Spatial Big Data Analytics: Classification Techniques for Earth Observation Imagery

A THESIS
SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL
OF THE UNIVERSITY OF MINNESOTA
BY

Zhe Jiang

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
Doctor of Philosophy

SHASHI SHEKHAR, advisor

August, 2016

Acknowledgements

There are many people that have earned my gratitude for their contribution to my life in graduate school. First, I would like to thank my advisor Prof. Shashi Shekhar for opening the door of interdisciplinary research to me, and helping me develop various scientific skills from identifying societally important problems to effectively communicating results to different audiences. I still remember the night six years ago when I was sitting in my dormitory at USTC, and receiving an email reply from Prof. Shekhar about an official offer letter. Without his trust, support, guidance, encouragement, and patience, this dissertation would not have been possible and I could not have grown into a research scholar from a bashful fresh Ph.D. student. I would also thank my committee members and collaborators including Prof. Joseph Knight, Prof. Vipin Kumar, Prof. Arindam Banerjee, Prof. Snigdhansu Chatterjee, Prof. Ce Yang, Dr. Michael Steinbach, Dr. Jennifer Corcoran, and Dr. Lian Rampi. Particularly, I want to thank Prof. Knight for helping me learn the background knowledge on remote sensing and wetland mapping, and providing an enjoyable and rewarding collaborative experience. Moreover, I would like to thank my mentor and friend, Prof. Hui Xiong, for his guidance. Without him, I would not have the opportunity to connect to Prof. Shekhar and join the spatial computing group.

I want to thank my lovely teammates and friends in the spatial computing research group, including Reem Ali, Emre Eftelioglu, Michael Evans, Viswanath Gunturi, Yan Li, Pradeep Mohan, Dev Oliver, Xun Tang, Yiqun Xie, Kwangsoo Yang, and Xun Zhou. You really make my Ph.D. life much easier and much more enjoyable. Also, I would like to thank my friends in the department including Jie Bao, Xi Chen, Yanhua Li, Huanan Zhang, Wei Zhang, Jaya Kawale, Anuj Karpatne, and Gowtham Atluri for their help.

Finally, I want to thank my father Zhenchen Jiang and my mother Minghong Che, for their love, support, and encouragement. They set a good example for me to be a life learner.

Dedication

To my parents Zhenchen Jiang and Minghong Che.

Abstract

Spatial Big Data (SBD), e.g., earth observation imagery, GPS trajectories, temporally detailed road networks, etc., refers to geo-referenced data whose volume, velocity, and variety exceed the capability of current spatial computing platforms. SBD has the potential to transform our society. Vehicle GPS trajectories together with engine measurement data provide a new way to recommend environmentally friendly routes. Satellite and airborne earth observation imagery plays a crucial role in hurricane tracking, crop yield prediction, and global water management. The potential value of earth observation data is so significant that the White House recently declared that full utilization of this data is one of the nation's highest priorities. However, SBD poses significant challenges to current big data analytics. In addition to its huge dataset size (NASA collects petabytes of earth images every year), SBD exhibits four unique properties related to the nature of spatial data that must be accounted for in any data analysis. First, SBD exhibits spatial autocorrelation effects. In other words, we cannot assume that nearby samples are statistically independent. Current analytics techniques that ignore spatial autocorrelation often perform poorly such as low prediction accuracy and salt-and-pepper noise (i.e., pixels predicted as different from neighbors by mistake). Second, spatial interactions are not isotropic and vary across directions. Third, spatial dependency exists in multiple spatial scales. Finally, spatial big data exhibits heterogeneity, i.e., identical feature values may correspond to distinct class labels in different regions. Thus, learned predictive models may perform poorly in many local regions.

My thesis investigates novel SBD analytics techniques to address some of these challenges. To date, I have been mostly focusing on the challenges of spatial autocorrelation and anisotropy via developing novel spatial classification models such as spatial decision trees for raster SBD (e.g., earth observation imagery). To scale up the proposed models, I developed efficient learning algorithms via computational pruning. The proposed techniques have been applied to real world remote sensing imagery for wetland mapping. I also had developed spatial ensemble learning framework to address the challenge of spatial heterogeneity, particularly the class ambiguity issues in geographical classification, i.e., samples with the same feature values belong to different classes in different spatial zones. Evaluations on three real world remote sensing datasets confirmed that proposed spatial ensemble learning outperforms current approaches such as bagging, boosting, and mixture of experts when class ambiguity exists.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Spatial Big Data Analytics

Spatial big data (SBD) is geo-referenced data whose volume, velocity, and variety exceed the capability of current common spatial computing platforms. Examples of SBD include GPS trajectories (10^{13} records per year), earth observation imagery (10^{15} bytes per year by NASA only), and check-in location history (10^{6} per day). Spatial big data analytic is the process of discovering interesting, previously unknown, but potentially useful patterns from SBD. Figure 1.1 shows the entire knowledge discovery process. The core component is spatial big data analytic algorithms, which take input spatial big data and produce desired output pattern families, including spatial or spatiotemporal outliers, associations and tele-connections, predictive models, partitions and summarization, hotspots, as well as change patterns. For example, spatial prediction can be used to classify earth observation images into different land cover types. Spatial colocation patterns can identify crime event types that frequently occur together. These algorithms have statistical foundations and integrate scalable computational techniques and platforms. The type of input data and the choice of output patterns often determine which kind of algorithms are appropriate to use.

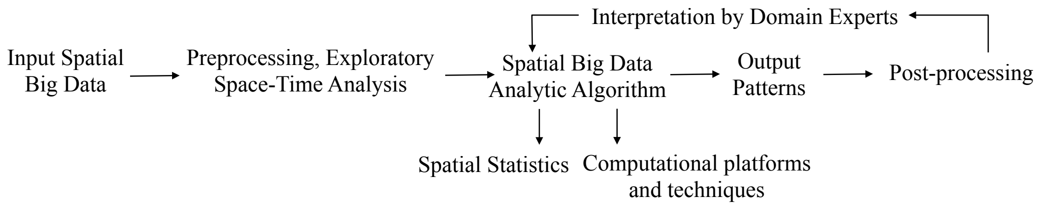

Interpretation by Domain Experts
Input Spatial Big Data
Preprocessing, Exploratory Space-Time Analysis
Spatial Big Data Analytic Algorithm
Output Patterns
Post-processing
Spatial Statistics
Computational platforms and techniques

Figure 1.1: The process of spatial big data analytics.

1.2 Societal Applications

Spatial big data analytics are crucial to organizations which make decisions based on large spatial and spatiotemporal datasets, including NASA, the National Geospatial-Intelligence Agency, the National Cancer Institute, the US Department of Transportation, and the National Institute of Justice. These organizations are spread across many application domains. In ecology and environmental management, researchers need tools to classify remote sensing images to map forest coverage. In public safety, crime analysts are interested in discovering hotspot patterns from crime event maps so as to effectively allocate police resources. In transportation, researchers analyze historical taxi GPS trajectories to recommend fast routes from places to places. Epidemiologists use spatial big data techniques to detect disease outbreak. There are also other application domains such as earth science, climatology, precision agriculture, and Internet of Things.

The interdisciplinary nature of spatial big data analytics means that techniques must be developed with awareness of the underlying physics or theories in their application domains [41]. For example, climate science studies find that observable predictors for climate phenomena discovered by data driven techniques can be misleading if they do not take into account climate models, locations, and seasons [42]. In this case, statistical significance testing is critically important in order to further validate or discard relationship patterns mined from data.

1.3 Challenges

Spatial big data analytics pose unique statistical and computational challenges. In addition to its huge volume, SBD has the following unique characteristics that challenge current big data analytic techniques.

1.3.1 Spatial Autocorrelation

According to Tobler's first law of geography, "Everything is related to everything else, but near things are more related than distant things." For example, people with similar characteristics, occupation and background tend to cluster together in the same neighborhood. In spatial statistics, such spatial dependence is called the spatial autocorrelation effect. Ignoring autocorrelation and assuming an identical and independent distribution (i.i.d.) when analyzing data with spatial characteristics may produce hypotheses or models that are inaccurate or inconsistent with the data set [2] (e.g., salt-and-pepper noise in remote sensing image classification). Similarly, due to the fact that spatial big data is embedded in continuous space, many classical data mining techniques assuming discrete data (e.g., transactions in association rule mining) may not be effective (e.g., breaking neighboring locations into different transactions).

1.3.2 Spatial Anisotropy

Another challenge is that the degree of spatial dependency also varies across different directions (also called spatial anisotropy) due to irregular geographical terrains, topological relationships, etc. For example, biogeographical patterns on river networks are often constrained by the network topological structure and flow directions. Many current spatial statistics assume isotropy and use spatial neighborhoods with regular shapes (e.g., square window) to model spatial dependency. This may result in inaccurate models and predictions.

1.3.3 Multi-scale Effect

Modifiable area unit problem (MAUP) or multi-scale effect is another challenge since results of spatial analysis depends on the choice of an appropriate spatial scale (e.g., local, regional, global). For example, spatial autocorrelation values at local level may be significantly different from values at global level, especially when spatial outliers exist. As another instance of example, patterns of spatial interactions between two types of events may be significant in one region of the study area, but insignificant in other areas.

1.3.4 Spatial Heterogeneity

The last challenge is the spatial heterogeneity, i.e., spatial data samples do not follow an identical distribution across the entire space. One type of spatial heterogeneity is that samples with the same explanatory features may belong to different class labels in different zones. For example, upland forest looks very similar to wetland forest in spectral values on remote sensing images, but they are from different land cover classes due to different geographical terrains. Another types of spatial heterogeneity is different trends between explanatory variables and response variable in different locations. For instance, in economic studies, it may be possible that old houses are with high price in rural areas, but with low price in urban areas. Though house age is not an effective coefficient for house price when the entire study area is considered, it is an effective coefficient in each local areas (rural or urban).

One way to deal with implicit spatial relationships is to materialize the relationships into traditional data input columns and then apply classical big data analytic techniques. However, the materialization can result in loss of information [2]. Another issue is the existence of a semantic gap between traditional big data algorithms and spatial and spatiotemporal data. For example, Ring-shaped hotspot pattern is very important in environmental criminology but is hard to characterize in the matrix space as in traditional data mining. Finally, many traditional data mining methods are not spatial or spatiotemporal statistical aware and thus prone to produce spurious spatial patterns. A more preferable way to capture implicit spatial and temporal relationships is to develop statistics and techniques to incorporate spatial and temporal information into the data analytic process.

1.4 Contributions

In this thesis, we overview current spatial data analytic techniques, and introduce two novel spatial big data classification approaches, i.e., spatial decision tree, and spatial ensemble learning, which address some of the above challenges.

- **Chapter 2** surveys current techniques in spatial and spatiotemporal data mining. Spatial and spatiotemporal (SST) data mining studies the process of discovering interesting, previously unknown, but potentially useful patterns from large SST databases. It has broad application domains including ecology, environmental management, public safety, etc. The complexity of input data and intrinsic spatial and spatiotemporal relationships limits the usefulness of conventional data mining methods. We review recent computational techniques in SST data mining. Compared with other surveys, this chapter emphasizes the statistical foundation and proposes a taxonomy of major pattern families to categorize recent research.

- **Chapter 3** discusses a novel spatial classification technique called spatial decision tree to address the challenge of spatial autocorrelation and anisotropy. Given learning samples from a raster dataset, spatial decision tree learning aims to find a decision tree classifier that minimizes classification errors as well as salt-and-pepper noise. The problem has important societal applications such as land cover classification for natural resource management. However, the problem is challenging due to the fact that learning samples show spatial autocorrelation in class labels, instead of being independently identically distributed. Related work relies on local tests (i.e., testing feature information of a location) and cannot adequately model the spatial autocorrelation effect, resulting in salt-and-pepper noise. In contrast, we recently proposed a focal-test-based spatial decision tree (FTSDT), in which the tree traversal direction of a sample is based on both local and focal (neighborhood) information. Preliminary results showed that FTSDT reduces classification errors and salt-and-pepper noise. We also extend our recent work by introducing a new focal test approach with anisotropic spatial neighborhoods that avoids over-smoothing in wedge-shaped areas. We also conduct computational refinement on the FTSDT training algorithm by reusing focal values across candidate thresholds. Theoretical analysis shows that the refined training algorithm is correct and more scalable. Experiment results on real world datasets show that new FTSDT with adaptive neighborhoods improves classification accuracy, and that our computational refinement significantly reduces training time.

- **Chapter 4** discusses a novel ensemble learning framework called spatial ensemble to address the challenge of spatial heterogeneity. Given geographical data with class ambiguity, i.e., samples with similar features belonging to different classes in different zones, the spatial ensemble learning (SEL) problem aims to find a decomposition of the geographical area into disjoint zones minimizing class ambiguity and to learn a local classifier in each zone. Class ambiguity is a common issue in many geographical classification applications. For example, in remote sensing image classification, pixels with the same spectral signatures may correspond to different land cover classes in different locations due to heterogeneous geographical terrains. A global classifier may mistakenly classify those ambiguous pixels into one land cover class. However, SEL problem is challenging due to class ambiguity issue, unknown and arbitrary shapes of zonal footprints, and high computational cost due to the potential exponential number of candidate zonal partitions. Related work in ensemble learning either assumes an identical and independent distribution of input data (e.g., bagging, boosting) or decomposes multi-modular input data in the feature vector space (e.g., mixture of experts), and thus cannot effectively decompose the input data in geographical space to reduce class ambiguity. In contrast, we propose a spatial ensemble learning framework that explicitly partition input data in geographical space: first, the input data is preprocessed into homogeneous "patches" via constrained hierarchical spatial clustering; second, patches are grouped into several footprints via greedy seed growing and spatial adjustments. Experimental evaluation on three real world remote sensing datasets show that the proposed approach outperforms related work in classification accuracy.

Chapter 2

Spatial and Spatiotemporal Data Mining Overview

This chapter overviews the state of the art spatial and spatial temporal data mining techniques. Current overview tutorials and surveys in spatial and spatiotemporal data mining can be categorized into two groups: early papers in the 1990s without a focus on spatial and spatiotemopral statistical foundations, and recent papers with a focus on statistical foundation. Two early survey papers [43, 44] review spatial data mining from a database approach. Recent papers include brief tutorials on current spatial [45] and spatiotemporal data mining [46] techniques. There are also other relevant book chapters [47, 48, 2], as well as survey papers on specific spatial or spatiotemporal data mining tasks such as spatiotemporal clustering [49], spatial outlier detection [50], and spatial and spatiotemporal change footprint detection [51].

This chapter makes the following contributions: (1) we provide a categorization of input spatial and spatiotemporal data types; (2) we provide a summary of spatial and spatiotemporal statistical foundations categorized by different data types; (3) we create a taxonomy of six major output pattern families, including spatial and spatiotemporal outliers, associations and tele-connections, predictive models, partitioning (clustering) and summarization, hotspots and changes. Within each pattern family, common computational approaches are categorized by the input data types; (4) we analyze the research trends and future research needs.

Organization of the chapter: This chapter starts with a summary of input spatial and spatiotemporal data (Section 2.1) and an overview of statistical foundation (Section 2.2). It then describes in detail six main output pattern families including spatial and spatiotemporal outliers, associations and tele-connections, predictive models, partitioning (clustering) and summarization, hotspots, and changes (Section 2.3). An examination of research trend and future research needs is in Section 2.4. Section 2.5 summarizes the chapter.

2.1 Input: Spatial and Spatiotemporal Data

2.1.1 Types of Spatial and Spatiotemporal Data

The data inputs of spatial and spatiotemporal data mining tasks are more complex than the inputs of classical data mining tasks because they include discrete representations of continuous space and time. Table 2.1 gives a taxonomy of different spatial and spatiotemporal data types (or models). Spatial data can be categorized into three models, *i.e.*, the object model, the field model, and the spatial network model [40, 52]. Spatiotemporal data, based on how temporal information is additionally modeled, can be categorized into three types, *i.e.*, temporal snapshot model, temporal change model, and event or process model [53, 54, 55]. In the temporal snapshot model, spatial layers of the same theme are time-stamped. For instance, if the spatial layers are points or multi-points, their temporal snapshots are trajectories of points or spatial time series (*i.e.*, variables observed at different times on fixed locations). Similarly, snapshots can represent trajectories of lines and polygons, raster time series, and spatiotemporal networks such as time expanded graphs (TEGs) and time aggregate graphs (TEGs) [56, 57]. The temporal change model represents spatiotemporal data with a spatial layer at a given start time together with incremental changes occurring afterward. For instance, it can represent motion (e.g., Brownian motion, random walk [5]) as well as speed and acceleration on spatial points, as well as rotation and deformation on lines and polygons. Event and process models represent temporal information in terms of *events* or *processes*. One way to distinguish events from processes is that events are entities whose properties are possessed timelessly and therefore are not subject to change over time, whereas processes are entities that are subject to change over time (e.g., a process may be said to be accelerating or slowing down) [58].

Table 2.1: Taxonomy of Spatial and Spatiotemporal Data Models.

Spatial Data	**Temporal Snapshots (Time Series)**	**Temporal Change (Delta/Derivative)**	**Events/Processes**
Object model	Trajectories, Spatial time series	Motion, speed, acceleration, split or merge	Spatial or spatiotemporal point process
Field model	Raster time series	Change across raster snapshots	Cellular automation
Spatial network	Spatiotemporal network	Addition or removal of nodes, edges	

2.1.2 Data Attributes and Relationships

There are three distinct types of data attributes for spatiotemporal data, including non-spatiotemporal attributes, spatial attributes, and temporal attributes. Non spatiotemporal attributes are used to characterize non-contextual features of objects, such as name, population, and unemployment rate for a city. They are the same as the attributes used in the data inputs of classical data mining [59]. Spatial attributes are used to define the spatial location (e.g., longitude and latitude), spatial extent (e.g., area, perimeter) [60, 61], shape, as well as elevation defined in a spatial reference frame. Temporal attributes include the timestamp of a spatial object, a raster layer, or a spatial network snapshot, as well as the duration of a process. Relationships on non-spatial attributes are often explicit, including arithmetic, ordering, and subclass, etc. Relationships on spatial attributes, in contrast, are often implicit, including those in topological space (e.g., meet, within, overlap), set space (e.g., union, intersection), metric space (e.g., distance) and directions. Relationships on spatiotemporal attributes are more sophisticated, as summarized in Table 2.2.

Table 2.2: Relationships on Spatiotemporal Data.

Spatial Data	**Temporal Snapshots (Time Series)**	**Change (Delta/Derivative)**	**Event/Process**
Object model	Spatiotemporal predicates [62], Trajectory distance [63, 64], Spatial time series correlation [65], tele-connection [66]	Motion, speed, acceleration, attraction or repulsion, split/merge	Spatiotemporal covariance [5], Spatiotemporal coupling for point events or extended spatial objects [67, 68, 69, 70, 71, 72]
Field model	Cubic map algebra [73], Temporal correlation, tele-connection	Local, focal, zonal change across snapshots [51]	Cellular automation [74]

One way to deal with implicit spatiotemporal relationships is to materialize the relationships into traditional data input columns and then apply classical data mining techniques [75, 76, 77, 78, 79]. However, the materialization can result in loss of information [2]. The spatial and temporal vagueness which naturally exists in data and relationships usually creates further modeling and processing difficulty in spatial and spatiotemporal data mining. A more preferable way to capture implicit spatial and spatiotemporal relationships is to develop statistics and techniques to incorporate spatial and temporal information into the data mining process. These statistics and techniques are the main focus the survey.

2.2 Statistical Foundations

2.2.1 Spatial Statistics for Different Types of Spatial Data

Spatial statistics [3, 4, 5, 6] is a branch of statistics concerned with the analysis and modeling of spatial data. The main difference between spatial statistics and classical statistics is that spatial data often fails to meet the assumption of an identical and independent distribution (i.i.d.). As summarized in Table 2.3, spatial statistics can be categorized according to their underlying spatial data type: Geostatistics for point referenced data, lattice statistics for areal data, and spatial point process for spatial point patterns.

Geostatistics: Geostatistics [6] deal with the analysis of the properties of point reference data, including spatial continuity (i.e., dependence across locations), weak stationarity (i.e., first and second moments do not vary with respect to locations) and isotropy (i.e., uniformity in all directions). For example, under the assumption of weak stationarity (or more specifically intrinsic stationarity), variance of the difference of non-spatial attribute values at two point locations is a function of point location difference regardless of specific point locations. This function is called a variogram [29]. If the variogram only depends on distance between two locations (not varying with respect to directions), it is further called isotropic. Under the assumptions of these properties, Geostatistics also provides a set of statistical tools such as Kriging [29], which can be used to interpolate non-spatial attribute values at unsampled locations. Finally, real world spatial data may not always satisfy the stationarity assumption. For example, different jurisdictions tend to produce different laws (e.g., speed limit differences between Minnesota and Wisconsin). This effect is called spatial heterogeneity or non-stationarity. Special models (e.g., geographically weighted regression, or GWR [37]) can be further used to model the varying co-efficients at different locations.

Lattice statistics: Lattice statistics studies statistics for spatial data in the field (or areal) model. Here a lattice refers to a countable collection of regular or irregular cells in a spatial framework. The range of spatial dependency among cells is reflected by a neighborhood relationship, which can be represented by a contiguity matrix called a W-matrix. A spatial neighborhood relationship can be defined based on spatial adjacency (e.g., rook or queen neighborhoods) or Euclidean distance, or in more general models, cliques and hypergraphs [9]. Based on a W-matrix, spatial autocorrelation statistics can be defined to measure the correlation of a non-spatial attribute across neighboring locations. Common spatial autocorrelation statistics include Moran's I, Getis-Ord $Gi*$, Geary's C, Gamma index Γ [7], *etc.*, as well as their local versions called local indicators of spatial association (LISA) [10]. Several spatial statistical models, including the spatial autoregressive model (SAR), conditional autoregressive model (CAR), Markov random fields (MRF), as well as other Bayesian hierarchical models [3], can be used to model lattice data. Another important issue is the modifiable areal unit problem (MAUP) (also called the multi-scale effect) [11], an effect in spatial analysis that results for the same analysis method will change on different aggregation scales. For example, analysis using data aggregated by states will differ from analysis using data at individual family level.

Table 2.3: Taxonomy of Spatial and Spatiotemporal Statistics.

