Spatial Classification with Limited Observations Based on Physics-Aware Structural Constraint

Arpan Man Sainju,[1] **Wenchong He,**[1] **Zhe Jiang,**[1*] **Da Yan**[2]
[1]Department of Computer Science, The University of Alabama
[2]Department of Computer Science, The University of Alabama at Birmingham
{asainju, whe11}@crimson.ua.edu, zjiang@cs.ua.edu, yanda@uab.edu

Abstract

Spatial classification with limited feature observations has been a challenging problem in machine learning. The problem exists in applications where only a subset of sensors are deployed at certain regions or partial responses are collected in field surveys. Existing research mostly focuses on addressing incomplete or missing data, e.g., data cleaning and imputation, classification models that allow for missing feature values, or modeling missing features as hidden variables and applying the EM algorithm. These methods, however, assume that incomplete feature observations only happen on a small subset of samples, and thus cannot solve problems where the vast majority of samples have missing feature observations. To address this issue, we propose a new approach that incorporates physics-aware structural constraints into the model representation. Our approach assumes that a spatial contextual feature is observed for all sample locations and establishes spatial structural constraint from the spatial contextual feature map. We design efficient algorithms for model parameter learning and class inference. Evaluations on real-world hydrological applications show that our approach significantly outperforms several baseline methods in classification accuracy, and the proposed solution is computationally efficient on a large data volume.

Introduction

Given a spatial raster framework with explanatory feature layers, a spatial contextual layer (e.g., a potential field), as well as a set of training samples with class labels outside the framework, the spatial classification problem (Jiang 2019) aims to learn a model that can predict a class layer. We particularly focus on spatial classification with limited feature observations, i.e., only limited pixel locations in the raster framework have explanatory feature data available. For example, in earth imagery classification, the explanatory feature layers are spectral bands of earth imagery pixels; the spatial contextual layer can be elevation, and the target class layer consists of pixel classes (e.g., flood or dry). In the example, it often happens that the elevation values are available for all pixels in the framework, but only limited pixel locations have spectral data (e.g., a drone or aerial plane could not cover the entire region due to limited time during a flood disaster).

The problem is important in many societal applications such as flood extent mapping for disaster response and national water forecasting. Flood extent mapping plays a crucial role in addressing grand societal challenges such as disaster management, national water forecasting, as well as energy and food security (Jiang and Shekhar 2017; Xie, Jiang, and Sainju 2018; Jiang, Xie, and Sainju 2019). For example, during Hurricane Harvey floods in 2017, first responders needed to know where floodwater was in order to plan rescue efforts. In national water forecasting, detailed flood extent maps can be used to calibrate and validate the NOAA National Water Model (National Oceanic and Atmospheric Administration 2018). In current practice, flood extent maps are mostly generated by flood forecasting models, whose accuracy is often unsatisfactory in a high spatial resolution (Cline 2009; Merwade et al. 2008). Other ways to generate flood maps involve sending a field crew on the ground to mark down the floodwater extent on a map, but the process is both expensive and time-consuming. A promising alternative is to utilize observation data from groundwater sensors and remote sensors on aerial planes or drones. However, sensor observations may have limited spatial coverage due to only a subset of sensors being deployed at certain regions, making the problem in spatial classification with limited features. For example, a drone during a flood disaster can only collect spectral images in limited areas. Note that though we use flood mapping application as an example, the problem can potentially be applied to other broad applications such as water quality monitoring (Yang and Jin 2010) along river networks.

The problem poses several unique challenges that are not well addressed by traditional classification techniques. First, there are limited feature observations on samples in the raster framework due to only a subset of sensors being deployed in certain regions. In other words, only a subset of samples has complete explanatory feature values, making it hard to predict classes for all samples. Second, among the sample pixels with complete explanatory feature values, their feature values may contain rich noise and obstacles.

[*] Contact author: zjiang@cs.ua.edu

For example, high-resolution earth imagery often has noise: clouds and shadows. Third, the explanatory features of image pixels can be insufficient to distinguish classes (also called class confusion) due to heterogeneity. For instance, pixels of tree canopies overlaying flood water have the same spectral features with trees in dry areas, yet their classes are different. Finally, the number of pixel locations can be very large for high-resolution data (e.g., hundreds of millions of locations in one city), requiring scalable algorithms.

Over the years, various techniques have been developed to address missing feature observations (or incomplete data) in classification (García-Laencina, Sancho-Gómez, and Figueiras-Vidal 2010). Existing methods can be categorized into data cleaning or imputation, utilizing classification models that allow for missing feature values, and modeling missing features as hidden variables with the EM (Expectation-Maximization) algorithm. Data cleaning will remove samples that miss significant feature values. Data imputation focuses on filling in missing values either by statistical methods (Little and Rubin 2019) (e.g., mean feature values from observed samples) or by predicted feature values using a regression model based on observed samples (Schafer 1997; Batista, Monard, and others 2002; Yoon and Lee 1999; Bengio and Gingras 1996; Rubin 2004). A different approach focuses on classification models and algorithms that allow for missing feature values in learning and prediction without data imputation. For example, a decision tree model allows for samples with missing features in learning and classification (Quinlan 2014; 1989; Webb 1998). During training, the weight is one for an observed feature value but is lower than one if the feature value is missing (the weight for different possible values are based on their frequencies in completely observed samples). During classification, a decision tree can explore all possible tree traversal paths for samples with missing features and select the final class prediction with the highest probability. Similarly, there are some other models or algorithms that can be extended to allow for missing feature values, such as neural network ensembles (Jiang, Chen, and Yuan 2005), support vector machine (Chechik et al. 2007; Smola, Vishwanathan, and Hofmann 2005; Pelckmans et al. 2005), etc. The last category is to model missing feature values as hidden variables and use the EM (Expectation-Maximization) algorithm for effective learning and inference (McLachlan and Krishnan 2007; Ghahramani and Jordan 1994b; 1995; Williams et al. 2007). Specifically, the joint distribution of all samples' features (both observed and missing features) can be represented by a mixture model with fixed by yet unknown parameters. In the EM algorithm, we can use initialized parameters and observed features to estimate the posterior distribution of hidden variables (missing features), and then further update the parameters for the next iteration. However, all these existing methods above assume that incomplete feature observations only happen on a small subset of samples, and thus cannot adequately be applied to our problem where the vast majority of samples have missing features (limited feature observations).

