Spatial-Net: A Self-Adaptive and Model-Agnostic Deep Learning Framework for Spatially Heterogeneous Datasets

Yiqun Xie*
University of Maryland
xie@umd.edu

Xiaowei Jia*
University of Pittsburgh
xiaowei@pitt.edu

Han Bao, Xun Zhou
University of Iowa
{han-bao,xun-zhou}@uiowa.edu

Jia Yu
Washington State University
jia.yu1@wsu.edu

Rahul Ghosh
University of Minnesota
ghosh128@umn.edu

Praveen Ravirathinam
University of Minnesota
pravirat@umn.edu

ABSTRACT

Knowledge discovery from spatial data is essential for many important societal applications including crop monitoring, solar energy estimation, traffic prediction and public health. This paper aims to tackle a key challenge posed by spatial data – the intrinsic spatial heterogeneity commonly embedded in their generation processes – in the context of deep learning. In related work, the early rise of convolutional neural networks showed the promising value of explicit spatial-awareness in deep architectures (i.e., preservation of spatial structure among input cells and the use of local connection). However, the issue of spatial heterogeneity has not been sufficiently explored. While recent developments have tried to incorporate awareness of spatial variability (e.g., SVANN), these methods either rely on manually-defined space partitioning or only support very limited partitions (e.g., two) due to reduction of training data. To address these limitations, we propose a Spatial-Net to simultaneously learn a space-partitioning scheme and a deep network architecture with a Significance-based Grow-and-Collapse (SIG-GAC) framework. SIG-GAC allows collaborative training between partitions and uses an exponential reduction tree to control the network size. Experiments using real-world datasets show that Spatial-Net can automatically learn the pattern underlying heterogeneous spatial process and greatly improve model performance.

CCS CONCEPTS

• **Computing methodologies** → **Neural networks**.

KEYWORDS

Spatial-Net, spatial heterogeneity, model-agnostic, deep learning

ACM Reference Format:
Yiqun Xie, Xiaowei Jia, Han Bao, Xun Zhou, Jia Yu, Rahul Ghosh, and Praveen Ravirathinam. 2021. Spatial-Net: A Self-Adaptive and Model-Agnostic Deep Learning Framework for Spatially Heterogeneous Datasets. In *29th International Conference on Advances in Geographic Information Systems (SIGSPATIAL '21), November 2–5, 2021, Beijing, China.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3474717.3483970

*Both authors contributed equally to the paper.

SIGSPATIAL '21, November 2–5, 2021, Beijing, China

ACM ISBN 978-1-4503-8664-7/21/11…$15.00
https://doi.org/10.1145/3474717.3483970

1 INTRODUCTION

Spatial data are geo-located datasets where the spatial coordinates of each sample is explicitly recorded (e.g., census tracks, COVID-19 cases, trajectories) or can be implicitly inferred (e.g., pixels in satellite or UAV imagery). Spatial datasets are of high value and commonly used across various important societal domains [1]. In agriculture, for example, crop monitoring heavily relies on remote sensing imagery (e.g., GEOGLAM [8]) to generate timely statistics of crop types and growing conditions to ensure global food security. Such imagery is also widely utilized in forest fire surveillance, flood mapping, solar energy estimation, and many others to support policy making. In addition, many other types of spatial data, including points (e.g., crime or disease cases), road networks and trajectories and geo-tagged tweets, are also critical assets to various domain applications in public health, public safety, transportation, etc.

While spatial data are both important and widely used, they pose two key challenges to traditional learning approaches due to two intrinsic properties violating the typical independent and identical distribution (i.i.d.) assumption [1, 22]. First, spatial data are autocorrelated (e.g., temperature and soil compositions) and the spatial dependency among samples violates the independence assumption. Second, spatial heterogeneity are naturally embedded in spatial datasets, which means the underlying process generating datasets varies across different regions. In addition, such variability may not be reflected by variations in observed features. As an example, in agriculture, farmers often use different land management practices (e.g., conservation tillage, low phosphorous) based on their own experience, future crop rotation plan, and local social exchanges, which barely exist in current agricultural data and are extremely difficult to collect. Such spatially heterogeneous processes pose a significant challenge for analyses beyond local scales.

In related work (more in Sec. 5), the earlier rise of convolutional neural networks (CNNs) incorporated the awareness of spatial autocorrelation and greatly outperformed traditional fully-connected architectures [14]. Specifically, the spatial structure of cells or pixels in an input is preserved without the cells being broken and concatenated into a vector (e.g., non-fully-connected layers in original CNNs [14], region-layer in YOLO [18], etc.). The use of local connections (i.e., convolutional kernels) also provides a mechanism

to favor spatially-nearby samples than distant ones. While this addresses the first challenge on spatial sample dependency, the other key issue of spatial heterogeneity has not been sufficiently studied. Recently, a spatial-variability-aware neural network (SVANN) was proposed to explicitly consider spatial variability [9, 10]. However, this work mainly stayed at a conceptual level. It only explored a manual partitioning of data at geographically distant locations, and showed that separately trained networks outperform a single one-size-fits-all model. Spatial ensemble approaches were also developed [12] to adaptively partition a dataset. However, these approaches rely on assumptions that only allow two partitions and can only work for two-class classification problems. Outside recent deep learning related literature, geographically weighted regression (GWR) is a traditional non-parametric model that learns a detailed map of weights at different locations [3]. A major limitation of GWR is that it can only perform linear regression and cannot learn complex phenomenons. In addition, these methods (GWR, SVANN, and spatial ensemble methods) require very dense training data across space to train individual models after space-partitioning.

To address these limitations, we propose a Spatial-Net – a general deep learning framework that can be used with various models – to explicitly incorporate awareness of spatial heterogeneity. Spatial-Net is a self-adaptive architecture that simultaneously learns a space-partitioning scheme and a corresponding deep network architecture, which is a spatial extension for a user-selected deep learning model. Specifically, our contributions are:

- We propose a two-phase framework, namely Statistical Significance based Grow-and-Collapse framework (SIG-GAC), which automatically learns the space-partitioning and layer architectures using dependent statistical tests.
- We present an exponential-reduction-based network style for SIG-GAC to control the network size of Spatial-Net.
- We formulate the Spatial-Net learned by SIG-GAC as a hierarchical structure to enable weight-sharing across partitions at different spatial scales and training via multi-task learning.

Through experiments using real-world and synthetic datasets, we show that Spatial-Net can effectively capture the underlying spatial heterogeneity in data generation processes and can greatly improve solution quality using the SIG-GAC framework.

2 PROBLEM DEFINITION

The general problem is formulated as follows:

Inputs:

- Geo-located feature **X** and label **y** in a spatial domain $\mathcal{D}$;
- A deep learning model $\mathcal{F}$ of interest;
- A significance level α;
- Maximum parameter increase ratio λ where $\lambda > 1$;

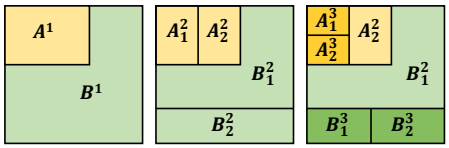
Outputs:

- A space-partitioning scheme D_{part} of $\mathcal{D}$;
- A spatial-heterogeneity-aware $\mathcal{F}$: $\mathcal{F}_{spatial}$ on D_{part};

Objective: Solution quality (e.g., F1-score);

Constraint: The size of the resulting network (e.g., number of layers or parameters) $|\mathcal{F}_{spatial}| \leq \lambda|\mathcal{F}|$.

The goal of this work is to convert an input deep learning model into a spatial-heterogeneity-aware version using input spatial data.



(a) Level 1 (b) Level 2 (c) Level 3

Figure 1: Spatial processes at multiple scales.

The significance level α (threshold for p-value, e.g., 0.05) is used to control the network architecture, and a higher level corresponds to lower sensitivity to spatial heterogeneity. The maximum parameter increase ratio λ (e.g., 2) guarantees that the size of resulting network $\mathcal{F}_{spatial}$ (e.g., measured by the number of layers or parameters) is within a scalar proportion of the original network $\mathcal{F}$.

3 METHOD: A SPATIAL-NET

Spatial-Net aims to simultaneously learns a spatial-partition of a input dataset in a spatial domain and a corresponding network architecture to explicitly model spatial heterogeneity reflected by the partition. The learning of spatial-partition and network-architecture can be considered as a coupled process, where the two highly-dependent sub-tasks work together to converge to a final solution.

