023. Such tests have been labeled as W-TFR. Then, possible temporal gaps in the sensors' observation have been filled using TTTs and new tests have been carried out considering the whole set of months, labelling this case as M-TFR. In all these experiments, the full TFRGCN + GH + FE architecture has been used, the granular graph representation G_G have been selected (with dummy edges), and past densities have been initialized at zero. The leave-one-out evaluation method has been used, according to Section 4.3 and Eqs. (8) and (9).

In Table 6, the achieved RMSEs (expressed as number of vehicles per 20 m) have been reported. As it can be seen, in all cases the proposed

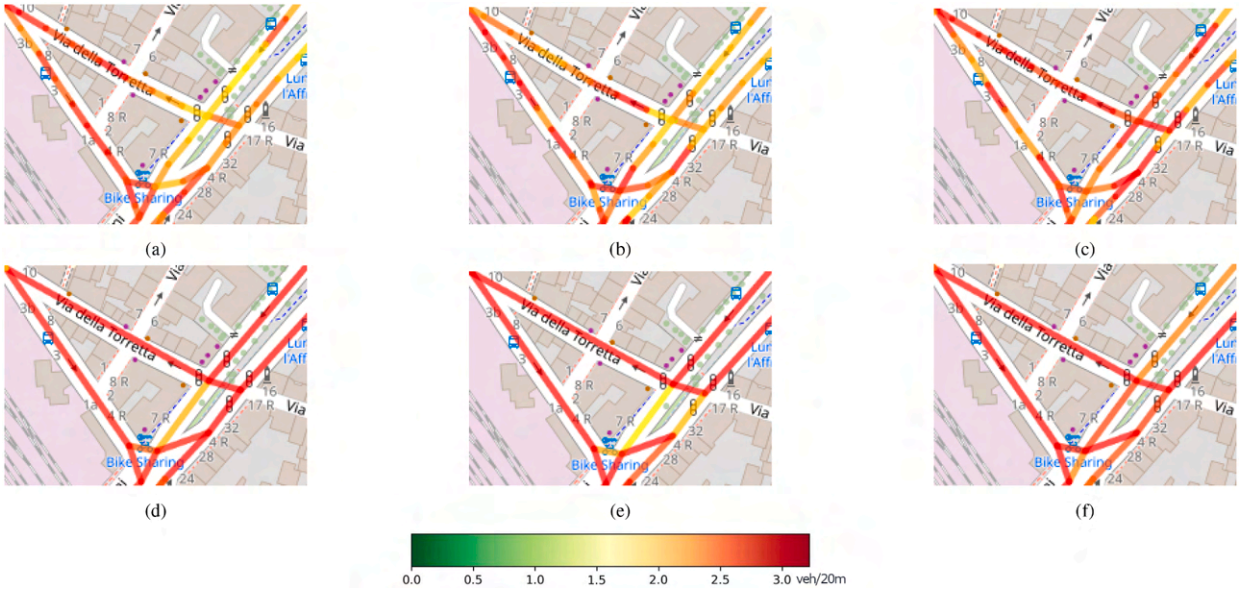

Fig. 6. Examples of traffic reconstructions obtained using different graph representations. Traffic densities are represented in color from green to red according to the level of congestion following the color-bar below the figures. In (a), (b), and (c) successive time instant reconstruction using the granular graph G_G. In (d), (e), and (f) the same times reconstructed using the backbone G_B graph. It can be noticed that, using G_G it is possible to appreciate finer evolution of congestions, while using G_B most of the details are lost due to the usage of longer road segments producing a density averaging. (Better viewed in color).

Table 6
Results of experiments on datasets not used in training. Reported RMSE of traffic densities are expressed in vehicles per 20 m.

Dataset	W-TFR	M-TFR
Ju2023	0.38	0.36
Se2023	0.37	0.35
PI2022	0.15	0.16

solution has been capable to provide accurate predictions, obtaining low RMSE values. Even if with respect to the RMSE obtained on the test week of Ja2023 there is a slight increase in the single week test (W-TFR) of Ju2023 and Se2023, achieved errors remains low and demonstrated the applicability of the proposed solution on unseen data from different periods of the year. A noticeably lower RMSE has been obtained for PI2022. This is probably due to the graph of Pisa: being a smaller city the graph requires a minor number of nodes (9953 against 20,775 of Florence), something that has been shown in Section 5.1 to be a relevant characteristic of the TFR problem.