To fill the gap, we propose a new approach that incorporates physics-aware structural constraints into model representation. Our approach assumes that a spatial contextual feature is observed for all sample locations, and establishes spatial structural constraints from the spatial contextual feature map. We design efficient algorithms for model parameter learning and class inference and conduct experimental evaluations to validate the effectiveness and efficiency of the proposed approach against existing works. More specifically, we make the following contributions:

- We propose a new approach that utilizes physics-aware spatial structural constraints to handle limited feature observations in spatial classification.
- We design efficient algorithms for model parameter learning and class inference.
- We evaluated the proposed model on two real-world hydrological datasets. Results show that our approach significantly outperforms several baseline methods in classification accuracy, and the proposed solution is computationally efficient on a large data volume.

Problem Statement

Preliminaries

Here we define several basic concepts to define the problem formally.

A *spatial raster framework* is a tessellation of a two-dimensional plane into a regular grid of N cells. Spatial neighborhood relationship exists between cells based on cell adjacency. The framework can contain m non-spatial explanatory feature layers (e.g., spectral bands in earth imagery), one potential field layer (e.g., elevation), and one class layer (e.g., *flood*, *dry*).

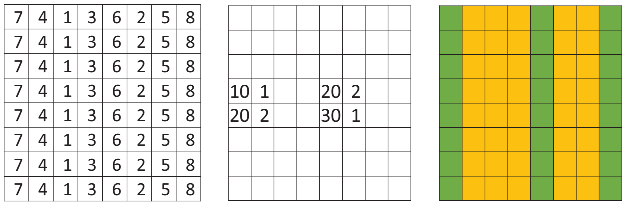
Each cell in a raster framework is a *spatial data sample*, noted as $\mathbf{s_n} = (\mathbf{x}_n, \phi_n, y_n)$, where $n \in \mathbb{N}, 1 \leq n \leq N$, $\mathbf{x}_n \in \mathbb{R}^{m\times 1}$ is a vector of m non-spatial explanatory feature values with each element corresponding to one feature layer, $\phi_n \in \mathbb{R}$ is a cell's potential field value, and $y_n \in \{0, 1\}$ is a binary class label.

A raster framework with all samples is noted as $\mathcal{F} = \{\mathbf{s_n} | n \in \mathbb{N}, 1 \leq n \leq N\}$, non-spatial explanatory feature matrix of all samples are noted as $\mathbf{X} = [\mathbf{x}_1, ..., \mathbf{x}_N]^T$, the potential field vector is noted as $\mathbf{\Phi} = [\phi_1, ..., \phi_N]^T$, and the class vector is noted as $\mathbf{Y} = [y_1, ..., y_N]^T$.

In a raster framework, it may happen that only a limited number of samples have non-spatial explanatory features being observed. We define $\mathcal{O}$ as the corresponding set of these sample indices. Samples with complete explanatory features are noted as $\{\mathbf{x}_n | n \in \mathcal{O}\}$. The corresponding sample feature matrix is noted as $\mathbf{X_o}$.

Problem Definition

Given a raster framework with the explanatory features of a limited number of samples in the framework $\mathbf{X}_o$, the potential field layer of all samples in the framework $\mathbf{\Phi} = [\phi_1, ..., \phi_N]^T$, and a set of training samples with class labels outside the framework, the spatial classification problem aims to learn a classifier f to predict the class layer $\mathbf{Y} = f(\mathbf{X}_o, \mathbf{\Phi})$. For example, in earth imagery classification, the explanatory feature layers are spectral bands of earth imagery pixels; the spatial contextual layer can be elevation, and the target class layer consists of pixel classes (e.g., flood or dry). In the example, it often happens that the elevation values are available for all pixels in the framework (elevation values do not change over time and thus can be collected at once), but only limited pixel locations have spectral data (e.g., a drone or aerial plane could not cover the entire region due to limited time during a flood disaster). Figure 1 shows a toy example of a raster framework that consists of sixty-four samples with a one-dimensional explanatory feature and a full potential field layer. There are only eight samples with observed explanatory features (four non-empty cells in Figure 1(b)). The goal is to learn a model that can predict the class layer in Figure 1(c).