Fig. 1 shows an illustrative example to describe the high-level ideas of Spatial-Net. In Fig. 1, spatial heterogeneity are presented at two spatial scales, where A^1 and B^1 are two higher level (level 1) spatial processes; $\{A_1^2, A_2^2, A_1^3, A_2^3\}$ and $\{B_1^2, B_2^2, B_1^3, B_2^3\}$ are more fine-grained local processes (levels 2 and 3) within A^1 and B^1, respectively. For example, large-scale heterogeneity may be caused by state-level policies, climate zones, or major geographical barriers (e.g., mountains, rivers), whereas fine-scale processes may vary by local policies, social and cultural contexts, and personal decisions (e.g., farmers' experience in agriculture).

Spatial-Net provides a framework to automatically learn an approximation of the footprints of spatially heterogeneous processes, and uses the learned spatial knowledge to transform a given deep network architecture (e.g., ANN, CNN, YOLO, UNet) into a spatially-explicit version. To improve the robustness of the solution, statistical tests are used to guide the learning process.

In the following, we introduce the three key components of Spatial-Net: a hierarchical multi-task learning structure for weight-sharing; a significance based grow-and-collapse framework (SIG-GAC) for space-partitioning and network architecture learning; and finally an exponential reduction tree for network size control.

3.1 Hierarchical multi-task learning

Here we will introduce the overall structure of Spatial-Net, which is independent from the choice of deep network for the target problem. To avoid redundancy, here we temporarily assume the patterns of spatial heterogeneity is known (e.g., spatial partitions in Fig. 1), and the partitioning approach will be discussed in Sec. 3.2.

A common approach in handling spatial heterogeneity is to train a model for each local process (e.g., SVANN [9, 10], GWR [3], SE [12]). However, a major issue in these methods is the greatly reduced amount of training data in each partition or at each location, which is especially problematic for data-hungry deep learning techniques.

To mitigate this problem, we use a hierarchical multi-task learning structure to leverage the natural spatial hierarchy embedded in

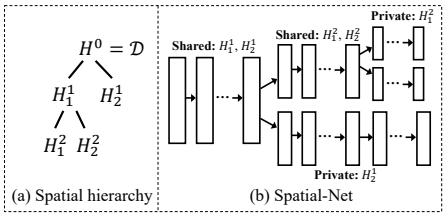


Figure 2: Hierarchical multi-task learning with $\mathcal{H}$.

the heterogeneous processes across multiple scales (e.g., scales of A_1, A_{21}, A_{31}) using the following two definitions:

Definition 3.1. **Spatial process $\Phi : \mathbf{X} \rightarrow \mathbf{y}$:** A function governing data generation in a spatial region, which may include observed and unobserved features as variables.

Definition 3.2. **Spatial hierarchy $\mathcal{H}$:** A multi-scale representation of spatial heterogeneity. $\mathcal{H}$ can be considered as a tree-structured representation of the input spatial domain $\mathcal{D}$; each node $\mathcal{H}_j^i \in \mathcal{H}$ is a partition of $\mathcal{D}$, where i denote the level in the hierarchy, and j is a unique ID for each partition at level i. Children of a partition $\mathcal{H}_j^i$ share the same lower-level processes (i.e., processes at levels $i' \leq i$), and heterogeneity at smaller scales is a mixture of processes at larger scales. Processes $\{\Phi\}$ are homogeneous within each leaf-node and heterogeneous across leaf-nodes.

Based on Def. 3.1 and 3.2, the hierarchical multi-task learning structure considers the learning for the process Φ_j^i at each leaf-node $\mathcal{H}_j^i$ as a task. However, this task modeling itself does not fully utilize the hierarchical spatial relationships across scales represented by $\mathcal{H}$. Denote $\mathcal{H}_1^{i+1}$ and $\mathcal{H}_2^{i+1}$ as two nodes that are children of $\mathcal{H}_1^i$; and $\mathcal{H}_3^{i+1}$ and $\mathcal{H}_4^{i+1}$ as two nodes belonging to $\mathcal{H}_2^i$. Intuitively, we would desire children of the same parent (e.g., $\mathcal{H}_1^{i+1}$ and $\mathcal{H}_2^{i+1}$) to share more common weights than children across parents (e.g., $\mathcal{H}_1^{i+1}$ and $\mathcal{H}_3^{i+1}$). More generally, we would desire leaf-nodes that are closer to each other in the spatial hierarchy $\mathcal{H}$ to share more common weights, where node-distance is defined as:

Definition 3.3. **Spatial hierarchy distance $d_{\mathcal{H}}$.** The minimum number of links needed to connect two nodes in $\mathcal{H}$.

In order to realize this weight-sharing structure in multi-task learning, we enforce Spatial-Net to share the same structure as $\mathcal{H}$. Specifically, a layer is split into branches whenever a node in $\mathcal{H}$ is split into children. Fig. 2 shows an illustrative example where the spatial hierarchy $\mathcal{H}$ in Fig. 2 (a) is integrated into Spatial-Net's hierarchical multi-task learning structure in Fig. 2 (b). This guarantees that two leaf-nodes at levels i and i' (i may be equal to i') share more common weights if their distance $d_{\mathcal{H}}$ in the spatial hierarchy is smaller.

3.2 A SIG-GAC framework

The hierarchical multi-task learning structure effectively models the relationships among tasks over space and reflects the mixtures of processes across scales (Def. 3.2). However, it requires a spatial hierarchy $\mathcal{H}$ of heterogeneity as an input. In this section, we introduce a Significance based Grow-and-Collapse framework (SIG-GAC) to automatically learn the hierarchical structure of spatial heterogeneity from data.

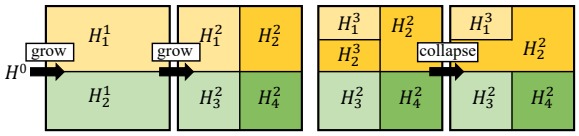


Figure 3: An illustrative example of grow-and-collapse.

SIG-GAC is a coupled process between the learning of hierarchical space-partitioning and the network architecture, which is a natural reflection of the twin-structure of $\mathcal{H}$ and network layers shown in Fig. 2. Specifically, space-partitioning is used to generate scenarios of network architectures and, in turn, local learning profile from the scenarios will be used to decide the final splits in the hierarchy $\mathcal{H}$. This coupled process is supervised by dependent statistical tests for robust and automated decision making. In the following, we will introduce SIG-GAC through its two key phases, i.e., significance-based growth and collapse.

3.2.1 *Significance-based growth.*

The growth phase has a **Grower** (G) that aims to gradually partition the original spatial domain $\mathcal{D}$ into smaller sub-regions (or segments), in an hierarchical manner, with each sub-region in the final partitioning having a homogeneous spatial process Φ (Def. 3.1). Note that this phase only intends to monotonically increase the number of partitions, so potentially some adjacent sub-regions sharing the same Φ may be left as different nodes in the hierarchy, which will be handled by the collapse phase.

The grower G involves two collaborative players – a recommender G_R and a verifier G_V. The recommender G_R propagates via a hierarchical bi-partitioning of the space. In each growth step:

- **Step-1:** The recommender G_R dequeues a current level-i node $\mathcal{H}_j^i$ ($\mathcal{H}^0 = \mathcal{D}$) and returns two equal-size level-$(i+1)$ nodes $\mathcal{H}_{j1}^{i+1}$ and $\mathcal{H}_{j2}^{i+1}$ as a hypothesis of potential spatial heterogeneity. The split direction is vertical for nodes at even levels (e.g., 0,2,...) and horizontal for odd levels (e.g., 1,3,...), as shown in Fig. 3.
- **Step-2:** The verifier G_V verifies if the split is plausible and returns the decision, either "accept" or "reject" to G_R;
- **Step-3:** If the split is accepted, G_R adds the two nodes $\mathcal{H}_{j1}^{i+1}$ and $\mathcal{H}_{j2}^{i+1}$ to the node-queue (FIFO) as candidates for next recommendations; otherwise, G_R drops $\mathcal{H}_{j1}^{i+1}$ and $\mathcal{H}_{j2}^{i+1}$ and moves their parent node $\mathcal{H}_j^i$ to a stable set $\mathcal{H}_{leaf}$ containing all the leaf-nodes of the final hierarchy $\mathcal{H}$;
- **Step-4:** If the queue in G_R is not empty, return to **Step-1**; otherwise, return the stable set of leaf-nodes $\mathcal{H}_{leaf}$ and the synchronized network architecture.

