Complex & Intelligent Systems (2024) 10:7943–7964
https://doi.org/10.1007/s40747-024-01578-x

ORIGINAL ARTICLE

Generalized spatial–temporal regression graph convolutional transformer for traffic forecasting

Lang Xiong[1] · Liyun Su[1,2] · Shiyi Zeng[1] · Xiangjing Li[1] · Tong Wang[1] · Feng Zhao[1]

Received: 12 April 2024 / Accepted: 21 July 2024 / Published online: 10 August 2024

Abstract

Spatial–temporal data is widely available in intelligent transportation systems, and accurately solving non-stationary of spatial–temporal regression is critical. In most traffic flow prediction research, the non-stationary solution of deep spatial–temporal regression tasks is typically formulated as a spatial–temporal graph modeling problem. However, there are several issues: (1) the coupled spatial–temporal regression approach renders it unfeasible to accurately learn the dependencies of diverse modalities; (2) the intricate stacking design of deep spatial–temporal network modules limits the interpretation and migration capability; (3) the ability to model dynamic spatial–temporal relationships is inadequate. To tackle the challenges mentioned above, we propose a novel unified spatial–temporal regression framework named Generalized Spatial–Temporal Regression Graph Convolutional Transformer (GSTRGCT) that extends panel model in spatial econometrics and combines it with deep neural networks to effectively model non-stationary relationships of spatial–temporal regression. Considering the coupling of existing deep spatial–temporal networks, we introduce the tensor decomposition to explicitly decompose the panel model into a tensor product of spatial regression on the spatial hyper-plane and temporal regression on the temporal hyper-plane. On the spatial hyper-plane, we present dynamic adaptive spatial weight network (DASWNN) to capture the global and local spatial correlations. Specifically, DASWNN adopts spatial weight neural network (SWNN) to learn the semantic global spatial correlation and dynamically adjusts the local changing spatial correlation by multiplying between spatial nodes embedding. On the temporal hyper-plane, we introduce the Auto-Correlation attention mechanism to capture the period-based temporal dependence. Extensive experiments on the two real-world traffic datasets show that GSTRGCT consistently outperforms other competitive methods with an average of 62% and 59% on predictive performance.

Keywords Spatio-temporal decoupling · Graph convolutional network · Auto-correlation · Tensor decomposition · Transformer

Introduction

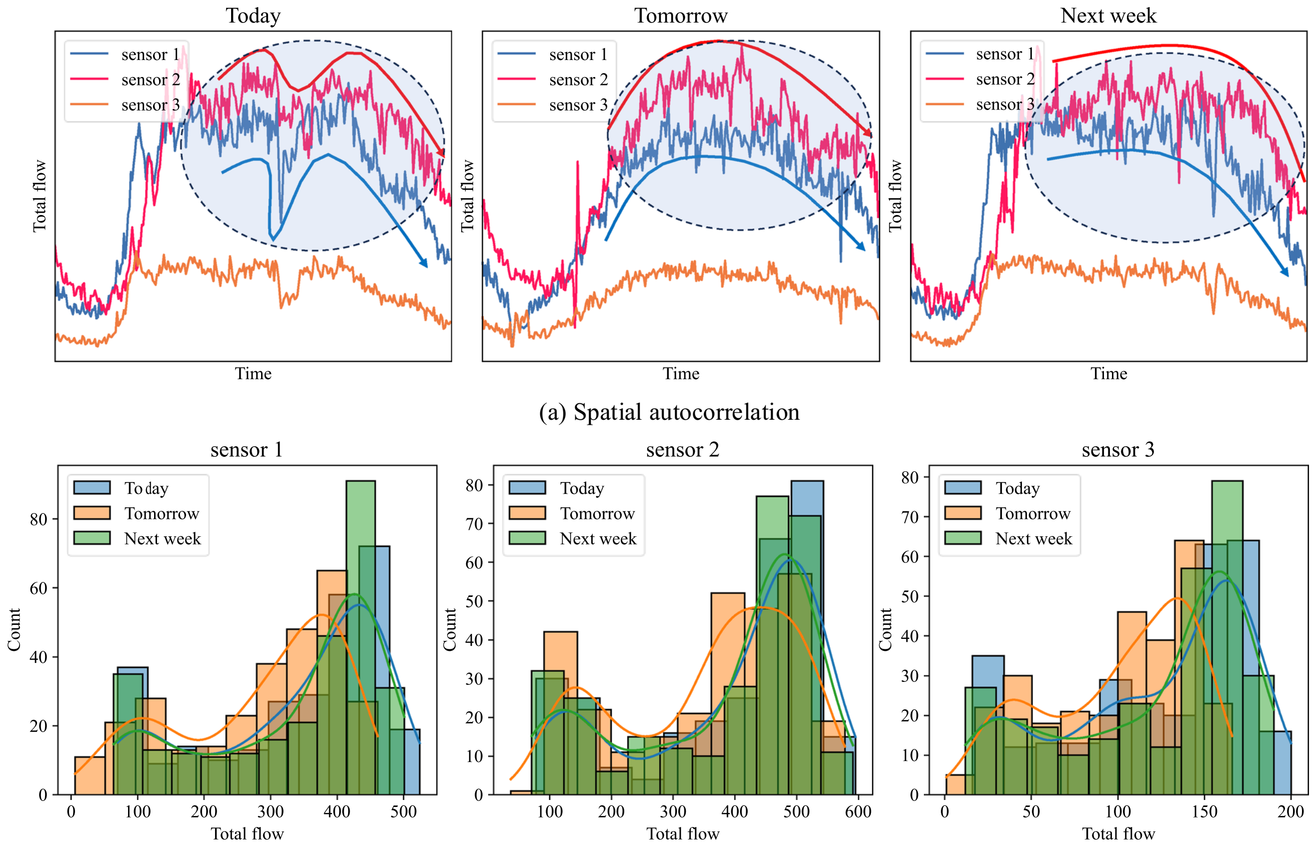
Accurate solutions of non-stationary in spatial–temporal regression have gained increasing attention due to the rapid development of artificial intelligence. Spatial–temporal prediction is a concrete manifestation of spatial–temporal non-stationary solving, such as traffic flow forecasting [1–3], financial stock trend prediction [4–6], and environmental and weather forecasting [7, 8]. Intelligent Transportation Systems (ITS) has become a major research area among these. Accurately predicting future traffic flows using historical spatial–temporal traffic data helps traffic management systems make effective decisions to reduce congestion [9].

In this paper, we discuss the precise solution of spatial–temporal non-stationary through traffic flow prediction. The spatial–temporal non-stationary comprises primarily the spatial correlation of cross-sectional data with the autocorrelation of the spatial unit measurements over time in the dynamical system. Figure 1 illustrates the variation and distribution of traffic flow detected by the sensors on three different roads at different time cycles. “today”, “tomorrow”, “next week” represent different time periods, which are “2016.07.01”, “2016.07.02”, and “2016.07.08”, respectively.We can observe the following phenomena: (1) Both

✉ Liyun Su
cloudhopping@163.com

[1] School of Science, Chongqing University of Technology, Chongqing 400054, China

[2] Center for Spatial-Temporal Big Data Research, Chongqing University of Technology, Chongqing 400054, China



(a) Spatial autocorrelation

(b) Temporal autocorrelation

Fig. 1 Spatial–temporal non-stationarity of traffic flow. **a** Spatial autocorrelation. **b** Temporal autocorrelation

Sensor 1 and Sensor 3 exhibit similar flow trends over time, yet differ from Sensor 2 in Fig. 1a, reflecting the spatial correlation of spatial–temporal data; (2) The distributions of flow from Sensors 1, 2, and 3 seem comparable in the weekly intervals in Fig. 1b, unlike the distribution in the interval day, which manifests the autocorrelation and periodicity of the time series.

Therefore, precisely solving for the spatial–temporal non-stationary and mining the prospective operation patterns of traffic flow are critical for the management efficiency of ITS. Early efforts mainly involved traditional statistical methods to extract linear features [10, 11]. However, they are not sufficiently capable of modeling the nonlinear features of large-scale spatial–temporal data. Due to the superior fitting and nonlinear feature extraction capabilities of artificial intelligence methods, researchers have employed various machine learning methods to predict traffic flow [12, 13]. However, these methods rely on manual feature extraction, which consumes considerable labor and resources. In recent years, deep learning applied to traffic flow prediction has been prevalent. Initially, researchers chose Recurrent Neural Networks (RNN) and Convolutional Neural Networks (CNN) to extract temporal features and spatial features, respectively and finally combined them to capture the spatial–temporal correlation of traffic flow [14–16]. However, RNNs are prone to gradient vanishing and explosion problems while CNNs can only model regular spatial structure data. Subsequently, researchers gradually employed graph convolutional networks (GCN) instead of CNNs to model spatial relationships [17, 18]. In addition, Transformer has exhibited remarkable performance in the areas of time series [19], computer vision [20], and natural language processing [21], especially in parallelizing the modeling of sequence relations.

Despite the success of these efforts in modeling spatial–temporal non-stationarity, they still suffer from several problems as follows:

(1) Coupled spatial–temporal regression approach renders it unfeasible to accurately learn the dependencies of diverse modalities. Spatial–temporal correlation of traffic flow is sophisticated while existing deep methods learn spatial–temporal dependencies by stacking different spatial network modules and temporal network modules to form a coupled spatial–temporal network module. For example, Yu et al. propose Spatio-Temporal

Graph Convolutional Networks (STGCN) to learn spatial–temporal dependence by stacking two gated CNNs coupled in series with GCN modules to form a spatial–temporal convolutional block [22]. Thus, current methodologies rarely sufficiently investigate the decoupling of spatial–temporal relationships.

(2) The intricate stacking design of deep spatial–temporal network modules limits the interpretation and migration capability. The present methods are typically based on a priori domain knowledge to design complicated networks and lack corresponding theoretical support. As a result, their interpretability is poor, and their migration ability for the downstream tasks of the traffic flow is relatively weak.

(3) The ability to model dynamic spatial–temporal relationships is inadequate. The existing GNN-based methods either exclusively use static graphs based on distance measurements or allow traffic roads at different moments to share identical dynamic graphs [23, 24]. Moreover, they scarcely incorporate the existence of periodic dependence in the traffic flow.

We treat traffic flow prediction as a unified spatial–temporal regression problem. Aiming at the aforementioned issues of traffic flow prediction, we proceed from the traditional spatial–temporal regression theory and propose Generalized Spatial–Temporal Regression Graph Convolutional Transformer (GSTRGCT) as a novel deep learning paradigm. Firstly, we theoretically prove that the panel model, GCN, and Transformer have similar propagation mechanisms. Then, the tensor decomposition is introduced to explicitly decompose the panel model into a tensor product of spatial regression on the spatial hyper-plane and temporal regression on the temporal hyper-plane. Finally, we utilize GCN and Transformer encoder with Auto-Correlation to learn the spatial hyper-plane and temporal hyper-plane dependencies respectively. Furthermore, on the spatial hyperplane, we present dynamic adaptive spatial weight network (DASWNN) to capture the global and local spatial correlations. Specifically, DASWNN adopts spatial weight neural network (SWNN) to learn the semantic global spatial correlation and dynamically adjusts the local changing spatial correlation by multiplying between spatial nodes embedding. On the temporal hyper-plane, we introduce the Auto-Correlation attention mechanism to capture the period-based temporal dependence. To sum up, the contributions of this paper are as follows:

- We propose the Generalized Spatial–Temporal Regression Graph Convolutional Transformer (GSTRGCT), a unique deep-learning paradigm to address complex nonlinear spatial–temporal regression tasks such as traffic forecasting.
- To reduce the complexity of calculating the spatial–temporal weight, a tensor decomposition approach is introduced to decouple the panel model and decompose the coupled spatial–temporal weight into spatial weight and temporal weight.
- Based on tensor decomposition, a dynamic adaptive spatial weight neural network (DASWNN) is proposed to combine the prior spatial weight with the posterior spatial weight to accurately solve adaptive spatial structure relationships and characterize the spatial weight. Meanwhile, an auto-correlation attention mechanism is introduced to capture the period-based dependence of the time series and characterize the temporal weight with autocorrelation.
- The results of extensive experiments on the two real-world traffic datasets (PeMS04 and PeMS08) prove that GSTRGCT achieves an excellent prediction effect and is superior to existing competitive methods.

The remainder of this paper is organized as follows: In Sect. "Related work", we briefly review the current panel models, related studies on graph-based deep learning and Transformer, and the mainstream of spatial–temporal prediction. In Sect. "Preliminaries", we provide preliminaries for spatial–temporal prediction. In Sect. "Methodology", we describe the main process of our proposed GSTRGCT model. In Sect. "Experiments and results", we introduce the datasets, experiment settings, and the results of the proposed models compared with the other competitive methods in traffic forecasting. In Sect. "Conclusion and future work", we conclude this paper, discuss the results, and outline some directions for future research.

Related work

Panel models

In spatial econometrics, the modeling approach of spatial–temporal regression models exists mainly in panel models that study spatial correlation and heterogeneity. Panel models are extended to spatial–temporal models with specific spatial or temporal effects by adding a temporal dimension to a generalized nested spatial model (GNS) based on cross-sectional data that takes into account spatial and temporal heterogeneity. The application of panel models and parameter estimation are described in detail in Anselin's book Spatial Econometrics: Methods and Models [25]. A systematic review of the research progress in spatial econometrics is presented in [26]. Here, our focus is on the spatial–temporal regression models (or panel models) discussed. Since it takes the form of adding a temporal dimension to the generalized nested model, we call it a generalized spatial–temporal regression model (GSTR) in this paper and will not repeat it subsequently. The GSTR model can be written in the following form:

$$\begin{aligned} \boldsymbol{Y}_t &= \rho W \boldsymbol{Y}_t + \alpha \boldsymbol{\iota}_N + \boldsymbol{X}_t \beta + W \boldsymbol{X}_t \theta + \mu + \xi_t \boldsymbol{\iota}_N + \boldsymbol{u}_t \\ \boldsymbol{u}_t &= \lambda W \boldsymbol{u}_t + \boldsymbol{\varepsilon}_t \quad (t = 1,2, \cdots, T) \end{aligned} \tag{1}$$

where $\boldsymbol{Y}_t \in \mathbb{R}^{N \times F} (F = 1$ in panel models) and $\boldsymbol{X}_t \in \mathbb{R}^{N \times C}$ represent the response variable (consists of all observations of the explanatory variable for N nodes in the sample, $i = 1, 2, \cdots, N$), and independent variable, respectively. $\boldsymbol{W} \in \mathbb{R}^{N \times N}$(diagonal elements is 0) represents the adjacency matrix or spatial weight matrix with spatial structural relationship. The value of its elements consists of 0 and 1. $w_{ij} = 0$ indicates that the node i is adjacent to the node j. Conversely, $w_{ij} = 1$ means that the node i is not adjacent to the node $j.\boldsymbol{X}_t \in \mathbb{R}^{N \times C}$ represents the independent variable at the time t, with ρ and $\boldsymbol{\beta}$ the parameters to be estimated. $\boldsymbol{\mu} = (\mu_1, ..., \mu_N)^T$ the intercept term. μ_i and ξ_t represent the specific effect of a spatial unit and the specific effect of time, respectively, and are usually treated together with ε_{it} as independent identically distributed random variables with mean 0 and variance σ^2. α, $\boldsymbol{\beta}$ and $\boldsymbol{\theta}$ are the parameters to be estimated. $\boldsymbol{\iota}_N$ is a vector or matrix with all elements 1. ρ and λ represent the spatial autoregressive and spatial autocorrelation coefficients, respectively.

