Geoinformatica (2022) 26:645–676
https://doi.org/10.1007/s10707-021-00454-x

Spatial regression graph convolutional neural networks: A deep learning paradigm for spatial multivariate distributions

Di Zhu[1] · **Yu Liu**[2] · **Xin Yao**[3] · **Manfred M. Fischer**[4]

Received: 21 May 2021 / Revised: 8 October 2021 / Accepted: 13 October 2021 /
Published online: 2 November 2021

Abstract

Geospatial artificial intelligence (GeoAI) has emerged as a subfield of GIScience that uses artificial intelligence approaches and machine learning techniques for geographic knowledge discovery. The non-regularity of data structures has recently led to different variants of graph neural networks in the field of computer science, with graph convolutional neural networks being one of the most prominent that operate on non-euclidean structured data where the numbers of nodes connections vary and the nodes are unordered. These networks use graph convolution – commonly known as filters or kernels – in place of general matrix multiplication in at least one of their layers. This paper suggests spatial regression graph convolutional neural networks (SRGCNNs) as a deep learning paradigm that is capable of handling a wide range of geographical tasks where multivariate spatial data needs modeling and prediction. The feasibility of SRGCNNs lies in the feature propagation mechanisms, the spatial locality nature, and a semi-supervised training strategy. In the experiments, this paper demonstrates the operation of SRGCNNs with social media check-in data in Beijing and house price data in San Diego. The results indicate that a well-trained SRGCNN model is capable of learning from samples and performing reasonable predictions for unobserved locations. The paper also presents the effectiveness of incorporating the idea of geographically weighted regression for handling heterogeneity between locations in the model approach. Compared to conventional spatial regression approaches, SRGCNN-based models tend to generate much more accurate and stable results, especially when the sampling ratio is low. This study offers to bridge the methodological gap between graph deep learning and spatial regression analytics. The proposed idea serves as an example to illustrate how spatial analytics can be combined with state-of-the-art deep learning models, and to enlighten future research at the front of GeoAI.

Keywords Spatial regression · Graph convolutional neural networks · Deep learning · GeoAI · Social sensing

✉ Di Zhu
dizhu@umn.edu

Extended author information available on the last page of the article.

1 Introduction

As a long-established spatial analytical method, spatial regression grows in the fields of regional science and spatial econometrics [1], where the applied works rely heavily on observed variables with reference to location measures. Spatial regression models focus on two critical aspects of data introduced by locations: 1) the spatial autocorrelation (dependence) of samples and 2) the spatial heterogeneity of the relationships been modelled [2]. In a cross-sectional setting, i.e., leaving out of consideration the temporal aspect of data, a typical spatial regression analysis is performed mainly in four steps. First, representing the structure of spatial dependence using a spatial contiguity or weights matrix [3]. Second, specifying an econometric model that incorporates the potential spatial effects [4]. Third, estimating parameters in the model. Fourth, utilizing the fitted model for spatial prediction [5]. This classical econometric paradigm [6] has been widely adopted in exploratory spatial data analysis (ESDA) where the spatial multivariate (cross-sectional) distribution data is available and when we seek to capture the relationships between the observations of variables [7].

Recently, the emergence of big geo-data that incorporates fine-resolution spatiotemporal information has made it possible to characterize a variety of socioeconomic attributes in the geographical space [8, 9]. Nevertheless, the spatial multivariate distribution data for a study area of interest can be incomplete after the space-time slicing [10]. For instance, there could be missing values in certain dimensions of the attributes at a certain time [11]. In scenarios where the spatial cross-sectional data is high-dimension and yet incomplete, one of the applications of spatial regression models would be to consider the spatial relationships between the independent and/or dependent variables observed at nearby neighbors in predicting the missing values of the dependent variable at specific locations [12].

Current endeavors, however, are somehow constrained by the paradigm of spatial econometrics. Applications are limited by prior assumptions in econometrics, such as the linearity of regression model and the normality of data distributions [6, 12]. Traditional spatial regression models deal with the formal mathematical expression of spatial effects through predetermined functional forms, linear combinations of variables often the case, while the non-linear nature of spatial relationships has been overlooked [13]. Another drawback is that the ad hoc spatial dependence structure incorporated in the spatial weights matrix can only be defined among the locations where both independent and dependent variables are observed, without considering the other locations where the dependent variable is unobserved. This prerequisite enforces the specification of spatial relationships to be within the observed location set, which could result in model overfitting [14] and is not an ideal solution for transductive spatial prediction under missing data [15].

Therefore, it is crucial to adopt new computation frameworks as a supplement to spatial econometrics, such that spatial regression analysis can be implemented in a more flexible manner and with a higher data capacity. With the recent progress in computing techniques and the availability of high-quality geospatial data, deep learning (DL) methods have been increasingly used to model spatial processes [16]. More specifically, geospatial artificial intelligence (GeoAI) has emerged as a subfield of GIScience that utilizes AI approaches for geographic knowledge discovery [17]. As there exist challenges in adapting deep learning models into spatial analysis, applied GeoAI research so far has paid more attention to the classification of spatial features, based on mature DL architectures in image classification and object detection [18–21]. Enlightened by these applications, researchers started to bridge the methodological gap between deep learning models and spatial analytics. For

example, [22] reframed the workflow of spatial interpolation as a generative procedure and designed an adversarial DL architecture to predict unsampled spatial univariate values. [23] demonstrated how remote sensing imagery can be used for reliable estimation of human activity volumes by adding neighbor effects into a raster-based convolutional neural network (CNN). [24] proposed a geographically neural network weighted regression model that combines the linear coefficients of ordinary least squares (OLS) with artificial neural networks. Notably, [25] and [26] introduced the use of graph-based deep learning to explicitly model the relationships among connected locations and to predict the missing values of dependent variable based on independent variables.

As indicated in an earlier statement, [13] systematically outlined how neural networks could become a promising paradigm for spatial analysis. In this paper, we introduce that spatial regression, one of the prominent spatial analytical methods, can be conducted in the manner of deep learning through graph convolutional neural networks (GCNNs). We propose to use the spatial regression graph convolutional neural networks (SRGCNNs) as a deep learning paradigm to cope with similar tasks that apply to spatial regression analysis. Methodologically, we demonstrate the feasibility of SRGCNNs in spatial regression by going over its forward and back propagation mechanisms, reviewing its spatial locality learning nature and introducing its semi-supervised training strategy. The commonalities and differences between SRGCNNs and linear spatial regression models are further elaborated to help understand how each component in a traditional spatial regression analysis can be formalized in the new paradigm. We also provide an alternative way to enable geographically weighted regression in SRGCNN-based models. Practically, we provide a case study of the social media check-ins data at points of interest (POIs) to describe how SRGCNNs can be adopted in the urban scenario, where irregular distributed spatial multivariate data needs modeling and prediction. Appendix experiments on house price data are included to further show the data flexibility of SRGCNNs.

The remainder of this paper is structured as follows. Section 2 briefly reviews the current spatial regression models and related studies on graph-based deep learning. Section 3 formally explains how the standard spatial regression can be performed through the SRGCNNs paradigm and then proposes SRGCNN-based models. Section 4 introduces the datasets and experiment settings in the case study. Section 5 presents the results. Section 6 discusses the results and outlines some directions for future research. Finally, concluding comments are given in Section 7.

2 Related works

2.1 Graph-based deep learning

Through the back propagation of gradients in deep neural networks (DNNs), DL models have been proven to be extremely powerful in learning a way of transforming the input data into an ideal output representation [27]. More importantly, there has been a surge of interest in graph-based deep learning when the data is not structured in the regular spatial domain. Such kinds of data arise in various applications. For instance, in spatial networks, the attributes of spatial units can be modeled as the signals on the vertices of a graph [28]. The non-stationary neighborhood structures in a graph make it problematic to adopt the classical, raster-based convolution strategy [29]. To extend the well-established CNNs into

the irregular domain, graph convolutional neural networks (GCNNs) were developed, which follows an aggregation scheme where each node propagates characteristics of its neighbors to learn a deep representation of the contextual information [30].

Two types of approaches are proposed in general: 1) the spatial GCNNs and 2) the spectral GCNNs. On the one hand, the spatial approaches map the unordered graph space into a regularly ordered space via graph normalization [31] or by sampling and aggregation [32], so that the traditional convolution can be applied. On the other hand, the spectral GCNNs utilize graph Fourier transformation (GFT) to achieve the signal convolution on graphs. The idea is to transform the graph signals into the spectral domain based on structural features (i.e., Laplacian eigenvectors of the graph) [33, 34], then a graph convolutional filter with trainable parameters can be defined. The first formal model of spectral GCNNs was proposed by [35], where a parameterized diagonal matrix was used as the spectral convolutional filter. Based upon that, [30] and [36] proposed to use the polynomial expansion for the diagonal matrix in order to reduce the complexity of computation and to ensure a localized learning. A review of GCNNs' mathematical explanations can be found in [28].

As with geography applications, recent endeavors were focused on integrating GCNN modules into temporal deep learning frameworks such as recurrent neural network (RNN) and long-short term memory (LSTM) models, in order to achieve better forecasting accuracies in areas such as disaster warning [37] and traffic prediction [38–41]. On the other hand, GCNN applications for spatial analysis were designed mostly for the spatial pattern classification. [19] proposed a two-layer GCNNs architecture to perform a binary classification on vectorized building patterns. [21] introduced a semi-supervised approach to classify geo-located social media posts into multiple categories. The latest works started to use spectral GCNNs to predict urban characteristics [25, 26, 42], the experiments were designed similar to the typical spatial regression workflow. However, since the emphasis was on different spatial weighting measures, they did not provide a comparison between GCNNs and benchmark spatial regression variants.

2.2 Spatial regression models

Systematically reviews on the progress of spatial econometrics can be found in [4, 6, 7, 43–47] with respect to different focuses and specialties. Here, our focus is on spatial regression models (or linear spatial models) in a cross-sectional setting, among which the dominant family would be the spatial lag model (SLM). SLM allows observations of the dependent variable y_i at a spatial unit i ($i = 1, \ldots, N$) to depend on the observations in nearby units $j \neq i$. There are different specifications of the spatial dependence as the lag terms in SLM. The basic one would be spatial autoregressive (SAR) model that includes the spatial lag of dependent variables [2]:

$$y_i = \rho \sum_{j=1}^{N} w_{ij} y_j + \sum_{k=1}^{K} \beta_k x_{k,i} + \epsilon_i, \tag{1}$$

where w_{ij} is the element at the i-th row and j-th column of a N-by-N spatial weight matrix $\boldsymbol{W}$, $x_{k,i}$ is the k-th independent variable out of K, with ρ and β_k the parameters to be estimated, and $\epsilon_i \sim N(0, \sigma^2)$ the error term. The spatial lag can also be introduced in the independent variables, which is referred to as the spatially lagged X (SLX) models in [48]:

$$y_i = \sum_{k=1}^{K} \delta_k \sum_{j=1}^{N} w_{ij} x_{k,j} + \sum_{k=1}^{K} \beta_k x_{k,i} + \epsilon_i, \tag{2}$$

where δ_k is the spatial parameter of independent variable x_k.

Despite of many variants, a generalized model called spatial Durbin model (SDM) might be of interest if we are to include the spatial lags among all variables:

$$y_i = \rho \sum_{j=1}^{N} w_{ij} y_j + \sum_{k=1}^{K} \beta_k x_{k,i} + \sum_{k=1}^{K} \delta_k \sum_{j=1}^{N} w_{ij} x_{k,j} + \epsilon_i. \tag{3}$$

We assume SDM to be a comprehensive model for global spatial regression because it nests many of the models [48]. Apart from these global regression models, geographically weighted regression (GWR) and multi-scale geographically weighted regression (MGWR) refer to a range of specifications that focus on the spatial non-stationarity of relationships, where localized parameters are used to capture the spatial heterogeneous effects [49].

Many specialized estimation methods have been developed for spatial regression models, because the autocorrelation and collinearity between variables make it problematic to apply ordinary least squares (OLS) methods to estimate the model parameters [2]. The representative methods include maximum likelihood (ML) estimation [50], instrumental variables (IV) [51], general method of moments (GMM) [52], and Bayesian methods [3]. Back-fitting algorithms, which calibrate the optimal parameters through an iteration manner, are also found to be computational effective when the model incorporates complex spatial weightings and a large number of parameters [49].

3 Methodology

The workflow for a typical spatial regression analysis is illustrated in Fig. 1. In the cross-sectional setting, we start by constructing a spatial weights matrix $\boldsymbol{W} \in \mathbb{R}^{N \times N}$ for the N spatial units that have complete observations for both K independent variables $\boldsymbol{X} \in \mathbb{R}^{N \times K}$ and the dependent variable $\boldsymbol{y} \in \mathbb{R}^{N \times 1}$ (step 1). Then, the task is to specify a spatial regression model in the following generalized form (step 2):

$$\boldsymbol{y} = f_{\boldsymbol{\Theta}}(\boldsymbol{y}, \boldsymbol{W}, \boldsymbol{X}) + \boldsymbol{\epsilon}. \tag{4}$$

For simplicity, $\boldsymbol{\Theta}$ is a parameter set to be estimated, which contains all spatial and non-spatial effects when we fit the model to the observations (step 3). Finally, prediction for unobserved locations (step 4) is one of the most straightforward applications based on the fitted model [12].

