Available online at https://sanad.iau.ir/journal/jrors
Publisher: Ya.C., Islamic Azad University
Journal of Radar and Optical Remote Sensing and GIS
Print ISSN:2645-484X Online ISSN:2645-5161

Application of Machine Learning in Spatial Data Analysis

Mahsa Torki

1. Master of Computer Engineering. E-mail: m.mahsa.torki@gmail.com

Article Info

Article type:
Research Article
(Article, Review, Short Communication, etc.)

Article history:
Received June 12, 2025
Revised form Oct. 22, 2025
Accepted Dec. 25, 2025
Published online Dec. 28, 2025

Keywords:
Machine Learning,
Spatial Data,
Remote Sensing,

ABSTRACT

Objective: Spatial data explicit features are often ignored or inadequately utilized in machine learning for spatial application domains. Meanwhile, resources that can identify these features and investigate their impact and management methods in machine learning applications have lagged behind.

Methods: In this literature review, we seek to identify and discuss the spatial data features that affect machine learning performance. We examine some best practices in managing such features in spatial domains and discuss their advantages and disadvantages. We study two broad sections in this research.

Results: In the first, spatial data features are developed in the spatial observation matrix without modification to the learning algorithm's core.

Conclusion: On the other hand, spatial data features are managed within the learning algorithm itself. While the latter has been studied much less, we argue that they offer the most promising prospect for the future of spatial machine learning.

Cite this article: Author name, Initial., Author name, Initial., & Author name, Initial. (2025). Title of paper. ***Journal of Radar and Optical Remote Sensing and GIS***, *8*(4), pages.

Publisher: Yazd Campus (Ya.C.), Islamic Azad University

Introduction

The rapid development of technology, especially in remote sensing, mobile devices, and the Internet of Things (IoT), has resulted in an explosive growth of spatial data production. Spatial data encompasses information where the geographical location is a crucial component, distinguishing it fundamentally from general non-spatial datasets. Examples include satellite imagery, GPS trajectories, demographic maps, and environmental sensor readings. The analysis and interpretation of this data are paramount for decision-making across diverse fields such as urban planning, environmental modeling, public health, and resource management.

Machine Learning (ML) algorithms have demonstrated remarkable success in handling complex, high-dimensional datasets across various scientific and engineering domains. However, applying standard ML techniques directly to spatial data often yields suboptimal results because they inherently fail to account for the unique characteristics embedded within spatial information—namely, spatial dependency (Tobler's First Law of Geography) and spatial heterogeneity (non-stationarity).

Spatial Dependency implies that observations closer to each other in space are more likely to be similar than those farther apart. This spatial autocorrelation violates the core assumption of independence among observations often required by classical statistical and ML models.

Spatial Heterogeneity refers to the notion that relationships between variables might vary across different locations in the study area. The underlying processes generating the data are non-stationary.

Ignoring these spatial explicit features leads to models that are statistically biased, less accurate, and often fail to capture the true underlying spatial processes. Researchers have long recognized the necessity of incorporating geographical knowledge into analytical frameworks. Traditional spatial statistics, such as Geographically Weighted Regression (GWR) or spatial econometrics models (e.g., Spatial Autoregressive Models, SAR), explicitly handle spatial structures, usually through a pre-defined spatial weight matrix (W).

The recent confluence of powerful ML techniques (e.g., Deep Learning) and geospatial analysis presents a new frontier: Spatial Machine Learning (SML).

SML aims to leverage the predictive power of ML while respecting the spatial constraints of the data. This review systematically surveys the emerging landscape of SML. We categorize the current approaches based on where the spatial information is incorporated into the ML pipeline:

1. Modifying the Spatial Observation Matrix ($\mathbf{X}$): Strategies where spatial structures (like neighbors or local contexts) are explicitly engineered into the input feature space before feeding the data into a standard,

unmodified ML algorithm.

2. Modifying the Learning Algorithm ($\mathcal{A}$): Strategies where the core mechanics of the learning algorithm itself are adapted or extended to intrinsically handle spatial dependency and heterogeneity during the training and prediction phases.

