Hypergraph-Enhanced Multi-Granularity Stochastic Weight Completion in Sparse Road Networks

XIAOLIN HAN, Northwestern Polytechnical University, Xi'an, China and Laboratory for Advanced Computing and Intelligence Engineering, Wuxi, China
YIKUN ZHANG, Northwestern Polytechnical University, Xi'an, China
CHENHAO MA, The Chinese University of Hong Kong—Shenzhen, Shenzhen, China
XUEQUN SHANG, Northwestern Polytechnical University, Xi'an, China
REYNOLD CHENG, The University of Hong Kong, Hong Kong, Hong Kong
TOBIAS GRUBENMANN, Edinburgh Napier University, Edinburgh, United Kingdom of Great Britain and Northern Ireland
XIAODONG LI, Xiamen University, Xiamen, China and Key Laboratory of Multimedia Trusted Perception and Efficient Computing, Ministry of Education of China, Xiamen, China

Road network applications, such as navigation, incident detection, and Point-of-Interest (POI) recommendation, make extensive use of network edge weights (e.g., traveling times). Some of these weights can be missing, especially in a road network where traffic data may not be available for every road. In this article, we study the *stochastic weight completion* (SWC) problem, which computes the weight distributions of missing road edges. This is difficult, due to the intricate temporal and spatial correlations among neighboring edges. Besides, the road network can be *sparse*, i.e., there is a lack of traveling information in a large portion of the network. To tackle these challenges, we propose a multi-granularity framework for **Region-Wise Graph Completion (RegGC)**. To learn coarse spatial correlations among distantly located roads, we construct a region-wise hypergraph neural architecture based on semantic region dependencies. For finer spatial correlations, we incorporate contextual road network properties (e.g., speed limits, lane counts, and road types). Moreover, it

Xiaolin Han, Xuequn Shang, and Yikun Zhang were supported by the National Natural Science Foundation of China (Nos. 62302397, 62433016), the fund of Laboratory for Advanced Computing and Intelligence Engineering (No. 2023-LYJJ-01-021), and Fundamental Research Funds for the Central Universities, China under Grant No. D5000230191. Chenhao Ma was partially supported by NSFC under Grant No. 62302421, Basic and Applied Basic Research Fund in Guangdong Province under Grant No. 2023A1515011280, Ant Group through CCF-Ant Research Fund, Shenzhen Research Institute of Big Data under Grant No. SIF20240004, and the Guangdong Provincial Key Laboratory of Big Data Computing, The Chinese University of Hong Kong, Shenzhen. Reynold Cheng and Xiaodong Li were supported by the Hong Kong Jockey Club Charities Trust (Project 260920140), the University of Hong Kong (Project 2409100399), the HKU Outstanding Research Student Supervisor Award 2022–2023, and the HKU Faculty Exchange Award 2024 (Faculty of Engineering).
Authors' Contact Information: Xiaolin Han, Northwestern Polytechnical University, Xi'an, China and Laboratory for Advanced Computing and Intelligence Engineering, Wuxi, China; e-mail: xiaolinh@nwpu.edu.cn; Yikun Zhang, Northwestern Polytechnical University, Xi'an, China; e-mail: yikunzhang@mail.nwpu.edu.cn; Chenhao Ma, The Chinese University of Hong Kong—Shenzhen, Shenzhen, China; e-mail: machenhao@cuhk.edu.cn; Xuequn Shang (corresponding author), Northwestern Polytechnical University, Xi'an, China; e-mail: shang@nwpu.edu.cn; Reynold Cheng, The University of Hong Kong, Hong Kong, Hong Kong; e-mail: ckcheng@cs.hku.hk; Tobias Grubenmann, Edinburgh Napier University, Edinburgh, United Kingdom of Great Britain and Northern Ireland; e-mail: tobias.grubenmann@gmail.com; Xiaodong Li, Xiamen University, Xiamen, China and Key Laboratory of Multimedia Trusted Perception and Efficient Computing, Ministry of Education of China, Xiamen, China; e-mail: xdli@xmu.edu.cn .

ACM 1556-472X/2025/4-ART77
https://doi.org/10.1145/3719013

incorporates recent and periodic dimensions of road traffic. We evaluate RegGC against 10 existing methods on 3 real road network datasets. They show that RegGC is more effective and efficient than state-of-the-art solutions.

CCS Concepts: • **Information systems** → **Location based services**;

Additional Key Words and Phrases: Spatial Data Mining, Stochastic Weight Completion, Hypergraphs, Sparsity

Associate Editor: Jingrui He

ACM Reference format:
Xiaolin Han, Yikun Zhang, Chenhao Ma, Xuequn Shang, Reynold Cheng, Tobias Grubenmann, and Xiaodong Li. 2025. Hypergraph-Enhanced Multi-Granularity Stochastic Weight Completion in Sparse Road Networks. *ACM Trans. Knowl. Discov. Data.* 19, 3, Article 77 (April 2025), 23 pages.
https://doi.org/10.1145/3719013

1 Introduction

A road network, or a graph with roads as edges and road junctions as nodes, enables Big Transportation Data applications (e.g., navigation [31, 48, 78], traffic prediction [8, 9, 20, 75], incident detection [24, 25], outlier detection [23, 36, 42], **Point-of-Interest (POI)** recommendation [15, 77, 84], and trajectory similarity search [57, 66, 72]). The road network can be treated as a graph with vertices and edges [43–46, 63, 64]. The edge weights of the road network (e.g., the time needed to traverse a road) are often extensively used by the above applications. A navigation app, for instance, returns a path to a user with the smallest sum of edge weights along the path.

However, it is common for a road network to have edges whose weights are missing [18, 54, 65, 68, 69]. For example, in a dataset that contains the GPS of taxis in Hong Kong during 2010, the taxi locations are concentrated in commercial and residential districts, covering only 1.4% of all roads in Hong Kong. And roads in rural areas are seldom visited by vehicles. During late nights, most roads are traversed by few vehicles, making it difficult to collect traffic data. To tackle this issue, a few **Weight Completion (WC)** methods have been proposed (e.g., [8, 27, 40, 58]).

In general, WC problems can be classified into two types: *deterministic* and *stochastic*. Most WC methods are targeted towards deterministic problems (e.g., [19, 40, 81]). They learn deterministic, or non-probabilistic, weights for edges (e.g., the average traveling time of Nathan Road in Hong Kong between 10 and 10:30 am on Sunday is 5 minutes on average). However, these methods do not perform well in stochastic settings [27]. In addition, these methods often ignore the sparsity of traffic data, which leads to inaccurate results. In contrast, very few methods are targeting the **Stochastic WC (SWC)** problem (e.g., A-GCWC [27]), and are optimized towards learning the *distributions* of edge weights (e.g., the traveling time of Nathan Road during Sunday 10–10:30 am is 5 minutes with 80% probability and 10 minutes with 20% probability). Such methods, which take into account the time-varying uncertainty of real-time traffic conditions, give more precise weights than their counterparts targeting only deterministic weights.

As shown in [1, 51–53, 74, 76], SWC improves the accuracy of path routing. To illustrate this, suppose that a person departs from her home to catch a football event in Figure 1. She needs to arrive at the stadium within 50 minutes. There are two possible paths: P_1 and P_2. If average time is used, P_2 should be chosen, since it requires a lower cost (29.75 minutes). However, if the time distributions are considered, then P_1 is the preferred choice, because there is a higher chance (95%) for P_1 to be completed within 50 minutes (compared to 70% for P_2). By using stochastic weights, the chance of finding a better route can be improved.

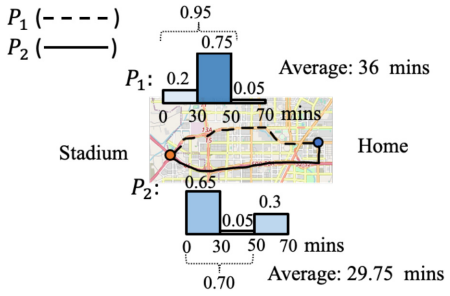

Fig. 1. An example of stochastic edge weights.

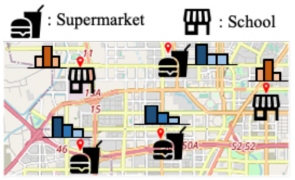

Fig. 2. An example of region-wise traffic correlation.

Despite the benefits of using stochastic weights, stochastic weights completion has not been well studied. The best solution for SWC so far is A-GCWC [27], which uses a graph **Convolutional Neural Network (CNN)** to propagate weights from edges whose traffic data is available to the edges with missing values. However, it does not perform well on very sparse traffic data (e.g., the Hong Kong dataset mentioned) in our experiments, i.e., an average 5.5% gap compared with our model in the Hong Kong dataset.

To improve SWC on sparse traffic data, we observe that traffic distributions among the same semantic regions follow close correlations among each other even though they are distantly located. Figure 2 shows the traffic correlation among regions at 6 pm on Sunday. As we can see, traffic distributions among the same semantic regions, e.g., supermarkets, exhibit similar behaviors. The reason is that most people usually go to supermarkets on Sunday afternoon, therefore the traffic conditions could become congested. On the other hand, people do not come to school on Sunday, so the traffic is relatively smooth. Therefore, we propose a region-wise hypergraph neural architecture to learn the coarse granularity of spatial correlations, in which we construct the hypergraph based on the semantic region dependencies.

For SWC, we utilize a *contextual graph*, which describes road information such as speed limits, number of lanes, and road types, to provide finer granularity of spatial correlations instead of coarse semantic region information discussed previously. Figure 3 illustrates a *contextual graph*. The intuition is that road properties depicted by a contextual graph provide informative contexts about road similarity among neighbors. As we will show, it is helpful especially when traffic data is sparse, because edge correlations captured from contextual graphs enhance the performance. In this way, multi-granularity spatial correlations can be captured behind coarse semantic regions and finer road properties.

A salient feature of our model is that it incorporates not just the spatial dimension of the traffic data, but also two aspects of the temporal dimension—recency and periodicity. The intuition is that recent and periodic traffic has a high correlation with the target time. By propagating traffic data along both space and time dimensions, more data could be provided to learn the stochastic weights

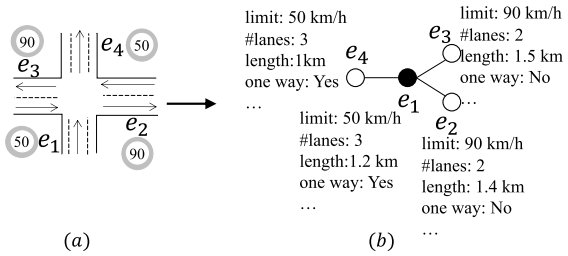

Fig. 3. (a) A road network; (b) a contextual graph.

for the edge concerned, which also alleviates the data sparsity problem. This work is an extension of our previous study [22].

To summarize, our contributions are:

—We propose a hypergraph neural architecture to capture the coarse granularity of spatial correlations based on semantic region functions.
—We present the graph neural network facilitated model, which collectively leverages contextual graphs with traffic dynamics to provide finer granularity of spatial correlations.
—Our model incorporates topological traffics, recent trends, and periodic patterns to effectively and efficiently complete stochastic weights in a multi-level way.
—We have performed substantial experiments on three real road network datasets against 10 existing methods. Our model is 21.96% and 4.22% more effective than deterministic WC methods and SWC methods, respectively. And it is efficient on million-scale road networks.