Regarding the monthly tests (M-TFR), only very minimal RMSE variations with respect to week test (W-TFR) have been obtained, demonstrating the stability of the proposed solution even when used during longer periods of time.

In order to quantitatively evaluate the reconstruction accuracy on the whole graph, JaOBU2024 dataset has been used, since it contains measurements for all the road segments. Road segments corresponding to sensors' position in the other datasets in Florence (i.e., Ja2023, Ju2023, and Se2023) have been selected as virtual sensors to provide input to the neural network. Neither retraining nor fine-tuning have been used. Then, the predicted density values for the road segments considered have been compared with the measurements. Note that in this case the leave-one-out testing approach was not used; instead all virtual sensors have been used in input and results have been evaluated on all the unmonitored roads, following Eq. (10). An mRMSE of 0.05 veh/20m has been achieved, demonstrating how the proposed method is able to correctly estimate traffic densities on the full road graph. It is observed that, the achieved error is lower with respect to those obtained with the other datasets, in which data comes from traffic sensors. This reflects the different nature of data: readings from OBUs provide a sample of the urban traffic. Nevertheless, JaOBU2024 preserves the relative distribution of vehicles over the road network. Therefore, the obtained low error highlights the capability of the proposed method to reconstruct the spatial dynamics of the vehicle density, capturing the traffic propagation over the road graph.

Finally, to evaluate the robustness of the proposed method with respect to different sensors' positions and temporary road closures, two additional tests have been carried out.

JaOBU2024 has been used to evaluate the impact of different sensors' positions. Starting from the positions of the virtual sensors (coinciding with the real positions of the sensors in Florence), each sensor has been randomly moved to a different road: the new monitored roads have been selected by sampling from a bivariate Gaussian distribution centered on the original position and with a standard deviation σ, with the additional constraint that the new road segments must have the same road-type of the original road in which the real sensor is placed (e.g., a sensor on a primary road is moved only to a different primary road). Two standard deviations have been considered: $\sigma = 100$m and $\sigma = 500$m. In this case, errors have been computed with Eq. (10), yet using as true values the densities reconstructed considering the sensors placed in the real positions. In Fig. 8, the hour-by-hour mRMSE have been shown (note that the mRMSE scale range from 0 to 0.00035 veh/20m). In both cases identical densities or with negligible differences have been observed, with mRMSE on the all timestamps achieving 0.00014 and 0.00015 veh/20m for $\sigma = 100$m and $\sigma = 500$m, respectively . This confirms that the proposed method is able to handle sensors in any position without requiring retraining or fine-tuning.

As a final test, the impact of road closures on the reconstructions of the proposed TFR method has been evaluated. An analysis has been provided as follows. Two TFRs have been computed considering a day in the test set of Ja2023 (20/01/2023): the first TFR is obtained without any change in the road graph, the second one after blocking (i.e., removing from the graph) two critical road segments (indicated with the purple rectangle in Fig. 9a). By comparing the reconstructions, with-

