A Statistically-Guided Deep Network Transformation and Moderation Framework for Data with Spatial Heterogeneity

Yiqun Xie*
University of Maryland
xie@umd.edu

Erhu He*, Xiaowei Jia
University of Pittsburgh
erh108,xiaowei@pitt.edu

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

Rahul Ghosh, Praveen Ravirathinam
University of Minnesota
ghosh128,pravirat@umn.edu

***Abstract*—Spatial data are ubiquitous, massively collected, and widely used to support critical decision-making in many societal domains, including public health (e.g., COVID-19 pandemic control), agricultural crop monitoring, transportation, etc. While recent advances in machine learning and deep learning offer new promising ways to mine such rich datasets (e.g., satellite imagery, COVID statistics), spatial heterogeneity – an intrinsic characteristic embedded in spatial data – poses a major challenge as data distributions or generative processes often vary across space at different scales, with their spatial extents unknown. Recent studies (e.g., SVANN, spatial ensemble) targeting this difficult problem either require a known space-partitioning as the input, or can only support very limited number of partitions or classes (e.g., two) due to the decrease in training data size and the complexity of analysis. To address these limitations, we propose a model-agnostic framework to automatically transform a deep learning model into a spatial-heterogeneity-aware architecture, where the learning of arbitrary space partitionings is guided by a learning-engaged generalization of multivariate scan statistic and parameters are shared based on spatial relationships. We also propose a spatial moderator to generalize learned space partitionings to new test regions. Experiment results on real-world datasets show that the spatial transformation and moderation framework can effectively capture flexibly-shaped heterogeneous footprints and substantially improve prediction performances.**

***Index Terms*—Deep learning, statistics, spatial heterogeneity**

I. Introduction

Spatial datasets are ubiquitous and collected at ever-growing scale, resolution, frequency and variety. Common types of spatial data include satellite/UAV imagery, points-of-interest (POI), GPS locations/trajectories, geo-tagged tweets, census data, maps (e.g., land cover, crimes, traffic accidents, COVID statistics), and many more. These spatial datasets are critical in a variety of societal applications, such as Earth observation (e.g., crop monitoring [1]), public health (e.g., COVID-19 mobility analysis [2]), public safety, transportation, etc.

While spatial datasets are both important and widely used, they have two intrinsic properties – spatial autocorrelation and heterogeneity – that often undermine the traditional independent and identical distribution (i.i.d.) assumption of data samples [3], [4]. Spatial autocorrelation violates the independence assumption as nearby data samples (e.g., landcover, temperature, mobility) tend to share higher similarity. Spatial heterogeneity, on the other hand, violates the identical distribution assumption as the data generative processes often vary over space. Even more challenging, such differences in distributions may not be reflected by variations in observed features, and the spatial footprints of the generative processes could be arbitrary in shape due to complex social and physical contexts. For example, in satellite-based crop monitoring, relationships between observed spectral characteristics and crop types are affected by many unobserved or hard-to-collect information such as each farmer's adoption of land management practices (e.g., tillage type, applications of phosphorous and pesticides, etc.); these choices often depend on personal experience, planned crop rotation, and local exchanges with other farmers. Similarly, in COVID human mobility projection, travel patterns often differ across regions due to mixed differences in local policy and implementation, social culture, events, community setting (e.g., rural, urban), etc. Unknown spatial footprints of these heterogeneous processes pose significant challenges to applications beyond a very local focus.

In related work (more in Sec. V), the wide adoption of convolutional kernels [5] in deep learning architectures have explicitly filled the missing representation to capture spatial autocorrelation (e.g., local connections and maintained spatial relationships between cells). However, the complex spatial heterogeneity challenge has not been sufficiently addressed. In a recent study, a spatial-variability aware neural network (SVANN) approach was developed [6]. SVANN mainly demonstrates the benefit (e.g., increase in accuracy) of separating out training data subsets belonging to known different distributions, but it requires the spatial footprints of heterogeneous processes to be known as an input, which is often unavailable in real applications. Explicit spatial ensemble approaches aim to adaptively partition a dataset [7], but the algorithm and its variation are specifically designed for two-class classification problems and only allow two partitions; both training and prediction are performed separately for each partition. Outside recent literature on deep learning, a traditional approach to handle spatial heterogeneity is geographically-weighted regression (GWR) [8]. However, GWR is mainly designed for inference and linear regression, and cannot handle complex prediction tasks commonly addressed by deep learning. Most existing methods also require dense training data across space to train models for individual partitions or locations. Finally, they cannot be applied to other regions

outside the spatial extent of the training data.

To address these limitations, we propose a model-agnostic Spatial Transformation And modeRation (STAR) framework with the following contributions:

- We propose a spatial transformation approach to capture arbitrarily-shaped footprints of spatial heterogeneity at multiple scales during deep network training, and synchronously transform the network into a new "spatialized" architecture. The transformation is guided by a dynamic and learning-engaged generalization of multivariate scan statistic;
- We propose a spatial moderator to generalize the learned spatial patterns and transformed network architecture from the original region to new test regions;
- We implement the model-agnostic STAR framework using both snapshot and time-series based input network architectures (i.e., DNN, LSTM and LSTM-attention), and present the statistically guided transformation module for both classification and regression tasks.

Through experiments on real world datasets, i.e., satellite-based crop monitoring and COVID-19 human mobility projection, we show that the STAR framework can substantially improve model performance, capture flexibly-shaped spatial footprints of heterogeneous processes, and can be effectively applied to prediction tasks in new test regions.

II. Problem Formulation

The general problem is formulated as follows:

Inputs:

– Geo-located feature **X** and label **y** in a spatial domain $\mathcal{D}$;
– Spatial locations **L** of data samples;
– A deep learning model $\mathcal{F}$ selected for the task;
– A significance level α;

Outputs:

– A flexibly-shaped space-partitioning scheme D_{part} of $\mathcal{D}$;
– A spatially-transformed $\mathcal{F}$: $\mathcal{F}_{spatial}$ on D_{part};

Objective: The goal is to improve solution for:

– Classification (e.g., precision, recall, F1-scores);
– Regression (e.g., MAE, RMSE).

As our spatial transformation and moderation framework aims to incorporate awareness of spatial heterogeneity into a deep learning model selected by the user, input data to this framework need to contain location information, which can be either explicitly recorded (e.g., POI visits; trajectories) or implicitly inferred (e.g., pixels in a satellite imagery). In many real-world use cases, ground-truth labels (e.g., crop types) are collected through field surveys only at certain sample locations (i.e., not a complete map), so location information also allows those labels to be matched onto the observed features (e.g., spectral bands in satellite imagery). Based on the prediction task and data types, a user can specify a desired deep learning model (e.g., DNN, LSTM, CNN) as an input. Using this as a base model, our framework will simultaneously capture the spatial heterogeneity in the data via flexibly-shaped space-partitioning, and transform the base model into its spatial version. The significance level α will be used to guide decisions during the transformation.

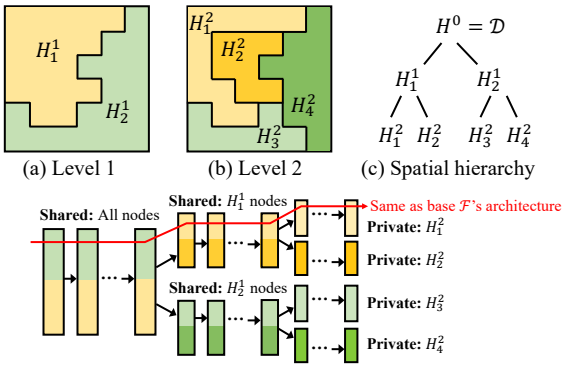

Fig. 1: Spatial processes, hierarchy and network architecture.

III. A Model-Agnostic Deep Network Transformation and Moderation Framework

A. Spatially heterogeneous processes

In this section, we first define basic concepts on spatial heterogeneity and then outline key questions to address it.