A critical step in the collaborative growing $G = \{G_R \circlearrowleft G_V\}$ is for G_V to verify the plausibility of the hypothesis (i.e., the node split $\mathcal{H}_j^i \rightarrow \{\mathcal{H}_{j1}^{i+1}, \mathcal{H}_{j2}^{i+1}\}$) suggested by G_R. Spatial-Net incorporates dependent statistical tests into the training process of the network to determine if the hypothesis should be accepted or rejected:

Definition 3.4. Denote Φ_1 and Φ_2 as the data generation processes (Def. 3.1) at the two higher-level nodes $\mathcal{H}_{j1}^{i+1}$ and $\mathcal{H}_{j2}^{i+1}$, respectively; and θ_1 and θ_2 as the parameters of an input network $\mathcal{F}$ to capture the two processes, respectively. The null hypothesis H_0 states that $\theta_1 = \theta_2$ whereas the alternative hypothesis states $\theta_1 \neq \theta_2$.

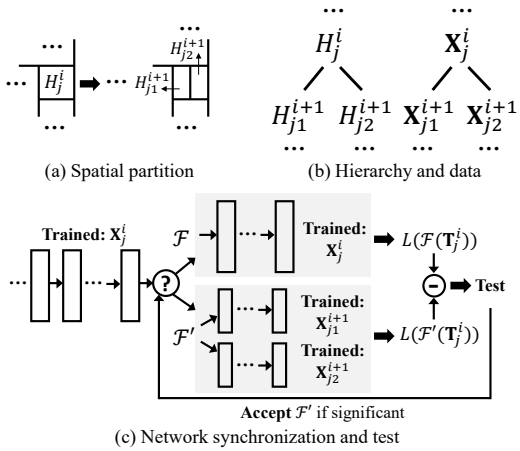


Figure 4: Network synchronization and test.

In order to determine if the null hypothesis H_0 should be rejected, we formulate the problem as an upper-tailed dependent T-test, where the node-split as well as its corresponding new network branches can be considered as a "treatment" (i.e., whether the split is effective in improving the performance).

Denote $\mathbf{X}_j^i$ as the set of training data samples in partition (node) $\mathcal{H}_j^i$ where j is the ID of the partition at level i; and $\mathbf{X}_{j1}^{i+1}, \mathbf{X}_{j2}^{i+1} \subset \mathbf{X}_j^i$ as the data samples in the split nodes $\mathcal{H}_{j1}^{i+1}$, $\mathcal{H}_{j2}^{i+1}$, respectively. Similarly, denote $\mathbf{T}_j^i$ as the set of validation data samples in the nodes. Finally, denote $\mathcal{F}$ and $\mathcal{F}'$ as the network architecture before and after the split, respectively, where $\mathcal{F}'$ synchronizes the split with a newly formed branch (Fig. 4). Note that the new branches in $\mathcal{F}'$ are separately trained with $\mathbf{X}_{j1}^{i+1}, \mathbf{X}_{j2}^{i+1}$ to reflect the space partitioning (i.e., parameters are separated into θ_1 and θ_2 as described in Def. 3.4). This test is then formally defined as:

Definition 3.5. **Spatial heterogeneity test.** Considering the node-split and corresponding network architecture changes as a "treatment" to spatial heterogeneity described in Def. 3.4 (i.e., $\theta_1 \neq \theta_2$), and samples in $\mathbf{T}_j^i$ as the group of participants. Using the test statistic defined by a loss function L, the test is then an upper-tailed dependent T-test, because: (1) participants are essentially from the same group $\mathbf{T}_j^i$ so there are dependency between $L(\mathcal{F}(\mathbf{T}_j^i))$ and $L(\mathcal{F}'(\mathbf{T}_j^i))$ (i.e., same group of participants before and after the "treatment"); and (2) we are only interested in an effective "treatment" that reduces L by separating the processes, i.e., $L(\mathcal{F}'(\mathbf{T}_j^i)) < L(\mathcal{F}(\mathbf{T}_j^i))$, so only the upper-tail (one-side of the distribution) can be used to reject H_0. Since the test is a dependent T-test, the test statistic needs to be first formulated as the difference between the two losses (before and after):

$$L_{diff} = L(\mathcal{F}(\mathbf{T}_j^i)) - L(\mathcal{F}'(\mathbf{T}_j^i)) \tag{1}$$

Further, in order to standardize the distribution into a T-test for finding critical values, the final form of the test statistic is:

$$L_{test} = \frac{\overline{L_{diff}}}{\sigma(L_{diff}) \cdot (DF+1)^{-\frac{1}{2}}} \tag{2}$$

where $\overline{L_{diff}}$ and $\sigma(L_{diff})$ are the mean and standard deviation of L_{diff} over the set of samples in $\mathbf{T}_j^i$; DF is the degree of freedom, which is equal to the sample cardinality $|\mathbf{T}_j^i| - 1$.

Since the normalized test statistic follows the standard T-test, the critical value $CV(DF, \alpha)$ associated with the degree of freedom DF and the input significance level α (e.g., 0.01, 0.05) is then retrieved in the one-sided (upper-tailed) T-distribution table (note that the critical values cannot be computed in closed-form so pre-computed T-distribution tables are typically used for look-ups). Finally, the test result is given by:

$$\text{Significance} = \begin{cases} 1 \text{ (reject } H_0), & \text{if } L_{test} > CV(DF, \alpha) \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

Using Def. 3.5, the verifier G_V accepts the suggested node-split and corresponding network architecture change if the test is significant (i.e., rejecting the null hypothesis H_0). One minor issue left is that it is impossible for the T-distribution table to include all possible values of degrees of freedom DF (i.e., number of samples - 1). Since in the case for Spatial-Net the number of samples in each partition is not fixed and highly variable, it is very likely that the actual DF is not present in the T-distribution table. Thus, we employ mathematical approximations of critical values to estimate $CV(DF, \alpha)$. Specifically, we approximate the $CV(DF, \alpha)$ using the linear interpolation with reciprocals of DF as it is known that critical values are linearly well proportional to the reciprocals:

$$CV_{apx}(DF, \alpha) = \frac{DF^{-1} - DF_{min}^{-1}}{DF_{max}^{-1} - DF_{min}^{-1}} \cdot (CV(DF_{max}, \alpha) - CV(DF_{min}, \alpha))$$

where $DF_{min} = \arg\max_{DF_i \in T-table, DF_i \leq DF} DF_i$ (i.e., lower bound of DF in T-table), and $DF_{max} = \arg\min_{DF_i \in T-table, DF_i \geq DF} DF_i$.

Finally, we define two additional constraints to improve its flexibility and practicality:

- Minimum level for statistical tests lv_{min}: Since in real-world applications the spatial extents represented by true leaf nodes may be small compared to the entire input spatial domain $\mathcal{D}$, the signal of such local heterogeneity may be diluted at larger scales, making it difficult to confirm it statistically. Thus, lv_{min} requires that the recommender G_R directly accepts the node split (without sending to the verifier G_V) if the parent node $\mathcal{H}_j^i$ is at a level $i \leq lv_{min}$. This helps the grower to zoom into smaller scales during the search, which may otherwise be terminated prematurely due to insignificant test results at larger scales. Unnecessary splits will be collapsed back in the next phase (Sec. 3.2.2).
- Maximum level lv_{max}: This is the maximum level $\mathcal{H}$ may have after the growth. lv_{max} is mainly used to constrain the size of the final network, and can also be interpreted as the smallest spatial extent of heterogeneity that domain users are interested in.

3.2.2 *Significance-based collapse.*

The role of this collapse phase finalizes the hierarchy $\mathcal{H}$ and the network architecture of Spatial-Net by collapsing spatially-adjacent partitions (not necessarily adjacent nodes in $\mathcal{H}$, e.g., children of different parent nodes) that share the same spatial process Φ (Def. 3.1). Opposite to the growth phase, this collapse phase only monotonically reduces the number of partitions (nodes) in $\mathcal{H}$.

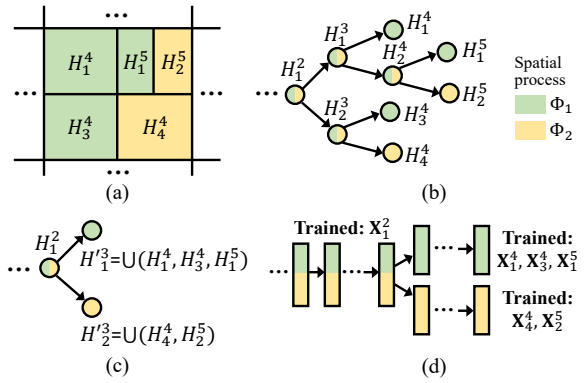


Figure 5: Illustrative examples for collapse.

In the growth phase, there are two scenarios where spatially adjacent partitions sharing the same spatial process Φ may be split into separate nodes: (1) The flexibility constraint imposed by lv_{min} forces node-splits at levels $i \leq lv_{min}$, regardless of whether heterogeneity exists; and (2) Partitions belonging to different parent nodes may be spatially adjacent, and they may share the same process Φ after being separated out from other siblings (with a different process) at finer-scale partitions (e.g, $\mathcal{H}_1^4$ and $\mathcal{H}_1^5$ in Fig. 5 (a)).