2 Related Work

There are two kinds of WC problems: deterministic and stochastic. The deterministic setting has been studied extensively. Most existing deterministic works target on predicting traffic based on past data. They use location data collected from static sensors installed along the freeways. Because these data are regularly obtained, they are "dense." In this way, all past data are available, and ignoring sparsity is fine. However, this assumption may not be valid when road sensors are not available. In such cases, we may still obtain vehicle location data by some other means (e.g., GPS); however, the traffic data so obtained are more sparse, making the WC more challenging.

Deterministic WC. Deterministic models focus on dense data. Traditional methods, e.g., Kriging [2, 14] and Markov Random Field [3], have been proposed to estimate missing observations in data. However, they are designed for deterministic values instead of stochastic ones. CNN-based models [81] have been proposed to predict traffic speed or crowd density. DSAE [13] utilizes denoising stacked autoencoders to fill in weights. DCRNN [40] has been proposed to forecast traffic data by capturing spatial-temporal correlations. Due to the success of Graph CNN [34] on many computer vision, natural language processing, and data mining tasks [35, 49, 60, 73], spectral-graph-theory-based GCN has been extensively introduced to predict traffic conditions. Spatial-temporal GCN [8, 12, 30, 55, 62] has been applied to forecast traffic data and citywide passenger demand by incorporating spatial-temporal correlation. Attention-based spatial-temporal **Graph Convolutional Networks (GCNs)** [19] are further proposed to pay different attention to nearby points in the graph. Most of these GCN-based deterministic traffic prediction models follow the spectral graph theory [7]. Recurrent neural networks [29, 79] are proposed to model the correlations among grid-based regions. DGSI [82] utilizes a deep geometric neural network for data estimation. Curb-**Generative Adversarial Network (GAN)** [83] proposes a GAN to estimate

the traffic data. Moreover, a hypergraph convolution method ST-SHN [71] is utilized to model the correlations among grid-based regions. However, the hypergraph is constructed directly from traffic data, which could lead to poorer performance when traffic values contain missing ones. Only fewer model, e.g., LSM-RN [11], considers the sparsity, which has been shown to perform worse than the existing SWC model in [27]. MBA-STNet [47] employs a discriminative multi-task learning framework coupled with a Bayesian neural network to predict traffic flow effectively. STUaNet [86] combines internal data quality estimation with external uncertainty modeling in an end-to-end approach to capture diverse contextual interactions for learning human mobility patterns. ODformer [28] uses a sparse self-attention mechanism to capture spatial correlations between origin-destination pairs and predict long-sequence origin-destination flows. MOHER [85] learns heterogeneous relationships for different modes in crowd flow prediction, using threshold values to address data sparsity issues. DMGC-GAN [32] integrates GAN with dynamic multi-GCNs to predict origin-destination ride-hailing demand, leveraging GAN's training mechanism to handle data sparsity [37]. PGCM [39] combines a GCN with a Bayesian approximation algorithm to predict public transit demand, mitigating sparsity in the origin-destination matrix using threshold values. Prob-GNN [59] models uncertainty in spatiotemporal travel demand using probabilistic graph neural networks. ST-GIN [67] imputes traffic data by integrating spatiotemporal graph attention with recurrent neural networks, while also considering uncertainty. DTIGNN [38] employs graph neural networks to iteratively impute missing traffic flows and predict them within a unified model. LSCGF [5] utilizes a multi-graph generation network to learn both sparse and continuous graph structures. Since most work ignores the sparsity in traffic data, the performance on sparse data is negatively affected. To overcome this, we reconstruct the real traffic data by an encoder-decoder framework from sparse traffic data in our proposed solutions.

SWC. A-GCWC [27] uses spectral-domain-based GCN to complete stochastic weights. It incorporates additional information, such as the time interval flag, by a Bayesian inference model. In terms of temporal dimension, the traffic behavior is assumed to be conditionally dependent on the time interval flag, which occurs periodically with a time period p. Given two time intervals T_1 and T_2 that are p time units apart, the traffic conditions of them are supposed to be the same in [27]. In fact, this is not always the case due to dynamic factors, e.g., road construction or incidents. Different from A-GCWC, our solution can capture dynamic changes, and learn the difference between T_1 and T_2. Our solution also considers recency and periodicity information, as well as contextual graph. Consequently, our methods perform better than A-GCWC, especially when traffic data are sparse.

The research on GCN has two branches, which are the spectral [10] and node domains [56]. Most traffic studies follow the spectral domain, in which GCN is applied to spectral graph theory. However, the model trained on one graph cannot be used for graphs with different structures since it is based on the Laplacian Eigenbasis [56]. Moreover, it requires high computation costs, e.g., matrix inversion. Fewer studies [40] follow the node domain. Although it has shown its benefits on many tasks [56], it has not been well studied in sparse traffic. Therefore, we follow the node domain to study the possibility of applying it to sparse traffic.

Graph Representation Learning. A classical GCN [34] was introduced to learn hidden node features using semi-supervised learning for classification tasks. GraphSage [21] extends this by sampling and aggregating the hidden features of neighboring nodes within an inductive framework. DAGNN [41] further adapts by incorporating messages from large receptive fields, enabling deeper graph neural networks. CGNN [70] generalizes existing graph neural networks from discrete dynamics to continuous ones.

In addition, graph attention networks have been developed to assign different levels of attention to neighboring nodes. GAT [56] introduces a novel masked self-attention mechanism to learn the

Table 1. Notations

Notation	Description
E	Edge set of the road network
$\mathcal{T}$	Set of the time intervals
W	The stochastic weight tensor
$\widehat{W}$	The completed stochastic weight tensor
$N^n_{\text{spat.}}(e)$	The spatial neighbors of edge e
$N^n_{\text{temp.}}(T)$	The temporal neighbors of time interval T
$N^{n,p}_{\text{period.}}(T)$	The periodic neighbors of time interval T
$h_{e_i,T}$	The stochastic weight vector of edge e_i at T
$\alpha^{\text{spat.}}_{e_i,e_j,T}$	The spatial traffic attention of edge e_j for e_i at T
$\alpha^{\text{temp.}}_{T_j,T_k,e_i}$	The temporal traffic attention of T_k for T_j of e_i
$\alpha^{\text{period.}}_{T_j,T_k,e_i}$	The periodic traffic attention of T_k for T_j of e_i
$\mathcal{G}$	The hypergraph
P	The incidence matrix of the hypergraph $\mathcal{G}$

importance of neighboring nodes and aggregate them based on these importance scores. HAN [61] applies node-level attention within each meta-path and then aggregates these representations using another attention mechanism across different meta-paths. HGRN [26] adaptively aggregates hidden features from multi-hop neighboring nodes using an attention mechanism to address the over-smoothing problem.

Moreover, hypergraph neural networks have been proposed to capture more complex relationships among nodes. HGNN [16] learns hidden features that account for high-order data structures to model complex node correlations. HGNN+ [17] extends this approach to a more general framework for learning multi-modal or multi-type data correlations in high-order structures. HSL [4] optimizes hypergraph neural networks together with hypergraph structure learning in an end-to-end manner. DHGNN [33] applies hypergraph convolution to dynamic graphs for capturing high-order data correlations. HGC-RNN [80] combines hypergraph convolution with recurrent neural networks to learn temporal dependencies.

3 Problem Definition

In this section, we first give definitions of the basic concepts. Then, we give a formal problem definition. The notations are in Table 1.

Definition 1 (Road Network). A road network is a graph where each vertex $v \in V$ represents a road intersection. Edge $e = (v_1, v_2) \in E \subseteq V \times V$ indicates that intersections v_1 and v_2 are directly connected.

Definition 2 (Stochastic Weight). Let $e \in E$ be a road, the stochastic weight of e at time interval T_i, $h_{e,T_i} \in \mathbb{R}^{|B|}$, is its travel cost distribution. Each stochastic weight consists of a set B of buckets which describe the histogram of the distribution.

In Definition 1, we capture each traveling direction of each road separately. However, we are not directly interested in the travelling direction but instead, we are only interested in whether the

road directions allow a vehicle to travel from one road to the next road. We are interested if two roads are *spatial neighbors*.

Definition 3 (Edge Graph). A directed road network can be transformed to an undirected edge graph $G = (E, A)$, in which A is adjacency matrix that captures the connectivity of the edges in E. $A_{e_i,e_j} = A_{e_j,e_i} = 1$, $e_i, e_j \in E$, if and only if a vehicle can travel from e_i to e_j or from e_j to e_i by passing through exactly one intersection $v \in V$, $A_{e_i,e_j} = A_{e_j,e_i} = 0$ otherwise.

Definition 4 (nth Order Spatial Neighbors). Given $n \in \mathbb{N}$, an edge graph G, two edges $e_i, e_j \in E$, the edges e_i and e_j are nth order spatial neighbors of each other if and only if a vehicle can travel from e_i to e_j or from e_j to e_i by passing through exactly n intersections. We denote with $N^n_{\text{spat.}}(e)$ the set of all spatial neighbors of e with order less or equal to n.

Definition 5 (nth Order Recent Neighbors). Given an interval $T_i \in \mathcal{T}$, the nth order recent neighbor for $n < i$ of T_i is the time interval T_{i-n}. We denote with $N^n_{\text{temp.}}(T)$ the set of all recent neighbors of time interval T with order less or equal to n.

Definition 6 (nth Order Periodic Neighbors). Given a period $p \in \mathbb{N}$ and a time interval $T_i \in \mathcal{T}$, the nth order periodic neighbor for $n \cdot p < i$ of T_i is the interval $T_{i-n\cdot p}$. We denote with $N^{n,p}_{\text{period.}}(T)$ the set of all periodic neighbors of T with order less or equal to n.

Problem Definition. Given $|\mathcal{T}|$ time intervals, $|E|$ roads, and bucket size $|B|$ for the stochastic weight tensor $W \in \mathbb{R}^{|E|\times|\mathcal{T}|\times|B|}$, we denote $h_{e,T} \in \mathbb{R}^{|B|}$ as the stochastic weight vector for the edge e at time interval T. Due to the sparsity issue, e.g., roads in rural areas are seldom visited by vehicles, W might have many missing values. The model takes W as input. The objective of SWC is to leverage a graph neural network to reconstruct a tensor $\widehat{W}$ that fills in the missing values in W within a road network. The reconstructed tensor $\widehat{W}$ is then produced as the output.

4 The Contextual Graph Completion (ConGC) Model

In this section, we first introduce the framework of ConGC (Section 4.1). After that, we explain the detailed steps in the model (Sections 4.2–4.8). Moreover, we show the time complexity of ConGC (Section 4.9).

4.1 Framework

Figure 4 shows the framework. It follows the encoder-decoder structure, in which Topological Traffic Propagator (Step ①), Contextual Traffic Diffusion (Step ②), Recent Trend Aggregator (Step ③), Periodic Pattern Explorer (Step ④), and Pooling form the encoder, and **Fully Connected (FC)** Layer forms the decoder. The idea is to encode the data by extracting key information from the transformation, then reconstruct the actual one by decoding it.

4.2 Topological Traffic Propagator

The spatial neighbors play an important role on the target edge.