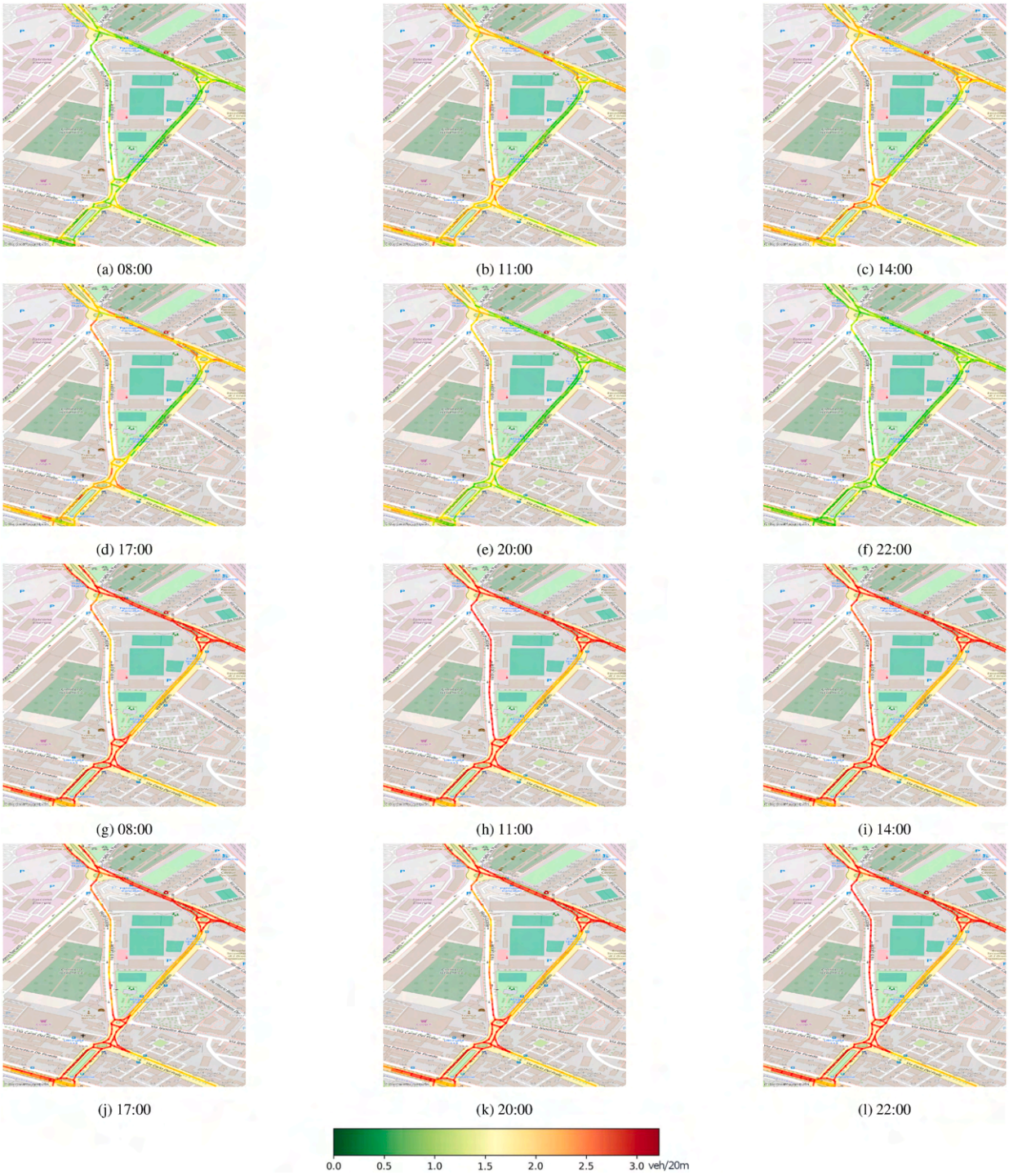

Fig. 7. Comparison of traffic reconstruction results using in input a graph with and without dummy edges. Level of traffic is encoded in color from green to red according the color-bar reported at the bottom of the figure. On the top row, from (a) to (f), results using the dummy edges, for a given day, in different times. In the bottom row, from (g) to (l), results without dummy edges, for the same day and times. As can be seen, without dummy edges the reconstruction appear more static over the day, while the introduction of dummy edges allow the net to better model traffic flow dynamics. (Better viewed in color).

out (Fig. 9a) and with road closure (Fig. 9b), it is evident how traffic reasonably redistributes in the local road network due to the closure, increasing the level of congestion in all the neighbor segments. This indicates that the proposed solution can be employed in what-if scenarios, providing indications about the effects of the introduced changes.

5.3. Computational times

Training and evaluations have been carried out on a workstation equipped with an Intel i9-13900 CPU, 125 GB RAM, exploiting a Nvidia GeForce RTX 4090 GPU with 24 GB of memory. The neural network has