Spatial Model	**Spatial Statistics**	**Spatiotemporal Statistics**
Object model	Geostatistics: • Stationarity, isotropy, variograms, Kriging Spatial Point Processes: • Poisson point process, Spatial scan statistics, Ripley's K function	Statistics for spatial time series: • Spatiotemporal stationarity, variograms, covariance, Kriging; • Temporal autocorrelation, telecoupling. Spatiotemporal Point Processes: • Spatiotemporal Poission point process; Spatiotemporal scan statistics; Spatiotemporal K-function.
Field model	Lattice Statistics (areal data model): • W-matrix, spatial autocorrelation, local indicators of spatial association (LISA); • MRF, SAR, CAR, Bayesian hierarchical model	Statistics for raster time series: • EOF analysis, CCA analysis; • Spatiotemporal autoregressive model (STAR), Bayesian hierarchical model, Dynamic Spatiotemporal model (Kalman filter), data assimilation
Spatial network	Spatial network autocorrelation, Network K function, Network Kriging	

Spatial point processes: A spatial point process is a model for the spatial distribution of the points in a point pattern. It differs from point reference data in that the random variables are locations. Examples include positions of trees in a forest and locations of bird habitats in a wetland. One basic type of point process is a homogeneous spatial Poisson point process (also called complete spatial randomness, or CSR) [5], where point locations are mutually independent with the same intensity over space. However, real world spatial point processes often show either spatial aggregation (clustering) or spatial inhibition instead of complete spatial independence as in CSR. Spatial statistics such as Ripley's K function [12, 13], *i.e.*, the average number of points within a certain distance of a given point normalized by the average intensity, can be used to test spatial aggregation of a point pattern against CSR. Moreover, real world spatial point processes such as crime events often contain hotspot areas instead of following homogeneous intensity across space. A spatial scan statistic [14] can be used to detect these hotspot patterns. It tests if the intensity of points inside a scanning window is significantly higher (or lower) than outside. Though both the K-function and spatial scan statistics have the same null hypothesis of CSR, their alternative hypotheses are quite different: the K-function tests if points exhibit spatial aggregation or inhibition instead of independence, while spatial scan statistics assume that points are independent and test if a local hotspot with much higher intensity than outside exists. Finally, there are other spatial point processes such as the Cox process, in which the intensity function itself is a random function over space, as well as a cluster process, which extends a basic point process with a small cluster centered on each original point [5]. For extended spatial objects such as lines and polygons, spatial point processes can be generalized to line processes and flat processes in stochastic geometry [15].

Spatial network statistics: Most spatial statistics research focuses on the Euclidean space. Spatial statistics on the network space are much less studied. Spatial network space, e.g., river networks and street networks, is important in applications of environmental science and public safety analysis. However, it poses unique challenges including directionality and anisotropy of spatial dependency, connectivity, as well as high computational cost. Statistical properties of random fields on a network are summarized in [16]. Recently, several spatial statistics, such as spatial autocorrelation, K-function, and Kriging, have been generalized to spatial networks [17, 18, 19]. Little research has been done on spatiotemporal statistics on the network space.

2.2.2 Spatiotemporal Statistics

Spatiotemporal statistics [80, 5] combine spatial statistics with temporal statistics (time series analysis [81], dynamic models [80]). Table 2.3 summarizes common statistics for different spatiotemporal data types, including spatial time series, spatiotemporal point process, and time series of lattice (areal) data.

Spatial time series: Spatial statistics for point reference data have been generalized for spatiotemporal data [82]. Examples include spatiotemporal stationarity, spatiotemporal covariance, spatiotemporal variograms, and spatiotemporal Kriging [80, 5]. There is also temporal autocorrelation and tele-coupling (high correlation across spatial time series at a long distance). Methods to model spatiotemporal process include physics inspired models (e.g., stochastically differential equations) [5] and hierarchical dynamic spatiotemporal models (e.g., Kalman filtering) for data assimilation [5].

Spatiotemporal point process: A spatiotemporal point process generalizes the spatial point process by incorporating the factor of time. As with spatial point processes, there are spatiotemporal Poisson process, Cox process, and cluster process. There are also corresponding statistical tests including a spatiotemporal K function and spatiotemporal scan statistics [5].

Time series of lattice (areal) data: Similar to lattice statistics, there are spatial and temporal autocorrelation, SpatioTemporal Autoregressive Regression (STAR) model [83], and Bayesian hierarchical models [3]. Other spatiotemporal statistics include empirical orthogonal function (EOF) analysis (principle component analysis in geophysics), canonical-correlation analysis (CCA), and dynamic spatiotemporal models (Kalman filter) for data assimilation [80].

2.3 Output Pattern Families

2.3.1 Spatial and Spatiotemporal Outlier Detection

This section reviews techniques for spatial and spatiotemporal outlier detection. The section begins with a definition of spatial or spatiotemporal outliers by comparison with global outliers. Spatial and spatiotemporal outlier detection techniques are summarized according to their input data types.

Problem definition: To understand the meaning of spatial and spatiotemporal outliers, it is useful first to consider global outliers. Global outliers [84, 85] have been informally defined as observations in a data set which appear to be inconsistent with the remainder of that set of data, or which deviate so much from other observations as to arouse suspicions that they were generated by a different mechanism. In contrast, a spatial outlier [86] is a spatially referenced object whose non-spatial attribute values differ significantly from those of other spatially referenced objects in its spatial neighborhood. Informally, a spatial outlier is a local *instability* or *discontinuity.* For example, a new house in an old neighborhood of a growing metropolitan area is a spatial outlier based on the non-spatial attribute house age. Similarly, a spatiotemporal outlier generalizes spatial outliers with a spatiotemporal neighborhood instead of a spatial neighborhood.

Statistical foundation: The spatial statistics for spatial outlier detection are also applicable to spatiotemporal outliers as long as spatiotemporal neighborhoods are well-defined. The literature provides two kinds of bi-partite multidimensional tests: graphical tests, including variogram clouds [87] and Moran scatterplots [10, 6], and quantitative tests, including scatterplot [88] and neighborhood spatial statistics [86].

Spatial outlier detection

The *visualization approach* plots spatial locations on a graph to identify spatial outliers. The common methods are variogram clouds and Moran scatterplot as introduced earlier.

The *neighborhood approach* defines a spatial neighborhood, and a spatial statistic is computed as the difference between the non-spatial attribute of the current location and that of the neighborhood aggregate [86]. Spatial neighborhoods can be identified by distance on spatial attributes (e.g., K nearest neighbors), or by graph connectivity (e.g., locations on road networks). This work has been extended in a number of ways to allow for multiple non-spatial attributes [89], average and median attribute value [90], weighted spatial outliers [91], categorical spatial outlier [92], local spatial outliers [93], and fast detection algorithms [94].

Spatiotemporal Outlier Detection

The intuition behind spatiotemporal outlier detection is that they reflect "discontinuity" on non-spatiotemporal attributes within a spatiotemporal neighborhood. Approaches can be summarized according to the input data types.

Outliers in spatial time series: For spatial time series (on point reference data, raster data, as well as graph data), basic spatial outlier detection methods, such as visualization based approaches and neighborhood based approaches, can be generalized with a definition of spatiotemporal neighborhoods.

Flow Anomalies: Given a set of observations across multiple spatial locations on a spatial network flow, flow anomaly discovery aims to identify dominant time intervals where the fraction of time instants of significantly mis-matched sensor readings exceeds the given percentage-threshold. Flow anomaly discovery can be considered as detecting *discontinuities* or *inconsistencies* of a non-spatiotemporal attribute within a neighborhood defined by the flow between nodes, and such discontinuities are persistent over a period of time. A time-scalable technique called SWEET (Smart Window Enumeration and Evaluation of persistent-Thresholds) was proposed [95] that utilizes several algebraic properties in the flow anomaly problem to discover these patterns efficiently.

2.3.2 Spatial and Spatiotemporal Associations, Tele-connections

This section reviews techniques for identifying spatial and spatiotemporal association as well as tele-connections. The section starts with the basic spatial association (or co-location) pattern, and moves on to spatiotemporal association (i.e., spatiotemporal co-occurrence, cascade, and sequential patterns) as well as spatiotemporal tele-connection.

Pattern definition: Spatial association, also known as spatial co-location patterns [96], represent subsets of spatial event types whose instances are often located in close geographic proximity. Real-world examples include symbiotic species, e.g., the Nile Crocodile and Egyptian Plover in ecology. Similarly, spatiotemporal association patterns represent spatiotemporal object types whose instances often occur in close geographic and temporal proximity. Spatiotemporal coupling patterns can be categorized according to whether there exists temporal ordering of object types: spatiotemporal (mixed drove) co-occurrences [97] are used for unordered patterns, spatiotemporal cascades [69] for partially ordered patterns, and spatiotemporal sequential patterns [71] for totally ordered patterns. Spatiotemporal tele-connection [65] represents patterns of significantly positive or negative temporal correlation between a pair of spatial time series.

Challenges: Mining patterns of spatial and spatiotemporal association is challenging due to the following reasons: first, there is no explicit transaction in continuous space and time; second, there is potential for over-counting; third, the number of candidate patterns is exponential, and a trade-off between statistical rigor of output patterns and computational efficiency has to be made.

Statistical foundation: The underlying statistic for spatiotemporal coupling patterns is the cross K function, which generalizes the basic Ripley's K function (introduced in Section 2.2) for multiple event types.

Common approaches: The following subsections categorize common computational approaches for discovering spatial and spatiotemporal couplings by different input data types.

Spatial colocation: Mining colocation patterns can be done via statistical approaches including cross-K function with Monte Carlo simulation [6], mean nearest-neighbor distance, and spatial regression model [98], but these methods are often computationally very expensive due to the exponential number of candidate patterns. In contrast, data mining approaches aim to identify colocation patterns like association rule mining. Within this category, there are transaction based approaches and distance based approaches. A transaction based approach defines transactions over space (e.g., around instances of a reference feature) and then uses an Apriori-like algorithm [99]. A distance based approach defines a distance-based pattern called k-neighboring class sets [100] or using an event centric model [96] based on a definition of *participation index*, which is an upper bound of cross-K function statistic and has an anti-monotone property. Recently, approaches have been proposed to identify colocations for extended spatial objects [101] or rare events [102], regional colocation patterns [103, 104] (i.e., pattern is significant only in a sub-region), statistically significant colocation [105], as well as design fast algorithms [106].

Spatiotemporal events associations represent subsets of two or more event-types whose instances are often located in close spatial and temporal proximity. Spatiotemproal event associations can be categorized into *spatiotemporal co-occurrences*, *spatiotemporal cascades*, and *spatiotemporal sequential patterns* for temporally unordered events, partially ordered events, and totally ordered events respectively. To discover spatiotemporal co-occurrences, a monotonic composite interest measure and novel mining algorithms are presented in [97]. A filter-and-refine approach has also been proposed to identify spatiotemporal co-occurrences on extended spatial objects [68]. A spatiotemporal sequential pattern represents a “chain reaction” from different event types. A measure of *sequence index*, which can be interpreted by K-function statistic, was proposed in [71], together with computationally efficient algorithms. For spatiotemporal cascade patterns, a statistically meaningful metric was proposed to quantify interestingness and pruning strategies were proposed to improve computational efficiency [69].

Spatiotemporal association from moving objects trajectories: Mining spatiotemporal association from trajectory data is more challenging than from spatiotemporal event data due to the existence of temporal duration, different moving directions, and imprecise locations. There are a variety of ways to define spatiotemporal association patterns from moving object trajectories. One way is to generalize the definition from spatiotemporal event data. For example, a pattern called spatiotemporal colocation episodes is defined to identify frequent sequences of colocation patterns that share a common event (object) type [107]. As another example, a spatiotemporal sequential pattern is defined based on decomposition of trajectories into line segments and identification of frequent region sequences around the segments [108]. Another way is to define spatiotemporal association as group of objects that frequently move together, either focusing on the footprints of subpaths (region sequences) that are commonly traversed [109] or subsets of objects that frequently move together (also called *travel companion*) [110].

Spatial time series oscillation and tele-connection: Given a collection of spatial time series at different locations, tele-connection discovery aims to identify pairs of spatial time series whose correlation is above a given threshold. Tele-connection patterns are important in understanding oscillations in climate science. Computational challenges arise from the large number of candidate pairs and the length of time series. An efficient index structure, called a cone-tree, as well as a filter and refine approach [65] have been proposed which utilize spatial autocorrelation of nearby spatial time series to filter out redundant pair-wise correlation computation. Another challenge is the existence of spurious 'high correlation' patterns that happen by chance. Recently, statistical significant tests have been proposed to identify statistically significant tele-connection patterns called dipoles from climate data [66]. The approach uses a "wild bootstrap" to capture the spatio-temporal dependencies, and takes account of the spatial autocorrelation, the seasonality and the trend in the time series over a period of time.

2.3.3 Spatial and Spatiotemporal Prediction

Problem definition: Given training samples with features and a target variable as well as a spatial neighborhood relationship among samples, the problem of *spatial prediction* aims to learn a model that can predict the target variable based on features. What distinguishes spatial prediction from traditional prediction problem in data mining is that data items are embedded in space, and often violate the common assumption of an identical and independent distribution (i.i.d.). Spatial prediction problems can be further categorized into *spatial classification* for nominal (i.e., categorical) target variables and *spatial regression* for numeric target variables.

Challenges: The unique challenges of spatial and spatiotemporal prediction come from the special characteristics of spatial and spatiotemporal data, which include spatial and temporal autocorrelation, spatial heterogeneity and temporal non-stationarity, as well as the multi-scale effect. These unique characteristics violate the common assumption in many traditional prediction techniques that samples follow an identical and independent distribution (i.i.d.). Simply applying traditional prediction techniques without incorporating these unique characteristics may produce hypotheses or models that are inaccurate or inconsistent with the data set.

Statistical foundations: Spatial and spatiotemporal prediction techniques are developed based on spatial and spatiotemporal statistics, including spatial and temporal autocorrelation, spatial heterogeneity, temporal non-stationarity, and multiple areal unit problems (MAUP) (see Section 2.2).

Computational approaches: The following subsections summarize common spatial and spatiotemporal prediction approaches for different data types. We further categorize these approaches according to the challenges that they address, including spatial and spatiotemporal autocorrelation, spatial heterogeneity, spatial multi-scale effect, and temporal non-stationarity, and introduce each category separately below.

Spatial Autocorrelation or Dependency

According to Tobler's first law of geography [20], "everything is related to everything else, but near things are more related than distant things." The spatial autocorrelation effect tells us that spatial samples are not statistically independent, and nearby samples tend to resemble each other. There are different ways to incorporate the effect of spatial autocorrelation or dependency into predictive models, including spatial feature creation, explicit model structure modification, and spatial regularization in objective functions.

Spatial feature creation: The main idea is to create new features that incorporate spatial contextual (neighborhood) information. Spatial features can be generated directly from spatial aggregation [21], indirectly from multi-relationship (or spatial association) rules between spatial entities [22, 23, 24], or from spatial transformation of raw features [25]. After spatial features are generated, they can be fed into a general prediction model. One advantage of this approach is that it could utilize many existing predictive models without significant modification. However, spatial feature creation in preprocessing phase is often application specific and time consuming.

Spatial interpolation: Given observations of a variable at a set of locations (point reference data), spatial interpolation aims to measure the variable value at an unsampled location [26]. These techniques are broadly classified into three categories: geostatistical, non-geosatistical and some combined approaches. Among the non-geostatistical approaches, the nearest neighbors, inverse distance weighting, etc. are the mostly used techniques in literature. *Kriging* is the most widely used geo-statistical interpolation technique, which represents a family of generalized least-squares regression based interpolation techniques [27]. *Kriging* can be broadly classified into two categories: univariate (only variable to be predicted) and multivariate (there are some *covariates*, also called explanatory variables). Unlike the non-geostatistical or traditional interpolation techniques, this estimator considers both the distance, as well as the degree of variation between the sampled and unsampled locations for the random variable estimation. Among the univariate kriging methods, the *simple kriging*, *ordinary kriging* and in multivariate scenario, the *ordinary co-kriging*, *universal kriging* and *kriging with external drift* are the most popular and widely used technique in the study of spatial interpolation [26, 28]. However, the *kriging* suffers from some acute shortcomings of assuming the isotopic nature of the random variables.

Markov Random Field (MRF): MRF [29] is a widely used model in image classification problems. It assumes that the class label of one pixel only depends on the class labels of its predefined neighbors (also called Markov property). In spatial classification problem, MRF is often integrated with other non-spatial classifiers to incorporate the spatial autocorrelation effect. For example, MRF has been integrated with maximum likelihood classifiers (MLC) to create Markov Random Field (MRF)-based Bayesian classifiers [30], in order to avoid salt-and-pepper noise in prediction [31]. Another example is the model of Support Vector Random Fields [32].

Spatial Autoregressive Model (SAR): In the spatial autoregression model, the spatial dependencies of the error term, or, the dependent variable, are directly modeled in the regression equation [33]. If the dependent values y_i are related to each other, then the regression equation can be modified as $y = \rho W y + X\beta + \epsilon$, where W is the neighborhood relationship contiguity matrix and ρ is a parameter that reflects the strength of the spatial dependencies between the elements of the dependent variable. For spatial classification problems, logistic transformation can be applied to SAR model for binary classes.

Conditional autoregressive model (CAR): In the conditional autoregressive model [29], the spatial autocorrelation effect is explicitly modeled by the conditional probability of the observation of a location given observations of neighbors. CAR is essentially a Markov random field. It is often used as a spatial term in Bayesian hierarchical models.

Spatial accuracy objective function: In traditional classification problems, the objective function (or loss function) often measures the zero-one loss on each sample, no matter how far the predicted class is from the location of the actuals. For example, in bird nest location prediction problem on a rasterized spatial field, a cell's predicted class (e.g., bird nest) is either correct or incorrect. However, if a cell mistakenly predicted as the bird nest class is very close to an actual bird nest cell, the prediction accuracy should not be considered as zero. Thus, spatial accuracy [34, 35] has been proposed to measure not only how accurate each cell is predicted itself but also how far it is from an actual class locations. A case study has shown that learning models based on proposed objective function produce better accuracy in bird nest location prediction problem. Spatial objective function has also been proposed in active learning [36], in which the cost of additional label not only consider accuracy but also travel cost between locations to be labeled.

Spatial Heterogeneity

Spatial heterogeneity describes the fact that samples often do not follow an identical distribution in the entire space due to varying geographic features. Thus, a global model for the entire space fails to capture the varying relationships between features and the target variable in different sub-region. The problem is essentially the multi-task learning problem, but a key challenge is how to identify different tasks (or regional or local models). Several approaches have been proposed to learn local or regional models. Some approaches first partition the space into homogeneous regions and learn a local model in each region. Others learn local models at each location but add spatial constraint that nearby models have similar parameters.

Geographically Weighted Regression (GWR): One limitation of the spatial autoregressive model (SAR) is that, it does not account for the underlying spatial heterogeneity that is natural in the geographic space. Thus, in a SAR model, coefficients β of covariates and the error term ϵ are assumed to be uniform throughout the entire geographic space. One proposed method to account for spatial variation in model parameters and errors is Geographically Weighted Regression [37]. The regression equation of GWR is $y = X\beta(s) + \epsilon(s)$, where $\beta(s)$ and $\epsilon(s)$ represent the spatially parameters and the errors, respectively. GWR has the same structure as standard linear regression, with the exception that the parameters are spatially varying. It also assumes that samples at nearby locations have higher influence on the parameter estimation of a current location. Recently, a multi-model regression approach is proposed to learn a regression model at each location but regularize the parameters to maintain spatial smoothness of parameters at neighboring locations [38].

Multi-scale Effect

One main challenge in spatial prediction is the Multiple Area Unit Problem (MAUP), which means that analysis results will vary with different choices of spatial scales. For example, a predictive model that is effective at the county level may perform poorly at states level. Recently, a computation technique has been proposed to learn a predict models from different candidate spatial scales or granularity [22].

Spatiotemporal Autocorrelation

Approaches that address the spatiotemporal autocorrelation are often extensions of previously introduced models that address spatial autocorrelation effect by further considering the time dimension. For example, SpatioTemporal Autoregressive Regression (STAR) model [6] extends SAR by further modeling temporal or spatiotemporal dependency across variables at different locations. Spatiotemporal Kriging [80] generalizes spatial kriging with a spatiotemporal covariance matrix and variograms. It can be used to make predictions from incomplete and noisy spatiotemporal data. *Spatiotemporal relational probability trees and forests* [111] extend decision tree classifiers with tree node tests on spatial properties on objects and random field as well as temporal changes. To model spatiotemporal events such as disease counts in different states at a sequence of times, *Bayesian hierarchical models* are often used, which incorporate the spatial and temporal autocorrelation effects in explicit terms.

Temporal Non-stationarity

Hierarchical dynamic spatiotemporal models (DSTMs) [80], as the name suggests, aim to model spatiotemporal processes dynamically with a Bayesian hierarchical framework. There are three levels of models in the hierarchy: a data model on the top, a process model in the middle, and a parameter model at the bottom. A data model represents the conditional dependency of (actual or potential) observations on the underlying hidden process with latent variables. A process model captures the spatiotemporal dependency within the process model. A parameter model characterizes the prior distributions of model parameters. DSTMs have been widely used in climate science and environment science, e.g., for simulating population growth or atmospheric and oceanic processes. For model inference, Kalman filter can be used under the assumption of linear and Gaussian models.

Prediction for Moving Objects

Mining moving object data such as GPS trajectories and check-in histories has become increasingly important. Due to space limit, we briefly discuss some representative techniques for three main problems: trajectory classification, location prediction and location recommendation.

Trajectory classification: This problem aims to predict the class of trajectories. Unlike spatial classification problems for spatial point locations, trajectory classification can utilize the order of locations visited by moving objects. An approach has been proposed that uses frequent sequential patterns within trajectories for classification [112].

Location prediction: Given historical locations of a moving object (e.g., GPS trajectories, check-in histories), the location prediction problem aims to forecast the next place that the object will visit. Various approaches have been proposed [113, 114, 115]. The main idea is to identify the frequent location sequences visited by moving objects, and then next location can be predicted by matching the current sequence with historical sequences. Social, temporal, and semantic information can also be incorporated to improve prediction accuracy. Some other approaches use Hidden Markov Model to capture the transition between different locations. Supervised approaches have also been used.