7	4	1	3	6	2	5	8
7	4	1	3	6	2	5	8
7	4	1	3	6	2	5	8
7	4	1	3	6	2	5	8
7	4	1	3	6	2	5	8
7	4	1	3	6	2	5	8
7	4	1	3	6	2	5	8
7	4	1	3	6	2	5	8

(a) Spatial potential field (elevation, $\mathbf{\Phi}$)

10	1			20	2		
20	2			30	1		

(b) Non-spatial feature ($\mathbf{X}_o$)

🟩	🟧	🟧	🟧	🟩	🟧	🟧	🟩
🟩	🟧	🟧	🟧	🟩	🟧	🟧	🟩
🟩	🟧	🟧	🟧	🟩	🟧	🟧	🟩
🟩	🟧	🟧	🟧	🟩	🟧	🟧	🟩
🟩	🟧	🟧	🟧	🟩	🟧	🟧	🟩
🟩	🟧	🟧	🟧	🟩	🟧	🟧	🟩
🟩	🟧	🟧	🟧	🟩	🟧	🟧	🟩
🟩	🟧	🟧	🟧	🟩	🟧	🟧	🟩

(c) True classes of all samples ($\mathbf{Y}$)

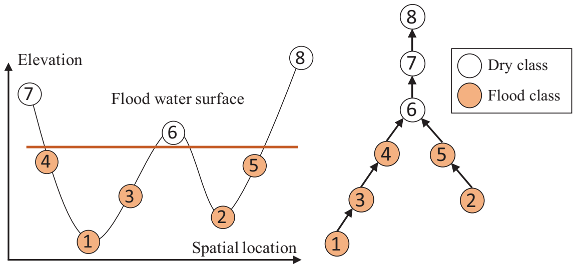
Figure 1: An illustration problem example (green color for the dry class, orange color for the flood class in (c))

Approach

In this section, we introduce our proposed approach. We start with physics-aware structural constraints and then introduce our probabilistic model and its learning and inference. We will introduce our approach in the context of flood mapping application example, but the proposed method can be potentially generalized to other applications such as material science and biochemistry.

Physics-Aware Structural Constraint

The main idea of our proposed approach is to establish a spatial dependency structure of sample class labels based on the physical constraint from the spatial potential field layers (e.g., water flow directions based on elevation). Figure 2 is an illustrative example. Figure 2(a) is the elevation values of eight pixels in one dimensional space (e.g., pixels on a row in Figure 1). Due to gravity, water flows from high locations to nearby lower locations. If location 4 is flooded, locations 1 and 3 must also be flooded. Such a dependency structure can be established based on the topology of the potential field surface (e.g., elevation). Figure 2(b) shows a directed tree structure that captures the flow dependency structure. If any node is *flood*, then all sub-tree nodes must be *flood* due to gravity. The structure is also called *split tree* in topology (Carr, Snoeyink, and Axen 2003; Edelsbrunner and Harer 2010), where a node represents a vertex on a mesh surface (spatial potential field), and the edges show the topological relationships between vertices. We can efficiently construct the tree structure from a potential field map following the topological order of pixels based on the union-find operator (the time complexity is $O(N\log N)$) (Carr, Snoeyink, and Axen 2003). We do not introduce details due to space limitations. It is worth to note that though our illustrative example in Figure 2 is in one-dimensional space for simplicity, the structure can be applicable to two-dimensional space in general cases (Jiang and Sainju 2018). For example, we can create a single tree structure for the entire elevation map in Figure 1(a).



(a) Eight consecutive sample locations in one dimensional space

(b) Partial order constraint in a reverse tree

Figure 2: Illustration of partial order class dependency

Model Probabilistic Formulation

Now we introduce our approach that integrates physics-aware structural constraint into the probabilistic model formulation to handle limited samples with observed features. Figure 3 illustrates the overall model structure. It consists of two layers: a hidden class layer with unknown sample classes (y_n) and an observation layer with limited sample feature vectors ($\mathbf{x}_n$). Each node corresponds to a spatial data sample (raster cell). Edge directions show a probabilistic conditional dependence structure based on physical constraint.

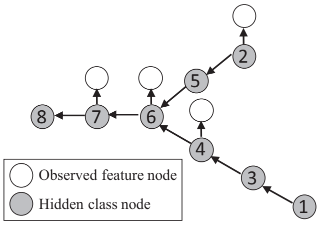


Figure 3: Illustration of model structure

The joint distribution of all observed samples' features and classes can be expressed as Equation 1, where $\mathcal{P}_n$ is the set of parent samples of the nth sample in the dependency tree, and $y_{k\in\mathcal{P}_n} \equiv \{y_k | k \in \mathcal{P}_n\}$ is the set of class nodes corresponding to parents of the nth sample. For a leaf node n, $\mathcal{P}_n = \emptyset$, and $P(y_n|y_{k\in\mathcal{P}_n}) = P(y_n)$.

$$P(X_o,\mathbf{Y}) = P(X_o|\mathbf{Y})P(\mathbf{Y}) = \prod_{n\in\mathcal{O}} P(\mathbf{x}_n|y_n) \prod_{n=1}^{N} P(y_n|y_{k\in\mathcal{P}_n}) \tag{1}$$

The conditional probability of sample feature vector given its class can be assumed i.i.d. Gaussian for simplicity, as shown in Equation 2, where $\boldsymbol{\mu}_{y_n}$ and $\boldsymbol{\Sigma}_{y_n}$ are the mean and covariance matrix of feature vector $\mathbf{x}_n$ for class y_n ($y_n = 0, 1$). It is worth noting that $P(\mathbf{x}_n|y_n)$ could be more general than i.i.d. Gaussian.

$$P(\mathbf{x}_n|y_n) \sim \mathcal{N}(\boldsymbol{\mu}_{y_n}, \boldsymbol{\Sigma}_{y_n}) \tag{2}$$

Class transitional probability follows the partial order constraint. For example, due to gravity, if any parent's class is *dry*, the child's class must be *dry*; if all parents' classes are *flood*, then the child has a high probability of being *flood*. Consider *flood* as the positive class (class value 1) and *dry* as the negative class (class value 0), the transitional probability is actually conditioned on the product of parent classes $y_{\mathcal{P}_n} \equiv \prod_{k\in\mathcal{P}_n} y_k$. Table 1 shows two parameters for class transitional probability (ρ) and class prior probability (π).