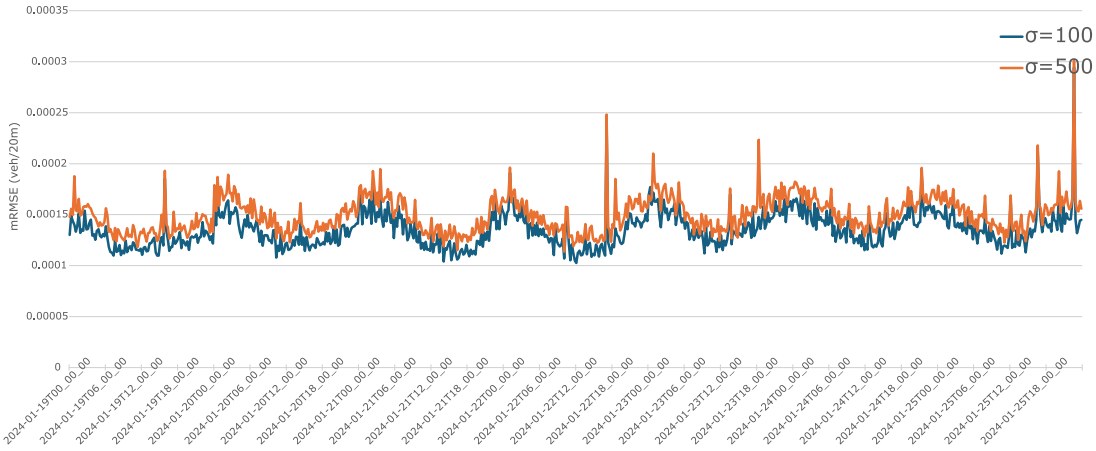

Fig. 8. Hour-by-hour RMSE (expressed as veh/20m) comparing the TFR obtained after moving the sensors' position using bivariate Gaussian with $\sigma = 100$m (blue) and $\sigma = 500$m (orange). (Better viewed in color).

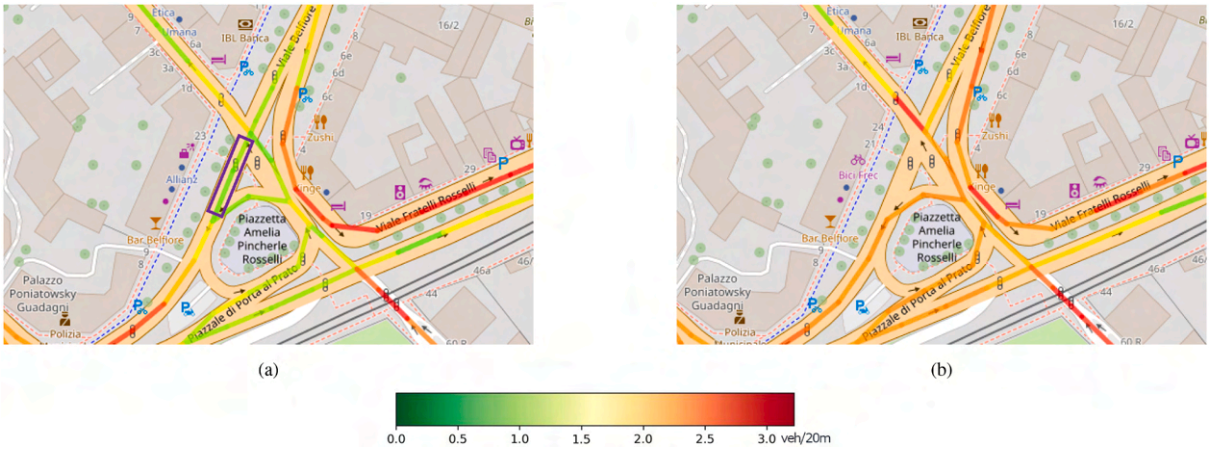

Fig. 9. Simulation of road closure. In (a) the usual condition. The purple rectangle highlight two road segments that have been removed in (b) simulating a road closure. As can be seen comparing the reconstruction obtained at the same time (20/01/2023 at 17:45), the removal of the two segments produce evident changes in traffic densities with all the road presenting increasing level of congestion. (Better viewed in color).

been trained for 301 epochs in 14 h. During inference, computing the TFR on the full city of Florence for 24 h, i.e. for 96 time-instants equal to four 15-min time slot per hour, requires 15 s.

Such computational times confirm that the proposed solution can run quickly on consumer hardware, thanks to its light-weight architecture, and allow its exploitation in real-time on real urban scenarios.

5.4. Discussion

The present study considers a specific problem within a well-defined urban scenario in a real-world context.

The experimental results presented in the previous sections confirm the validity of the proposed approach in solving the TFR problems. Comparing the predicted values with the ground truth traffic densities yielded RMSEs between 0.3 and 0.4 vehicles per 20 m, achieving also lower errors for easier setup (i.e., considering the backbone graph representation and working on small city like Pisa). Such RMSE indicate small discrepancies, in the order of a fraction of a vehicle.