Note that in the following of this article, we consider the case in which the dependent vector $\boldsymbol{y}$ is observed only for M ($\leq N$) spatial units, whilst $\boldsymbol{X}$ is observed for all N spatial units. We refer to this case as a common scenario for a range of geospatial tasks where the spatial multivariate data needs regression modeling and prediction.

Here, our objective is to bridge the methodological gap between GCNNs and traditional spatial regression models. We first clarify the forward and back propagation mechanisms in GCNNs given the context of a typical spatial regression workflow. Second, we show how the spatially localized learning is satisfied in graph convolutions. Third, a deep learning paradigm named spatial regression convolutional neural networks (SRGCNNs) is proposed to conduct spatial regression and prediction on spatial multivariate distributions. Fourth, we provide a basic SRGCNN model and a geographically weighted SRGCNN model.

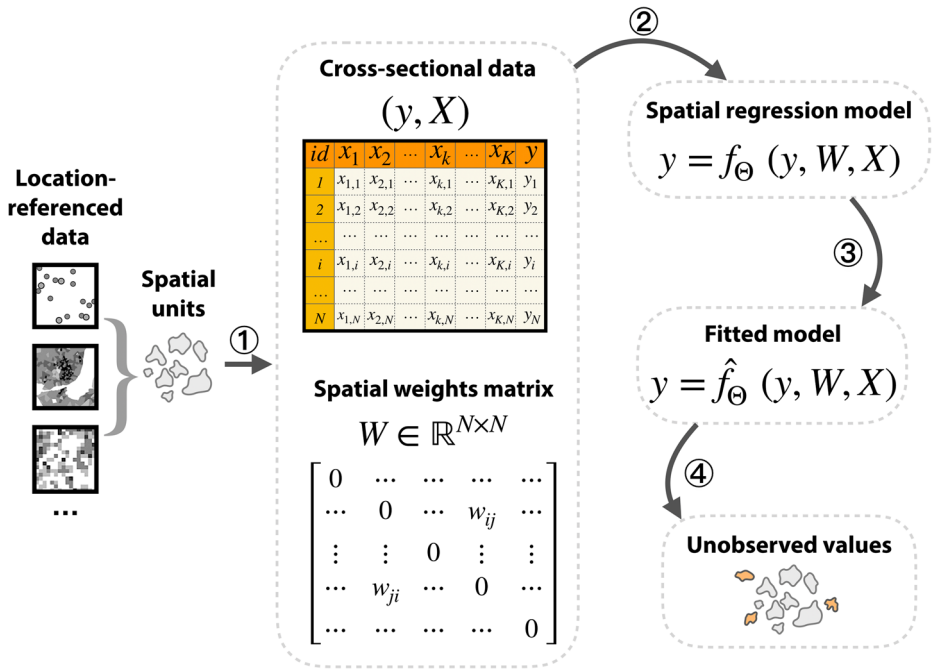


Fig. 1 Illustration of a typical workflow for the spatial regression analysis. Four major steps are marked as 1) collecting the cross-sectional data and constructing the spatial weights matrix given the spatial units, 2) specifying a regression model considering the potential spatial effects, 3) estimating parameters in the model, 4) predicting unobserved values for the dependent variable

3.1 The propagation mechanism in GCNNs

Steps 1,2 and 3 in the typical spatial regression workflow (Fig. 1) can be conducted in a very similar manner via a multi-layer GCNN model:

- Step 1: Spatial weights matrix $\boldsymbol{W}$ and cross-sectional data ($\boldsymbol{X}$, $\boldsymbol{y}$) are formalized as a full-connected graph in GCNNs.
- Step 2: The GCNN model incorporates potential spatial lag effects in the forward propagation of input graph features.
- Step 3: Neural network parameters $\boldsymbol{\Theta}$ are estimated through the back propagation and gradient descent with consideration of the spatial lag effects.

(1) For the first step, a graph $G = (V, E)$ is constructed to represent the irregular spatial structure of all the N spatial units. Each spatial unit i is formalized as a node $v_i \in V$ in G, and could be initialized with the corresponding node features from ($\boldsymbol{X}$, $\boldsymbol{y}$). The spatial weights matrix $\boldsymbol{W}$ quantified by a certain connectivity measures [25] is represented as the graph edges E, where $e_{ij} = (v_i, v_j, w_{ij}) \in E$ is the edge between v_i and v_j, and $w_{ij} \in \boldsymbol{W}$ is the weight of this edge. Note that w_{ij} takes on a non-zero value when i and j are considered to be neighbors, and a zero value otherwise, and the diagonal elements w_{ii} ($i = 1, \ldots, N$) are set to zero. Both GCNNs and spatial econometric models require $\boldsymbol{W}$ to be row-standardised, so that the sum of all w_{ij} over j is equal to one and $\boldsymbol{WX}$ is the weighted average of neighboring values or the so called first-order spatial lagged $\boldsymbol{X}$ [2].

(2) For the second step, we consider a fast implementation of GCNNs with the following layer-wise neural network forward propagation rule:

$$\boldsymbol{X}^{l+1} = \sigma(\boldsymbol{Z}^{l+1}) = \sigma(\boldsymbol{W_L}\boldsymbol{X}^{l}\boldsymbol{\Theta}^{l}) = \sigma(\widetilde{\boldsymbol{D}}^{-1/2}\widetilde{\boldsymbol{W}}\widetilde{\boldsymbol{D}}^{-1/2}\boldsymbol{X}^{l}\boldsymbol{\Theta}^{l}), \tag{5}$$

where $\boldsymbol{X}^l$ is the matrix of feature activations at the l-th layer, $\sigma(\cdot)$ denotes a non-linear activation function such as $Relu(\cdot) = max(0, \cdot)$, and $\boldsymbol{\Theta}^l$ is a layer-specific trainable parameter matrix. The term $\boldsymbol{W_L} = \widetilde{\boldsymbol{D}}^{-1/2}\widetilde{\boldsymbol{W}}\widetilde{\boldsymbol{D}}^{-1/2}$ is a renormalized Laplacian matrix of $\widetilde{\boldsymbol{W}} = \boldsymbol{W} + \boldsymbol{I}$, with $\widetilde{\boldsymbol{D}}_{ii} = \sum_j \widetilde{\boldsymbol{W}}_{ij}$ the degree matrix and $\boldsymbol{I}$ the identity matrix. The expression in Eq. 5 was derived based on the graph Fourier transformation and a polynomial expansion of the graph convolutional filter [36], which will be further clarified in Section 3.2. Intuitively, the output of a m-layer GCNNs is $\hat{\boldsymbol{y}} = \boldsymbol{X}^m$ and its input $\boldsymbol{X}^0 = \boldsymbol{X}$. Thus, the nature of forward propagation can be viewed analogously as progressively performing the layer-wise transformation:

$$\boldsymbol{y} = \sigma(\boldsymbol{W_L}\boldsymbol{X}\boldsymbol{\Theta}) \tag{6}$$

To compare it with spatial regression models, let us use SDM (3) as an example and rewrite it in the matrix notation:

$$\boldsymbol{y} = \rho\boldsymbol{W}\boldsymbol{y} + \boldsymbol{x}\beta + \boldsymbol{W}\boldsymbol{X}\boldsymbol{\delta} + \boldsymbol{\epsilon}. \tag{7}$$

If we assume that ρ in absolute value is less than 1, $\boldsymbol{W}$ is row-standardised, $\widetilde{\boldsymbol{X}} = [\boldsymbol{x}, \boldsymbol{W}\boldsymbol{X}]$ and $\boldsymbol{\Theta} = [\beta^T, \boldsymbol{\delta}^T]^T$, Eq. 7 can be expressed as the following by applying the well known infinite series expansion of the inverse of $(\boldsymbol{I} - \rho\boldsymbol{W})$:

$$\begin{aligned}
\boldsymbol{y} &= (\boldsymbol{I} - \rho\boldsymbol{W})^{-1}(\widetilde{\boldsymbol{X}}\boldsymbol{\Theta} + \boldsymbol{\epsilon}) \\
&= (\boldsymbol{I} + \rho\boldsymbol{W} + \rho^2\boldsymbol{W}^2 + \rho^3\boldsymbol{W}^3 + \ldots)(\widetilde{\mathbf{X}}\boldsymbol{\Theta} + \boldsymbol{\epsilon}) \\
&= \sum_{k=0}^{\infty}(\rho\boldsymbol{W})^k\widetilde{\boldsymbol{X}}\boldsymbol{\Theta} + \widetilde{\boldsymbol{\epsilon}}.
\end{aligned} \tag{8}$$

Here, needless to say, Eqs. 6 and 8 have exactly the same propagation mechanism that each spatial unit aggregates the independent variable values of its neighbors to learn the spatial lag effect. The difference, however, is that SDM considers the k-th order spatial lag effect as $\boldsymbol{W}^k\widetilde{\boldsymbol{X}}\boldsymbol{\Theta}$, whilst the fast implementation of GCNNs achieves the same modeling by repeating $\boldsymbol{W_L}\boldsymbol{X}\boldsymbol{\Theta}$ in a k-layer neural network.

(3) As with the third step, we derive how GCNNs optimizes parameters $\boldsymbol{\Theta}$ through its back propagation mechanism while considering the spatial lag effects of $\boldsymbol{X}$ and $\boldsymbol{y}$. Previous works [53] have demonstrated that the idea of maximizing the log-likelihood when estimating a linear model with Gaussian errors is logically the same as minimizing a loss function denoted by mean square errors (MSE) in a neural network [27, 54]. More specifically, an MSE loss function:

$$L(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \frac{1}{M}\sum(\boldsymbol{y} - \hat{\boldsymbol{y}})^2 \tag{9}$$

can be used in GCNNs to measure the output errors on the $M(\leq N)$ spatial units that have observed $\boldsymbol{y}$ values. Then, GCNNs updates the layer-wise parameters through the gradient descent of L with an η learning rate:

$$\boldsymbol{\Theta}^l \leftarrow \boldsymbol{\Theta}^l - \eta\frac{\partial L}{\partial \boldsymbol{\Theta}^l} = \boldsymbol{\Theta}^l - \eta\frac{\partial L}{\partial \boldsymbol{Z}^{l+1}}\frac{\partial \boldsymbol{Z}^{l+1}}{\partial \boldsymbol{\Theta}^l}. \tag{10}$$

Recalling the forward propagation rule indicated in Eq. 5, we have:

$$\frac{\partial \boldsymbol{Z}^{l+1}}{\partial \boldsymbol{\Theta}^l} = \frac{\partial(\boldsymbol{W_L}\boldsymbol{X}^l\boldsymbol{\Theta}^l)}{\partial \boldsymbol{\Theta}^l} = (\boldsymbol{W_L}\boldsymbol{X}^l)^T. \tag{11}$$

Then, the loss gradient at the l-th layer $\boldsymbol{J}^{(l)} = \frac{\partial L}{\partial \boldsymbol{Z}^l}$ can be calculated iteratively as:

$$\boldsymbol{J}^{(l)} = \frac{\partial L}{\partial \boldsymbol{Z}^l} = \frac{\partial L}{\partial \boldsymbol{X}^l}\frac{\partial \boldsymbol{X}^l}{\partial \boldsymbol{Z}^l} = \frac{\partial L}{\partial \boldsymbol{Z}^{l+1}}\frac{\partial \boldsymbol{Z}^{l+1}}{\partial \boldsymbol{X}^l}\frac{\partial \boldsymbol{X}^l}{\partial \boldsymbol{Z}^l} = \boldsymbol{J}^{(l+1)}{\boldsymbol{W_L}}^T\boldsymbol{\Theta}^l\sigma'. \tag{12}$$

This iterative relation of loss gradients actually includes a spatial lagged term of $\boldsymbol{y}$ when multiplying $\boldsymbol{J}$ and ${\boldsymbol{W_L}}^T$. Further, by substituting (11) and (12) into (10), we obtain the back propagation nature of GCNNs as following:

$$\boldsymbol{\Theta}^l \leftarrow \boldsymbol{\Theta}^l - \eta\boldsymbol{J}^{(l+1)}(\boldsymbol{W_L}\boldsymbol{X}^l)^T. \tag{13}$$

Therefore, it is obvious that the spatial lag effects of $\boldsymbol{X}$ and $\boldsymbol{y}$, i.e., $\boldsymbol{W_L}\boldsymbol{X}^l$ and $\boldsymbol{J}^{(l)}{\boldsymbol{W_L}}^T$, are both explicitly considered to optimize the model parameters.