Location recommendation: Location recommendation [116, 117, 118, 119, 120] aims to suggest potentially interesting locations to visitors. Sometimes, it is considered as a special location prediction problem which also utilizes location histories of other moving objects. Several factors are often considered for ranking candidate locations, such as local popularity and user interests. Different factors can be simultaneously incorporated via generative models such as Latent Dirichlet allocation (LDA) and probabilistic matrix factorization techniques.

2.3.4 Spatial and Spatiotemporal Partitioning (Clustering) and Summarization

Problem definition: Spatial partitioning aims to divide spatial items (e.g., vector objects, lattice cells) into groups such that items within the same group have high proximity. Spatial partitioning is often called *spatial clustering.* We use the name 'spatial partitioning' due to the unique nature of spatial data, i.e., grouping spatial items also means partitioning the underlying space. Similarly, *spatiotemporal partitioning*, or *spatiotemporal clustering*, aims to group similar spatioteompral data items, and thus partition the underlying space and time. After spatial or spatiotemporal partitioning, one often needs to find a compact representation of items in each partition, e.g., aggregated statistics or representative objects. This process is further called *spatial or spatiotemporal summarization.*

Challenges: The challenges of spatial and spatiotemporal partitioning come from three aspects. First, patterns of spatial partitions in real world datasets can be of various shapes and sizes, and are often mixed with noise and outliers. Second, relationships between spatial and spatiotemporal data items (e.g., polygons, trajectories) are more complicated than traditional non-spatial data. Third, there is a trade-off between quality of partitions and computational efficiency, especially for large datasets.

Computational approaches: Common spatial and spatiotemporal partitioning approaches are summarized in below according to the input data types.

Spatial partitioning (clustering)

Spatial and spatiotemporal partitioning approaches can be categorized by input data types, including spatial points, spatial time series, trajectories, spatial polygons, raster images, raster time series, spatial networks, and spatiotemporal points, etc.

Spatial point partitioning (clustering): The goal is to partition two dimensional points into clusters in Euclidean space. Approaches can be categorized into global methods, hierarchical methods, and density-based methods according to the underlying assumptions on the characteristics of clusters [121]. Global methods assume clusters to have 'compact' or globular shapes, and thus minimize the total distance from points to their cluster centers. These methods include K-means, K-Medoids, EM algorithm, CLIQUE, BIRCH, and CLARANS [59]. Hierarchical methods [59] form clusters hierarchically in a top-down or bottom up manner, and are robust to outliers since outliers are often easily separated out. Chameleon [122] is a graph-based hierarchical clustering method that first creates a sparse k nearest neighbor graph, then partitions the graph into small clusters, and hierarchically merges small clusters whose properties stay mostly unchanged after merging. Density-based methods such as DB-Scan [123] assume clusters to contain dense points and can have arbitrary shapes. When the density of points varies across space, the similarity measure of *shared nearest neighbors* [124] can be used. Voronoi diagram [125] is another space partitioning technique that is widely used in applications of location based service. Given a set of spatial points in Euclidean space, a Voronoi diagram partitions the space into cells according to the nearest spatial points.

Spatial polygon clustering: Spatial polygon clustering is more challenging than point clustering due to the complexity of distance measures between polygons. Distance measures on polygons can be defined based on dissimilarities on spatial attribute (e.g., Hausdorff distance, ratio of overlap, extent, direction, topology, etc.) as well as non-spatial attributes [126, 127]. Based on these distance measures, traditional point clustering algorithms such as K-means, CLARANS, and shared nearest neighbor algorithm can be applied.

Spatial areal data partitioning: Spatial areal data partitioning has been extensively studied for image segmentation tasks. The goal is to partition areal data (e.g., images) into regions that are homogeneous in non-spatial attributes (e.g., color or grey tone, texture, etc.) while maintaining spatial continuity (without small holes). Similar to spatial point clustering, there is no uniform solution. Common approaches can be categorized into non-spatial attribute guided spatial clustering, single, centroid, or hybrid linkage region growing schemes, and split-and-merge scheme. More details can be found in a survey on image segmentation [128].

Spatial network partitioning: Spatial network partitioning (clustering) is important in many applications such as transportation and VLSI design. Network Voronoi diagram is a simple method to partition spatial network based on common closest interesting nodes (e.g., service centers). Recently, a connectivity constraint network Voronoi diagram (CCNVD) has been proposed to add capacity constraint to each partition while maintaining spatial continuity [129]. METIS [130] provides a set of scalable graph partitioning algorithms, which have shown high partition quality and computational efficiency.

Spatiotemporal partitioning (clustering)

Spatiotemporal event partitioning (clustering): Most methods for 2-D spatial point clustering [121] can be easily generalized to 3-D spatiotemporal event data [131]. For example, ST-DBSCAN [132] is a spatiotemporal extension of the density-based spatial clustering method DBSCAN. ST-GRID [133] is another example that extends grid-based spatial clustering methods into 3-D grids.

Spatial time-series partitioning (clustering): Spatial time series clustering aims to divide the space into regions such that the similarity between time series within the same region is maximized. Global partitioning methods such as K Means, K Medoids, and EM as well as the hierarchical methods can be applied. Common (dis)similarity measures include Euclidean distance, Pearson's correlation, dynamic time warping (DTW) distance, etc. More details can be found in a recent survey [134]. However, due to the high dimensionality of spatial time series, density-based approaches and graph-based approaches are often not used. When computing similarities between spatial time series, a filter-and-refine approach [65] can be used to avoid redundant computation.

Trajectory partitioning: Trajectory partitioning approaches can be categorized by their objectives, namely trajectory grouping, flock pattern detection, and trajectory segmentation. Trajectory grouping aims to partition trajectories into groups according to their similarity. There are mainly two types of approaches, i.e., distance-based and frequency-based. The *density based approaches* [135, 136, 137] first break trajectories into small segments and apply distance-based clustering algorithms similar to K-means or DBSCAN to connect dense areas of segments. The *frequency based approach* [138] uses association rule mining [78] algorithms to identify subsections of trajectories which have high frequencies (also called high 'support').

Spatial and spatiotemporal summarization

Data summarization aims to find compact representation of a data set [139]. It is important for data compression as well as for making pattern analysis more convenient. Summarization can be done on classical data, spatial data, as well as spatiotemporal data.

Classical data summarization: Classical data can be summarized with aggregation statistics such as count, mean, median, etc. Many modern database systems provide query support for this operation, e.g., 'Group by' operator in SQL.

Spatial data summarization: Spatial data summarization is more difficult than classical data summarization due to its non-numeric nature. For Euclidean space, the task can be done by first conducting spatial partitioning and then identifying representative spatial objects. For example, spatial data can be summarized with the centroids or medoids computed from K Means or K Medoids algorithms. For network space, especially for spatial network activities, summarization can be done by identifying several primary routes that cover those activities as much as possible. A K-Main-Routes (KMR) algorithm [140] has been proposed to efficiently compute such routes to summarize spatial network activities. To reduce the computational cost, the KMR algorithm uses network Voronoi diagrams, divide and conquer, and pruning techniques.

Spatiotemporal data summarization: For spatial time series data, summarization can be done by removing spatial and temporal redundancy due to the effect of autocorrelation. A family of such algorithms has been used to summarize traffic data streams [141]. Similarly, the centroids from K Means can also be used to summarize spatial time series. For trajectory data, especially spatial network trajectories, summarization is more challenging due to the huge cost of similarity computation. A recent approach summarizes network trajectories into k primary corridors [142, 143]. The work proposes efficient algorithms to reduce the huge cost for network trajectory distance computation.

2.3.5 Spatial and Spatiotemporal Hotspot Detection

Problem definition: Given a set of spatial objects (e.g. points) in a study area, the problem of *spatial hotspot detection* aims to find regions where the number of objects is unexpectedly or anomalously high. Spatial hotspot detection is different from spatial partitioning or clustering, since spatial hotspots are a special kind of clusters whose intensity is “significantly” higher than the outside. *Spatiotemporal hotspots* can be seen as a generalization of spatial hotspots with a specified time window.

Challenges: Spatial and spatiotemporal hotspot detection is a challenging task since the location, size and shape of a hotspot is unknown beforehand. In addition, the number of hotspots in a study area is often not known either. Moreover, ‘false’ hotspots that aggregate events only by chance should often be avoided, since these false hotspots impede proper response by authorities (e.g., wasting police resources). Thus, it is often important to test the statistical significance of candidate spatial or spatiotemporal hotspots.

Statistical foundation: Spatial (or spatiotemporal) scan statistics [14, 144] (also discussed in Section 3.1) are used to detect statistically significant hotspots from spatial (or spatiotemporal) datasets. It uses a window (or cylinder) to scan the space (or space-time) for candidate hotspots and performs hypothesis testing. The null hypothesis states that the spatial (or spatiotemporal) points are completely spatially random (a homogeneous Poisson point process). The alternative hypothesis states that the points inside of the window (or cylinder) has higher intensity of points than outside. A test statistic called log likelihood ratio is computed for each candidate hotspot and the candidate with the highest likelihood ratio can be further evaluated by its significance value (i.e., p-value).

Computational approaches: The following subsections summarize common spatial and spatiotemporal hotspot detection approaches by different input data types.

Spatial Hotspot from Spatial Point Pattern

Spatial partitioning approaches: Spatial point partitioning or clustering methods (Section 4.4.1) can be used to identify candidate hotspot patterns. After this, statistical tools may be used to evaluate the statistical significance of candidate patterns. Many of these methods have been implemented in CrimeStat, a software package for crime hotspot analysis [145].

Spatial scan statistics based approaches: These approaches use a window with varying sizes to scan the 2-D plane and identifies the candidate window with the highest likelihood ratio. Statistical significance (p-value) is computed for this candidate based on Monte Carlo simulation. Scanning windows with different shapes, including circular, elliptical, as well as ring-shaped, have been proposed together with efficient computational pruning strategies [144, 146, 147, 148]. SaTScan [144] is a popular spatial scan statistics tool in epidemiology to analyze circular or elliptical hotspots.

Kernel Density Estimation: Kernel density estimation (KDE) [149] identifies spatial hotspots via a density map of point events. It first creates a grid over the study area and uses a kernel function with a user-defined radius (bandwidth) on each point to estimate the density of points on centers of grid cells. A subset of grid cells with high density are returned as spatial hotspots.

Spatial Hotspot from Areal Model

Local Indicators of Spatial Association: Local indicators of spatial association (LISA) [150, 151] is a set of local spatial autocorrelation statistics, including local Moran's I, Geary's C or Ord Gi and Gi* functions. It differs from global spatial autocorrelation in that the statistics are computed within the neighborhood of a location. For example, a high local Moran's I indicates that values of the current location as well as its neighbors' are both extremely high (or low) compared to values at other locations, and thus the neighborhood is a spatial hotspot (or "cold spot").

Spatiotemporal Hotspot Detection

Hot routes from spatial network trajectories: Hot routes detection from spatial network trajectories aims to detect network paths with high density [135] or frequency of trajectories [138]. Other approaches include organizing police patrol routes [152], main streets [153], and clumping [154], etc.

Spatiotemporal Scan Statistics based approaches: Two types of spatiotemporal hotspots can be detected by spatiotemporal scan statistics: "persistent" spatiotemporal hotspots and “emerging” spatiotemporal hotspots. A “persistent” spatiotemporal hotspot is a region where the rate of increase in observations is a high and almost constant value over time. Thus, approaches to detect a persistent spatiotemporal hotspot involves counting observations in each time interval [144]. An “emerging” spatiotemporal hotspot is a region where the rate of observations monotonically increases over time [155, 147]. This kind of spatiotemporal hotspot occurs when an outbreak emerges causing a sudden increase in the number observations. Tools for the detection of emerging spatiotemporal hotspots use spatial scan statistics to identify changes in expectation over time [156].

2.3.6 Spatiotemporal Change

What are Spatiotemporal Changes and Change Footprints

Although the single term “change” is used to name the spatiotemporal change footprint patterns in different applications, the underlying phenomena may differ significantly. This section briefly summarizes the main ways a change may be defined and detected in spatiotemporal data [51].

Change in Statistical Parameter: In this case, the data is assumed to follow a certain distribution and the change is defined as a shift in this statistical distribution. For example, in statistical quality control, a change in the mean or variance of the sensor readings is used to detect a fault.

Change in Actual Value: Here, change is modeled as the difference between a data value and its spatial or temporal neighborhood. For example, in a one-dimensional continuous function, the magnitude of change can be characterized by the derivative function, while on a two-dimensional surface, it can be characterized by the gradient magnitude.

Change in Models Fitted to Data: This type of change is identified when a number of models are fitted to the data and one or more of the models exhibits a change (e.g., a discontinuity between consecutive linear functions) [157].

Common Approaches

This section follows the taxonomy of spatiotemporal change footprint patterns as proposed in [51]. In this taxonomy, spatiotemporal change footprints are classified along two dimensions: temporal and spatial. Temporal footprints are classified into four categories: single snapshot, set of snapshots, point in a long series, and interval in a long series. Single snapshot refers to a purely spatial change that does not have a temporal context. A set of snapshots indicates a change between two or more snapshots of the same spatial field, e.g., satellite images of the same region.

Spatial footprints can be classified as raster footprints or vector footprints. Vector footprints are further classified into four categories: point(s), line(s), polygon(s), and network footprint patterns. Raster footprints are classified based on the scale of the pattern, namely, local, focal, or zonal patterns. This classification describes the scale of the change operation of a given phenomenon in the spatial raster field [158]. Local patterns are patterns in which change at a given location depends only on attributes at this location. Focal patterns are patterns in which change in a location depends on attributes in that location and its assumed neighborhood. Zonal patterns define change using an aggregation of location values in a region.

Spatiotemporal Change Patterns with Raster-based Spatial Footprint: This includes patterns of *spatial changes between snapshots.* In remote sensing, detecting changes between satellite images can help identify land cover change due to human activity, natural disasters, or climate change [159, 160, 161]. Given two geographically aligned raster images, this problem aims to find a collection of pixels that have significant changes between the two images [162]. This pattern is classified as a local change between snapshots since the change at a given pixel is assumed to be independent of changes at other pixels. Alternative definitions have assumed that a change at a pixel also depends on its neighborhoods [163]. For example, the pixel values in each block may be assumed to follow a Gaussian distribution [164]. We refer to this type of change footprint pattern as a focal spatial change between snapshots. Researchers in remote sensing and image processing have also tried to apply image change detection to objects instead of pixels [165, 166, 167], yielding zonal spatial change patterns between snapshots.

A well-known technique for detecting a local change footprint is simple differencing. The technique starts by calculating the differences between the corresponding pixels intensities in the two images. A change at a pixel is flagged if the difference at the pixel exceeds a certain threshold. Alternative approaches have also been proposed to discover focal change footprints between images. For example, the block-based density ratio test detects change based on a group of pixels, known as a block [168, 169]. Object-based approaches in remote sensing [170, 167, 171] employ image segmentation techniques to partition temporal snapshots of images into homogeneous objects [172] and then classify object pairs in the two temporal snapshots of images into no change or change classes.

Spatiotemporal Change Patterns with Vector-based Spatial Footprint: This includes the *Spatiotemporal Volume Change Footprint* pattern. This pattern represents a change process occurring in a spatial region (a polygon) during a time interval. For example, an outbreak event of a disease can be defined as an increase in disease reports in a certain region during a certain time window up to the current time. Change patterns known to have an spatiotemporal volume footprint include the spatiotemporal scan statistics [173, 174], a generalization of the spatial scan statistic, and emerging spatiotemporal clusters defined by [156].

2.4 Research Trend and Future Research Needs

Most current research in spatial and spatiotemporal data mining uses Euclidean space, which often assumes isotropic property and symmetric neighborhoods. However, in many real world applications, the underlying space is network space, such as river networks and road networks [175, 176, 140]. One of the main challenges in spatial and spatiotemporal network data mining is to account for the network structure in the dataset. For example, in anomaly detection, spatial techniques do not consider the spatial network structure of the dataset, that is, they may not be able to model graph properties such as one-ways, connectivity, left-turns, *etc.* The network structure often violates the isotropic property and symmetry of neighborhoods, and instead, requires asymmetric neighborhood and directionality of neighborhood relationship (e.g., network flow direction).

Recently, some cutting edge research has been conducted in the spatial network statistics and data mining [18]. For example, several spatial network statistical methods have been developed, e.g., network K function and network spatial autocorrelation. Several spatial analysis methods have also been generalized to the network space, such as network point cluster analysis and clumping method, network point density estimation, network spatial interpolation (Kriging), as well as network Huff model. Due to the nature of spatial network space as distinct from Euclidean space, these statistics and analysis often rely on advanced spatial network computational techniques [18].

We believe more spatial and spatiotemporal data mining research is still needed in the network space. First, though several spatial statistics and data mining techniques have been generalized to the network space, few spatiotemporal network statistics and data mining have been developed, and the vast majority of research is still in the Euclidean space. Future research is needed to develop more spatial network statistics, such as spatial network scan statistics, spatial network random field model, as well as spatiotemporal autoregressive models for networks. Furthermore, phenomena observed on spatiotemporal networks need to be interpreted in an appropriate frame of reference to prevent a mismatch between the nature of the observed phenomena and the mining algorithm. For instance, moving objects on a spatiotemporal network need to be studied from a traveler's perspective, *i.e.*, the Lagrangian frame of reference [177, 178, 179] instead of a snapshot view. This is because a traveler moving along a chosen path in a spatiotemporal network would experience a road-segment (and its properties such as fuel efficiency, travel-time *etc.*) for the time at which he/she arrives at that segment, which may be distinct from the original departure-time at the start of the journey. These unique requirements (non-isotropy and Lagrangian reference frame) call for novel spatiotemporal statistical foundations [175] as well as new computational approaches for spatiotemporal network data mining.

Another future research need is to develope spatiotemporal graph big data platforms, motivated by the upcoming rich spatiotemporal network data collected from vehicles. Modern vehicles have rich instrumentation to measure hundreds of attributes at high frequency and are generating big data (Exabyte [180]). This vehicle measurement big data (VMBD) consist of a collection of trips on a transportation graph such as a road map annotated with several measurements of engine sub-systems. Collecting and analyzing VMBD during real-world driving conditions can aid in understanding the underlying factors which govern real world fuel inefficiencies or high greenhouse gas (GHG) emissions [181]. Current relevant big data platforms for spatial and spatiotemporal data mining include ESRI GIS Tools for Hadoop [182, 183], Hadoop GIS [184], *etc.* These provide distributed systems for geometric-data (e.g., lines, points and polygons) including geometric indexing and partitioning methods such as R-tree, R+-tree, or Quad tree. Recently, SpatialHadoop has been developed [185]. SpatialHadoop embeds geometric notions in language, visualization, storage, MapReduce, and operations layers. However, spatio-temporal graphs (STGs) violate the core assumptions of current spatial big data platforms that the geometric concepts are adequate for conveniently representing STG analytics operations and for partition data for load-balancing. STGs also violate core assumptions underlying graph analytics software (e.g., Giraph [186], GraphLab [187] and Pregel [188]) that traditional location-unaware graphs are adequate for conveniently representing STG analytics operations and for partition data for load-balancing. Therefore, novel spatiotemporal graph big data platforms are needed. Several challenges should be addressed, e.g., spatiotemporal graph big data requires novel distributed file systems (DFS) to partition the graph, and a novel programming model is still needed to support abstract data types and fundamental STG operations, *etc.*

2.5 Summary

This paper provides an over view of current research in the field of spatial and spatiotemporal (SST) data mining from a computational perspective. SST data mining has broad application domains including ecology and environmental management, public safety, transportation, earth science, epidemiology, and climatology. However, the complexity of SST data and intrinsic SST relationships limit the usefulness of conventional data mining techniques. We provide a taxonomy of different SST data types and underlying statistics. We also review common SST data mining techniques organized by major output pattern families: SST outlier, coupling and tele-coupling, prediction, partitioning and summarization, hotspots, and change patterns. Finally, we discuss the recent research trends and future research needs.

Chapter 3

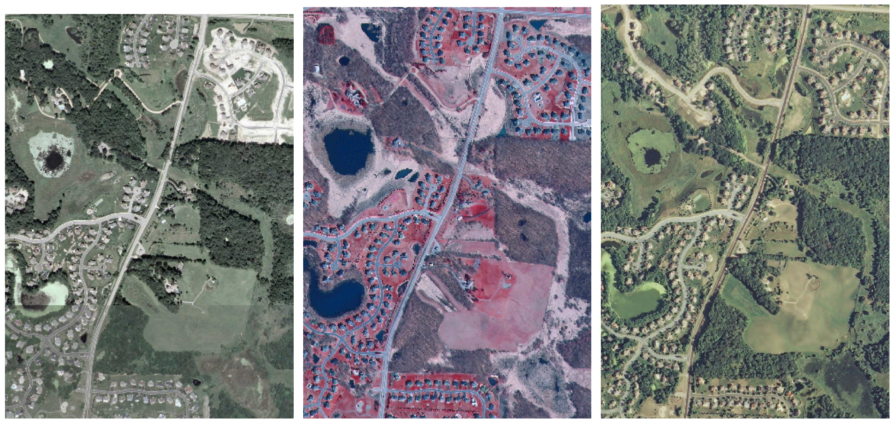
Focal-Test-Based Spatial Decision Tree

3.1 Introduction

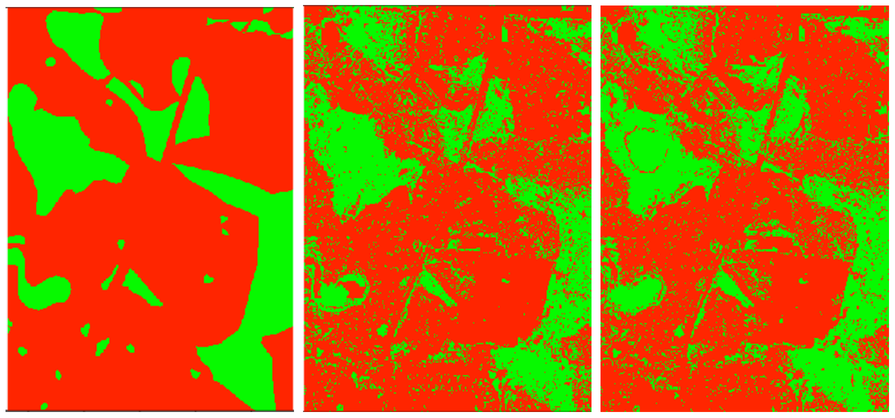
Given a spatial raster framework, as well as training and test sets, the spatial decision tree learning (SDTL) problem aims to find a decision tree model that minimizes classification errors as well as salt-and-pepper noise. Figure 3.1 is a motivation example from a real world wetland mapping application. Input features are bands of three aerial photos (Figure 3.1(a-c)). Classification results by two existing decision tree classifiers [189], [190] are shown in Figure 3.1(e) and 3.1(f) respectively. Both predicted maps exhibit poor appearance accuracy with high levels of salt-and-pepper noise, when compared with ground truth classes (Figure 3.1(d)).

