Nonparametric spatial autoregressive model using deep neural networks

November 14, 2023

Submitted by

Souvik Roy (221433)

Shubham Kr. Rajak (221423)

Abhraraj Haldar (221255)

Supervisor

Dr. Arnab Hazra

1 Introduction

Spatial data often exhibits intricate dependencies and non-linear relationships that are challenging to capture using traditional statistical models. In this project, we explore the application of Non-parametric Spatial Autoregressive Models (NSAR) enhanced by deep neural networks to address the complexities inherent in spatial datasets. Unlike conventional parametric spatial models that assume specific functional forms for spatial relationships, nonparametric models leverage the flexibility and power of deep learning to adaptively capture intricate spatial dependencies without imposing strong assumptions.

The project aims to contribute to the field of spatial analysis by showcasing the advantages of employing nonparametric spatial models with deep neural networks. Insights gained from the project can inform researchers and practitioners working with spatial data, offering a more flexible and powerful tool for understanding and predicting spatial phenomena.

2 Non-parametric spatial autoregression neural network

2.1 Spatial autoregression

Spatial autoregression, often referred to as **spatial autoregressive models** or SAR models, is a class of statistical models used to analyze spatial data where observations at one location are influenced by the values at nearby locations. Spatial autoregressive models differ from standard regression models by having the response variable Y, on both sides of the equal sign, in order to account for within variable correlation.

Given a vector of dependent variable $Y \in \mathbb{R}^n$, a matrix of independent variable $X \in \mathbb{R}^{n \times p}$ and a spatial weight matrix $W \in \mathbb{R}^{n \times n}$, we can establish the following SAR model

$$Y = \rho WY + X\beta + \epsilon \tag{1}$$

where $\epsilon \in \mathbb{R}^n$ is the error term, $\rho \in \mathbb{R}$ and $\beta \in \mathbb{R}^p$ are the parameters to be estimated.

Spatial autoregressive parameter ρ reflects the strength of the spatial dependencies between the elements of the dependent variable. A positive ρ suggests positive spatial autocorrelation, meaning that similar values are more likely to be found near each other. A negative ρ suggests negative spatial autocorrelation. When $\rho = 0$, this model is reduced to the ordinary linear regression equation.

Spatial weight matrix W is a square matrix where each element W_{ij} represents the weight or strength of the spatial relationship between spatial units i and j.

The term ρWY is often referred to as the spatial lag, indicating that the value at a given location is influenced by the spatially lagged values at neighboring locations, weighted by the spatial weight matrix W.

2.2 Artificial neural network

An Artificial Neural Network (ANN) is a computational model inspired by the way biological neural networks in the human brain work. It's a machine learning model that consists of interconnected nodes, also known as neurons. ANNs are used for various tasks, including classification, regression, pattern recognition, and decision-making.

Components and concepts of Neural Network:

- **Neuron/Node**: The basic computational unit of an ANN is a neuron. It takes input, performs a weighted sum, applies an activation function, and produces an output.
- **Input Layer**: The input layer of an ANN receives the initial input features. Each neuron in this layer represents a feature of the input data.
- **Hidden Layer**: Hidden layer is the set of neurons where all the computations are performed on the input data. There can be any number of hidden layers in a neural network. The simplest network consists of a single hidden layer.
- **Weights and Biases**: Each connection between neurons has a weight associated with it, representing the strength of the connection. After assigning the weights, a bias variable is added.
- **Activation Function**: The activation function is a nonlinear transformation that is applied to the input before sending it to the next layer of neurons. The importance of the activation function is to inculcate nonlinearity in the model.
- **Output Layer**: The output layer is the output/conclusions of the model derived from all the computations performed. There can be single or multiple nodes in the output layer.

2.3 Nonparametric spatial autoregression neural network

When spatial effects are present in the data, we often use spatial autoregressive models to explore the relationship between the response variable Y and the p-dimensional predictors, as shown in Eq. (1). However, the ordinary SAR model cannot solve the problem of nonlinear regression. We therefore constructed the nonparametric spatial autoregression model by combining nonparametric regression

$$Y = \rho WY + g(x_1, x_2, ..., x_p) + \epsilon \tag{2}$$

where ϵ is the random error term and $g(\cdot)$ is regression function.

This model has many evolved forms, including spatial linear autoregressive, spatial nonlinear autoregressive, partially linear additive autoregressive and SAR with nonparametric endogenous effect models. The model has the advantages of nonparametric, self-learning and having no model setting error, and it also takes into account the spatial effects of the research object, thus improving the accuracy of the model.