3.2 Spatial locality in GCNNs

Besides from the spatial lag effects, GCNNs strictly follows a spatially localized nature where each node propagates data from nearby ones denoted in the graph structure. As aforementioned, formal spectral GCNN models (e.g., Eq. 5) were mostly derived based on the graph Fourier transformation and a polynomial expansion of the graph convolutional filter. The graph convolution operator is defined as first multiplying $\boldsymbol{X}$ in the Fourier domain ($\hat{\boldsymbol{X}}$) with a graph filter g_θ and then transforming it back to the spatial domain [34], which is formulated as:

$$\boldsymbol{y} = \boldsymbol{U}g_\theta\hat{\boldsymbol{X}} = \boldsymbol{U}g_\theta(\Lambda)(\boldsymbol{U}^T\boldsymbol{X}) = g_\theta(\boldsymbol{U}\Lambda\boldsymbol{U}^T)\boldsymbol{X} = g_\theta(\boldsymbol{L}_s)\boldsymbol{X}, \tag{14}$$

where $\boldsymbol{L}_s = \boldsymbol{I} - \Delta^{-1/2}\boldsymbol{W}\Delta^{-1/2}$ is the normalized Laplacian matrix of $\boldsymbol{W}$ with the degree matrix $\Delta_{ii} = \sum_j \boldsymbol{W}_{ij}$. Since $\boldsymbol{L}_s$ is a real symmetric positive semi-definite matrix, it has a complete set of orthonormal eigenvectors $\boldsymbol{U} = (u_1, \cdots, u_N)$ and associated nonnegative eigenvalues $\Lambda = diag([\lambda_1, \cdots, \lambda_N])$. The Laplacian is then diagonalized by $\boldsymbol{U}$ such that $\boldsymbol{L}_s = \boldsymbol{U}\Lambda\boldsymbol{U}^T$, and the graph Fourier transformation of $\boldsymbol{X}$ is represented as $\hat{\boldsymbol{X}} = \boldsymbol{U}^T\boldsymbol{X}$.

To ensure the spatially localized learning, a K^{th}-order graph convolution operator can be defined to approximate (14) using polynomials up to K^{th} order:

$$\boldsymbol{y} = g_\theta(\boldsymbol{L}_s)\boldsymbol{X} = \sum_{k=0}^{K}\theta_k\boldsymbol{L}_s^k\boldsymbol{X} = \theta_0\boldsymbol{L}_s^0\boldsymbol{X} + \theta_1\boldsymbol{L}_s^1\boldsymbol{X} + \cdots + \theta_K\boldsymbol{L}_s^K\boldsymbol{X}, \tag{15}$$

where $\boldsymbol{L}_s^k$ is the k^{th} power of the graph Laplacian, the kernel size K of graph filter g_θ implies a vector of trainable parameters $\theta = (\theta_0, \theta_1, \ldots, \theta_K)$ shared by all graph nodes. According to the property of graph Laplacian, the self-multiplication $(\boldsymbol{L}_s^K)_{i,j} = 0$ when the shortest path between nodes v_i and v_j is greater than K edges in the spatial domain [34]. Therefore, a graph convolution approximated using K^{th} order polynomials is constrained within the K-hop local neighborhood for each node $v_i \in V$.

Despite of various polynomial types and parameter simplifications to define the graph convolutions [21, 25, 30, 36], state-of-the-art GCNNs generally follow such spatially localized learning scheme. For example, the fast implementation introduced in Eq. 5 is actually a simple specification that considers up to the 1^{st}-order neighbors in each neural network layer. Such a model with K layers is capable of approaching the K^{th}-order neighbors, details can be found in [36].

3.3 SRGCNNs: a deep learning paradigm for spatial multivariate distributions

Given these knowledge on how GCNNs could imitate a typical spatial regression analysis, we propose spatial regression convolutional neural networks (SRGCNNs) as a deep learning paradigm to conduct spatial regression and prediction on spatial multivariate distributions. The major contribution of SRGCNNs is to formalize GCNNs in the context of spatial regression, which enables geographers to intuitively understand the linkages between graph convolution mechanisms and the key concepts in spatial regression models.

As illustrated in Fig. 2, the workflow of SRGCNNs is similar to the traditional spatial regression analysis in all four steps (Fig. 1).

SRGCNNs starts by collecting the location-referenced data on the N spatial units and building a spatial graph that encodes the spatial weights matrix and cross-sectional data (step 1). Then, we initialize all nodes with the observed $\boldsymbol{X}$ values. These values are forward propagated through a specific GCNNs architecture following the idea of Eq. 5, whilst the $\boldsymbol{y}$ values sampled at the training nodes are input to the last GCNNs layer to calculate the output errors and to enable the back propagation of the model (step 2). Next, SRGCNNs optimizes parameters in its GCNN model following a semi-supervised learning strategy (step 3), which explicitly considers the spatial weights among all the N spatial units, even though the $\boldsymbol{y}$ values are observed only at M ($\leq N$) training nodes (Fig. 3). Profited by the propagation mechanism (Section 3.1) and spatial locality nature (Section 3.2) of GCNNs, SRGCNNs optimizes a GCNN model to achieve the most credible approximation of spatial relationships, and as the output, to predict the unobserved $\boldsymbol{y}$ values (step 4).

More importantly, SRGCNNs uses a transductive semi-supervised learning strategy to deal with the common scenario where $\boldsymbol{y}$ is partially sampled, whilst $\boldsymbol{X}$ is observed for all

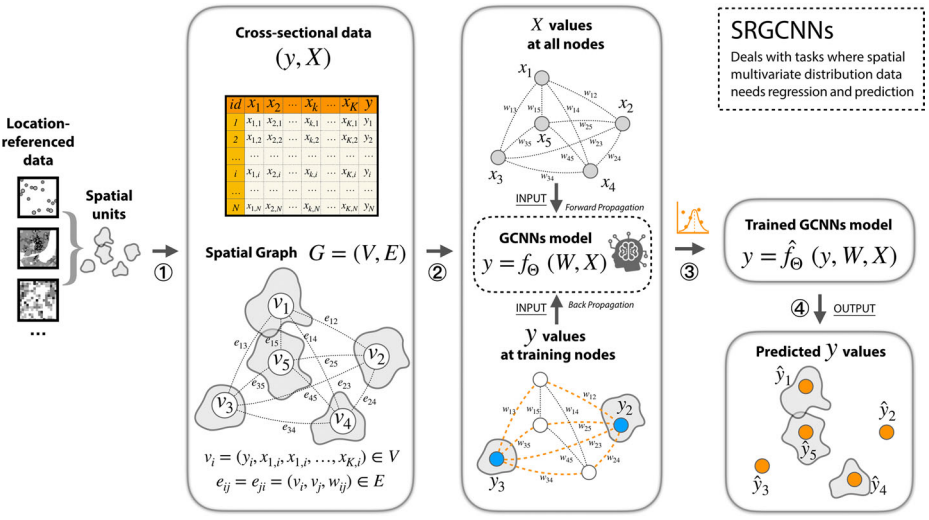


Fig. 2 Spatial regression graph convolutional neural networks (SRGCNNs) as a deep learning paradigm for the regression analysis of spatial multivariate distributions. The four steps of SRGCNNs are similar to that of the traditional spatial regression analysis: 1) collecting the cross-sectional data on spatial units and constructing the spatial graph that incorporates spatial weights and observed variables in edges and nodes, respectively; 2) specifying a GCNN model architecture that takes $\boldsymbol{X}$ at all nodes and the observed $\boldsymbol{y}$ values at sampled nodes as the inputs; 3) evaluating the output errors and estimating neural network parameters via semi-supervised learning; 4) predicting the unobserved $\boldsymbol{y}$ values

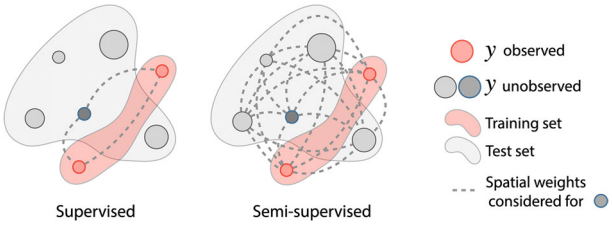


Fig. 3 Supervised learning v.s. semi-supervised learning in the context of spatial regression and prediction. The $\boldsymbol{y}$ values at all test locations are to be predicted while the dark grey location is chosen as an example to illustrate the spatial weights considered when the prediction is performed at one of the locations

spatial units. In comparison, most linear spatial models [6] are based on the supervised strategy, where only the locations with observed $\boldsymbol{y}$ can be included into the model training, thus they are not directly suitable for predicting all unobserved values when the training locations are improperly sampled or under-sampled [55].

Figure 3 presents a schematic representation of how the semi-supervised learning is different from the supervised learning given the context of spatial regression and prediction. Compared with the supervised learning, the significant distinction of semi-supervised learning is quite easy to understand: all spatial units are connected as a comprehensive graph, such that the spatial effects of unsampled locations are also included during modeling and prediction. SRGCNNs captures a broad picture of spatial dependence with all spatial weights encoded, thus the updating of model parameters is not only based on the training set but also a large pool of test locations. When the sampling ratio is low, the number of spatial weights considered in a supervised model can be much smaller than in a semi-supervised model.

In Table 1, we summarized the commonalities and differences between the proposed SRGCNNs and several representative linear spatial regression models, i.e., spatial lagged X model (SLX), spatial autoregressive model (SAR), spatial Durbin model (SDM) and geographically weighted regression (GWR).

First, speaking of spatial structure, SRGCNNs is the only one that considers an $N \times N$ weights matrix whilst all other benchmarks build their spatial weights matrix on the M training locations with observed $\boldsymbol{y}$ values. Second, these spatial econometric models are specified in linear forms, while SRGCNNs designs its model as a neural network which does not assume any form of the linear functions to be fitted. Third, spatial lag effects can vary from models. SDM and SRGCNNs are the two that explicitly consider both the lags for $\boldsymbol{X}$ and $\boldsymbol{y}$. It is also worth noting that the spatial lags are captured for all the N locations in SRGCNNs due to its $N \times N$ spatial structure and the semi-supervised training. Fourth, GWR is a special model that allows the parameters to vary spatially, so as to capture the heterogeneity of spatial localized relationships. However, the common implementation of GCNNs (Eq. 5) is not able to model the heterogeneity since the graph convolutional filters are learned via a parameter-sharing scheme [30]. To overcome this, we will present a way to support geographically weighted learning in SRGCNNs in Section 3.4.2. Lastly, parameters can be directly interpreted as the coefficients of one variable on another in linear spatial models, while the interpretability of SRGCNNs is not as good because the high-dimensional graph neural network parameters require specialized tools to be visualized and understood.

Table 1 Commonalities and differences between SRGCNNs and traditional spatial regression analysis

	Spatial structure $\boldsymbol{W}$	Regression equation	Lag effects	Estimation	Parameter sharing	Learning strategy	Interpretability
SLX	$M \times M$	$\boldsymbol{y} = \boldsymbol{x}\beta + \boldsymbol{WX}\delta + \epsilon$	$\boldsymbol{WX}$	OLS	YES	Supervised	YES
SAR	$M \times M$	$\boldsymbol{y} = \rho\boldsymbol{Wy} + x\beta + \epsilon$	$\boldsymbol{WY}$	ML/IV	YES	Supervised	YES
SDM	$M \times M$	$\boldsymbol{y} = \rho\boldsymbol{Wy} + x\beta + \boldsymbol{WX}\delta + \epsilon$	$\boldsymbol{WX}$ & $\boldsymbol{WY}$	ML/IV	YES	Supervised	YES
GWR	$M \times M \times M$	$y_i = x_i\beta_i + \boldsymbol{W}^{(i)}\boldsymbol{X}\delta_i + \epsilon_i$	$\boldsymbol{WX}$	OLS	NO	Supervised	NO
SRGCNNs	$N \times N$	e.g., $\boldsymbol{X}^{l+1} = \sigma(\boldsymbol{W_L}\boldsymbol{X}^l\Theta^l)$	$\boldsymbol{WX}$ & $\boldsymbol{WY}$	BP	YES	Semi-Supervised	NOT YET

M is the number of locations with observed y values; N is the total number of interested locations in the study area

OLS: Ordinary least square; ML: Maximum likelihood; IV: Instrumental variables; BP: Back propagation

3.4 SRGCNN-based models

3.4.1 Basic SRGCNN model

Graph convolution is the key component in SRGCNNs. In this section, we first design a two-layer basic SRGCNN model based on the fast implementation of GCNNs [25, 36]. The forward propagation of our basic model takes the simple form:

$$\hat{\boldsymbol{y}} = \boldsymbol{W_L}(\sigma(\boldsymbol{W_L}\boldsymbol{X}\boldsymbol{\Theta}^{(0)}))\boldsymbol{\Theta}^{(1)}. \tag{16}$$

Note that a renormalized graph Laplacian $\boldsymbol{W_L} = \widetilde{D}^{-1/2}(\boldsymbol{W} + \boldsymbol{I})\widetilde{D}^{-1/2}$ derived based on the Chebyshev polynomials expansion of Eq. 14 is used as the spatial structure. Details of $\boldsymbol{W_L}$ can be found in [36]. Similar to the discussion of Eq. 15, we could expect spatial learning with a single, shared parameter θ been used as the graph filter in this basic SRGCNN model. During model training, θ is embedded into the layer-wise weights $\boldsymbol{\Theta}^{(0)}$ and $\boldsymbol{\Theta}^{(1)}$. Therefore, θ is omitted in Eq. 16.

3.4.2 Geographically weighted SRGCNN model

As indicated in Table 1, spatial regression models can vary in terms of whether they allow the parameters to change spatially. Similar to SLX, SAR and SDM, our basic SRGCNN model in Fig. 6 is on the side of global learning, which adopts shared parameters to characterize the stationary knowledge of spatial relationships and doesn't consider the heterogeneity between locations. In contrast, local models such as GWR explicitly considers the spatial non-stationarity, incorporating different spatial weights matrix $\boldsymbol{W}^{(i)}$ and the corresponding parameters β_i, δ_i at each location to enable a geographically weighted learning (Table 1). Given that, we try to introduce here a way to enable the geographically weighted learning in SRGCNN-based models.