Table 1: Class transition probability and prior probability

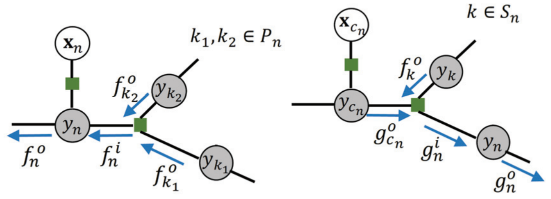
$P(y_n\|y_{\mathcal{P}_n})$	$y_{\mathcal{P}_n}=0$	$y_{\mathcal{P}_n}=1$
$y_n = 0$	1	$1-\rho$
$y_n = 1$	0	ρ

	$P(y_n)$
$y_n = 0$	$1-\pi$
$y_n = 1$	π

Model Parameter Learning and Class Inference

Our model parameters include the mean and covariance matrix of sample features in each class, the prior probability of leaf node classes, and class transition probability for non-leaf nodes. We denote the entire set of parameters as $\boldsymbol{\Theta} = \{\rho, \pi, \boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c | c = 0, 1\}$. Learning the set of parameters poses two major challenges: first, Equation 1 both unknown parameters and hidden class variables $\mathbf{Y} = [y_1, ..., y_N]^T$ that are non-i.i.d.; second, the number of samples (N) can be huge (e.g., millions of pixels).

To address these challenges, we propose to use the expectation-maximization (EM) algorithm together with message (belief) propagation. The main idea of the EM algorithm is to first initialize a parameter setting, and compute the posterior expectation of log-likelihood (Equation 1) on hidden class variables (E-step). The posterior expectation is a function of unknown parameters, and thus can be maximized by updating the parameter values (M-step). The two steps can be done iteratively until the parameter values converge. One remaining issue is the calculation of posterior expectation of log-likelihood on hidden class variables. The requires to compute the marginal posterior distribution of $P(y_n, y_{k\in\mathcal{P}_n}|\mathbf{O}, \boldsymbol{\Theta}_\mathbf{0})$ and $P(y_n)$ for each node n. This is very challenging due to its high dimensionality of $\mathbf{Y}$. To address this challenge, we use message propagation. Message propagation is based on the sum and product algorithm (Kschischang, Frey, and Loeliger 2001; Ronen, Rohlicek, and Ostendorf 1995). Propagation of message along nodes in a graph (or tree) is equivalent to marginalizing out node variables in the overall joint distribution in Equation 1. Due to the space limit, we only show the major steps in the following discussion. More details of the theoretical proof are in the appendix.



(a) From leaves to root (b) From root to leaves

Figure 4: Illustration of message propagation in split tree

The message passing process is illustrated in Figure 4. Specifically, forward message propagation from leaves to root is based on Equation 3 and Equation 4, where $f^i_n(y_n)$ and $f^o_n(y_n)$ are the incoming message into and outgoing message from a hidden class node y_n respectively. Backward message propagation from root to leaves also follows a recursive process, as shown in Equation 5 and Equation 6, where $g^i_n(y_n)$ and $g^o_n(y_n)$ are the incoming and outgoing messages for class node y_n respectively. For those samples without feature vector x_n, the outgoing forward and backward messages are the same with incoming forward and backward messages respectively, because we do not consider the feature probability for those samples.

$$f^i_n(y_n) = \begin{cases} P(y_n) & \text{if } y_n \text{ is leaf} \\ \sum\limits_{y_{k\in\mathcal{P}_n}} P(y_n|y_{k\in\mathcal{P}_n}) \prod\limits_{k\in\mathcal{P}_n} f^o_k(y_k) & \text{otherwise} \end{cases} \tag{3}$$

$$f^o_n(y_n) = \begin{cases} f^i_n(y_n)P(\mathbf{x}_n|y_n) & \text{if } n \in \mathcal{O} \\ f^i_n(y_n) & \text{otherwise} \end{cases} \tag{4}$$

$$g^i_n(y_n) = \begin{cases} 1 & \text{if } y_n \text{ is root} \\ \sum\limits_{y_{c_n}, y_{k\in\mathcal{S}_n}} g^o_{c_n} P(y_{c_n}|y_n) \prod\limits_{k\in\mathcal{S}_n} f^o_k(y_k) & \text{otherwise} \end{cases} \tag{5}$$

$$g^o_n(y_n) = \begin{cases} g^i_n(y_n)P(\mathbf{x}_n|y_n) & \text{if } n \in \mathcal{O} \\ g^i_n(y_n) & \text{otherwise} \end{cases} \tag{6}$$

After both forward and backward message propagation, we can compute the marginal posterior distribution of hidden class variables based on the equations below, where P' is unnormalized marginal distribution.