The collapse phase identifies these unnecessary splits and merges them back to generate the final Spatial-Net. Fig. 5 shows illustrative examples for collapse operations. In the following we will describe the details of the process via Def. 3.6 and the examples in Fig. 5.

Definition 3.6. **Active collapsing node (AC-node).** The node whose leaf decedents (nodes/partitions in the current $\mathcal{H}$) will be enumerated as candidate for collapsing.

The collapse phase performs node-collapse gradually from finer spatial scales towards larger scales. For an AC-node, there are three requirements for selecting pairs of candidate nodes for collapse:

- Denote H_1^i and H_2^i as two original leaf-nodes of the same parent in the output $\mathcal{H}$ from the grower. The pair of candidate nodes are invalid if one of them contains (or equals) H_1^i and the other contains (or equals) H_2^i (i.e., known to contain heterogeneous leaf regions); unless $i \leq lv_{min}$ which means the split of H_1^i and H_2^i was not a decision by significance testing (Def. 3.5).
- The pair of candidate nodes for collapse are spatially adjacent in order to maintain the spatial contiguity of each node in the spatial hierarchy $\mathcal{H}$.
- The level difference between a candidate node and the current AC-node is at most 2 (node levels will be reduced in the new $\mathcal{H}$ as they collapse). This requirement is mainly due to computational consideration and may be relaxed in future work.

In Fig. 5 (b), if H_1^3 is the AC-node, then the valid pair of candidate nodes to be enumerated is (H_1^4, H_1^5). (H_1^5, H_2^5) is invalid due to violation of the first requirement above, and (H_1^4, H_2^5) is invalid as the candidates are spatially disjoint (the second requirement).

If a pair of nodes is collapsed, both the hierarchy $\mathcal{H}$ and the network architecture will be simplified to remove unnecessary branches as shown in Fig. 5 (c) and (d). Specifically, if a parent only has one child (i.e., identical after the exclusion of the other child node), they will be reduced to only the parent node.

Similar to the growth phase, we use statistical test to determine if a pair of candidates should be collapsed. The difference is that here we perform a reversed version of the spatial heterogeneity test (Def. 3.5). In the growth phase, a node-split is accepted if the test is significant, i.e., the loss after the split $L(\mathcal{F}'(\mathbf{T}_j^i))$ is significantly reduced compared to the loss before $L(\mathcal{F}(\mathbf{T}_j^i))$. The key difference here is that, in the collapse phase, we are no longer interested in identifying heterogeneous processes; rather the objective is to identify splits that are unnecessary, i.e., insignificant.

To implement the change, in the **reversed spatial heterogeneity test**, we still perform the spatial heterogeneity test using the pairs of candidate nodes. However, the criterion for accepting the collapse is reversed. Instead of requiring the differences before and after the collapse to be significant (Eq. (3)), we accept the collapse only if the difference is insignificant. This is intuitive because such a result indicates there is no significant difference in the two data-generation processes Φ of the candidate nodes.

3.2.3 Note on multiple testing. Finally, we would like to note that Spatial-Net performs multiple tests as there are potentially many branching decisions to be verified during both the growing and collapse phases. As a result, for a given significance level α (e.g., 0.01), the actual rate of false positives may be higher than α. However, since the cost of false positives (e.g., an unnecessary split of a node) is not as high as those in typical statistical applications (e.g., effectiveness of a drug to a disease), Spatial-Net currently uses the tests in a relaxed manner and does not intend to enforce strict constraint the overall false positive rate (e.g., 0.01). However, if in certain applications strict constraints are needed, a revised significance level α' can be conveniently calculated and used for individual tests based on the estimated total number of candidate splits E_{split} (e.g., $E_{split} = 2^{lv_{max}} - 2^{lv_{min}}$), with $\alpha' = 1-(1-\alpha)^{\frac{1}{E_{split}}}$ (or the Bonferroni adjustment $\alpha' = \alpha/E_{split}$) [16].

3.3 Exponential reduction tree

Since the growth phase of the spatial hierarchy propagates with exponential growth, i.e., number of leaf nodes can be at most $2^{lv_{max}}$ where lv_{max} is the upper-bound on the level, we use an exponential reduction tree style to constrain the size of the corresponding network architecture in Spatial-Net.

Denote the input deep learning model (Sec. 2) as $\mathcal{F}$, and the corresponding Spatial-Net as $\mathcal{F}'$. In this paper, we will use the number of network layers, instead of the number of parameters, to represent the relative size of networks $|\mathcal{F}|$ and $|\mathcal{F}'|$ because the current version of Spatial-Net uses a network layer (Fig. 4) as the minimum unit for branching (i.e., no split within a layer). Here we will use an exponential reduction tree as the style of Spatial-Net to explicitly control the total number of layers $|\mathcal{F}'|$:

Definition 3.7. An **exponential reduction tree** (ERT) has the form of a binary tree where the roles of nodes and edges are reversed. Specifically, attributes (e.g., edge length) are stored on edges rather than nodes, and nodes mainly serve as conjunction points to topologically connect the edges (i.e., roles of edges in an ordinary binary tree). Denote β ($\beta > 1$) as the base for the exponential reduction (e.g., $\beta = 2$). In an ERT, whenever a parent edge e is split into two children e_1 and e_2, each child will have length

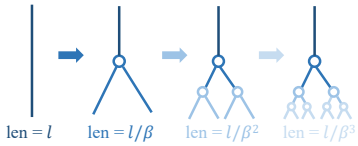


Figure 6: Exponential reduction tree style and tail lengths.

$e_1.len = e_2.len = e.len/\beta$, and the parent edge's new length $e.len'$ will be reduced to $e.len \cdot (1 - 1/\beta)$. The sum of edge lengths of the updated parent and any of the two children (e.g., $e.len' + e_1.len$) is identical to the parent's edge length before the split $e.len$.

The ERT style (Fig. 6) can be directly applied during the growth phase of Spatial-Net, where the expansion of network branches follows a binary tree style and the number of layers within a branch segment (i.e., layers between two consecutive splits) can be represented by ERT's edge length. As an example, using β as the base for exponential reduction, the first split (from $\mathcal{H}^0$ to $\mathcal{H}_1^1, \mathcal{H}_2^1$) will keep the first $\lceil (1 - 1/\beta) \cdot |\mathcal{F}| \rceil$ layers unchanged, and create a new branch starting from the next layer. Following the same rule, the number of layers in each tail branch after the i^{th} split is:

$$\underbrace{\lfloor 1/\beta \cdot \lfloor .....\lfloor 1/\beta \cdot}_{i \text{ times of "} \lfloor 1/\beta \cdot \text{"}} |\mathcal{F}| \underbrace{\rfloor .....\rfloor \rfloor}_{i \times \text{"} \rfloor \text{"}} \leq (1/\beta)^i \cdot |\mathcal{F}| \tag{4}$$

Thus, the total number of layers in Spatial-Net $|\mathcal{F}'|$ after the growth phase is:

$$\begin{aligned} |\mathcal{F}'| &\leq |\mathcal{F}| + \sum_{i=1}^{I} 2^{i-1} \cdot (1/\beta)^i \cdot |\mathcal{F}| \\ &\leq |\mathcal{F}| + \frac{|\mathcal{F}|}{2} \sum_{i=1}^{I} (\frac{2}{\beta})^i \end{aligned} \tag{5}$$

where I is the actual maximum level in the resulting hierarchy. Based on Eq. (5), with $\beta = 2$ the size of Spatial-Net is bounded by $\frac{I+2}{2}|\mathcal{F}|$; with $\beta > 2$, the sequence is convergent as $\frac{2}{\beta} < 1$.

In the current version of Spatial-Net, we set $\beta = 2$. Using Eq. (5), for $I = 6$, we have the maximum number of leaf-nodes (each leaf-node can form a unique path in $\mathcal{F}'$) as $2^6 = 64$ and the actual network size $|\mathcal{F}'| \leq 4|\mathcal{F}|$ (i.e., suitable for an input $\lambda = 4$ in Sec. 2). As we can see, the use of the exponential reduction tree is effective in controlling the number of layers in Spatial-Net. Intuitively, spatial processes at finer resolutions tend to have smaller differences, and thus require less "private" parameters to capture such differences.

In practice, the size of Spatial-Net is often smaller than $\frac{I+2}{2}|\mathcal{F}|$ because many of the candidates for node-split may not pass the spatial heterogeneity test (Def. 3.5), and many of the unnecessary splits will be reduced during the collapse phase.