The Topological Traffic Propagator learns the importance scores of the edge's informative spatial neighbors, and updates the edge by aggregating these neighbors. The intuition is that among the edge's spatial neighbors, there are some neighbors following a similar traffic condition as the target edge, but some may have different traffic conditions. For example, in Figure 4(a), road e_1 should pay more attention to road e_2 since vehicles can turn right when the red light is on. But road e_3 and e_4 have less correlation with road e_1 compared to road e_2. Inspired by attention mechanism [56] which

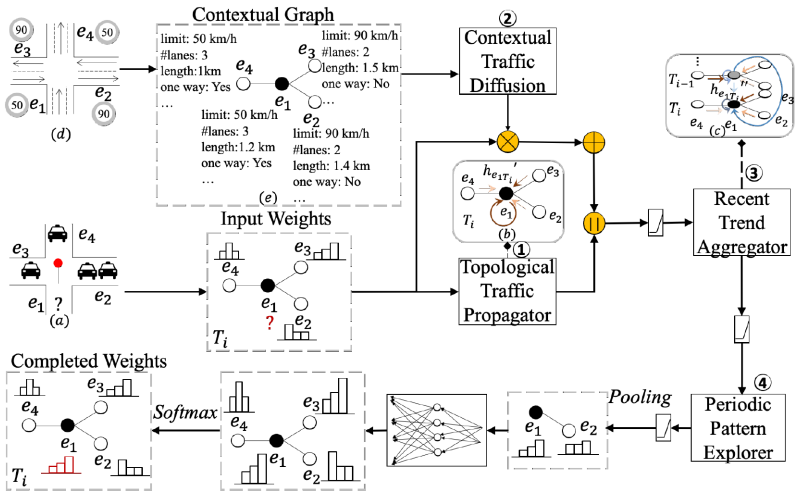

Fig. 4. The overall framework of the ConGC model. Details of each step are in Sections 4.2 to 4.7. ConGC, Contextual graph completion.

was originally proposed to learn attention for neighbors in the standard graph, we propose to exploit hidden correlations in sparse traffic data which contain complex information with missing data in neighbors.

The stochastic weight $h_{e_i,T} \in \mathbb{R}^{|B|}$ is transformed to latent embedding to allow deep expression:

$$H_{e_i,T} = U\, h_{e_i,T}\,, \tag{1}$$

where $U \in \mathbb{R}^{B' \times |B|}$ is the learnable parameter, and $H_{e_i,T} \in \mathbb{R}^{B'}$ is the latent representation of edge e_i at T.

The importance of edge e_j to edge e_i at the time interval T is

$$\eta^{\text{spat.}}_{e_i,e_j,T} = \text{ReLU}(a^{\text{tr}} \cdot [H_{e_i,T} || H_{e_j,T}])\,, \tag{2}$$

where ReLU is the activation function, a^{tr} is the transpose of the trainable parameter vector a, $||$ is the concatenation operator, $e_j \in N^n_{\text{spat.}}(e_i)$ is the at most nth order spatial neighbors of e_i.

The spatial traffic attention can be calculated as

$$\alpha^{\text{spat.}}_{e_i,e_j,T} = \frac{\exp(\eta^{\text{spat.}}_{e_i,e_j,T})}{\sum_{e_k \in \text{Mask}_T(N^n_{\text{spat.}}(e_i))} \exp(\eta^{\text{spat.}}_{e_i,e_k,T})}\,, \tag{3}$$

where $\text{Mask}_T(N^n_{\text{spat.}}(e_i)) = \{e_j \mid h_{e_j,T} \text{ is valid}, e_j \in N^n_{\text{spat.}}(e_i)\}$ is the masked set of spatial neighbors that only contain edges with traffic data at time interval T.

The spatial graph convolution is calculated as

$$h^{\text{spat.}}_{e_i,T} = \sum_{e_j \in \text{Mask}_T N^n_{\text{spat.}}(e_i)} \alpha^{\text{spat.}}_{e_i,e_j,T} \cdot H_{e_j,T}\,, \tag{4}$$

where $h^{\text{spat.}}_{e_i,T} \in \mathbb{R}^{B'}$ is the updated embedding of e_i at T after spatial attentional graph convolution.

4.3 Contextual Traffic Diffusion

Roads in the road network have properties, e.g., speed limit, the number of lanes, length, road type, and one-way flag. These properties in the road network form a contextual graph (Figure 4(e)).

The insight is that they can provide useful information about road similarity in nature, which is important especially under data sparsity scenarios.

The ConGC updates based on road properties in the contextual graph and traffic condition in the dynamic graph collectively. The road properties can be transformed as contextual embeddings. For categorical features, e.g., the number of lanes *nl*, one-way flag *ow*, road type *rt*, we apply one-hot encoding to transform them as f_{nl}, f_{ow}, and f_{rt}. For continuous features, e.g., speed limit *sl* and road length *rl*, we use binning strategy to transform them into discrete ones as f_{sl} and f_{rl}. Then we concatenate them as $f_c \in \mathbb{R}^{|E| \times nf}$:

$$f_c = f_{nl} \,||\, f_{ow} \,||\, f_{rt} \,||\, f_{sl} \,||\, f_{rl}\,, \tag{5}$$

where nf is the dimension of feature f_c of edge e_i.

We apply a transformation to obtain the latent feature of graph contexts as

$$F_{c_{e_i}} = R\, f_{c_{e_i}}\,, \tag{6}$$

where $f_{c_{e_i}} \in \mathbb{R}^{nf}$ is the contextual embedding of edge e_i, $R \in \mathbb{R}^{B' \times nf}$ is the learnable parameter, and $F_{c_{e_i}} \in \mathbb{R}^{B'}$ is the latent context representation of edge e_i.

The contextual importance of edge e_j to edge e_i is

$$\eta^{\text{cont.}}_{e_i,e_j} = \text{ReLU}(d^{\text{tr}} \cdot [F_{c_{e_i}} || F_{c_{e_j}}])\,, \tag{7}$$

where d is the trainable parameter vector. The contextual similarity score $\alpha^{\text{cont.}}_{e_i,e_j} \in \mathbb{R}$ can be calculated as

$$\alpha^{\text{cont.}}_{e_i,e_j} = \frac{\exp(\eta^{\text{cont.}}_{e_i,e_j})}{\sum_{e_k \in (N^n_{\text{spat.}}(e_i))} \exp(\eta^{\text{cont.}}_{e_i,e_k})}\,. \tag{8}$$

We then diffuse the transformed stochastic weight embedding $H_{e_j,T} \in \mathbb{R}^{B'}$ (Equation (1)) based on the contextual similarity score $\alpha^{\text{cont.}}_{e_i,e_j}$ as

$$h^{\text{cont.}}_{e_i,T} = \sum_{e_j \in N^n_{\text{spat.}}(e_i)} \alpha^{\text{cont.}}_{e_i,e_j} \cdot H_{e_j,T}\,. \tag{9}$$

The embedding $h'_{e_i,T} \in \mathbb{R}^{2B'Q}$ of edge e_i at the time interval T after contextual and spatial graph convolution is updated as

$$h'_{e_i,T} = \|_{q=1}^{Q} \text{ReLU}\left(h^{\text{cont.}}_{e_i,T} || h^{\text{spat.}}_{e_i,T}\right)\,. \tag{10}$$

To make the learning robust, we concatenate the learned embedding $Q \in \mathbb{N}$ times as the multi-head attention. The attention mechanism ensures that edges with significant differences or discrepancies receive lower importance scores, thereby minimizing their influence on the target edge.

4.4 Recent Trend Aggregator

Temporal neighbors of edge e_i have high correlations with the target time interval T_j of e_i.

Recent Trend Aggregator with a masked operator aims to calculate the importance scores of the edge's masked recent neighbors, then update the edge by aggregating its masked recent neighbors with importance scores. The intuition is that recent neighbors of different orders follow different degrees of traffic correlation with the target time interval T_i. For example, in Figure 5(a), road e_1 at T_3 should pay more attention to its second-order recent neighbor at T_1 since the traffic lights are red at both T_1 and T_3. In contrast, the first-order recent neighbor T_2 has less correlation with T_3 compared to T_1, since the traffic light is green at T_2.

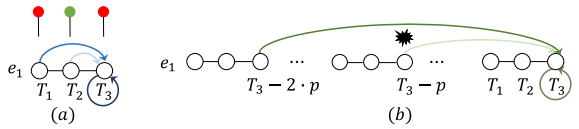

Fig. 5. (a) Recent neighbors; (b) periodic neighbors.

The $h'_{e_i,T_j} \in \mathbb{R}^{2B'Q}$ is mapped to latent representation $H'_{e_i,T_j} \in \mathbb{R}^{B'}$ to allow sufficient expression as

$$H'_{e_i,T_j} = O\, h'_{e_i,T_j}\,, \tag{11}$$

where $O \in \mathbb{R}^{B' \times 2B'Q}$ is the learnable parameter.

The importance of T_k for T_j at e_i is defined as

$$\eta^{\text{temp.}}_{T_j,T_k,e_i} = \text{ReLU}(p^{\text{tr}} \cdot [H'_{e_i,T_j} || H'_{e_i,T_k}])\,, \tag{12}$$

where p is the trainable parameter vector, the time interval $T_k \in N^n_{\text{temp.}}(T_j)$, and $j - n \le k \le j$, n is the maximum order of recent neighbors.

The recent attention coefficient is defined as

$$\alpha^{\text{temp.}}_{T_j,T_k,e_i} = \frac{\exp(\eta^{\text{temp.}}_{T_j,T_k,e_i})}{\sum_{T_l \in \text{Mask}_{e_i} N^n_{\text{temp.}}(T_j)} \exp(\eta^{\text{temp.}}_{T_j,T_l,e_i})}\,, \tag{13}$$

where $\text{Mask}_{e_i} N^n_{\text{temp.}}(T_j) = \{T_l \mid h_{e_i,T_l} \text{ is valid},\ T_l \in N^n_{\text{temp.}}(T_j)\}$ is the set of masked recent neighbors of T_j at e_i that filters out neighbors with missing traffic data.

The embedding $h''_{e_i,T_j} \in \mathbb{R}^{B'Q}$ is calculated as

$$h''_{e_i,T_j} = \|_{q=1}^{Q} \text{ReLU}\left(\sum_{T_k \in \text{Mask}_{e_i} N^n_{\text{temp.}}(T_j)} \alpha^{\text{temp.}}_{T_j,T_k,e_i} H'_{e_i,T_k} \right). \tag{14}$$

4.5 Periodic Pattern Explorer

The Periodic Pattern Explorer with masked neighbors learns importance scores for informative periodic neighbors of the target edge e_i at time interval T, then integrates these periodic neighbors. The intuition is that periodic neighbors usually follow a similar correlation as the target one. However, there may be some divergence among them. For example, in Figure 5(b), an incident happened at time interval $T_3 - p$ at edge e_1, where p is the period. Consequently, the correlation between T_3 and $T_3 - p$ is lower than that between T_3 and $T_3 - 2 \cdot p$. The period p could be 1 day or 1 week.

The $h''_{e_i,T_j} \in \mathbb{R}^{B'Q}$ is transformed to latent embedding by parameter $P \in \mathbb{R}^{B' \times B'Q}$ as

$$H''_{e_i,T_j} = P\, h''_{e_i,T_j}\,, \tag{15}$$

where $H''_{e_i,T_j} \in \mathbb{R}^{B'}$ is the latent representation.

The importance of temporal periodic neighbor T_k for time interval T_j of edge e_i is defined as

$$\eta^{\text{period.}}_{T_j,T_k,e_i} = \text{ReLU}(u^{\text{tr}} \cdot [H''_{e_i,T_j} || H''_{e_i,T_k}])\,, \tag{16}$$

where u is the parameter vector, the time interval $T_k \in N^{n,p}_{\text{period.}}(T_j)$, and $j - n \cdot p \le k \le j$, n is the maximum order of periodic neighbors, p is the period.