$$P'(y_n|\mathbf{X}_o, \boldsymbol{\Theta}_\mathbf{0}) = \begin{cases} f^i_n(y_n)g^i_n(y_n)P(\mathbf{x}_n|y_n) & \text{if } n \in \mathcal{O} \\ f^i_n(y_n)g^i_n(y_n) & \text{otherwise} \end{cases} \tag{7}$$

$$P'(y_n, y_{k\in\mathcal{P}_n}|\mathbf{X}_o, \boldsymbol{\Theta}_\mathbf{0}) = \prod_{k\in\mathcal{P}_n} f^o_k(y_k)g^o_n(y_n) \tag{8}$$

$$P(y_n|\mathbf{X}_o, \boldsymbol{\Theta}_\mathbf{0}) \leftarrow \frac{P'(y_n|\mathbf{X}_o, \boldsymbol{\Theta}_\mathbf{0})}{\sum\limits_{y_n} P'(y_n|\mathbf{X}_o, \boldsymbol{\Theta}_\mathbf{0})} \tag{9}$$

$$P(y_n, y_{k\in\mathcal{P}_n}|\mathbf{X}_o, \mathbf{\Theta_0}) \leftarrow \frac{P'(y_n, y_{k\in\mathcal{P}_n}|\mathbf{X}_o, \mathbf{\Theta_0})}{\sum\limits_{y_n, y_{k\in\mathcal{P}_n}} P'(y_n, y_{k\in\mathcal{P}_n}|\mathbf{X}_o, \mathbf{\Theta_0})} \tag{10}$$

After the computation of marginal posterior distribution, we can update model parameters by maximizing the posterior expectation of log-likelihood (the maximization or M step in EM), as shown by equations below.

$$\rho = \frac{\sum\limits_{n|\mathcal{P}_n\neq\emptyset} \sum\limits_{y_n} \sum\limits_{y_{\mathcal{P}_n}} y_{\mathcal{P}_n}(1-y_n)P(y_n, y_{\mathcal{P}_n}|\mathbf{X}, \mathbf{\Theta_0})}{\sum\limits_{n|\mathcal{P}_n\neq\emptyset} \sum\limits_{y_n} \sum\limits_{y_{\mathcal{P}_n}} y_{\mathcal{P}_n} P(y_n, y_{\mathcal{P}_n}|\mathbf{X}, \mathbf{\Theta_0})} \tag{11}$$

$$\pi = \frac{\sum\limits_{n|\mathcal{P}_n=\emptyset} \sum\limits_{y_n} y_n P(y_n|\mathbf{X}, \mathbf{\Theta_0})}{\sum\limits_{n|\mathcal{P}_n=\emptyset} \sum\limits_{y_n} P(y_n|\mathbf{X}, \mathbf{\Theta_0})} \tag{12}$$

$$\mu_c = \frac{\sum\limits_{n\in\mathcal{O}} \mathbf{x}_n P(y_n = c|\mathbf{X}, \mathbf{\Theta_0})}{\sum\limits_{n\in\mathcal{O}} P(y_n = c|\mathbf{X}, \mathbf{\Theta_0})}, c = 0, 1 \tag{13}$$

$$\Sigma_c = \frac{\sum\limits_{n\in\mathcal{O}} (\mathbf{x}_n - \boldsymbol{\mu}_c)(\mathbf{x}_n - \boldsymbol{\mu}_c)^T P(y_n = c|\mathbf{X}, \mathbf{\Theta_0})}{\sum\limits_{n\in\mathcal{O}} P(y_n = c|\mathbf{X}, \mathbf{\Theta_0})}, c = 0, 1 \tag{14}$$

After learning model parameters, we can infer hidden class variables by maximizing the overall probability. A naive approach that enumerates all combinations of class assignments is infeasible due to the exponential cost. We use a dynamic programming-based method called *max-sum* (Rabiner 1989). The process is similar to the sum and product algorithm above. The main difference is that instead of using sum operation, we need to use max operation in message propagation, and also memorize the optimal variable values. We omit the details due to space limit.

How the Model Address Limited Observations?

The main intuition behind how our model handles limited observations is that the model can capture physical constraints between sample classes. The spatial structural constraints are derived from the potential field layer that is fully observed on the entire raster framework, regardless of whether non-spatial features are available or not. The topological structure in a split tree is consistent with the physical law of water flow directions on a topographic surface based on gravity. In this sense, even though many samples in the raster framework do not have non-spatial explanatory features observed, we can still infer their classes based on information from the pixels in the upstream or downstream locations.

Another potential question is how our model can effectively learn parameters given very limited observations. This question can be answered from the perspective of how model learning works. The major task of model learning is to effectively update parameters of $P(\mathbf{x}_n|y_n)$ for observed pixels in the test region, so that we can infer the posterior class probabilities on these pixels and further infer hidden classes on other pixels. As long as the training samples could give the model a reasonable initial estimate of posterior class probabilities on the observed pixels (e.g., truly dry pixels having a higher probability of being dry), the update of parameters should be effective. This is because that parameter updates are largely weighted average of the sample mean and covariance matrices on fully observed pixels. The weights are the posterior class probability of observed samples.

Experimental Evaluation

Experiment Setup

In this section, we compared our proposed approach with baseline methods in related works on real-world datasets. Evaluation candidate methods are listed below. Note that we did not include data imputation methods (e.g., filling in mean feature values) due to its low capability of handling very limited observations. Unless specified otherwise, we used default parameters in open source tools for baseline methods. Experiments were conducted on a Dell workstation with Intel(R) Xeon(R) CPU E5-2687w v4 @ 3.00GHz, 64GB main memory, and Windows 10.