The logic of our basic SRGCNN model in Eq. 16 can be simplified as the following layer-wise graph convolution:

$$\boldsymbol{X}^{(l+1)} = \boldsymbol{W} \times \boldsymbol{X}^{(l)} \times \boldsymbol{\Theta}^{(l)}. \tag{17}$$

Assuming N spatial units, input features with C_{in} channels and output features with C_{out} channels, we have $\boldsymbol{X}^{(l+1)} \in \mathbb{R}^{N\times C_{out}}$, $\boldsymbol{X}^{(l)} \in \mathbb{R}^{N\times C_{in}}$, $\boldsymbol{W} \in \mathbb{R}^{N\times N}$ and $\boldsymbol{\Theta}^{(l)} \in \mathbb{R}^{C_{in}\times C_{out}}$. The parameters $\boldsymbol{\Theta}^{(l)}$ are shared among all GCNN neurons, leading to a global spatial regression within a local-connected graph structure.

Similar to how GWR extends global spatial regression, we propose the geographically weighted SRGCNN model (SRGCNN-GW) with a geographically weighted layer-wise graph convolution:

$$\boldsymbol{X}^{(l+1)} = \boldsymbol{W} \times (\boldsymbol{X}^{l} \otimes \boldsymbol{\Theta}^{(l)}_{local}) \times \boldsymbol{\Theta}^{(l)}, \tag{18}$$

where $\boldsymbol{\Theta}^{(l)}_{local} \in \mathbb{R}^{N\times C_{in}}$ contains the geographically weighted parameters. SRGCNN-GW calculates the element-wise product between input spatial features and the geographically weighted parameters $\boldsymbol{X}^{l} \otimes \boldsymbol{\Theta}^{(l)}_{local}$. Thus, the features are parameterized, with each spatial unit having an independent set of trainable parameters. Compared with the basic SRGCNN model, the forward propagation of a two-layer SRGCNN-GW model takes the following form:

$$\hat{\boldsymbol{y}} = \boldsymbol{W_L}(\sigma(\boldsymbol{W_L}(\boldsymbol{X} \otimes \boldsymbol{\Theta}^{(0)}_{local})\boldsymbol{\Theta}^{(0)})) \otimes \boldsymbol{\Theta}^{(1)}_{local})\boldsymbol{\Theta}^{(1)}. \tag{19}$$

4 Data and experiment setup

As a novel deep learning paradigm for spatial regression and prediction, SRGCNNs has both its strengths and weaknesses compared with traditional spatial regression models. To further evaluate SRGCNNs in scenarios where the multivariate spatial distribution data needs regression and prediction, we provide case experiments within the urban context in the following sections. We intend to elaborate on the common experiment settings for SRGCNNs, outline the results of SRGCNN-based models, and compare them with benchmark models quantitatively.

4.1 Study area and data descriptions

We selected the urban region within the fifth ring road [56] of Beijing, China, as the study area. The region has an area of 668.72 km^2, a giant metropolis where the fast-developing economy has bred rich urban facilities and diverse human activities. This study area is considered the most complex and populous region in Beijing regarding socioeconomic vitalities such as dining, residence, transportation and business [25]. As for the spatial multivariate distributions, we utilized a dataset from a social media platform named Sina Weibo [57]. The original dataset contains over 868 million annual check-in records in 2014 on 143,576 points of interest (POIs) in Beijing.

Further, we find that the original check-in values are heavy-tail distributed. Most POIs are not very active and only a few POIs have very high check-in numbers. No matter the type, most POIs have around 10^2 check-ins and the highest check-in intensity is more than 10^4. This indicates a very agglomerated spatial pattern of the human check-in activities in Beijing that maybe difficult for the spatial models to estimate. Thus, we computed the logarithmic numbers of the check-in activities to the base 10 as the feature values, denoted as log_{10}(# of check-ins), in order to reduce the skewness of check-in numbers.

Within the study area, we selected the POIs with over 100 annual check-in records as the basic spatial analysis units (4,636 POIs in total) in the case study (Fig. 4a). By selecting only the POIs with over 100 annual check-ins, inactive POIs with less check-ins are left out to balance the data distribution (Fig. 4b). Based on the 242 sub-labels detailed in the raw dataset, these POIs are reclassified into six dominant categories (corresponding POI numbers are denoted by #) according to their functional types, i.e. dining (#=1,923), residence (#=1,117), transport (#=351), business (#=524), recreation (#=528) and medical (#=133). Generally speaking, POIs are more clustered in the northern and eastern areas. Residential facilities (orange) are scattered around the ring roads, active business POIs (cyan) are more concentrated in CBD areas in the northwest and the east, while recreation (pink) and dining (blue) POIs are scattered in the whole area, with a diverse range of check-in numbers (Fig. 4a). It can be seen from Fig. 4b and c that the logarithm and the threshold cut-off operations flattened the check-in data, leading to a considerably balanced target data distribution that benefits the modeling.

In the case study, we treated POIs' types as the independent variables ($\boldsymbol{X}$), the layout structure of POIs as the spatial support to build the spatial weights matrix and graph ($\boldsymbol{W}$), while the logarithmized check-in numbers are the dependent variable values ($\boldsymbol{y}$) to be predicted at the POI level.

We do not pursue a very high prediction accuracy in case experiments, instead, we try to discuss the usage of SRGCNNs with comparison to baseline linear spatial regression models. Despite of this, we do invite future research to extend and improve our modeling

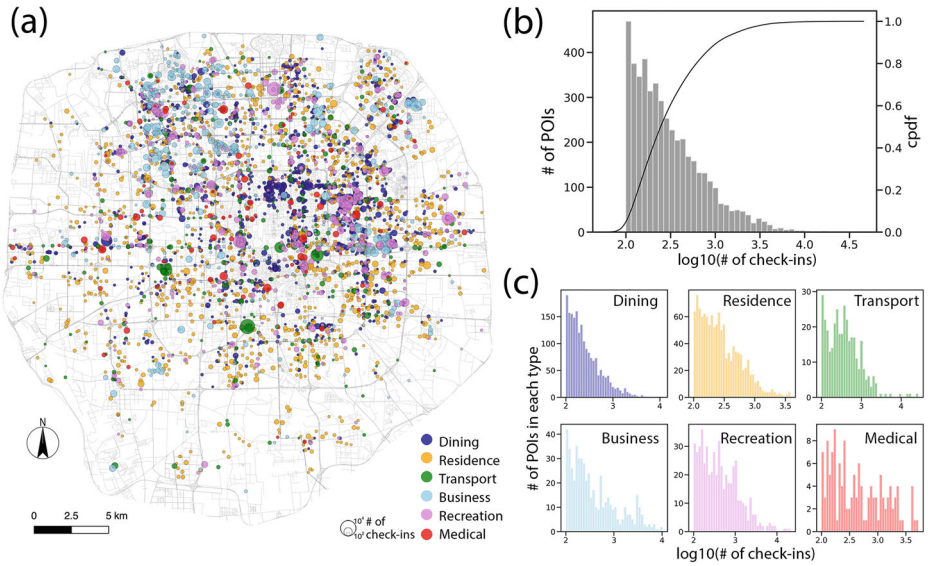


Fig. 4 Study area and data descriptions. **a** Spatial distribution of the POIs that contain more than 100 annual check-in records and their functional types within the fifth ring road of Beijing (for simplicity, the north arrows and scale bars in all maps later in this paper are omitted). **b** Histogram and the cumulative probability distribution of the logarithmic check-in numbers. **c** Histograms of the logarithmic check-in numbers in the six dominant categories

framework into a wider range of scenarios where higher practical value and more concrete contribution are expected.

4.2 Experiment setup

4.2.1 Graph construction

We built the spatial weights matrix and graph based on a categorical k nearest neighbor searching, that is, considering up to the k^{th} nearest neighbors regarding each POI types to be the adjacent locations. Assuming that the number of GCNN layers is γ and the sampling ratio of training POIs among all POIs is ϕ, the smallest number of k should satisfy $k^{\gamma} \geq \frac{1}{\phi}$, such that the model covers all POIs after the self-multiplication of weights matrix when features of training POIs are propagated in the GCNN. For instance, if we implement a two-layer GCNN model to 10% training POIs and 90% testing POIs, i.e., $\gamma = 2$ and $\phi = \frac{1}{10}$, then the required k needs to be at least 4. In order to make sure that we cover all POIs in each type under a 10% random sampling (the lowest ratio in all settings) in the case study, we used $k = 5$ when building the graph. Unless explicitly mentioned, we fixed the spatial weights matrix to be of the same nearest neighbor structure in all following experiments, so that the results are comparable across spatial regression models.

To build the graph, all elements in the spatial weights matrix $\boldsymbol{W} \in \mathbb{R}^{4636\times4636}$ were first initialized to be zero. Then, we searched the five nearest neighbors in each functional type for each POI and set the corresponding element in $\boldsymbol{W}$ to be one. To ensure that $\boldsymbol{W}$ satisfies the symmetry property required in the graph Laplacian transformation (14), we used a symmetric positive semi-definite matrix $\boldsymbol{W} = \boldsymbol{W} + \boldsymbol{W}^T$ as the input weights matrix to build the

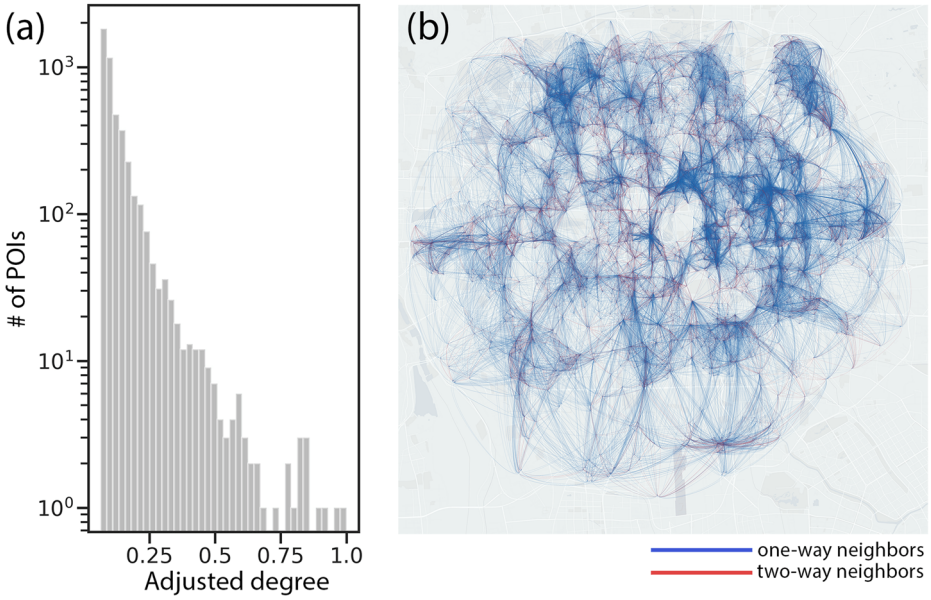


Fig. 5 The graph structure constructed using the categorical five nearest neighbors. **a** Adjusted degree distribution of the graph. **b** Visualization of graph edges

graph. The computed matrix $\boldsymbol{W}$ has three unique values, where the value two means the two POIs are within each other's nearest neighbors (two-way neighbors), value one denotes the one-way neighbors and the value zero denotes that two POIs are not neighbors.

The constructed graph using the categorical five nearest neighbors is illustrated in Fig. 5. After standardizing the node degrees according to the largest nodal degree, we saw an negative exponential distribution for the adjusted degrees as shown in Fig. 5a, indicating a sparse graph where most POIs have smaller degrees while only a few POIs act as the hub nodes with very high degrees. We obtained 61,548 two-way connections and 155,064 one-way connections for 4,636 POIs, these POIs are sparsely connected in the constructed graph but there is no isolated node (Fig. 5b). Admittedly, the definition of spatial weights matrix would affect the regression analysis, a more informative geographical context would lead to a better model fitting and a higher spatial predictability [25].

4.2.2 Training architecture

The training architecture of our SRGCNN-based models adopted in this case study are illustrated in Fig. 6. POIs' types are input as the independent variables $\boldsymbol{X}$, renormalized graph Laplacian $\boldsymbol{W_L}$ is applied for the graph convolution operation, while the logarithmized check-in numbers $\hat{\boldsymbol{y}}$ are the dependent variable values to be estimated. The loss L between the ground-truthing $\boldsymbol{y}$ and $\hat{\boldsymbol{y}}$ is computed only for the training POIs, and L is then used for the back propagation and gradient decent in the model. A final output $\hat{\boldsymbol{y}}$ for all POIs can be obtained after sufficient training epochs.