Societal Applications: The SDTL problem has many applications. In the field of remote sensing, a large amount of images of the earth surface are collected (e.g., NASA collects about 5TB data per day). SDTL can be used to classify remote sensing images into different land cover types [191]. For example, in wetland mapping [192] [193], explanatory features, including spectral bands (e.g., red, green, blue, near-infrared) from remote sensors, are used to map land surface into wetland areas and dryland areas. Land cover classification is important for climate change research[194], natural resource management [195] [196], and disaster management [197], etc. In medical image processing, SDTL can help in lesion classification and brain tissue segmentation [198]

[199] on MRI images. It can also be used for galaxy classification [200] in astronomy and semi-conductor inspection [201] in materials science.



(a) aerial photo in 2003 (b) aerial photo in 2005 (c) aerial photo in 2008



(d) ground truth classes (red for dry land, green for wetland) (e) prediction of a C4.5 decision tree (f) prediction of a spatial-information-gain-based decision tree

Figure 3.1: Real world problem example

Challenges: A key challenge in the SDTL problem is that learning samples show spatial autocorrelation in class labels. For example, the ground truth class labels in Figure 3.1(d) show strong spatial autocorrelation due to the phenomenon of "patches" (i.e., regions of the same class tend to be contiguous). Testing only local feature information in decision nodes results in salt-and-pepper noise, i.e., locations or pixels whose

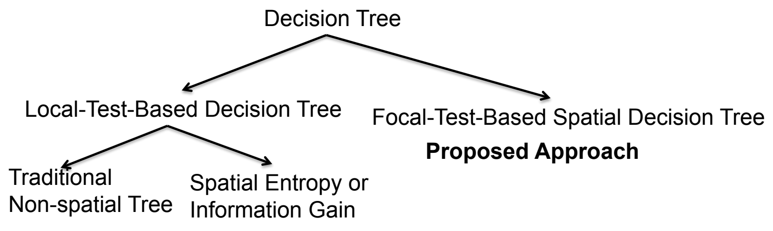

Decision Tree
Local-Test-Based Decision Tree
Focal-Test-Based Spatial Decision Tree
Proposed Approach
Traditional Non-spatial Tree
Spatial Entropy or Information Gain

Figure 3.2: Related work classification

class labels are different from those of their neighbors, as illustrated in Figure 3.1(e). However, incorporating focal (i.e., neighborhood) information increases both the number and the complexity of candidate tree node tests. Instead of simple linear scanning and thresholding on one dimensional feature values, tree node tests must incorporate the spatial relationships of various neighborhood sizes. Thus, SDTL problem is also computationally challenging.

Related work and limitations: Figure 4.2 presents a classification of related work. Traditional decision tree algorithms include ID3 [202], C4.5 [189] and CART [203]). These classifiers follow the classic assumption that learning samples are independently and identically distributed. This assumption does not hold for spatial data and leads to salt-and-pepper noise in predictions. A second category are the spatial entropy or information gain based decision tree classifiers [204] [205] [206] [190]. These newer methods use spatial autocorrelation level as well as information gain to select candidate tree node tests. While they do a better job if there exists some feature that favors spatial autocorrelation but does not provide the largest information gain in one tree node test, they still relies on local testing of information by tree nodes. Thus if all the candidate tests have poor spatial autocorrelation, this type of decision tree will still select one of them, resulting in salt-and-pepper noise. This means neither approach adequately accounts for spatial autocorrelation in the prediction phase.

To address this limitation, we recently defined a focal-test-based spatial decision tree (FTSDT) model [207], whereby the tree traversal direction of a learning sample is based on not only local but also focal (neighborhood) properties of features. We proposed FTSDT learning algorithms and evaluated the classification performance of the proposed approach on real world remote sensing datasets. We also extended the basic

FTSDT algorithm in a journal paper [208] with the following additional contributions:

1. We add a new design decision in the FTSDT model to allow focal function computation with adaptive neighborhoods (i.e., FTSDT-adaptive). Compared with previous FTSDT with fixed neighborhoods (i.e., FTSDT-fixed), the new design decision can adjust the neighborhood shape to avoid over-smoothing in wedge-shaped areas.

2. We characterize the computational structure of the FTSDT learning algorithm and confirm that the computational bottleneck is a vast number of focal function computations. We design a refined algorithm (FTSDT-Refined) that reuses focal values across candidate thresholds and prove its correctness.

3. We also provide cost models of our previous baseline algorithm and our refined algorithm, and show that the refined algorithm improves computational scalability.

4. We compare the classification performance of FTSDT-adaptive with FTSDT-fixed as well as LTDT on real world datasets. Results show that FTSDT-adaptive improves classification accuracy of FTSDT-fixed and LTDT.

5. We also conduct experimental evaluations of computational performance on real world datasets with various parameter settings. Experiment results show that our refined algorithm significantly reduces computational time cost.

Scope: This work focuses on incorporating focal tests inside a decision tree for raster data classification. Other classification algorithms such as Markov Random Field [209], Spatial Autoregression (SAR) model [210], logistic regression, neural network, etc., are beyond the scope. In addition, for simplicity, this work only considers learning samples with continuous features. The case of discrete features is not addressed.

Outline: The chapter is organized as follows: Section 3.1 introduces basic concepts and formalizes the SDTL problem; Section 3.2 presents our FTSDT learning algorithm, especially a new design decision to allow focal function with adaptive neighborhoods. Section 3.3 describes computational optimization, and the refined algorithm design with theoretical analysis. Computational and classification performance of the proposed algorithms are evaluated in Section 3.4. Section 3.6 discusses some other relevant techniques in the literature. Section 3.7 concludes the paper with future work.

3.2 Basic Concepts and Problem Formulation

This section introduces basic concepts and formally defines the spatial decision tree learning problem.

3.2.1 Basic Concepts

Spatial raster framework: A spatial raster framework F is a tessellation of a 2-D plane into a regular grid. On a spatial raster framework, there may exist a set of explanatory feature maps, as well as a class label map. For example, Figure 3.3 shows a spatial raster framework with explanatory features f^1, f^2, ..., f^m, and a class label map c. Each grid cell on the raster framework is a *spatial data sample* (e.g., location i in Figure 3.3). For simplicity, we use the words "sample," "pixel," "location," and "spatial data sample" interchangeably in the remainder of the paper.

Neighborhood relationship: a spatial neighborhood relationship describes the range of dependency between spatial locations. It is commonly represented as a W-matrix, whose element $W_{i,j}$ has a non-zero value when locations i and j are *neighbors*, and a zero value otherwise. For example, in Figure 3.3, the pixel in dark grey has eight neighbors indicated in light grey in a 3-by-3 neighborhood.

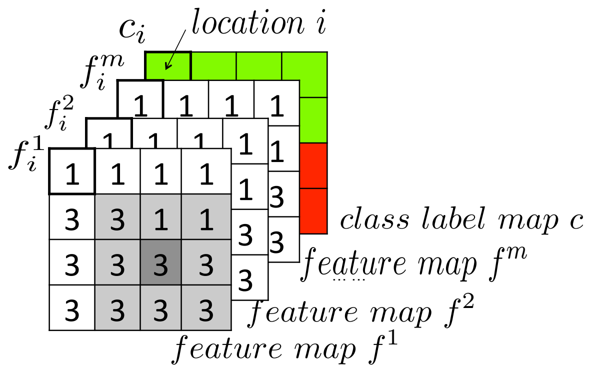


Figure 3.3: Example of a spatial raster framework and a neighborhood relationship

Salt-and-pepper noise: Salt-and-pepper noise is defined as a kind of fat-tail impulse noise whose values are often extreme (e.g., minimum or maximum) [211]. In a predicted class label map, salt-and-pepper noise can be considered as a single pixel (or a small group of contiguous pixels) that is distinct from its (or their) spatial neighborhood. For example, in Figure 3.4(i), the central pixel is salt-and-pepper noise.

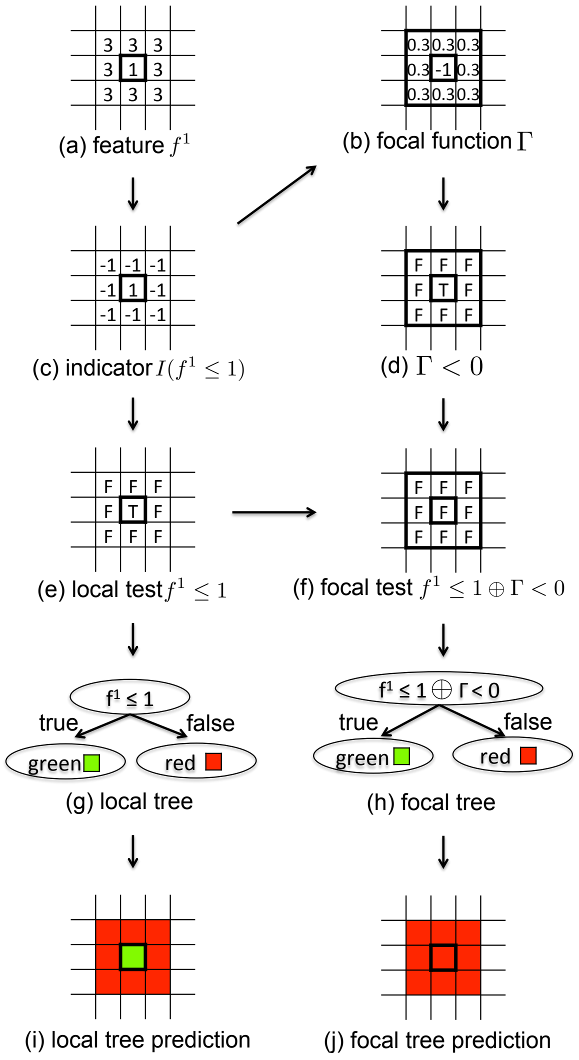

(a) feature f^1
(b) focal function Γ
(c) indicator $I(f^1 \leq 1)$
(d) $\Gamma < 0$
(e) local test $f^1 \leq 1$
(f) focal test $f^1 \leq 1 \oplus \Gamma < 0$
$f^1 \leq 1$
true
false
green
red
(g) local tree
$f^1 \leq 1 \oplus \Gamma < 0$
true
false
green
red
(h) focal tree
(i) local tree prediction
(j) focal tree prediction

Figure 3.4: Comparison of a local test v.s. a focal test, a local-test-based decision tree v.s. a focal-test-based spatial decision tree. ("T" is "true", "F" is "false")

Local test and *indicators*: A *local test* $f^m \leq \delta$ checks the value of feature f^m at a sample's location against a threshold δ. The local test results can be represented as *indicator variable* $I(f^m \leq \delta)$ or simply I, whose value is 1 when $f^m \leq \delta$ is true and -1

Table 3.1: List of symbols and descriptions

Symbols	Descriptions
δ	a local test threshold
$\oplus$	logic operator “xor”, i.e., $0 \oplus 1 = 1$, $1 \oplus 0 = 1$, $0 \oplus 0 = 0$, $1 \oplus 1 = 0$
$W_{i,j}$	neighborhood relationship between location i and location j
f^m, f_i^m	value of feature m, the feature value at location i
$I(f^m \leq \delta)$, I_i	indicator variable of local test $f^m \leq \delta$, the indicator variable at location i
Γ, Γ_i	focal Gamma autocorrelation statistic, the focal Gamma at location i

otherwise. A decision tree whose tree nodes conducts local tests is called a *local-test-based decision tree (LTDT)*. For example, given the feature f^1 shown in Figure 3.4(a), the local test results of $f^1 \leq 1$ and corresponding indicator variables are shown in Figure 3.4(e) and 3.4(c) respectively. The corresponding LTDT and its class predictions with salt-and-pepper noise are shown in Figure 3.4(g) and 3.4(i).

Focal function and *spatial autocorrelation statistic*: A focal function is an aggregate of non-spatial attribute values in the neighborhood of a location. One important kind of focal function is *focal autocorrelation statistic*, which measures the dependency between attribute values of a location and the values of its neighbors. For example, the focal Gamma index [212] on local test indicators is defined as

$$\Gamma_i = \frac{\sum_j W_{i,j} I_i I_j}{\sum_j W_{i,j}},$$

where i and j are locations, $W_{i,j}$ is a W-matrix element, and I_i, I_j are indicator variables of a local test. A negative focal Gamma value (i.e., $\Gamma < 0$) indicates that the current location is potentially salt-and-pepper noise. Figure 3.4(b) shows an example of focal Gamma values computed on indicator variables in Figure 3.4(c) with a 3-by-3 neighborhood. The central location has a negative Gamma because its local test result is different from its neighbors'.

Focal test: A focal test is a test or a combination of tests on attribute values in a neighborhood of a location. For example, $f \leq \delta \oplus \Gamma < 0$, where $\oplus$ is an "xor" logical operator, is a focal test that combines a local test $f \leq \delta$ and the test $\Gamma < 0$. This combined focal test is less prone to salt-and-pepper noise, compared with the local test $f \leq \delta$ only. The reason is that salt-and-pepper noise pixels often have a negative focal gamma index (i.e., $\Gamma < 0$ is true), and their local test results ($f \leq \delta$) are flipped by logical operator $\oplus$ (i.e., "false" xor true becomes "true," and "true" xor true becomes "false"). For instance, the local test result of the central pixel in Figure 3.4(e) is true, but false for its neighborhood; while the focal test result of the same pixel in Figure 3.4(f) is false, and the same as for its neighborhood.

Focal-test-based spatial decision tree (FTSDT): An FTSDT is a tree whose nodes conduct focal tests. An example of FTSDT is in Figure 3.4(h) and its class predictions are in Figure 3.4(j). In our approach, both local tests and focal tests are defined on a single feature. When multiple features exist, the local test or focal test on each feature is considered as a candidate tree node test and the best candidate test is selected for a tree node, similar to the situation of a traditional decision tree.

3.2.2 Problem Definition

Based on the concepts above, the spatial decision tree learning problem is formally defined as follows:

Given:

- A spatial raster framework F
- A spatial neighborhood definition, and its maximum size S_{max}
- Training and test samples drawn from F

Find:

- A decision tree model based on training samples.

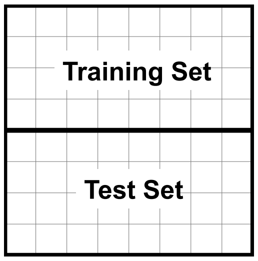
Objective:

- Minimize classification errors as well as salt-and-pepper noise

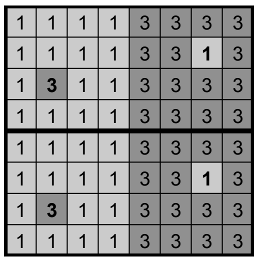
Constraints:

- Training samples form contiguous patches of locations in F

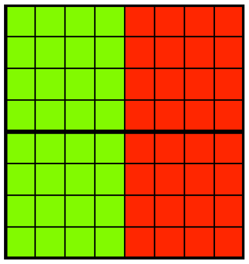
• Spatial autocorrelation exists in class labels



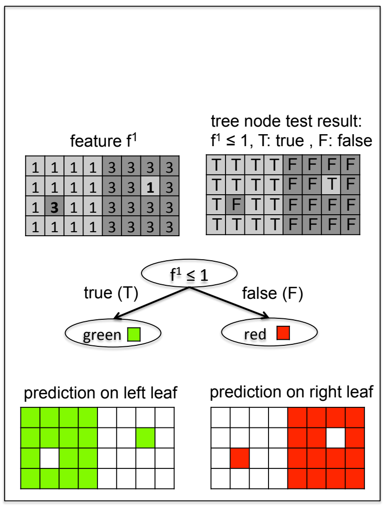
(a) a raster framework



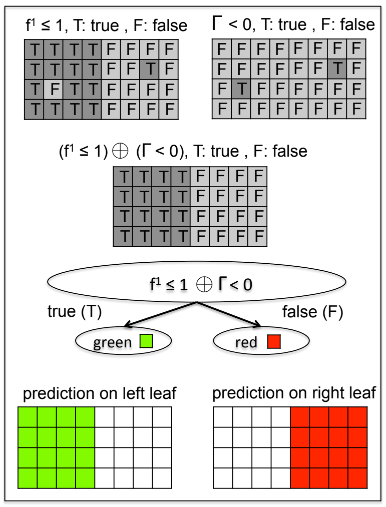
(b) a feature



(c) ground truth class labels



(d) an LTDT and its predictions


f¹ ≤ 1, T: true , F: false
Γ < 0, T: true , F: false
(f¹ ≤ 1) ⊕ (Γ < 0), T: true , F: false
f¹ ≤ 1 ⊕ Γ < 0
true (T)
false (F)
green
red
prediction on left leaf
prediction on right leaf

(e) an FTSDT and its predictions

Figure 3.5: An illustrative problem example (best viewed in color).

Problem description: The output decision tree model can be a local-test-based decision tree or a focal-test-based spatial decision tree, depending on the selected approach. Parameters to be learned from the training set are the tree structure, as well as which feature f, test thresholds δ, and proper neighborhood size s (in the case of FTSDT) to use in each tree node.

Example: Consider the example of problem inputs and outputs in Figure 3.5. The raster spatial framework F, shown in Figure 3.5(a), consists of training pixels on the upper half and test pixels on the lower half. Neighborhood relationship is defined as a 3-by-3 square window. The minimum tree node size is 4. Figure 3.5(b) shows a candidate feature f^1. Figure 3.5(c) shows the ground truth class labels. The output local-test-based traditional decision tree learned from the training set and its predictions with salt-and-pepper noise are shown in Figure 3.5(d). In contrast, the output focal-test-based spatial decision tree and its predictions without salt-and-pepper noise are shown in Figure 3.5(e).

3.3 FTSDT Learning Algorithms

This section describes the baseline FTSDT learning algorithm (i.e., without computational optimization) of the focal-test-based spatial decision tree. The learning algorithm has two phases: a training phase, FTSDT-Train; and a prediction phase, FTSDT-Predict. FTSDT-Train here extends the previous one we proposed in [207] by allowing focal function tests with adaptive neighborhoods to avoid over-smoothing in wedge-shaped areas.

3.3.1 Training Phase

FTSDT-Train (Algorithm 1) learns an FTSDT classifier from training samples. It includes two sub-routines (Node-Split and Focal Function). Similar to traditional C4.5 [189], it is a divide and conquer method with a greedy strategy (i.e., maximize information gain).

Steps 1 to 3 check the stopping criteria. If the training samples are less than the minimum tree node size, or all the class labels are identical, a leaf labeled with the majority class will be returned.

Steps 4 to 13 enumerate through every candidate feature f, every neighborhood size s, and every candidate threshold δ to select the best setting for a model tree node. Candidate thresholds δ are generated from distinct values of feature f in the training samples (steps 8 to 9). Step 10 calls a Node-Split subroutine to split training samples. Step 11 evaluates the corresponding information gain on the column of class labels. Steps 12-13 update the current best candidate test.

Steps 14 to 18 create an internal node with the best test, split the training samples

Algorithm 1 FTSDT-Train(T, S_{max}, N_0, $neighType$)

```
Input:
  • $T$: rows are samples, columns are features (last column as class)
  • $S_{max}$: maximum neighborhood size
  • $N_0$: minimum decision tree node size
  • $neighType$: neighborhood type, 0 for fixed neighborhood, 1 for adaptive neighborhood
Output:
  • root of an FTSDT model
1: let $N$ be number of samples, $F$ be number of features, $c$ be column index of classes; $IG_0 = -\infty$
2: if $N < N_0$ or $T$ same class then
3:    return a leaf node;
4: for each $f \in \{1...F\}$ do
5:    sort rows of $T$ //in ascending order of $f^{th}$ column
6:    for each $s \in \{0...S_{max}\}$ do
7:       for each $i \in \{N_0...(N - N_0)\}$ do
8:          if $T[i][f] < T[i+1][f]$ then
9:             $\delta = T[i][f]$
10:            $\{T_1, T_2\}$ = Node-Split($T$, $f$, $\delta$, $s$, $neighType$);
11:            $IG$ = InformationGain($T[\,][c]$, $T_1[\,][c]$, $T_2[\,][c]$)
12:            if $IG > IG_0$ then
13:               $IG_0 = IG$; $s_0 = s$; $f_0 = f$; $\delta_0 = \delta = T[i][f]$;
14: $I$ = CreateInternalNode($f_0$, $\delta_0$, $s_0$);
15: $\{T_1, T_2\}$ = Node-Split($T$, $f_0$, $\delta_0$, $s_0$, $neighType$);
16: $I.left$ = FTSDT-Train($T_1$, $S_{max}$, $N_0$, $neighType$)
17: $I.right$ = FTSDT-Train($T_2$, $S_{max}$, $N_0$, $neighType$)
18: return $I$
```

into two subsets accordingly, recursively call FTSDT-Train on each subset, and return the internal node.

Node-Split: The Node-Split subroutine (Algorithm 2) splits the training samples into two subsets based on their focal test results, and proceeds as follows:

Step 1 initializes the two subsets as empty sets. Samples with node test results TRUE will be assigned to one subset and samples with test results FALSE will be assigned to the other.