Comparing the results obtained with those reported in Bilotta and Nesi (2021), both solutions are able to produce a full TFR in just a few seconds achieving a similar accuracy. Nevertheless, exploiting a deep learning approach instead of relying on the explicit resolution of the underlying fluid dynamic problem, has several advantages.

First, the proposed solution can be applied to any kind of graph, while Bilotta and Nesi (2021) required to preprocess the road graph by means of spatial units needed to solve the TFR problem using finite difference methods. Also finding the correct number of iterations of finite differences to obtain a reasonable estimate is not easy and requires careful tests, something that can be completely avoided using the proposed solution.

The most relevant benefit of the proposed solution is its wide deployability. As demonstrated in Section 5.2 the network can be reliably used on unseen scenarios without requiring retraining or fine-tuning. Differently, Bilotta and Nesi (2021) depends on the estimation of TDMs that encode the traffic distribution at junctions. TDMs are required for each time instant for each day of the week and must be recomputed for different road graphs. Due to the time required for estimating TDMs (several days using GPUs to solve a stochastic relaxation problem) deploying the fluid dynamic TFR solver on novel cities is not an easy task. Note that this limit can be further critical when working on what-if analysis or when embedding the TFR tool into automatic optimizations (Bellini et al., 2024): computing the TFR using (Bilotta & Nesi, 2021) on a modified road graph would require novel TDMs, extremely slowing down the possibility of simulating alternative scenarios, while the proposed solution could be readily applied.

For these reasons, the proposed method is a valid alternative to state-of-the-art solutions for solving the TFR problem, easily deployable on any scenario and producing accurate traffic density estimates.

6. Conclusions

In this paper a novel solution for solving the TFR problem is presented. TFR aims at estimating the vehicle densities for all the segments in a road graph starting from few and sparse measures to allow traffic control activities and management without requiring a widespread, pervasive, and costly deployment of sensors. The proposed approach exploits a lightweight deep learning architecture based on GNN to overcome limitations of existing solution based on complex neural architectures or finite difference solvers that are difficult to adapt to different urban contexts without significant redesign. In this work, the experimental evaluation is conducted on real-world large-scale road networks. In particular, a network comprising 20,775 road segments is considered, of which only 96 are equipped with sensors. Many GNN-based methods at the state-of-the-art, including GCN (S & Zacharias, 2024), autoencoders (Xia et al., 2024), and reinforced dynamic GCNs for traffic prediction (Chen & Chen, 2022), have been proposed for reconstructing missing data in graph-structured datasets. However, these approaches have significant limitations in traffic reconstruction problems under extreme data sparsity (1-3% observed data). Recent efforts to integrate LLMs with GNNs (Chen et al., 2024; Ren et al., 2024) improve general graph learning but are not directly applicable to TFR, which relies exclusively on spatio-temporal numerical data. Moreover, existing physics-informed deep learning (PIDL) methods often do not consider road-crossing modeling at junctions, limiting their applicability in real-world urban traffic networks (Huang & Agarwal, 2022; Rempe et al., 2021; Shi et al., 2021). Similarly, PINN-TSE frameworks (Syum Gebre et al., 2025) achieve real-time traffic estimation on sparse sensor data, but are related to fixed scenarios (e.g., the US-101 freeway) and lack generalizability. In contrast, the approach proposed in this work ensures adaptability across multiple datasets and network configurations, enabling its applicability in real-world urban traffic settings.

Thanks to a carefully designed road graph representation capturing forward and backward traffic propagation, and well-designed node characteristics, the system is able to compute accurate reconstructed traffic data. The network exploits feature encoding blocks to obtain a rich latent feature representation, graph convolutional layers to learn the node interaction and fully connected layers to regress the density values. In addition, the introduction of a leave-one-out training strategy allows the model to be trained without requiring full ground-truth density maps, while still ensuring robust transferability across different sensor configurations and urban scenarios. Moreover, a continuous traffic estimation strategy has been proposed to enable the method to be reliably applied in real scenarios. Extensive experiments demonstrate the wide applicability of the network on data and scenarios not seen during training achieving an RMSE error of 0.33 veh/20m on sensors' measurements, and a mean RMSE of 0.05 veh/20m on the whole road network.