The periodic traffic attention can be calculated as

$$\alpha^{\text{period.}}_{T_j,T_k,e_i} = \frac{\exp(\eta^{\text{period.}}_{T_j,T_k,e_i})}{\sum_{T_l \in \text{Mask}_{e_i} N^{n,p}_{\text{period.}}(T_j)} \exp(\eta^{\text{period.}}_{T_j,T_l,e_i})}, \tag{17}$$

where $\text{Mask}_{e_i} N^{n,p}_{\text{period.}}(T_j) = \{T_l \mid h_{e_i,T_l} \text{ is valid}, \ T_l \in N^{n,p}_{\text{period.}}(T_j)\}$ is masked periodic neighbors of T_j at e_i that only contains periodic neighbors with traffic data.

The periodic graph convolution is calculated as

$$h^{\text{updated}}_{e_i,T_j} = \|_{q=1}^{Q} \ \text{ReLU}\left(\sum_{T_k \in \text{Mask}_{e_i} N^{n,p}_{\text{period.}}(T_j)} \alpha^{\text{period.}}_{T_j,T_k,e_i} H''_{e_i,T_k} \right), \tag{18}$$

where $h^{\text{updated}}_{e_i,T_j} \in \mathbb{R}^{B'Q}$ is the updated embedding after periodic graph convolution.

4.6 Pooling

The $h^{\text{updated}}_{e_i,T_j}$ of all edges at all time intervals form the tensor W^{updated}. Then, pooling is applied to extract key information from W^{updated} by a pooling size of k_t and k_e, where k_t is designed to extract temporal key information, and k_e is used for pooling important spatial information. Under traffic sparsity, there may still exist missing values in W^{updated} even after the operations we applied. Pooling is necessary since it only keeps key information from W^{updated} instead of missing values. After pooling, we obtain the encoded W^{PL} with a smaller size compared to W^{updated}.

Take the max Pooling on $W^{\text{updated}} \in \mathbb{R}^{|E| \times |\mathcal{T}| \times B'Q}$ in the spatial and temporal dimensions as an example:

$$W^{\text{PL}}_{e,t,bq} = \max_{i=k_e*e}^{k_e(e+1)-1} \ \max_{j=k_t*t}^{k_t(t+1)-1} W^{\text{updated}}_{i,j,bq}, \tag{19}$$

where $W^{\text{PL}} \in \mathbb{R}^{\frac{|E|}{k_e} \times \frac{|\mathcal{T}|}{k_t} \times B'Q}$ is the tensor after Pooling.

4.7 FC Layers

The FC Layers are applied to decode key information which is encoded in W^{PL}, and restore the tensor to the original shape $|E| \times |\mathcal{T}| \times |B|$. After that, softmax is applied to make sure the sum of each stochastic weight equals one. Finally, the completed stochastic weight tensor $\widehat{W}$ is obtained.

4.8 Optimization

After the encoder-decoder structure, we finally obtain the completed stochastic weight tensor $\widehat{W} \in \mathbb{R}^{|E| \times |\mathcal{T}| \times |B|}$. The goal is that $\widehat{W}$ is as close as the actual ground truth stochastic weight tensor W_G as possible, where $W_G \in \mathbb{R}^{|E| \times |\mathcal{T}| \times |B|}$ is set according to the label of model functionalities (see Section 6.1.3). Therefore, the loss function is formulated as

$$\mathcal{L}_1(\widehat{W}, W_G) = \sum_{i=1}^{|\mathcal{T}|} \sum_{j=1}^{|E|} \mathbb{1}_{T_i,e_j} \cdot \text{KL}(\widehat{h}_{e_j,T_i} || h_{e_j,Ti}), \tag{20}$$

where $\text{KL}(\cdot||\cdot)$ measures the KL-divergence between the completed stochastic weight and the ground truth, and $\mathbb{1}_{T_i,e_j}$ is an indicator function that is 0 if data are missing in T_i at e_j, and 1 otherwise. The reason is that we can only establish the quality of the completed stochastic weight if the actual one is available as ground truth.

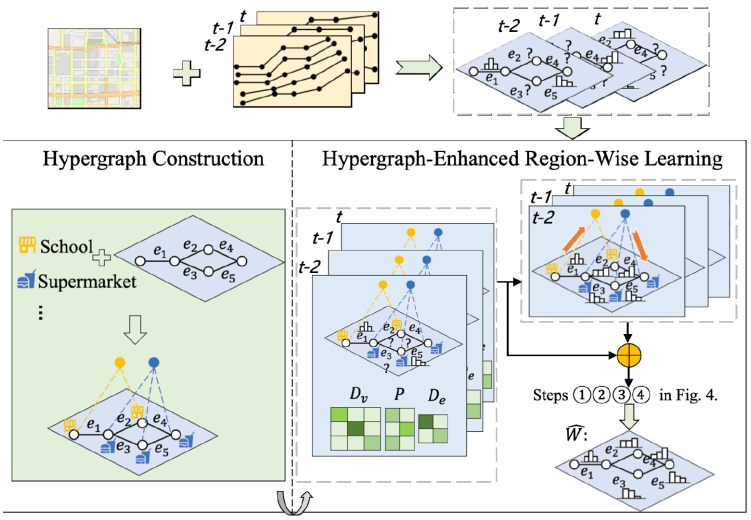

Fig. 6. The overall framework of the RegGC model. It follows steps in the framework of the ConGC model in Figure 4.

4.9 Complexity Analysis

The time complexity of ConGC is $O(Q \cdot |\mathcal{T}| \cdot |E| \cdot |B| \cdot B' \cdot (2 \cdot |N^{\max}_{\text{spat.}}| + |N^{n}_{\text{temp.}}| + |N^{n,p}_{\text{period.}}|))$, where $|N^{\max}_{\text{spat.}}|$, $|N^{n}_{\text{temp.}}|$, and $|N^{n,p}_{\text{period.}}|$ are the maximal number of spatial neighbors, the number of recent, and periodic neighbors, respectively. After removing constants, the complexity is dominated by $O(|\mathcal{T}| \cdot |E|)$. Hence, ConGC has a polynomial time complexity, and is efficient as shown in Section 6.

5 The Region-Wise Graph Completion (RegGC) Model

Since ConGC ignores the coarse spatial correlations among semantic regions, which is beneficial for propagating messages among distantly located roads, we further propose a general multi-granularity framework to enhance ConGC. In this section, we first introduce the framework of RegGC (Section 5.1). After that, we explain the detailed steps in the model (Sections 5.2 and 5.3). Then, we show the time complexity of RegGC (Section 5.4). Finally, we give a discussion to summarize the properties of the RegGC model (Section 5.5).

5.1 Framework

Figure 6 shows the framework of the RegGC model. It first constructs a hypergraph (Section 5.2) based on semantic region functions, e.g., POI information. Next, hypergraph-enhanced region-wise learning (Section 5.3) is utilized to propagate region-based traffic similarity among all roads.

5.2 Hypergraph Construction

Since roads near the same semantic regions exhibit similar traffic patterns, the first step is to organize the connections between roads and semantic regions.

Challenges. Some regions, e.g., rural areas, may serve few meaningful semantic functions; however, some regions, e.g., the center of one city, may provide multiple semantic functions, e.g., schools and supermarkets. If we split the city by grids strictly on the map, this may result in many regions with few meaningful semantic functions. Finding a way to define meaningful semantic regions

is the first challenge. Since stochastic weights contain missing values, especially for some roads that are seldom traversed by vehicles in the evening, traditional approaches [71], which learn the hypergraph connection from traffic data, cannot provide accurate connections. Finding a reasonable way to construct the hypergraph structure for sparse traffic data is the second challenge.

Design. To fix the first challenge, we propose to utilize the POIs for defining the semantic regions in a city. To address the second challenge, we propose to construct the hypergraph based on the semantic region function instead of constructing it from sparse traffic data as in [71]. The reason is that the semantic region function provides a reasonable way to define the traffic similarity among roads that are located near the same semantic regions, e.g., schools, hospitals, and supermarkets.

For example, even if three roads e_1, e_2, e_3 are located very far away from each other, as long as they are near the same semantic region, e.g., schools, they will be connected via one hyperedge since they could share the similar traffic patterns during peak hours. Specifically, we map all roads to their nearest semantic regions based on a distance threshold γ. Then roads e that are located near the regions r_m sharing the same semantic function are connected with one hyperedge m. Formally, the semantic region-based hypergraph is defined as $\mathcal{G} = (\mathcal{E}, \mathcal{M}, P)$, where $\mathcal{E}$ denotes the road set, $\mathcal{M}$ represent the hyperedge set, and P is the incidence matrix of the hypergraph $\mathcal{G}$. The incidence matrix P of the hypergraph $\mathcal{G}$ is a $|\mathcal{E}| \times |\mathcal{M}|$ matrix as

$$p(e, m) = \begin{cases} 1, & \text{if } d(e, r_m) < \gamma, \\ 0, & \text{if } d(e, r_m) \geq \gamma. \end{cases} \tag{21}$$

In this way, the hypergraph $\mathcal{G}$ is constructed to reflect similar traffic patterns for roads near the same semantic regions.

5.3 Hypergraph-Enhanced Region-Wise Learning

Hypergraph is utilized to capture the region-wise correlations among roads, even when they are located distantly with each other.

Challenges. Hypergraph convolution aims at propagating traffic information among roads belonging to the same hyperedge. However, as the stochastic weights contain missing values, learning becomes more difficult. The key challenge is enabling the model to effectively learn from the sparse traffic data.

Design. To address this challenge, we propose to learn the key information from the hypergraph convolution, in which we design a residual module to learn the key difference compared with the transformed stochastic weights.

Firstly, the observed stochastic weight $h_{e_i,T} \in \mathbb{R}^{|B|}$ is transformed as

$$J_{e_i,T} = N\, h_{e_i,T}\,, \tag{22}$$

where $N \in \mathbb{R}^{B' \times |B|}$ is the learnable parameter, B' is the dimension of hidden state, and $J_{e_i,T} \in \mathbb{R}^{B'}$ is the latent representation of edge e_i at T.

Next, the spectral convolutions on hypergraphs are defined as

$$g_\theta \star J = Q g_\theta Q^T J\,, \tag{23}$$

where Q is the eigenvector of the hypergraph Laplacian.

After applying Chebyshev polynomials $T_k(x)$ with the maximal of the eigenvalues $\lambda_{\max} \approx 2$, we have the transformed embedding as

$$J^H = D_v^{-1/2} P D_e^{-1} P^T D_v^{-1/2} J\Theta + J\,, \tag{24}$$

where P is the incidence matrix of the hypergraph $\mathcal{G}$, and D_v, D_e represent the diagonal matrices of the edge degrees and the vertex degrees of the incidence matrix P of the hypergraph $\mathcal{G}$, Θ is the trainable parameters. Here we focus on learning the residual of the hypergraph convolution so that the model can target on the key difference compared with the transformed stochastic weights.

After that, we apply Topological Traffic Propagator, Contextual Traffic Diffusion, Recent Trend Aggregator, and Periodic Pattern Explorer from Section 4 on J^H to capture the spatial-temporal correlations among traffic data.

Since the goal is to maximize the marginal log-likelihood of each observation in the stochastic weight, we introduce a latent variable $s \sim q(s|W)$ to encode the key information from the observed stochastic weight W. Note that the observed stochastic weight W might have many missing values; therefore we only learn the latent variable s from the cells that contain observed values in W.