- **Label propagation with structure (LP-Structure)**: In the implementation of this baseline method, we used the maximum likelihood classifier (MLC) and GBM respectively to pre-classify fully observed samples and then ran label propagation (Wang and Zhang 2007) on the topography tree structure. We named them as **LP-Structure-MLC** and **LP-Structure-GBM**. The initial classifiers were from R packages.
- **EM with i.i.d. assumption (EM-i.i.d.)**: In the implementation of this baseline method (Ghahramani and Jordan 1994a), we treated missing features and unknown classes as latent variables and used the EM algorithm assuming that sample features follow i.i.d. Gaussian distribution in each class. Moreover, we assumed RGB (red, green, blue) features and elevation features are uncorrelated.
- **EM with structure (EM-Structure)**: This is our proposed approach. We treated unknown classes as latent variables and used the EM algorithm assuming that samples follow the topography tree dependency structure. The codes were implemented in C++.

Data Description: Our real-world datasets were collected from Kinston North Carolina and Grimesland North Carolina in Hurricane Matthew 2016. We used aerial imageries from NOAA National Geodetic Survey (National Oceanic and Atmospheric Administration 2017) with red, green, blue bands in a 2-meter spatial resolution and digital elevation map from the University of North Carolina Libraries (NCSU Libraries 2018). The test region size was 1743 by 1349 in Kinston and 2757 by 3853 in Grimesland. The number of observation samples was 31,168 in Kinston and 237,312 in Grimesland. The number of training and testing samples (pixels) are listed in Table 2.

Evaluation Metric: For classification performance evaluation, we used precision, recall, and F-score. For computational performance evaluation, we measured the computational time costs in seconds.

Table 2: Dataset description

Dataset	Training Set		Testing Set	
	Dry	Flood	Dry	Flood
Matthew, Kinston	5,000	5,000	48,071	47,967
Matthew, Grimesland	5,000	5,000	75,670	59,405

Classification Performance Evaluation

Table 3: Comparison on Mathew, Kinston flood data

Classifiers	Class	Prec.	Recall	F	Avg. F
LP-Structure-GBM	Dry	0.91	0.56	0.69	0.74
	Flood	0.68	0.94	0.79	
LP-Structure-MLC	Dry	0.86	0.55	0.67	0.72
	Flood	0.67	0.91	0.77	
EM-i.i.d.	Dry	1.00	0.39	0.56	0.66
	Flood	0.62	1.00	0.76	
EM-Structure	Dry	0.94	0.99	0.96	0.96
	Flood	0.99	0.94	0.96	

Table 4: Comparison on Mathew, Grimesland flood data

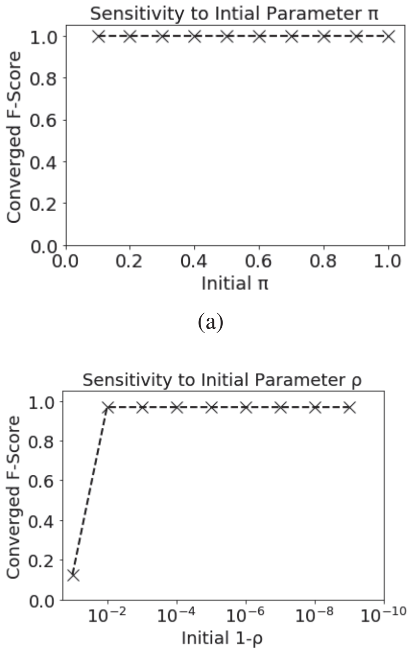
Classifiers	Class	Prec.	Recall	F	Avg. F
LP-Structure-GBM	Dry	0.81	0.60	0.69	0.70
	Flood	0.61	0.82	0.70	
LP-Structure-MLC	Dry	0.90	0.75	0.82	0.81
	Flood	0.73	0.90	0.81	
EM-i.i.d.	Dry	0.83	0.74	0.78	0.77
	Flood	0.71	0.80	0.75	
EM-Structure	Dry	0.99	0.96	0.97	0.97
	Flood	0.95	0.99	0.97	

We first compared different methods on their precision, recall, and F-score on the two real-world datasets. The results were summarized in Table 3 and Table 4 respectively. On the Kinston dataset, EM algorithm with the i.i.d. assumption performed the worst with an average F-score of 0.66. The reason was that this method was not able to utilize the spatial structural constraint between sample classes. Its training process only updated the parameter of Gaussian feature distribution in each class. When predicting the classes of samples with only elevation feature, the method used only the learned Gaussian distribution of elevation feature on each class without considering spatial structure based on elevation values. On the same dataset, label propagation after pre-classification with the GBM model and the maximum likelihood classifier slightly outperformed the EM algorithm with the i.i.d. assumption. The main reason was that label propagation on the topography tree (split tree) structure utilized the physical constraint between sample classes when inferring the classes of unobserved samples without RGB features. However, label propagation still showed significant errors, particularly in the low recall on the dry class. Through analyzing the predicted map, we observed that the label propagation algorithm was very sensitive to the pre-classified class labels on the observed samples in the test region. Errors in the pre-classification phase may propagation into unobserved samples (those without RGB feature values). In label propagation methods, once the errors were propagated to unobserved samples, they were hard to be reverted. This was different from the EM algorithm, which could update the probabilities in iterations. We did not report the results of label propagation on a grid graph structure (only considering spatial neighborhood structure without physics-aware constraint) due to poor results. Our model based on the EM algorithm assuming structural dependency between class labels performed the best with an average F-score of 0.96. The main reason was that our model could leverage the physical constraint to infer unobserved samples, and also could effectively update sample probabilities during iterations with the EM algorithm. In our model, we used training samples to initialize the parameters of the Gaussian distribution of sample features in each class. Based on the reasonable initial parameters, we can have a reasonable estimation of the posterior class probabilities of all samples in the test region. Based on the posterior class probabilities, the distribution parameters could be further updated. The representative training samples helped make sure that parameter iterations would converge in the right path.