We posit that while the first approach has been more extensively explored, the second approach, involving algorithmic modifications, holds the most significant promise for developing generalizable and robust SML frameworks for the future.

The structure of this paper is as follows: Section 2 provides a brief background on essential ML concepts. Section 3 reviews the fundamentals of spatial data types and common GIS operations relevant to preprocessing. Section 4 details methods for integrating spatial information into the observation matrix. Section 5 focuses on integrating spatial awareness directly into learning algorithms.

Section 6 summarizes key findings and trends, and Section 7 concludes the review.

Machine Learning Background

Machine learning, at its core, involves creating algorithms that can learn patterns from data without being explicitly programmed for the task. These algorithms typically learn a function f that maps input data $\mathbf{X}$ to output $\mathbf{y}$, i.e., $\mathbf{y} = f(\mathbf{X})$. ML paradigms are generally categorized into three main types: Supervised Learning, Unsupervised Learning, and Reinforcement Learning.

Supervised Learning

Supervised learning algorithms learn a mapping function based on a labeled training dataset consisting of input-output pairs $(\mathbf{X}_i, y_i)$. The goal is to generalize this mapping to unseen data.

Classification: Used when the output variable y is categorical (e.g., land cover type, presence/absence of a disease). Common algorithms include: * Support Vector Machines (SVM): Finds the optimal hyperplane that maximally separates classes in the feature space. * Decision Trees (DT) and Ensemble Methods: Algorithms like Random Forest (RF) and Gradient Boosting Machines (GBM) build an ensemble of trees to improve predictive performance and robustness. * Neural Networks (NN): Multi-layer perceptrons capable of learning highly complex, non-linear relationships.

Regression: Used when the output variable y is continuous (e.g., temperature, housing price). Algorithms include Linear Regression, Support Vector Regression (SVR), and deep neural networks configured for continuous output.

Unsupervised Learning

Unsupervised learning deals with unlabeled data. The objective is to discover inherent structures, patterns, or groupings within the input data $\mathbf{X}$.

Clustering: The process of grouping data points such that points in the same group (cluster) are more similar to each other than to those in other groups. Examples include K-Means and DBSCAN.

Dimensionality Reduction: Techniques used to reduce the number of input features while retaining most of the variance or critical information, often used for visualization or mitigating the curse of dimensionality. Principal Component Analysis (PCA) is a standard technique here.

Reinforcement Learning (RL)

RL involves an agent interacting with an environment to maximize a cumulative reward. The agent learns an optimal policy through trial and error, making sequential decisions rather than relying on pre-existing input-output pairs. While

less common in traditional static spatial analysis, RL is increasingly relevant in dynamic spatial modeling, such as autonomous navigation or adaptive environmental management.

Variations on Learning Paradigms

• Semi-Supervised Learning (SSL): Combines a small amount of labeled data with a large amount of unlabeled data during training. This is highly

relevant in spatial domains where labeling spatial data (e.g., ground truth validation) is expensive or time-consuming, while unlabeled spatial data is abundant.

• Active Learning (AL): A technique where the algorithm intelligently selects the most informative unlabeled data points for a human oracle to label, optimizing the labeling budget.

Spatial Data and Analysis

Spatial analysis fundamentally relies on understanding the structure and representation of geographical information.

Geographic Information Systems (GIS) Fundamentals

GIS provides the foundational framework for managing, analyzing, and visualizing spatial data. Spatial data is generally represented in two main structures: Vector and Raster.

Vector Data: Represents geographic features using explicit geometric shapes: points (e.g., trees, fire hydrants), lines (e.g., roads, rivers), and polygons (e.g., countries, lakes). Vector data explicitly stores topological relationships.

Raster Data: Represents the geographic space as a regular grid of cells (pixels), where each cell stores a thematic value (e.g., elevation, spectral reflectance in satellite imagery). The spatial relationship is implicit, defined by the cell's row and column index and the defined spatial resolution.

Spatial Operations and Feature Engineering

Before applying ML, spatial data often requires manipulation using standard GIS operations. These operations inherently deal with spatial relationships and are critical for feature engineering in SML:

• Geometric Operations:

◦ Clip: Extracting a subset of features based on the extent of an overlay feature.