The process of building a nonparametric spatial autoregression neural network is mainly to introduce spatial autocorrelation of data by adding a spatial lag vector WY to the input layer of the neural network. On the one hand, spatial autoregressive can deal with the spatial data regression problem, and on the other hand, neural network can fit the nonlinear structure in the model well without setting explicit functional forms. Combining the above two advantages, we hope the model can solve some nonlinear spatial data modeling problems, so as to avoid the model setting errors caused by the spatial autoregressive model setting spatial linearity in advance. We expand Eq. (2) in the form of a matrix multiplication

$$
\begin{aligned}
Y &= \rho WY + g(x_1, x_2, ..., x_p) + \epsilon \\
&= \rho \begin{bmatrix} 0 & w_{12} & \cdots & w_{1n} \\ w_{21} & 0 & \cdots & w_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ w_{n1} & w_n 2 & \cdots & 0 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} + \begin{bmatrix} g(x_{11}, x_{21}, ..., x_{p1}) \\ g(x_{12}, x_{22}, ..., x_{p2}) \\ \vdots \\ g(x_{1n}, x_{2n}, ..., x_{pn}) \end{bmatrix} + \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \epsilon_n \end{bmatrix} \\
&= \rho \begin{bmatrix} 0 + w_{12}y_2 + \cdots + w_{1n}y_n \\ w_{21}y_1 + 0 + \cdots + w_{2n}y_n \\ \vdots \\ w_{n1}y_1 + w_n 2y_2 + \cdots + 0 \end{bmatrix} + \begin{bmatrix} g(x_{11}, x_{21}, ..., x_{p1}) \\ g(x_{12}, x_{22}, ..., x_{p2}) \\ \vdots \\ g(x_{1n}, x_{2n}, ..., x_{pn}) \end{bmatrix} + \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \epsilon_n \end{bmatrix}
\end{aligned} \tag{3}
$$

Therefore, our predicted value for $\hat{y_i}$

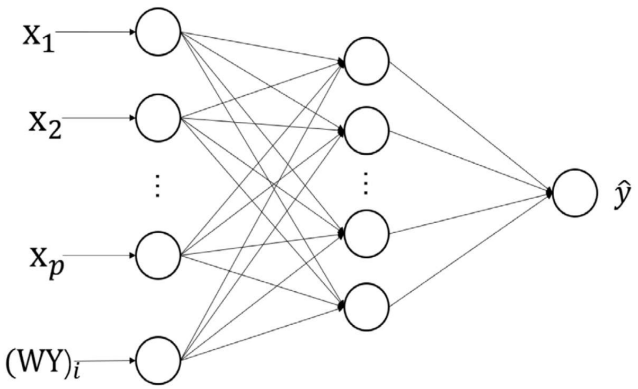

Figure 1: The neural network structure

$$
\hat{y_i} = \rho(w_{i1}y_1 + w_{i2}y_2 + \cdots + 0 + \cdots + w_{1n}y_n) + g(x_{1i}, x_{2i}, ..., x_{pi}) + \epsilon_i \tag{4}
$$

Since the feature that the diagonal elements of the spatial weight matrix are zero, the value of $\hat{y_i}$ is related to $y_j (j = 1, 2, ..., n, j \neq i)$ and X but independent of itself. So we use this feature to construct the model as follows

$$\begin{aligned} Y_i &= f(x_{1i}, x_{2i}, ..., x_{pi}, (WY)_i) + \epsilon \\ &= f(x_{1i}, x_{2i}, ..., x_{pi}, x_{(p+1)i}) + \epsilon, i = 1, 2, ..., n \end{aligned} \tag{5}$$

where $(WY)_i$ denotes the ith element of vector WY and we unify it as $x_{(p+1)i}$, n is the number of sample, p is the number of covariates. The neural network structure graph is shown in Fig. 1.