Finally, the FC Layers are utilized to decode the complete stochastic weights from the encoder. The loss function is formulated as

$$\mathcal{L} = KL\left(q(s|W)||p(s)\right) + \mathcal{L}_1(\widehat{W}, W_G), \tag{25}$$

where $KL\left(q(s|W)||p(s)\right)$ is the KL-divergence between the posterior and prior distributions of the latent variable s. After back-propagation of the loss function, it will give the completed stochastic weight $\widehat{W}$.

5.4 Complexity Analysis

The time complexity of RegGC is $O(Q \cdot |\mathcal{T}| \cdot |E| \cdot |B| \cdot B' \cdot (2 \cdot |N^{\max}_{\text{spat.}}| + |N^{n}_{\text{temp.}}| + |N^{n,p}_{\text{period.}}|) + p_{nz} \cdot |E|)$, where $|N^{\max}_{\text{spat.}}|$, $|N^{n}_{\text{temp.}}|$, and $|N^{n,p}_{\text{period.}}|$ are the maximal number of spatial neighbors, the number of recent, and periodic neighbors, respectively, p_{nz} is the number of non-zero values in the sparse matrix P. After removing constants, the complexity is dominated by $O(|\mathcal{T}| \cdot |E|)$. Hence, RegGC has a polynomial time complexity and is efficient as shown in Section 6.

5.5 Discussion

We observe the following about the RegGC model:

- The RegGC model is *efficient* due to its polynomial time complexity.
- The RegGC model is *scalable* since its complexity is linear along with the increasing data size. It is also confirmed with experimental results in Figure 9 of Section 6.
- The ConGC model is essentially an adaptation of the RegGC model, designed to perform well in situations where semantic region information is unavailable or limited. It can be viewed as a more lightweight version of RegGC. The ConGC model is faster than RegGC, as the complexity of ConGC is roughly $O(p_{nz} \cdot |E|)$ smaller than that of RegGC. If computational resources are limited, or semantic region information is not available, ConGC can be used alternatively.

6 Experiments

This section aims to evaluate the following **Research Questions (RQs)** through extensive experiments:

- RQ1: How does the proposed RegGC model perform compared with SOTA algorithms on different datasets?
- RQ2: How do different modeling designs affect the result of RegGC?
- RQ3: How does the RegGC model perform when enlarging data size?
- RQ4: How do different hyper-parameters affect the performance of the RegGC model?

6.1 Experimental Setup

6.1.1 Datasets. We evaluate on three real datasets.

- The HK is a taxi GPS dataset[1] in Hong Kong in 2010. It contains 35 gigabytes of trajectories, which involves 155,589 nodes.
- The XN is an open GPS dataset[2] in Xi'an in 2016, which contains 137 gigabytes of trajectories, and 10,910 nodes.
- The CD is an open GPS dataset,[2] which contains 196 gigabytes of GPS data in Chengdu in 2016, and has 9,583 nodes.

We set the histogram with eight 5-m/s buckets ranging from 0 m/s to 40 m/s, and partition a day into 96 15-minute intervals as in [27].

For the RegGC model, we utilize the attribute POI that existed in the OpenStreetMap[3] to represent the semantic region function. Based on that, hypergraphs in these three datasets are constructed (see Section 5.2). Besides, five attributes from the OpenStreetMap, i.e., the number of lanes *nl*, one-way flag *ow*, road type *rt*, speed limit *sl*, and road length *rl*, are utilized for the ConGC model.

6.1.2 Pre-Processing. After map matching [50], the two pre-processing steps are conducted.

Edge Graph Transformation. We choose the largest connected subgraph as in [27]. In the HK, XN, and CD datasets, the numbers of selected edges are 1,158, 64, and 175, respectively. We then transform the directed road network into an undirected edge graph.

Input Data Preparation. We construct ground truth stochastic weight W_G from the available GPS data. In particular, we partition the day into 96 15-minute intervals and create a histogram for each edge with eight 5 m/s buckets ranging from 0 m/s to 40 m/s for a given time interval, as in [27]. We only instantiate weights for edges that have at least five speed records, following the setting in [27]. Since W_G may contain missing values due to sparsity, we use only the available data from W_G as the ground truth. For input weight W, we construct it by randomly removing edge weights in W_G with removal ratio *rm*. Then we evaluate the quality of completed $\widehat{W}$ by comparing with W_G.

6.1.3 Model Functionalities. Our method is flexible to support two model functionalities.

Estimation. The input is stochastic weight $W@T_i$ at T_i with missing values. The output is the completed $\widehat{W}@T_i$ at T_i. The label is the ground truth $W_G@T_i$.

Prediction. The input is $W@T_i$ at T_i with missing values. The output is the predicted $\widehat{W}@T_{i+1}$ at the next T_{i+1}. The label is the ground truth $W_G@T_{i+1}$.

6.1.4 Competitors. We compare with 10 methods.

Deterministic WC Methods. There are two kinds of deterministic methods. The first kind contains **Random Forest (RF)**, *CNN*, and *DSAE* [13], which only consider *spatial data*. Another kind incorporates *temporal data*. They are *DCRNN* [40], *ASTGCN* [19], *ST-ResNet* [81], and *ST-SHN* [71]. We also compare our model with advanced methods designed to address the sparsity issue, including *DTIGNN* [38] and *LSCGF* [5]. We complete weights of each bucket in the histogram separately for RF as [27] for stochastic setting. For other learning-based baselines that were not originally designed for WC tasks, we modified their output size from one to $|B|$. The differences between ASTGCN and ours are threefold. First, they are designed on dense data. They ignore missing values in the

[1]The HK dataset is confidential, which is provided by Prof. S. C. Wong of Civil Engineering in HKU.
[2]http://outreach.didichuxing.com. You can submit an application to Didi Chuxing for data access.
[3]https://www.openstreetmap.org/

model, which negatively affects performance. Second, they are designed for deterministic weights. Third, they follow the spectral GCN. We follow the node domain, which enables propagation of correlations based on structure. The first two differences also hold for DCRNN. Since ST-ResNet is a grid-based method, we map the grid results to the roads that overlap with them, and average the mapped results for each road if there are multiple grids overlapping with them.

SWC Methods. A-GCWC [27] is the state-of-the-art model (GCN) for SWC. We denote *GC* as the basic version of ConGC that does not involve contexts.

6.1.5 Hyper-Parameter Tuning. We partition datasets into five folds as [27], where four folds are used for training and validation, and one fold for testing. We run 10 times in total, and report the average. We conduct hyper-parameter tuning by Bayesian optimizer. The scopes are learning rate [0.0001, 0.1], number of spatial, recent, periodic neighbors {2, 3, 4, 5}, p {"1 day," "1 week"}, Q {4, 8, 16, 32}, k_t, k_e {2, 4, 8}, kernel number {8, 16, 32}, kernel size {8, 16, 32}, dimension of hidden state B' {8, 16, 64, 128}, and distance threshold γ {50, 100, 200, 300} meters.

6.1.6 Performance Metrics. We evaluate by **Mean Kullback-Leibler Divergence Ratio (MKLR)** and **Fraction of Likelihood Ratio (FLR)** as [27]. **Historical Average (HA)** is the average of training data. The smaller the MKLR is, the better the quality is:

$$MKLR = \frac{\sum_{i=1}^{|\mathcal{T}|}\sum_{j=1}^{|E|}\mathbb{1}_{T_i,e_j}\cdot \mathrm{KL}(h^G_{e_j,T_i}||\widehat{h}_{e_j,T_i})}{\sum_{i=1}^{|\mathcal{T}|}\sum_{j=1}^{|E|}\mathbb{1}_{T_i,e_j}\cdot \mathrm{KL}(h^G_{e_j,T_i}||\mathrm{HA}_{e_j})}, \tag{26}$$

$$FLR = \frac{\sum_{i=1}^{|\mathcal{T}|}\sum_{j=1}^{|E|}\mathbb{1}_{T_i,e_j}|LR_{e_j,T_i} > 1|}{\sum_{i=1}^{|\mathcal{T}|}\sum_{j=1}^{|E|}\mathbb{1}_{T_i,e_j}}, \tag{27}$$

where $LR_{e_j,T_i} = \frac{\prod_{k=1}^{|o|}(P_{\widehat{h}}(o_k))}{\prod_{k=1}^{|o|}(P_{\mathrm{HA}}(o_k))}$, $|o|$ is the total number of ground truth records, $P_{\widehat{h}}(o_k)$ and $P_{\mathrm{HA}}(o_k)$ are the probabilities of observing o_k from $\widehat{h}$ and HA. The higher the FLR value is, the better the method is.

6.2 Effectiveness Evaluation (RQ1)

We set *rm* as 0.5–0.8 for XN and CD datasets as [27]. In HK, we do not remove any data and only evaluate *prediction* as the sparsity is already 90% ("None" for *rm* in Tables 2 and 3).

6.2.1 Estimation. In Tables 2 and 3, the MKLR values on the *estimation* task in the XN and CD datasets increase—recall that a low MKLR value is better—as *rm* increases. The reason is that when more edges are removed, less information can be used when propagating the correlation among edges and time intervals. And the FLR values in the XN and CD datasets decrease—recall that a high FLR value is better—as *rm* increases. The reason is the same as for MKLR since fewer data provide less information. RegGC achieves the best performance on all datasets. For example, its average improvements of MKLR and FLR values over advanced model A-GCWC on *estimation* are 7.75% and 9.32%. And its average improvements of MKLR and FLR values over the ConGC model on *estimation* are 1.50% and 2.24%. Moreover, RegGC has better performance compared with ST-SHN [71], e.g., 3.31% improvements on *estimation* task on average. The reason is that ST-SHN [71] constructs the hypergraph in a data-driven way, which cannot achieve the optimal results with missing values. Furthermore, RegGC outperforms advanced methods designed to address the sparsity issue, such as DTIGNN [38] and LSCGF [5], particularly when the removal ratio (*rm*) is

Table 2. MKLR (*Lower Is Better*) on Three Datasets

Datasets	Functionalities	*rm*	RF	CNN	DSAE	DCRNN	ASTGCN	ST-ResNet	A-GCWC	GC	ConGC	ST-SHN	LSCGF	DTIGNN	RegGC
XN	Estimation	0.5	0.91	0.49	0.37	1.07	0.47	0.72	0.22	0.21	0.14	0.16	0.23	0.25	**0.13**
		0.6	1.00	0.54	0.58	1.29	0.57	0.76	0.27	0.24	**0.23**	0.29	0.29	0.36	**0.23**
		0.7	0.95	0.54	1.13	1.16	0.52	0.60	0.39	0.37	0.26	0.28	0.41	0.37	**0.25**
		0.8	0.95	0.60	1.93	1.16	0.61	0.65	0.59	0.59	0.59	0.58	0.61	0.6	**0.57**
	Prediction	0.5	0.91	0.71	1.17	1.02	0.74	0.74	0.66	0.65	0.65	0.68	0.76	0.75	**0.64**
		0.6	1.04	0.71	1.30	1.25	0.74	0.73	0.66	0.65	0.65	0.68	0.75	0.75	**0.64**
		0.7	0.94	0.71	1.42	1.10	0.71	0.72	0.69	0.63	0.62	0.71	0.73	0.72	**0.61**
		0.8	0.95	0.71	1.63	1.07	0.70	0.75	0.69	0.62	**0.61**	0.62	0.76	0.73	**0.61**
CD	Estimation	0.5	0.98	0.70	0.74	0.87	0.77	0.62	0.53	0.44	0.40	0.39	0.55	0.49	**0.38**
		0.6	0.91	0.73	0.94	0.88	0.68	0.64	0.56	0.57	0.54	0.53	0.57	0.61	**0.52**
		0.7	0.99	0.77	1.19	0.89	0.70	0.64	0.72	0.65	0.64	0.68	0.73	0.66	**0.62**
		0.8	0.92	0.80	1.36	0.94	0.79	0.77	0.79	0.80	0.77	0.86	0.81	0.78	**0.74**
	Prediction	0.5	0.99	0.74	1.12	0.90	0.71	0.69	0.72	0.69	**0.68**	**0.68**	0.73	0.72	**0.68**
		0.6	0.91	0.74	1.18	0.91	0.71	0.70	0.73	0.70	0.69	0.69	0.74	0.72	**0.68**
		0.7	0.99	0.75	1.24	0.91	0.72	0.70	0.75	0.70	**0.69**	**0.69**	0.77	0.73	**0.69**
		0.8	0.93	0.76	1.30	0.92	0.72	0.70	0.76	0.71	**0.70**	**0.70**	0.78	0.75	**0.70**
HK	Prediction	None	0.92	0.82	1.14	1.27	0.88	0.77	0.80	0.80	0.74	0.75	0.83	0.81	**0.72**

For each method, we report average results over 10 runs.
MKLR, Mean Kullback-Leibler Divergence Ratio. The best values are highlighted in bold.