Steps 2 to 11 compute the focal tree node test result of each training sample and

Algorithm 2 Node-Split(T, f, δ, S, $neighType$)

```
Input:
  • $T$: rows as samples, columns as features (last column as class)
  • $f$: a feature index
  • $\delta$: threshold of feature test
  • $S$: neighborhood size
  • $neighType$: neighborhood type, 0 for fixed neighborhood, 1 for adaptive neigh-
    borhood
Output:
  • $\{T_1, T_2\}$: sample subsets with test results true and false respectively
 1: $T_1 = T_2 = \emptyset$
 2: for each $i \in \{1...N\}$ do
 3:     indicators $I[i] = I(T[i][f] \leq \delta)$
 4: for each $i \in \{1...N\}$ do
 5:     $focalFun[i] =$ FocalFunction($I[], i, s, neighType$)
 6:     $focalTest[i] = FocalTest(I[i], focalFun[i])$
 7:     if $focalTest[i] == true$ then
 8:         $T_1 = T_1 \cup \{T[i]\}$
 9:     else
10:         $T_2 = T_2 \cup \{T[i]\}$
11: return $\{T_1, T_2\}$
```

add the sample to its appropriate subset accordingly. The algorithm begins by computing local test indicators (I) of all samples. It then computes the focal function value ($focalFun[i]$) via a *FocalFunction* subroutine, and computes the focal test result ($focalTest[i]$) on each sample location. For example, we may specify the focal function as Γ, and the focal test as "$f \leq \delta \oplus \Gamma < 0$".

FocalFunction: The FocalFunction subroutine (Algorithm 3) computes the focal function values of local test indicators in the neighborhood of a location. It has an important parameter *neighType*, whose value is 0 for a fixed neighborhood, and is 1 for an adaptive neighborhood. The intuition behind an adaptive neighborhood is to utilize spatial topological relationships to select proper neighbors of the central pixel in a fixed window.

Step 1 identifies all locations within the window of size 2s+1 by 2s+1 centered on the current location. These locations are potential neighbors of i.

Algorithm 3 FocalFunction(I, i, s, $neighType$)

```
Input:
  • $I$: vector of indicator variable values
  • $i$: current location
  • $s$: neighborhood window size
  • $neighType$: neighborhood type, 0 for fixed neighborhood, 1 for adaptive neighborhood
Output:
  • $FocalFun[i]$: focal function value at current location $i$
1: identify the 2s+1 by 2s+1 window centered on location $i$
2: if $adaptNeigh == 0$ then
3:     $W_{i,j} = 1$ for all $j$ in the window, $W_{i,j} = 0$ otherwise.
4: else
5:     get connected components of same $I$ values in the window
6:     identify the topologically outermost component $cc_0$ that contains or surrounds location $i$
7:     $W_{i,j} = 1$ for all $j$ in $cc_0$, $W_{i,j} = 0$ otherwise.
8: compute focal function value $foc$ at location $i$ based on $W_{i,j}$
9: return $foc$
```

Steps 2 and 3 determine that all the locations in the window are neighbors of i if a fixed neighborhood is used, similar to our previous work in [207].

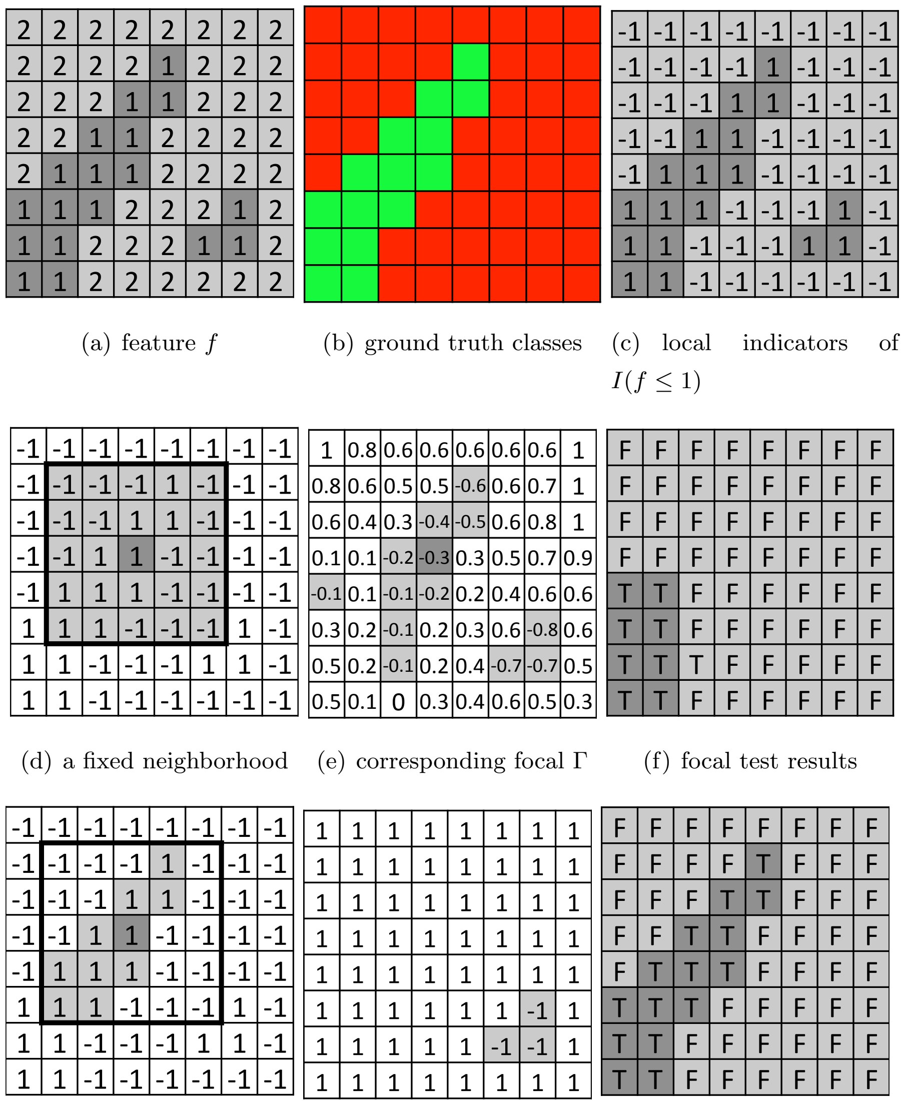
Steps 4 to 7 determine which locations in the window are neighbors if an adaptive neighborhood is used. The window is first segmented into different connected components, each of which has the same the indicator value. Then the component that is the outermost, and that surrounds or contains the current location i, is considered as the actual set of neighbors.

Steps 8 and 9 compute a focal function value based on the neighbors identified, and return the value.

Illustration: The entire execution trace of FTSDT-Train with fixed neighborhoods can be found in [207]. Due to space limitations, here we only illustrate the extension of focal tests with adaptive neighborhoods by comparing them with fixed neighborhoods. Consider the example in Figure 3.6, which describes one iteration of candidate test selection (steps 9 to 10 in Algorithm 1). Assume that the current neighborhood window size is $s = 2$ (i.e., 5 by 5).

Figure 3.6(a)-(c) shows current candidate feature f, ground truth classes, and local test indicators on the current test threshold $\delta = 1$ respectively. The feature f map (Figure 3.6(a)) contains a wedge-shaped patch (fifteen pixels with feature value 1 on the lower left corner) and three salt-and-pepper noise pixels. The pixels on the wedge-shaped patch are not salt-and-pepper noise, and thus shoud not be smoothed (i.e., should avoid over-smoothing).

Figure 3.6(d)-(f) shows the focal test results with fixed neighborhoods. For instance, Figure 3.6(d) highlights the fixed neighborhood (in light grey) of a central pixel (in dark grey), which contains too many irrelevant pixels (with indicator value -1) outside the wedge-shaped patch (the one we previously describe in the last paragraph). The focal function Γ of the central pixel is -0.3 (i.e., $\Gamma < 0$) as shown in dark grey in Figure 3.6(e), mistakenly indicating that it is salt-and-pepper noise. Thus, its final focal test result mistakenly flips its local test result from "true" to "false". Similarly, several other pixels in the wedge-shaped patch are also over-smoothed. The final over-smoothed focal test results of the patch is shown in dark grey in Figure 3.6(f).



(a) feature f

(b) ground truth classes

(c) local indicators of $I(f \leq 1)$

(d) a fixed neighborhood

(e) corresponding focal Γ

(f) focal test results

(g) an adaptive neighborhood

(h) corresponding focal Γ of AN

(i) focal test results

Figure 3.6: Comparison of focal tests with fixed and adaptive neighborhoods, $s = 2$ (i.e., 5 by 5 window).

In contrast, the bottom row of Figure 3.6 shows focal test results with adaptive neighborhoods. Figure 3.6(g) highlights an adaptive neighborhood (in light grey) of the central pixel (in dark grey), which is a connected component (with indicator value 1)

Algorithm 4 FTSDT-Predict(R, T)

```
Input:
    • $R$: root of an FTSDT model
    • $T$: rows as samples, columns as features
Output:
    • $C$: $C[i]$ as class label of $i^{th}$ sample
1: if $R.type == Leaf$ then
2:     assign $C$ with $R.class$
3:     return $C$
4: $f_0 = R.f$, $\delta_0 = R.\delta$, $s_0 = R.s$
5: $\{T_1, T_2\}$ = Node-Split ($T$, $f_0$, $\delta_0$, $s_0$)
6: $C_1$ = FTSDT-Predict ($R.Left$, $T_1$);
7: $C_2$ = FTSDT-Predict ($R.Right$, $T_2$);
8: return $C$ = combine($C_1$, $C_2$)
```

that contains the central dark pixel. The focal function Γ of the central pixel is now 1 (Figure 3.6(b)) based on the adaptive neighborhood. Thus, the final focal test is still "true" ($\Gamma < 0$ is false, "true" xor false is still "true"). The three salt-and-pepper noise pixels are still smoothed. Comparing Figure 3.6(f) and 3.6(i), it is clear that focal tests with adaptive neighborhoods can better separate the two classes (i.e., give higher information gain) due to less over-smoothing of the wedge-shaped area.

3.3.2 Prediction Phase

The FTSDT-Predict algorithm (Algorithm 4) uses an FTSDT to predict the class labels of test samples based on their feature values and a spatial neighborhood structure. It is a recursive algorithm. If the tree node is a leaf, then the class label of the leaf is assigned to all current samples. Otherwise, samples are split into two subsets according to the focal test results in the root node, and each subset is classified by its corresponding sub-tree.

3.4 Computational Optimization: A Refined Algorithm

This section addresses the computational challenges of the focal-test-based spatial decision tree learning process. It first identifies the computational bottleneck of the baseline training algorithm; then proposes a refined algorithm, proves its correctness; and finally provides a cost model for the computational complexity. For simplicity, examples in this section are with a fixed neighborhood. However, the proposed refined algorithm and its analysis are also applicable to the case of adaptive neighborhoods.

3.4.1 Computational Bottleneck Analysis

Recall that the baseline algorithm (Algorithm 1) calls a Node-Split sub-routine for every distinct value (i.e., candidate threshold) on every feature and neighborhood size. Each call involves focal function computations for all samples, and is therefore a likely computational bottleneck. To verify this hypothesis, we conducted computational bottleneck analysis with parameter settings $S_{max} = 5$ and $N_0 = 50$. The results, shown in Figure 3.7, confirm that the focal function computation accounts for the vast majority of total time cost. Furthermore, this cost increases much faster than other costs as training sample sizes increase.

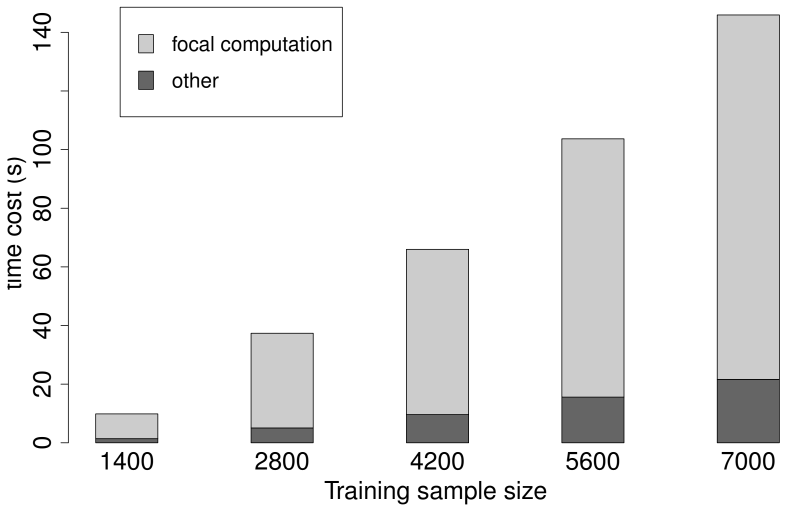
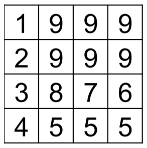


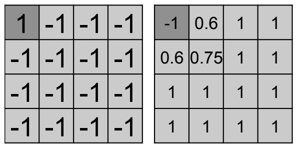
Figure 3.7: Computational bottleneck analysis in training algorithms

3.4.2 A Refined Algorithm

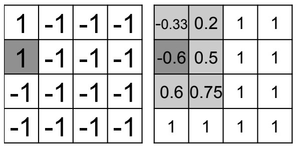
To reduce the computational bottleneck shown above, we designed a refined approach called *cross-threshold-reuse*. This approach is based on the observation that when the candidate threshold value increases, only a small number of samples have their local and focal test results updated. In other words, once computation is completed for one candidate threshold, the test results of most samples will remain the same and can be reused for consecutive thresholds. An illustrative example is given in Figure 3.8(a)-(c), where the values of feature f are shown in Figure 3.8(a), and the local indicators and focal gamma values for test thresholds $\delta = 1$ and $\delta = 2$ are shown in Figure 3.8(b) and 3.8(c), respectively. As can be seen, only location 2 and its neighbors update local indicators and focal values (shown in grey color in Figure 3.8(b)).



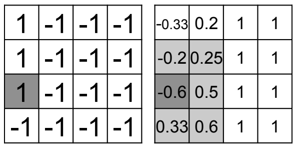
(a) feature f map



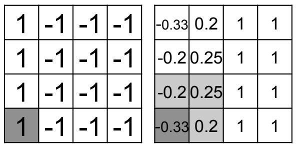
(b) local indicator and focal values for $f \leq 1$



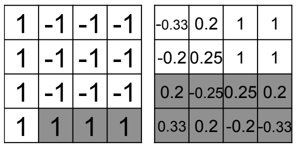
(c) local indicator and focal values for $f \leq 2$



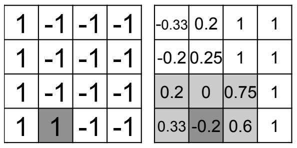
(d) local indicator and focal values for $f \leq 3$



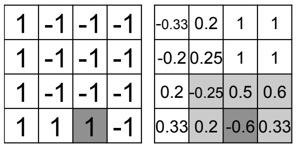
(e) local indicator and focal values for $f \leq 4$



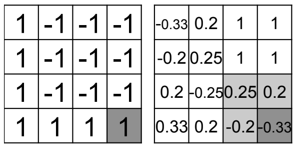
(f) local indicator and focal values for $f \leq 5$ step 1



(g) local indicator and focal values for $f \leq 5$ step 2



(h) local indicator and focal values for $f \leq 5$ step 3



(i) local indicator and focal values for $f \leq 5$ final

Figure 3.8: Illustrative example of redundant focal Γ computation.

The cross-threshold-reuse approach updates one sample at a time together with its neighbors. The details of this approach are given in the algorithm FTSDT-Train-Refined (Algorithm 5). The key difference from previous FTSDT-Train is that the refined algorithm calls the Node-Split subroutine (Algorithm 2) only once. For subsequent sample indices (potential candidate thresholds), it calls Node-Split-Update sub-routine

(Algorithm 6) instead. More specifically, in step 7, it finds the first effective candidate threshold (guaranteed minimum node size N_0). In step 8, it enumerates all possible sample indices. If it is the first enumeration, Node-Split is called to completely compute and memorize local and focal results for all samples (steps 9 to 10). Otherwise, Node-Split-Update is called to avoid redundant computation (step 12). Steps 13 to 16 check if the current split index i is an effective split threshold, and if so, information gain is evaluated to maintain the current best candidate test.

Algorithm 5 FTSDT-Train-Refined(T, S_{max}, N_0)

```
Input:
  • $T$: rows as samples, columns as features (last column as class)
  • $S_{max}$: maximum neighborhood size
  • $N_0$: minimum decision tree node size
Output:
  • root of an FTSDT model
1: denote $N$ as number of samples , $F$ as number of features, $c$ as column index of
   classes ; $IG_0 = -\infty$
2: if $N < N_0$ or $T$ same class then
3:   return a leaf node;
4: for each $f \in \{1...F\}$ do
5:   sort rows of $T$ //in ascending order of $f^{th}$ column
6:   for each $s \in \{0...S_{max}\}$ do
7:     $i_0$ = first $i$ with $T[i][f] > T[N_0][f]$
8:     for each $i \in \{(i_0 - 1)...(N - N_0)\}$ do
9:       if first time then
10:        // memorize $indicator$, $focalFunc$, $T_1, T_2$
           $\{T_1, T_2\}$ = Node-Split($T$, $f$, $\delta = T[i][f]$, $s$)
11:      else
12:        // update $indicator$, $focalFunc$, $T_1, T_2$
           Node-Split-Update($indicator$, $focalFunc$, $i$, $s$,     $\{T_1, T_2\}$)
13:      if $T[i][f] < T[i+1][f]$ then
14:        $IG$ = InformationGain($T[\,][c]$, $T_1[\,][c]$, $T_2[\,][c]$)
15:        if $IG > IG_0$ then
16:          $IG_0 = IG$; $s_0 = s$; $f_0 = f$; $\delta_0 = \delta = T[i][f]$
17: $I$ = CreateInternalNode($f_0$, $\delta_0$, $s_0$);
18: $\{T_1, T_2\}$ = Node-Split($T$, $f_0$, $\delta_0$, $s_0$);
19: $I.left$ = FTSDT-Train-Refined($T_1$, $S_{max}$, $N_0$)
20: $I.right$ = FTSDT-Train-Refined($T_2$, $S_{max}$, $N_0$)
21: return $I$
```

Details of the Node-Split-Update sub-routine are given in Algorithm 6. The input of this sub-routine includes the current local and focal test results, the index of the currently enumerated sample i, neighborhood size s, and the current split result $\{T_1, T_2\}$. Node-Split-Update begins by updating the local and focal tests of sample i, and adjusting $\{T_1, T_2\}$ accordingly. Then it updates the test results of every neighbor j of sample i, and adjusts $\{T_1, T_2\}$. These updates all carry a constant time cost since these are done in only one small neighborhood window.

```
Algorithm 6 Node-Split-Update($indicator$, $focalFunc$, $i$, $s$, $\{T_1, T_2\}$)
Input:
  • $indicator$: array of local result as $I(f \leq \delta)$
  • $focalFunc$: array of focal function values, e.g., $\Gamma_I^S$
  • $i$: index of sample shifted below threshold
  • $s$: neighborhood size
  • $\{T_1, T_2\}$: two subsets of samples
1: $indicator[i] = 1$; update $focalFunc[i]$, and then $\{T_1, T_2\}$
2: for each $j \in N^s(i)$ // $N^s(i)$ is $i$'s neighborhood of size $s$ do
3:     update $indicator[j]$, $focalFunc[j]$, and then $\{T_1, T_2\}$
```

Execution trace: Figure 3.8 illustrates the execution trace of steps 8 to 12 of the new update algorithm. The context is as follows: feature f is shown in Figure 3.8(a); $s = 1$ (3 by 3 fixed neighborhood); $N_0 = 1$; the local indicator is $I(f \leq \delta)$; and the focal function is Γ as before. Figure 3.8 (b)-(i) are local indicators and focal function values under different candidate test thresholds (1 to 5). The refined algorithm only updates the local indicators and focal values, shown in grey colors of Figure 3.8(c-e) and 3.8(g-i).

3.4.3 Theoretical analysis

We now prove the correctness of proposed computationally refined algorithm. We also provide a cost model of computational complexity. The proof of correctness is nontrivial, because when the candidate threshold changes, multiple sample locations as well as their neighbors may need to update their focal values (e.g., Figure 3.8(f)), and these updates are at the same time. However, our approach still simply changing one sample location as well as its neighbors each time (e.g., Figure 3.8(g-i)), and it has the same results.

Theorem 1. *The FTSDT-Train-Refined algorithm is correct, i.e., it returns the same output as FTSDT-Train.*

Proof. We need only look at steps 7 to 16. The initial value of i in the for-loop here (i.e., one step ahead of the first sample whose feature value is greater than that of N_0) is the same as that value in FTSDT-Train, due to the if-condition in step 8 of FTSDT-Train. Now we focus on the local and focal computation part in steps 9 to 12. We will prove it in two cases as below.

Case 1: a new threshold shifts only one sample i, i.e., $T[i][f] < T[i+1][f]$. In this case, the local result $I(f \leq \delta)$ changes only on sample i, i.e., $I[i] = 1$, while the feature value f is unchanged. Since the focal function of a sample only depends on f and I in its neighborhood, its value changes only on sample i and its neighbor js. Thus, Node-Split-Update in this case is correct. One example of this case appears in Figure 3.8(b)(c).

Case 2: a new threshold shifts multiple samples, i.e., $T[i][f] = T[i+1][f] = ... = T[i+k][f] < T[i+k+1][f]$. In this case, our refined algorithm still only updates sample i and its neighbors, as though $T[i][f] < T[i+1][f]$ (in other words, as though $T[i][f]$ were an effective candidate threshold). This updating process continues until new i becomes $i+k$, i.e., the next effective candidate threshold. If the feature value $T[i][f]$ were strictly increasing, the final local and focal values should be correct, as proved in case 1. Meanwhile, it is also obvious that whether feature values are strictly increasing or not before $i = i+k$ does not influence the final local and focal values. Thus, the final updated result for $i = i+k$ is also correct. An example for this case is given in Figure 3.8(c)(d). $\square$

To analyze the cost model of the two proposed training algorithms, we denote the following variables:

- N: number of samples
- N_d: number of distinct feature values
- S_{max}: maximum neighborhood size

Table 3.2: Simplified cost model with different numbers of distinct feature values (N_d)

Algorithm	$N_d = O(1)$	$N_d = O(N)$
FTSDT-Train	$O(N^2 \log N)$	$O(N^3)$
FTSDT-Train-Refined	$O(N^2 \log N)$	$O(N^2 \log N)$

- N_0: minimum tree node size
- F: number of features

Lemma 1. *The baseline algorithm FTSDT-Train has a time complexity of* $O(FN^2(\log N + N_d S_{max}^3)/N_0)$.