◦ Erase: Removing portions of one layer that overlap with another.

◦ Buffer: Creating a zone of specified distance around a feature (point, line, or polygon). This is a primitive way to define local spatial

neighborhoods.

◦ Union: Combining features from two layers to create a new layer representing the overlay of all features.

• Spatial Interpolation: Estimating values at unmeasured locations based on known sample points. Techniques like Inverse Distance Weighting (IDW) or Kriging are fundamentally spatial models that generate dense raster surfaces from sparse point data.

The output of these operations often forms the basis of the engineered features in the spatial observation matrix, explicitly capturing neighborhood effects or proximity relationships needed for SML models.

Spatial Features in the Observation Matrix

This section addresses the first major strategy: incorporating spatial knowledge by transforming the input data matrix $\mathbf{X}$ or generating auxiliary matrices that reflect spatial structure, without altering the fundamental mathematics of the subsequent learning algorithm

$\mathcal{A}$. This often involves pre-calculating neighborhood statistics or defining spatial autocorrelation measures.

Defining Spatial Context via Weight Matrices

The cornerstone of incorporating spatial dependency into the observation matrix is the Spatial Weight Matrix ($\mathbf{W}$). $\mathbf{W}$ is an $N \times N$ matrix (where N is the number of spatial observations), describing the connectivity or influence between any two locations i and j.

The element wij quantifies the spatial relationship: * If $w_{ij} = 0$, there is no direct spatial relationship assumed between i and j. * If $w_{ij} > 0$, there is a defined relationship, which can be binary (e.g., adjacency), inverse distance, or based on other metrics.

Common constructions for $\mathbf{W}$: 1. Contiguity Weights: Based on shared boundaries (Rook, Queen cases). 2. Distance-Based Weights: For example, inverse distance weighting: w ij= 1/dij, where d_{ij} is the distance between i and j, and k is a power parameter.

Once $\mathbf{W}$ is defined, spatial features can be engineered by calculating spatially lagged variables: y L = Wy where yL represents the spatially lagged response, effectively summarizing the neighborhood values of the target variable. Similarly, lagged predictors X L =WX can be calculated. These lagged variables are then added as new features to the original observation matrix $\mathbf{X}$, which is then fed into a standard ML classifier or regressor.

4.2. Feature Engineering for Spatial Heterogeneity

Spatial heterogeneity implies that the relationship between predictors and the response varies spatially. While $\mathbf{W}$ captures dependency, methods must be employed to capture varying local relationships within the feature set.

4.2.1. Clustering for Local Models

One approach is to use unsupervised learning (clustering) to partition the study area into distinct spatial regimes where stationarity might hold locally. 1. Cluster the spatial locations based on feature values ($\mathbf{X}$) or potentially spatial coordinates. 2. Train a separate, specific ML model for each cluster. The resulting prediction for a new point i is determined by the prediction of the model corresponding to the cluster i belongs to.

4.2.2. Dimensionality Reduction Techniques

Dimensionality reduction techniques, while generally agnostic to spatial location, become SML tools when adapted to handle spatial autocorrelation in the feature space.

Principal Component Analysis (PCA): Standard PCA transforms the data into orthogonal principal components that maximize variance. If spatial autocorrelation exists strongly across the original features, PCA might mix spatially dependent and independent components without explicit spatial awareness.

Locally Weighted PCA (LWPCA) and Geographically Weighted PCA (GWPCA): These methods adapt PCA to account for spatial non-stationarity. * LWPCA/ GWPCA: Instead of calculating a single covariance matrix for the entire dataset, LWPCA calculates a localized covariance matrix for each observation i using a kernel function (similar to a spatial weight matrix) centered at i. The resulting Principal Components calculated at location i are therefore more

representative of the local feature structure.

C =N

∑j=1K (j)(x −x¯i)(x −x¯i) T

where $K_i(j)$ is the spatial weight (kernel) applied to observation j when calculating the components for location i.

By using these spatially weighted components as input features for a subsequent ML model, the model implicitly benefits from a feature representation that respects local data structure, albeit through pre-processing.