First, an output from the jth hidden layer node $g_{j,l}(X)$ is obtained by adding the hidden layer bias $b_l^{(h)}$ to the inner product of the predictors and hidden layer weights $w_{jl}^{(h)}$, and then applying the heavy path activation function $f^{(h)}$. The output of the hidden layer neurons is

$$g_{j,l}(X) = f^{(h)}\left(\sum_{k=1}^{p+1} x_{ik} w_{kl}^{(h)} + b_l^{(h)}\right) \tag{6}$$

where $\mathbf{w}^{(h)} = (w_{11}^{(h)}, w_{12}^{(h)}, ..., w_{p+1,L}^{(h)})'$ is a weight vector of hidden layer, $\mathbf{b}^{(h)} = (b_1^{(h)}, b_2^{(h)}, ..., b_L^{(h)})'$ is a bias vector of the hidden layer, L is the number of hidden layer nodes, and $f^{(h)}$ denotes activation function, we choose relu activation function. The output layer neurons receive input from the hidden layer neurons and calculate the output of the entire network output.

$$Q_{y_i}(X) = f^{(o)}\left(\sum_{l=1}^{L} w_l^{(o)} g_{j,l}(X) + b^{(o)}\right) \tag{7}$$

where $\mathbf{w}^{(o)} = (w_1^{(o)}, w_2^{(o)}, ..., w_L^{(o)})'$ is an output layer weight vector, $b^{(o)}$ is an output layer bias, and $f^{(o)}$ is the transfer function of the output layer, and we use a linear function. The loss function is then minimized by a back propagation algorithm

$$Loss = \frac{1}{N}\sum_{i=1}^{n} (y_i - Q_{y_i}(X))^2 \tag{8}$$

where $X = (x_1, x_2, ..., x_p, x_{p+1})$. We use the Adam optimization methods to train our NSARNN. Adam is an optimization algorithm that can be used instead of the classical stochastic gradient descent procedure to update network weights iterative based in training data.

3 Numerical simulations

In this section, we investigate the performance of the NSARNN model through Monte Carlo simulations. We brought the data with spatial effects into the NSARNN model and the ANN model to compare their prediction accuracy in the presence of spatial effects in the data. And in the experiments, the parameter settings of ANN are kept consistent with those of NSARNN.

In the simulations, to illustrate the robustness of our method, we consider four different types of random errors: $N(0, 0.25), t(3), \chi^2(3)$. In each example, we generate 400 observations that are

randomly divided into 399 to fit the model, leaving one observation for prediction. The simulation results are based on 25 replications to examine the performance of the competing models.

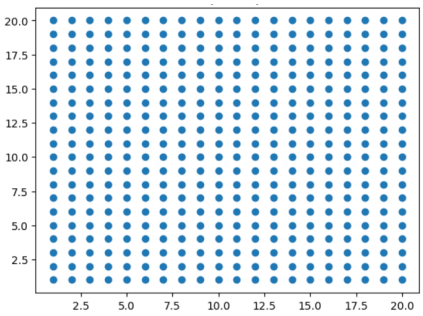

Figure 2:

We constructed a spatial weight matrix W where the $(i,j)^{th}$ element w_{ij} is 1 if the points or vertices i and j are adjacent in the plot of Fig. 2, and 0 if there is no edge between the vertices i and j. The diagonal elements of the matrix are all zero.

3.1 Data generation

Here we consider four different simulation examples.

Example 1: First, we consider the ordinary spatial linear autoregression model. Here we consider four predictors. The data are generated from

$$y = \rho W y + 4x_1 + 3.5x_2 + 3x_3 + 2.5x_4 + \epsilon \tag{9}$$

where x_1, x_2, x_3 and x_4 are independent and follow N(0,1).

Simulated data with spatial effects were generated, and the same data were put into the spatial autoregressive neural network model of this paper and the ordinary artificial neural network (ANN) model to compare the prediction results. Compare the prediction accuracy of the two methods when data with spatial effects but different models are applied.

Example 2: Nonlinear spatial autoregressive model.

For this example we consider two predictors. The data are generated from

$$y = \rho W y + sin(\pi x_1) + 4x_2(1 - x_2^2) - 1 + \epsilon \tag{10}$$

where $x_1 \sim U(-1,1), x_2 \sim U(0,1)$.

Example 3: Partially linear additive spatial autoregressive model

$$y = \rho W y + X\beta + \sum_{k=1}^{d} g_k(U_k) + \epsilon \tag{11}$$

where Y is an n-dimensional response variable and W is a known $n \times n$ matrix of spatial weights

with zero diagonal elements. $X = (X_1, X_2, ..., X_p)$ is an $n \times p$ linear regression matrix, and $U = (U_1, U_2, ..., U_d)$ is the nonparametric regression element of $n \times d, U_1, U_2, ..., U_d$ of the regression matrix with $k = 1, 2, ..., d$; $g_k()$ is a one-element unknown smooth function. To ensure the identifiability of the function component $g_k()$, assume that $E_{g_k}(U_k) = 0, k = 1, 2, ..., d$.