Graph convolutional network for spatial dependence learning

Compared to convolutional neural networks (CNNs) used to learn structured features in regular spatial domains, graph-based learning focuses more on irregular domains of spatial nodes. Recent literature has focused on graph convolutional networks (GCNs) to characterize the relationships between spatial nodes [27, 28]. GCNs can be divided into two types of approaches, spectral-based GCNs and spatial-based GCNs. The spatial-based approach represents graph convolution as aggregating feature information from the neighborhood. So, the key to this type of approach is to find the domain of nodes [29]. Spectral-based methods define graph convolution by introducing filters from the perspective of graph signal processing, where the graph convolution operation is interpreted as removing noise from the graph signal. Fourier transform is proposed to extract signals for the spectral-based methods [28]. To simplify the computation of the eigenvalues of the graph convolution, Defferrard et al. proposed to use Chebyshev polynomials to approximate the graph convolution process [30]. Many researchers have used Chebyshev polynomial-based graphical convolution to extract spatial node representation information, and have carried out related application work, showing the powerful spatial feature learning capability of graphical convolution [22, 23]. Some other work focuses on the improvement and design of adaptive graph adjacency matrix [31–33]. The above work provides the foundation for the design of spatial structures for the extension of our panel model. However, due to their overly complex design, the dynamic spatial weights relative to the panel model are not compatible. In this paper, we directly use the Chebyshev polynomial-based graph convolution network to learn the spatial representation, which is isomorphic to spatial regression. Specific details will be elaborated later.

Transformer for temporal dependence learning

RNNs and CNNs are prevalent for modeling dependencies, but they have limitations due to their vanishing gradients, and explosion problems in training [34]. Although Gated Recurrent Units (GRUs) [35] and Long-Short Term Memory (LSTM) [36] are developed to alleviate these drawbacks, they still suffer from time-consuming iterations and error accumulation due to their serial operation for temporal dependence. In 2017, Transformer was proposed as a model architecture that uses a purely attentional mechanism to process temporal dependencies in parallel, completely breaking the status of previous RNNs. In recent years, Transformer has been widely and effectively used in various fields, such as Bert [37], GPT-3 [38] in natural language processing, VIT [39] in computer vision, etc. It is worth noting that Transformer also has a broad application prospect in various time series tasks, such as Informer [40] and Autoformer [41], which realize the extraction of long-term dependencies of time series. The above work is for 2-D multidimensional time series prediction. For 3-d spatial–temporal data, Xu et al. used the Transformer architecture to extract the time series dependence to predict traffic flow and achieved the best results [42].

Transformer consists of encoder and decoder. In this paper, we consider the advantages of the parallel architecture of the Transformer attention mechanism and the adaptability of the encoder autoregression task to the panel modeling session task, so we employ the improved Transformer encoder as the temporal dependency extraction architecture in this paper. The details of the improvement are described in Sect. "Methodology".

Preliminaries

Notations and definitions

In this section, we introduce the notations and definitions presented for spatial–temporal non-stationary solving problems in this paper. Note that the spatial–temporal non-stationarity solving problem is transformed into a spatial–temporal prediction problem in this paper.

Definition 1 (*Graph structure*) Graph is composed of vertex and edges, the general definition of a graph is:

$$G = (V, E, \boldsymbol{A}) \tag{2}$$

where $V = \{v_1, v_2, \cdots, v_N\}$ and $E = \{e_1, e_2, \cdots, e_m\}$ represent the set of vertices and the set of edges, respectively. Each edge represents the connection of two vertices, e.g., $e_{ij} = \left(v_i, v_j\right) \in E$ means that the node i is connected to the node j. $\boldsymbol{A} \in \mathbb{R}^{N \times N}$ is the adjacency matrix, which is similar to the spatial weight in this paper.

Definition 2 (*Tensor*) A tensor can be thought of as a multidimensional array whose elements can be given coordinates to be accessed by index. The order of a tensor indicates how many indexes there are. Here, plain letters (e.g., x) are used to denote scalars. The boldface lowercase (e.g., $\boldsymbol{x} \in \mathbb{R}^I$) and uppercase letters (e.g., $\boldsymbol{X} \in \mathbb{R}^{I \times J}$) are used for vectors and matrices, respectively. For tensors, they are denoted by boldface calligraphic letters (e.g., $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$). In this paper, we define the spatial–temporal input tensor as $\mathcal{X} \in \mathbb{R}^{T \times N \times C}$ and the spatial–temporal output tensor as $\mathcal{Y} \in \mathbb{R}^{T \times N \times F}$.

Definition 3 (*Spatial hyper-plane*) In this paper, spatial hyper-plane refers to the two-dimensional plane consisting of spatial nodes and channels. It provides a virtual and potential representation for characterizing spatial dependence. At a specific moment t, the input of the spatial hyper-plane is $\boldsymbol{X}_S \in \mathbb{R}^{N \times C}$. For the spatial–temporal input tensor, we employ $\mathcal{X}_S \in \mathbb{R}^{T \times N \times C}$ as the input tensor of the spatial hyper-plane. To simplify the form of expression, we denote $\mathcal{X}_S^{t,:,:}$ as the input tensor of the spatial hyper-plane at time t.

Definition 4 (*Temporal hyper-plane*). In this paper, temporal hyper-plane refers to the two-dimensional plane consisting of temporal sequences and channels. It provides a virtual and potential representation for characterizing temporal dependence. For each node i, the input of the temporal hyper-plane is $\boldsymbol{X}_T \in \mathbb{R}^{T \times C}$. For the spatial–temporal input tensor, we employ $\mathcal{X}_T \in \mathbb{R}^{N \times T \times C}$ as the input tensor of the temporal hyper-plane. To simplify the form of expression, we denote $\mathcal{X}_T^{i,:,:}$ as the input tensor of the temporal hyper-plane for the node i.

Problem formulation

The goal of spatial–temporal prediction is outputting the response variable in T_p future time slots and inputting the independent variable in T past time slots. Then, there is a mapping: $f : \mathbb{R}^{T \times N \times C} \rightarrow \mathbb{R}^{T_p \times N \times F}$, denoted as $\mathcal{Y}_{t+1:t+T_p,:,:} = f\left(\mathcal{X}_{t-T+1:t,:,:}, \mathcal{W}_{ST}\right)$. $\mathcal{W}_{ST}$ is a learnable spatial–temporal weight. In this paper, we choose traffic flow forecasting as an empirical study, which will be presented later.

Methodology

Proposed model

Based on the aforementioned considerations in Sect. "Introduction", we propose a generalized spatial–temporal regression graph transformer with the auto-correlation mechanism (GSTRGCT) model, which consists of two different hyperplanes with a similar but heterogeneous structure. Figure 2 displays the overall framework of GSTRGCT from a macro perspective. As an extension of the panel model for deep learning, firstly, GSTRGCT decouples the spatial–temporal regression into spatial regression on the spatial hyper-plane and temporal regression on the temporal hyper-plane based on the tensor decomposition, which employs improved GCN and Transformer Encoder to learn representations dynamically at different scales. Then, the spatial and temporal representations are aggregated by the tensor product to obtain the spatial–temporal representation. Finally, the full connectivity of MLP is applied to achieve the regression task.

Decoupling of spatial–temporal regression based on tensor decomposition

Since traditional panel models are incapable of extracting nonlinear features, which makes it challenging to precisely mine spatial–temporal patterns, it is necessary to extend it to the nonlinear domain. Deep learning architecture for spatial–temporal regression is an important way to address this problem. Recall the formula Eq. (1) of GSTR whose corresponding spatial–temporal regression modeling task can be represented as a flow in Fig. 3. The main steps are: (1) Collection of 3-dimensional spatial–temporal data in the real environment; (2) Specify panel models with various temporal and spatial effects; (3) Estimate parameters in the model; (4) Predict values of response variables for each spatial unit. From Fig. 3, it can be seen that each space unit enjoys the same spatial structural relationships at different moments, which means the spatial weight remains constant over time. Obviously, this assumption is not reasonable due to the complexity of the real world. In this paper, we assume that the spatial weight is dynamically changing, which covers both spatially correlated and temporally correlated. To rationalize the application of dynamically varying spatial weight to GSTR, we simplify Eq. (1) as follows:

$$\begin{aligned}\boldsymbol{Y}_t = {} & (\boldsymbol{I}_N - \rho \boldsymbol{W})^{-1}[\alpha \boldsymbol{\iota}_N + \boldsymbol{X}_t \boldsymbol{\beta} + \boldsymbol{W} \boldsymbol{X}_t \boldsymbol{\theta} \\ & + \boldsymbol{\mu} + \xi_t \boldsymbol{\iota}_N + (\boldsymbol{I}_N - \lambda \boldsymbol{W})^{-1} \boldsymbol{\varepsilon}_t\big]\end{aligned}$$

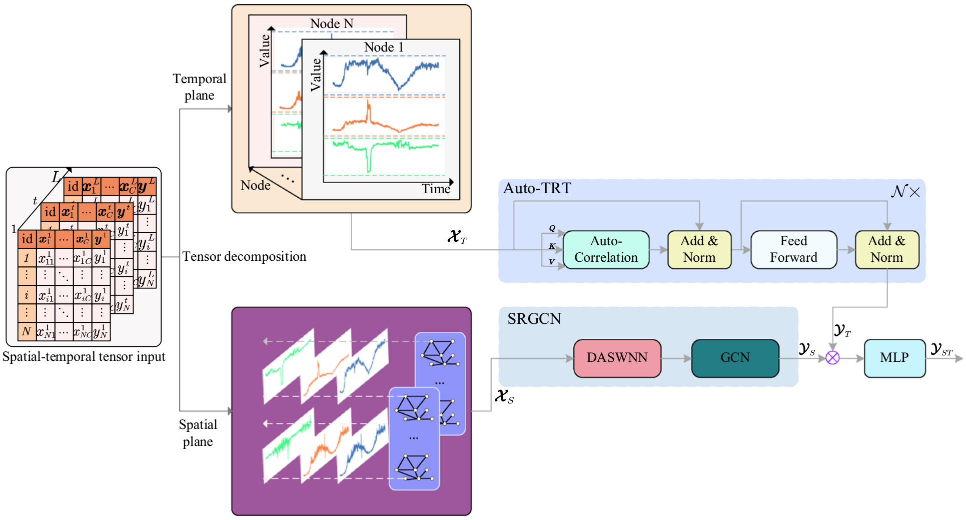


Fig. 2 The framework of GSTRGCT

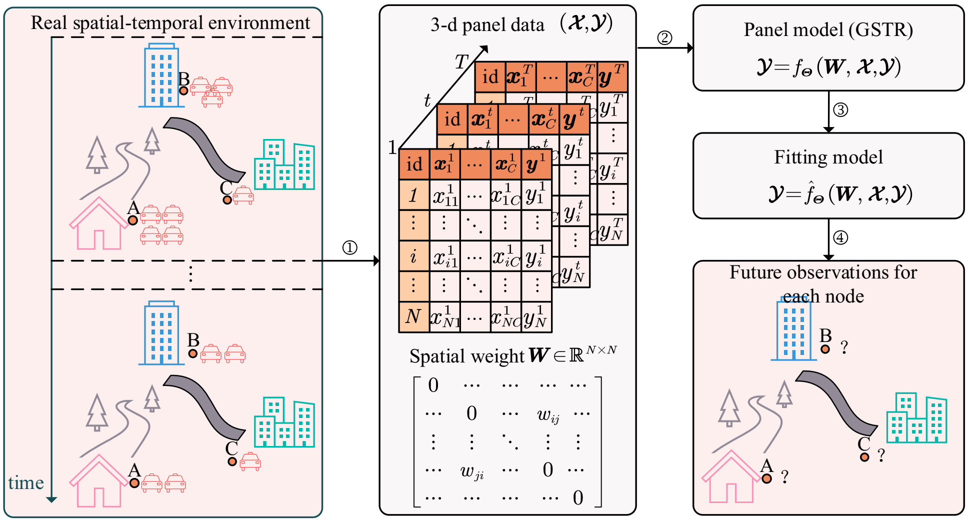

Real spatial-temporal environment
B
A
C
time
①
3-d panel data $(\mathcal{X},\mathcal{Y})$
Spatial weight $\boldsymbol{W}\in\mathbb{R}^{N\times N}$
②
Panel model (GSTR)
$\mathcal{Y}=f_\Theta(\boldsymbol{W},\mathcal{X},\mathcal{Y})$
③
Fitting model
$\mathcal{Y}=\hat{f}_\Theta(\boldsymbol{W},\mathcal{X},\mathcal{Y})$
④
Future observations for each node
?