Table 3. FLR (*Higher Is Better*) on Three Datasets

Datasets	Functionalities	*rm*	RF	CNN	DSAE	DCRNN	ASTGCN	ST-ResNet	A-GCWC	GC	ConGC	ST-SHN	LSCGF	DTIGNN	RegGC
XN	Estimation	0.5	0.27	0.53	0.77	0.48	0.53	0.45	0.76	0.80	0.90	0.87	0.75	0.86	**0.92**
		0.6	0.31	0.53	0.74	0.40	0.51	0.43	0.76	0.78	0.82	0.81	0.74	0.8	**0.84**
		0.7	0.25	0.54	0.64	0.44	0.54	0.47	0.71	0.76	0.82	0.82	0.71	0.79	**0.83**
		0.8	0.21	0.49	0.47	0.43	0.50	0.46	0.47	0.52	0.54	0.54	0.46	0.49	**0.55**
	Prediction	0.5	0.27	0.43	0.42	0.50	0.41	0.42	0.44	0.51	0.52	0.46	0.42	0.51	**0.53**
		0.6	0.30	0.43	0.40	0.42	0.41	0.44	0.44	0.51	0.52	0.48	0.41	0.51	**0.53**
		0.7	0.25	0.40	0.38	0.44	0.39	0.41	0.36	0.49	0.50	0.44	0.34	0.5	**0.51**
		0.8	0.22	0.40	0.34	0.45	0.39	0.37	0.36	0.49	**0.50**	0.48	0.33	0.49	**0.50**
CD	Estimation	0.5	0.17	0.47	0.61	0.56	0.48	0.53	0.60	0.62	0.67	0.68	0.59	0.67	**0.71**
		0.6	0.15	0.46	0.52	0.56	0.50	0.52	0.57	0.56	0.60	0.60	0.55	0.62	**0.63**
		0.7	0.12	0.44	0.43	0.40	0.49	0.50	0.47	0.51	0.51	0.53	0.46	0.49	**0.54**
		0.8	0.09	0.42	0.35	0.36	0.44	0.47	0.43	0.46	0.48	0.45	0.41	0.45	**0.50**
	Prediction	0.5	0.17	0.44	0.40	0.51	0.49	0.54	0.49	0.54	0.55	0.55	0.47	0.52	**0.56**
		0.6	0.14	0.44	0.36	0.50	0.49	0.54	0.48	0.54	0.55	0.55	0.46	0.52	**0.56**
		0.7	0.12	0.43	0.33	0.49	0.48	0.54	0.47	0.54	0.54	**0.55**	0.44	0.51	**0.55**
		0.8	0.08	0.42	0.31	0.50	0.48	0.53	0.46	0.52	0.53	**0.54**	0.42	0.51	**0.54**
HK	Prediction	None	0.19	0.37	0.44	0.37	0.30	0.43	0.43	0.43	0.45	0.44	0.39	0.42	**0.46**

For each method, we report average results over 10 runs. The best values are highlighted in bold.

high. This demonstrates RegGC's ability to transfer available traffic data to roads with missing values through its multi-granularity spatial-temporal correlation learning approach.

6.2.2 Prediction. In Tables 2 and 3, RegGC beats other methods as well on the *prediction* task. The average improvements of MKLR and FLR values over advanced model A-GCWC are 5.00% and 9.67%. And its improvement of FLR values over the ConGC model is 0.71%.

Moreover, RegGC is clearly better than ConGC when *rm* is large. For the largest *rm* 0.8 on XN and CD datasets, the average improvement of RegGC is 1.13% more accurate than ConGC. As for HK, its average improvement is 1.5% more accurate than ConGC.

6.3 Ablation Study (RQ2)

We conduct an ablation study on RegGC to evaluate the effectiveness of its main steps. One basic variant ignores coarser spatial correlations among semantic regions modeled by the hypergraph (Section 5), which is denoted as *ConGC*. Another variant assigns equal weights for neighbors without an attention mechanism. We call this variant *RegGC_na (no attention)*. To show the effect

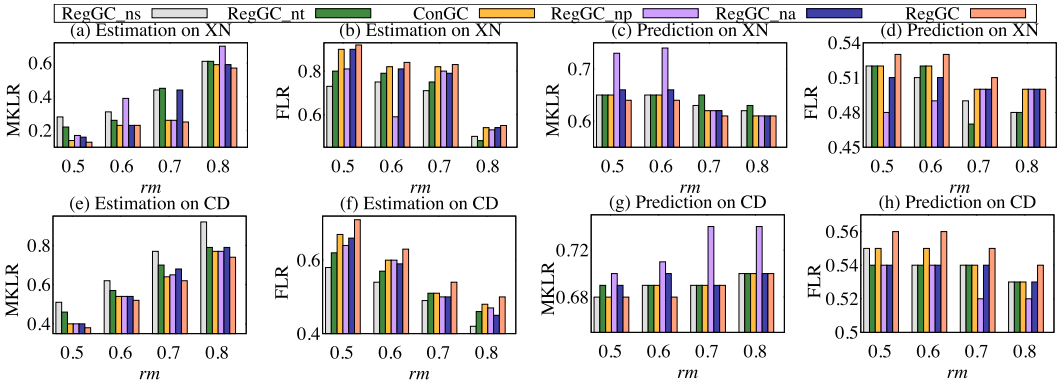

Fig. 7. The ablation study results on XN and CD w.r.t. MKLR (lower is better) and FLR (higher is better).

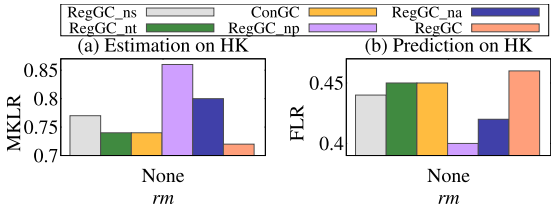

Fig. 8. The ablation study results on HK w.r.t. MKLR (lower is better) and FLR (higher is better).

of finer spatial correlations (Sections 4.2 and 4.3), we train the model without finer spatial correlations. We call this variant *RegGC_ns (no spatial)*. Moreover, we remove recent trend aggregator (Section 4.4) to evaluate the effect of recent trend aggregator. We call this variant *RegGC_nt (no temporal)*. Finally, we consider a variant of RegGC that does not incorporate periodic pattern explorer (Section 4.5). We call it *RegGC_np (no periodic)*.

We compare RegGC with five variants on three datasets w.r.t. MKLR and FLR metrics in Figures 7 and 8. Similar to Section 6.2, as the sparsity in HK dataset is very high, we do not remove any data from the HK dataset and only evaluate the *prediction* task. This is indicated by the word "None" for *rm* in Figure 8. From these figures, we make the following observations:

(1) RegGC beats ConGC on three datasets. In particular, the performance gap becomes larger under high *rm* values, e.g., 0.7 and 0.8, on estimation task for MKLR values. It shows that using coarser spatial correlations among semantic regions is important for information propagation in the graph.
(2) RegGC achieves better performance than RegGC_na, especially on the HK dataset, and *estimation* of XN and CD datasets. It confirms the importance of neighbors by applying the attention mechanism.
(3) RegGC performs significantly better than RegGC_ns on three datasets. In particular, the performance gap on the *estimation* task is the largest gap. It means that topological traffic propagator is essential in learning stochastic weights.
(4) RegGC outperforms RegGC_nt on three datasets. This shows that recent trend aggregator is very important for useful temporal information aggregator.
(5) RegGC works better than RegGC_np on three datasets. Specifically, the performance gap on the *prediction* task is very significant. It confirms that periodic pattern explorer is important in learning stochastic weights.

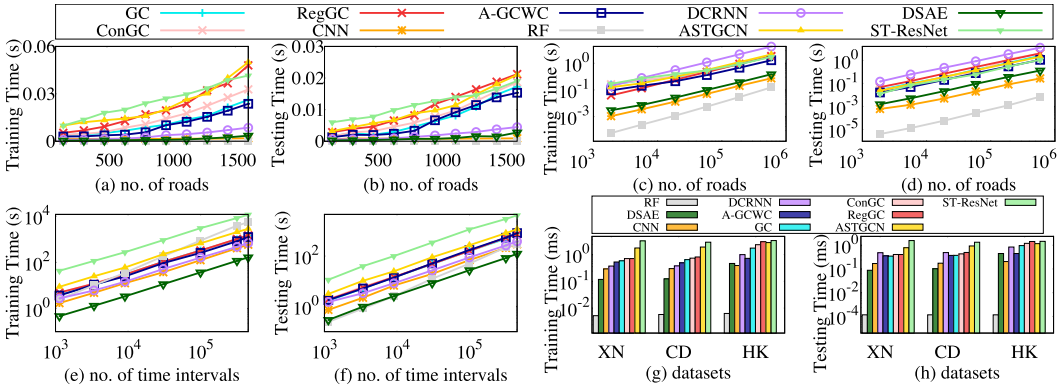

Fig. 9. Scalability on (a) and (b) moderate; (c) and (d) large road networks; (e) and (f) time dimension; (g) and (h) efficiency.

6.4 Efficiency Evaluation (RQ3)

We use GeForce GTX 1080 Ti 11 GB GPU for evaluation.

6.4.1 Efficiency Comparison. Figure 9(g) and (h) shows the average training and testing time for a single instance (i.e., a weight matrix at one time interval for all edges). Firstly, RF, DSAE, and CNN are faster than others since they only consider *spatial* data. Second, DCRNN, A-GCWC, GC, ConGC (one basic variant ignores coarser region semantics), and RegGC have comparable performance, while ASTGCN, ST-ResNet are much slower. Note that ST-ResNet, ASTGCN, GC, ConGC, and RegGC consider more data, i.e., *spatial*, *temporal*, and *periodic* data, while DCRNN only considers *spatial* and *temporal* data. Besides, RegGC utilizes multi-level *spatial* data, i.e., coarser region semantics and finer road properties, while other advanced models only consider single-level *spatial* data, e.g., road properties. It means that RegGC has comparable performance even considering more data than other advanced methods.