The study in Bilotta and Nesi (2021) directly addressed the traffic reconstruction problem within the same application domain. However, it relies on a stochastic relaxation method to estimate TDMs by iteratively testing and refining road assignment configurations until the overall error is minimized. While that strategy can achieve accurate traffic reconstructions, it is computationally heavy in terms of processing time. Moreover, the TDMs estimation depends on the characteristics of a given road network. In contrast, the approach presented in this paper offers a significantly more efficient and scalable solution, enabling direct adaptability across different traffic scenarios. In particular, without requiring any retraining or fine-tuning, the proposed solution is able to obtain accurate density estimates for different time periods, various road graph representations and sensors' positions, and diverse urban context. Furthermore, the proposed method can be used to perform what-if analysis to evaluate impacts of road closures. Such capabilities together with fast execution on consumer hardware make the proposed solution a valid tool for intelligent transport systems supporting decision makers in managing urban traffic.

CRediT authorship contribution statement

Nathan Acciai: Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Software, Validation, Visualization; **Stefano Bilotta:** Conceptualization, Formal analysis, Investigation, Methodology, Supervision, Validation, Visualization, Writing – original draft, Writing – review & editing; **Marco Fanfani:** Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Supervision, Validation, Visualization, Writing – original draft, Writing – review & editing; **Paolo Nesi:** Conceptualization, Formal analysis, Funding acquisition, Investigation, Methodology, Project administration, Resources, Supervision, Validation, Writing – review & editing.

Data availability

Data will be made available on request

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.

Acknowledgment

This study has been carried out in the context of the Spoke 9 of the Italian National Center for Sustainable Mobility (CN MOST), and the CN MOST flagship project OPTIFaaS and scalability project SASUAM.

References

Adhikari, D., Jiang, W., Zhan, J., et al., (2023). A comprehensive survey on imputation of missing data in internet of things. *ACM Computing Surveys*, *55*(7), 133.

Adreani, L., Bellini, P., Bilotta, S., Bologna, D., Collini, E., Fanfani, M., & Nesi, P. (2024a). Smart city scenario editor for general what-if analysis. *Sensors*, *24*(7), 2225.

Adreani, L., Bellini, P., Colombo, C., et al., (2024b). Implementing integrated digital twin modelling and representation into the snap4city platform for smart city solutions. *Multimedia Tools and Applications*, *83*, 37121–37146.

Adreani, L., Bellini, P., Fanfani, M., Nesi, P., & Pantaleo, G. (2024c). Smart city digital twin framework for real-time multi-data integration and wide public distribution. *IEEE Access*, *12*, 76277–76303.

Alam, G. M. I., Tanim, S., Sarker, S., et al., (2025). Deep learning model based prediction of vehicle CO2 emissions with explainable AI integration for sustainable environment. *Scientific Reports*, *15*, 3655.

Alberti, F., Alessandrini, A., Bubboloni, D., Catalano, C., Fanfani, M., Loda, M., Marino, A., Masiero, A., Meocci, M., Nesi, P., & Paliotto, A. (2023). Mobile mapping to support an integrated transport-territory modelling approach. In *The international archives of the photogrammetry, remote sensing and spatial information sciences* (pp. 1–7). (*vol. 48*).

Anciaes, P., Cheng, Y., & Watkins, S. (2025). Policy measures to reduce road congestion: What worked? *Journal of Transport & Health*, *41*, 101984.

Bellini, P., Bilotta, S., Collini, E., Fanfani, M., & Nesi, P. (2023). Mobility and transport data for city digital twin modeling and exploitation. In *IEEE international smart cities conference (isc2), Bucharest, Romania* (pp. 1–7).

Bellini, P., Collini, E., Fanfani, M., Palesi, L. A. I., & Nesi, P. (2024). Smart city digital twin platform architecture for mobility and transport decision support systems. In *2024 IEEE international conference on big data (bigdata)* (pp. 5486–5495).

Bilotta, S., Bonsignori, V., & Nesi, P. (2024). High precision traffic flow reconstruction via hybrid method. *IEEE Transactions on Intelligent Transportation Systems*, *25*(5), 4066–4076.

Bilotta, S., Collini, E., Nesi, P., & Pantaleo, G. (2022). Short-term prediction of city traffic flow via convolutional deep learning. *IEEE Access*, *10*, 113086–113099.