Fig. 3 Spatial–temporal regression modeling process for GSTR

$$= \sum_{k=0}^{\infty}(\rho \boldsymbol{W})^k \widetilde{\boldsymbol{X}}_t\widetilde{\boldsymbol{\theta}} + \left[\begin{array}{c} \sum\limits_{k=0}^{\infty}(\rho \boldsymbol{W})^k \sum\limits_{l=0}^{\infty}(\lambda \boldsymbol{W})^l \boldsymbol{\varepsilon_t} \\ + \sum\limits_{k=0}^{\infty}(\rho \boldsymbol{W})^k \xi_t \boldsymbol{\iota}_N + \sum\limits_{k=0}^{\infty}(\rho \boldsymbol{W})^k \boldsymbol{\mu} \end{array}\right]$$

$$= \lim_{K\to\infty}\sum_{k=0}^{K}(\rho \boldsymbol{W})^k \widetilde{\boldsymbol{X}}_t\widetilde{\boldsymbol{\theta}} + \widetilde{\boldsymbol{\varepsilon}}_t \tag{3}$$

where $\widetilde{\boldsymbol{X}}_t = (\boldsymbol{\iota}_N, \boldsymbol{X}_t, \boldsymbol{W}\boldsymbol{X}_t) \in \mathbb{R}^{N\times(2C+1)}$, $\boldsymbol{\Theta} = \widetilde{\boldsymbol{\theta}} = \left[\alpha, \boldsymbol{\beta}^T, \boldsymbol{\theta}^T\right]^T \in \mathbb{R}^{(2C+1)\times F}$, $|\rho|\leq 1$, $|\lambda|\leq 1$, $\widetilde{\boldsymbol{\varepsilon}}_t = \sum_{k=0}^{\infty}(\rho \boldsymbol{W})^k\sum_{l=0}^{\infty}(\lambda \boldsymbol{W})^l\boldsymbol{\varepsilon_t} + \sum_{k=0}^{\infty}(\rho \boldsymbol{W})^k\xi_t\boldsymbol{\iota}_N + \sum_{k=0}^{\infty}(\rho \boldsymbol{W})^k\boldsymbol{\mu}$ denotes the negligible residual. Let $\widetilde{\boldsymbol{W}} = (\boldsymbol{I}_N - \rho \boldsymbol{W})^{-1}$, Eq. (3) can be written:

$$\boldsymbol{Y}_t = \widetilde{\boldsymbol{W}}\widetilde{\boldsymbol{X}}_t\Theta + \widetilde{\boldsymbol{\varepsilon}}_t \tag{4}$$

For the 3-d spatial–temporal tensor, the above equation can be expressed in tensor form:

$$\mathcal{Y} = \mathcal{W} \circledast \mathcal{X} \circledast \boldsymbol{\Phi} + \mathcal{E} \tag{5}$$

where $\circledast$ represents the tensor product, following Einstein's law of summation conventions. $\mathcal{W} \in \mathbb{R}^{T\times N\times N}$, $\mathcal{X} \in \mathbb{R}^{T\times N\times \widetilde{C}}$, $\boldsymbol{\Phi} \in \mathbb{R}^{T\times \widetilde{C}\times F}$, $\mathcal{Y} \in \mathbb{R}^{T\times N\times F}$, $\mathcal{E} \in \mathbb{R}^{T\times N}$, where $\widetilde{C} = 2C + 1$. Similar to matrix multiplication, the product form $\circledast$ of the tensor is omitted in this paper, i.e., Eq. (5) is abbreviated as $\mathcal{Y} = \mathcal{W}\mathcal{X}\boldsymbol{\Phi} + \mathcal{E}$, which will not be repeated in the subsequent text. The spatial weight $\mathcal{W}$, which changes dynamically over time, is called spatial–temporal weight and uniformly denoted as $\mathcal{W}_{ST}$. Tensor $\mathcal{Y}$ is written $\mathcal{Y}_{ST}$, Tensor $\mathcal{X}$ is written $\mathcal{X}$ or $\mathcal{X}_{ST}$. Assuming that spatial–temporal weight can be decomposed into a tensor product of spatial weight and temporal weight, i.e., $\mathcal{W}_{ST} = \mathcal{W}_S \otimes \mathcal{W}_T$, where $\otimes$ represents Hadamard product. Since the response variable of GSTR is independent for each time point and the temporal weight does not take into account the time-series autocorrelation, we have $\mathcal{W}_T = \mathcal{I}$, where $\mathcal{I}$ consists of a vector of unit arrays of length T. However, considering the autocorrelation of the time series in the real world and the future response variable is correlated with the past response variable, let $\mathcal{Y}_{future} \in \mathbb{R}^{T_p \times N \times F}$ as the future response variable for T_p time slots and $\mathcal{Y}_{history} \in \mathbb{R}^{T \times N \times F}$ as the past response variable for T time slots, there is a time weight $\mathcal{W}_T$ to satisfy:

$$\mathcal{Y}_{future} = MLP\left(\mathcal{W}_T \mathcal{Y}_{history} \mathbf{\Phi}_T\right) \tag{6}$$

MLP is a linear multilayer perceptual machine and is denoted as a prediction head, which is regarded as a fully connected network. $\mathbf{\Phi}_T$ denotes temporal regression parameters. Considering that there are mapping and spatial autocorrelation between the historical response variable and historical independent variable, i.e., $\mathcal{Y}_{history} = \mathcal{W}_S \mathcal{X}_{history} \mathbf{\Phi}_S$, then Eq. (6) can be written as:

$$\mathcal{Y}_{future} = MLP\left(\mathcal{W}_T \left(\mathcal{W}_S \mathcal{X}_{history} \mathbf{\Phi}_S\right) \mathbf{\Phi}_T\right) \tag{7}$$

where $\mathbf{\Phi}_S$ denotes spatial regression parameter, $\mathcal{X}_{history} \in \mathbb{R}^{T \times N \times \widetilde{C}}$. On the other hand, the future response variable can be represented by the mapping of the spatial regression of the predicted future independent variable, then we have:

$$\begin{aligned} \mathcal{Y}_{future} &= \mathcal{W}_S MLP\left(\mathcal{W}_T \mathcal{X}'_{history} \mathbf{\Phi}_T\right) \mathbf{\Phi}_S \\ &= MLP\left[\mathcal{W}_S \left(\mathcal{W}_T \mathcal{X}'_{history} \mathbf{\Phi}_T\right) \mathbf{\Phi}_S\right] \end{aligned} \tag{8}$$

where $\mathcal{X}_{\text{history}'} \in \mathbb{R}^{\text{N} \times \text{T} \times \tilde{\text{C}}}$, is the transpose of $\mathcal{X}_{\text{history}}$ in the first two dimensions. From Eqs. (7) and (8), it is clearly seen that the spatial–temporal $\mathcal{Y}_{\text{future}}$ can be mapped by the input $\mathcal{X}_{\text{history}}$ of spatal hyper-plane and the input $\mathcal{X}_{\text{history}'}$ of temporal hyper-plane with spatial–temporal weight in a coupled form, i.e., $\mathcal{Y}_{\text{ST}} = \text{MLP}(\text{f}(\mathcal{W}_\text{S}, \mathcal{W}_\text{T}, \mathcal{X}, \mathbf{\Phi}_\text{S}, \mathbf{\Phi}_\text{T}))$. However, due to the sophistication of coupled form, which leads to $\mathcal{O}\left(\text{N}^2\text{T}^2\right)$ computation complexity in the settlement of the spatial–temporal weight. To reduce its computational complexity, this paper adopts a decoupling form based on tensor decomposition to transform the above equation:

$$\begin{aligned} \mathcal{Y}_{ST} &= MLP(f(\mathcal{W}_S, \mathcal{W}_T, \mathcal{X}, \mathbf{\Phi}_S, \mathbf{\Phi}_T)) \\ &= MLP\left(f_S(\mathcal{W}_S, \mathcal{X}_S, \mathbf{\Phi}_S) \otimes f'_T(\mathcal{W}_T, \mathcal{X}_T, \mathbf{\Phi}_T)\right) \\ &= MLP\left(\mathcal{Y}_S \otimes \mathcal{Y}'_T\right) \end{aligned} \tag{9}$$

where $\mathcal{X}_S \in \mathbb{R}^{T \times N \times \widetilde{C}}$ and $\mathcal{X}_T \in \mathbb{R}^{N \times T \times \widetilde{C}}$ are two transformed forms of original spatial–temporal tensor input $\mathcal{X}$, which are transposed to each other, representing the input of the spatial hyper-plane and the input of the temporal hyper-plane, respectively. f_S is the mapping function in the spatial hyper-plane, called SRGCN in this paper. f_T is the mapping function in the temporal hyper-plane, called Auto-TRT in this paper. $\mathcal{Y}_S \in \mathbb{R}^{T \times N \times d_{\text{model}}}$ and $\mathcal{Y}_T \in \mathbb{R}^{N \times T \times d_{\text{model}}}$ are the output of the spatial hyper-plane and the output of the temporal hyper-plane, respectively.$\mathcal{Y}_{T'} \in \mathbb{R}^{T \times N \times d_{\text{model}}}$ is the transpose of $\mathcal{Y}_T$ in the first two dimensions and d_{mdoel} is the dimension of the hidden layer (or $\mathbf{\Phi}_S$, $\mathbf{\Phi}_T$). $\mathcal{W}_S$, $\mathcal{W}_T$ are the spatial weight and temporal weight to be learned, respectively. In this paper, DASWNN is designed to estimate $\mathcal{W}_S$, and Auto-Correlation is designed to estimate $\mathcal{W}_T$. The computational complexity of the spatial–temporal weight is reduced to $\mathcal{O}\left(TN^2 + NT^2\right)$ by decoupling the computation based on tensor decomposition when N and T are large.

Spatial regression graph convolutional network (SRGCN)

SRGCN performs modeling spatial dependence with the input $\mathcal{X}_S \in \mathbb{R}^{T \times N \times \widetilde{C}}$ on the spatial hyper-plane. As shown in Fig. 2, it consists of two modules including DASWNN and GCN.

Dynamic adaptive spatial weight network (DASWNN)

DASWNN is designed to accurately solve adaptive spatial structure relationships, which consist of a prior spatial weight weighted with a posterior adaptive spatial weight.

Prior spatial weight In this paper, we use the node relationship data of metric space and topological space as the input of the prior information, and then we design a spatial weight neural network (SWNN) to merge the prior information to receive a prior spatial weight with semantic information. Prior spatial weight represents the global stable spatial structure. The structure of SWNN is shown in Fig. 4.

Specifically, let the coordinates of any point in the metric space $P(u, v, z, \theta, \phi, r)$, then the distance

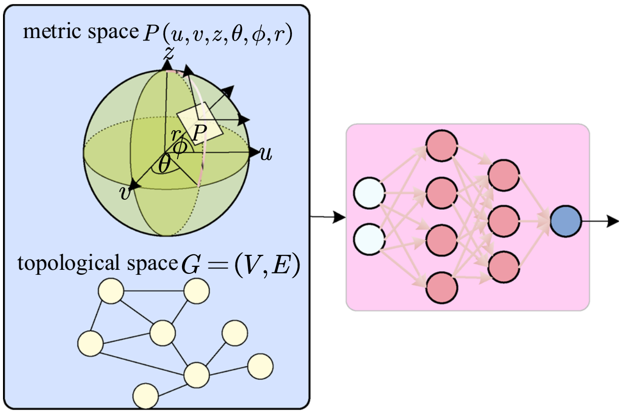

metric space $P(u,v,z,\theta,\phi,r)$
z
P
r
φ
θ
u
v
topological space $G=(V,E)$

Fig. 4 SWNN architecture

between node i and node j can be expressed as $d^S_{ij} = f_{MD}\left(u_i - u_j, v_i - v_j, z_i - z_j, \theta_i - \theta_j, \phi_i - \phi_j, r_i - r_j\right)$, where f_{MD} denotes the distance function of the metric space, such as Euclidean distance function; For the topological space, the distance between nodes is composed of the set of vertices V and edges E, then the distance between node i and node j can be expressed as $d^S_{ij} = f_{TD}(V, E, i, j)$, where f_{TD} denotes the distance function of the topological space. SWNN receives the two combined as input.

$$\boldsymbol{D}^{prior}_{ij} = \Big\{ f_{MD}\left(du_{ij}, dv_{ij}, dz_{ij}, d\theta_{ij}, d\phi_{ij}, dr_{ij}\right), \\ f_{TD}(V, E, i, j), \cdots \Big\} \tag{10}$$

Omitted symbols indicate that other prior spatial structure information can be added. Finally, the prior spatial weight between node i and node j can be computed:

$$\boldsymbol{W}^{prior}_{ij} = softmax\left(\text{SWNN}\left(\boldsymbol{D}^{prior}_{ij}\right)\right) \tag{11}$$

Posterior spatial weight In a real complex environment, the prior structural relationships of spatial nodes are often inaccessible, making it impossible to accurately calculate spatial non-stationarity for spatial–temporal regression. To address this problem, inspired by the literature [24], this paper utilizes a data-driven approach to adaptively compute the posterior spatial weight. The posterior spatial weight represents the local spatial structure with slight perturbations. Specifically, considering the complexity of calculating the spatial correlation at a certain moment is $\mathcal{O}\left(N^2\right)$, when N is larger, the complexity is higher. To reduce its computational complexity, we employ the potential spatial embedding $\mathcal{E}_{t,:,:} \in \mathbb{R}^{N \times E} (E \ll N)$ for spatial nodes at each moment, which is solved adaptively by parametric learning. The posterior spatial weight is calculated as follows:

$$\mathcal{W}^{posteriori}_{t,:,:} = softmax\left(\text{ReLU}\left(\mathcal{E}_{t,:,:}, \mathcal{E}^T{}_{t,:,:}\right)\right) \tag{12}$$

Finally, the dynamic adaptive spatial weight $\mathcal{W}_{t,:,:} \in \mathbb{R}^{N \times N}$ can be calculated by weighted summing the prior spatial weight $\boldsymbol{W}^{\text{prior}}$ with the posterior spatial weight $\mathcal{W}^{\text{posterior}}_{t,:,:}$:

$$\mathcal{W}_{t,:,:} = softmax\left(\alpha \boldsymbol{W}^{prior} + (1-\alpha)\mathcal{W}^{posteriori}_{t,:}\right) \tag{13}$$

where $\alpha \in [0, 1]$ denotes the weighting factor. In particular, $\alpha = 0$ if there is no prior spatial structure information.

Graph convolutional network (GCN)

Although there are two forms of graph convolutional networks, including the spatial-based GCN and the spectral-based GCN, we find that the spectral-based GCN of Chebyshev polynomial approximation has a similar propagation mechanism to GSTR. Here, we directly give the convolutional formula of the signal on a graph:

$$x *_G g_\theta = U g_\theta(\boldsymbol{\Lambda}) \boldsymbol{U}^T x = g_\theta(\boldsymbol{L}) x \tag{14}$$

where $\boldsymbol{U} \in \mathbb{R}^{N \times N}$ is the orthogonal matrix consisting of the eigenvectors of the normalized Laplace matrix $\boldsymbol{L} = \boldsymbol{I}_N - \boldsymbol{D}^{-\frac{1}{2}} \boldsymbol{A} \boldsymbol{D}^{-\frac{1}{2}} = \boldsymbol{U} \boldsymbol{\Lambda} \boldsymbol{U}^T$ ($\boldsymbol{I}_N$ is the identity matrix, $\boldsymbol{D} \in \mathbb{R}^{N \times N}$ is the diagonal degree matrix with $\boldsymbol{D}_{ii} = \sum_j \boldsymbol{A}_{ij}$. Considering the high complexity of computing the eigenvectors of the Laplace matrix, literature [30] proposes to approximate $g_\theta(\boldsymbol{\Lambda})$ by the Chebyshev K-order polynomial, i.e., $g_\theta\left(\widetilde{\boldsymbol{\Lambda}}\right) \approx \sum_{k=0}^{K} \theta_k T_k\left(\widetilde{\boldsymbol{\Lambda}}\right)$, where $\widetilde{\boldsymbol{\Lambda}} = 2\boldsymbol{\Lambda}/\lambda_{\max} - \boldsymbol{I}_N$ ($\lambda_{\max}$ is the largest eigenvalue of $\boldsymbol{L}$). Finally, the GCN formula of the Chebyshev approximation can be written:

$$x *_G g_\theta = \sum_{k=0}^{K} \theta_k T_k\left(\widetilde{\boldsymbol{L}}\right) x \tag{15}$$

where $T_k(x) = 2x T_{k-1}(x) - T_{k-2}(x)$ is the Chebyshev polynomial of order k, satisfying $T_0(x) = 1$, and $T_1(x) = x$. $\widetilde{\boldsymbol{L}} = 2\boldsymbol{L}/\lambda_{\max} - \boldsymbol{I}_N$.