Similar results were observed on the Grimesland dataset. In the label propagation method, pre-classification based on GBM performed worse than pre-classification based on MLC. The reason may be due to overfitting of GBM compared with MLC when predicting initial labels on the fully observed samples. The EM algorithm with the i.i.d. assumption performed slightly better on this dataset. The reason might be that the final prediction of classes of the unobserved samples (with only elevation feature but without RGB features) was based on a slightly better fitted normal distribution. Our model showed the best performance with an F-score of 0.97.

The effect of model initial parameters We now analyzed the sensitivity of our proposed model on different initial parameter settings. The parameters of $\boldsymbol{\mu}_c$ and $\boldsymbol{\Sigma}_c$ were estimated from training data, but parameters ρ and π were from user input. Since ρ captured the transitional probability of a sample being flood given its parents were all flood, its value should be very high (close to 1) due to spatial autocorrelation. π is the initial class prior probabilities for samples without parent nodes (local lowest location). We could set it close to 0.5. We tested the sensitivity of our model to different initial values of ρ and π on the Kinston dataset. We first fixed ρ as 0.999 and varied the value of π from 0.1 to 0.9. Then we fixed π as 0.3 and varied the value of ρ from 0.9. The results were shown in Figure 5. We can see that the model was generally not very sensitive to the initial parameter values. For parameter ρ, as long as $1-\rho$ was smaller than 0.01 (ρ greater than 0.99), the converged F-score was good. For parameter π, the results were consistently good for our model with an initial π between 0.1 to 0.9. The main reason was that π influenced only a small number of samples at the local lowest locations on the elevation map.

The parameter iterations of our model were shown in Figure 6. The model converged fast with only 20 iterations. Due to the space limit, we only showed the parameters of ρ and



(a)

(b)

Figure 5: Sensitivity of our model to initial parameters π and ρ

π.

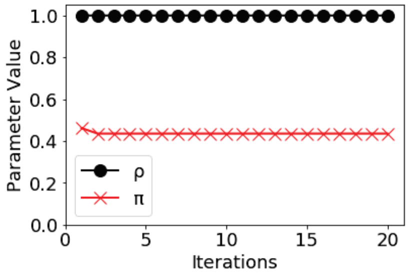


Figure 6: Parameter iterations and convergence in our model

Computational Performance Evaluation

We also evaluated the computational performance of our model learning and inference on different input data sizes. We used the Grimesland dataset to test the effect of different test region sizes. We varied the region size from around 2 million pixels to over 10 million pixels. The computational time costs of our model was shown in Figure 7. It can be seen that the time cost grows almost linearly with the size of the test region. This was because our learning and inference algorithms involve tree traversal operations with a linear time complexity on the tree size (the number of pixels on the test region). The model was computationally efficient. It could classify around 10 million pixels in around 2 minutes.

We further analyzed the time costs of different components in our model, including split tree construction, model parameter learning, and class inference. We analyzed the results on both datasets (same as the settings in Table 3 and Table 4. Results showed that tree construction and class inference took less time than parameter learning. This was because the learning involves multiple iterations of message propagation (tree traversal operations).

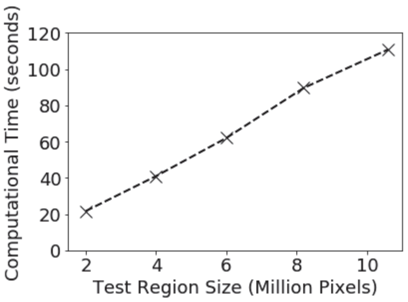


Figure 7: Computational performance of out model on varying test region sizes

Table 5: Time Costs Analysis of Our Model (seconds)

	Kinston	Grimesland
Tree construction	3.2	8.39
Parameter learning	25.74	86.79
Class inference	3.8	15.62
Total time	32.74	110.80

Conclusions and Future Work

In this paper, we address the problem of spatial classification with limited feature observations. The problem is important in many applications where only a subset of sensors are deployed at certain regions or partial responses are collected in field surveys. Existing research on incomplete or missing data has limitations in assuming that incomplete feature observations only happen on a small subset of samples, and thus cannot solve problems whereby the vast majority of samples have missing feature observations. To address this issue, we propose a new approach that incorporates physics-aware structural constraints into model representation. We propose efficient algorithms for model parameter learning and class inference. Evaluations on real-world hydrological applications show that our approach significantly outperforms several baseline methods in classification accuracy, and the proposed solution is computationally efficient on a large data volume.

In future work, we plan to extend our proposed model to address other problems such as fusing noisy, incomplete, and multi-modal observation data such as volunteered geographic information (VGI). We also plan to explore the integration of deep learning framework with our approach.

Acknowledgement

This material is based upon work supported by the NSF under Grant No. IIS 1850546, the University Corporation for Atmospheric Research (UCAR) and Camgian Microsystems.

References

Batista, G. E.; Monard, M. C.; et al. 2002. A study of k-nearest neighbour as an imputation method. *HIS* 87(251-260):48.