Proof. Given N samples and minimum node size N_0, tree node number is at most N/N_0, i.e., $O(N/N_0)$. For each tree node, the algorithm sorts samples for all features and enumerates through all $O(N_d)$ thresholds for all the F features under all the $S_{max}+1$ different neighborhood sizes. In each enumeration, a Node-Split sub-routine is called, which has time complexity $O(NS_{max}^2)$, where $O(S_{max}^2)$ is the number of neighbors under a square neighborhood. Thus, for each node, the time cost is $O(F \cdot (N \log N + S_{max} \cdot N_d \cdot NS_{max}^2)) = O(FN(\log N + N_d S_{max}^3)$. Finally, the total time cost is $O(N/N_0 \cdot FN(\log N + N_d S_{max}^3)) = O(FN^2(\log N + N_d S_{max}^3)/N_0)$. □

Lemma 2. *The SDT-Train-Refined algorithm has a time complexity of* $O(FN^2(\log N + S_{max}^3)/N_0)$.

Proof. The number of tree nodes is $O(N/N_0)$. For each node and each of the $O(F)$ features, the refined algorithm sorts and enumerates through all $O(N)$ samples under all $O(S_{max} + 1)$ neighborhood sizes. Node-Split is called only once (with time cost $O(NS_{max}^2)$) in these enumerations and Node-Split-Update is called for the rest (each with time cost $O(S_{max}^2)$). Thus, for each node, the time cost is $O(F \cdot N \cdot \log N + FS_{max} \cdot (NS_{max}^2 + N \cdot S_{max}^2)) = O(FN(\log N + S_{max}^3)$, and the total time cost is $O(FN^2(\log N + S_{max}^3)/N_0)$. □

Theorem 2. *FTSDT-Train-Refined is faster than FTSDT-Train when* $N_d \gg 1$ *(i.e.,* N_d *is much greater than* 1*).*

Proof. From the lemmas above, the cost models of the two algorithms only differs in one factor, which is $O(\log N + N_d S_{max}^3)$ for FTSDT-Train and $O(\log N + S_{max}^3)$ for the refined algorithm. Since $N_d \gg 1$, the cost of the refined algorithm is always smaller. The same can be proved by simplifying the two cost models, if we assume $N_d \propto N$, and F, S_{max}, N_0 are constants. Then the cost of FTSDT-Train is $O(N^3)$, while the cost of FTSDT-Train-Refined is $O(N^2 \log N)$, as shown in Table 3.2. Note that the condition $N_d \gg 1$ is often satisfied for continuous features. □

3.5 Experimental Evaluation

The goal was to investigate the following questions:

- How do LTDT, FTSDT-fixed, and FTSDT-adaptive compare with each other in classification accuracy?
- How do LTDT, FTSDT-fixed, and FTSDT-adaptive compare with each other in salt-and-pepper noise level?
- Does the FTSDT-Train-Refined algorithm reduce the computational cost of baseline FTSDT-Train algorithm?

3.5.1 Experiment Setup

Experiment design: The experiment design is shown in Figure 4.8. To evaluate classification performance, we compared the LTDT learner (i.e., C4.5), the FTSDT learner with fixed neighborhoods (i.e., FTSDT-fixed), as well as the FTSDT learner with adaptive neighborhoods (i.e., FTSDT-adaptive) on test accuracy and autocorrelation level. To evaluate computational performance, we used FTSDT with fixed neighborhoods for simplicity, and compared the baseline approach (i.e., FTSDT-Train) with the computationally refined approach (i.e., FTSDT-Train-Refined). Computational time reported was the average of 10 runs. All the algorithms were implemented in C language. Experiments were conducted on a Dell workstation with Quad-core Intel Xeon CPU E5630 @ 2.53GHz, and 12 GB RAM.

Table 3.3: Classification performance of LTDT, FTSDT-fixed, FTSDT-adaptive

Scene	Models	Confusion Matrix		Precision	Recall	F-score	Γ index
1	LTDT	99141	10688	0.81	0.75	**0.78**	0.87
		15346	45805				
	FTSDT-fixed	99755	10074	0.83	0.80	**0.81**	0.96
		12470	48681				
	FTSDT-adapt	99390	10439	0.83	0.83	**0.83**	0.93
		10618	50533				
2	LTDT	104615	10820	0.70	0.66	**0.68**	0.87
		13254	25612				
	FTSDT-fixed	107297	8138	0.77	0.69	**0.73**	0.96
		11984	26882				
	FTSDT-adapt	105999	9436	0.76	0.75	**0.75**	0.92
		9744	29122				

Table 3.4: $\hat{K}$ statistics of confusion matrices

Scene	LTDT		FTSDT-fixed		FTSDT-adaptive	
	$\hat{K}$	$\hat{K}$ variance	$\hat{K}$	$\hat{K}$ variance	$\hat{K}$	$\hat{K}$ variance
1	0.66	$3.6 * 10^{-6}$	0.71	$3.2 * 10^{-6}$	0.73	$3.0 * 10^{-6}$
2	0.58	$5.9 * 10^{-6}$	0.64	$5.3 * 10^{-6}$	0.67	$4.8 * 10^{-6}$

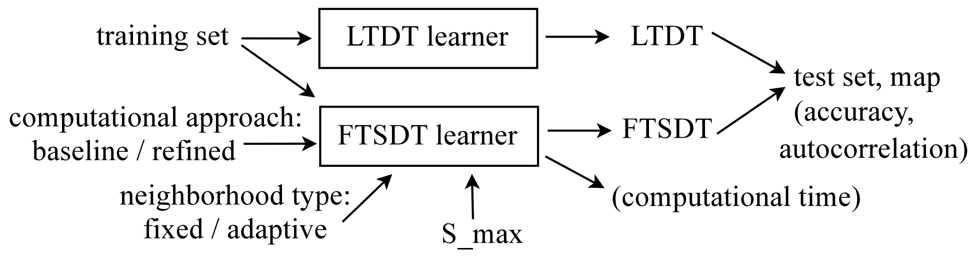

training set
LTDT learner
LTDT
computational approach: baseline / refined
FTSDT learner
FTSDT
test set, map (accuracy, autocorrelation)
(computational time)
neighborhood type: fixed / adaptive
S_max

Figure 3.9: Experiment Design

Dataset description: We used high resolution (3m by 3m) remote sensing imagery collected from the city of Chanhassen, MN, by the National Agricultural Imagery Program and Markhurd Inc. There were 12 continuous explanatory features including multi-temporal (for the years 2003, 2005, and 2008) spectral information (R, G, B, NIR) and Normalized Difference Vegetation Index (NDVI). Class labels (wet land and

Table 3.5: Significance test on difference of confusion matrices

Scene	LTDT v.s. FTSDT-fixed		FTSDT-fixed v.s. FTSDT-adaptive	
	Z score	Result	Z score	Result
1	18.2	significant	8.6	significant
2	19.4	significant	8.5	significant

Table 3.6: Description of datasets

Scene	Size	Training samples
1	476 by 396	11837(dryland class); 5679 (wetland class)
2	482 by 341	7326 (dryland class); 2735 (wetland class)

dry land) were created by a field crew and photo interpreters between 2004 and 2005.

To evaluate classification performance, we selected two scenes from the city. On each scene, we used systematic clustered sampling to select a number of wetland and dryland contiguous clusters of pixels as the training set and the remaining pixels as test sets. More details are given in Table 3.6. To evaluate computational performance, we used scene 1 and created training sets with different sizes and number of distinct feature values to test sensitivity of computational cost on various settings. The variables tested were previsouly defined in Section 4.3.

Choice of focal test functions: For the focal-test-based spatial decision tree, we used the specific focal test $(f \leq \delta) \oplus (\Gamma < 0)$ described in Section 2.1.

3.5.2 Classification Performance

How do LTDT, FTSDT-fixed, and FTSDT-adaptive compare in classification accuracy?

Parameter settings were $S_{max} = 5$, $N_0 = 200$ for the first dataset, and $N_0 = 50$ for the second dataset. We compared the classification performance of the proposed FTSDT-adaptive and FTSDT-fixed with LTDT in terms of confusion matrices, precision & recall, and F-measure (i.e., harmonic mean of precision and recall) on the test set. The results are listed in Table 3.3. In the confusion matrix, the columns are test samples classified as dryland and wetland respectively, and the two rows are test samples whose true class labels are dryland and wetland respectively. Precision and recall were computed on the wetland class. As can be seen, on the first dataset, FTSDT-fixed improves the F-measure of LTDT from 0.78 to 0.81 (e.g., false negatives decrease by around 20% from 15346 to 12470), and FTSDT-adaptive further improves the F-measure of FTSDT-fixed from 0.81 to 0.83 (e.g., false negatives decrease by around 15% from 12470 to 10618). Similar improvements are also seen in the results on the second dataset.

We also conducted significance tests on the difference of the confusion matrices between LTDT and FTSDT-fixed, and between FTSDT-fixed and FTSDT-adaptive. The statistic used was $\hat{K}$ (estimate of Kappa coefficient) [213] [214] [215], defined as

$$\hat{K} = \frac{n\sum_{i=1}^{k} n_{ii} - \sum_{i=1}^{k} n_{i+}n_{+i}}{n^2 - \sum_{i=1}^{k} n_{i+}n_{+i}},$$

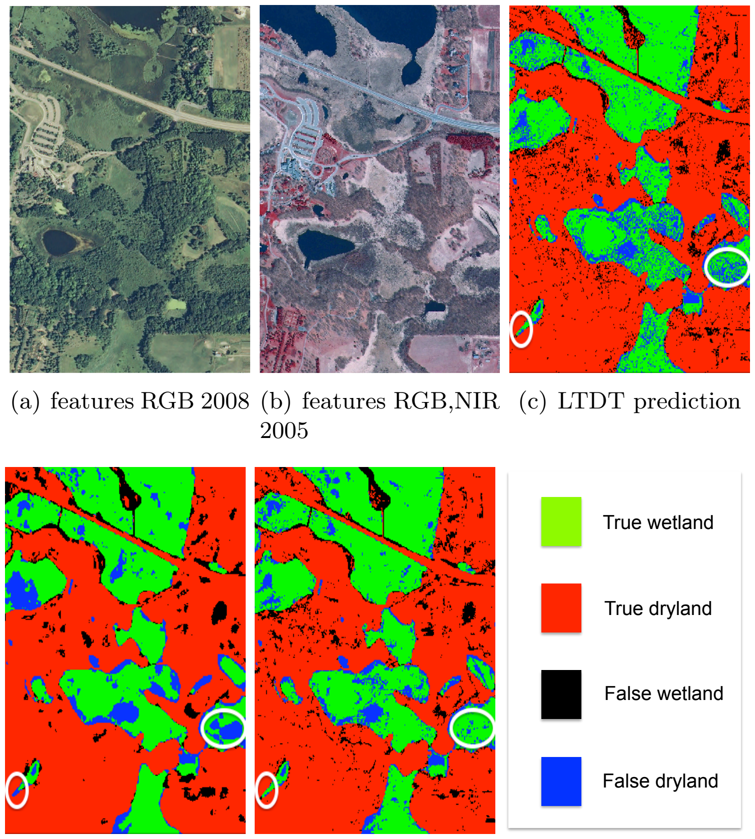
where n is the sum of all elements, and n_{ii}, n_{i+} and n_{+i} are the diagonal, row sum and column sum respectively. $\hat{K}$ reflects the degree to which a confusion matrix is different from a random guess. First we computed $\hat{K}$ and its variance for each evaluation candidate as shown in Table 3.4. Then we conducted a Z-test on pairs of $\hat{K}$ statistics of LTDT and FTSDT-fixed, as well as FTSDT-fixed and FTSDT-adaptive. The results show that improvements of FTSDT-adaptive over LTDT and FTSDT-fixed in confusion matrices are significant (Table 3.5).

How do LTDT, FTSDT-fixed, and FTSDT-adaptive compare with each other in salt-and-pepper noise level?

We compared prediction maps by LTDT, FTSDT-fixed, and FTSDT-adaptive on the amount of salt-and-pepper noise, as measured by a spatial autocorrelation statistic, i.e., gamma index Γ with queen neighborhoods. This index ranges from 0 to 1, and a larger index value indicates less salt-and-pepper noise. Parameter settings were $S_{max} = 5$, $N_0 = 200$ for the first dataset, and $N_0 = 50$ for the second dataset. The last column of Table 3.3 shows the spatial autocorrelation levels of the LTDT, FTSDT-fixed, and FTSDT-adaptive predictions on the two datasets. As can be seen, both FTSDT-fixed and FTSDT-adaptive improve the spatial autocorrelation (i.e., reducing salt-and-pepper noise) over LTDT significantly. The spatial autocorrelation of FTSDT-adaptive predictions is somewhat lower than for FTSDT-fixed since it uses flexible neighborhoods to avoid FTSDT-fixed's over-smoothing. Nonetheless, the overall classification accuracy of FTSDT-adaptive is better.

Case Study

We ran a case study to illustrate the difference of predictions among the LTDT algorithm, the FTSDT-fixed and FTSDT-adaptive learning algorithms. The dataset was again the Scene 1 images from the city of Chanhassen. Several of the input multi-temporal optical features are mapped in Figure 3.10(a) and 3.10(b). Target classes were wetland and dryland. The maximum neighborhood size was set to 5 (11-by-11 window) and minimum tree node size was 200.



(a) features RGB 2008 (b) features RGB,NIR 2005 (c) LTDT prediction

(d) FTSDT-fixed prediction (e) FTSDT-adaptive prediction

Figure 3.10: Case study dataset and prediction results of LTDT and FTSDT.

The predictions of LTDT, FTSDT-fixed, and FTSDT-adaptive are shown in Figure 3.10(c)-(e) respectively. The green and red colors represent correctly classified wetland and correctly classified dryland. The black and blue colors represent errors of false wetland and false dryland. As can be seen, the prediction by LTDT has lots of salt-and-pepper noise (in black and blue colors) due to the high local variation of features within wetland or dry land patches. The predictions of FTSDT-fixed (Figure 3.10(d)) and FTSDT-adaptive (Figure 3.10(e)) show a dramatic reduction in salt-and-pepper noise. FTSDT-fixed appears to over-smooth some areas (e.g., blue color in the white circles of Figure 3.10(d)), likely due to fixed square neighborhoods. In contrast, FTSDT-adaptive's predictions show less over-smoothing effect in the white circles. The reason is that its focal function is computed based on flexible neighborhoods adapted to the spatial topological relationship among locations. FTSDT-adaptive has somewhat lower spatial autocorrelation in predictions than FTSDT-fixed due to less aggressive smoothing, but its overall accuracy is better.

3.5.3 Computational Performance

This section compares the computational performance of the new FTSDT-Train-Refined algorithm with the baseline FTSDT-Train algorithm on different parameter settings. For simplicity, we fixed the neighborhood type as fixed neighborhoods.

Different numbers of training samples N

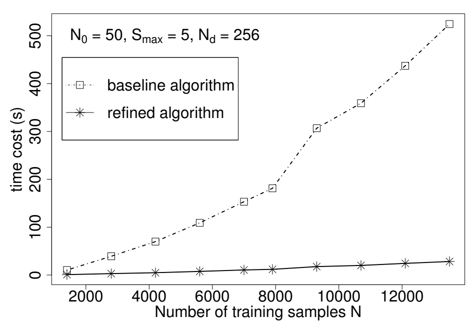
We fixed the variables as follows: $N_0 = 50$, $S_{max} = 5$, $N_d = 256$, and increased the number of training samples.

Figure 3.11 (a) shows the result. As can be seen, when the training sample size is very small (e.g. 1000), the time cost of both algorithms is close. However, as the training sample size increases, the time cost of the baseline algorithm increases at a much higher rate than the refined algorithm. This result accords with cost models in *Lemmas 1* and *2*, which showed that the baseline algorithm FTSDT-Train has a larger constant factor on the $O(\log N + N_d S^3)$ term.

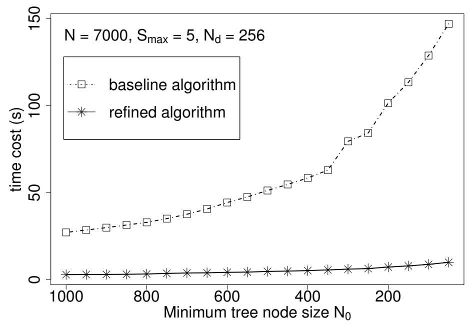
Different minimum tree node sizes N_0

We fixed the variables $N = 7000$, $S_{max} = 5$, $N_d = 256$, and increased the minimum tree node size.

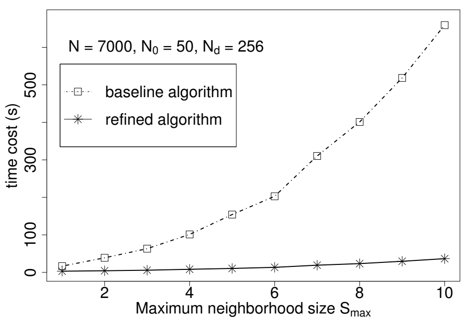
Figure 3.11 (b) shows the result. As can be seen, as the minimum tree node size



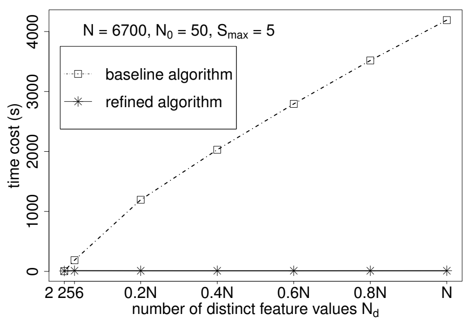
(a)



(b)



(c)



(d)

Figure 3.11: Computational performance comparison between basic and refined algorithms

increases, the time cost of both algorithms decreases. The reason is that fewer tree nodes are constructed and thus less computation is needed. But our refined algorithm has persistently lower cost than our baseline algorithm. This result aligns with previous cost models in *Lemmas 1* and *2*, where the baseline algorithm has a larger numerator.

Different maximum neighborhood sizes S_{max}

We fixed the variables $N = 7000$, $N_0 = 50$, $N_d = 256$, and increased the maximum neighborhood size.

Figure 3.11 (c) shows the result. As can be seen, when the maximum neighborhood size is very small (i.e., 1), the time cost of both algorithms is close, due to the low time cost when S_{max} is very small. However, as the maximum neighborhood size increases, the time cost of the baseline algorithm grows dramatically faster than the refined algorithm. This result matches the cost models in *Lemmas 1* and *2*, where the baseline algorithm has a larger constant factor N_d on the $O(N_d S^3)$ term.

Different number of distinct feature values N_d

We fixed the variables $N = 6700$, $N_0 = 50$, and $S_{max} = 5$ and increased the number of distinct feature values N_d from 2 to 256 (default value without adding simulation), and $0.2N$, $0.4N$, $0.6N$, $0.8N$ to approximately N. In order to control N_d in datasets, we independently added random noise in uniform distribution $U(0, 0.01)$ to each feature value and then specified the precision of the decimal part or even the integer part (greater precision increases N_d values). The reported N_d values are the averages across all features.

Figure 3.11(d) shows the result. As can be seen, when $N_d = 2$ (the first tick mark), the time cost of the two algorithms is very close (baseline algorithm costs 3.3s and refined algorithm costs 7.2s). The reason is that the focal computation cost of even the baseline approach is very small when N_d is close to 1 (baseline cost is slightly lower due to other constant factors). However, as N_d increases, the time cost of the baseline algorithm grows almost linearly to N_d while that of the refined algorithm remains the same.

This result can be explained by the cost models in *Lemmas 1* and *2*, where the baseline algorithm has a factor $O(\log N + N_d S^3)$ while the refined algorithm has a corresponding term $O(\log N + S^3)$. As the cost models imply, when N_d is close to 1 (e.g., 2), the two algorithms' time costs are very close. But as N_d increases, the cost of the baseline algorithm is a linear function of N_d given other variables are constant.

Different image sizes N

We fixed the variables $N_0 = 50$, $S_{max} = 5$, $N_d = 256$, and increased the size of training image (in terms of the amount of pixels) from 3960, 7920, ..., to 39600.

Figure 3.12 shows the result. As can be seen, when the training image size is very small (e.g. 3960 pixels), the time cost of both algorithms is close. However, as the training image size increases, the time cost of the baseline algorithm increases at a

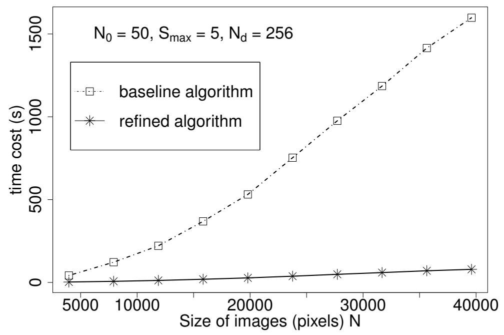


Figure 3.12: Computational performance comparison on different image sizes

much higher rate than the refined algorithm. This shows that the refined algorithm is much more scalable to large image sizes.

3.6 Discussion

A number of other relevant techniques exist for reducing salt-and-pepper noise. Examples include pre-processing (median filtering [211], weighted median filtering [216], adaptive median filtering [217], decision based filtering [218] [219]), post-processing (per parcel classification [220], spectral and spatial classification [221]), and adding contextual variables to the input features [222]. An increasingly popular technique is image segmentation, especially Geographic-Object-Based Image Analysis (GEOBIA) [223]. In this technique, the image is first segmented into different objects. Then each object is a minimum classification unit. All these techniques can help reduce salt-and-pepper noise. However, they require labor-intensive manual tuning by domain experts beyond model learning and prediction. Our FTSDT approach automates the tuning process in model learning and prediction, potentially saving users hours of labor.