In the final Spatial-Net $\mathcal{F}'$, each branch is formed by a unique path from the input to the output layer, and each unique path has the same architecture as the input deep learning model $\mathcal{F}$.

3.4 Training and prediction models

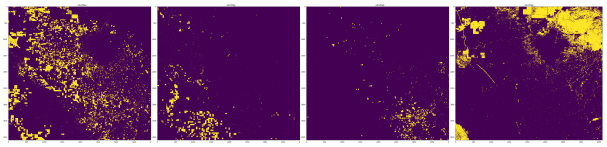
Our proposed Spatial-Net can be easily combined with a variety of predictive models. In our tests, we consider two types of models, standard artificial neural networks (ANN) and segmentation networks. Both of these models aim to learn a mapping relationship from input $\mathbf{X}$ to the target label of each location. We represent such models using $\mathcal{F}(\mathbf{X}; \theta)$, where θ represents the weight parameters in neural networks.

For the segmentation model, one of the most fundamental techniques is the Fully Convoluted Network (FCN) [15] which uses deconvolutional layers to convert the compact representation learned from input data to pixel-wise class labels. Researchers have also developed several variants of FCN such as SegNet [2], DeconvNet [17] and UNet [19]. As Spatial-Net is a model-agnostic framework, in this test we use UNet as an example without loss of generality. UNet is a widely used structure consisting of a multi-layer encoder and a multi-layer decoder. The encoder extracts representative feature representation while reducing the data resolution over layers. The decoder then transforms the representation to the original resolution and predict class labels. Moreover, it supplements the output of the decoder layers with the representation extracted by encoder layers with the same resolution.

For both ANN and UNet, their parameters θ can be estimated through a training process on a labeled dataset by minimizing an objective function of empirical risk, such as the pixel-wise cross entropy for classification, as follows:

$$L_{CE}(\boldsymbol{\theta}|\mathbf{X}, \mathbf{y}) = -\frac{1}{N} \sum_i \sum_k (\mathbf{y}_i)_k \log \mathcal{F}(\mathbf{X}_i; \theta)_k \tag{6}$$

where $\mathcal{F}(\mathbf{X}_i; \theta)_k$ is the predicted score of the i^{th} sample belonging to the class k and $(\mathbf{y}_i)_k = 1$ if the i^{th} sample belongs to the class k.



(a) Almonds (b) Tomatoes (c) Walnut (d) Grasslands

Figure 7: Distributions of land cover examples (yellow).

4 VALIDATION

4.1 Datasets

We use satellite imagery observed by the multi-spectral instrument on Sentinel-2 Constellation as features, which has 13 spectral bands at three different spatial resolutions of 10, 20 and 60 meters [21]. We leave out the atmospheric bands (Band 1, 9 and 10) of 60 metres resolution and re-sample all the bands to 20 metres (by majority voting for 10m bands). For our experiment, we aim to classify each pixel to a class label, where the labels for this data set are taken from the USDA Crop Data Layer (CDL) [4].

Minnesota agriculture dataset D1: We consider an agriculture-intensive region with 4096-by-4096 pixels (~80km by 80km extent in 20m resolution) in southwestern Minnesota, US. While this area is dominated by farm patches such as corn and soybean, different farmers have different preferences in land management practices and thus even the same crop patches can appear different over space. Specifically, we use the composite multi-spectral data taken on August, 2019 to classify each pixel to a class label in {Corn, Soybean, Sugarbeats, Water and Wetlands, Urban}. We consider other classes as background, which will not be used for evaluation.

Table 1: F1-scores for land-cover classification in Minnesota (Dataset D1).

Class	ANN	Cluster-ANN	SVANN	UNet	AE-UNet	Spatial-Net^{ANN}	Spatial-Net^{UNet}
Corn	0.915	0.831	0.890	**0.930****	0.882	0.917	**0.933***
Soybeans	0.900	0.755	0.851	0.928	**0.931****	0.900	**0.938***
Sugarbeets	**0.846****	0.537	0.704	0.000	0.000	**0.854***	0.485
Water and Wetlands	0.892	0.888	0.777	0.914	**0.918****	0.898	**0.928***
Developed Area	0.000	0.000	0.000	0.000	0.000	**0.363****	**0.667***
MEAN	0.889	0.773	0.847	0.857	0.832	**0.897****	**0.903***

Note: * for best results; ** for runner-ups.

Table 2: F1-scores for land-cover classification in California (Dataset D2).

Class	ANN	Cluster-ANN	SVANN	UNet	AE-UNet	Spatial-Net^{ANN}	Spatial-Net^{UNet}
Corn	0.573	0.433	0.403	0.583	**0.626****	0.616	**0.772***
Cotton	0.593	0.591	0.529	**0.731****	0.707	0.699	**0.840***
Sorghum	0.021	0.000	0.000	0.311	**0.371****	**0.386***	0.350
Wheat	0.000	0.000	0.000	**0.263****	0.000	0.233	**0.628***
Alfalfa	0.014	0.000	0.000	**0.598****	0.589	0.383	**0.777***
Grapes	0.605	0.534	0.477	0.703	**0.720****	0.718	**0.846***
Citrus	0.000	0.000	0.000	0.000	**0.364****	**0.381***	0.000
Almonds	0.496	0.487	0.472	0.606	**0.652****	0.632	**0.812***
Walnut	0.000	0.000	0.315	0.000	0.000	**0.381****	**0.733***
Pistachio	0.216	0.056	0.283	0.000	0.649	**0.656****	**0.891***
Tomatoes	0.000	0.249	0.000	**0.599****	0.000	0.530	**0.847***
Garlic	0.000	0.340	0.000	0.000	0.000	**0.349****	**0.797***
Tree crops	0.000	0.000	0.000	0.000	0.210	**0.328****	**0.464***
Grasslands	0.763	0.685	0.721	0.765	0.770	**0.799****	**0.840***
Barren land	0.482	0.441	0.474	0.534	0.540	**0.560****	**0.676***
Water	0.442	0.584	0.000	0.673	**0.683****	0.644	**0.763***
Urban	0.538	0.481	0.596	0.676	**0.699****	0.692	**0.762***
MEAN	0.461	0.421	0.431	0.558	0.599	**0.627****	**0.766***

Note: * for best results; ** for runner-ups.

California land cover dataset D2: This area is in Central Valley, California, and has 4096-by-4096 pixels (~80km by 80km extent in 20m resolution). The multi-spectral data is taken in August, 2018. It contains a wide variety of crops with heterogeneous patterns, where the land cover distribution changes over space (Fig. 7). We consider major classes in the data (~ 95%) as listed in Table 2. Low-sample classes (e.g., <1% of data) and an unidentified miscellaneous class, where all baseline methods struggle, are not included.

4.2 Experiment Setup

In our tests, we implement ANN as a seven-layer network where the first six layers use the sigmoid activation function and the output layer uses the softmax function to generate class probabilities. The dimension of each hidden layer is set to 10. The ANN model is trained with the Adam's optimizer with a learning rate of 0.01. The UNet encoder consists of two encoding blocks followed by two convolutional layers; and each block contains two convolutional layers and a max-pooling layer. The UNet decoder uses two decoding blocks where each block contains an deconvolutional layer, a residual layer and two convolutional layers. The hidden representation extracted by the encoder has a dimension of 128. UNet is trained by the Adam's optimizer using a 10^{-3} learning rate. Using the fully-connected ANN and convolutional UNet as examples, we implement the model-agnostic Spatial-Net for these two types of common architectures to demonstrate the improvements.