More specifically, independent variable values are encoded as one-hot vectors $\boldsymbol{X} \in \mathbb{R}^{N \times C}$, where $N = 4,636$ is the number of all POIs and $C = 6$ denotes the six type channels ordered by dining, residence, transport, business, recreation and medical. For example,

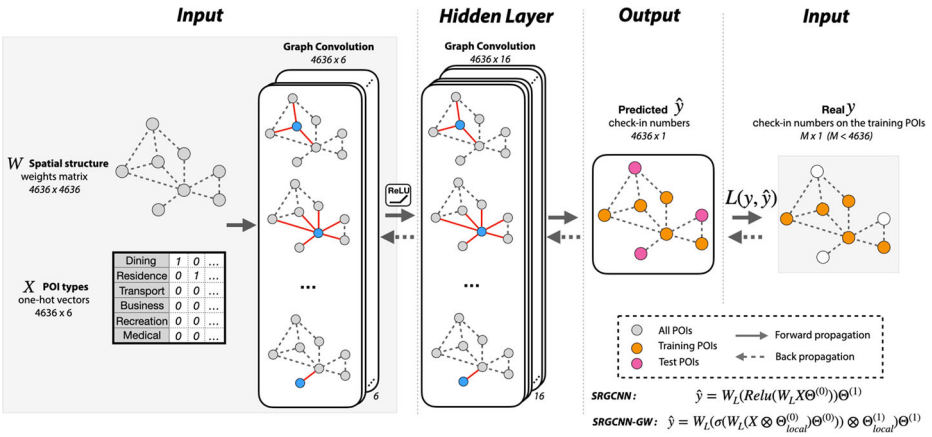


Fig. 6 SRGCNN-based training architecture implemented for the check-in number regression and prediction in Beijing urban area

if the POI with $id = 1$ is labeled as dining, then the corresponding one-hot vector passed to the graph convolution would be $x_1 = [1, 0, 0, 0, 0, 0]$. The input features are output as $\hat{\boldsymbol{y}} = [\hat{y_1}, \hat{y_2}, \ldots, \hat{y_N}] \in \mathbb{R}^N$ after graph convolutions and layer-wise transformations. For the basic SRGCNN model in Eq. 16, $\boldsymbol{\Theta}^{(0)} \in \mathbb{R}^{6\times16}$ is the layer-wise weights connecting the input layer with six feature channels and the hidden layer with 16 feature channels, and $\boldsymbol{\Theta}^{(1)} \in \mathbb{R}^{16\times1}$ maps the hidden features into the output $\hat{\boldsymbol{y}}$. For the SRGCNN-GW model in Eq. 19, $\boldsymbol{\Theta}^{(0)}_{local} \in \mathbb{R}^{4,636\times6}$ and $\boldsymbol{\Theta}^{(1)}_{local} \in \mathbb{R}^{4,636\times16}$ are further applied to enable the geographically weighted parameter learning.

We use ReLU as the activation function in the hidden layer. No activation function is used before the output layer, since we would like the regression model to approximate accurate numbers of $\boldsymbol{y}$ instead of generating classified labels. No drop-out is applied because we hope to maintain the spatial structure as well the attributes at every location to make comprehensive predictions.

4.2.3 Training settings

We conducted the experiments based on multiple training ratios, i.e., 10%, 20%, 40%, 60%, 80% to evaluate the model performances. For each of the training ratio, fifty parallel simulations with randomly selected training POIs and test POIs were carried out to achieve stable results. For each experiment, we evaluated the model fitting of training data, and more importantly, we investigate the accuracy of prediction on the test POIs. We didn't use any validation set as in most machine learning tasks, because our context is the spatial regression analysis without parameter tuning, where the training data is fitted first, and then the fitted models are directly used for prediction.

Two common quantitative evaluation metrics were used to help measure the model performances regarding the discrepancy between $\boldsymbol{y}$ and $\hat{\boldsymbol{y}}$: mean absolute percentage error (MAPE) on the test locations and mean square error (MSE) on the training locations. MAPE ranges from 0 to 1, with a higher value indicating a larger dissimilarity between $\boldsymbol{y}$ and $\hat{\boldsymbol{y}}$. MSE measures the squared error of estimation, which is widely used for comparing the sta-

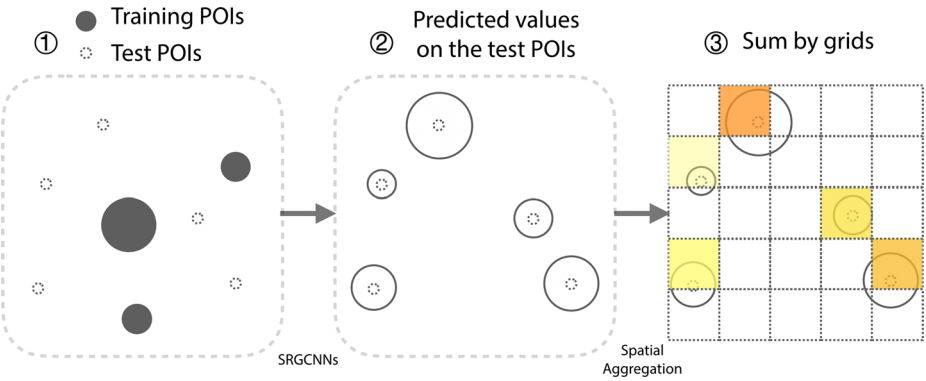


Fig. 7 Spatial visualization of the predicted check-in numbers

bility of prediction models. Assuming that there are M training POIs and $(N - M)$ test POIs, the calculation of MAPE on the test POIs is:

$$MAPE = \frac{100\%}{N-M} \sum_{i=1}^{N-M} \frac{|\hat{y}_i - y_i|}{y_i}, \tag{20}$$

and the MSE on training POIs is:

$$MSE = \frac{1}{M} \sum_{i=1}^{M} (\hat{y}_i - y_i)^2. \tag{21}$$

As for training the SRGCNN-based models, we set the loss function $L(\hat{\boldsymbol{y}}, \boldsymbol{y})$ to be L2-loss, which has exactly the same mathematical form as MSE. We used Adam optimizer, where $\beta_1 = 0.5$ and $\beta_2 = 0.999$. The learning rate was $\eta = 10^{-3}$ for SRGCNN model and $\eta = 3 \times 10^{-4}$ for the SRGCNN-GW model. Training epochs were capped at 8,000, and we recorded the best results among all epochs. The experiments were implemented using PyTorch[1], a deep learning framework in python with GPU acceleration. All the benchmark linear spatial models were implemented using PySAL, an open-source python package designed to support spatial data science[2]. The computational environment was a Linux server with one NVIDIA 1080TI GPU, a 2.40GHz Intel E5-2680 CPU, and 128 GB RAM. All codes and data needed to replicate this work would be available online once the paper is published.

4.2.4 Visualization

We followed the visualization approach illustrated in Fig. 7 to plot the irregular spatial distribution of check-in records in a map. Since it is hard to directly interpret the differences between two spatial distributions of point-level check-ins, we aggregated the values on the test POIs onto 0.5 $km \times 0.5$ km geographical grid cells, a reasonable scale of the regular spatial grid for social media data analysis, as indicated in [58]. The saturation of color in each cell denotes the corresponding summed check-in numbers of all test POIs within the

[1] https://pytorch.org
[2] http://pysal.org/packages

cell's extent. By transforming vector points into raster cells, we hope to provide relatively clear spatial patterns for the visual comparison between real numbers $\boldsymbol{y}$ and predicted $\hat{\boldsymbol{y}}$ on the test POIs.

5 Results

In the following sections, we will present the results of our experiments. First, we show how the basic SRGCNN model performed in a sample experiment with only 10% sampling ratio. Second, we compare the results between the SRGCNN model and the SRGCNN-GW model. Last, we quantitatively compare SRGCNN-based models to the benchmark linear spatial models across different sampling ratios, with respect to both fitting the training data and predicting the test data.

5.1 SRGCNN model performance in a sample experiment

Here, we use a sample experiment with only 10% sampling ratio to present an intuitive result of how SRGCNNs performs in modeling the check-in data. The randomly initialized training POIs (10%) and test POIs (90%) are visualized in Fig. 8a. The real numbers of check-ins on the training POIs (black points) are used ($\boldsymbol{y}$), while the numbers of check-ins on the test POIs (while points) are to be predicted ($\hat{\boldsymbol{y}}$). Using the graph structure mentioned in Section 4.2.1, a basic SRGCNN model with the two-layer GCNN model (see Section 4.2.2) is adopted for data regression and prediction. Our training and evaluation followed the settings in Section 4.2.3. As a result, the spatial distribution patterns of real and predicted check-ins on test POIs are visualized in Fig. 8b and c, respectively. The colors from yellow to black denote the numbers from the minimum to the maximum in each $0.5\ km \times 0.5\ km$ grid cell, the mapping between colors and logarithmic check-in numbers is based on the quantile method of six categories.

The overall prediction accuracy obtained is MAPE≈11.75% in the sample experiment, based upon only the 10% sampled $\boldsymbol{y}$ and the comprehensive graph structure $\boldsymbol{W} \in \mathbb{R}^{4636\times4636}$. As shown in Fig. 8, the predicted spatial pattern of check-in numbers is very similar to the real pattern. The rank-size relationships in most local regions are well reproduced. Local regions are distinguishable in the predicted pattern in terms of the spatial autocorrelation of check-in numbers. However, we can see that the clustered

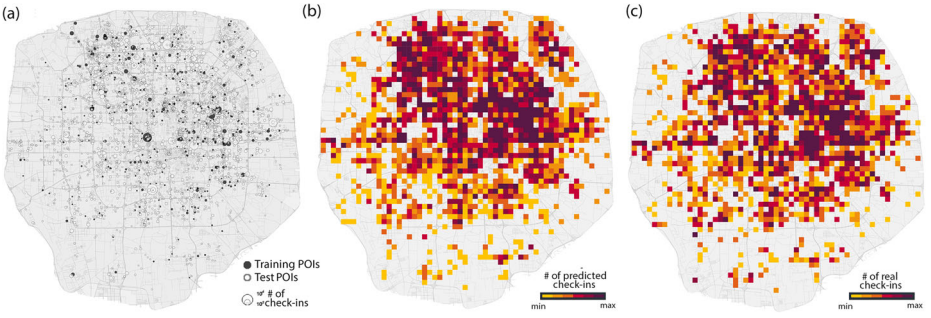


Fig. 8 Spatial distribution patterns of check-ins on the test POIs in a sample experiment with 10% sampling ratio. **a** Initialization of training (10%) and test (90%) POIs in the sample experiment. **b** Predicted spatial pattern of check-ins on the test POIs using SRGCNNs. **c** Real spatial pattern of check-ins on the test POIs

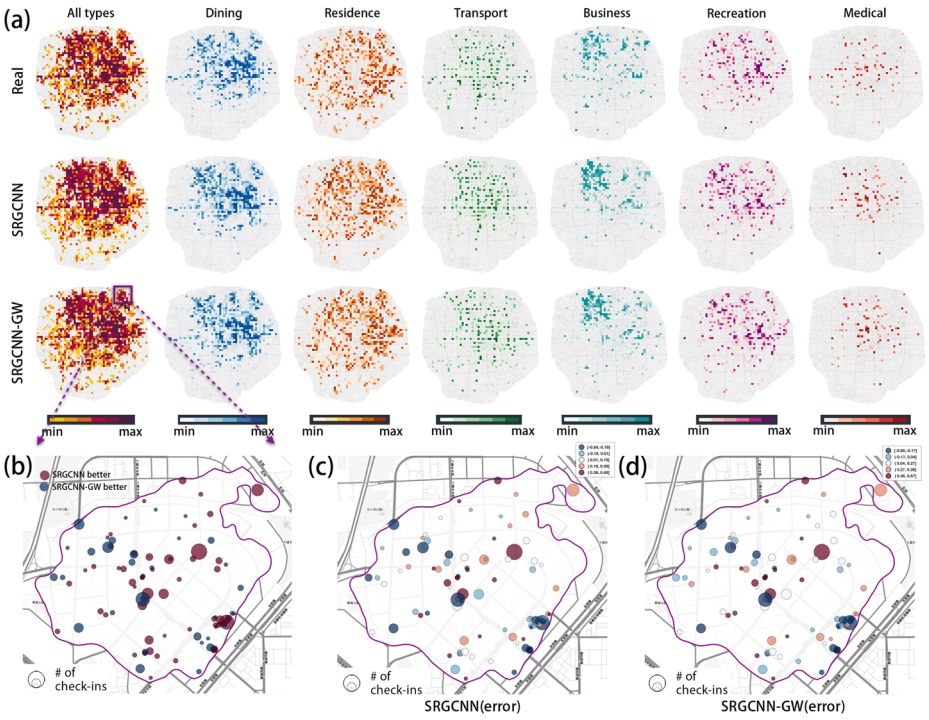


Fig. 9 Visual comparison between the basic SRGCNN model and the SRGCNN-GW model in a sample experiment. **a** The predicted spatial patterns across the check-in types. **b** A zoom-in map at the Wangjing area. **c** The spatial distribution of SRGCNN model's prediction error in Wangjing. **d** The spatial distribution of SRGCNN-GW model's prediction error in Wangjing

patches in Fig. 8b are often slightly larger than those in Fig. 8c. The reason for that is the weight-sharing graph convolution scheme introduces a smoothing effect, which tends to overestimate low-value locations and underestimate the high-value locations. The prediction is towards the middle as the basic SRGCNN model is trying to fit a shared spatial relationship between the urban layout of POI types and the check-in activities.