3.7 Summary

This work explores the spatial decision tree learning problem for raster image classification. The problem is challenging due to the spatial autocorrelation effect and computational cost. Related work is limited to using local tests in tree nodes. In contrast, we propose a focal-test-based spatial decision tree (FTSDT) model and its learning algorithm. We further conduct computational optimization and design a refined algorithm that selectively updates focal values. Both theoretical analysis and experimental evaluation show that our refined algorithm is more scalable than our baseline algorithm. We also design a new focal test approach with adaptive neighborhoods to avoid over-smoothing in wedge-shaped areas. Experiment results on real world datasets show that new FTSDT with adaptive neighborhoods improves classification accuracy of both the default FTSDT with fixed neighborhoods and traditional LTDT.

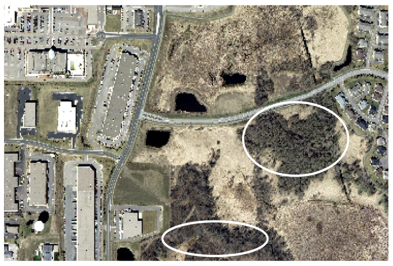
Chapter 4

Spatial Ensemble Learning

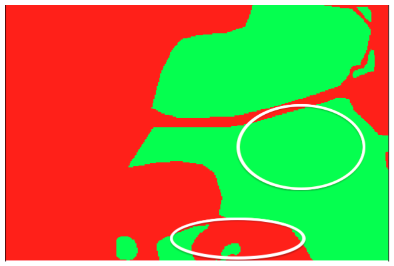
4.1 Introduction

Class ambiguity, i.e., sample feature values being ineffective in discriminate classes, is a fundamental challenge in supervised learning [224, 225, 226, 227]. Given geographical data with class ambiguity, i.e., samples with the same feature values belong to different classes in different zones, spatial ensemble learning (SEL) aims to find a decomposition of the geographical area into zones so as to minimize class ambiguity and to learn a local model in each zone. Figure 4.1 shows a real world example of land cover mapping, in which the goal is to classify remote sensing image pixels (Figure 4.1(a)) into wetland and dry land classes. The ground truth of the region is shown in Figure 4.1(b). The white circles in Figure 4.1(a) and (b) highlight two groups of pixels with class ambiguity, i.e., their spectral (feature) values are very close (Figure 4.1(a)), but they belong to two different classes (”wetland” and ”dry land”). This is also shown in Figure 4.1(c). The prediction of a decision tree classifier learned from the entire image is shown in Figure 4.1(d); it contains errors within ambiguous areas. The goal of SEL is to reduce class ambiguity by decomposition of geographical space into zones.

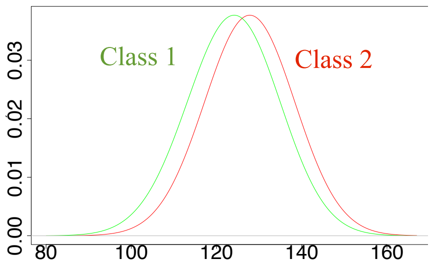
Societal applications: Class ambiguity is a common issue in geographical classification applications [228]. Due to heterogeneous geographical and topographic factors, the same spectral signatures on remote sensing images may correspond to different land cover classes in different sub-regions [229, 230, 231, 227]. The issue is particularly important in countries where access to multiple type of data (that may reduce spectral



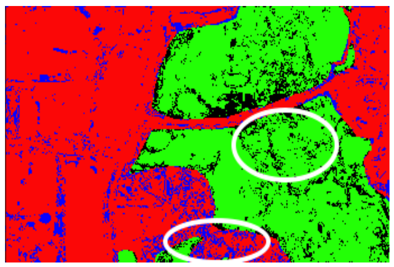
(a) Spectral features in remote sensing image



(b) Ground truth classes (red for dry land, green for wetland)



(c) Distribution of feature values (near infrared band) in circles



(d) Decision tree predictions (black and blue show errors)

Figure 4.1: Real world example of geographical classification with class ambiguity: class ambiguity exists in two white circles of (a) (best viewed in color)

confusion) such as elevation data, spectral imagery collected in different type of season or high resolution imagery might not be available. Class ambiguity exists in other domains as well. In economic study, it is possible that old house age indicates high price in rural areas but low price in urban areas [232]. Similar examples also exist in studies of culture. For instance, touching somebody during conversation is welcomed in France and Italy, but considered offensive in Britain unless in a sport field; the "V-Sign" gesture can mean "two" in America, "victory" in German, but "up yours" in Britain [233]. In these types of applications, there is no universal or global classifier that can effectively discriminate between different classes. Spatial ensemble methods are needed to learn geographically local models to avoid class ambiguity.

Challenges: Spatial ensemble learning poses three main challenges. First, the footprints of effective spatial partition that minimizes class ambiguity are often unknown beforehand with arbitrary shapes in continuous geographical space. Second, local footprints sometimes need to satisfy certain geographical constraints (e.g., spatial contiguity, spatial topographic relationships) in order to be interpretable for geographical analysis. Finally, the number of possible ways a space can be partitioned is exponential, making the selection of a best choice computationally challenging.

Related work: Ensemble learning [234, 235, 236, 237] is the process by which a set of diverse weak models are combined to boost prediction accuracy. Conventional ensemble methods, including bagging [238], boosting [239], and random forest [240], assume an identical and independent distribution of samples. Thus they cannot address heterogeneous geographical data with class ambiguity. Decomposition based ensemble methods (also called divide-and-conquer), including mixture of experts [241, 242, 243, 244] and multimodal ensemble [245, 246], go beyond the identical and independent distribution assumption in that these methods can partition multi-modular input data and learn models in local partitions. Partitioning is usually conducted in feature vector space via a gating network, which can be learned simultaneously by an EM algorithm, or modeled by radius basis functions [247] or multiple local ellipsoids [248]. However, partitioning input data in feature vector space cannot effectively separate samples with class ambiguity because such samples are very "close" in non-spatial feature attributes (Figure 4.5(a)). Moreover, adding spatial coordinates into feature vectors creates geographical partitions whose footprints are hard to interpret and can be too rigid to separate ambiguous samples with arbitrary footprint shapes (Figure 4.5(b)). It is worth noting that mixture-of-experts approach has been widely used in image classification via partitioning images into sub-blocks and combining local experts learned from individual sub-blocks. However, the problem it solves is scene classification where an entire image (not individual pixels) is classified into one class (e.g., indoor, outdoor) [249, 250, 251]. Our chapter deals with pixel-wise classification, and particularly class ambiguity among pixels.

To address the limitations of related work, we propose a spatial-based ensemble learning framework that explicitly partition input data in geographical space: first, the input data is preprocessed into homogeneous "patches" via constrained hierarchical spatial clustering; second, patches are grouped into several footprints via greedy seed growing and spatial adjustment.

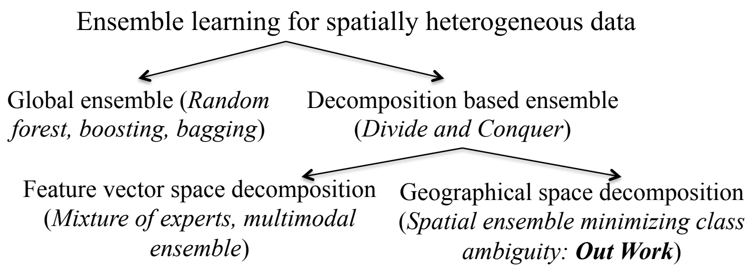

Ensemble learning for spatially heterogeneous data
Global ensemble (*Random forest, boosting, bagging*)
Decomposition based ensemble (*Divide and Conquer*)
Feature vector space decomposition (*Mixture of experts, multimodal ensemble*)
Geographical space decomposition (*Spatial ensemble minimizing class ambiguity: **Out Work***)

Figure 4.2: Related work summary

We make the following contributions in this work: (1) we formulate the spatial ensemble learning problem to address class ambiguity issue in geographical data due to spatial heterogeneity; (2) we develop spatial ensemble algorithms based on greedy heuristics; (3) we evaluate the proposed algorithms on three real world remote sensing datasets and show that they outperform related work in classification performance.

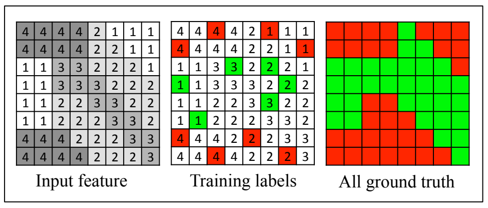
Scope: Our focus is spatial ensemble learning to address class ambiguity that originates from unknown heterogeneous geographical factors (e.g., terrain). Post-processing approaches in image classification to address class ambiguity [227] are beyond the scope of this work. Though spatial ensemble learning can be developed for general geographical data, we only consider raster imagery in this chapter for simplicity. We focus on pixel-wise classification. Per-field (parcel, object) classification [227] is not addressed.

Outline: This chapter is organized as follows: Section 4.2 defines basic concepts and formalizes the spatial ensemble learning problem; Section 4.3 introduces our approach. Experimental evaluations are in Section 4.4. Section 4.5 discusses some other relevant techniques in the literature. Section 4.6 makes a summary with discussion on potential future work.

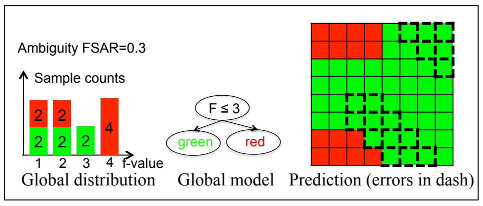
4.2 Problem Statement

4.2.1 Basic Concepts

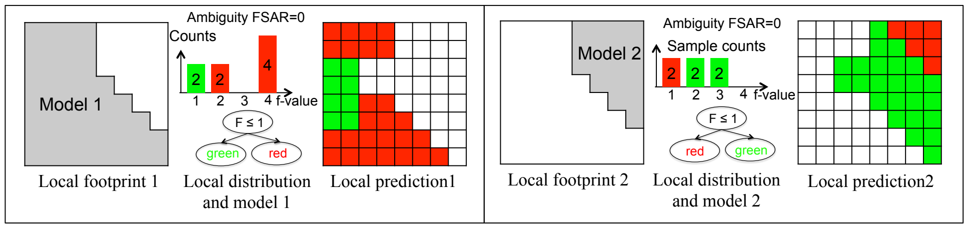
Geographical raster data: A geographical raster framework F is a tessellation of the 2-D plane into a regular grid. It contains a set of explanatory feature maps and a class label map. Each grid cell on the raster framework is a *spatial data sample* with non-spatial attribute features, spatial coordinates, and a class label. For simplicity, we use the words “sample” and “pixel” interchangeably.



(a) Problem inputs



(b) Problem outputs for global model



(c) Problem outputs for spatial ensemble

Figure 4.3: Illustrative example of problem inputs and outputs

Class ambiguity: Class ambiguity refers to the phenomenon that sample feature values are ineffective in discriminate classes [224, 225, 226, 227]. Class ambiguity in geographical data, i.e., same feature values correspond to different classes in different zones, is often due to spatial heterogeneity influenced by unknown geographical confounding factors. An illustrative example is in Figure 4.3(a), where the same feature values ($f = 1$ or $f = 2$) correspond to red or green classes in different subregions. To quantify the extent of class ambiguity given a training set, we propose the use of *feature space ambiguity ratio* (FSAR). FSAR is defined as $FSAR = \frac{1}{N}\sum_{i=1}^{N} FSAR_i = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{k}\sum_{x_j \in N_k(x_i)} I(c_j \neq c_i)$, where x_i and c_i are the feature vector and class of training sample i, N is the total number of training samples, $N_k(x_i)$ is the neighborhood of sample i determined by k-nearest-neighbor in feature vector distance. For instance, the $FSAR$ for sample distribution in Figure 4.3(b) is $(0.5\times4+0.5\times4+0\times2+0\times4)/14 = 0.3$ given $k = 2$.

Theorem 3. *Expectation of proposed class ambiguity measure FSAR is an upper bound of Bayesian error.*

Proof. We prove the theorem in the case of binary class, as illustrated by Figure 4.4, where $P(C_1)$ and $P(C_2)$ are the probability of class C_1 and class C_2 in the population, $f_{C_1}(x)$ and $f_{C_2}(x)$ are the probability density function of feature values for samples in class C_1 and C_2 respectively. From definition of $FSAR$, we can get:

$$\begin{aligned} E(FSAR) = \int_x & P(C_1)f_{C_1}(x)dx \frac{P(C_2)f_{C_2}(x)dx}{P(C_1)f_{C_1}(x)dx + P(C_2)f_{C_2}(x)dx} \\ & + P(C_2)f_{C_2}(x)dx \frac{P(C_1)f_{C_1}(x)dx}{P(C_1)f_{C_1}(x)dx + P(C_2)f_{C_2}(x)dx} \\ & = \int_x \frac{2P(C_1)f_{C_1}(x)P(C_2)f_{C_2}(x)}{P(C_1)f_{C_1}(x) + P(C_2)f_{C_2}(x)} dx \end{aligned}$$

The Bayesian error is:

$$\begin{aligned} BayesianError &= \int_x min(p(C_1)P(x|C_1), P(C_2)P(x|C_2)) \\ &= \int_x min(p(C_1)f_{C_1}(x)dx, P(C_2)f_{C_2}(x)dx) \\ &= \int_x min(p(C_1)f_{C_1}(x), P(C_2)f_{C_2}(x))dx \end{aligned}$$

Given the fact that Harmonic mean of $p(C_1)f_{C_1}(x)$ and $P(C_2)f_{C_2}(x))$ should be no less than their minimum, we have $BayesianError \leq E(FSAR)$ □

Corollary 3.1. *If* $E(FSAR) \to 0$, $BayesianError \to 0$.

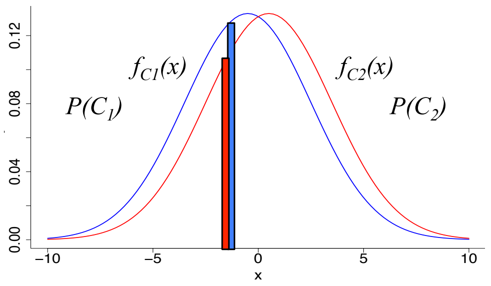


Figure 4.4: Interpretation of class ambiguity for binary class

Spatial ensemble: Spatial ensemble is a decomposition of geographical space into zones to minimize class ambiguity and learn a model in each zone. Figure 4.3(c) shows a spatial ensemble with two zones, in which class ambiguity is reduced from 0.3 (globally) to 0 (in each zone).

Patchiness of zones: Sometimes, zonal footprints in a spatial ensemble need to somehow spatially contiguous, in order to be interpretable for geographical analysis. To measure the degree of contiguity, we use *patchiness*, i.e., amount of isolated components a zone contains in the map [252]. For example, each zone in Figure 4.3(c) only has one isolated component (or absolutely contiguous), so the patchiness is only 1. The word "patchiness" (poor contiguity) should not be confused with "patch" (homogeneous area).

4.2.2 Problem Definition

The spatial ensemble learning problem is formally defined as follows:

Given:

• a geographical (raster) framework F

• m explanatory (non-spatial) feature maps in F

• training and test samples with class labels in F

• size of spatial ensemble: k

Find: k zones for spatial ensemble

Objective: minimize class ambiguity within zones

Constraints:

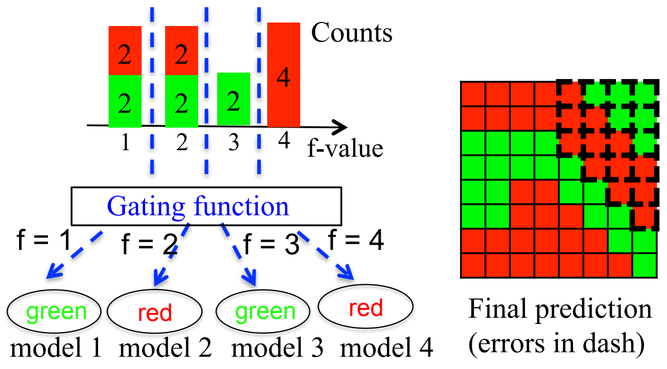
• each footprint has patchiness smaller than a threshold

• spatial autocorrelation exists (pixel size $\ll$ class parcel size)

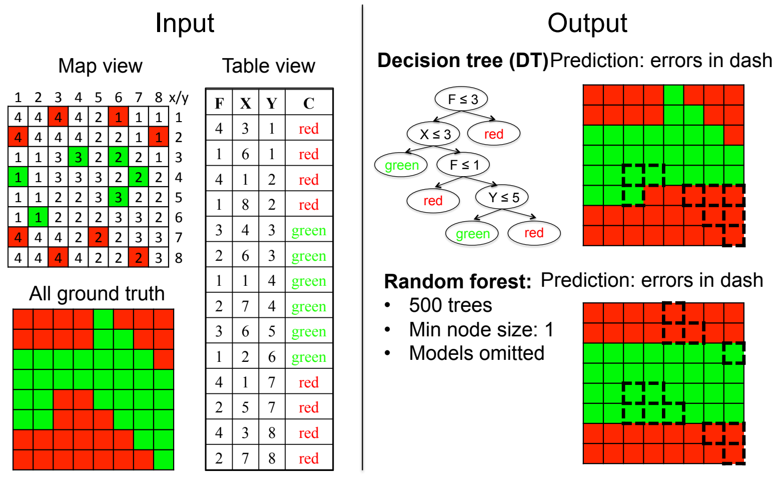
• test samples exist in the same framework • class ambiguity exists across samples but not within a sample

Discussion: The input feature maps cover all samples (including both training and test samples) in the framework. Feature values of test samples on the map can help identify spatial structure of homogenous areas. The footprint patchiness constraint is for interpretability. The last two constraints specify the assumptions that spatial ensemble relies on. First, nearby location should resemble each other (often true when pixels are much smaller than homogeneous parcel size). Second, test samples should be in the same spatial framework as training samples so that the model learned in a zone can be applied to test samples falling into it.

Illustrative example: Figure 4.3 illustrates the problem inputs and outputs. Inputs include input feature and training labels 4.3(a). The training samples (colored in red and green in "training labels" of Figure 4.3(a)) contain class ambiguity $FSAR = 0.3$, as shown in the global distribution of Figure 4.3(b) (e.g., samples with $f = 1$ have two different classes). Thus, the global decision tree produced prediction errors. In contrast, our spatial ensemble decomposes the spatial framework (Figure 4.3(c)) reducing class ambiguity to 0, as shown by local sample distributions in Figure 4.3(c). Predictions of local models show fewer errors.



(a) Mixture of experts approach



1	2	3	4	5	6	7	8	x/y
4	4	4	4	2	1	1	1	1
4	4	4	4	2	2	1	1	2
1	1	3	3	2	2	2	1	3
1	1	3	3	3	2	2	2	4
1	1	2	2	3	3	2	2	5
1	1	2	2	2	3	3	2	6
4	4	4	2	2	2	3	3	7
4	4	4	4	2	2	2	3	8

F	X	Y	C
4	3	1	red
1	6	1	red
4	1	2	red
1	8	2	red
3	4	3	green
2	6	3	green
1	1	4	green
2	7	4	green
3	6	5	green
1	2	6	green
4	1	7	red
2	5	7	red
4	3	8	red
2	7	8	red

(b) Adding spatial coordinate features

Figure 4.5: Illustrative examples of related work approaches (best viewed in color)

Comparison with related work: Given the same problem input as Figure 4.3(a), a

Mixture of Experts approach partitions the input data in feature vector space via a gating function. Results are illustrated in Figure 4.5(a), where feature f is partitioned into four zones, and zones with feature $f = 1$ and $f = 2$ have class ambiguity. The final predictions (Figure 4.5(a) right side) show errors in the upper right corner. Another related approach is to simply add spatial coordinates into feature vectors and then runs a global model or an ensemble model. However, this method is sensitive to training sample locations and may be insufficient to address the arbitrary shapes of local footprints, as illustrated in Figure 4.5(b).

4.3 Proposed Approach

Spatial ensemble learning, i.e., finding a decomposition of space to minimize class ambiguity, has three main challenges: large number of sample units; exponential possible ways of partitioning; and the computational cost of calculating class ambiguity in candidate evaluation. To address the first challenge, we propose to preprocess inputs to cluster all samples into homogeneous patches, and use patches as minimum units. To address the last two challenges, we propose a heuristic-based approach that first separates ambiguous pairs of patches into different footprints, and then grows these footprints according to spatial proximity.

4.3.1 Preprocessing: Patch Generation

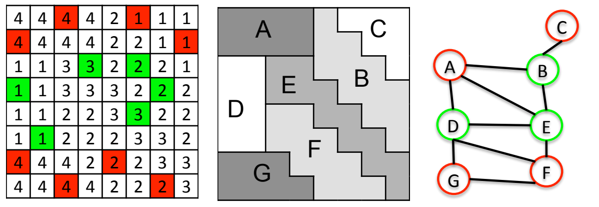
We begin by preprocessing input spatial data samples to generate homogeneous spatial clusters called *patches*, in order to reduce the number of combinations in the partitioning step. A patch is constrained to contain either no training samples or training samples from only one class. Real world geographic data often has a patch structure due to spatial autocorrelation, i.e., nearby locations often resemble each other [20]. For geographical raster data (e.g., remote sensing images), image segmentation [128] can be used with an additional constraint that each segment can have at most one type of training class label.

We provide a simple hierarchical algorithm (Algorithm 7). The algorithm inputs include all data samples (labeled and unlabeled), a neighborhood graph on samples (e.g., generated by a distance threshold or k-nearest-neighbors), and the number of patches

Algorithm 7 Preprocessing: Patch Generation

```
Input:
  • $s_i, i = 1..N$: labeled and unlabeled spatial data samples
  • $G$: neighborhood graph on samples
  • $n$: the number of patches, $n < N$
Output:
  • $P = \{P_j, j = 1..k\}$ homogeneous patches
1: Initialize with each sample as a patch $P_j, j = 1..N$
2: while number of patches $\leq n$ do
3:     Among neighboring patches without different training labels, find the pair with
       smallest feature distance
4:     Merge the neighboring patches into one patch
```

that controls spatial scale (a smaller number leading to larger patches). The algorithm starts with each sample as a patch, and then merge the most ”similar” (by feature distance) neighboring patches until the number of patches is reduced to the input value. An illustrative example is shown in Figure 4.6, where the samples have been merged into seven patches from A to G. The parameter of patch number can be determined according to the size of the study area (a large area requires more patches), usual size of parcels (large parcels require fewer patches), and the number of training samples (fewer training samples need fewer patches so that patches with training samples in it contain sufficient training samples). A median filter can be applied to sample feature maps before the algorithm to avoid outputting anomalously small patches due to noise.