Spatial Features in the Learning Algorithm

The second, more advanced strategy involves embedding spatial concepts directly into the architecture or objective function of the learning algorithm itself. This avoids the limitations of manually defined spatial weight matrices ($ \mathbf{W}$) and allows the algorithm to learn the optimal spatial relationships or local model parameters during training.

Spatial Regularization and Objective Functions

For standard regression or classification tasks, the objective function (loss function) $L(\mathbf{y}, f(\mathbf{X}))$ can be augmented with a spatial penalty term Ω_S:

L SML=L(y, f(X)) + λΩ (W, θ)

where λ is a regularization parameter, and Ω_S quantifies the spatial inconsistency of the model parameters (θ) or predictions.

Spatial Autoregressive Models (SAR) Analogy: In traditional econometrics, SAR models introduce spatial dependence into the error term:

$y = X\beta + \rho W y + \epsilon$. In SML, this concept can be adapted. For instance, a neural network output $\hat{\mathbf{y}}$ might be constrained such that the

residual error $e = y - \hat{y}$ exhibits low spatial autocorrelation, or the prediction itself is encouraged to be spatially smooth according to $\mathbf{W}$.

Geographically Weighted Machine Learning

Methods like Geographically Weighted Regression (GWR) demonstrate the power of local parameterization. Extending this concept to non-linear ML models (like Neural Networks) leads to Geographically Weighted Neural Networks (GWNNs). In a GWNN, instead of having a single set of weights ($\mathbf{W}_{NN}$) for the entire network, the network parameters (weights and biases) are treated as functions of location: W_{NN}

(i) . The network trained at location i is heavily influenced by data points j close to i, defined by a spatial kernel $K_i(j)$: Prediction $= f(X ; W_{NN}(i))$

where $\mathbf{W}_{NN}(i)$ is optimized locally using only the weighted training data in the neighborhood of i. This explicitly addresses spatial heterogeneity by allowing local functional forms.

Learning the Spatial Weight Matrix ($\mathbf{W}$)

Perhaps the most significant gap in the feature engineering approach (Section 4) is the reliance on a user-defined $\mathbf{W}$. If $\mathbf{W}$ is poorly specified, the resulting SML model suffers. A promising SML avenue is to learn $\mathbf{W}$ concurrently with the primary prediction task. If we assume the model structure is a linear combination of observed features and their spatially lagged versions:

$y = x\beta + i(Wx)\gamma + i\epsilon i$

The goal is to find both the parameters (β, γ) and the weight matrix $\mathbf{W}$ (or the parameters defining $\mathbf{W}$ if $\mathbf{W}$ is defined by parameters, e.g., distance decay exponents). This often requires embedding the estimation of $\mathbf{W}$ within a differentiable optimization framework, which points strongly toward Deep Learning architectures.

Spatial Hyperparameter Optimization

In many ML algorithms (e.g., optimizing the depth of a decision tree, or the regularization strength C in SVM), hyperparameters are globally optimized via cross-validation. In SML, spatial sensitivity suggests that the optimal

hyperparameters might themselves vary spatially (spatial hyperparameter heterogeneity).

Standard global optimization might average out local optima, leading to sub-optimal performance in specific sub-regions. Spatial hyperparameter

optimization involves tailoring the validation process, perhaps by performing localized cross-validation or using methods that map hyperparameter space to geographic space.

Results

The survey reveals a clear trend towards integrating spatial awareness more deeply within the learning mechanism rather than solely relying on feature pre-processing.

Convolutional Neural Networks (CNNs) for Spatial Structures

CNNs excel at automatically extracting hierarchical features from grid-like data (e.g., images or gridded spatial datasets). Their success stems from the use of shared local filters (kernels) that translate across the input, embodying a form of parameter sharing reminiscent of spatial contexts.

When applied to spatial modeling, standard 2D CNNs operate naturally on raster data where the spatial weight matrix is implicitly defined by the convolutional kernel size and connectivity (e.g., 3×3 neighborhood).

Automatic Estimation of Spatial Weight Matrices: A key finding is the potential for CNNs to automatically learn the spatial weight structure, bypassing the need for manual definition of $\mathbf{W}$. If the underlying spatial process is locally dependent, a CNN's convolutional filters, when trained on spatial data, effectively learn the optimal local weights that govern feature interactions within that local window. This is particularly powerful when the spatial structure itself is unknown or complex, such as in environmental modeling where connectivity changes based on factors like topography or flow direction.