Definition 1. ***Spatial process*** Φ*: A function* $\Phi : \boldsymbol{X} \mapsto \boldsymbol{y}$ *governing data generation in a spatial region, which may involve observed and unobserved (or unknown) features as variables. The process at a smaller/finer scale may be an aggregation of itself and processes at larger scales.*

Definition 2. ***Spatial heterogeneity:*** *An intrinsic property of spatial data [3], [4] stating that data are generated by different spatial processes* $\{\Phi\}$ *across space. Spatial heterogeneity leads to different data distributions in different regions.*

While deep networks can function as universal approximators for data following identical distributions [9], spatial heterogeneity commonly existed in spatial data violates this assumption (e.g., spatial data generated by two simple scalar functions $y = x$ and $y = -x$ across space cannot be approximated by a single network). As a result, the heterogeneous processes $\{\Phi\}$ will cause confusion on data distribution during training, and hamper prediction performance and stability.

Moreover, another complicating factor we need to consider is the hierarchy of spatial processes across scales and their corresponding heterogeneity. For example, higher-level heterogeneity in the hierarchy may be caused by policies at larger scales, climate zones, major geographical barriers (e.g., mountains), whereas lower-level processes may vary by local policies, demographics, social/cultural contexts, and personal decisions. In addition, the spatial footprints of these different processes may be arbitrary in shape. Fig. 1 (a) and (b) show an example of mixtures of spatial processes at two different scales/levels, and this hierarchy is formally defined in Def. 3.

Definition 3. ***Spatial hierarchy of processes*** $\mathcal{H}$*: A multi-scale representation of spatial heterogeneity [10].* $\mathcal{H}$ *represents the input spatial domain* $\mathcal{D}$ *as a tree; each node* $\mathcal{H}_j^i \in \mathcal{H}$ *is a partition of* $\mathcal{D}$*, where* i *denotes the level in the hierarchy, and* j *is the unique ID for each partition at level-*i*. Children of a partition* $\mathcal{H}_j^i$ *share the same lower-level processes (processes*

$\{\Phi\}$ *at levels* $i' < i$*). The process* Φ *is homogeneous within a leaf-node and heterogeneous across leaf-nodes.*

Based on the definitions and concepts, there are three key questions we need to address to transform an input deep learning model $\mathcal{F}$ into a spatial-heterogeneity-aware $\mathcal{F}_{spatial}$:

– What is a learning representation to utilize spatial relationships among data samples to allow: (1) samples following heterogeneous processes to contribute to different models, and (2) effective weight-sharing among models?
– How to adaptively learn the often arbitrarily-shaped footprints of spatially heterogeneous processes, which may contain a mixture of processes across multiple scales?
– How to generalize $\mathcal{F}_{spatial}$ and space-partitioning learned in one region to be effectively used in other test regions?

In the following Sec. III-B to III-D, we will address the questions with a representation choice, a statistically-guided spatial transformation of $\mathcal{F}$, and a spatial moderator.

B. Representation choice: Hierarchical multi-task learning

To handle spatial heterogeneity, the representation needs to be specified at both data and deep network model levels. Fortunately, the spatial hierarchy defined in Def. 3 [10] not only provides a natural way to represent spatial heterogeneity across scales, but also an effective structure to hierarchically group deep network parameters for the training process. To illustrate this, Fig. 1 (c) shows an example of spatial hierarchy $\mathcal{H}$, where each node $\mathcal{H}_j^i \in \mathcal{H}$ can be considered as a spatial region with a spatial process Φ_j^i; here i is the level in the hierarchy and j is a unique ID of a node at this level. Based on this hierarchical representation of spatial partitions, Fig. 1 (d) shows the deep network representation that synchronizes the structure of $\mathcal{H}$, where each unique path from the input to output has the same architecture as the input deep network $\mathcal{F}$. Using this representation, model parameters at each layer are shared by all leaf nodes branched out from the layer. This means nodes that share more common parent nodes in the spatial hierarchy $\mathcal{H}$ also share more common weights. Another intuitive interpretation is that spatial partitions that share the same parent $\mathcal{H}_j^i$ inherit the same higher level spatial process Φ_j^i. The learning at each leaf-node can be considered as a task in this multi-task learning context.

For the hierarchy-network synchronization (Fig. 1), a final detail is the selection of the layer, at which the following layers will be split into two parallel branches. To make this more formal, we use an optional parameter β ($\beta \leq 1$; default to 1/2) to denote the proportion of the layers to split.

C. Statistically-guided deep network transformation

While transforming an input network $\mathcal{F}$ into the hierarchical spatial representation $\mathcal{F}_{spatial}$ is straightforward, the most critical task is to actually learn this spatial hierarchy in the first place. We propose a statistically-guided transformation algorithm to adaptively capture the hierarchy $\mathcal{H}$ and the synchronized network architecture $\mathcal{F}_{spatial}$.

Following the spatial hierarchical structure (Def. 3), the space-partitioning (and network transformation) will propagate in a hierarchical bi-partitioning fashion, where at each step, a partition or node $\mathcal{H}_j^i \in \mathcal{H}$ at the current level will be split into two children $\mathcal{H}_{j1}^{i+1}$ and $\mathcal{H}_{j2}^{i+1}$ with arbitrarily-shaped spatial footprints. As shown in Fig. 2, this process is governed and automated by spatial statistical tests on the following overarching hypotheses: (1) **Null hypothesis** H_0: The spatial process Φ_j^i at node $\mathcal{H}_j^i$ is homogeneous (i.e., no need for partitioning), and (2) **Alternative hypothesis** H_1: Φ_j^i is a mixture of heterogeneous spatial processes.

Our transformation framework is a dynamic and learning-engaged generalization of the multivariate scan statistic [11]–[13] as we will discuss over the next two sections.

1) Multivariate scan statistic (MSS): MSS [11], [12] is a widely applied spatial statistical approach in event detection (e.g., disease surveillance) [13]. It identifies if there exists a spatial region with a significantly higher rate of generating incidents or cases of certain events (e.g., disease, crime) compared to the rest. To better illustrate the formulation of MSS, denote $c_{k,m}$ and $b_{k,m}$ as the observed and expected (baseline) number of cases or incidents of event m at spatial location s_k, respectively; where $m = 1, ..., M$, and the expectation $b_{k,m}$ can be calculated using the total number of cases C_m of event m and the proportion of "base population" at location s_k. For example, using COVID-19 as the event, the "base population" can be the total number of tested people, and the number of cases will cover those tested positive. Next, the null hypothesis H_0 states that the data generative process is homogeneous across the whole space (the expectation or baseline $b_{k,m}$ is calculated under this hypothesis); and H_1 states that there exists a region S where the rate of generating instances of an event is q_m times the expected rate under H_0, i.e., the expectation in S is $q_m \cdot b_{k,m}$. As there exist a large number of spatial regions S, MSS finds the most "divergent" region by maximizing the Poisson-based log likelihood ratio [12]:

$$\begin{aligned} S^* &= \arg\max_S \Gamma_{mss}(S) = \arg\max_S \log \frac{\text{Likelihood}(H_1, S)}{\text{Likelihood}(H_0)} \\ &= \arg\max_S \log \prod_{s_k \in S} \prod_{m=1}^{M} \frac{\Pr(c_{k,m} \sim \text{Poisson}(q_m \cdot b_{k,m}))}{\Pr(c_{k,m} \sim \text{Poisson}(b_{k,m}))} \end{aligned} \tag{1}$$

For each specific candidate region S, q_m is estimated by maximizing Likelihood(H_1, S), yielding:

$$S^* = \arg\max_S \sum_{m=1}^{M} \left(C_{m,S} \cdot \log \left(\frac{C_{m,S}}{B_{m,S}} \right) + B_{m,S} - C_{m,S} \right) \tag{2}$$

where $C_{m,S} = \sum_{s_k \in S} c_{k,m}$; $B_{m,S} = \sum_{s_k \in S} b_{k,m}$; q_m is replaced by its maximum likelihood estimate: $\max\{\frac{C_{m,S}}{B_{m,S}}, 1\}$.