6.4.2 Scalability w.r.t. the Number of Roads. We manually enlarge the dataset as [27], since large road networks with dense data are unavailable. The maximum number of edges that one GPU can process with a batch size 8 is 1,600 for RegGC. We manually enlarge the road network of XN to 1,536 for Figure 9(a) and (b), and enlarge it to 1 million for Figure 9(c) and (d), and measure the average running time for an instance (i.e., an instance denotes a weight matrix at one time interval for all edges). We follow the two settings in [27] and evaluate (1) the scalability of moderate road networks that fit into one GPU (Figure 9(a) and (b)) and (2) the scalability of very large road networks which have to be partitioned into multiple small road networks that can be trained in sequence by batches in one GPU (Figure 9(c) and (d)). We adapt the learning-based methods using the advanced partitioning-based approach [6] to handle very large road networks feasible. Specifically, the large road network is partitioned into multiple parts, which are then processed in batches to fit within the GPU memory. Figure 9(a) and (b) shows that RegGC is scalable on moderate road networks, and Figure 9(c) and (d) shows that RegGC is scalable on vast road networks.

6.4.3 Scalability w.r.t. the Number of Time Intervals. Similarly, we manually enlarge the dataset w.r.t. $|\mathcal{T}|$ from 1,131 to 452,400, and measure the average training time for 1 epoch and the testing time for all testing data. Here, $|\mathcal{T}| = 452,400$ time intervals represent 12.9 years of data with 15-minute intervals. The number of edges is set to 64. Figure 9(e) and (f) shows that the RegGC is scalable on the time dimension.

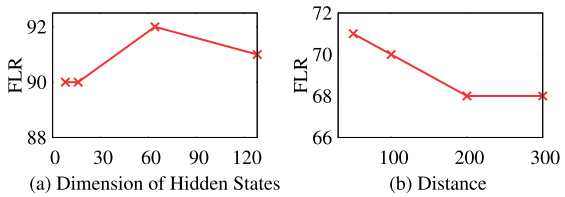

Fig. 10. Hyper-parameter study of RegGC.

6.5 Hyper-Parameter Study (RQ4)

We study the sensitivity of the hyper-parameters in our proposed model. We show the results about the sensitivity trend of the dimension of hidden state B' on the CD dataset with FLR values, and the sensitivity trend of distance γ on the XN dataset with FLR values. Note that other settings have similar sensitivity trends as Figure 10, we omit them due to the space limit.

Dimension of Hidden State B'. The performance is evaluated by varying B' from {8, 16, 64, 128}. The performance increases when $B' < 64$, and it achieves the best when $B' = 64$, then the performance degrades due to the overfitting issue. It shows that the performance becomes better when we enlarge B' until the best setting, then the performance becomes worse when B' increases further due to overfitting.

Distance Threshold γ. We evaluate performance by varying γ as 50, 100, 200, 300 m. Results show best performance at $\gamma = 50$ m, with degradation as γ increases. This is because roads closely located around the same POIs should exhibit similar traffic dynamics. However, as the road distance to the same POIs grows, traffic similarity between roads diminishes.

7 Conclusions

We study the SWC problem under data sparsity. We propose RegGC to utilize coarser semantic region functions and finer road properties for learning weights in a multi-granularity way. The model propagates correlations in both spatial and temporal dimensions from edges with weights to edges with missing weights. We give a formal definition of our problem setting and evaluate the effectiveness of the model compared with other state-of-the-art models. Our evaluation results show that RegGC is more effective, than other competitors, and can scale to large road networks.

References

[1] Georgi Andonov and Bin Yang. 2018. Stochastic shortest path finding in path-centric uncertain road networks. In *MDM*, 40–45.

[2] Bumjoon Bae, Hyun Kim, Hyeonsup Lim, Yuandong Liu, Lee D. Han, and Phillip B. Freeze. 2018. Missing data imputation for traffic flow speed using spatio-temporal Cokriging. *Transportation Research Part C: Emerging Technologies* 88 (2018), 124–139.

[3] Juliette Blanchet and Matthieu Vignes. 2009. A model-based approach to gene clustering with missing observation reconstruction in a Markov random field framework. *Journal of Computational Biology* 16, 3 (2009), 475–486.

[4] Derun Cai, Moxian Song, Chenxi Sun, Baofeng Zhang, Shenda Hong, and Hongyan Li. 2022. Hypergraph structure learning for hypergraph neural networks. In *IJCAI*, 1923–1929.

[5] Weijun Chen, Yanze Wang, Chengshuo Du, Zhenglong Jia, Feng Liu, and Ran Chen. 2023. Balanced spatial-temporal graph structure learning for multivariate time series forecasting: A trade-off between efficiency and flexibility. In *ACML*. PMLR, 185–200.

[6] Wei-Lin Chiang, Xuanqing Liu, SiSi, Yang Li, Samy Bengio, and Cho-Jui Hsieh. 2019. Cluster-GCN: An efficient algorithm for training deep and large graph convolutional networks. In *SIGKDD*, 257–266.

[7] Fan R. K. Chung and Fan Chung Graham. 1997. *Spectral Graph Theory*. American Mathematical Society.

[8] Rui Dai, Shenkun Xu, Qian Gu, Chenguang Ji, and Kaikui Liu. 2020. Hybrid spatio-temporal graph convolutional network: Improving traffic prediction with navigation data. In *SIGKDD*, 3074–3082.
[9] Shaojie Dai, Jinshuai Wang, Chao Huang, Yanwei Yu, and Junyu Dong. 2023. Dynamic multi-view graph neural networks for citywide traffic inference. *ACM Transactions on Knowledge Discovery from Data* 17, 4 (2023), 1–22.
[10] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. In *NIPS*, 29.
[11] Dingxiong Deng, Cyrus Shahabi, Ugur Demiryurek, Linhong Zhu, Rose Yu, and Yans Liu. 2016. Latent space model for road networks to predict time-varying traffic. In *SIGKDD*, 1525–1534.
[12] Zulong Diao, Xin Wang, Dafang Zhang, Yingru Liu, Kun Xie, and Shaoyao He. 2019. Dynamic spatial-temporal graph convolutional neural networks for traffic forecasting. In *AAAI*, Article 110, 890–897.
[13] Yanjie Duan, Yisheng Lv, Yu-Liang Liu, and Fei-Yue Wang. 2016. An efficient realization of deep learning for traffic data imputation. *Transportation Research Part C: Emerging Technologies* 72 (2016), 168–181.
[14] Paul Daniel Dumitru, Marin Plopeanu, and Dragos Badea. 2013. Comparative study regarding the methods of interpolation. *Recent Advances in Geodesy and Geomatics Engineering* 1 (2013), 45–52.
[15] Shanshan Feng, Xutao Li, Yifeng Zeng, Gao Cong, Yeow Meng Chee, and Quan Yuan. 2015. Personalized ranking metric embedding for next new POI recommendation. In *IJCAI*, 2069–2075.
[16] Yifan Feng, Haoxuan You, Zizhao Zhang, Rongrong Ji, and Yue Gao. 2019. Hypergraph neural networks. In *AAAI*, Vol. 33, 3558–3565.
[17] Yue Gao, Yifan Feng, Shuyi Ji, and Rongrong Ji. 2022. HGNN+: General hypergraph neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45, 3 (2022), 3181–3199.
[18] Chenjuan Guo, Bin Yang, Jilin Hu, and Christian Jensen. 2018. Learning to route with sparse trajectory sets. In *ICDE*, 1073–1084.
[19] Shengnan Guo, Youfang Lin, Ning Feng, Chao Song, and Huaiyu Wan. 2019. Attention based spatial-temporal graph convolutional networks for traffic flow forecasting. In *AAAI*, Article 114, 922–929.
[20] Shengnan Guo, Youfang Lin, Huaiyu Wan, Xiucheng Li, and Gao Cong. 2021. Learning dynamics and heterogeneity of spatial-temporal graph data for traffic forecasting. *IEEE Transactions on Knowledge and Data Engineering* (2021) 34, 11 (2021), 5415–5428.
[21] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *NIPS*, 30.
[22] Xiaolin Han, Reynold Cheng, Tobias Grubenmann, Silviu Maniu, Chenhao Ma, and Xiaodong Li. 2022. Leveraging contextual graphs for stochastic weight completion in sparse road networks. In *SDM. SIAM*, 64–72.
[23] Xiaolin Han, Reynold Cheng, Chenhao Ma, and Tobias Grubenmann. 2022. DeepTEA: Effective and efficient online time-dependent trajectory outlier detection. *Proceedings of the VLDB Endowment* 15, 7 (2022), 1493–1505.
[24] Xiaolin Han, Tobias Grubenmann, Reynold Cheng, Sze Chun Wong, Xiaodong Li, and Wenya Sun. 2020. Traffic incident detection: A trajectory-based approach. In *ICDE*. IEEE, 1866–1869.
[25] Xiaolin Han, Tobias Grubenmann, Chenhao Ma, Xiaodong Li, Wenya Sun, Sze Chun Wong, Xuequn Shang, and Reynold Cheng. 2024. FDM: Effective and efficient incident detection on sparse trajectory data. *Information Systems* 125 (2024), 102418.
[26] Liancheng He, Liang Bai, Xian Yang, Hangyuan Du, and Jiye Liang. 2023. High-order graph attention network. *Information Sciences* 630 (2023), 222–234.
[27] Jilin Hu, Chenjuan Guo, Bin Yang, and Christian S. Jensen. 2019. Stochastic weight completion for road networks using graph convolutional networks. In *ICDE*, 1274–1285.
[28] Bosong Huang, Ke Ruan, Weihao Yu, Jing Xiao, Ruzhong Xie, and Jin Huang. 2023. ODformer: Spatial–temporal transformers for long sequence Origin–Destination matrix forecasting against cross application scenario. *Expert Systems with Applications* 222 (2023), 119835.
[29] Chao Huang, Junbo Zhang, Yu Zheng, and Nitesh V. Chawla. 2018. DeepCrime: Attentive hierarchical recurrent networks for crime prediction. In *CIKM*, 1423–1432.
[30] Rongzhou Huang, Chuyin Huang, Yubao Liu, Genan Dai, and Weiyang Kong. 2020. LSGCN: Long short-term traffic prediction with graph convolutional networks. *IJCAI* 7 (2020), 2355–2361.
[31] Yu Huang, Josh Jia-Ching Ying, Philip S. Yu, and Vincent S. Tseng. 2020. Dynamic graph mining for multi-weight multi-destination route planning with deadlines constraints. *ACM Transactions on Knowledge Discovery from Data* 15, 1 (2020), 1–32.
[32] Ziheng Huang, Weihan Zhang, Dujuan Wang, and Yunqiang Yin. 2022. A GAN framework-based dynamic multi-graph convolutional network for origin–destination-based ride-hailing demand prediction. *Information Sciences* 601 (2022), 129–146.
[33] Jianwen Jiang, Yuxuan Wei, Yifan Feng, Jingxuan Cao, and Yue Gao. 2019. Dynamic hypergraph neural networks. In *IJCAI*, 2635–2641.
[34] Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *ICLR*.