In this paper, for the spatial node signal $\mathcal{X}^{t,:,:}_S \in \mathbb{R}^{N \times \widetilde{C}}$ at the moment t, according to Eq. (15), we have

$$\mathcal{X}^{t,:,:}_S *_G g_{\boldsymbol{\Theta}} = \sum_{k=0}^{K} T_k\left(\widetilde{\boldsymbol{L}}\right) \mathcal{X}^{t,:,:}_S \boldsymbol{\Theta}^{t,:,:}_k \tag{16}$$

where $\boldsymbol{\Theta}^{t,:,:}_k \in \mathbb{R}^{\widetilde{C} \times d_{\text{model}}}$ is the shared parameter at moment t. To make the model more robust, if the input spatial node signal is transmitted through the GCN with a single layer by letting $\boldsymbol{\Theta}_t = \boldsymbol{\Theta}^{t,:,:}_k$, the output $\mathcal{Y}^{t,:,:}_S \in \mathbb{R}^{N \times d_{\text{model}}}$ can be written as:

$$\mathcal{Y}^{t,:,:}_S = \sum_{k=0}^{K} T_k\left(\widetilde{\boldsymbol{L}}\right) \mathcal{X}^{t,:,:}_S \boldsymbol{\Theta}_t \tag{17}$$

Compared Eq. (17) with Eq. (3), it is clear that the propagation mechanism of GCN based on the Chebyshev approximation is similar to the propagation mechanism of GSTR for the spatial regression at a certain moment t, which explains why we chose GCN to fit the spatial regression for panel models. Finally, to capture the dynamic variability of the spatial structure, we take the dynamic adaptive spatial weight $\mathcal{W}_{t,:,:} \in \mathbb{R}^{N \times N}$ as the new adjacency matrix into the GCN to obtain the final SRGCN expression:

$$\mathcal{Y}^{t,:,:}_S = \sum_{k=0}^{K} T_k\left(\widetilde{\boldsymbol{L}}_{t,:,:}\right) \mathcal{X}^{t,:,:}_S \boldsymbol{\Theta}_t \tag{18}$$

where $\widetilde{\boldsymbol{L}}_{t,:,:} = \frac{2\boldsymbol{L}_{t,:,:}}{\lambda_{\max}^{t}} - \boldsymbol{I}_N, \boldsymbol{L}_{t,:,:} = \boldsymbol{D}_{t,:,:} - \mathcal{W}_{t,:,:}, \boldsymbol{D}_{t,i,i} = \sum_j \mathcal{W}_{t,i,j}$ ($\lambda_{\max}^{t}$ is the largest eigenvalue of$\boldsymbol{L}_{t,:,:}$.

Temporal regression transformer with auto-correlation (auto-TRT)

As a regression modeling method for the temporal hyper-plane, Auto-TRT only adopts the Encoder structure of Transformer instead of the Encoder-Decoder structure of Transformer, which significantly decreases training costs. Meanwhile, we replace the original self-attention mechanism of Transformer with the Auto-Correlation attention mechanism presented in the literature [41] for efficient temporal weight representation and learning the periodic dependence of time series. The structure is shown in Fig. 2.

Why choose transformer for the regression in the temporal hyper-plane?

Here, we provide a brief derivation to demonstrate that Transformer has a similar architecture to GCN and GSTR. Since the multi-head attention mechanism is the core of Transformer, the other architectures are mainly designed for network gradient propagation and optimization. Therefore, this paper focuses on deriving the multi-head attention mechanism. Taking single-head attention as an example, assuming that the input $\mathcal{X}_T^{i,:,:} \in \mathbb{R}^{L\times\widetilde{C}}$ for the i_{th} spatial node in the temporal hyper-plane is retrieved as $\mathcal{X}_T^{i,:,:} \in \mathbb{R}^{L\times d_{model}}$ after the position encoding, then we have:

$$\begin{aligned} Attention &= softmax\left(\frac{\widetilde{\boldsymbol{X}}_T^{i,:,:}\boldsymbol{\theta}^Q\left(\boldsymbol{\theta}^K\right)^T\left(\widetilde{\boldsymbol{X}}_T^{i,:,:}\right)^T}{\sqrt{d_k}}\right)\widetilde{\boldsymbol{X}}_T^{i,:,:}\boldsymbol{\theta}^V \\ &= \mathcal{W}_T^{i,:,:}\widetilde{\boldsymbol{X}}_T^{i,:,:}\boldsymbol{\theta}^V \end{aligned} \tag{19}$$

where $\mathcal{W}_T^{i,:,:} \in \mathbb{R}^{L\times L}$ denotes the weight of attention, which reflects the correlation between the various time points of the sequence, i.e., the temporal weight in this paper (L is the length of the sequence), $\boldsymbol{\theta}^Q, \boldsymbol{\theta}^K$ and $\boldsymbol{\theta}^V$ are the parameters to be estimated. Obviously, Eqs. (3), (17) and (19) have a similar propagation mechanism, which explains why we chose Transformer Encoder to fit the temporal regression for panel models. The GSTR, GCN, and Transformer can be precisely stringed together by the decoupled spatial–temporal regression presented in this research.

Auto-correlation attention mechanism

Long time series has a relationship between time points that is based on their temporal proximity, such as the distance between them, as well as their periodicity and seasonality, which shows the auto-correlation nature of time series. Specifically, Auto-Correlation attention mechanism captures cycle-based dependence by calculating the autocorrelation of sequences, and similar subsequences are aggregated by time delay aggregation. The structure is shown in Fig. 5. For simplicity, the embedding $\mathcal{X}_T^{i,:,:} \in \mathbb{R}^{L\times d_{model}}$ of the i_{th} spatial node in the temporal hyper-plane is written as $\mathcal{X}_T^{i}$. For the single-head attention and the series $\mathcal{X}_T^{i}$ with the length L, after the projector, we get query $\mathcal{Q}^i$, key $\mathcal{K}^i$ and value $\mathcal{V}^i$. The formula of Auto-Correlation attention for the i_{th} spatial node is as follows:

$$\begin{aligned} \tau_1^i, \cdots, \tau_k^i &= \underset{\tau\in\{1,\cdots,L\}}{\text{argTopk}}\left(\mathcal{R}_{\mathcal{Q}^i,\mathcal{K}^i}(\tau)\right) \\ &\mathcal{R}_{\mathcal{Q}^i,\mathcal{K}^i}\left(\tau_1^i\right), \cdots, \mathcal{R}_{\mathcal{Q}^i,\mathcal{K}^i}\left(\tau_k^i\right) \\ &= softmax\left(\mathcal{R}_{\mathcal{Q}^i,\mathcal{K}^i}\left(\tau_1^i\right), \cdots, \mathcal{R}_{\mathcal{Q}^i,\mathcal{K}^i}\left(\tau_1^i\right)\right) \\ &Auto-Correlation\left(\mathcal{Q}^i, \mathcal{K}^i, \mathcal{V}^i\right) \\ &= \sum_{j=1}^{k}\text{Roll}\left(\mathcal{V}^i, \tau_j^i\right)\mathcal{R}_{\mathcal{Q}^i,\mathcal{K}^i}\left(\tau_j^i\right) \end{aligned} \tag{20}$$

where $\mathcal{R}_{\mathcal{Q}^i,\mathcal{K}^i}(\tau)$ represents the autocorrelation between series $\mathcal{Q}^i$ and $\mathcal{K}^i$, which is calculated by Fast Fourier Transform, $\mathcal{R}_{\boldsymbol{XX}}(\tau) = F^{-1}(S_{\boldsymbol{XX}}(f)) = \int_{-\infty}^{+\infty} S_{\boldsymbol{XX}}(f)e^{2\pi i\tau f}df,\ S_{\boldsymbol{XX}}(f) = F(\boldsymbol{X})F^{*}(\boldsymbol{X}) = \int_{-\infty}^{+\infty}\boldsymbol{X}e^{-2\pi itf}dt\overline{\int_{-\infty}^{+\infty}\boldsymbol{X}e^{-2\pi itf}dt}$, $*$ represents the conjugate.

argTopk$(\cdot)$ is to obtain the arguments of the top k autocorrelations and $k = c\log L$, c is a hyper-parameter. Roll$(\boldsymbol{X}, \tau)$ denotes the operation of delaying the sequence $\boldsymbol{X}$ by time delay τ. Specifically, the values at the first τ positions at the head of the original sequence are shifted at the last position, as shown in Fig. 5.

For the multi-head attention of the i_{th} spatial node with h heads, the query, key, and values for the j_{th} head are $\boldsymbol{Q}_j^i$,$\boldsymbol{K}_j^i$, $j \in \{1, \ldots, h\}$. We have:

$$\begin{aligned} M\text{H}^i &= Concat\left(\text{head}_1^i, \cdots, \text{head}_h^i\right)\boldsymbol{W}_O^i \\ head_j^i &= Auto-Correlation\left(\boldsymbol{Q}_j^i, \boldsymbol{K}_j^i, \boldsymbol{V}_j^i\right) \end{aligned} \tag{21}$$

It should be noted that this paper is only based on the Transformer's Encoder architecture to introduce the Auto-Correlation attention mechanism to better express the temporal weight with serial autocorrelation, thus reflecting the rationality of the design of this paper's temporal regression. The suggested Auto-Correlation attention mechanism's inventor believes that by applying the Fast Fourier Transform, the original attention computational cost $\mathcal{O}\left(L^2\right)$ can be reduced to $\mathcal{O}(L\log L)$. Thus, for all the spatial

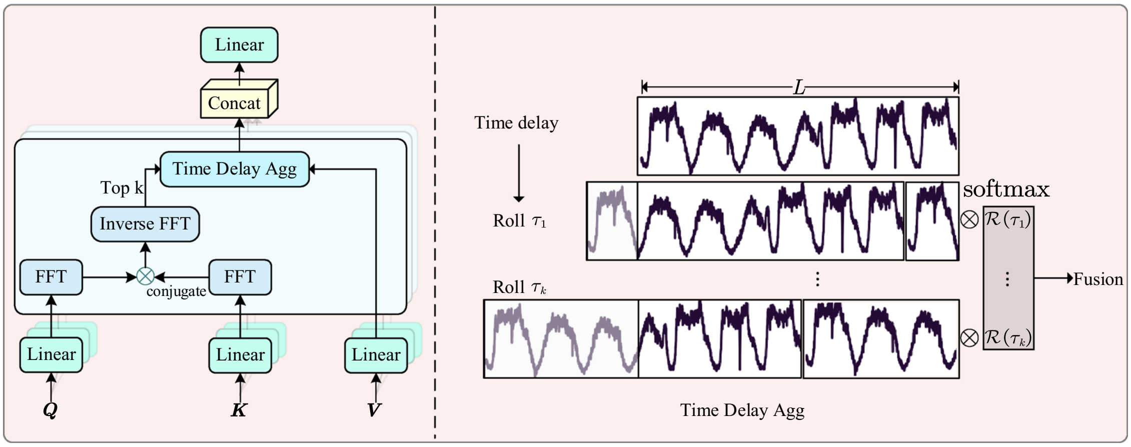


Fig. 5 The structure of auto-correlation attention mechanism. The autocorrelation $\mathcal{R}(\tau)$ is calculated by Fast Fourier Transform, which reflects the time-delay similarities. Then, based on the selected delay τ, related sub-processes are rolled to the same index and aggregated by $\mathcal{R}(\tau)$

nodes, the computational complexity of Auto-Correlation is $\mathcal{O}(NL\log L)$ in this paper.

Input and output of auto-TRT

We adopt the Encoder structure of Transformer as the temporal regression modeling and replace self-attention with Auto-Correlation to capture the period dependence and obtain the temporal weight with autocorrelation, which achieves lower computational complexity and maximally matches the decoupled GSTR (Eq. (9)). As shown in Fig. 2, assuming Auto-TRT has $\mathcal{N}$ encoder layers. The input of Auto-TRT, after positional encoding, we get $\mathcal{X}_T \in \mathbb{R}^{N\times L\times d_{\text{model}}}$ as the input of Auto-TRT's Encoder, and the embedding of the i_{th} spatial node is $\mathcal{X}_T^{i,:} \in \mathbb{R}^{L\times d_{\text{model}}}$ (to simplify the description, we use $\mathcal{X}$ to represent it). The output of the l_{th} encoder can be written as $\mathcal{X}_{en}^{l} = \text{Encoder}\left(\mathcal{X}_{en}^{l-1}\right)$, specific details are as follows:

$$\mathcal{Y}_{en}^{l,1} = LayerNorm\left(\text{Auto} - \text{Correlation}\left(\mathcal{X}_{en}^{l-1}\right) + \mathcal{X}_{en}^{l-1}\right)$$
$$\mathcal{Y}_{en}^{l,2} = LayerNorm\left(\text{FeedForward}\left(\mathcal{Y}_{en}^{l,1}\right) + \mathcal{Y}_{en}^{l,1}\right) \quad (22)$$

where $\mathcal{X}_{en}^{l} = \mathcal{Y}_{en}^{l,2}$ represents the output of the l_{th} encoder, $l = 1, \cdots, \mathcal{N}$, $\mathcal{X}_{en}^{0} = \mathcal{X}$ is the input of the Auto-TRT. LayerNorm denotes residual connectivity and layer normalization. FeedForward is the feed-forward neural network, which consists of two fully connected and one ReLU activation layer between. In particular, for the one encoder layer, we have:

$$\mathcal{Z}_T^{i,:,:} = LayerNorm\left(\text{Auto} - \text{Correlation}\left(\mathcal{X}_T^{i,:,:}\right) + \mathcal{X}_T^{i,:,:}\right)$$
$$\mathcal{Y}_T^{i,:,:} = LayerNorm\left(\text{FeedForward}\left(\mathcal{Z}_T^{i,:,:}\right) + \mathcal{Z}_T^{i,:,:}\right) \quad (23)$$

where $\mathcal{Y}_T^{i,:,:} \in \mathbb{R}^{L\times d_{\text{model}}}$ represents the output of Auto-TRT for the i_{th} spatial node in the temporal hyper-plane.