4.3 Results

Here we evaluate the performance of our proposed Spatial-Net on the real-world datasets D1 and D2. We compare to the standard ANN and UNet which are trained on the entire dataset. We also include three other baselines: Cluster-ANN, SVANN, and AE-UNet. The Cluster-ANN method first uses K-means++ to cluster the entire dataset into 64 clusters and then trains individual ANN models for different clusters. Similarly, SVANN trains separate models for different data partitions but it requires the partitions to be known as an input, which is unavailable here. Thus, we equally divide the whole region into 4 smaller squared regions (similar to the example used in [10]). Also, SVANN is sensitive to training data reduction (no parameter-sharing). Additionally, we compare with the AE-UNet, which is a semi-supervised learning method. This model is first pre-trained on all the locations in the region (including labeled and unlabeled) by minimizing the reconstruction loss (as an auto-encoder (AE)) and then gets fine-tuned using training data. For all these approaches, we sample 25% data for training and another

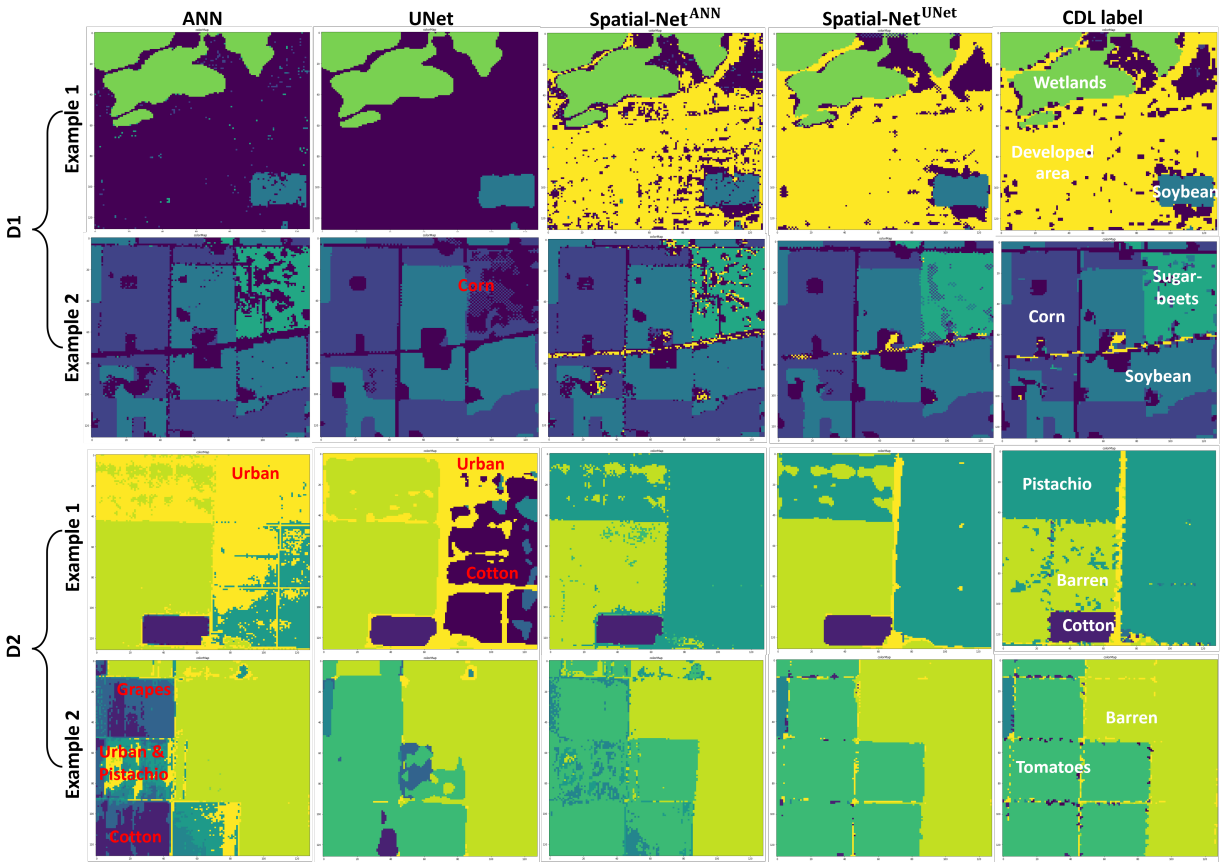


Figure 8: Visualization of the results for candidate methods and CDL in D1 (Minnesota) and D2 (California).

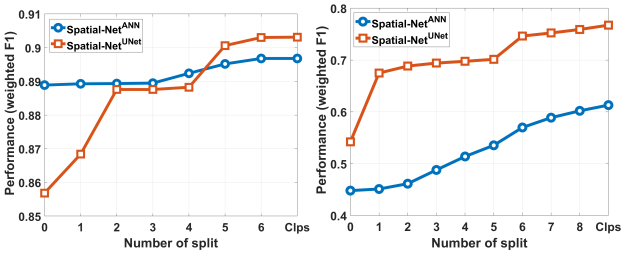
25% data for validation, and use the remaining 50% data for testing (in many real-world applications, ground truth samples are also typically sampled by land surveyors at random locations).

Predictive performance: In Tables 1 and 2, we report the classification performance (in F1-score) of different methods on each land cover as well as the weighted average F1-score (weighted based on proportion of each land cover). We can see that the F1-score of all the methods drop from D1 to D2 due to the increased data heterogeneity. However, in the experiments our proposed methods – Spatial-Net^{ANN} and Spatial-Net^{UNet} – outperform the other methods by a considerable margin for most land cover classes. In particular, Spatial-Net^{ANN} and Spatial-Net^{UNet} performs much better than their counterpart, i.e., ANN and UNet, since these standard classification methods cannot fully capture the spatial heterogeneity. This can be confirmed by the lower F1-score achieved by ANN and UNet for minority classes, e.g., developed area in D1, and wheat, walnut, garlic, and other tree crops in D2. Cluster-ANN performs poorly because the clustering-based partitioning relies solely on observed features and cannot effectively capture the heterogeneity in spatial processes. Similarly, SVANN uses a pre-defined space-partitioning which is not data-adaptive or data-aware. Moreover, both Cluster-ANN and SVANN significantly reduce the available training data for each small region. Finally, AE-UNet does not perform well as the pre-training driven by the reconstruction loss is not sufficient for capturing the spatial heterogeneity.

Case study: Fig. 8 shows several example regions with the CDL labels and the classification results made by ANN, UNet, Spatial-Net^{ANN}, and Spatial-Net^{UNet}. Here we can see that the ANN and Spatial-Net^{ANN} commonly disturbed by noise in remote sensing data at individual pixels and thus their generated maps are less contiguous over space. These are much improved by the UNet versions with the modeling of spatial autocorrelation (e.g., convolutional layers). However, both ANN and UNet suffer from spatial heterogeneity and tend to misclassify smaller classes as dominating classes (sample-size-wise over the entire data). For example, sugarbeets are misclassified as corn by UNet in Example 2 of D1 (Fig. 8). These errors are reduced in the Spatial-Net versions. Additionally, we can see that Spatial-Net^{UNet} generally has a much better performance in detecting roads across farm patches (yellow lines in Fig. 8).

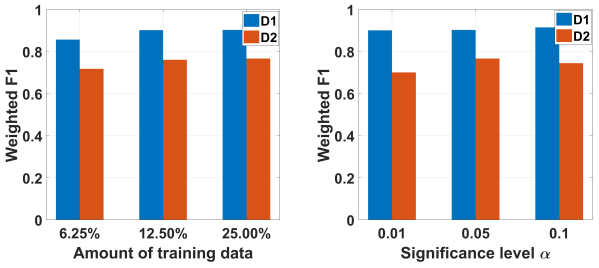
4.4 Model Sensitivity

Performance change during the growth phase: Fig. 9 shows the classification performance as we partition the spatial data during the growth phase. It can be seen that both Spatial-Net^{ANN} and Spatial-Net^{UNet} achieve better performance as we learn the hierarchy, guided by statistical tests. As the search propagates, the growth phase of Spatial-Net gradually generates more significant branches to capture heterogeneous data distributions across the space. We



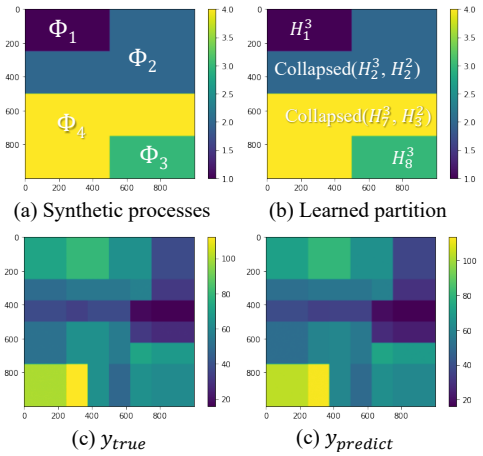
(a) D1 (b) D2

Figure 9: Performance improvement during the growth phase. Each split passing the statistical test doubles the partitions. 'Clps' in x-axis indicates the final result after collapse.



(a) Training data size (b) Significance level α

Figure 10: Performance with respect to data size and α.



(a) Synthetic processes (b) Learned partition

(c) y_{true} (c) $y_{predict}$

Figure 11: Capturing synthetic spatially heterogeneous processes. Partition in (a) is unseen to the Spatial-Net.

can also observe that Spatial-Net$^{\text{UNet}}$ has a sudden increase of performance in the initial stage of the growth phase while the F-1 score of Spatial-Net$^{\text{UNet}}$ increases gradually as we create more partitions.