Bengio, Y., and Gingras, F. 1996. Recurrent neural networks for missing or asynchronous data. In *Advances in neural information processing systems*, 395–401.

Carr, H.; Snoeyink, J.; and Axen, U. 2003. Computing contour trees in all dimensions. *Computational Geometry* 24(2):75–94.

Chechik, G.; Heitz, G.; Elidan, G.; Abbeel, P.; and Koller, D. 2007. Max-margin classification of incomplete data. In *Advances in Neural Information Processing Systems*, 233–240.

Cline, D. 2009. Integrated water resources science and services: an integrated and adaptive roadmap for operational implementation. Technical report, National Oceanic and Atmospheric Administration.

Edelsbrunner, H., and Harer, J. 2010. *Computational topology: an introduction*. American Mathematical Soc.

García-Laencina, P. J.; Sancho-Gómez, J.-L.; and Figueiras-Vidal, A. R. 2010. Pattern classification with missing data: a review. *Neural Computing and Applications* 19(2):263–282.

Ghahramani, Z., and Jordan, M. 1994a. Learning from incomplete data (tech. rep. no. aim-1509).

Ghahramani, Z., and Jordan, M. I. 1994b. Supervised learning from incomplete data via an em approach. In *Advances in neural information processing systems*, 120–127.

Ghahramani, Z., and Jordan, M. I. 1995. Learning from incomplete data.

Jiang, Z., and Sainju, A. M. 2018. Hidden markov contour tree: A spatial structured model for hydrological applications. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '19. ACM.

Jiang, Z., and Shekhar, S. 2017. *Spatial big data science*. Springer International Publishing.

Jiang, K.; Chen, H.; and Yuan, S. 2005. Classification for incomplete data using classifier ensembles. In *2005 International Conference on Neural Networks and Brain*, volume 1, 559–563. IEEE.

Jiang, Z.; Xie, M.; and Sainju, A. M. 2019. Geographical hidden markov tree. *IEEE Transactions on Knowledge and Data Engineering*.

Jiang, Z. 2019. A survey on spatial prediction methods. *IEEE Transactions on Knowledge and Data Engineering* 31(9):1645–1664.

Kschischang, F. R.; Frey, B. J.; and Loeliger, H.-A. 2001. Factor graphs and the sum-product algorithm. *IEEE Transactions on information theory* 47(2):498–519.

Little, R. J., and Rubin, D. B. 2019. *Statistical analysis with missing data*, volume 793. John Wiley & Sons.

McLachlan, G., and Krishnan, T. 2007. *The EM algorithm and extensions*, volume 382. John Wiley & Sons.

Merwade, V.; Olivera, F.; Arabi, M.; and Edleman, S. 2008. Uncertainty in flood inundation mapping: current issues and future directions. *Journal of Hydrologic Engineering* 13(7):608–620.

National Oceanic and Atmospheric Administration. 2017. Data and imagery from noaa's national geodetic survey. https://www.ngs.noaa.gov.

National Oceanic and Atmospheric Administration. 2018. National Water Model: Improving NOAA's Water Prediction Services. http://water.noaa.gov/documents/wrn-national-water-model.pdf.

NCSU Libraries. 2018. LIDAR Based Elevation Data for North Carolina. https://www.lib.ncsu.edu/gis/elevation.

Pelckmans, K.; De Brabanter, J.; Suykens, J. A.; and De Moor, B. 2005. Handling missing values in support vector machine classifiers. *Neural Networks* 18(5-6):684–692.

Quinlan, J. R. 1989. Unknown attribute values in induction. In *Proceedings of the sixth international workshop on Machine learning*, 164–168. Elsevier.

Quinlan, J. R. 2014. *C4. 5: programs for machine learning*. Elsevier.

Rabiner, L. R. 1989. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE* 77(2):257–286.

Ronen, O.; Rohlicek, J.; and Ostendorf, M. 1995. Parameter estimation of dependence tree models using the em algorithm. *IEEE Signal Processing Letters* 2(8):157–159.

Rubin, D. B. 2004. *Multiple imputation for nonresponse in surveys*, volume 81. John Wiley & Sons.

Schafer, J. L. 1997. *Analysis of incomplete multivariate data*. Chapman and Hall/CRC.

Smola, A. J.; Vishwanathan, S.; and Hofmann, T. 2005. Kernel methods for missing variables. In *AISTATS*. Citeseer.

Wang, F., and Zhang, C. 2007. Label propagation through linear neighborhoods. *IEEE Transactions on Knowledge and Data Engineering* 20(1):55–67.

Webb, G. I. 1998. The problem of missing values in decision tree grafting. In *Australian Joint Conference on Artificial Intelligence*, 273–283. Springer.

Williams, D.; Liao, X.; Xue, Y.; Carin, L.; and Krishnapuram, B. 2007. On classification with incomplete data. *IEEE transactions on pattern analysis and machine intelligence* 29(3):427–436.

Xie, M.; Jiang, Z.; and Sainju, A. M. 2018. Geographical hidden markov tree for flood extent mapping. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '18, 2545–2554. ACM.

Yang, X., and Jin, W. 2010. Gis-based spatial regression and prediction of water quality in river networks: A case study in iowa. *Journal of Environmental Management* 91(10):1943–1951.

Yoon, S.-Y., and Lee, S.-Y. 1999. Training algorithm with incomplete data for feed-forward neural networks. *Neural Processing Letters* 10(3):171–179.