Training and optimization

Unlike the traditional coupled spatial–temporal regression for training, GSTRGCT considers a novel decoupled mechanism to factorize the spatial–temporal graph based on tensor decomposition, which obtains a spatial representation tensor, a temporal representation tensor, and a spatial–temporal tensor for the regression result. Therefore, it needs to optimize three objectives: spatial–temporal regression loss, spatial regularization optimization loss, and temporal regularization optimization loss. Therefore, the total loss function for the spatial–temporal regression task is formulated as:

$$\mathcal{L}_{total} = ||\mathcal{Y}_{ST} - \mathcal{Y}_{ST}||_F^2 + \lambda_S g_S(\mathcal{Y}_S, \mathbf{\Phi}_S) + \lambda_T g_T(\mathcal{Y}_T, \mathbf{\Phi}_T) \quad (24)$$

where $\mathcal{Y}_{ST} \in \mathbb{R}^{\tau_p\times N\times F}$ denotes the regression result, $||\cdot||_k$ represents the Frobenius $k-$ norm, g_S and g_T are the spatial regularization function and the temporal regularization function, respectively. In this paper, we choose the L1 regularization for g_S and g_T. λ_S and λ_T are regularity coefficients for the spatial regression and temporal regression, respectively. F equals 2.

Experiments and results

To validate the predictive regression capability of our proposed novel deep learning paradigm GSTRGCT based on panel modeling, we present our experiments on two large real-world datasets to evaluate the prediction performance and interpretability of our model for traffic forecasting. We briefly introduce the migration motivation of GSTRGCT for choosing traffic flow prediction.

Table 1 Statistics of datasets

Type	Dataset	# Node	# Edge	# Time step
Total flow				
Average occupancy	PEMS04	307	680	16,992
Average speed	PEMS08	170	590	17,856

Source codes are available at https://github.com/CQULangXiong/GSTRGCT. Official codes will be made public on GitHub once accepted.

Experimental settings

Datasets

We conducted experiments on two commonly used real-world large-scale datasets, PEMS04 and PEMS08, which have tens of thousands of timesteps and hundreds of sensors. The statistical information is summarized in Table 1. The datasets are collected by the Caltrans Performance Measurement System (PeMS) [43] in real-time every 30 s. The traffic data is aggregated at every 5-min interval from the raw data. Geographic information about the sensor stations is recorded in the datasets. There are three kinds of traffic measurements considered in our experiments, including total flow, average speed, and average occupancy. The total number of time slices for PeMS04 and PeMS08 are 16,992 and 17,856, respectively.

Preprocessing

All spatial–temporal data are transformed by z-score normalization $X\prime = \frac{X - \mathrm{mean}(X)}{\mathrm{std}(X)}$, where mean(·) and std(·) are the mean and the standard deviation of the time series. Each detector has 12 data points per hour and three traffic measurements. We use one hour as the historical time window and all traffic measurements as the input channels, so 12 observed data points ($T = L = 12$) are used to predict the traffic flow in the next 15 min ($T_p = 3$). Then, the adjacency matrix of the road graph is computed using the distance and topological spatial connection among the spatial nodes. Here, we use the connection distance between detectors as the prior input information for the spatial weight neural network (SWNN).

Implementation details

All experiments are compiled and tested on Ubuntu 20.04.5 LTS (CPU: 11th Gen Inter® Core™ i7-11700F @ 2.50GHz, GPU: NVIDIA GeForce RTX 3080). We implement all models based on the Pytorch framework and Python 3.9. In the training process, we set the batch size as 32 and the learning rate as 0.001 for all models. Meanwhile, we use the AdamW optimizer [44] to minimize it and adopt the mean square error (MSE) as the loss function. The maximum training epoch is set to 100 and an early stopping strategy is employed to avoid the overfitting problem. Performance on the validation set determines the hyper-parameters. Table 2 provides the setting values of the optimized hyper-parameters for the different components of GSTRGCT. SWNN is like a feed-forward neural network with one hidden layer (dimension 64).

Table 2 Hyper-parameters setting

Source	Params	Description	Value
Spatial hyper-plane (SRGCN)	E	Embedding dimension of spatial nodes	5
	α	The weighting factor for prior spatial weight	0.5
	K	Order of Chebyshev polynomials	2
Temporal hyper-plane (auto-TRT)	d_{model}	Embedding dimension in Auto-Correlation	64
	h	Number of heads in multi-head attention	8
	$\mathcal{N}$	Number of layers for Auto-TRT encoder	1
	c	For computing top k autocorrelation	5
Loss	λ_S	Regularity coefficient in spatial regression	0.0
	λ_T	Regularity coefficient in temporal regression	0.0

Baselines

We compare 3 kinds of methods with 14 baselines, including Simple baseline, Statistic models, and Deep models. For the multivariate-spatial–temporal setting, we select the GCN-based models: STGCN [22], ASTGCN [23], AGCRN [24], D2STGNN [45]; Transformer-based models: STTN [42], Informer [41], and Autoformer [42]. For the multivariate-temporal setting, we choose RNN-based models: GRU [35], LSTM [36]. For the traditional regression setting, panel-based models: ME, FE, and RE [46]; time-series-based models: ARIMA [10].

Training settings

For a fair comparison, we follow the dataset division into uniforms. For PeMS04 and PeMS08, we use about 60% of the data for training, 20% of the data for validation, and 20% of the data for testing. During the training process, the training set is shuffled, and the orders of the validation set and test set do not change during the evaluation and prediction.

Comparison and result analysis

Evaluation

To more intuitively understand the effect of the above models in practical application and correctly evaluate the performance of the proposed model, some commonly used indicators to evaluate the prediction ability of the models are selected in this article, including the root mean square error (RMSE), and mean absolute error (MAE). The definitions of these two performance metrics are as follows:

(1) Root mean square error (RMSE)

$$RMSE = \sqrt{\frac{1}{NT_p}\sum_{s=1}^{N}\sum_{t=1}^{T_p}(\mathcal{Y}_{st} - \mathcal{Y}_{st})^2} \tag{25}$$

(2) Mean absolute error (MAE)

$$MAE = \frac{1}{NT_p}\sum_{s=1}^{N}\sum_{t=1}^{T_p}|\mathcal{Y}_{st} - \mathcal{Y}_{st}| \tag{26}$$

(3) R-Square (R2)

$$R^2 = 1 - \frac{\sum_{s=1}^{N}\sum_{t=1}^{T}(\mathcal{Y}_{st} - \mathcal{Y}_{st})^2}{\sum_{s=1}^{N}\sum_{t=1}^{T}(\mathcal{Y}_{st} - \mathcal{Y}_{st})^2} \tag{27}$$

where $\mathcal{Y}_{st}$, $\mathcal{Y}_{st}$, and $\mathcal{Y}_{st}$ are the spatial–temporal regression target value, the spatial–temporal regression prediction value, and the mean value of the spatial–temporal regression target. T_p is the number of predicted time steps and N is the number of spatial nodes. RMSE is more sensitive to abnormal values, and MAE reflects prediction accuracy. R^2 measures the fit optimality of the regression model. The lower the values of RMSE and MAE, the better the predictive performance. The R^2 value is closer to 1, the better the model fitting performance.

Comparison results

Table 3 shows the average results of traffic flow prediction performance over the next 15 min. As shown in Table 3, our proposed model GSTRGCT consistently achieves the best performance in both datasets in terms of all evaluation metrics. Specifically, traditional methods including simple baseline and statistic models for traffic forecasting are usually ineffective, because they only work in short-term forecasting, smoother time series conditions, and simple spatial structures. Among deep learning models, the models of RNN classes including LSTM and GRU have large RMSE, probably due to their inability to capture the varying spatial structure of the traffic flow. Surprisingly, the performance of GSTRGCT surpasses the previously proposed spatial–temporal regression methods such as STGCN, ASTGCN, STTN, and AGCRN by 30–70%, which capture spatial–temporal interactions in a coupled processing paradigm. Compared to the decoupling framework D2STGNN, our proposed model still exhibits much better performance. Essentially, it is mostly because while D2STGNN decomposes the inputs, it still adopts the coupling method to extract spatial–temporal correlations, which may overfit toward noise in the training process, resulting in significant degradation in the fitting performance of the spatial–temporal regression. It is amazing that Transformers with improved attention mechanisms, such as Informer and Autoformer, achieve competitive results compared to other spatial–temporal regression models. This is because Informer and Autoformer employ time series analysis techniques such as series decomposition, which brings inductive bias and benefits the ability to capture temporal dependence. Overall, the outperformance of GSTRGCT may be attributed to the following aspects:

- We provide a decoupled spatial–temporal regression technique that can benefit computation efficiency and information utilization for spatial–temporal correlation.
- We combine the prior spatial structure information with posterior spatial embedding to construct the adaptive spatial weight to capture the dynamic spatial correlation. Meanwhile, the introduction of the Auto-Correlation attention mechanism better captures the period-based dependence and temporal correlation.
- The spatial–temporal regression model designed from traditional theories is more flexible and logical than the constructed framework design.

Comparison with different prediction steps

As shown in Table 3, four methods exhibit competitive results compared with our model GSTRGCT, which are ASTGCN, STGCN, Autoformer, and Informer. To further measure the predictive ability of the model, we show the results of the comparison of these four methods with different prediction intervals in Fig. 6. Existing GCN-RNN-based

Table 3 Traffic forecasting performance on the PEMS04 and PEMS08 datasets

Type	Model	PeMS04			PeMS08		
		RMSE	MAE	R^2	RMSE	MAE	R^2
Simple baseline	HA	128.89	104.70	0.336	112.13	89.40	0.413
Statistic models	ME	113.46	83.27	0.485	100.15	71.24	0.532
	FE	120.38	96.28	0.420	100.68	77.56	0.526
	RE	119.13	94.75	0.432	99.89	75.43	0.534
	ARIMA	213.72	163.41	–	167.54	121.68	–
Deep models	LSTM	125.07	81.29	0.374	95.44	75.08	0.575
	GRU	127.96	84.13	0.345	102.82	80.73	0.506
	STTN	123.75	100.03	0.060	104.35	82.24	0.110
	STGCN	69.85	47.66	0.760	59.02	39.69	0.806
	ASTGCN	64.48	46.35	0.777	56.64	41.32	0.802
	AGCRN	144.71	112.33	0.163	85.03	63.75	0.662
	D2STGNN	145.58	106.06	0.153	98.42	73.32	0.547
	Informer	59.93	40.63	0.751	52.41	36.82	0.738
	Autoformer	61.67	41.92	0.721	50.09	34.06	0.762
Ours	GSTRGCT	**46.24**	**30.30**	**0.903**	**39.49**	**27.24**	**0.905**

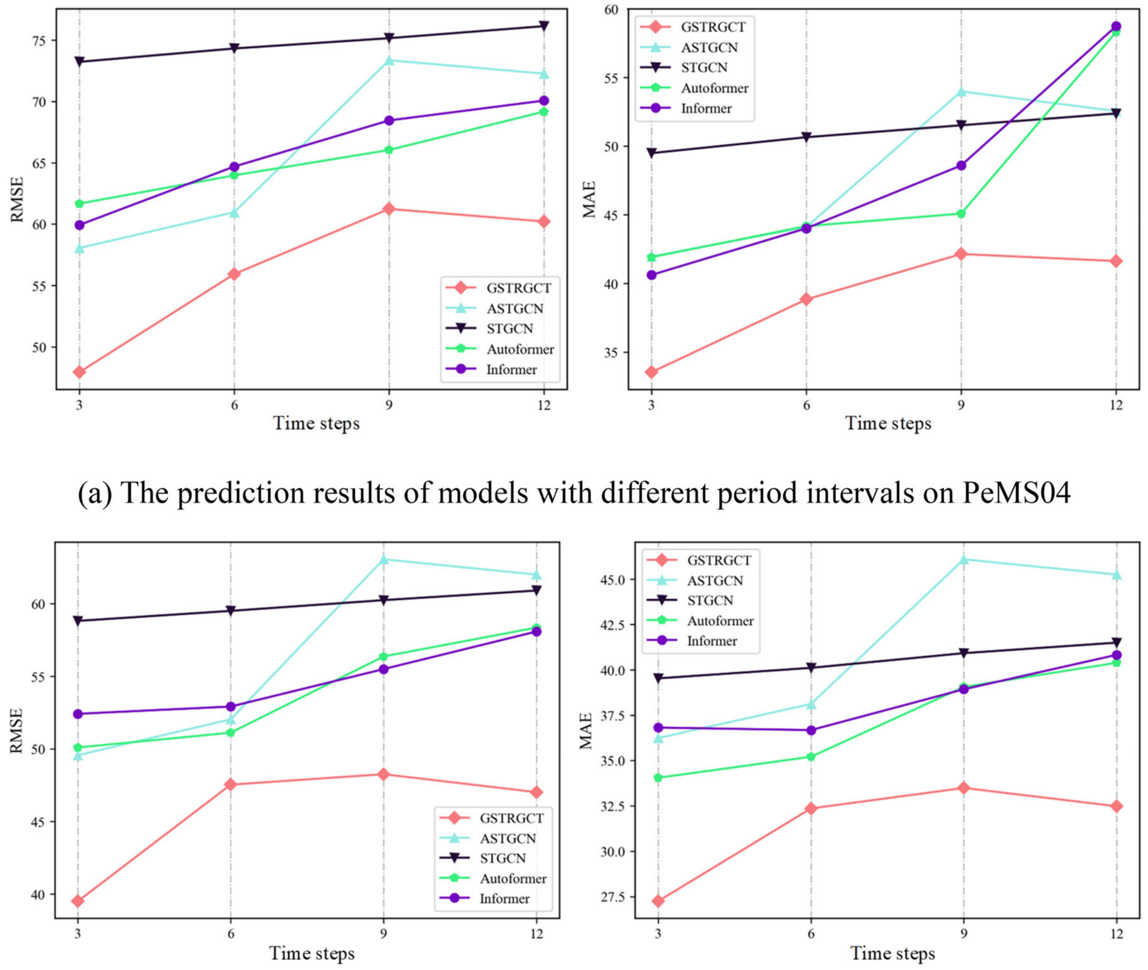
(ASTGCN and STGCN) and Transformer-based (Autoformer and Informer) models' performance deteriorates as the period interval increases. However, there are apparent differences in performance between the GCN-based models and Transformer-based models. In PeMS04, the MAE result of the Transformer-based models is highly volatile compared with GCN-based models, even though the Transformer-based models' RMSE is smaller relative. However, in PeMS08, the MAE results tend to be the opposite. This difference may be due to the more uniform spatial structure relationship of PeMS08 compared with PeMS04. In contrast, our proposed model GSTRGCT's performance in RMSE and MAE is less volatile and achieves the best prediction performance almost all the time compared with other models.