In MSS, after S^* is identified from the observed dataset, it evaluates the statistical significance of S^* through Monte Carlo estimation with T trials (e.g., 999): $MC_1, ..., MC_T$. In each trial MC_t, a simulation data is generated using H_0, and the optimal S_t^* and its score $\frac{\text{Likelihood}(H_1, S_t^*)}{\text{Likelihood}(H_0)}$ are extracted from it. Finally, given an input significance level α, S^* is significant

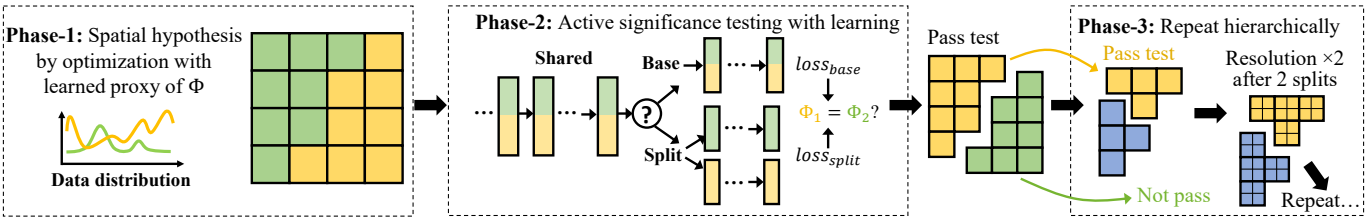

Fig. 2: Illustrative example of the spatial transformation framework with dynamic and learning-engaged MSS.

(i.e., the data generative process is heterogeneous) if its score $\frac{\text{Likelihood}(H_1,S^*)}{\text{Likelihood}(H_0)}$ is in the top α portion of the optimal scores $S^* \cup \{S_t^* | t = 1, ..., T\}$. The enumeration of region candidates S will be discussed later.

2) A dynamic and learning-engaged generalization of MSS (DL-MSS): There are three gaps in integrating MSS into the spatial-heterogeneity-aware deep learning transformation process: (1) *Data compatibility:* Input data to a deep learning model are features $\mathbf{X} \in \mathbb{R}^{N \times d}$ (using 1D data samples as an example) and labels $\mathbf{y} \in \mathbb{Z}^N$ (or $\mathbb{R}^N$), which are not directly compatible with MSS inputs (e.g., observed and expected cases $(c_{k,m}, b_{k,m})$ at a location s_k); (2) *Significance in action:* In MSS, the statistical test is performed using Monte Carlo estimation, which is more "descriptive" about the past or presence. However, we are more interested in "futuristic" impact of a spatial pattern S^*, i.e., our real goal is to know if the partitioning based on the pattern can truly make a statistically significant improvement on the learning; and (3) *Dynamics of learning:* As the complex relationships between $\mathbf{X}$ and $\mathbf{y}$ are not known in input data (unlike MSS), footprints of heterogeneous spatial processes $\{\Phi\}$ need to be dynamically captured as new partitions in $\mathcal{H}$ are created and new parameters are learned.

We propose a Dynamic and Learning-engaged MSS (DL-MSS) to bridge the gaps through three phases.

DL-MSS Phase-1: Prediction error distribution as a proxy to heterogeneous processes Φ. To transform input features $\mathbf{X}$ and labels $\mathbf{y}$ to a "observation vs. expectation" distribution as needed by MSS, we use the spatial distribution of prediction errors as a proxy to the spatial processes. Here we will use classification as an example to illustrate the modeling and the regression variation will be discussed in Sec. III-C5.

There are two main reasons of using error distribution as the proxy for spatial processes: (1) If all data belonging to a partition $H_j^i \in \mathcal{H}$ are generated by a homogeneous spatial process Φ_j^i, we expect the error distribution for each class – that are predicted by a single model at node H_j^i – to follow a homogeneous distribution as well. Otherwise, errors following spatially heterogeneous distributions would indicate that the data generative process Φ_j^i is heterogeneous (i.e., $\mathbf{X} \mapsto \mathbf{y}$ are different across locations within partition $\mathcal{H}_j^i$); and (2) The use of prediction errors enables the use of deep learners to generate statistics (MSS inputs) to describe processes $\Phi : \mathbf{X} \mapsto \mathbf{y}$, which are otherwise unavailable or hidden from the input data.

Denote $\hat{\mathbf{y}}_{k,m}$ as the predicted labels for samples with class m (i.e., true labels are m) at spatial location s_k (e.g., a cell in a grid-partitioning of space). The number of misclassified samples of class m at s_k is then $err_{k,m} = |\hat{\mathbf{y}}_{k,m} \neq m|$. Further, denote $n_{k,m}$ as the number of samples of class m at location s_k; and ERR_m and N_m as the number of misclassified and all samples of class m in the entire space. Using $n_{k,m}$ as the "base population", the expected number of misclassified samples at location s_k is then $E(err_{k,m}) = ERR_m \cdot \frac{n_{k,m}}{N_m}$. With this modeling, the error distribution across space can be now characterized by MSS by replacing $c_{k,m}$ in Eq. (1) with $err_{k,m}$ and $b_{k,m}$ with $E(err_{k,m})$. In other words, we are trying to find a spatial region S that has the most divergent error distribution from the rest of the space.

The optimal solution S^* can still be given by Eq. (2), and a worth-mentioning property is that the multivariate likelihood ratio in Eq. (2) automatically adjusts for sample size in a spatial region S, improving the flexibility of the approach.

Once the optimal S^* is identified, the current node $\mathcal{H}_j^i$ will be temporarily split into two children $\mathcal{H}_{j1}^{i+1}$ and $\mathcal{H}_{j2}^{i+1}$, where one child corresponds to S^* and the other for the rest of the space in $\mathcal{H}_j^i$. The temporary split will only be implemented in $\mathcal{H}$ if it passes the significance test in the next phase.

DL-MSS Phase-2: Active significance testing with learning. As described in Sec. III-C1, MSS performs significance testing via expensive Monte Carlo simulation. More importantly, the result is by design "descriptive", meaning it only intends to tell if the error distribution in S^* differs from the rest for "the current $\mathcal{H}$-node and model".

As our goal is to know whether a node-split suggested by Phase-1 really partitions the current $\Phi_j^i : \mathbf{X} \mapsto \mathbf{y}$ into two distinct processes Φ_{j1}^{i+1} and Φ_{j2}^{i+1} that lead to a statistically significant improvement on learning, in DL-MSS, we change the Monte-Carlo-based descriptive test to a learning-engaged active test. Denote $\mathbf{\Theta}_j^i$ as deep network parameters for partition $\mathcal{H}_j^i \in \mathcal{H}$; note that $\mathbf{\Theta}_j^i$ shares part of the parameters with other partitions having common parents (Sec. III-B). DL-MSS carries out two sets of learning-engaged experiments to prepare for the statistical test:

- **Split scenario:** Using the temporary split $(\mathcal{H}_j^i \rightarrow (\mathcal{H}_{j1}^{i+1}, \mathcal{H}_{j2}^{i+1}))$ from Phase-1, DL-MSS trains their network parameters $(\mathbf{\Theta}_{j1}^{i+1}, \mathbf{\Theta}_{j2}^{i+1})$ separately using training samples from the two partitions, evaluates the element-wise loss separately on validation samples, and concatenates the two sets of loss to $loss_{split} \in \mathbb{R}^n$, where n is the number of validation samples in $\mathcal{H}_j^i$.
- **Base scenario:** DL-MSS trains parameters $\mathbf{\Theta}_j^i$ for the base node (unsplit) with all training samples from the node together, and evaluates the element-wise loss $loss_{base} \in \mathbb{R}^n$

on validation samples. The order of validation samples are kept the same as in the split scenario. In addition, we also output $loss'_{base}$, which is the loss before the extra training is performed here (to get $loss_{base}$).

Then, DL-MSS performs significance testing using $loss_{base}$ and $loss_{split}$ as the observed measurements of learning performance on the samples. As both $loss_{base}$ and $loss_{split}$ refer to the same set of samples, evaluation of their statistical difference needs to be done using dependent statistical tests to adjust for "same-group" comparisons. Specifically, we use the upper-tailed dependent T-test [10], where $loss_{base}$ and $loss_{split}$ are considered as the scores "before" and "after" the split, and we are only interested in the case where the performance improves. The test statistic is then:

$$diff = \frac{\mu(loss_{split} - loss_{base})}{\sigma(loss_{split} - loss_{base}) \cdot (DF+1)^{-\frac{1}{2}}} \tag{3}$$

where $\mu(\cdot)$ and $\sigma(\cdot)$ are the mean and standard deviation, $DF = n - 1$ is the degree of freedom.