Sensitivity to parameters: Fig. 10 shows the performance of Spatial-Net$^{\text{UNet}}$ as the data size and significance level α change. As the amount of training data grows (test data remain the same), the performance increases slightly. We can see that Spatial-Net$^{\text{UNet}}$ can reach good performance even with 6.25% training data (better than other methods in Tables 1 and 2 using 25% data). On the other hand, we can see the model has slightly decreased performance when we set a very small significance level α. This is because the significance test becomes more strict for partitioning, making it less sensitive to the heterogeneity across regions. As we increase the significance value to 0.1, we find the performance is similar to the performance we get using 0.05. The performance in D2 drops (but marginally) for large α=0.1 because Spatial-Net creates very small regions where the chance for overfitting increases.

4.5 Capturing underlying spatial heterogeneity

We also generated an illustrative synthetic dataset to demonstrate Spatial-Net's automatic capturing of underlying (unobserved) heterogeneous spatial processes. As shown in Fig. 11 (a), the data contains four different spatial processes (Φ_1 to Φ_4).

In the dataset, y_{true} has continuous values, making this a regression problem. The ground truth (Fig. 11 (c)) is generated using a function of four features (one unobserved), where each feature is randomly simulated (details of the function and features are provided in Appendix A.2). To simulate heterogeneity, function parameters are different for each spatial process Φ. The base network we use here is a seven-layer ANN with Mean-Squared-Error (MSE) as the loss function. Fig. 11 shows the space-partitioning automatically learned by Spatial-Net with 25% data as training and validation (significance level $\alpha = 0.01$); RMSE for test samples is ~1.705. As we can see, SIG-GAC is able to automatically terminate growing if no significant split is found (e.g., $\mathcal{H}_1^3$) and unnecessary partitions can be merged back during collapse (e.g., $\mathcal{H}_2^3$ and $\mathcal{H}_2^2$). The resulting Spatial-Net has four branches to capture the processes.

5 OTHER RELATED WORK

Here we discuss some other related work in addition to those described in Sec. 1. Recent advances in deep learning models, e.g., CNNs [14], segmentation networks [5, 19], and object detection networks [18, 25] have provided unrealized potential for modeling spatial data. These models are capable of automatically extracting complex features from spatial data while also modeling the spatial autocorrelation. Note that the multi-resolution convolution used in CNNs is used within an input sample (i.e., an image), which is not the same as the multi-scale heterogeneity hierarchy here (cross samples). More importantly, these methods commonly produce a single model of the target variable over the entire population of training and test instances and do not to account for data heterogeneity over space, thus resulting in poor prediction performance across diverse spatial regions. It is worth mentioning that our proposed Spatial-Net does not intend to replace the existing networks; instead, it is a model-agnostic framework that can be applied to these deep learning models to handle spatial heterogeneity.

One may also address spatial heterogeneity by transferring knowledge learned from labeled spatial datasets to unseen regions. For example, researchers have developed domain adaptation methods that explore the invariant feature representation across different spatial datasets [11, 27]. Meta-learning methods have also been used to automatically learn models that can be easily adapted to different spatial regions [20, 26]. However, these methods are commonly based on assumptions of similarity on certain aspects of data distributions. For example, many domain adaptations require different regions have similar distribution of classes; and meta-learning methods, e.g., MAML [7, 20], may yield degraded performance

when regions have large discrepancy. In the future, meta-learners can also be integrated into Spatial-Net to speed-up training.

Our work is also related to data clustering, which has the potential of improving predictive performance by training individual models separately for different clusters [13, 23]. However, in real-world problems, related variables may not be observed directly (or often unobserved) but rather inferred through a proxy (e.g., spectral reflectance values captured by satellites). Hence, the collected input features have less expressive power and the clustering obtained using these features may not fully capture spatial heterogeneity. Moreover, clustering can significantly reduce the training data available for each individual model, making it difficult to train complex models. Recent work on mixture pattern mining [24] can identify regions with specified mixture signatures of point processes (e.g., homogeneous mixtures). However, it cannot handle data where processes $\Phi : \mathbf{X} \rightarrow \mathbf{y}$ cannot be described by explicit statistical models (e.g., Poisson), which are most often the case for deep learning tasks. In addition, it aims to find interesting sub-regions but cannot segment the whole space into homogeneous processes.

Another related but different direction is the attention mechanism. Currently, attention methods [6] mainly aim to find "focus" regions within a training or test sample (i.e., weighting information within a sample itself or its features) but do not aim to tackle distribution heterogeneity or non-stationarity, or, to provide explicit hierarchies or distribution-relationships across samples.

6 CONCLUSIONS AND FUTURE WORK

We proposed a novel Spatial-Net with a significance-based grow-and-collapse (SIG-GAC) framework to capture spatial heterogeneity commonly embedded in spatial datasets. Our evaluations demonstrated the effectiveness of the proposed method in improving the predictive performance and addressing spatial heterogeneity. In particular, we show that our method can partition data into spatially-contiguous and homogeneous regions. By sharing parameters over the hierarchical structure, Spatial-Net can achieve good performance even with reduced training data. While in this paper we evaluated Spatial-Net with agricultural and land cover mapping, the approach can be generally applied to various spatial tasks involving heterogeneity (e.g., traffic prediction, weather forecasting, climate change projection).

In future work, we will extend Spatial-Net by developing new methods to: (1) capture spatial processes with irregular footprints; (2) cover a greater variety of network architectures (e.g., LSTM for spatio-temporal tasks, GAN for generative tasks) and other types of machine learning methods; and (3) transfer heterogeneity knowledge learned in a spatial region to new areas. We will also expand synthetic data generation for more comprehensive evaluations.

ACKNOWLEDGMENTS

Yiqun Xie is supported in part by NSF awards 2105133 and 2126474, Google's AI for Social Good Impact Scholars program, and the DRI award at the University of Maryland; Xiaowei Jia is supported in part by USGS award G21AC10207, Pitt Momentum Funds award, and CRC at the University of Pittsburgh; Han Bao and Xun Zhou are supported in part by the ISSSF grant from the University of Iowa, and SAFER-SIM, which is funded by US-DOT award 69A3551747131; and Jia Yu is supported in part by NSF award 2126449.

REFERENCES

[1] Gowtham Atluri, Anuj Karpatne, and Vipin Kumar. 2018. Spatio-temporal data mining: A survey of problems and methods. *ACM Computing Surveys (CSUR)* 51, 4 (2018), 1–41.
[2] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. 2017. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence* 39, 12 (2017), 2481–2495.
[3] Chris Brunsdon, A Stewart Fotheringham, and Martin Charlton. 1999. Some notes on parametric significance tests for geographically weighted regression. *Journal of regional science* 39, 3 (1999), 497–524.
[4] CDL 2021. USDA Cropland Data Layer. https://www.nass.usda.gov/Research_and_Science/Cropland/SARS1a.php.
[5] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. 2018. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *ECCV*. 801–818.
[6] Shuhan Chen, Xiuli Tan, Ben Wang, and Xuelong Hu. 2018. Reverse attention for salient object detection. In *ECCV*.
[7] Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*.
[8] GEOGLAM 2021. GEOGLAM Global Crop Monitoring. https://earthobservations.org/geoglam.php.
[9] Jayant Gupta, Carl Molnar, Yiqun Xie, Joe Knight, and Shashi Shekhar. 2021. Spatial Variability Aware Deep Neural Networks (SVANN): A General Approach. *ACM Trans. on Intelligent Systems and Technology (TIST)* (2021). (in press).
[10] Jayant Gupta, Yiqun Xie, and Shashi Shekhar. 2020. Towards Spatial Variability Aware Deep Neural Networks (SVANN): A Summary of Results. In *ACM SIGKDD Workshop on Deep Learning for Spatiotemporal Data, Applications, and Systems.*
[11] Xiaowei Jia, Guruprasad Nayak, Ankush Khandelwal, Anuj Karpatne, and Vipin Kumar. 2019. Classifying heterogeneous sequential data by cyclic domain adaptation: An application in land cover detection. In *SDM*. SIAM.
[12] Zhe Jiang, Arpan Man Sainju, Yan Li, Shashi Shekhar, and Joseph Knight. 2019. Spatial ensemble learning for heterogeneous geographic data with class ambiguity. *ACM Trans. on Intelligent Systems and Technology (TIST)* 10, 4 (2019).
[13] Anuj Karpatne, Ankush Khandelwal, Shyam Boriah, and Vipin Kumar. 2014. Predictive learning in the presence of heterogeneity and limited training data. In *Proceedings of the 2014 SIAM Intl. Conf. on Data Mining*. SIAM, 253–261.
[14] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems* 25 (2012), 1097–1105.
[15] Jonathan Long, Evan Shelhamer, and Trevor Darrell. 2015. Fully convolutional networks for semantic segmentation. In *ICCV*.
[16] William S Noble. 2009. How does multiple testing correction work? *Nature biotechnology* 27, 12 (2009), 1135–1137.
[17] Hyeonwoo Noh et al. 2015. Learning deconvolution network for semantic segmentation. In *ICCV*.
[18] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. 2016. You only look once: Unified, real-time object detection. In *CVPR*.
[19] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*. Springer, 234–241.
[20] Marc Rußwurm, Sherrie Wang, Marco Korner, and David Lobell. 2020. Meta-learning for few-shot land cover classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*. 200–201.
[21] Sentinel-2 Multispectral Imagery 2021. GEE. https://developers.google.com/earth-engine/datasets/catalog/COPERNICUS_S2_SR#bands.
[22] Shashi Shekhar, Steven K Feiner, and Walid G Aref. 2015. Spatial computing. *Commun. ACM* 59, 1 (2015), 72–81.
[23] Yuliya Tarabalka, Jón Atli Benediktsson, and Jocelyn Chanussot. 2009. Spectral–spatial classification of hyperspectral imagery based on partitional clustering techniques. *IEEE Trans. on Geoscience and Remote Sensing* 47, 8 (2009), 2973–2987.
[24] Yiqun Xie, Han Bao, Yan Li, and Shashi Shekhar. 2020. Discovering Spatial Mixture Patterns of Interest. In *Proceedings of the 28th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. 608–617.
[25] Yiqun Xie, Rahul Bhojwani, Shashi Shekhar, and Joseph Knight. 2018. An unsupervised augmentation framework for deep learning based geospatial object detection: a summary of results. In *Proceedings of the 26th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. 349–358.
[26] Huaxiu Yao, Yiding Liu, Ying Wei, Xianfeng Tang, and Zhenhui Li. 2019. Learning from multiple cities: A meta-learning approach for spatial-temporal prediction. In *The World Wide Web Conference*. 2181–2191.
[27] Yang Zhang, Philip David, and Boqing Gong. 2017. Curriculum domain adaptation for semantic segmentation of urban scenes. In *Proceedings of the IEEE International Conference on Computer Vision*. 2020–2030.