Dynamic adaptive spatial weight learning

To identify the ability of GSTRGCT to learn spatial weight adaptively and dynamically, Fig. 7 shows the visualization of the spatial weights at different moments on the test sets of PeMS04 and PeMS08 (note that to more clearly observe the dynamic variability of the spatial weights, the spatial weights for Eq. (13) at each moment in Fig. 7 are visualized without the activation function such as softmax). We can see that as the time interval increases, the spatial weight between nodes has a dynamic variation in the PeMS04 and PeMS08. On PeMS04, as shown in Fig. 7a, the circled area in red shows the obvious change from time interval 1 to time interval 12, which conducts new connectable spatial nodes. In particular, the differential results of spatial weights between time interval 1 and time interval 12 are shown in the 3rd subfigure. We can see that the diff spatial weight is mainly distributed in localized areas such as the bottom right corner where spatial nodes are more concentrated, which is consistent with the distribution of the initial adjacency matrix of PeMS04, which proves the reliability of the model to learn spatial weights dynamically. Similarly, we can see a similar dynamical spatial distribution of PeMS08 in Fig. 7b. In conclusion, the spatial weights at each moment show the global static and local dynamic changes, which is due to the combination of the priori spatial weight and the posterior spatial weight.

Ablation studies

To study the effectiveness of our proposed model GSTRGCT, we have conducted ablation studies by gradually excluding the creative components to observe how the performance degrades on datasets PeMS04 and PeMS08. First, we study the significance of tensor factorization for spatial–temporal regression. We directly apply the output of SRGCN as the input of Auto-TRT instead of using the original input to forecast without tensor factorization (or tensor decomposition). Second, we get rid of the Auto-Correlation attention mechanism of Auto-TRT in the temporal lane and use the Transformer's self-attention to replace it. Finally, to study the theoretical supportability as well as the necessity of generalized spatial–temporal regression (panel models), we directly ignore the spatial autocorrelation and various spatial and temporal effects, using the original data as the input of SRGCN. Figure 8 shows the performance results of all the considered ablation studies, "w/o *" represents that GSTRGCT doesn't adopt the component of *. From Fig. 8a,b, we



(a) The prediction results of models with different period intervals on PeMS04

(b) The prediction results of models with different period intervals on PeMS08

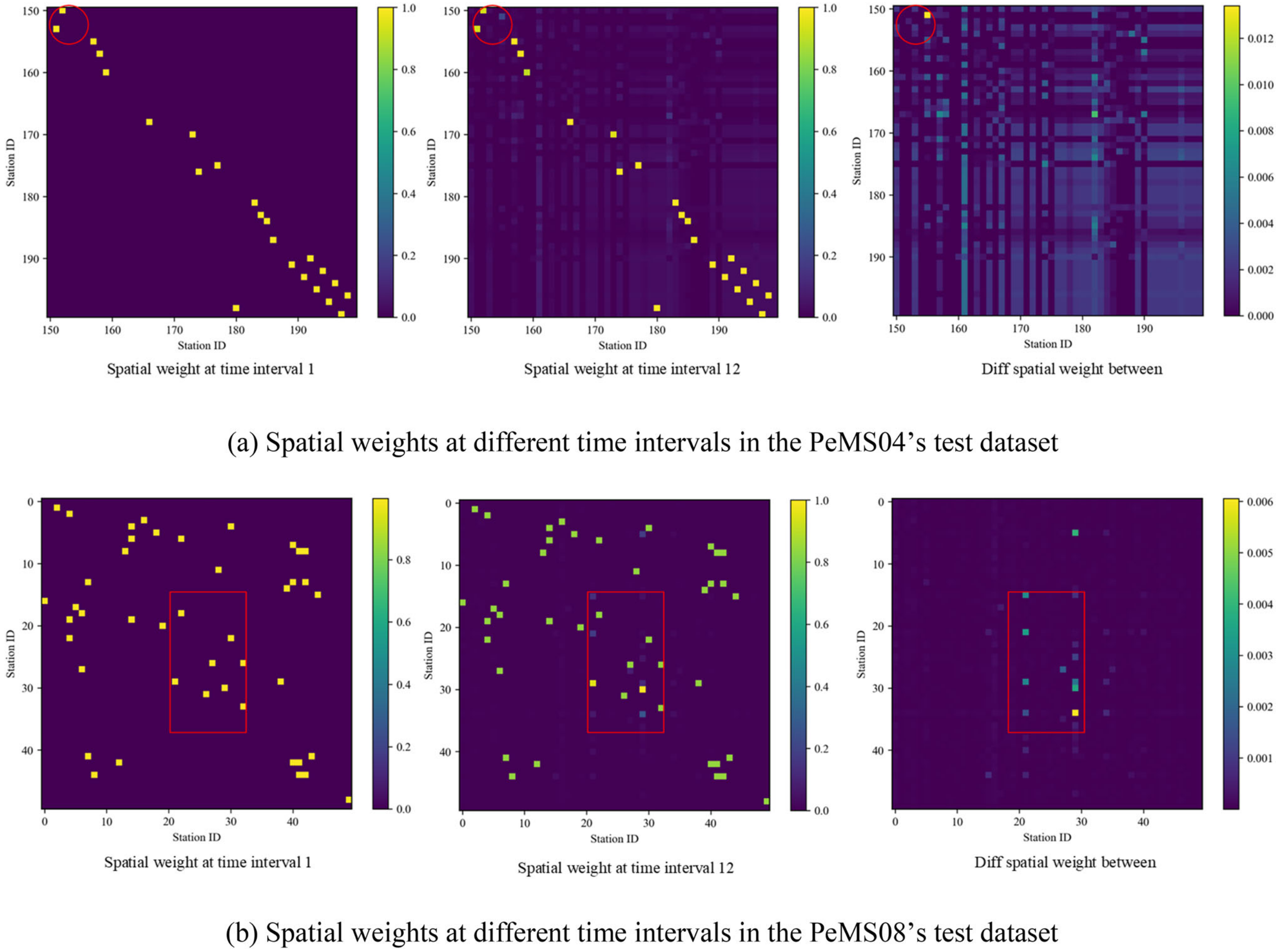
Fig. 6 Performance change of different methods as the prediction period interval increases. **a** The prediction results of models with different period intervals on PeMS04. **b** The prediction results of models with different period intervals on PeMS08

can conclude the following observations: (i) Our proposed model helps spatial regression obtain effective spatial dependences, helps temporal regression extract accurate temporal information, and helps spatial–temporal regression capture adaptive spatial–temporal correlation; (ii) There is a considerable gap between GSTRGCT and the variant without tensor factorization or Auto-Correlation. According to [47], tensor factorization is strictly based on autoregressive modeling. Tensor factorization without autocorrelation will greatly reduce its utility; iii) The deep learning framework based on generalized spatial–temporal regression (panel models) has theoretical support and interpretability, and its prediction effect is also appreciated. Moreover, these results again confirm that each component including generalized spatial–temporal regression, autocorrelation, and tensor factorization of our method is indispensable. In conclusion, our proposed deep learning paradigm-based panel models for transfer learning in traffic flow prediction are successful.

Other expriments and disscusions

Decoupling spatial–temporal regression is efficient?

In this subsection, we try to answer whether our proposed decoupled spatial–temporal regression method is efficient compared to the coupled spatial–temporal regression method. Therefore, we need to compare the coupled version of GSTRGCT with a serial connection between spatial regression and temporal regression and separate spatial/temporal regression. Meanwhile, we also compare other decoupled and coupled spatial–temporal regression models to validate the effectiveness of the decoupling method proposed in this paper. Figure 9 illustrates the difference in the process of feature dimension change between decoupled spatial–temporal regression and coupled spatial–temporal regression. As shown in Fig. 9, spatial–temporal decoupling can retain the spatial–temporal information of the original



(a) Spatial weights at different time intervals in the PeMS04's test dataset

(b) Spatial weights at different time intervals in the PeMS08's test dataset

Fig. 7 Dynamic spatial weight for the 50 sensors in different data. **a** Spatial weights at different time intervals in the PeMS04's test dataset. **b** Spatial weights at different time intervals in the PeMS08's test dataset

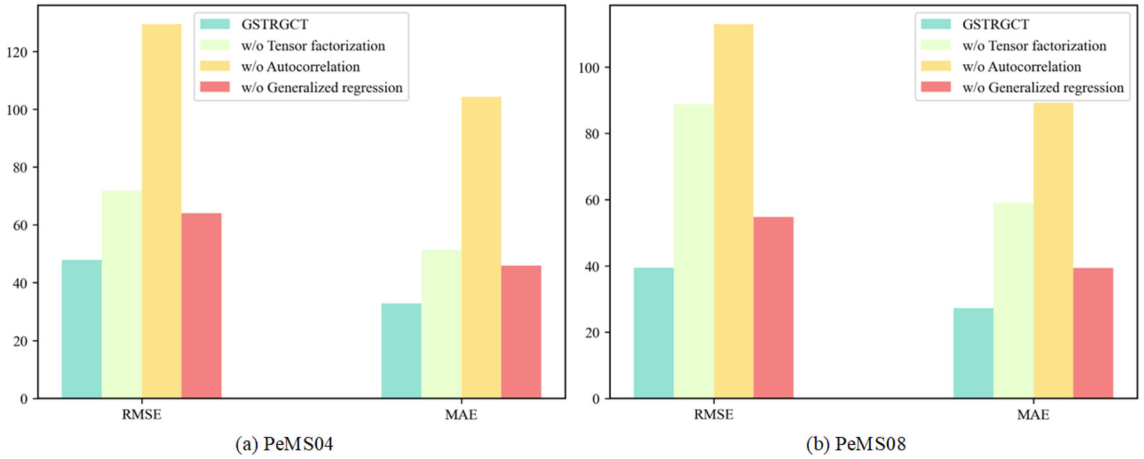


Fig. 8 Performance results of ablation studies in different datasets. **a** PeMS04. **b** PeMS08

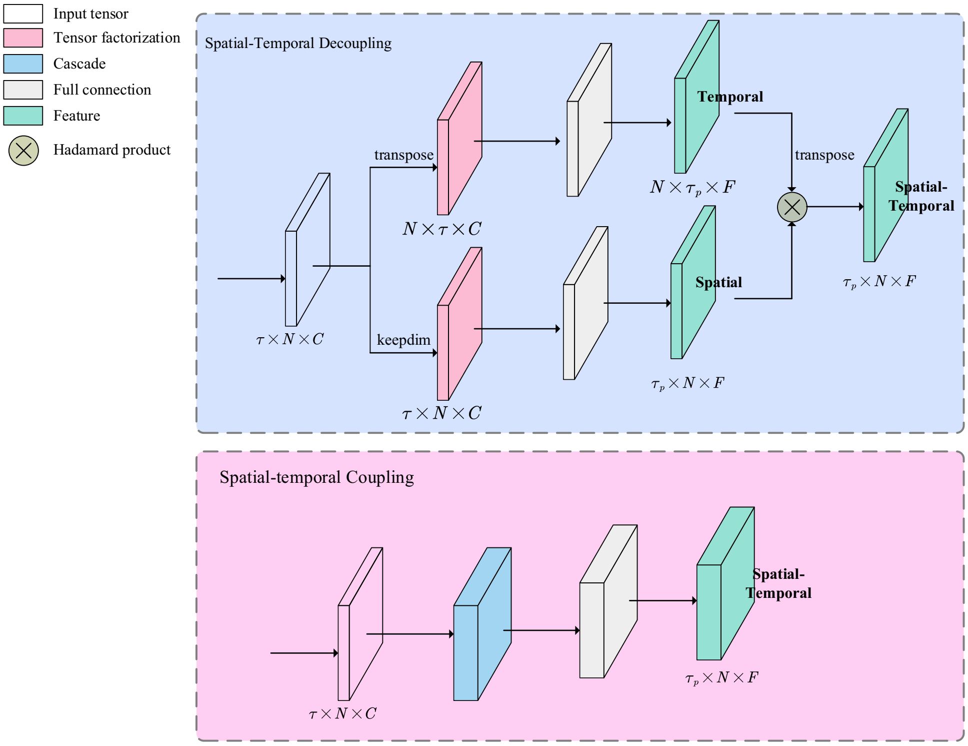


Fig. 9 Difference between decoupled and coupled spatial–temporal regression

data, while spatial–temporal coupling is prone to cause information misalignment due to modal mixing.

To this end, the coupled version of GSTRGCT, named GSTRGCT†, can be obtained from the following operations: (i) Firstly, we replace the dynamic adaptive spatial weight on the spatial hyper-plane with the original static adjacency matrix, i.e., $\mathcal{W}_{t,:,:} = \boldsymbol{A}$ in the Eq. (13), and keep the regression parameters of SRGCN shared across time steps, i.e., $\boldsymbol{\Theta}_t = \boldsymbol{\Theta}$; (ii) Secondly, we employ self-attention instead of auto-correlation attention on the temporal hyper-plane; (iii) Finally, the spatial hyper-plane remains serially connected to the temporal hyper-plane, i.e., $\mathcal{Y}_{ST} = \mathrm{MLP}(f_{T'}(\mathcal{W}_T, f_{S'}(\mathcal{W}_S, \mathcal{X}_S, \boldsymbol{\Phi}_S), \boldsymbol{\Phi}_T))$ in the Eq. (9). In addition, we remove the regression module on the spatial hyper-plane and the regression module on the temporal hyper-plane of GSTRGCT, named GSTRGCT-NS and GSTRGCT-NT, respectively. We select the two most representative spatial–temporal regression baselines, including the decoupled method (D2STGNN) and coupled method (STGNN, ASTGNN, AGCRN). For a fair comparison, we maintain consistency of experimental parameters, i.e., the batch size and the learning rate are uniformly set to 32 and 0.001, respectively. The parameters of the models maintain the optimal settings from the original paper.

Effectiveness To validate the effectiveness of the decoupling method, in Table 4, we compare the prediction performance for 3, 6, 9, and 12 future time steps with the same history time steps 12. From the experimental results, we can obtain the following findings. (i) GSTRGCT significantly outperforms GSTRGCT†, GSTRGCT-NS, GSTRGCT-NT, and baselines, which shows that spatial–temporal decomposition, DASWNN, and Auto-Correlation are crucial. (ii) The ablation versions GSTRGCT-NS and GSTRGCT-NT with the lowest number of parameters surpass the baselines, which demonstrates the rationality of the design for the spatial hyper-plane and temporal hyper-plane. (iii) The coupled version GSTRGCT† can still perform better than other spatial–temporal regression baselines, which indicates the effectiveness of the deep learning spatial–temporal regression paradigm designed based on the panel model theory.