The significance of $diff$ can be tested directly using standard upper-tailed T-test table with DF and significance level α. In addition, to improve the robustness of the testing, we add another effect size test to evaluate the size of improvement:

$$es = \frac{\mu(loss_{split} - loss_{base})}{\mu(loss_{base} - loss'_{base})} \tag{4}$$

Here the denominator measures the improvement achieved purely by the additional training itself, whereas the numerator measures the extra improvement gained from the node split. In our implementation, the threshold on es is defaulted to 1.

DL-MSS Phase-3: MSS in a dynamic and learning-engaged spatial hierarchy $\mathcal{H}$. The original MSS is more of a run-and-done algorithm that aims to detect all heterogeneous regions directly on the input dataset. In other words, it assumes all heterogeneous processes in the current data are readily detectable. However, in our problem, although the underlying spatial heterogeneity is fixed in input data $\mathbf{X}$ and $\mathbf{y}$, delineation of the heterogeneous footprints needs to: (1) engage learning so that the processes $\Phi : \mathbf{X} \mapsto \mathbf{y}$ become observable (e.g., via the error distribution); and (2) follow the dynamic construction process of the hierarchy $\mathcal{H}$, because parameters learned at $\mathcal{H}$-nodes needs to be dynamically refined as new nodes are created to gradually capture heterogeneity at finer scales.

Thus, to capture the spatial heterogeneity in a hierarchical manner, DL-MSS performs the first two phases as a subroutine at new nodes added to $\mathcal{H}$. If a node-split is determined to be significant, the DL-MSS will further expand that branch of $\mathcal{H}$; otherwise, DL-MSS terminates the exploration at the node and mark it as a leaf-node with a homogeneous process.

3) Computation and implementation: So far we have outlined the three phases of DL-MSS. From a computational perspective, the remaining key question is how to efficiently enumerate candidate regions $\{S\}$ in order to identify $S^* \in \{S\}$ at each node throughout the construction of $\mathcal{H}$ as well as its synchronized network architecture $\mathcal{F}_{spatial}$.

First, for general input datasets with location information ($\mathbf{L}$ defined in Sec. II), we use a $g_1 \times g_2$ grid G to represent the space at each node in $\mathcal{H}$. The resolution of the grid gradually increases as the depth of the hierarchy $\mathcal{H}$ increases so that heterogeneity at larger scales are captured at lower resolution and finer-scale heterogeneity are captured in more details. Specifically, when DL-MSS starts at the root of $\mathcal{H}$, G adopts the original $g_1 \times g_2$ resolution (e.g., 8×8). Then, as shown in Fig. 2 (Phase-3), each cell is divided into four equal size cells (i.e., doubling the resolution) if two node-splits have been made at the current resolution, which keeps the average number of cells per node similar for levels $i \in \{i | i \bmod 3 = 0\}$ (same if children nodes are constrained to have equal number of cells when split).

In DL-MSS, grid cells are used as spatial locations $\{s_k\}$ during the optimization of S^*, so the observed and expected number of misclassified samples $c_{k,m}$ and $b_{k,m}$ in Eq. (1) can be calculated as aggregated counts at cell levels.

Next, to identify arbitrarily-shaped S^*, the computational challenge is that the number of candidate regions $|\{S\}|$ (i.e., different subsets of locations) is exponential to the number n of locations or cells – $O(e^n)$. Thus, we utilize the linear-time subset scanning (LTSS) property to reduce the search space:

Definition 4. ***LTSS property [14].*** *Given: (1) a set of spatial locations $\{s_k\}$, and (2) a score function $\Gamma(S)$ for region-ranking where $S \subseteq \{s_k\}$ is a spatial region, the LTSS property holds if there exists a priority function $\gamma(s_k)$ so that:*

$$\max_S \Gamma(S) = \max_{\hat{s}} \Gamma\left(\bigcup_{\gamma(s_k) \geq \gamma(\hat{s}), \forall s_k} s_k \right) \tag{5}$$

When LTSS holds, all spatial locations can be pre-sorted using the priority function $\gamma(s_k)$ in a descending order. Then, by Def. 4, only a linear scan on the sorted list is needed to find the optimal $\hat{s}$ to partition the locations into two sets, where $S^* = \{s_k | \gamma(s_k) \geq \gamma(\hat{s})\}$. This reduces the search cost from $O(e^n)$ to $O(n \log n + n)$, where $O(n \log n)$ is for pre-sorting.

Fortunately, the likelihood ratio function we use here in DL-MSS (Eq. (1)) has been shown to satisfy the LTSS property [12], with the priority function given by:

$$\gamma(s_k) = \sum_{m=1}^{M} (c_{i,m} \log q_m + b_{i,m}(1 - q_m)) \tag{6}$$

where M is the total number of classes.

As introduced in Eq. (1) and (2), q_m here represents how many times the error generation rate in a region S is as high as the expected rate under H_0, and it is an unknown variable in H_1 that need to be estimated. Thus, for LTSS to work, values for q_m must be assigned before the optimal S^* is identified in order to use the priority function in Eq. (6).

To address this issue, we modify a coordinate ascent type of strategy used with LTSS to optimize q_m and S^* in an alternating manner over iterations (Alg. 1). In the algorithm we change the initialization method used by [12], which uses $q_m = e^u$ with $u \sim \text{Uniform}[0, 2]$ and did not perform stably

in our experiments as the randomly generated values are far outside the normal q_m value ranges in our input data. Instead, we initialize q_m values using observed sample values in input:

$$q_m = \arg\max_{q_m} \prod_{s_k \in S_{top}} \prod_{m=1}^{M} Pr(c_{k,m} \sim Poisson(q_m \cdot b_{k,m}))$$
$$= (\sum_{s_k \in S_{top}} c_{k,m}) / (\sum_{s_k \in S_{top}} b_{k,m})$$

where m is the class ID, $S_{top} = \{s_k | \frac{c_{k,m}}{b_{k,m}} \geq \tau, \forall s_k\}$, and τ is the median of $\frac{c_{k,m}}{b_{k,m}}$ at all locations. This initialization can be interpreted as optimizing the values of q_m (same maximum likelihood estimator as used for Eq. (2) and coordinate ascent iterations) using locations in S_{top}, whose members are selected using $\frac{c_{k,m}}{b_{k,m}}$ as a heuristic priority function (initialization only).

Algorithm 1 Coordinate ascent for q_m and S^*

Require:
- c_list: List of all $c_{k,m}$ values for input locations
- b_list: List of all $b_{k,m}$ values for input locations
- score function Γ and priority function γ

{Initialization}
for m = 1 to M **do**
  S_{top} = get_top_cells(c_list, b_list, m)
  $q[m]$ = optimize_q(S_{top}, c_list, b_list, m)
end for
{Coordinate ascent: S^* followed by q}
for i = 1 to $max\_iteration$ **do**
  γ_list = get_priority(q, c_list, b_list, priority_func: γ)
  γ_list = γ_list.sort('desc')
  S^* = maximize_score_by_LTSS(γ_list, c_list, b_list, score_func: Γ)
  for m = 1 to M **do**
    $q[m]$ = optimize_q(S^*, c_list, b_list, m)
  end for
end for
return S^*

Finally, as the region S^* detected by LTSS in Alg. 1 may not be necessarily spatially contiguous (i.e., locations that are consecutive by priority γ may not be spatially adjacent), we refine the partition with extra spatial smoothing (the localized scan in [12] does not work for our purpose as it tends to limit partitions to small and localized footprints). Specifically, at the final iteration of Alg. 1, connected components in S^* (i.e., subsets of grid cells) with a size that are smaller than a tolerance (defaulted to 3 cells) are swapped to the other partition $S' = S^i_j \backslash S^*$ where S^i_j is the entire space at node $\mathcal{H}^i_j$. Similarly, for S', we do the same swap of tiny components.

4) Complexity analysis: Here we provide the time complexity for Alg. 1 at a $\mathcal{H}$-node. Denote n as the total number of samples and grid cells at the node, m as the number of classes, and t as the number of iterations (e.g., 1000). The complexity is then $O(t \cdot (n \log n + n + mn))$ (the cost of initialization and contiguity refinement is minimal and skipped here). Here the number of classes can be often considered as a constant, so the complexity reduces to $O(t \cdot n \log n)$. As described in Phase-3 of DL-MSS (Sec. III-C2), the number of cells of the grid is often very small at each node (e.g., 10s to 100s). Overall, we noticed that the total time spent on S^* optimization is mostly negligible compared to the training time of network parameters in our experiments (e.g., second/minute vs. hour).