Bilotta, S., Fereidooni, Z., Ipsaro Palesi, L., & Nesi, P. (2025). Macroscopic ga-based multi-objective traffic light optimization prioritizing tramways. *Applied Soft Computing*, *178*, 113269.

Bilotta, S., & Nesi, P. (2021). Traffic flow reconstruction by solving indeterminacy on traffic distribution at junctions. *Future Generation Computer Systems*, *114*, 649–660.

Bilotta, S., & Nesi, P. (2022). Estimating CO2 emissions from IoT traffic flow sensors and reconstruction. *Sensors*, *22*(9), 3382.

Bouktif, S., Cheniki, A., Ouni, A., & El-Sayed, H. (2023). Deep reinforcement learning for traffic signal control with consistent state and reward design approach. *Knowledge-Based Systems*, *269*, 110440.

Caraveo Mena, C., Suastegui Macias, J., et al., (2025). Design and implementation of a distributed IoT system for monitoring of gases emitted by vehicles that use biofuels. *Sustainability*, *17*(3), 1153.

Cardaliaguet, P., & Souganidis, P. (2024). An optimal control problem of traffic flow on a junction. *ESAIM: Control, Optimisation and Calculus of Variations*, *30*, 88.

Chen, Y., & Chen, X. (2022). A novel reinforced dynamic graph convolutional network model with data imputation for network-wide traffic flow prediction. *Transportation Research Part C*, *143*, 103820.

Chen, Z., Mao, H., Li, H., Jin, W., Wen, H., Wei, X., Wang, S., Yin, D., Fan, W., Liu, H., & Tang, J. (2024). Exploring the potential of large language models (llms) in learning on graphs. *SIGKDD Explor. Newsl.*, *25*(2), 42–61.

Cheng, Y., Watkins, S., & Anciaes, P. (2024). What interventions are effective in reducing congestion? In *Advances in transport policy and planning* (pp. 197–225). Academic Press (*vol. 13*).

Elassy, M., Al-Hattab, M., Takruri, M., & Badawi, S. (2024). Intelligent transportation systems for sustainable smart cities. *Transportation Engineering*, *16*, 100252.

Gheorghiu, R., Iordache, V., & Stan, V. (2021). Urban traffic detectors – comparison between inductive loop and magnetic sensors. In *International conference on electronics, computers and artificial intelligence (ECAI)* (pp. 1–4).

Huang, A. J., & Agarwal, S. (2022). Physics-informed deep learning for traffic state estimation: Illustrations with LWR and ctm models. *IEEE Open Journal of Intelligent Transportation Systems*, *3*, 503–518.

Kaiser, S., Rodrigues, F., Lima Azevedo, C., & Kaack, L. (2026). Spatio-temporal graph neural network for urban spaces: Interpolating citywide traffic volume. *Expert Systems with Applications*, (p. 131823).

Khan, A., Fouda, M., Do, D.-T., et al., (2023). Short-term traffic prediction using deep learning long short-term memory: Taxonomy, applications, challenges, and future trends. *IEEE Access*, *11*, 94371–94391.

Khan, D., Khan, Z., Imran, W., et al., (2022). Macroscopic flow characterization at t-junctions. *Transportation Research Interdisciplinary Perspectives*, *14*, 100591.

Kumar, R., Sharma, N., & Chaurasiya, V. (2024). Adaptive traffic light control using deep reinforcement learning technique. *Multimedia Tools and Applications*, *83*, 13851–13872.

Lee, G., Ding, Z., & Laval, J. (2023). Effects of loop detector position on the macroscopic fundamental diagram. *Transportation Research Part C*, *154*, 104239.

Li, J., & Wang, J. (2025). Real-time traffic flow prediction model based on deep learning: Taking smart highways as an example. *International Journal of High Speed Electronics and Systems*, (p. 2540239).

Linardatos, P., Papastefanopoulos, V., Panagiotakopoulos, T., et al., (2023). CO2 Concentration forecasting in smart cities using a hybrid ARIMA–TFT model on multivariate time series IoT data. *Scientific Reports*, *13*, 17266.

Liu, B., Long, J., Deng, M., Yang, X., & Shi, Y. (2022). An adaptive route planning method of connected vehicles for improving the transport efficiency. *ISPRS International Journal of Geo-Information*, *11*(1), 39.