5.2 SRGCNN v.s. SRGCNN-GW

To compare the basic SRGCNN model and the SRGCNN-GW visually, we display the results of SRGCNN-GW in the sample experiment of Fig. 8a, as illustrated in Fig. 9. The learning rate was set as $\eta = 3 \times 10^{-4}$ for SRGCNN-GW and we recorded the best results in terms of test MAPE in the 8,000 training epochs. The real patterns and predicted patterns are plotted for all POI types as well as for each POI type. The prediction accuracy of SRGCNN-GW on the test POIs (MAPE$\approx$ 11.94%) is slightly worse than that of SRGCNN (MAPE$\approx$ 11.75%). The predicted patterns of SRGCNN-GW are more fractured with smaller clustered patches and stronger local variance. Both models are capable of generating visually similar spatial distributions compared to the ground truth. With respect to different POI types, we found that the predictions on dining, residence and transport POIs are better than business, recreation and medical (Fig. 9a). Fig. 9b displays a zoom-in map at the Wangjing area located in the northeast of the study area. The sizes of POI bubbles reflect

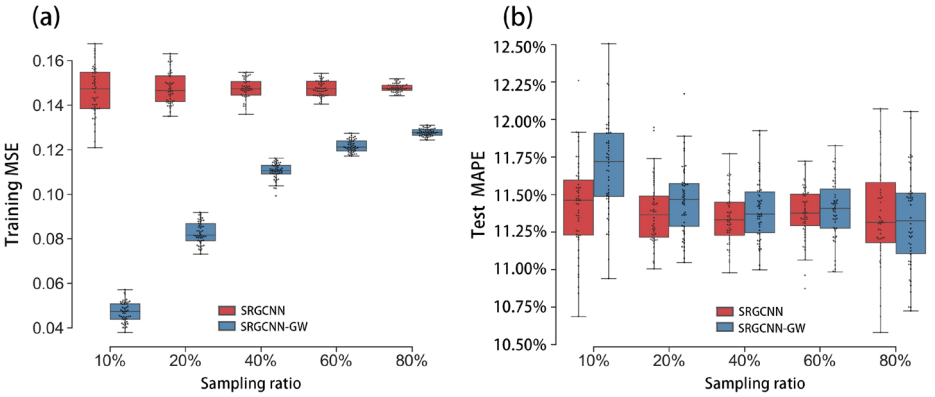


Fig. 10 Statistical comparison between the basic SRGCNN model and SRGCNN-GW across different sampling ratios. **a** MSEs on the training POIs. **b** MAPEs on the test POIs

the corresponding check-in numbers and are consistent with those plotted in Fig. 4. The map shows that SRGCNN performs better when points are clustered (red), while SRGCNN-GW exhibits better prediction at many scattered locations (blue) with lower check-in intensity. The spatial distributions of prediction errors for the two models are plotted in Fig. 9c and d, respectively, indicating similar error distribution across the space.

Previous evaluations are focused only on the 10% training ratio and one selected sample experiment. Furthermore, we conducted experiments based on all five training ratios: 10%, 20%, 40%, 60% and 80%. For each ratio, we repeated fifty parallel simulations to investigate the accuracy and the stability of model performances. Figure 10 illustrates the statistical comparison between the basic SRGCNN model and SRGCNN-GW in box plots.

Seen from Fig. 10a, SRGCNN-GWs outperform SRGCNNs in terms of fitting the training data across all sampling ratios. Basic SRGCNN models have both larger median values and standard deviations of the training MSEs. However, Fig. 10b proves that SRGCNN models actually perform better than SRGCNN-GW in terms of predicting the test data, with lower test MAPEs and smaller standard deviations. It is also interesting to see that the difference between basic SRGCNN models and SRGCNN-GW is more significant when the sampling ratio is low. The SRGCNN model, as a weight-sharing neural network, is more focused on prediction, while SRGCNN-GW, as a geographically weighted model, is more into data fitting. We may prefer to use the basic SRGCNN model when the sampling ratio is lower, while if the data is sufficiently sampled, SRGCNN-GW would be a better model. These findings are in accordance with spatial econometric models, where global spatial regression models are better at modeling and prediction, while GWR models are more powerful in fitting the sampled data for explanation.

5.3 Compared with benchmark spatial regression models

To evaluate the prediction accuracy based on different sampling ratios, we selected three classical benchmark models (SAR, SDM, GWR) that consider spatial lagged effects, so as to be comparable with our proposed SRGCNN-based models. Note that the GWR model is naturally not applicable in predicting multiple locations, we doesn't consider its performance here. For the SRGCNN-based models, we further test the influence of the graph

structure by introducing inverse distance based spatial weights into the graph construction. That is, we constructed a fully-connected graph by calculating the pair-wise euclidean distance (d_{ij}) between each two POIs, and then adopted the inverse distance value (d_{ij}^{-1}) as the corresponding element (w_{ij}) in the spatial weights matrix.

The model performances on predicting test locations are summarized in Table 2. The inverse distance based models are denoted as SRGCNN(*) and SRGCNN-GW(*), respectively. We can see that compared with SAR and SDM, the SRGCNN-based models are significantly less sensitive to the sampling ratio (with lower MAPEs). As the sampling ratio goes down, traditional models exhibit obvious decreasing in accuracies. SDM achieves the best prediction accuracy when the sampling ratio $\geq$ 40%, but is less stable than the deep learning models (higher standard deviations). When the sampling ratio is lower than 40%, SRGCNNs and SRGCNN-GWs are much more accurate and stable. Moreover, it is interesting to notice that the SRGCNN models with the five nearest neighbor graph actually outperform their competitors with the inverse distance graph. This is consistent with the finding in [25] that distance, as a measurement of geographical contexts, might not be as good as topological measurements when modeling the spatial structure of urban check-in activities.

To investigate the fitting abilities across models, we summarized the model performances on fitting all locations in Table 3, where all POIs were used as the training locations and we evaluated the MSEs and MAPEs on the training set (100%). The Morans' I of errors as well as their comparable Z-scored values are presented to help understand how the fitted patterns deviate from the real patterns regarding the spatial autocorrelation. For reference, we add a simple neural network model here as the machine learning baseline, i.e., a multilayer perceptron (MLP) that shares the same training configuration as our SRGCNN-based models, with only one hidden layer, 16 hidden units, and a learning rate $\eta = 10^{-3}$ capped at 5,000 training epochs. As a result, we can see that SRGCNN-GW is the most powerful model in fitting the data distribution, followed by SDM and GWR. Meanwhile, MLP is the worst model among all, with the highest MAPE and a significant positive Z-scored I of the prediction errors. This is because a MLP conducts the regression without considering the spatial structural information underlying data observations, thus the spatial lag effects are not modeled. More importantly, by looking at the Z-scored values of errors' spatial autocorrelation, we notice a significant variation across the models. SAR exhibits strong negative autocorrelation, SDM and the SRGCNN show near zero autocorrelations, while GWR and SRGCNN-GW give a slightly positive autocorrelations.

Please note that we are not trying to say SRGCNNs and SRGCNN-GWs outperform all the benchmarks. Instead, we treat each spatial regression models given its corresponding advantage; one should choose the most suitable model in practice. Despite of this, it is inspiring to just find that SRGCNN-based models are less sensitive to the spatial sampling ratio and could achieve better predictions when the observed data is insufficient. When fitting the whole data distribution, SRGCNN-based models can also reduce errors' global spatial autocorrelation.

6 Discussion

Admittedly, the proposed SRGCNNs can be improved in terms of the model design, the parameter interpretation, and the application, etc. Some potential directions are discussed here as an invitation for future works to explore and extend on our preliminary results.

Table 2 Model performances on predicting test locations

	SAR		SDM		SRGCNN		SRGCNN(*)		SRGCNN-GW		SRGCNN-GW(*)	
Ratio	MAPE	SD	MAPE	SD	MAPE	SD	MAPE	SD	MAPE	SD	MAPE	SD
10%	113.69%	57.47%	18.41%	10.79%	11.42%	0.30%	11.67%	0.35%	11.69%	0.32%	12.48%	0.26%
20%	97.94%	45.13%	12.49%	4.85%	11.38%	0.21%	11.71%	0.30%	11.46%	0.23%	12.48%	0.22%
40%	77.60%	26.08%	10.34%	1.69%	11.34%	0.18%	11.62%	0.22%	11.40%	0.20%	12.42%	0.20%
60%	50.19%	11.84%	9.94%	0.70%	11.38%	0.17%	11.58%	0.25%	11.40%	0.20%	12.36%	0.30%
80%	25.15%	2.67%	10.19%	0.38%	11.37%	0.33%	11.63%	0.38%	11.33%	0.32%	12.46%	0.46%

- (*) denotes a SRGCNN-based model with an inverse-distance weighted graph structure
- MAPE: mean absolute percentage error; SD: standard deviation
- Results are reported based on 50 simulations each sampling ratio

Table 3 Model performances on fitting all locations (sampling ratio: 100%)

	MLP	SAR	SDM	GWR	SRGCNN	SRGCNN-GW
MSE	0.1424	0.1471	0.1394	0.1409	0.1477	0.1313
MAPE	11.89%	10.81%	10.82%	10.90%	11.36%	10.41%
I(errors)	4.80e-2*	−4.29e-2**	−2.70e-2**	−1.02e-2**	6.33e-4**	−4.06e-2**
ZI(errors)	14.37*	−12.60**	0.23**	3.08**	0.25**	1.19**

- *: p <0.05, **: p <0.01

- I(errors) is the errors' spatial autocorrelation measured by Moran's I

- ZI(errors) is the Z-scored value of I(errors) under normality assumption, which is comparable across models

As aforementioned, there are other model types in the spatial regression family that are not considered in the general workflow of spatial regression. For example, the spatial error models (SEM), for which a typical motivation is that unmodeled effects could spill over across spatial units and thus result in spatially correlated errors [59]. We only compared SRGCNNs with the most commonly used spatial lagged models, while in many cases the error terms are also of interest. Future works could utilize different techniques in defining the loss function of SRGCNNs, so as to consider the effects of errors.

An obvious advantage of spatial econometric models, compared with SRGCNNs models, is the interpretability of estimated parameters. It is easy to explain the meaning of coefficients in an econometric model. While for black-box models, the high-dimensional neural network parameters need specialized techniques and domain knowledge for visualization and interpretation, so as to be understood by practitioners. Attention-based graph deep learning [60] and explicit parametric models [24, 61] can be explored in the future to help explain the complex geographical knowledge discovered by SRGCNNs.

Moreover, we fixed the spatial weights matrix to be of the same structure using categorical five nearest neighbors (Fig. 5). The reason we did that is to make sure the output results across different models and experiment settings are comparable. Another distance-based graph structure is added for reference in Table 2 but we didn't discuss much on that. Although the geographical contextual information [62] embedded in different spatial weights matrix may affect the performance of SRGCNN models, it is beyond the scope of this paper. We acknowledge that the upper limit of spatial predictability is naturally different given various geographical contexts, as discussed in [25] and [26]. Future studies are expected to evaluate diverse graph structures in building SRGCNNs, such as bi-directional graphs and dynamic graphs, in order to approaching better prediction results.

With respect to applications, the presented case experiment is just an example that takes the multivariate spatial distribution data as the input to model the dependent variable in space. The task presented in our case is actually challenging for all models: the input features (POI types) are not sufficiently informative in understanding the dependent variable (check-in numbers); the autocorrelation of all-type check-in numbers is near zero, implying a random spatial pattern to be predicted. Therefore, we invite future works in related fields, e.g., geodemography, public health, regional science and transportation to further evaluate SRGCNNs in other scenarios. As an example, we provide an additional experiment in Appendix, where we use a house price dataset in San Diego, C.A., U.S. to demonstrate the data flexibility of SRGCNNs and to test the influence of incorporating more dependent variables in SRGCNNs.

7 Conclusions

In this paper, we demonstrate the feasibility of using graph convolutional neural networks to perform spatial regression and prediction. Theoretically, we find encouraging evidences supporting that GCNNs is suitable for spatial regression, thanks to its propagation mechanisms, spatial locality learning nature and the semi-supervised training strategy. Accordingly, we propose the spatial regression graph convolutional neural networks (SRGCNNs) as a deep learning paradigm to cope with a range of geographical tasks where spatial multivariate data needs regression modeling and prediction. Compared with GCNNs applications in other domains, SRGCNNs have a specific emphasis on the conceptual mapping between GCNNs' graph structure and the spatial weights matrix in spatial regression, so as to capture the spatial lagged effects in observed geographic variables.

Case experiments in the urban context reveal that a basic SRGCNN model could achieve a superior prediction error near 11.40% in terms of MAPE compared with classic spatial econometric models (SAR: 113.69%, SDM: 18.41%), when the sampling ratio is only 10%. An alternative way to design the SRGCNN model is also provided to enable the geographically weighted learning in SRGCNNs, implying that SRGCNNs can be modified to capture the spatial non-stationarity of relationships. Through parallel experiments across sampling ratios, we find that compared with benchmark models, SRGCNN-based models are much less sensitive to the sampling in terms of standard deviation and could achieve better predictions when the observed data is insufficient.

Our work is an attempt to incorporate artificial intelligence into the traditional paradigm of spatial analytics. We offer to bridge the methodological gap between graph deep learning and spatial regression. The proposed SRGCNNs paradigm serves as an example to illustrate how spatial analytical methods can be enriched with more possibilities when combined with state-of-the-art deep learning models, and to enlighten future research at the front of GeoAI.

Acknowledgements The authors gratefully acknowledge editors, the anonymous reviewers, Dr. Tao Cheng, Dr. Yang Zhang, Dr. Ximeng Cheng, and Dr. Fan Zhang for their helpful comments. This work was partially supported by the New Faculty Set-up Funding of College of Liberal Arts, University of Minnesota (1000-10964-20042-5672018). Prof. Yu Liu is supported by the National Key Research and Development Program of China (2017YFB0503602) and the National Natural Science Foundation of China (41625003).