5) Regression version of DL-MSS: For regression, the general flow is remains the same and the major differences are for the score function $\Gamma(S)$ and priority function $\gamma(s_k)$, which are needed as the prediction changes from multi-class labels to continuous values.

In this paper we focus on the scenario where each sample has one target label, i.e., $\mathbf{y} \in \mathbb{R}^{N \times 1}$, where N is the number of samples. Also, instead of classification errors, we use mean squared errors e_k for regression. For the **score function** $\Gamma(S)$, we select the normal-based likelihood ratio [15] (Poisson is used for classification), where the null hypothesis H_0 states that $e_k \sim \text{Normal}(\mu_{all}, \sigma^2_{all})$ at all locations and H_1 states that there exists a region S where $e_k \in S \sim \text{Normal}(\mu_S, \sigma^2_{both})$, and $e_k \in S' \sim \text{Normal}(\mu_{S'}, \sigma^2_{both})$ for all other locations S' (a common variance σ^2_{both} is used for both). To avoid redundancy, the simplified $\Gamma(S)$ and S^* are (e.g., used by [15]):

$$\Gamma(S) = N \ln \frac{\sigma_{all}}{\sigma_{both}} - \frac{N}{2} + \sum_{s_k \in S \cup S'} \frac{(e_k - \mu_{all})^2}{2\sigma^2_{all}} \quad (7)$$

$$S^* = \arg\max_S \Gamma(S) = \arg\min_S \sigma_{both} \quad (8)$$

where only $N \ln \sigma^{-1}_{both}$ in $\Gamma(S)$ depends on S so maximizing $\Gamma(S)$ is equivalent to minimizing σ_{both} or the variance σ^2_{both}.

Based on Eq. (7), we have the following lemma:

Lemma 1. *For $\Gamma(S)$ given in Eq. (7), the following **priority function** satisfies the LTSS property:*

$$\gamma(s_k) = e_k \quad (9)$$

Proof. The set of $\{e_k\}$ for locations $\{s_k\}$ is equivalent to a set of points distributed on a one-dimensional line. Moreover, the maximum likelihood estimators for the means and variances are $\mu_S = |S|^{-1} \sum_{s_k \in S} e_k$, $\mu_{S'} = |S'|^{-1} \sum_{s_k \in S'} e_k$, and $\sigma^2_{both} = N^{-1}(\sum_{s_k \in S}(e_k - \mu_S)^2 + \sum_{s_k \in S'}(e_k - \mu_{S'})^2)$. Thus, minimizing the variance σ^2_{both} (or $\sigma_{both} > 0$) is equivalent to minimizing the k-means loss with $k = 2$. So for the two groups to be optimal, there should be no overlap in their e_k value ranges on the 1D space, i.e., $\min_{s_k \in S} e_k \geq \max_{s_k \in S'} e_k$ assuming $\mu_S \geq \mu_{S'}$ (proof is symmetric for the other direction). Otherwise, swapping the minimum $e_k \in S$ and maximum $e_k \in S'$ must reduce the k-means loss (i.e., $N\sigma^2_{both}$), either by center assignments or re-estimation. □

D. A spatial moderator for generalization

The spatial hierarchy $\mathcal{H}$ and "spatialized" deep network $\mathcal{F}_{spatial}$ learned and trained from the transformation step aim to capture spatial heterogeneity for the spatial extent of the input $\mathbf{X}$ and $\mathbf{y}$. However, the partitions cannot be directly applied to a new spatial region. To bridge this gap, we propose a spatial moderator, which translates the learned network branches in $\mathcal{F}_{spatial}$ to prediction tasks in a new region.

The key idea of the spatial moderator is to learn and predict a weight matrix $\mathbf{W}$ for all branches in $\mathcal{F}_{spatial}$ (corresponding

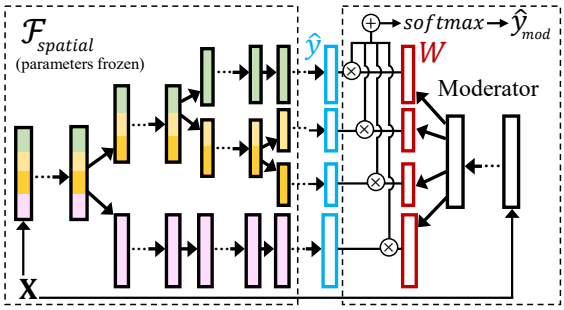

Fig. 3: Illustrative example of the spatial moderator.

to all leaf-nodes in the spatial hierarchy $\mathcal{H}$), and then use the weights to ensemble the prediction results from the branches to get the final result. As an example, suppose the output $\hat{\mathbf{y}}_i$ for a sample $\mathbf{x}_i$ is in 1D. Then, in the weight matrix $\mathbf{W} \in \mathbb{R}^{L \times M}$, the L rows each corresponds to a network branch in $\mathcal{F}_{spatial}$ (or a leaf node in $\mathcal{H}$), where the M columns correspond to the M labels (one-hot encoding for classification; $M = 1$ for regression in this paper). Thus, every column in $\mathbf{W}$ is a weight vector $\mathbf{w} \in \mathbb{R}^L$ that gives the weight distribution across the branches for each of the M labels.

In the spatial moderator, $\mathbf{W}$ is not stationary but predicted dynamically using each input data sample $\mathbf{x}_i$. Fig. 3 shows the general architecture of the moderator. The left side shows an example of "spatialized" network $\mathcal{F}_{spatial}$ and the right side shows the corresponding spatial moderator. For a given sample $\mathbf{x}_i$, the original $\mathcal{F}_{spatial}$ only generates predictions for the branch that $\mathbf{x}_i$ spatially belongs to, i.e., $\hat{\mathbf{y}}_i \in \mathbb{R}^M$. In this moderated version, $\mathcal{F}_{spatial}$ will generate predictions from all L branches for a sample $\mathbf{x}_i$, i.e., $\hat{\mathbf{y}}_i \in \mathbb{R}^{L \times M}$. Then, the moderator predicts the weight matrix $\mathbf{W} \in \mathbb{R}^{L \times M}$ using the same sample $\mathbf{x}_i$, and the final moderated prediction is:

$$\hat{\mathbf{y}}_{mod,i} = \text{softmax}(\mathbf{1}^T(\hat{\mathbf{y}}_i \odot \mathbf{W})) \tag{10}$$

where $\mathbf{1} \in \mathbb{R}^L$ is a vector of ones [1,1...,1], and $\odot$ is element-wise (or Hadarmad) product.

The layer structure of the moderator we use here is a network with four densely connected layers with ReLU activations. If input $\mathbf{x}_i$ is a time-series rather than a snapshot of features, the final layer is replaced with a LSTM layer (i.e., the first three layers are used to construct new features from the original input features, regardless of the timestamps, and the temporal patterns are learned through the final LSTM layer [16]). During the training of the moderator, all parameters from $\mathcal{F}_{spatial}$ will be frozen, and the moderator only needs to learn the weights for itself (the right side of Fig. 3).

IV. Experiments

A. Real-World Datasets

1) California land-cover classification: We use multi-spectral data from Sentinel-2 satellites in two regions in Central Valley, California. Each region has a size of 4096×4096 (∼6711 km^2 in 20m resolution). The regions contain a wide variety of crops with strong heterogeneous patterns, resulting in a challenging classification task. The land cover types are listed in Table I. We first learn the spatial partitioning using the data from Region $\mathcal{D}_A$ and then use the moderator to transfer it to Region $\mathcal{D}_B$. We use composite image series from May to October in 2018 (2 images/month) for time-series models, and one snapshot from August, 2018 for DNN. The labels are from the USDA Crop Data Layer (CDL) [17]. The training (and validation) set has 20% data at sampled locations in $\mathcal{D}_A$, and 1% data in $\mathcal{D}_B$ is used for fine-tuning.