Graph-Based Deep Learning

While CNNs are highly effective for structured, grid-like spatial data, many critical spatial datasets—such as social networks overlaid on geography, infrastructure networks, or census block data—possess irregular, graph-like topologies rather than perfect grids.

Graph Convolutional Networks (GCNs) offer a robust framework for applying deep learning principles to these irregular structures. A GCN layer aggregates information from a node's neighbors based on the adjacency matrix (which functions as the $\mathbf{W}$ matrix, but is inherent to the graph structure).

h (l+1)= 1W(l)h (l)

iσ j∈N(i)∪{i}∑cijj

where $\mathcal{N}(i)$ are the neighbors of node i, and c_{ij} is a normalization factor, often derived from the graph structure.

GCNs inherently manage spatial dependency because the aggregation process is directly defined by the graph connectivity, making them a prime example of features being managed within the learning algorithm itself. They are proving vital for tasks involving complex relational data in geographic space.

Spatio-Temporal Domains

Many real-world spatial phenomena evolve over time (e.g., pollution spread, traffic flow). Integrating space and time simultaneously requires models capable of capturing spatiotemporal autocorrelation. Deep learning architectures combining Long Short-Term Memory (LSTM) networks (excellent for temporal sequence modeling) with CNNs (excellent for spatial feature extraction) have shown significant promise. A ConvLSTM architecture treats the spatial feature map (from the CNN) as the input sequence for the LSTM unit. This allows the network to simultaneously learn how spatial patterns evolve over time, capturing spatial dependence, temporal dependence, scale variations, and hierarchical relationships within a single integrated framework.

When such sophisticated models are further enhanced with Reinforcement Learning components, where the model can receive feedback based on the consequences of its decisions in a dynamic simulation (common in urban planning or disaster response), the potential exists for developing truly adaptive, generalizable, universal ML methods tailored for complex spatial systems.

Conclusion

The review of the current research shows that the progress in the ML algorithm component compared to the progress made by increasing the spatial observation matrix is still in its early stages, and there is much more room for the development and application of some of these methods in various spatial fields.

Here, the main approach of this research is as follows: CNNs can be used to automatically estimate the spatial weight matrix, which is usually unknown and must be defined by the user to reflect the characteristics of spatial data in many spatial problems.

Progress can be foreseen in several areas. Deep neural networks with convolutional layers have been shown to automatically extract patterns from multiple scales and hierarchies. However, they have so far been mainly used for pattern recognition in gridded datasets. Therefore, use cases in a wider range of application domains are needed. Graph-based deep learning methods offer a new opportunity to apply CNN-based deep learning to graph structure problems (e.g., social networks) or when geographic units are irregular (e.g., census data). More studies should also be done on learning in spatio-temporal domains. Deep neural networks based on a combination of LSTM and CNN introduce simultaneous learning in space, time, scales, and hierarchies. When reinforced with reinforcement learning to add feedback into systems, which exists in many spatial, social, and environmental applications, they can realize the dream of a single universal ML method.

Figure 1. This is a figure.

References

Tobler, W. R. (1970). A computer model of aspects of human φonology. Canadian Geographer/Geéographe canadien, 14(1), 1-14.

Cressie, N. (1993). Statistics for Spatial Data. Revised Edition. Wiley-Interscience.

Anselin, L.(1988). Spatial econometrics: methods and models. Kluwer Academic Publishers.

Fotheringham, A. S., Brunsdon, C., & Charlton, M. (2002).Geographically Weighted Regression: The Analysis of Spatially Varying Relationships. Wiley.

Gholizadeh, M. H., & Gandomi, A. (2018). A review on spatial machine learning approaches. International Journal of Applied Earth Observation and Geoinformation, 65, 1-14.

Kipf, T. N., & Welling, M. (2017). Semi-supervised classification with graph convolutional networks. ICLR.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural computation, 9(8), 1735-1780.

Shi et al., (2015),ConvLSTM Papers.