Efficiency As shown in Fig. 10, on the one hand, compared to other spatial–temporal regression baselines, despite the increase in the number of covariates, GSTRGCT and its coupled version GSTRGCT† as well as its ablation versions GSTRGCT-NS and GSTRGCT-NT exhibit excellent performance in terms of fitting accuracy, and we believe it is worthwhile as it benefits the subsequent accurate solution of the spatial–temporal non-stationarity problem. On the other hand, compared to the coupled version GSTRGCT† and the decoupled spatial–temporal regression model D2STGNN, GSTRGCT has higher fitting performance with a shorter training time. This may be due to the fact that the decoupling framework decomposes spatial–temporal structural

Table 4 Comparison of decoupled and coupled spatial–temporal regression models on the PeMS08

T_p	Metric	STGCN	ASTGCN	AGCRN	D2STGNN	GSTRGCT-NS	GSTRGCT-NT	GSTRGCT†	GSTRGCT
3	RMSE	59.02	56.64	85.03	98.42	43.17	52.95	45.21	**39.49**
	MAE	39.69	41.32	63.75	73.33	28.82	37.88	31.16	**27.24**
	R^2	0.81	0.80	0.47	0.4196	0.90	0.85	0.89	**0.91**
6	RMSE	60.44	63.96	90.27	94.77	42.78	55.23	**42.11**	44.00
	MAE	40.98	48.94	67.63	72.23	28.91	39.29	**28.20**	29.78
	R^2	0.79	0.73	0.41	0.51	0.90	0.83	**0.91**	0.90
9	RMSE	61.58	69.85	107.48	95.51	46.03	55.78	46.09	**44.20**
	MAE	42.07	51.18	81.74	73.08	31.86	39.76	31.72	**30.03**
	R^2	0.78	0.76	0.02	0.42	0.88	0.83	0.89	**0.90**
12	RMSE	64.85	74.11	104.66	102.73	35.97	55.05	35.48	**34.23**
	MAE	45.02	56.24	80.46	77.32	24.76	39.56	24.09	**23.32**
	R^2	0.75	0.61	0.07	0.18	0.93	0.83	0.93	**0.94**

$R^2 \in [0, 1]$ represents the regression fitting accuracy. A larger R^2 and a smaller RMSE or MAE indicate a better prediction

relationships more rationally and has more efficient deductive capability.

What's the advantage of SRGCN for spatial analysis?

Although we confirmed the validity of SRGCN in the ablation experiments, we briefly describe some advantages of SRGCN over traditional spatial regression models. Due to the constraints of the article's length, the experimental details will be expanded in future work.

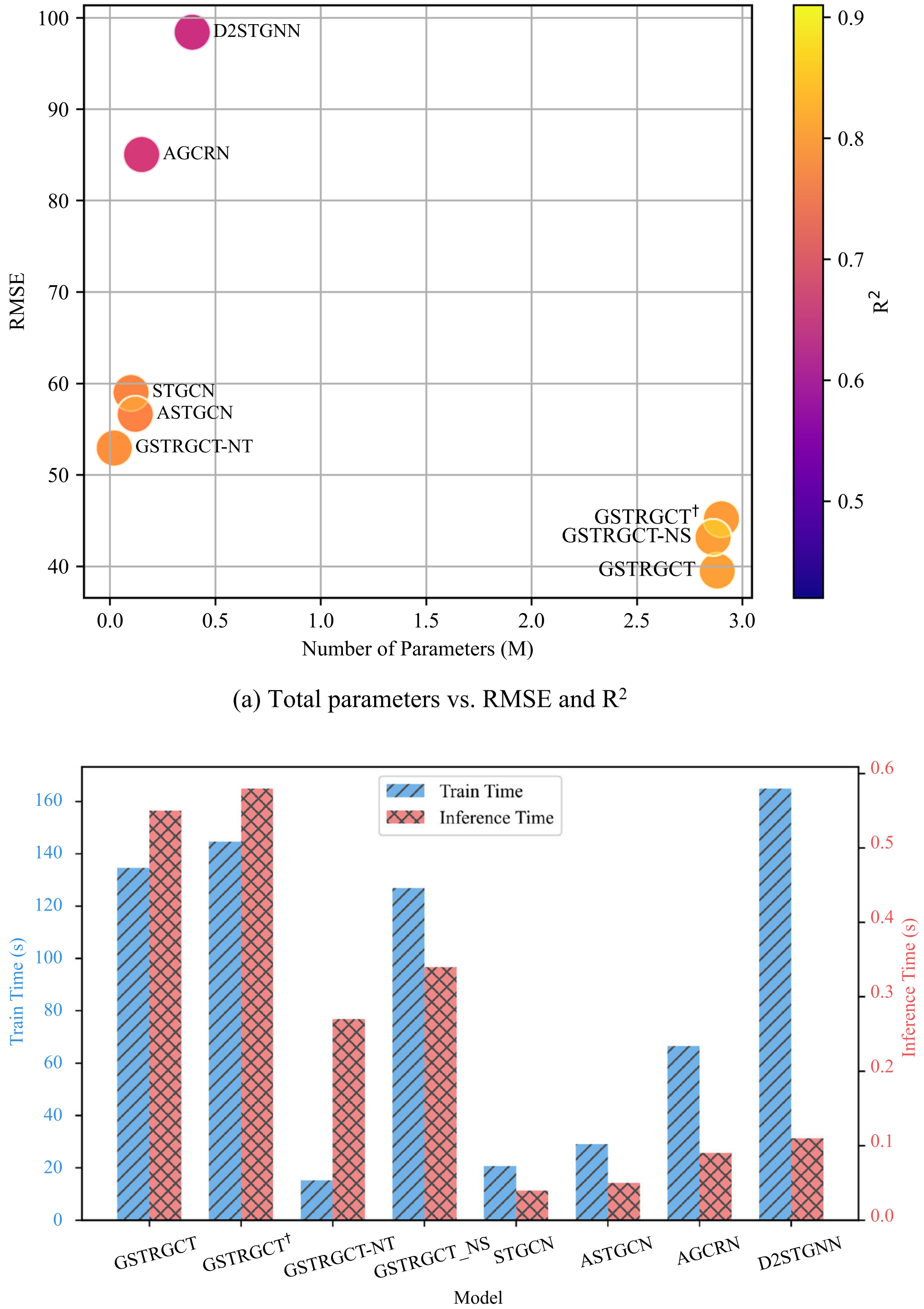
SRGCN vs. traditional spatial regression models In Table 5, we list the distinctions between SRGCN and several representative conventional spatial regression models. SLX, SAR, and SDM are the spatial lag of X model, the spatial autoregressive model, and the spatial durbin model, respectively, which describe the different spatial effects. Their expressions can be obtained from special forms in the GNS model:

$$\begin{aligned} Y &= \delta W Y + \alpha \iota_N + X\beta + W X\theta + \mu \\ u &= \lambda W u + \varepsilon \end{aligned} \tag{28}$$

Equation (1) provides a detailed explanation of each variable in the above equation without time t. Here, we give the special cases of GNS to represent SLX, SAR, and SDM. (i) SLX: $\lambda = 0$ and $\delta = 0$; (ii) SAR: $\theta = 0$ and $\lambda = 0$; (iii) SDM: $\lambda = 0$. GWR is geographically weighted regression, written as $\boldsymbol{Y}_i = \boldsymbol{X}_i\boldsymbol{\beta}(u_i, v_i) + \boldsymbol{\varepsilon}_i$, $\boldsymbol{\beta}(u_i, v_i)$ is related to geographic location. The parameter solution of GWR depends on the spatial weight, so it can also be written as $\boldsymbol{Y}_i = \boldsymbol{x}_i\boldsymbol{\beta}_i + \boldsymbol{W}^{(i)}\boldsymbol{X}\boldsymbol{\delta}_i + \boldsymbol{\varepsilon}_i$. In particular, we let the Chebyshev polynomial $K = 1, \lambda_{\max} = 2$, and $\theta = \theta_0 = -\theta_1$ in Eq. (15), we can obtain a simplified formulation for SRGCN with one layer: $\boldsymbol{Y} = \sigma(\boldsymbol{W}_L\boldsymbol{X}\boldsymbol{\Theta})$, $\boldsymbol{W}_L = \boldsymbol{I}_N + \boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{A}\boldsymbol{D}^{-\frac{1}{2}}$. Noting that $\boldsymbol{W}_L$ can be obtained by Eq. (13). Thus, it is learnable by DASWNN. SRGCN† uses the static adjacency matrix $\boldsymbol{A}$ as the spatial weight. We can summarize the benefits of SRGCN as follows: (i) In general, compared to traditional spatial regression models, SRGCN and its degenerate SRGCN† have the capability of extracting nonlinear features, which facilitates the accurate fitting of spatial regression. (ii) The spatial structure of SRGCN is adaptive, which is consistent with the complexity of spatial relationships in the real world. (iii) SRGCN can choose a derivable activation function such as Sigmoid to use ML to solve the parameters, which can reduce the training consumption in high-dimensional spatial regression analysis and make it interpretable.

Auto-correlation attention mechanism is effective for temporal regression?

To explore the effectiveness of Auto-Correlation, we compare GSTRGCT under different attention mechanisms for modeling series dependence. The inspection of the data in Table 6 reveals that Transformer Encoder for temporal regression with different attention mechanisms in GSTRGCT has excellent regression-fitting performance (R^2 is around 0.9), which may be due to the advantage of its global modeling relevance and further justifies the design of our framework. We can also summarize other findings as follows: (i) Auto-Correlation for temporal regression almost achieves the best performance under various input length (T or L) and output length (T_p) settings; (ii) Compared to Self-Attention, Auto-Correlation not only has lower computation complexity but also better performance, this may be due to the inherent periodicity and auto-correlation of predictable time series; (iii)



(a) Total parameters vs. RMSE and R^2

(b) Train and inference time comparison

Fig. 10 Comparison of practical accuracy and efficiency of GSTRGCT and other spatial–temporal regression models under $T = L = 12$ and $T_p = 3$ on the PeMS08. **a** Total parameters vs. RMSE and R^2. **b** Train and inference time comparison

Compared to ProbSparse Attention, even though it has the same computational complexity as Auto-Correlation, we find that as the length of the input sequence increases, its performance decreases compared to the original Self-Attention, probably due to its lack of ability to capture the periodic dependence of the time series. In conclusion, the above findings validate the effectiveness of Auto-Correlation for temporal regression in GSTRGCT.

Visualization

To further view and intuitively understand the prediction capability and the regression-fitting performance of the model, in this section, we randomly selected a sensor (50) to visualize the prediction results of GSTRGCT with other different spatial–temporal regression models in Fig. 11, and the regression-fitting result using scatter density plotting

Table 5 Differences between SRGCN and traditional spatial regression models

Model	Spatial weight $\boldsymbol{W}$	Lag effects	Estimation	Interpretability	Type
SLX	Unlearnable $M \times M$	$\boldsymbol{WX}$	OLS	Yes	Linear
SAR	Unlearnable $M \times M$	$\boldsymbol{WY}$	ML	Yes	Linear
SDM	Unlearnable $M \times M$	$\boldsymbol{WX}$ and $\boldsymbol{WY}$	ML	Yes	Linear
GWR	Unlearnable $M \times M \times M$	$\boldsymbol{WX}$	OLS	No	Linear
SRGCN†	Unlearnable $N \times N$	$\boldsymbol{W}_L\boldsymbol{X}$ and $\boldsymbol{W}_L\boldsymbol{Y}$	BP/ML	Uncertainty	Non-Linear
SRGCN	Learnable $N \times N$	$\boldsymbol{W}_L\boldsymbol{X}$ and $\boldsymbol{W}_L\boldsymbol{Y}$	BP/ML	Uncertainty	Non-Linear

M is the number of locations with observed values; *N* is the total number of locations in the study area
OLS ordinary least square, *ML* maximum likelihood, *BP* back propagation

Table 6 Auto-correlation vs. other attention mechanisms for temporal regression. auto-correlation in GSTRGCT is replaced by other attention mechanisms

$T\,(or\,L)$		12			24			36			Complexity
T_p		3	6	12	3	6	12	3	6	12	
Auto-correlation [41]	RMSE	**39.49**	44.00	**34.23**	**41.18**	**41.64**	**43.45**	**39.15**	**42.26**	**45.78**	$\mathcal{O}(NL\log L)$
	MAE	**27.24**	29.78	**23.32**	**27.53**	**27.77**	**29.78**	**26.56**	**28.38**	**31.60**	
	R^2	**0.91**	0.90	**0.94**	**0.91**	**0.91**	**0.90**	**0.92**	**0.91**	**0.89**	
Self-attention [48]	RMSE	44.18	45.29	38.10	41.58	43.51	45.01	40.66	42.72	45.93	$\mathcal{O}(NL^2)$
	MAE	29.71	30.99	26.43	27.82	29.58	31.02	27.90	28.85	31.90	
	R^2	0.90	0.89	0.92	0.91	0.90	0.90	0.91	0.91	0.89	
ProbSparse attention [40]	RMSE	42.98	**42.47**	34.80	42.48	43.91	46.56	41.31	43.94	46.45	$\mathcal{O}(NL\log L)$
	MAE	28.36	**28.02**	23.97	28.68	29.94	32.43	28.35	30.03	32.43	
	R^2	0.91	**0.91**	0.94	0.91	0.90	0.89	0.91	0.90	0.88	

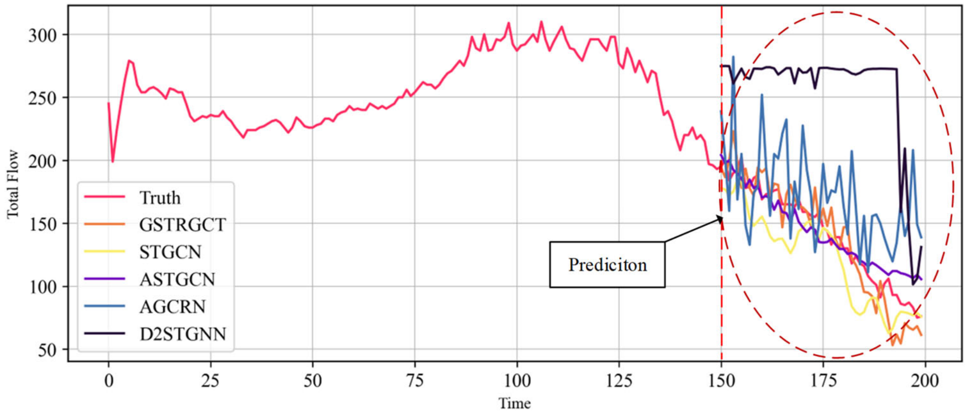


Fig. 11 Visualization of prediction on PeMS08 for different models (sensor 50)