2) Boston COVID-19 human mobility prediction: Human mobility provides critical information to COVID-19 transmission dynamics models. We acquired the Boston COVID-19 mobility dataset shared by [18], which includes data from US census, CDC COVID statistics, and SafeGraph patterns data. In this dataset, human mobility $\mathbf{y}$ is represented by the number of visits to points-of-interest (POIs; e.g., grocery stores, restaurants) and the counting is based smartphone trajectories. We keep the same features $\mathbf{X}$ used in [18], including population, weekly COVID-19 cases and deaths, number of POIs, week ID and income. The spatial representation of the data is a grid-partitioning (37×48) of the Boston area, and each cell is a data sample. Note that grid cells here are used to model the input data (similar to pixels in the California land-cover data), and is independent from the grid G we used in our partition-optimization approach (Sec. III-C2). The dataset contains 12 weeks of data, and according to [18], we use the first 11 weeks for training/validation and the final week for testing.

B. Base models $\mathcal{F}$, Implementation and Training

We implemented the spatial transformation and moderation framework for both snapshot- and time-series-based network models. Specifically, for snapshot models, we use densely connected network (DNN) to learn from data labels sampled at a subset of locations (commonly used in real-world field surveys for ground-truth collection). For time-series models, we use both LSTM and LSTM with attention [16], that were developed for land-cover mapping task with time-series data. All the models have 7 hidden layers, each with 10 neurons and a ReLU activation, to learn and construct new features from raw inputs. For LSTM, an extra LSTM layer is added at the end to learn temporal patterns, and another attention layer [16] is further attached for LSTM+Attetion. A softmax layer is used for classification.

Each model (i.e., DNN, LSTM, LSTM+Attention) is used as a base network architecture $\mathcal{F}$. Then, we obtain the learned spatial hierarchy $\mathcal{H}$ and synchronized architecture $\mathcal{F}_{\mathcal{H}}$ (same as $\mathcal{F}_{spatial}$; used to save space in result tables), and further train a spatial moderator $\mathcal{F}_M$ on top of each $\mathcal{F}_{\mathcal{H}}$. For training, we use the Adam optimizer with initial learning rate set to 0.01. All the model parameters in $\mathcal{F}$ and $\mathcal{F}_{\mathcal{H}}$ (regardless of branches) are trained with 600 epochs (the losses converged and remained stable in the final 100 epochs). All the models, when fine-tuned (e.g., for region $\mathcal{D}_B$ in California data), are allocated with an extra 600 epochs. For candidate methods with spatial moderator, only moderator weights are fine-tuned. The loss functions for classification and regression are

TABLE I: F1 scores: California land-cover classification using time-series of Sentinel-2 multi-spectral imagery

	Region $\mathcal{D}_A$ (20% training, 20% validation)									Region $\mathcal{D}_B$ (1% fine-tuning)								
	DNN (snapshot)			LSTM			LSTM+Attention			DNN (snapshot)			LSTM			LSTM+Attention		
	$\mathcal{F}$	$\mathcal{F}_{\mathcal{H}}$	$\mathcal{F}_M$	$\mathcal{F}$	$\mathcal{F}_{\mathcal{H}}$	$\mathcal{F}_M$	$\mathcal{F}$	$\mathcal{F}_{\mathcal{H}}$	$\mathcal{F}_M$	$\mathcal{F}$	$meta$	$\mathcal{F}_M$	$\mathcal{F}$	$meta$	$\mathcal{F}_M$	$\mathcal{F}$	$meta$	$\mathcal{F}_M$
Corn	.57	.62	**.65**	.73	.75	**.77**	.72	.74	**.77**	.38	.19	**.37**	.49	**.53**	**.53**	.52	.52	**.56**
Cotton	.00	**.79**	.75	.80	.82	**.83**	.79	.80	**.83**	.85	.77	**.88**	.90	.91	**.93**	.92	.91	**.93**
Sorghum	.06	**.47**	**.47**	.25	.51	**.63**	.00	.00	**.65**	.00	.00	**.25**	.12	.17	**.33**	.00	.13	**.43**
Wheat	.00	.00	**.30**	.16	.25	**.55**	.00	.00	**.59**	.00	.00	**.27**	.01	.56	**.60**	.00	.53	**.59**
Alfalfa	.48	.58	**.62**	.70	.73	**.74**	.72	.73	**.75**	.58	.10	**.62**	.71	.75	**.76**	.76	.75	**.77**
Grapes	.59	.63	**.71**	.67	.79	**.80**	.77	.79	**.82**	.03	.00	**.46**	.00	.66	**.75**	.69	.68	**.77**
Citrus	.00	.00	**.39**	.40	.40	**.53**	.06	.35	**.59**	.00	.00	.00	.28	.00	**.53**	.33	.33	**.64**
Almond	.54	.60	**.67**	.51	.73	**.77**	.73	.75	**.79**	.60	.37	**.68**	.71	.75	**.82**	.76	.77	**.82**
Walnut	.04	.48	**.51**	.59	.59	**.71**	.51	.57	**.72**	.00	.00	.00	.00	.00	**.21**	.00	.00	.00
Pistachio	.46	.60	**.68**	.71	.81	**.82**	.74	.80	**.85**	.70	.51	**.78**	.76	.79	**.87**	.78	.82	**.87**
Tomato	.00	.00	**.68**	.83	.85	**.87**	.83	.85	**.87**	.00	.00	**.35**	.55	.65	**.73**	.68	.66	**.72**
Garlic	.00	.00	**.64**	.00	.11	**.79**	.28	.68	**.80**	.00	.00	**.22**	.00	.00	.00	.00	.00	.00
Forest	.00	**.64**	.60	.00	.00	**.65**	.29	.36	**.66**	.00	.57	**.65**	.00	.69	**.74**	.61	.68	**.73**
Grass	.69	**.78**	.77	.75	.78	**.80**	.78	.79	**.81**	.69	.73	**.76**	.70	**.78**	**.78**	.74	.77	**.79**
Barren	.47	.51	**.55**	.57	.60	**.63**	.56	.59	**.64**	.53	.58	**.64**	.54	.69	**.71**	.57	.69	**.71**
Water	.56	.56	**.61**	.00	.08	**.66**	.63	.63	**.67**	.00	.00	**.51**	.00	.22	**.66**	.00	**.64**	.62
Urban	.57	.64	**.67**	.67	**.70**	**.70**	.66	.69	**.70**	.00	.00	**.08**	.02	.00	**.18**	.03	.00	**.19**
Mean_{uw}	.30	.46	**.60**	.49	.56	**.72**	.53	.60	**.74**	.26	.22	**.44**	.34	.48	**.60**	.43	.52	**.60**
Mean_{w}	.49	.59	**.67**	.63	.71	**.75**	.68	.71	**.77**	.53	.56	**.66**	.56	.70	**.74**	.63	.70	**.74**

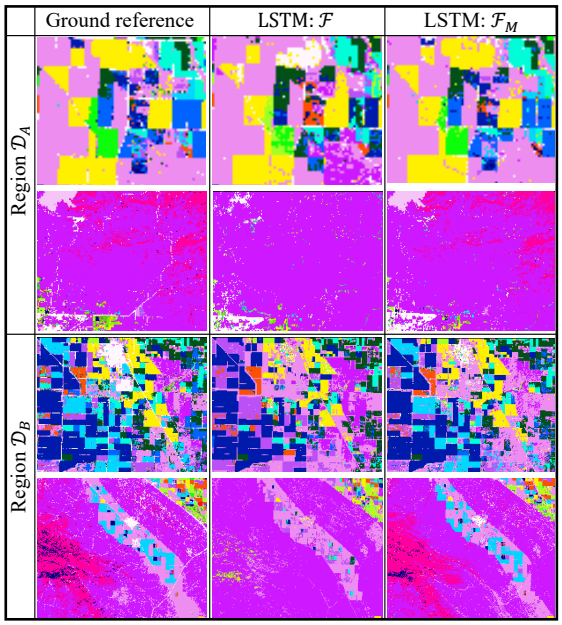

Fig. 4: Visualization of LSTM results at example locations.

cross-entropy and mean-squared errors, respectively. Code is available at: https://github.com/yqthanks/STAR

C. Candidate methods

For California land-cover classification, we have 12 candidate methods: (1) base $\mathcal{F}$, spatially transformed $\mathcal{F}_{\mathcal{H}}$ (no moderator), and $\mathcal{F}_M$ (with moderator), for the three models DNN, LSTM and LSTM+Attention; and (2) $\mathcal{F}$ integrated with the model-agnostic meta-learning (MAML; denoted as $meta$) for fast fine-tuning on region $\mathcal{D}_B$ with the available 1% of data ($\mathcal{D}_A$'s data are clustered into tasks), for the three models.