Appendix

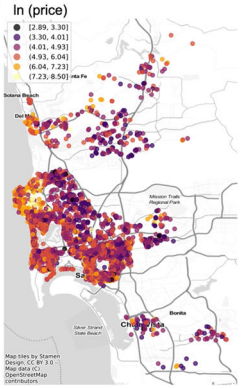
In this section, we discuss SRGCNNs in a more typical spatial regression scenario: house price modeling. Utilizing a house rent price dataset at San Diego, C.A., U.S., we carefully evaluate the regression accuracies across models, and investigate model performances given different sets of explanatory variables.

A.1 Data description and feature selection

The open data behind the Inside Airbnb site[3] was collected for this appendix experiment. The data is sourced from publicly available information in the Airbnb site, which includes the daily rent price of listed properties and many additional attributes for the listings.

[3] http://insideairbnb.com/about.html

(a)



(b)

id	price	ln(price)	accommodates	bathrooms	bedrooms	beds	coastal	pg_Apartment	pg_Condominium	pg_House	pg_Other	pg_Townhouse	rt_Entire_home/apt	rt_Private_room	rt_Shared_room
0	425.0	6.052089168924420	5	2.0	2.0	2.0	0	0	0	1	0	0	1	0	0
1	205.0	5.323009979384100	6	1.0	2.0	4.0	1	0	1	0	0	0	1	0	0
2	99.0	4.595119850134590	2	1.0	1.0	1.0	0	1	0	0	0	0	0	1	0
3	72.0	4.276666119016060	2	1.0	1.0	1.0	0	0	0	1	0	0	0	1	0
4	55.0	4.007333185232470	2	1.0	1.0	1.0	0	0	0	1	0	0	0	1	0
5	65.0	4.174387269895640	2	0.0	1.0	1.0	0	0	0	0	1	0	1	0	0
6	160.0	5.075173815233830	2	1.0	1.0	1.0	0	0	0	1	0	0	1	0	0
7	120.0	4.787491742782050	2	1.0	1.0	1.0	1	0	0	1	0	0	0	1	0
8	85.0	4.442651256490320	2	1.0	1.0	1.0	1	0	0	1	0	0	0	1	0
9	70.0	4.248495242049360	2	1.0	1.0	1.0	1	0	0	0	0	0	1	0	0
10	40.0	3.688879454113940	2	1.0	1.0	1.0	0	1	0	0	0	0	0	1	0
11	90.0	4.499809670330270	2	1.0	1.0	1.0	0	1	0	0	0	0	1	0	0
12	250.0	5.521460917862550	6	2.5	3.0	4.0	1	0	0	1	0	0	1	0	0
13	120.0	4.787491742782050	4	2.5	2.0	2.0	0	1	0	0	0	0	1	0	0
14	42.0	3.737669618283700	2	1.0	1.0	1.0	0	0	1	0	0	0	0	1	0
15	47.0	3.850147601710600	2	1.0	1.0	1.0	0	0	1	0	0	0	0	1	0
16	105.0	4.653960350157520	1	1.0	1.0	1.0	1	0	0	1	0	0	0	1	0
17	130.0	4.867534450455580	2	1.0	1.0	1.0	1	0	0	1	0	0	0	1	0
18	135.0	4.90527477843943	2	1.0	1.0	1.0	1	0	0	1	0	0	0	1	0
19	159.0	5.068904202202300	2	1.0	1.0	1.0	0	0	0	0	1	0	1	0	0
20	209.0	5.342334251964810	7	2.5	3.0	5.0	1	1	0	0	0	0	1	0	0
21	89.0	4.488636369073214	2	1.0	1.0	1.0	1	0	0	1	0	0	0	1	0
22	178.0	5.181783550292090	2	1.0	1.0	1.0	1	0	0	1	0	0	1	0	0
23	99.0	4.595119850134590	2	1.0	1.0	1.0	0	1	0	0	0	0	0	1	0
24	221.0	5.398162705157500	8	2.0	3.0	6.0	1	0	0	1	0	0	1	0	0
25	60.0	4.094344562221	2	1.0	0.0	1.0	1	1	0	0	0	0	1	0	0
26	99.0	4.595119850134590	2	1.0	0.0	1.0	1	1	0	0	0	0	1	0	0
27	99.0	4.595119850134590	3	1.0	1.0	2.0	0	1	0	0	0	0	1	0	0
28	49.0	3.891820298106300	1	1.0	1.0	1.0	0	0	0	1	0	0	0	1	0

Fig. 11 Daily price in local currency for Airbnb listings in San Diego, U.S., collected on July, 07, 2016. **a** Map visualization of $ln(price)$. **b** Listing's attributes as the interested variables

To compare SRGCNNs with traditional spatial regression models, we will examine some information in all Airbnb listings in San Diego, C.A., U.S. on July, 07, 2016.

In Fig. 11, we visualize the logarithm prices to the base e ($ln(price)$) in Fig. 11a. The map utilizes a percentile color scheme to highlight both the extreme high prices (head 1%, in yellow) and low prices (tail 1%, in black). There are 6,110 collected Airbnb listings in total, the prices are obviously spatial autocorrelated, with high-value clusters as well as low-value clusters within the study area. In Fig.11b, we present the house characteristics that are of interest in this experiment, with both continuous variables (e.g., number of accommodate people as in "*accommodates*", and number of beds as in "*beds*") and categorical variables (e.g., rent type as in "*rt_XXX*", and property groupy in "*pg_XXX*"). Also, we include a binary variable named "*coastal*" to indicate whether a house is near the ocean.

In the following regression analysis, we will use $ln(price)$ as the dependent variable $\boldsymbol{y}$, and examine two independent sets $\boldsymbol{X}$ (four variables) and $\boldsymbol{X_+}$ (eleven variables). The basic independent variable set $\boldsymbol{X}$ = {"*accommodates*", "*bathrooms*", "*bedrooms*", "*beds*"} contains only the four continuous intrinsic characteristics: number of accommodate people, number of bedrooms, number of bathrooms, and number of beds. While the extended independent variable set $\boldsymbol{X_+}$ = {"*accommodates*", "*bathrooms*", "*bedrooms*", "*beds*", "*rt_Private_room*", "*rt_Shared_room*", "*pg_House*", "*pg_Condominium*", "*pg_Townhouse*", "*pg_Other*", "*coastal*"} contains additional characteristics of rent type, property group, and the coastal indicator. The rent types are used as dummy variables, denoting whether a listing belongs to a private room, a shared room, or an entire home. The property groups are also used as dummy variables, indicating whether the listing is an apartment, a condom, a townhouse, a single family house or others.

Note that it is possible to include other surrounding environmental context as the independent variables, such as the distance to the highways and number of parks in the neighborhood to further improve the regression accuracy. However, the selection of informative feature variables is beyond the scope of our paper. Here, we just provide two different sets of independent variables in order to shed light on the influence of feature engineering in SRGCNN-based models. Future applications are invited to test out SRGCNNs with different feature combinations in specialized tasks.

A.2 Model training

The regressions on prices at all locations (100% training ratio) are performed using linear regression model (LR), spatial autoregressive model (SAR), and the SRGCNN-GW model (19). For simplicity, models with the additional variable set $\boldsymbol{X}_+$ are referred to as LR+, SAR+, and SRGCNN-GW+, respectively. We choose SRGCNN-GW model here rather than the basic SRGCNN model because SRGCNN-GW is better at fitting the training dataset, while the basic SRGCNN model is better for prediction (as discussed in Section 5.2).

We consider k=20 nearest neighbors for each location to construct the spatial weights matrix in SAR and the graph structure in SRGCNN-GW. It is optional to change the way of defining the spatial structure, e.g., a different k, or using other measurements such as distance, queen adjacency. We won't dive into this because the influence of geographic contexts on spatial regression is another topic to investigate [25] and it is beyond the scope of this paper.

We adopt similar training settings as introduced in Section 4.2.3. The learning rate is changed to $\eta = 3 \times 10^{-2}$. Training epochs are capped at 15,000 for SRGCNN-GW and 18,000 for SRGCNN-GW+. We record the best results among all epochs. The MSE Loss and MAPE during the training process are plotted in Fig. 12. The hidden feature units are set to be $4 \times 8 = 32$ for SRGCNN-GW and $11 \times 8 = 88$ for SRGCNN-GW+, considering the different input features provided in $\boldsymbol{X}$ and $\boldsymbol{X}_+$. As can be seen, SRGCNN-GW reaches the lowest MAPE at epoch 12,161, while SRGCNN-GW+ reaches its lowest MAPE at epoch 15,508. After that, both models exhibit overfitting as the MAPE starts to rise up again. For the whole training, SRGCNN-GW+ converged slightly slower (with more epochs) compared to SRGCNN-GW, because there are more feature parameters to be learned in the geographic weighted graph convolutions.

A.3 Evaluation of the results

The results across all models are summarized in Table 4. We report the R^2 and MAPE as two metrics to evaluate the goodness of model fitting. Also, the Z-scored Morans' I value

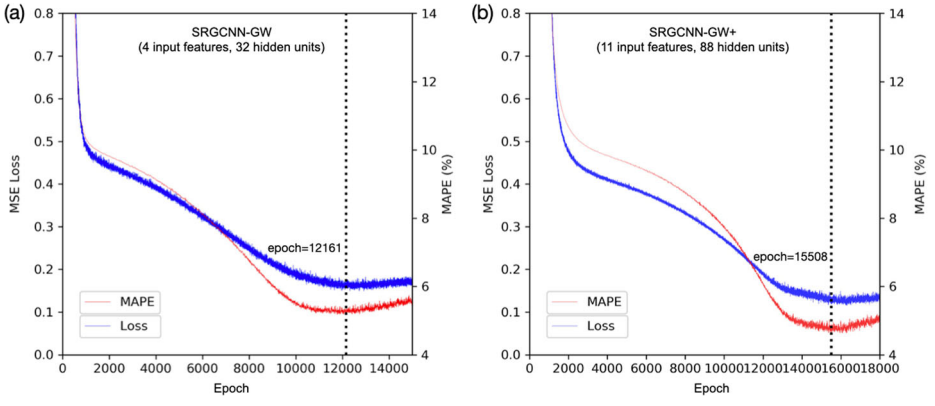


Fig. 12 Training process of the SRGCNN-GW models (**a**) SRGCNN-GW: $\boldsymbol{X}$ as the explanatory variables. **b** SRGCNN-GW+: $\boldsymbol{X}_+$ as the explanatory variables

Table 4 Model performances on fitting all prices (sampling ratio:100%)

	Data	LR	LR+	SAR	SAR+	SRGCNN-GW	SRGCNN-GW+
R^2	–	0.5583	0.6796	0.6218	0.7057	0.8323	0.8664
MAPE	–	6.54%	5.42%	5.98%	5.04%	4.17%	3.76%
ZI(errors)	–	50.15***	31.14***	10.54***	7.50***	16.20***	33.43***
ZI($ln(price)$)	74.25***	47.70***	70.77***	104.70***	100.21***	119.43***	109.24***

- ***: p <0.001
- ZI(errors): the Z-scored Morans' I value of prediction errors under normality assumption, which is comparable across models
- ZI($ln(price)$): the Z-scored Morans' I value of $ln(price)$

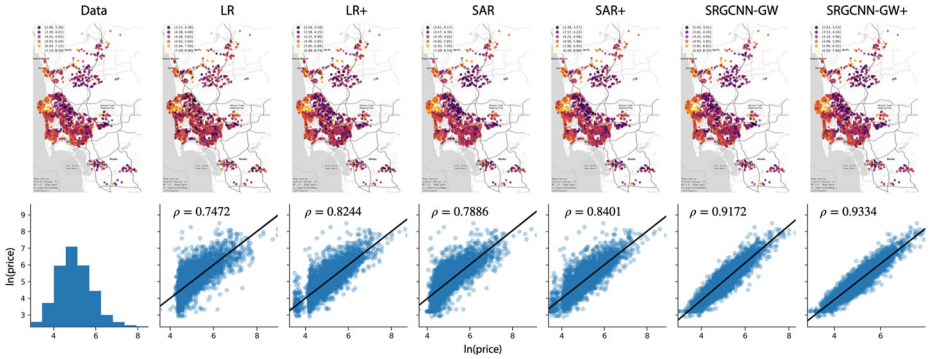


Fig. 13 Model comparison in maps and scatterplots

of prediction errors and the fitted $ln(price)$ are included to indicate how the models capture the spatial effects in data.

It is encouraging to find that SRGCNN-GW significantly outperforms LR and SAR regarding both R^2 and MAPE. Using the basic independent variable set $\boldsymbol{X}$, we can see that LR, the non-spatial linear regression model, can only explain about 56% of the real process; SAR, the most common used spatial lagged model, increases the goodness of fit to about 62%; SRGCNN-GW model, however, reaches a much higher goodness of fit around 83%. By adding more explanatory information, all models exhibit better results using $\boldsymbol{X}_+$. LR increases from 55% to 68%, SAR increases from 62% to 71%, and SRGCNN increases from 83% to 87%. Since SRGCNN-based models consider more complex spatial relationships during the modeling, the influence of additional independent variables is less than traditional models such as linear regression and spatial lagged models. With respect to the MAPE, conclusions are exactly the same, SRGCNN-GW+ reaches the lowest fitting error at only 3.76%. Seen from the Z-scored autocorrelations, SAR and SAR+ are better at handling the spatial errors. SRGCNN-GW models also have lower error autocorrelations compared to LR models. SRGCNN-GW models reports higher global autocorrelation of the fitted price than SAR, indicating an explicitly modeling of spatial structure in its graph convolution layers.