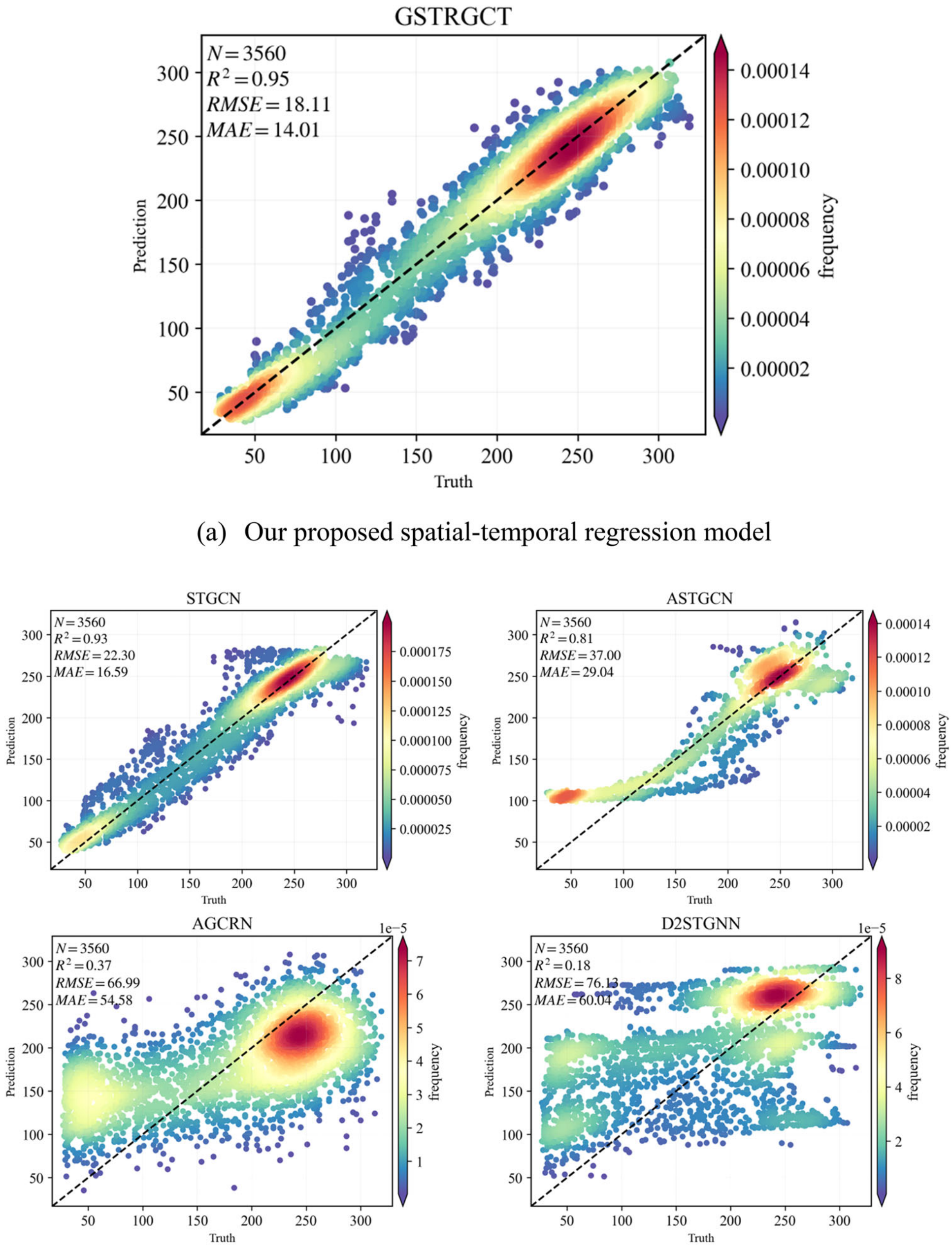
between truth and prediction in Fig. 12. As can be seen from Figs. 11 and 12, it is easy for GSTRGCT to capture the serial trend as well as the period dependence compared to other models; Our proposed model, GSTRGCT, can capture the nonlinear spatial–temporal correlation well, thereby achieving an accurate fit regardless of the high or low values.

Conclusion and future work

Conclusion

This work focuses on modeling and designing a unified spatial–temporal regression framework based on the theoretical

Fig. 12 Visualization of regression-fitting result (GSTRGCT vs. other models) (sensor 50). **a** Our proposed spatial–temporal regression model. **b** Other spatial–temporal regression models



(a) Our proposed spatial-temporal regression model

(b) Other spatial-temporal regression models

foundations of spatial statistics rather than designing complicated deep spatial–temporal regression networks with prior domain knowledge. In this paper, we first propose to decouple spatial–temporal regression into a tensor product between the regression on the spatial hyper-plane and temporal hyper-plane based on tensor decomposition and the theory of panel models, which allows modeling the relationships for different scales of spatial–temporal data more accurately and effectively with lower computational complexity, thus promising to improve the regression-fitting and prediction performance. Based on the decoupled spatial–temporal regression framework, the Generalized Spatial–Temporal Regression Graph Convolutional Transformer (GSTRGCT) is proposed by rationally designing the spatial and temporal regression as well as the dynamic adaptive spatial structure learning with prior and posterior spatial weight and period-based dependence learning according to the autocorrelation characteristic of time series. Specifically, on the spatial hyper-plane, the dynamic adaptive spatial weight network comprehensively exploits prior spatial information and posterior adaptive embedding techniques to adjust the spatial structure dynamically over time and adopts GCN with isomorphic spatial regression to capture spatial dependence for spatial regression analysis. On the temporal hyper-plane, the auto-correlation attention mechanism with lower computational complexity replaces the self-attention mechanism of the Transformer Encoder to model period-based dependence and characterize temporal weight for temporal regression. Our proposed model is migrated to the case of popular traffic forecasting. The extensive experiments on the two real-world datasets reveal that our proposal can yield consistent and significant state-of-the-art regression-fitting and prediction performance compared to the spatial–temporal regression baselines.

Future work

GSTRGCT is a quite simple-minded spatial–temporal regression framework and has the theoretical underpinnings of panel models. Thus, it can increase the interpretability by statistical diagnostics like the traditional spatial–temporal regression methods. However, we have not fully explored the effect of different tensor decomposition methods on GSTRGCT and how to further improve the computational efficiency of the algorithm under large-scale spatial cells, which is a topic that needs to be further investigated in the future. Additionally, how to utilize multimodal information to improve the solution accuracy of spatial–temporal non-stationary, and utilize GSTRGCT as a pre-trained spatial–temporal representation to improve the migration ability of spatial–temporal data for downstream tasks, we believe there is great potential for improved GSTRGCT.

Acknowledgements This study was supported by the Natural Science Foundation of Chongqing in China (Grant No. CSTB2023NSCQ-MSX0374, Grant No. CSTB2023NSCQ-LZX0048) and 23 Key Research Projects in Humanities and Social Sciences of Chongqing Municipal Education Commission (23SKGH251).

Data availability Our research data and code will be stored on GitHub for access and use by other researchers. The specific GitHub repository address is as follows: https://github.com/CQULangXiong/GSTRGCT. We believe that this open sharing of data and code will contribute to further research and validation within the scientific community.

Declarations

Conflict of interest We declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. Zheng G, Chai WK, Duanmu JL, Katos V (2023) Hybrid deep learning models for traffic prediction in large-scale road networks. Inf Fusion 92:93–114
2. Wang C, Tian R, Hu J, Ma Z (2023) A trend graph attention network for traffic prediction. Inf Sci 623:275–292
3. Wang Y, Ren Q, Li J (2023) Spatial–temporal multi-feature fusion network for long short-term traffic prediction. Expert Syst Appl 224:119959
4. Chaudhari K, Thakkar A (2023) Data fusion with factored quantization for stock trend prediction using neural networks. Inf Process Manage 60(3):103293
5. Chaudhari K, Thakkar A (2023) Neural network systems with an integrated coefficient of variation-based feature selection for stock price and trend prediction. Expert Syst Appl 219:119527
6. Cui C, Li X, Zhang C, Guan W, Wang M (2023) Temporal-relational hypergraph tri-attention networks for stock trend prediction. Pattern Recogn 143:109759
7. Bi K, Xie L, Zhang H, Chen X, Gu X, Tian Q (2023) Accurate medium-range global weather forecasting with 3D neural networks. Nature 619(7970):533–538
8. Wu H, Zhou H, Long M, Wang J (2023) Interpretable weather forecasting for worldwide stations with a unified deep model. Nat Mach Intell 5(6):602–611
9. Ma Y, Lou H, Yan M, Sun F, Li G (2024) Spatio-temporal fusion graph convolutional network for traffic flow forecasting. Inf Fusion 104:102196
10. Williams BM, Hoel LA (2003) Modeling and forecasting vehicular traffic flow as a seasonal ARIMA process: theoretical basis and empirical results. J Transp Eng 129(6):664–672
11. Lu Z, Zhou C, Wu J, Jiang H, Cui S (2016) Integrating granger causality and vector auto-regression for traffic prediction of large-scale WLANs. KSII Trans Internet Inf Syst (TIIS) 10(1):136–151
12. Ramchandra NR, Rajabhushanam C (2022) Machine learning algorithms performance evaluation in traffic flow prediction. Mater Today Proc 51:1046–1050
13. Dai G, Tang J, Luo W (2023) Short-term traffic flow prediction: an ensemble machine learning approach. Alex Eng J 74:467–480
14. Xu J, Song R, Wei H, Guo J, Zhou Y, Huang X (2021) A fast human action recognition network based on spatio-temporal features. Neurocomputing 441:350–358
15. Méndez M, Merayo MG, Núñez M (2023) Long-term traffic flow forecasting using a hybrid CNN-BiLSTM model. Eng Appl Artif Intell 121:106041
16. Hu X, Liu W, Huo H (2024) An intelligent network traffic prediction method based on Butterworth filter and CNN–LSTM. Comput Netw 240:110172
17. Wang L, Guo D, Wu H, Li K, Yu W (2024) TC-GCN: triple cross-attention and graph convolutional network for traffic forecasting. Inf Fusion 105:102229
18. Bao Y, Liu J, Shen Q, Cao Y, Ding W, Shi Q (2023) PKET-GCN: prior knowledge enhanced time-varying graph convolution network for traffic flow prediction. Inf Sci 634:359–381
19. Su L, Xiong L, Yang J (2023) Multi-Attn BLS: multi-head attention mechanism with broad learning system for chaotic time series prediction. Appl Soft Comput 132:109831

20. Ji W, Chung AC (2023) Unsupervised domain adaptation for medical image segmentation using transformer with meta attention. IEEE Trans Med Imaging
21. Trisedya BD, Qi J, Zheng H, Salim FD, Zhang R (2023) TransCP: a transformer pointer network for generic entity description generation with explicit content-planning. IEEE Trans Knowl Data Eng
22. Yu B, Yin H, Zhu Z (2017) Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. arXiv preprint arXiv:1709.04875
23. Guo S, Lin Y, Feng N, Song C, Wan H (2019) Attention based spatial-temporal graph convolutional networks for traffic flow forecasting. In: Proceedings of the AAAI conference on artificial intelligence, vol 33, no. 01, pp 922–929.
24. Bai L, Yao L, Li C, Wang X, Wang C (2020) Adaptive graph convolutional recurrent network for traffic forecasting. Adv Neural Inf Process Syst 33:17804–17815
25. Anselin L (2013) Spatial econometrics: methods and models (Vol. 4). Springer
26. Halaby CN (2004) Panel models in sociological research: theory into practice. Annu Rev Sociol 30:507–544
27. Zhai P, Yang Y, Zhang C (2023) Causality-based CTR prediction using graph neural networks. Inf Process Manage 60(1):103137
28. Waikhom L, Singh Y, Patgiri R (2023) PO-GNN: position-observant inductive graph neural networks for position-based prediction. Inf Process Manage 60(3):103333
29. Bai L, Cui L, Jiao Y, Rossi L, Hancock ER (2020) Learning backtrackless aligned-spatial graph convolutional networks for graph classification. IEEE Trans Pattern Anal Mach Intell 44(2):783–798
30. Defferrard M, Bresson X, Vandergheynst P (2016) Convolutional neural networks on graphs with fast localized spectral filtering. Adv Neural Inf Process Syst 29:1
31. Geng Z, Xu J, Wu R, Zhao C, Wang J, Li Y, Zhang C (2024) STGAFormer: spatial–temporal gated attention transformer based graph neural network for traffic flow forecasting. Inf. Fusion 105:102228
32. Ma X, Li X, Feng W, Fang L, Zhang C (2023) Dynamic graph construction via motif detection for stock prediction. Inf Process Manage 60(6):103480
33. Jin J, Song Y, Kan D, Zhang B, Lyu Y, Zhang J, Lu H (2024) Learning context-aware region similarity with effective spatial normalization over Point-of-Interest data. Inf Process Manage 61(3):103673
34. Ma X, Tao Z, Wang Y, Yu H, Wang Y (2015) Long short-term memory neural network for traffic speed prediction using remote microwave sensor data. Transp Res Part C Emerg Technol 54:187–197
35. Chung J, Gulcehre C, Cho K, Bengio Y (2014) Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv:1412.3555. In: NIPS 2014 Deep Learning and Representation Learning Workshop
36. Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural Comput 9(8):1735–1780
37. Devlin J, Chang MW, Lee K, Toutanova K (2018) Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv:1810.04805
38. Brown T, Mann B, Ryder N, Subbiah M, Kaplan JD et al (2020) Language models are few-shot learners. Adv Neural Inf Process Syst 33:1877–1901
39. Dosovitskiy A, Beyer L, Kolesnikov A, Weissenborn D et al (2020) An image is worth 16x16 words: transformers for image recognition at scale. arXiv:2010.11929
40. Zhou H, Zhang S, Peng J, Zhang S, Li J, Jong H, Zhang W (2021) Informer: beyond efficient transformer for long sequence time-series forecasting. In: Proceedings of the AAAI conference on artificial intelligence, vol 35, no 12, pp 11106–11115
41. Wu H, Xu J, Wang J, Long M (2021) Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. Adv Neural Inf Process Syst 34:22419–22430
42. Xu M, Dai W, Liu C et al (2022) Spatial-temporal transformer networks for traffic flow forecasting. arXiv:2001.02908.
43. Chen C, Petty K, Skabardonis A, Varaiya P, Jia Z (2001) Freeway performance measurement system: mining loop detector data. Transp Res Rec 1748(1):96–102
44. Loshchilov I, Hutter F (2017) Decoupled weight decay regularization. arXiv:1711.05101
45. Shao Z, Zhang Z, Wei W, Wang F, Xu Y, Cao X, Jensen CS (2022) Decoupled dynamic spatial-temporal graph neural network for traffic forecasting. arXiv:2206.09112
46. Baltagi BH, Baltagi BH (2008) Econometric analysis of panel data, vol 4. Wiley:Chichester, pp 135–145.
47. Takeuchi K, Kashima H, Ueda N (2017) Autoregressive tensor factorization for spatio-temporal predictions. In: 2017 IEEE international conference on data mining (ICDM), pp. 1105–1110
48. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Polosukhin I (2017) Attention is all you need. Adv Neural Inf Process Syst 30:1

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.