For Boston COVID-19 human mobility regression, we have 5 candidate methods for comparison, i.e., ridge regression, geographically-weighted regression (GWR), COVID-GAN (using code shared in [18]), DNN, and $\text{DNN}_{\mathcal{H}}$ (spatially transformed version). As time-series is not used to construct additional features in [18] and some features are aggregated to week-levels, we follow the same strategy and only use week IDs as features, which also allows a more direct comparison with COVID-GAN. In addition, as training and testing samples are from the same set of spatial locations (timestamps are different, Sec. IV-A), we directly use spatial transformation and skipped the moderator in this comparison.

D. Results

1) Land-cover classification: Table I shows the F1-scores of the 12 candidate methods for both spatial regions $\mathcal{D}_A$ (20% for training, and 20% for validation) and $\mathcal{D}_B$ (1% of data for fine-tuning). In addition to class-wise results, two means are shown at the bottom: (1) Mean_w is the standard weighted average over the classes based on the number of samples in each class; (2) Mean_{uw} is the unweighted (or direct) average over the classes, which helps reveal if a method's performance is balanced across classes. This is important in many applications including land-cover classification. For example, many relatively rare classes often have much higher values per acre. For the proposed $\mathcal{F}_{\mathcal{H}}$ and $\mathcal{F}_M$, the spatial hierarchy $\mathcal{H}$ and $\mathcal{F}_{\mathcal{H}}$ are learned with training data in region $\mathcal{D}_A$. Then, in region $\mathcal{D}_B$, the learned weights in $\mathcal{F}_{\mathcal{H}}$ are frozen and $\mathcal{F}_M$ only fine-tunes the moderator with the 1% samples. This helps evaluate if the heterogeneous spatial processes $\{\Phi\}$ learned in $\mathcal{D}_A$ can be generalized to the new region $\mathcal{D}_B$ with the moderator, which re-mixes the processes $\{\Phi\}$ based on characteristics of test data samples. For other architectures, i.e., the base $\mathcal{F}$ and $\mathcal{F}$+MAML ($meta$ in Table I), the network weights are fine-tuned. We also implemented SVANN [6] for testing. However, SVANN requires an input space-partitioning (Sec. I) which is unavailable in this case. We tried it with both an equal-quad and a k-means ($k = 64$) based partitioning, which did not improve over the base models due to data reduction from partitioning. Moreover, SVANN relies on fixed partitioning and cannot be applied outside the original spatial area. Thus, we skipped its results in Table I.

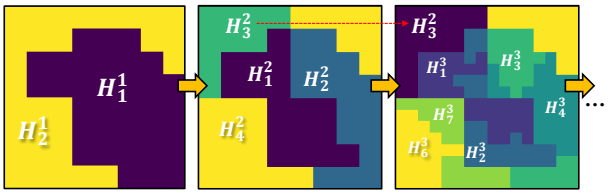

Fig. 5: Spatial hierarchy learned in region $\mathcal{D}_A$ (first 3 levels).

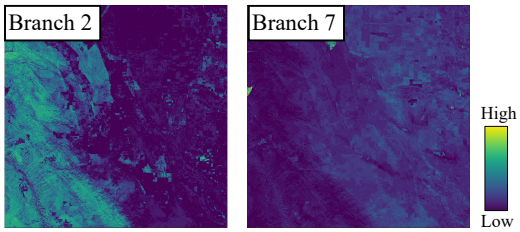

Fig. 6: Learned branch weights across space (across samples).

As we can see, the "spatialized" network architectures overall achieved the highest F1-scores for different types of base models in both regions. Moreover, according to the Mean$_{uw}$ results, the added awareness of spatial heterogeneity allows $\mathcal{F}_\mathcal{H}$ and $\mathcal{F}_M$ to achieve a more stable performance across different classes. For region $\mathcal{D}_A$, the three results $\mathcal{F}$, $\mathcal{F}_\mathcal{H}$ and $\mathcal{F}_M$ for each base model can be used for ablation analysis. The general trend is that the results of a base model $\mathcal{F}$ gradually improve with the addition of spatial transformation $\mathcal{F}_\mathcal{H}$, and the spatial moderator $\mathcal{F}_M$ (e.g., Mean$_w$ increases from 0.49 to 0.59, and finally to 0.67 for DNN). To visualize the improvements, Fig. 4 shows local maps of land-cover classifications for LSTM's base $\mathcal{F}$ and moderator $\mathcal{F}_M$ in four sample areas, with two for each region.

In addition, Fig. 5 shows the hierarchical process of space-partitioning with DL-MSS (Sec. III-C2) for the first 3 levels. In the first level (largest scale), for example, $\mathcal{H}_1^1$ is a mix of urban and suburban areas, whereas $\mathcal{H}_2^1$ contains more rural and mountainous areas. Note that some partitions (e.g., $\mathcal{H}_3^2$) are not further split, as determined by significance testing. Also, a partition is allowed to contain multiple disconnected areas as long as they satisfy the minimum footprint size enforced for contiguity (Sec. III-C3). Nonetheless, we can see the automatically captured footprints are in general spatially contiguous as a result of spatial auto-correlation. Finally, Fig. 6 visualizes the weights predicted by the moderator for two example network branches in $\mathcal{F}_\mathcal{H}$ for DNN (paths from input to output layers; Fig. 3) for all locations in region $\mathcal{D}_B$. For each branch, the weight is averaged over all classes in the predicted $\mathbf{W}$ at each location. As we can see, in the new region $\mathcal{D}_B$, branch-2 is given higher weights for the left-side of the region, which is a mountainous area, whereas the weights for branch-7 shows the opposite spatial pattern.

2) COVID-19 mobility regression: Table II shows the results of the five candidate methods. We include two sets of measures: (1) Mean absolute error (MAE) and root mean-squared-error (RMSE); and (2) The total mobility aggregated over the entire region, which is a useful indicator for policy-making at the global level; difference (denoted by "Diff.") in Table II refers to the deviation from the ground truth aggregation. Fig. 7 shows maps of the ground truth, COVID-GAN, DNN and DNN$_\mathcal{H}$. Several potential causes of spatial heterogeneity here include different mobility patterns in the more populous downtown area versus the suburban regions, and several "hotspot" areas of POI visits that are a bit abnormal compared to the rest.

TABLE II: COVID-19 human mobility projection

	Ridge	GWR	COVID-GAN	DNN	DNN$_\mathcal{H}$
MAE	220	160	178	159	**139**
RMSE	440	388	388	405	**341**
Total	335,225	246,615	402,471	213,362	**440,627**
Diff.	-86,998	-175,608	-19,752	-208,861	**18,404**

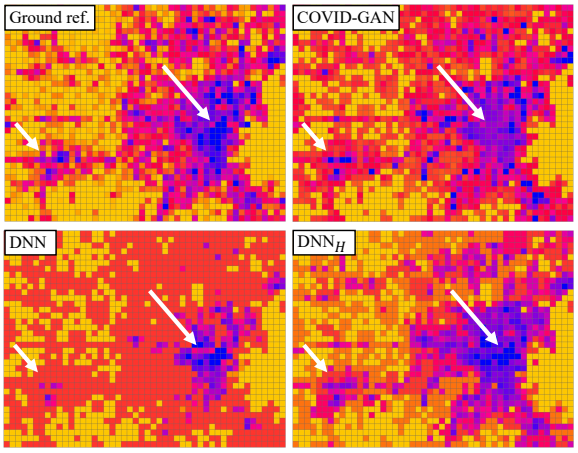

Fig. 7: Visualization of human mobility maps (blue: high).