[35] Boris Knyazev, Xiao Lin, Mohamed R. Amer, and Graham W. Taylor. 2018. Spectral multigraph networks for discovering and fusing relationships in molecules. *arXiv:1811.09595*. Retrieved from https://arxiv.org/abs/1811.09595
[36] Jae-Gil Lee, Jiawei Han, and Xiaolei Li. 2008. Trajectory outlier detection: A partition-and-detect framework. In *ICDE*, 140–149.
[37] Kai Lei, Meng Qin, Bo Bai, Gong Zhang, and Min Yang. 2019. GCN-GAN: A non-linear temporal link prediction model for weighted dynamic networks. In *IEEE INFOCOM*. IEEE, 388–396.
[38] Xiaoliang Lei, Hao Mei, Bin Shi, and Hua Wei. 2022. Modeling network-level traffic flow transitions on sparse data. In *ACM SIGKDD*, 835–845.
[39] Can Li, Lei Bai, Wei Liu, Lina Yao, and S. Travis Waller. 2020. Graph neural network for robust public transit demand prediction. *IEEE Transactions on Intelligent Transportation Systems* 23, 5 (2020), 4086–4098.
[40] Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. 2017. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. *arXiv:1707.01926*. Retrieved from https://arxiv.org/abs/1707.01926
[41] Meng Liu, Hongyang Gao, and Shuiwang Ji. 2020. Towards deeper graph neural networks. In *ACM SIGKDD*, 338–348.
[42] Yiding Liu, Kaiqi Zhao, Gao Cong, and Zhifeng Bao. 2020. Online anomalous trajectory detection with deep generative sequence modeling. In *ICDE*. IEEE, 949–960.
[43] Chenhao Ma, Reynold Cheng, Laks V. S. Lakshmanan, and Xiaolin Han. 2022. Finding locally densest subgraphs: A convex programming approach. *Proceedings of the VLDB Endowment* 15 (2022), 11.
[44] Chenhao Ma, Yixiang Fang, Reynold Cheng, Laks V. S. Lakshmanan, and Xiaolin Han. 2022. A convex-programming approach for efficient directed densest subgraph discovery. In *SIGMOD*, 845–859.
[45] Chenhao Ma, Yixiang Fang, Reynold Cheng, Laks V. S. Lakshmanan, Xiaolin Han, and Xiaodong Li. 2024. Accelerating directed densest subgraph queries with software and hardware approaches. *The VLDB Journal* 33, 1 (2024), 207–230.
[46] Chenhao Ma, Yixiang Fang, Reynold Cheng, Laks V. S. Lakshmanan, Wenjie Zhang, and Xuemin Lin. 2020. Efficient algorithms for densest subgraph discovery on large directed graphs. In *ACM SIGMOD*, 1051–1066.
[47] Hao Miao, Jiaxing Shen, Jiannong Cao, Jiangnan Xia, and Senzhang Wang. 2022. MBA-STNet: Bayes-enhanced discriminative multi-task learning for flow prediction. *IEEE Transactions on Knowledge and Data Engineering* 35, 7 (2022), 7164–7177.
[48] Benjamin A. Miller, Zohair Shafi, Wheeler Ruml, Yevgeniy Vorobeychik, Tina Eliassi-Rad, and Scott Alfeld. 2023. Attacking shortest paths by cutting edges. *ACM Transactions on Knowledge Discovery from Data* 18, 2 (2023), 1–42.
[49] Medhini Narasimhan, Svetlana Lazebnik, and Alexander G. Schwing. 2018. Out of the box: Reasoning with graph convolution nets for factual visual question answering. In *NIPS*, 31.
[50] Paul Newson and John Krumm. 2009. Hidden Markov map matching through noise and sparseness. In *SIGSPATIAL*.
[51] Simon Aagaard Pedersen, Bin Yang, and Christian S. Jensen. 2019. Fast stochastic routing under time-varying uncertainty. *The VLDB Journal* 29 (2019), 819–839.
[52] Simon Aagaard Pedersen, Bin Yang, and Christian S. Jensen. 2020. Anytime stochastic routing with hybrid learning. *The VLDB Journal* 13, 9 (2020), 1555–1567.
[53] A. Arun Prakash. 2018. Pruning algorithm for the least expected travel time path on stochastic and time-dependent networks. *Transportation Research Part B: Methodological* 108 (2018), 127–147.
[54] Jingbo Shang, Yu Zheng, Wenzhu Tong, Eric Chang, and Yong Yu. 2014. Inferring gas consumption and pollution emission of vehicles throughout a city. In *SIGKDD*, 1027–1036.
[55] Chao Song, Youfang Lin, Shengnan Guo, and Huaiyu Wan. 2020. Spatial-temporal synchronous graph convolutional networks: A new framework for spatial-temporal network data forecasting. *Proceedings of the AAAI Conference on Artificial Intelligence* 34, 01 (2020), 914–921.
[56] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *ICLR*, 1050(20): 10–48550.
[57] Haibo Wang and Kuien Liu. 2012. User oriented trajectory similarity search. In *ACM SIGKDD*, 103–110.
[58] Jingyuan Wang, Qian Gu, Junjie Wu, Guannan Liu, and Zhang Xiong. 2016. Traffic speed prediction and congestion source exploration: A deep learning method. In *ICDM*, 499-508.
[59] Qingyi Wang, Shenhao Wang, Dingyi Zhuang, Haris Koutsopoulos, and Jinhua Zhao. 2024. Uncertainty quantification of spatiotemporal travel demand with probabilistic graph neural networks. *IEEE Transactions on Intelligent Transportation Systems*25, 8 (2024), 8770–8781.
[60] Xiaolong Wang and Abhinav Gupta. 2018. Videos as space-time region graphs. In *ECCV*, 399–417.
[61] Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S. Yu. 2019. Heterogeneous graph attention network. In *WWW*, 2022–2032.
[62] Xiaoyang Wang, Yao Ma, Yiqi Wang, Wei Jin, Xin Wang, Jiliang Tang, Caiyan Jia, and Jian Yu. 2020. Traffic flow prediction via spatial temporal graph neural network. In *WWW*, 1082–1092.
[63] Yongtian Wang, Xinmeng Liu, Yewei Shen, Xuerui Song, Tao Wang, Xuequn Shang, and Jiajie Peng. 2023. Collaborative deep learning improves disease-related circRNA prediction based on multi-source functional information. *Briefings in Bioinformatics* 24, 2 (2023), bbad069.

[64] Yongtian Wang, Wenkai Shen, Yewei Shen, Shang Feng, Tao Wang, Xuequn Shang, and Jiajie Peng. 2024. Integrative graph-based framework for predicting circRNA drug resistance using disease contextualization and deep learning. *IEEE Journal of Biomedical and Health Informatics (2024)*, 1–12.
[65] Yang Wang, Yiwei Xiao, Xike Xie, Ruoyu Chen, and Hengchang Liu. 2018. Real-time traffic pattern analysis and inference with sparse video surveillance information. In *IJCAI*, 3571–3577.
[66] Zheng Wang, Cheng Long, Gao Cong, and Yiding Liu. 2020. Efficient and effective similar subtrajectory search with deep reinforcement learning. arXiv:2003.02542. Retrieved from https://arxiv.org/abs/2003.02542
[67] Zepu Wang, Dingyi Zhuang, Yankai Li, Jinhua Zhao, Peng Sun, Shenhao Wang, and Yulin Hu. 2023. ST-GIN: An uncertainty quantification approach in traffic data imputation with spatio-temporal graph attention and bidirectional recurrent united neural networks. In *ITSC*. IEEE, 1454–1459.
[68] Hua Wei, Chacha Chen, Chang Liu, Guanjie Zheng, and Zhenhui Li. [n. d.]. Learning to simulate on sparse trajectory data, 530–545.
[69] Dawn Woodard, Galina Nogin, Paul Koch, David Racz, Moises Goldszmidt, and Eric Horvitz. 2017. Predicting travel time reliability using mobile phone GPS data. *Transportation Research Part C: Emerging Technologies* 75 (2017), 30-44.
[70] Louis-Pascal Xhonneux, Meng Qu, and Jian Tang. 2020. Continuous graph neural networks. In *ICML*. PMLR, 10432–10441.
[71] Lianghao Xia, Chao Huang, Yong Xu, Peng Dai, Liefeng Bo, Xiyue Zhang, and Tianyi Chen. 2021. Spatial-temporal sequential hypergraph network for crime prediction with dynamic multiplex relation learning. In *IJCAI*, 1631–1637.
[72] Dong Xie, Feifei Li, and Jeff M. Phillips. 2017. Distributed trajectory similarity search. *Proceedings of the VLDB Endowment* 10, 11 (2017), 1478–1489.
[73] Sijie Yan, Yuanjun Xiong, and Dahua Lin. 2018. Spatial temporal graph convolutional networks for skeleton-based action recognition. *Proceedings of the AAAI Conference on Artificial Intelligence 32,* 1 (2018).
[74] Bin Yang, Jian Dai, Chenjuan Guo, Christian S. Jensen, and Jilin Hu. 2018. PACE: A PAth-CEntric paradigm for stochastic path finding. *The VLDB Journal* 27 (2018), 153-178.
[75] Bin Yang, Chenjuan Guo, and Christian S. Jensen. 2013. Travel cost inference from sparse, spatio temporally correlated time series using Markov models. *The VLDB Journal* 6, 9 (2013), 769–780.
[76] Bin Yang, Chenjuan Guo, Christian S. Jensen, Manohar Kaul, and Shuo Shang. 2014. Stochastic skyline route planning under time-varying uncertainty. In *ICDE*,136-147.
[77] Carl Yang, Lanxiao Bai, Chao Zhang, Quan Yuan, and Jiawei Han. 2017. Bridging collaborative filtering and semi-supervised learning: A neural approach for poi recommendation. In *SIGKDD*, 1245–1254.
[78] Sean Bin Yang and Bin Yang. 2020. Learning to rank paths in spatial networks. In *ICDE*, 2006–2009.
[79] Huaxiu Yao, Xianfeng Tang, Hua Wei, Guanjie Zheng, and Zhenhui Li. 2019. Revisiting spatial-temporal similarity: A deep learning framework for traffic prediction. In *AAAI*, Vol. 33, 5668–5675.
[80] Jaehyuk Yi and Jinkyoo Park. 2020. Hypergraph convolutional recurrent neural network. In *ACM SIGKDD*, 3366–3376.
[81] Junbo Zhang, Yu Zheng, and Dekang Qi. 2017. Deep spatio-temporal residual networks for citywide crowd flows prediction. In *AAAI*, Vol. 31.
[82] Minxing Zhang, Dazhou Yu, Yun Li, and Liang Zhao. 2022. Deep geometric neural network for spatial interpolation. In *SIGSPATIAL*, 1–4.
[83] Yingxue Zhang, Yanhua Li, Xun Zhou, Xiangnan Kong, and Jun Luo. 2020. Curb-GAN: Conditional urban traffic estimation through spatio-temporal generative adversarial networks. In *ACM SIGKDD*, 842–852.
[84] Pengpeng Zhao, Anjing Luo, Yanchi Liu, Fuzhen Zhuang, Jiajie Xu, Zhixu Li, Victor S. Sheng, and Xiaofang Zhou. 2020. Where to go next: A spatio-temporal gated network for next POI recommendation. *IEEE Transactions on Knowledge and Data Engineering* 34, 5 (2020), 2512-2524.
[85] Qiang Zhou, Jingjing Gu, Xinjiang Lu, Fuzhen Zhuang, Yanchao Zhao, Qiuhong Wang, and Xiao Zhang. 2021. Modeling heterogeneous relations across multiple modes for potential crowd flow prediction. In *AAAI*, Vol. 35, 4723–4731.
[86] Zhengyang Zhou, Yang Wang, Xike Xie, Lei Qiao, and Yuantao Li. 2021. Stuanet: Understanding uncertainty in spatiotemporal collective human mobility. In *WWW*, 1868–1879.

Received 8 January 2024; revised 7 September 2024; accepted 8 February 2025