Liu, Z., Zhou, P., Li, Z., & Li, M. (2019). Think like a graph: Real-time traffic estimation at city-scale. *IEEE Transactions on Mobile Computing*, (pp. 18:2446–2459). https://doi.org/10.1109/ITSC.2018.8569938

Lopez, P., Behrisch, M., Bieker-Walz, L., Erdmann, J., Flötteröd, Y.-P., Hilbrich, R., Lücken, L., Rummel, J., Wagner, P., & Wiessner, E. (2018). Microscopic traffic simulation using sumo. In *2018 21st international conference on intelligent transportation systems (ITSC)* (pp. 2575–2582).

OpenStreetMap (2026). OpenStreetMap. https://www.openstreetmap.org. Accessed: January 14, 2026.

Optuna (2026). Optuna: Hyperparameter Optimization Framework. https://optuna.org. Accessed: January 14, 2026.

Rahmani, S., Baghbani, A., Bouguila, N., & Patterson, Z. (2023). Black for intelligent transportation systems: A survey. *IEEE Transactions on Intelligent Transportation Systems*, *24*(8), 8846–8885.

Raissi, M., Perdikaris, P., & Karniadakis, G. (2019). Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, *378*, 686–707.

Rempe, F., Loder, A., & Bogenberger, K. (2021). Estimating motorway traffic states with data fusion and physics-informed deep learning. In *2021 IEEE international intelligent transportation systems conference (ITSC)* (pp. 2208–2214).

Ren, X., Tang, J., Yin, D., Chawla, N., & Huang, C. (2024). A survey of large language models for graphs. In *Proceedings of the 30th ACM SIGKDD conference on knowledge discovery and data mining, KDD '24, Barcelona, Spain* (pp. 6616–6626).

S, A., & Zacharias, J. (2024). Handling missing data in graph networks with adaptive graph convolutional network. In *2024 International conference on advancements in power, communication and intelligent systems (apci)* (pp. 1–6).

Saliev, D. N., & Damyanov, I. S. (2024). Traffic flow density study at traffic light junction. *AIP Conference Proceedings*, *3078*, 050004.

Santana, E. F. Z., Lago, N., Kon, F., & Milojicic, D. S. (2018). InterSCSimulator: Large-scale traffic simulation in smart cities using erlang. In Dimuro, G. P., Antunes, & L. (Eds.), *Multi-agent based simulation XVIII* (pp. 211–227). Springer International Publishing.

Shi, R., Mo, Z., Huang, K., Di, X., & Du, Q. (2021). Physics-informed deep learning for traffic state estimation. arXiv:2101.06580.

Sun, T., Zhu, S., Hao, R., et al., (2022). Traffic missing data imputation: A selective overview of temporal theories and algorithms. *Mathematics*, *10*(14), 2544.

Syum Gebre, T., Endale Ashebir, S., Blay, J., Anokye, M., Pandey, V., & Hashemi-Beni, L. (2025). Real-time traffic insights with physics-informed neural networks: Integrating the AW-Rascle model and LLMs. *IEEE Access*, *13*, 181246–181266.

Thabit, A. S. M., Kerrache, C., & Calafate, C. (2024). A survey on monitoring and management techniques for road traffic congestion in vehicular networks. *ICT Express*, (pp. 1186–1198).

United Nations (2026). Sustainable development goals (SDG): Sustainable transport. https://sdgs.un.org/topics/sustainable-transport. Accessed January 14, 2026.

Wang, C., Atkison, T., & Park, H. (2024). Dynamic adaptive vehicle re-routing strategy for traffic congestion mitigation of grid network. *International Journal of Transportation Science and Technology*, *14*, 120–136.

Xia, R., Zhang, C., Li, A., et al., (2024). Attribute attribution autoencoders for attribute-missing graphs. *Knowledge-Based Systems*, *291*, 111583.

Xing, J., Wu, W., Cheng, Q., & Liu, R. (2022). Traffic state estimation of urban road networks by multi-source data fusion: Review and new insights. *Physica A: Statistical Mechanics and its Applications*, *595*, 127079.

Zheng, J., Wang, M., & Huang, M. (2024). Exploring the relationship between data sample size and traffic flow prediction accuracy. *Transportation Engineering*, *18*, 100279.