As we can see, overall DNN$_\mathcal{H}$ achieved better results for both sets of measures. One interesting observation is that DNN (the base model $\mathcal{F}$ used for DNN$_\mathcal{H}$), while obtained better MAE than ridge regression and COVID-GAN, substantially underestimates the total mobility, which could be a result of incorrect predictions on several mobility hotspots, whose patterns do not follow the global pattern. Similarly, GWR shows a similar trend. Although GWR performs spatially-localized regression, it can only handle simple linear relationships using input variables and apply the same spatial neighborhood for all locations, which cannot well capture non-stationary mobility hotspots and variation in the data. Moreover, the data-reduction problem caused by local-regression in GWR frequently causes existing standard libraries to run into ill-conditioned problems if small band-widths are used. Finally, DNN$_\mathcal{H}$ automatically identified three heterogeneous partitions (other splits are statistically insignificant) and branched out downtown, suburban and several mobility hotspots (our method allows large footprints at multiple locations to be in one node), greatly improving the performance on both types of measures.

V. Other Related Work

Existing methods for handling heterogeneity can be generally divided into two categories. The first class aims to transfer parameters learned from one source to another, such as domain adaptation [19], [20] and meta-learning (e.g., MAML) [21]–[23]. However, these methods mainly focus on the learning of robust features and fast adaptation, and may yield degraded

performance when spatial regions have large discrepancy. Moreover, they require a pre-defined space-partitioning of heterogeneous processes. It is worth-mentioning though that these methods and the proposed STAR framework are complementary and can be integrated for further enhancements. The second direction is based on explicit data partitioning. For example, researchers have separately trained individual local models for different data clusters [24], [25] and have shown improved performance against a single global model. However, the clustering only uses input features that are not sufficient to capture underlying heterogeneous processes. Similarly, local training has been used for manually-defined spatial regions, e.g., ANN [6] and RNN [26]. Furthermore, all these methods can significantly reduce the training data available for local models, making it difficult to train complex models. Spatial-Net also uses the hierarchical multi-task representation for parameter sharing [10], but cannot handle irregular partitionings (i.e., spatial footprints with arbitrary shapes) or be generalized to new regions; it does guarantee each partition's spatial contiguity. Finally, mixture process mining [27] can find regions where data are generated by homogeneous-mixture processes $\{\Phi\}$ but cannot handle deep learning inputs where $\{\Phi : \mathbf{X} \rightarrow \mathbf{y}\}$ are often unknown and cannot be directly defined by statistical models (e.g., Poisson).

VI. Conclusions and Future Work

We proposed a model-agnostic spatial transformation and moderation framework for spatial data and problems. The framework can: (1) simultaneously learn arbitrarily-shaped space-partitionings of heterogeneous processes and a "spatialized" network architecture; and (2) generalize learned spatial structures to new regions. We demonstrated the statistically-guided approach on different types of tasks and networks (e.g., snapshot-based, time-series). Experiments on real world datasets showed that the framework can substantially improve the performance of base networks on spatial problems.

In future work, we will explore the use of the framework on other types of network architectures such as GAN, CNN and GCN, and traditional machine learning methods. Furthermore, we plan to investigate specific characteristics of each type of network architecture in the context of spatial heterogeneity and identify dedicated customizations of the current framework. Finally, we will explore generalizations with other types of space-partitioning schemes, statistical formulations, etc.

Acknowledgment

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; Erhu He and Xiaowei Jia are 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 funded by US-DOT award 69A3551747131.

References

[1] "Group on earth observations global agricultural monitoring initiative," 2021, https://earthobservations.org/geoglam.php.
[2] M. U. Kraemer *et al.*, "The effect of human mobility and control measures on the covid-19 epidemic in china," *Science*, vol. 368, no. 6490, pp. 493–497, 2020.
[3] G. Atluri, A. Karpatne, and V. Kumar, "Spatio-temporal data mining: A survey of problems and methods," *ACM Computing Surveys (CSUR)*, vol. 51, no. 4, pp. 1–41, 2018.
[4] S. Shekhar, S. K. Feiner, and W. G. Aref, "Spatial computing," *Communications of the ACM*, vol. 59, no. 1, pp. 72–81, 2015.
[5] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, pp. 1097–1105, 2012.
[6] J. Gupta *et al.*, "Towards spatial variability aware deep neural networks (svann): A summary of results," in *ACM SIGKDD workshop on deep learning for spatiotemporal data, app. & sys.*, 2020.
[7] Z. Jiang *et al.*, "Spatial ensemble learning for heterogeneous geographic data with class ambiguity," *ACM Trans. on Intelligent Sys. and Tech. (TIST)*, vol. 10, no. 4, 2019.
[8] C. Brunsdon, A. S. Fotheringham, and M. Charlton, "Some notes on parametric significance tests for geographically weighted regression," *Journal of regional science*, vol. 39, no. 3, pp. 497–524, 1999.
[9] A. Kratsios and I. Bilokopytov, "Non-euclidean universal approximation," *NeurIPS*, vol. 33, 2020.
[10] Y. Xie, X. Jia, H. Bao, X. Zhou, J. Yu, R. Ghosh, and P. Ravirathinam, "A self-adaptive and model-agnostic deep learning framework for spatially heterogeneous datasets," in *ACM SIGSPATIAL*, 2021.
[11] M. Kulldorff *et al.*, "Multivariate scan statistics for disease surveillance," *Statistics in medicine*, vol. 26, no. 8, pp. 1824–1833, 2007.
[12] D. B. Neill *et al.*, "Fast subset scan for multivariate event detection," *Statistics in medicine*, vol. 32, no. 13, pp. 2185–2208, 2013.
[13] Y. Xie, S. Shekhar, and Y. Li, "Statistically-robust clustering techniques for mapping spatial hotspots: A survey," *ACM Computing Surveys (CSUR)*, 2021.
[14] D. B. Neill, "Fast subset scan for spatial pattern detection," *Journal of the Royal Statistical Society*, vol. 74, no. 2, pp. 337–360, 2012.
[15] M. Kulldorff, L. Huang, and K. Konty, "A scan statistic for continuous data based on the normal probability model," *International journal of health geographics*, vol. 8, no. 1, pp. 1–9, 2009.
[16] X. Jia, S. Li, A. Khandelwal, G. Nayak, A. Karpatne, and V. Kumar, "Spatial context-aware networks for mining temporal discriminative period in land cover detection," in *SDM*. SIAM, 2019, pp. 513–521.
[17] "USDA cropland data layer," 2021, https://www.nass.usda.gov/Research_and_Science/Cropland/SARS1a.php.
[18] H. Bao *et al.*, "Covid-gan: Estimating human mobility responses to covid-19 pandemic through spatio-temporal conditional generative adversarial networks," in *ACM SIGSPATIAL*, 2020, pp. 273–282.
[19] Y. Zhang, P. David, and B. Gong, "Curriculum domain adaptation for semantic segmentation of urban scenes," in *ICCV*, 2017, pp. 2020–2030.
[20] X. Jia *et al.*, "Classifying heterogeneous sequential data by cyclic domain adaptation: An application in land cover detection," in *SDM*. SIAM, 2019.
[21] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *ICML*, 2017.
[22] M. Rußwurm *et al.*, "Meta-learning for few-shot land cover classification," in *CVPR Workshops*, 2020, pp. 200–201.
[23] H. Yao *et al.*, "Learning from multiple cities: A meta-learning approach for spatial-temporal prediction," in *WWW*, 2019, pp. 2181–2191.
[24] A. Karpatne, A. Khandelwal, S. Boriah, and V. Kumar, "Predictive learning in the presence of heterogeneity and limited training data," in *SDM*. SIAM, 2014, pp. 253–261.
[25] Y. Tarabalka, J. A. Benediktsson, and J. Chanussot, "Spectral–spatial classification of hyperspectral imagery based on partitional clustering techniques," *IEEE Trans. on Geoscience and Remote Sensing*, vol. 47, no. 8, pp. 2973–2987, 2009.
[26] Z. Yuan, X. Zhou, and T. Yang, "Hetero-convlstm: A deep learning approach to traffic accident prediction on heterogeneous spatio-temporal data," in *ACM SIGKDD*, 2018, pp. 984–992.
[27] Y. Xie, H. Bao, Y. Li, and S. Shekhar, "Discovering spatial mixture patterns of interest," in *Proceedings of the 28th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, 2020, pp. 608–617.