The simulation data are generated from

$$y = \rho W y + 2x_1 + 4x_2 + cos(z_1) + z_2(1 - z_2^2) + \epsilon \tag{12}$$

where x_1, x_2 are mutually independent standard normal random variables, $z_1 \sim U(-1, 1), z_2 \sim U(0, 1)$.

This model has a linear relationship between the dependent variables and some explanatory variables, and a non-linear relationship with some other explanatory variables, so it retains all the advantages of a partially linear additive regression model.

Simulated data with spatial effects were generated, and the same data were put into NSARNN and ANN, then the prediction accuracy was compared.

3.2 Performance results

To compare the performance of the estimates of each model f (x) in-sample and out-of-sample, we use the mean square error (MSE) as measurements of forecasting performance:

$$\frac{1}{n} \sum_{i=1}^{n} \left(y_i - \hat{f}(x_i) \right)^2 \tag{13}$$

where $\hat{f}(x_i)$ is an estimator of $f(x_i)$.

To ensure the validity of the experiments, we uniformly use 1 hidden layer containing 100 neurons in each experiment. And we construct a neural network structure with only one node in the output layer as the predicted output of the dependent variable.

Table 1: Simulation results of Example 1 under different random errors and spatial correlation

ρ	Error	Train		Test	
		ANN	NSARNN	ANN	NSARNN
0.2	N(0, 0.25)	25.301	0.259	30.655	0.362
	t(3)	24.300	2.303	35.738	1.029
	$\chi^2(3)$	43.422	5.203	45.002	9.317
0.4	N(0, 0.25)	32869.860	3.802	32503.525	2.467
	t(3)	13007.525	5.033	14095.363	3.202
	$\chi^2(3)$	4556.652	6.475	6437.905	6.266
0.6	N(0, 0.25)	1558.910	0.274	1294.029	0.271
	t(3)	6484.632	2.401	8719.259	2.168
	$\chi^2(3)$	3798.743	5.533	3104.367	7.068

Table 2: Simulation results of Example 2 under different random errors and spatial correlation

ρ	Error	Train		Test	
		ANN	NSARNN	ANN	NSARNN
0.2	N(0, 0.25)	1.320	0.464	1.206	0.481
	t(3)	5.953	3.273	4.293	2.553
	$\chi^2(3)$	14.724	5.597	21.651	9.249
0.4	N(0, 0.25)	125.505	0.702	84.600	1.049
	t(3)	466.990	4.722	777.318	4.596
	$\chi^2(3)$	5142.336	8.872	5325.126	4.416
0.6	N(0, 0.25)	47.196	0.671	40.348	0.642
	t(3)	1607.716	22.262	1541.336	4.685
	$\chi^2(3)$	909.314	5.949	717.470	5.260

Table 3: Simulation results of Example 3 under different random errors and spatial correlation

ρ	Error	Train		Test	
		ANN	NSARNN	ANN	NSARNN
0.2	N(0, 0.25)	11.021	0.259	10.932	0.255
	t(3)	17.144	1.904	15.821	1.417
	$\chi^2(3)$	30.664	5.834	43.386	4.946
0.4	N(0, 0.25)	15262.393	1.138	16684.402	1.397
	t(3)	7427.272	3.744	7015.032	4.233
	$\chi^2(3)$	4205.157	7.078	3740.542	5.223
0.6	N(0, 0.25)	779.278	0.324	512.984	0.466
	t(3)	3454.204	4.412	3451.426	1.529
	$\chi^2(3)$	4099.591	7.303	3643.548	9.546

From above tables we can see that train and test MSE for ANN model is much larger than the train and test MSE for NSARNN model, because here the data generation formulas include a spatial dependence along with an AR term, which cannot be captured by simple ANN model; but can be captured by NSARNN model. Also, for any distribution of the error term along with any ρ, NSARNN shows its robustness, whereas ANN becomes very unpredictable (if the data would be a real data, it is expected that we would have seen an increasing pattern in the ANN MSE's, with respect to ρ).

4 Conclusion

So, in this project we combined statistical methods with machine learning methods to construct the NSARNN model by adding a neural network structure to the nonparametric spatial autoregressive model.

In our comparative analysis between the Nonparametric Spatial Autoregressive Neural Network (NSARNN) and the standard Artificial Neural Network (ANN), a clear distinction in prediction accuracy emerged, particularly when there is spatial effect in the data.

5 Worklog

- Paper finding: Everyone
- Data Simulation: Shubham Kr. Rajak
- Modelling: Souvik Roy
- Report writing: Abhraraj Haldar, Souvik Roy