Results are also compared in Fig. 13. The spatial distributions of $ln(price)$ are plotted in the first row for both the original data and the model predictions. The scatter plots and the Pearson correlation coefficients ρ are presented in the second row to further evaluate the models. As shown, SRGCNN-GW+ has done an outstanding job fitting the price data, with a modeled spatial pattern really similar to the original one and a highest Pearson correlation $\rho = 0.9334$. The number of input features does have influences on the modeling accuracies, but it is still not clear on how to select informative variables for SRGCNN models. Future works are to develop specialized methods for the visualization and analysis of complex feature parameters in SRGCNN models with regard to regression statistics.

References

1. Paelinck JHP, Klaassen LH, Ancot J-P, Verster ACP (1979) Spatial econometrics, vol 1. Saxon House
2. Anselin L (1988) Spatial econometrics: Methods and models, vol 4. Springer Science & Business Media, Berlin

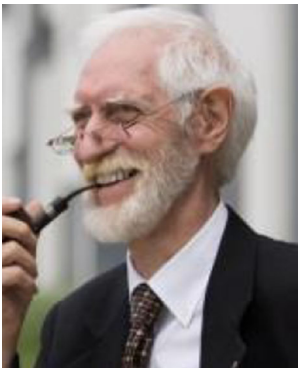
3. LeSage JP (1997) Regression analysis of spatial data. J Region Anal Policy 27:83–94
4. LeSage JP, Fischer MM (2008) Spatial growth regressions: model specification, estimation and interpretation. Spatial Economic Analysis 3:275–304
5. Lehmann A, Overton JMcC, Leathwick JR (2002) Grasp: generalized regression analysis and spatial prediction. Ecol Modell 157:189–207
6. Anselin L (2010) Thirty years of spatial econometrics. Papers Region Sci 89(1):3–25
7. Fischer MM, Wang J (2011) Spatial data analysis: models, methods and techniques. Springer Science & Business Media
8. Liu Y, Liu X, Gao S, Gong L, Kang C, Zhi Y, Chi G, Shi L (2015) Social sensing: A new approach to understanding our socioeconomic environments. Ann Assoc Amer Geograph 105:512–530
9. Vatsavai R, Chandola V (2016) Guest editorial: big spatial data. GeoInformatica 20:797–799
10. Cheng T, Adepeju M (2014) Modifiable temporal unit problem (mtup) and its effect on space-time cluster detection. PloS one 9:e100465
11. Haworth James, Cheng Tao (2012) Non-parametric regression for space–time forecasting under missing data. Comput Environ Urban Syst 36:538–550
12. Kelejian HH, Prucha IR (2007) The relative efficiencies of various predictors in spatial econometric models containing spatial lags. Region Sci Urban Econ 37:363–374
13. Fischer MM (1998) Computational neural networks: a new paradigm for spatial analysis. Environ Plann A 30:1873–1891
14. Gu Y, Wylie BK, Boyte SP, Picotte J, Howard DM, Smith K, Nelson KJ (2016) An optimal sample data usage strategy to minimize overfitting and underfitting effects in regression tree models based on remotely-sensed data. Remote Sens 8:943
15. Mennis J, Guo D (2009) Spatial data mining and geographic knowledge discovery–an introduction. Comput Environ Urban Syst 33:403–408
16. Reichstein M, Camps-Valls G, Stevens B, Jung M, Denzler J, Carvalhais N et al (2019) Deep learning and process understanding for data-driven earth system science. Nature 566:195–204
17. Janowicz K, Gao S, McKenzie G, Hu Y, Bhaduri B (2020) GeoAI: spatially explicit artificial intelligence techniques for geographic knowledge discovery and beyond. Int J Geograph Inf Sci 34:625–636
18. Li W, Hsu C-Y (2020) Automated terrain feature identification from remote sensing imagery: a deep learning approach. Int J Geograph Inf Sci 34(4):637–660
19. Yan X, Ai T, Yang M, Yin H (2019) A graph convolutional neural network for classification of building patterns using spatial vector data. ISPRS J Photogramm Remote Sens 150:259–273
20. Zhang F, Wu L, Zhu D, Liu Y (2019) Social sensing from street-level imagery: A case study in learning spatio-temporal urban mobility patterns. ISPRS J Photogramm Remote Sens 153:48–58
21. Liu P, De Sabbata S (2021) A graph-based semi-supervised approach to classification learning in digital geographies. Comput Environ Urban Syst 86:101583
22. Zhu D, Cheng X, Zhang F, Yao X, Gao Y, Liu Y (2020) Spatial interpolation using conditional generative adversarial neural networks. Int J Geograph Inf Sci 34:735–758
23. Xing X, Huang Z, Cheng X, Zhu D, Kang C, Zhang F, Liu Y (2020) Mapping human activity volumes through remote sensing imagery. IEEE J Sel Top Appl Earth Observ Remote Sens 13:5652–5668
24. Du Z, Wang Z, Wu S, Zhang F, Liu R (2020) Geographically neural network weighted regression for the accurate estimation of spatial non-stationarity. Int J Geograph Inf Sci 34:1353–1377
25. Zhu D, Zhang F, Wang S, Wang Y, Cheng X, Huang Z, Liu Y (2020) Understanding place characteristics in geographic contexts through graph convolutional neural networks. Ann Amer Assoc Geograph 110:408–420
26. Xiao L, Lo S, Zhou J, Liu J, Yang L (2020) Predicting vibrancy of metro station areas considering spatial relationships through graph convolutional neural networks: The case of shenzhen, china. Environ Plann B: Urban Anal City Sci:2399808320977866
27. Schmidhuber J (2015) Deep learning in neural networks: An overview. Neural networks 61:85–117
28. Bronstein MM, Bruna J, LeCun Y, Szlam A, Vandergheynst P (2017) Geometric deep learning: going beyond euclidean data. IEEE Signal Process Mag 34:18–42
29. LeCun Y, Bengio Y, Hinton G (2015) Deep learning. Nature 521:436–444
30. Defferrard M, Bresson X, Vandergheynst P (2016) Convolutional neural networks on graphs with fast localized spectral filtering. In: Advances in Neural Information Processing Systems, pp 3844–3852
31. Niepert M, Ahmed M, Kutzkov K (2016) Learning convolutional neural networks for graphs. In: International conference on machine learning, pp 2014–2023
32. Hamilton W, Ying Z, Leskovec J (2017) Inductive representation learning on large graphs. In: Advances in neural information processing systems, pp 1024–1034
33. Fan RKC (1997) Spectral graph theory. American Mathematical Society

34. Hammond DK, Vandergheynst P, Gribonval R (2009) Wavelets on graphs via spectral graph theory. Appl Comput Harmon Anal 30:129–150
35. Bruna J, Zaremba W, Szlam A, Lecun Y (2014) Spectral networks and locally connected networks on graphs. In: International Conference on Learning Representations
36. Kipf TN, Welling M (2017) Semi-supervised classification with graph convolutional networks. In: International Conference on Learning Representations
37. Chen R, Wang X, Zhang W, Zhu X, Li A, Yang C (2019) A hybrid CNN-LSTM model for typhoon formation forecasting. GeoInformatica 23:375–396
38. Zhao L, Song Y, Zhang C, Liu Y, Wang P, Lin T, Deng M, Li H (2019) T-GCN: A temporal graph convolutional network for traffic prediction. IEEE Transactions on Intelligent Transportation Systems 21(9):3848–3858
39. Zhang Y, Cheng T, Ren Y, Xie K (2020) A novel residual graph convolution deep learning model for short-term network-based traffic forecasting. Int J Geograph Inf Sci 34:969–995
40. Bai J, Zhu J, Song Y, Zhao L, Hou Z, Du R, Li H (2021) A3t-gcn: Attention temporal graph convolutional network for traffic forecasting. ISPRS Int J Geo-Inf 10:485
41. Bui K-HN, Cho J, Yi H (2021) Spatial-temporal graph neural network for traffic forecasting: An overview and open research issues. Appl Intell:1–12
42. Hu S, Gao S, Wu L, Xu Y, Zhang Z, Cui H, Gong X (2021) Urban function classification at road segment level using taxi trajectory data: A graph convolutional neural network approach. Comput Environ Urban Syst 87:101619
43. Griffith DA, Paelinck JHP (2011) Non-standard spatial statistics and spatial econometrics. Springer Science & Business Media
44. Arbia G (2014) A primer for spatial econometrics with applications in r. Springer
45. Anselin L, Rey SJ (2014) Modern spatial econometrics in practice: A guide to geoda, geodaspace and pysal. GeoDa Press LLC
46. Kelejian H, Piras G (2017) Spatial econometrics. Academic Press
47. Yamagata Y, Seya H (2019) Spatial analysis using big data: Methods and urban applications. Academic Press
48. LeSage JP, Pace RK (2009) Introduction to spatial econometrics. CRC Press/Taylor & Francis, Boca Raton
49. Fotheringham AS, Yang W, Kang W (2017) Multiscale Geographically Weighted Regression (MGWR). Ann Amer Assoc Geograph 107:1247–1265 (en)
50. Ord K (1975) Estimation methods for models of spatial interaction. J Amer Stat Assoc 70:120–126
51. Kelejian HH, Prucha IR (1998) A generalized spatial two-stage least squares procedure for estimating a spatial autoregressive model with autoregressive disturbances. J Real Estate Finance Econ 17(1):99–121
52. Kelejian HH, Prucha IR (1999) A generalized moments estimator for the autoregressive parameter in a spatial model. Int Econ Rev 40(2):509–533
53. Goodfellow I, Bengio Y, Courville A, Bengio Y (2016) Deep learning, vol 1. MIT press Cambridge
54. Paola JD, Schowengerdt RA (1995) A detailed comparison of backpropagation neural network and maximum-likelihood classifiers for urban land use classification. IEEE Trans Geosci Remote Sens 33(4):981–996
55. Vatsavai RR, Bhaduri B (2011) A hybrid classification scheme for mining multisource geospatial data. GeoInformatica 15:29–47
56. Zhu D, Wang N, Wu L, Liu Y (2017) Street as a big geo-data assembly and analysis unit in urban studies: A case study using beijing taxi data. Appl Geograph 86:152–164
57. Long Y, Liu X (2013) How mixed is beijing, china? a visual exploration of mixed land use. Environ Plann A 45:2797–2798
58. Chen L, Gao Y, Zhu D, Yuan Y, Liu Y (2019) Quantifying the scale effect in geospatial big data using semi-variograms. PloS one 14:e0225139
59. Anselin L (2009) Spatial regression. In: Fotheringham AS, Rogerson PA (eds) The SAGE Handbook of Spatial Analysis. SAGE Publications, Los Angeles, pp 255–275
60. Veličković P, Cucurull G, Casanova A, Romero A, Lio P, Bengio Y (2017) Graph attention networks. arXiv:1710.10903
61. Wu S, Wang Z, Du Z, Huang B, Zhang F, Liu R (2020) Geographically and temporally neural network weighted regression for modeling spatiotemporal non-stationary relationships. Int J Geograph Inf Sci:1–27
62. Kwan M-P (2012) The uncertain geographic context problem. Ann Assoc Amer Geograph 102:958–968

Di Zhu is an Assistant Professor of GIScience in the Department of Geography, Environment and Society at University of Minnesota, Twin Cities. He received the B.S. and Ph.D. degrees in Cartology and GIS and the B.A. in Economy from Peking University. His research interests include GeoAI, spatial analytics, social sensing and urban complexities.

Yu Liu received the B.S., M.S., and Ph.D. degrees from Peking University in 1994, 1997, and 2003, respectively. He is currently a Professor of GIScience with the Institute of Remote Sensing and Geographical Information Systems, Peking University. His research interest mainly concentrates on the humanities and social sciences based on big geo-data.

Xin Yao is a research scientist at Alibaba group Beijing. He received his PhD degree from Peking University and a B.S. degree from Wuhan University. His primary research interests include spatial data mining and geographical information visualization.



Manfred M. Fischer is Professor Emeritus in the Department of SocioEconomics, Vienna University of Economics and Business, and Adjunct Professor in the Institute for Geographical Sciences and National Resources Research, Chinese Academy of Sciences, Beijing. His research interests are regional science, economic geography, spatial econometrics, and spatial data analysis.

Affiliations

Di Zhu[1] · Yu Liu[2] · Xin Yao[3] · Manfred M. Fischer[4]

Yu Liu
liuyu@urban.pku.edu.cn

Xin Yao
yaox@alibaba-inc.com

Manfred M. Fischer
manfred.fischer@wu.ac.at

1 Department of Geography, Environment and Society, University of Minnesota, Minneapolis, MN, USA

2 Institute of Remote Sensing and Geographical Information Systems, Peking University, Beijing, People's Republic of China

3 Alibaba Group, Beijing, People's Republic of China

4 Department of Socioeconomics, Vienna University of Economics and Business, Vienna, Austria