A APPENDIX FOR REPRODUCIBILITY

A.1 Additional notes on SIG-GAC framework

Fig. 12 shows a flow-chart describing the high-level key steps of the growth phase of the SIG-GAC framework to help with implementation. Specifically, the grower G_R is responsible for managing and generating a sequence of candidates for node-split at multiple scales, while the verifier G_V determines whether the impact of a split is statistically significant to reject the null hypothesis H_0. Insignificant splits or candidates with a level greater than lv_{max} will be removed from G_R's candidate set and added to the set $\mathcal{H}_{leaf}$ of stable nodes. Nodes in $\mathcal{H}_{leaf}$ are leaf nodes of the spatial hierarchy $\mathcal{H}$. In our experiments, lv_{min} and lv_{max} are set to 2 and 6, respectively. Alg. 1 shows the pseudo-code for one example implementation of SIG-GAC. We recommend to not use recursive implementations of SIG-GAC as they may incur memory issues.

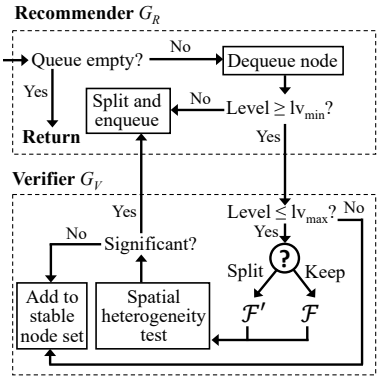


Figure 12: High-level key steps in grower G.

For more robust significance testing, especially for scenarios where sample size is large, we additionally incorporated a test on effect size to reduce non-interesting splits (i.e., two "different" distributions but "small differences"), as recommended for typical p-value based statistical tests. Specifically, effect size here is defined by $\frac{\overline{L(\mathcal{F}(\mathbf{T}_j^i))-L(\mathcal{F}'(\mathbf{T}_j^i))}}{\sigma(L(\mathcal{F}(\mathbf{T}_j^i))-L(\mathcal{F}'(\mathbf{T}_j^i)))}$, where $L(\mathcal{F}(\mathbf{T}_j^i))$ and $L(\mathcal{F}'(\mathbf{T}_j^i))$ are the losses on validation samples before and after the split. It evaluates the mean difference between two groups (i.e., validation samples in $\mathbf{T}_j^i$ before and after the split), which is normalized by the standard deviation of the differences. In other words, effect size measures the scale of the difference as a proportion of the standard deviation. Typically, effect sizes below 0.4 are considered small and values above 0.8 and 1.4 are considered moderate and large, respectively. Effect size is not sensitive to sample size (not using degrees of freedom). In the future, more statistical measures and other conditions will be incorporated to further improve the robustness of the decisions.

A.2 Additional details on synthetic data

The synthetic dataset is generated using four random features with three observed $\{\mathbf{X}_1, \mathbf{X}_2, \mathbf{X}_3\}$ and one unobserved $\mathbf{X}_4$. To incorporate spatial autocorrelation in the synthetic data, the space is randomly partitioned into small tiles for each feature separately, and the values of a feature in each tile (randomly chosen integers) are the same. To mimic autocorrelation at different scales, the unobserved feature is not further partitioned into smaller random tiles, but directly uses the partitions of spatial processes (Φ_1 to Φ_4) and the values in each partition are set to the same random integer value.

Spatial heterogeneity is modeled by the generation function which uses different weights for each spatial process (footprints defined by partitions): $\mathbf{y}_{true}^{part} = \mathbf{X}_4 \odot (\sum_{i=1}^{3} \mathbf{W}_i^{part} \odot \mathbf{X}_i)$, where $part \in \{1 \text{ to } 4\}$ is the ID of the spatial partition of each distinct spatial process; $\odot$ is the element-wise (Hadamard) product; $\mathbf{W}_i^{part}$ is the weight for feature i in partition $part$, and the weight values for each partition are the same for each observed feature i (unobserved feature is used as a multiplier so no additional weights are added).

The partition information is not observed by a learning model, and a model only sees $\{\mathbf{X}_1, \mathbf{X}_2, \mathbf{X}_3\}$ (without unobserved feature $\mathbf{X}_4$) and samples of $\mathbf{y}_{true}$. The generated data has a dimension of 1000×1000. Finally, each training sample is a pair of three features and a y_{true} value at each pixel; 25% of pixels are used for training, 25% for validation, and 50% for testing.

Algorithm 1: SIG-GAC

Require:
- Input data $\mathbf{X}$ and $\mathbf{y}$ in space $\mathcal{D}$
- Deep network $\mathcal{F}$
- Significance level α
- Minimum and maximum split levels lv_{min} and lv_{max}

$\mathcal{F}$.initTrain($\mathbf{X}$, $\mathbf{y}$)
$queue$.init().add(Node($\mathcal{D}$))
{Grow}
while $queue$.notEmpty() **do**
  H = $queue$.dequeue()
  if $H.lv == lv_{max}$ **then** H_stable.add(H), **continue**
  if H.prev_split **is** None **or** H.prev_split == 'vertical' **then**
    $H_1, H_2 = H$.split('horizontal')
  else
    $H_1, H_2 = H$.split('vertical')
  end if
  {Statistical test: lv_{min} is optional (only used in the growth phase)}
  $pvalue$, $\mathcal{F}_{split}$, $\mathcal{F}_{nosplit}$ = testHeterogeneity($\mathcal{F}$, H_1, H_2, $\mathbf{X}$, $\mathbf{y}$, lv_{min})
  if $pvalue < \alpha$ **then**
    $queue$.enqueue(H_1, H_2)
    $\mathcal{F}$.sync($\mathcal{F}_{split}$)
  else
    H_stable.add(H)
  end if
end while
{Collapse}
for lv = lv_{max} to 1 **do**
  while AC_node = getUnvisitedNode(H_stable, lv) **do**
    for (H_a, H_b) in getUnvisitedChildrenPairs(AC_node) **do**
      if isValidPair(H_a, H_b) **then**
        $pvalue$, $\mathcal{F}_{split}$, $\mathcal{F}_{nosplit}$ = testHeterogeneity($\mathcal{F}$, H_a, H_b, $\mathbf{X}$, $\mathbf{y}$)
        if $pvalue \geq \alpha$ **then**
          $\mathcal{F}$.sync($\mathcal{F}_{nosplit}$)
          H_stable.merge(H_a, H_b)
        end if
      end if
    end for
    updateVisitStatus(H_stable, AC_node)
  end while
end for
return $\mathcal{F}$, H_stable

A.3 Code and implementation details

To promote open science and reproducibility, the code and datasets used in the experiments are shared at: https://github.com/yqthanks