(a) Spatial data samples (b) Patches generated (c) Neighbor graph

Figure 4.6: Illustrative example of patch generation

4.3.2 Zonal Footprint Generation

The second phase of spatial ensemble approach is spatial footprint generation. Given homogeneous patches from the preprocessing step, spatial footprint generation aims to divide these patches into two groups to minimize class ambiguity within each group while satisfying the spatial patchiness constraint for contiguity. The exponential number of possible partitions makes it computationally infeasible to enumerate through all partitions. Thus, we propose a heuristic based approach that first separates most ambiguous pairs of patches into two footprints (Seed Assignment phase), uses them as seeds to grow footprints spatially (Seed Growing phase), and finally adjust spatial outlier patches (in different footprints compared to most adjacent patches) to reduce patchiness of footprints (Spatial Adjustment phase).

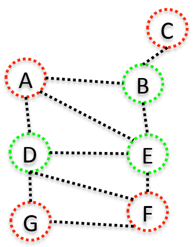
More details in Algorithm 8. The algorithm inputs include a set of patches (a patch can be positively labeled, negatively labeled, and unlabeled, according to the class of training sample within it), a spatial adjacency graph of the patches, and a threshold of maximum patchiness of two output footprints (to control spatial contiguity of footprints). The output is a partition of patches into two disjoint subsets as footprints.

Phase I: Seed Assignment. The Seed Assignment phase (lines 3 to 19 in Algorithm 8) computes class ambiguity scores ($a_{i,j}$)of pairs of positively labeled and negatively labeled patches. It then assigns ambiguous ($a_{i,j} > 0$) pairs into different footprints by a decreasing order of ambiguity scores (i.e., most ambiguous pair first). The assignment is conducted in various scenarios. At the beginning when two footprints are empty, two patches in the most ambiguous pair are randomly assigned to two footprints (lines 6 to 7). Later, the algorithm checks if any patch in an ambiguous pair is already within a footprint. If so, it assigns the other patch in the pair to a different footprint if it's not there already (lines 8 to 16). If neither patch in an ambiguous pair is within any footprint, the algorithm will assign the two patches to different footprints according to spatial proximity (lines 18 to 19). More specifically, in line 18, $d(p_i, S_k)$ is the adjacency graph distance from patch p_i to the closest patch in footprint S_k ($k \in \{1, 2\}$). Thus, the value of k in line 18 corresponds to the footprint to which assigning p_i minimizes the longest distance from the current two patches to their footprints.

Phase II: Seed Growing. After Phase I assigns ambiguous patches into different footprints. Phase II (lines 21 to 26) assigns any remaining patches. It uses previously assigned patches as footprint seeds, and grows the footprints on a patch adjacency graph. More specifically, at each step, the algorithm evaluates one remaining patch (from set R) that is adjacent to any footprint. The evaluation criterion is the improvement of training sample class balance if the patch is merged into its adjacent footprint. Class balance can be measured by Shannon entropy on ratios of training samples in different classes ($-r_1 \log r_1 - r_2 \log r_2$, where r_1, r_2 are the ratios of training samples from class 1 and 2). The intent is to encourage class balance of training samples within footprints during footprint expansion.

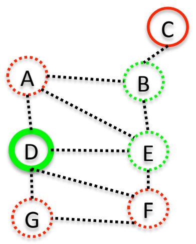
Phase III: Spatial Adjustment. The first two phases separate ambiguous patches into two footprints, and expand the footprints on a patch adjacency graph. The generated footprints S_1, S_2 may still lack spatial contiguity or have a patchiness score above the given minimum threshold. In other words, some patches may be assigned to different footprints from most of their adjacent patches (such patches can also be called "spatial outlier patches"). Thus, Phase III (lines 28 to 34) conducts a spatial adjustment on footprints produced from the previous phases to "smooth out" outlier patches. More specifically, line 28 checks if the current patch footprint map satisfies the patchiness (spatial contiguity) constraint (e.g., whether the number of connected components with the same footprint ids on a patch adjacent graph is smaller than a threshold). If the spatial constraint is not satisfied, the algorithm will compute local spatial autocorrelation statistics (e.g., local Moran's I, local Geary's C, local Gamma index) [212] of each patch to identify spatial outliers. We use the local Gamma index ($Gamma_i = \frac{\sum_j S_{i,j} W_{i,j}}{\sum_j W_{i,j}}$) where $S_{i,j}$ is similarity of footprint ids (e.g., its value is 1 if patch i and patch j belong to the same footprint) and $W_{i,j}$ is an element of the patch adjacency matrix (e.g., $W_{i,j} = 1$ if patch i and patch j are adjacent and $W_{i,j} = 0$ otherwise). Once spatial outlier patches in the footprint map are identified, the algorithm identifies the outlier patch whose reassignment to a new footprint (from S_k to S_{3-k} in line 32) creates the least class ambiguity increase. Such adjustment continues until the spatial contiguity constraint on footprint patchiness is satisfied.

Example: A running example of Algorithm 8 on the same toy dataset of Figure 4.6 is shown in Figure 4.7, where circles with dash line, solid line, and bold solid line represent patches with no footprint, with footprint 1, and with footprint 2 respectively. Assume the input patchiness threshold is at most two same-footprint-id connected components.

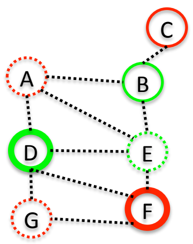


Patch i	Patch j	Ambiguity
C	D	0.5
F	B	0.5

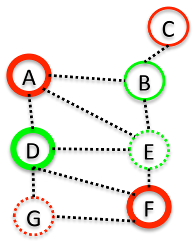
(a) Initial empty footprints



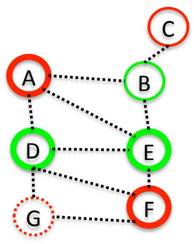
(b) Assign seed (C, D)



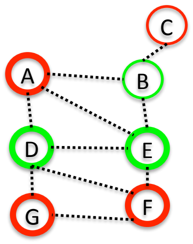
(c) Assign seed (E, F)



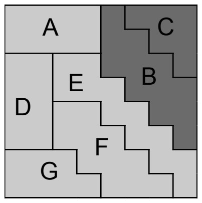
(d) seed growing A



(e) seed growing E



(f) seed growing G



(g) Final footprint map

Figure 4.7: Illustrative example of patch grouping: patches in solid line belong to footprint 1, patches in bold solid line belong to footprint 2 (best viewed in color)

Line 1 of Algorithm 8 initializes two empty footprints, when all patches are not assigned (with dash line in Figure 4.7(a)). Lines 3 to 4 in the Seed Assignment phase compute ambiguity score and identify two ambiguous pairs of patches ((C, D) and (F, B) as shown in the table of Figure 4.7(a)). After this, line 7 randomly assigns the ambiguous pair (C, D) into two footprints as shown in Figure 4.7(b). For the second ambiguous pair (F, B), neither patch belongs to any footprint, so line 18 computes which footprint assignment has minimum patch to footprint distances. It turns out that assigning patch B to the same footprint as C), and patch F to the same footprint as D has the minimum distance (shown in Figure 4.7(c)). Thus, in Phase I, patches C, B are assigned to footprint 1, patches D, F are assigned to footprint 2, while patches A, E, G are remaining. In Phase II (lines 21 to 26), the algorithm grows the two footprints on an adjacency graph to assign each remaining patch. Lines 23 to 24 compute the change-of-class balance when assigning any one patch among A, E, G to any adjacency footprint. For example, if A is assigned to footprint 2 (bold solid line), the training sample classes of footprint 2 will be changed from (two red, two green) to (four red, two green) according to Figure 4.6(a), so the entropy on class ratios will be changed from $-\frac{2}{4}\log\frac{2}{4} - \frac{2}{4}\log\frac{2}{4} = 1$ to $-\frac{4}{6}\log\frac{4}{6} - \frac{2}{6}\log\frac{2}{6} = 0.92$. Similarly, the class balance impacts of assigning A, E, G to any adjacent footprint are the same. So the algorithm can randomly assign A to footprint 2 (Figure 4.7(d)). After this, assigning E to footprint 2 will be best for class balance (Figure 4.7(e)). Finally, patch G is assign to its only adjacent footprint (footprint 2) (Figure 4.7(f)). Now, the footprint map contains two connected components, i.e., (B, C) and (A, D, E, F, G), as shown in Figure 4.7(g), so no spatial adjustment is needed.

4.4 Experimental Evaluation

The goal of the experiments was to investigate the following questions:

- How does the spatial ensemble approach compare with feature space ensemble (e.g., mixture of experts), bagging, and boosting in classification accuracy?
- How sensitive is the proposed approach to various parameter settings?
- Will adding spatial coordinate features always be sufficient in reducing class ambiguity? If not, in which case will it fail?
- How can the classification results be interpreted in real world case study?

4.4.1 Experiment Setup

Experiment design: The experiment design is shown in Figure 4.8. We compare our spatial ensemble method with feature vector space ensemble (mixture of experts), bagging and boosting. For mixture of expert method, we used a Matlab package for hierarchical mixture of experts with logistic regression local experts [253]. Other types of local experts beyond logistic regression was not used due to difficulty in finding open source codes. Bagging and boosting were from Weka toolbox [254]. Common parameters include the number and the type of base classifiers, as well as the size of training data. We used a spatial constraint in the spatial ensemble that the footprint map have at most 5 isolated parcels to maintain spatial contiguity. There is one additional parameter for spatial ensemble method that we tested, i.e., the number of patches in pre-processing step. We set the number of base classifiers to 2 for the spatial ensemble method, 4 for mixture of expert, and 100 for bagging and boosting. We tested the sensitivity of other parameters in evaluation.

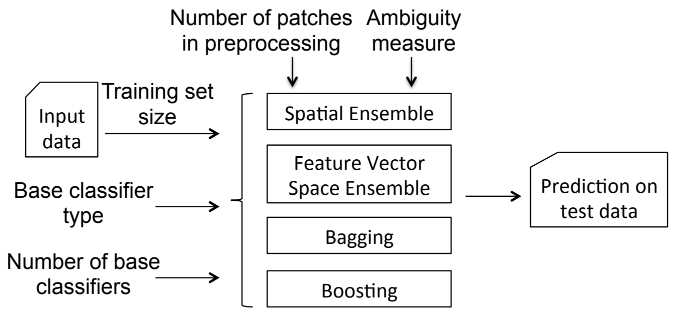


Figure 4.8: Experiment Setup

Dataset description: We used three high resolution (3m by 3m) remote sensing datasets from different study areas including Minnesota River Headwaters watershed, Swan Lake watershed, and the city of Chanhassen, MN [228]. Explanatory features include 4 spectral layers (R, G, B, Near Infrared or NIR) of aerial photos from National Agricultural Imagery Program during leaf-off season. Class labels (wetland and dry land) were from the updated National Wetland Inventory [255], which were conducted through a collaborative effort coordinated by the Minnesota Department of Natural Resources, MN. To evaluate the classification performance, we conducted random sampling to select a number of wetland and dry land contiguous clusters of pixels as the training set and the remaining pixels as test sets.

Evaluation metric: We evaluated the classification performance with confusion matrices, the precision and recall, as well as F-score. Since the application problem was wetland mapping, we considered the wetland class as the positive class.

4.4.2 Classification Performance Comparison

Comparison on Precision, Recall, F-score

The type of base classifier was decision trees (except for mixture of experts with logistic regression). The training set included 2434 wetland samples and 1758 dry land samples.

For the spatial ensemble approach, the number of patches in the preprocessing step was set to 200 for Chanhassen dataset and 1000 for the other two datasets, and the ambiguity measure was $FSAR$ (ratio of same class samples in K-nearest-neighbors) with $k = 10$.

Analysis of results: The classification performance on three datasets are summarized in Table 4.1, 4.2, 4.3 respectively. From the results, we can see that the mixture of experts approach generally has the worst accuracy on three datasets, probably due to limitation of its linear base model (logistic regression), and also due to its incapability of separate ambiguous samples in the feature vector space. Decision tree also has relatively low accuracy, due to the class ambiguity issue. Bagging and boosting are only slightly better than decision tree, since they cannot address the class ambiguity either. Spatial ensemble approach has the best overall performance, e.g., its F-score was around 0.91 (versus around 0.83 from boosting) on the first two datasets. One the third dataset, the gap between spatial ensemble and other approaches is smaller. The reason may be that some of the errors on the third dataset were not due to class ambiguity so spatial ensemble cannot improve those errors.

Table 4.1: Comparison of classification performance on Chanhassen Data

Ensemble Method	Confusion Matrix		Precision	Recall	F score
Single model	38567	6136	0.82	0.81	0.82
	6276	27483			
Bagging	39298	5405	0.84	0.82	0.83
	5939	27820			
Boosting	38579	6124	0.82	0.83	0.83
	5653	28106			
Mixture of Experts	36336	8307	0.75	0.75	0.75
	8306	25513			
Spatial Ensemble	40732	3971	**0.89**	**0.93**	**0.91**
	2421	31338			

Effect of the Number of Training Sample

The parameter settings were the same as those in Section IV.B.1). We used the Chanhassen dataset, and varied the number of training samples as 1444, 2857, and 4192, corresponding to 50, 100, and 150 circular clusters on class maps respectively.

Analysis of results: The classification performance on different training sample sizes

Table 4.2: Comparison of classification performance on Swan Lake Data

Ensemble Method	Confusion Matrix		Precision	Recall	F score
Single model	50073	6520	0.85	0.85	0.85
	6618	36860			
Bagging	50959	5634	0.87	0.86	0.86
	6207	37271			
Boosting	50478	6115	0.86	0.86	0.86
	5895	37583			
Mixture of Experts	50882	5711	0.85	0.74	0.79
	11135	32343			
Spatial Ensemble	55221	1372	**0.97**	**0.87**	**0.92**
	5445	38033			

Table 4.3: Comparison of classification performance on Big Stone Data

Ensemble Method	Confusion Matrix		Precision	Recall	F score
Single model	23990	5726	0.86	0.86	0.86
	6067	36216			
Bagging	24524	5192	**0.88**	0.86	0.87
	5869	36414			
Boosting	24273	5443	0.87	0.86	0.86
	6104	36179			
Mixture of Experts	23568	6148	0.85	0.80	0.82
	8290	33993			
Spatial Ensemble	23657	6059	0.86	**0.92**	**0.89**
	3555	38728			

is summarized in Figure 4.9. From the results, we can observe a similar trend as in Section IV.B.1), i.e., spatial ensemble consistently improved the accuracy over other ensemble methods. Future experiments on several more training sample sizes may be needed to observe a trend.

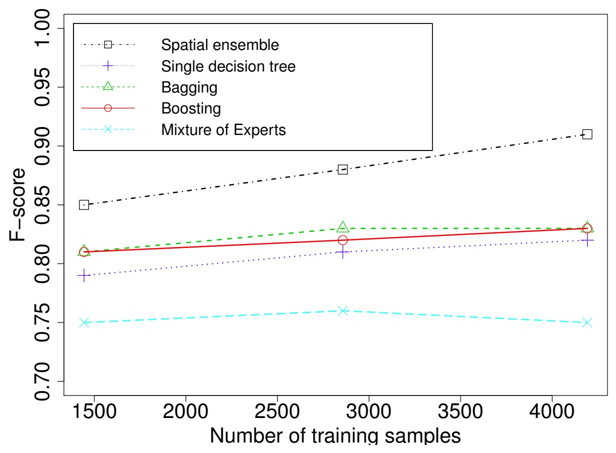


Figure 4.9: Effect of the number of training samples

Effect of Base Classifier Type

The training set included 1903 wetland samples and 2296 dry land samples drawn from the Chanhassen data. We compared approaches on three different types of base classifiers, i.e., decision tree, SVM, and neural network. The mixture of experts approach was excluded in this experiment due to difficulty in finding open source packages with the three base classifier types. The other parameters configurations were the same as Section IV.B.1). Results are shown in Figure 4.10. Spatial ensemble approach has the best accuracy for all three base classifier types.

Sensitivity of Spatial Ensemble Approach to Number of Patches in Preprocessing

We used the Chanhassen dataset and the same parameter settings as Section IV.B.1), except that we varied the number of patches in the preprocessing steps from 200 to 600. Results in Figure 4.11 showed that the performance of spatial ensemble approach was generally stable, with slightly lower accuracy when the number of patches was 300, but all of them were better than bagging, boosting and mixture of experts, whose F-scores were no higher than 0.83.

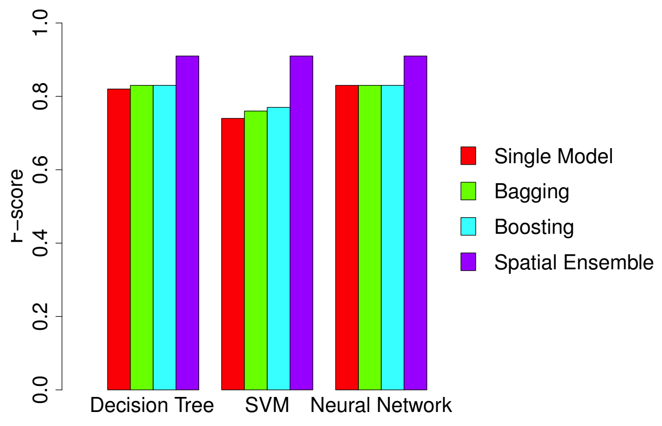


Figure 4.10: Effect of the base classifier type

Effect of Adding Spatial Coordinate Features

In this experiment, we investigated if adding spatial coordinates in feature vectors will always be effective in reducing class ambiguity. We used the Chanhassen data. The parameter settings were the same as Section IV.B.1) except that the training set size was smaller (624 wetland samples and 820 dry land samples, within 50 small circular clusters). Training sample locations were shown in Figure 4.12(a), where almost all training samples on the left half belonged to the dry land class (red). Due to this reason, decision tree and random forest models mistakenly "learned" that almost all samples in the left half should be predicted as dry land class (red). Thus, we can see that parts of wetland parcels in the left half of the image were misclassified (black errors in Figure 4.12(b-c)). Mixture of experts approach also made similar mistakes (black errors in Figure 4.12(d)), though the errors were slightly less serious. In contrast, spatial ensemble did not have same misclassification due to its more flexible spatial partition. The experiment showed that adding spatial coordinates in feature vectors in related work may not always be sufficient, particularly when sample locations are too sparse to capture the footprint shapes of class patches.

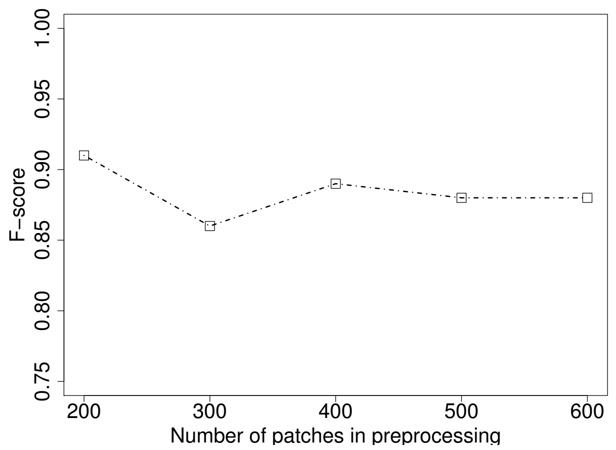
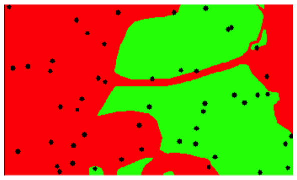


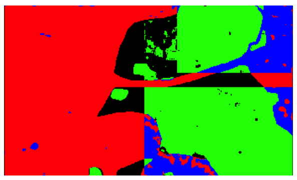
Figure 4.11: Sensitivity to number of patches in preprocessing

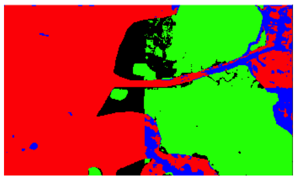
4.4.3 Case Studies

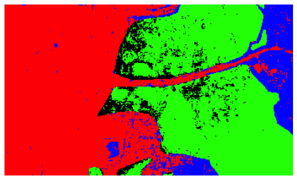
Figure 4.13 4.14 4.15 shows three case study for three different landscape areas in Minnesota including Chanhassen, Swan Lake, and Big Stone, and the results of the spatial ensemble approach were interpreted by domain experts in remote sensing and wetland mapping. The datasets and parameter configurations were the same as those in Section IV B 1). The input spectral image features, ground truth wetland class map, as well as output predictions from a single decision tree and spatial ensemble (SE) were all shown in the figure, numbered by different study areas.

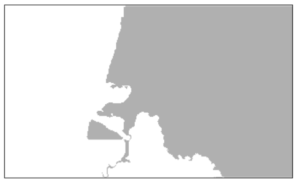
The three study areas in general show a good spectral separability for the SE prediction results (Figure 4.13(e), 4.14(e), 4.15(e)) between true dry land representing "red" that is uplands land cover and true wetland represented as "green" for wetlands land cover. On the other hand, there was higher spectral confusion when the Decision tree prediction (Figure 4.13(c), 4.14(c), 4.15(c)) was used compared to the SE prediction results. This spectral confusion can be explained primarily because of the different types of wetland and upland features found in these areas. For example, for the Chanhassen data (Figure 4.13(a)) two main different features were found as the main cause of spectral confusion: tree canopy vs. forested wetlands; these two features have different physical characteristics but similar spectral properties in the image data. This makes difficult to discriminate because a forested type of wetlands will appear cover


(a) Training samples on truth map


(b) Decision tree results


(c) Random forest results


(d) Mixture of expert results


(e) Spatial ensemble footprints

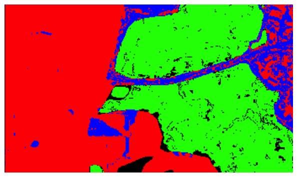

(f) Spatial ensemble results

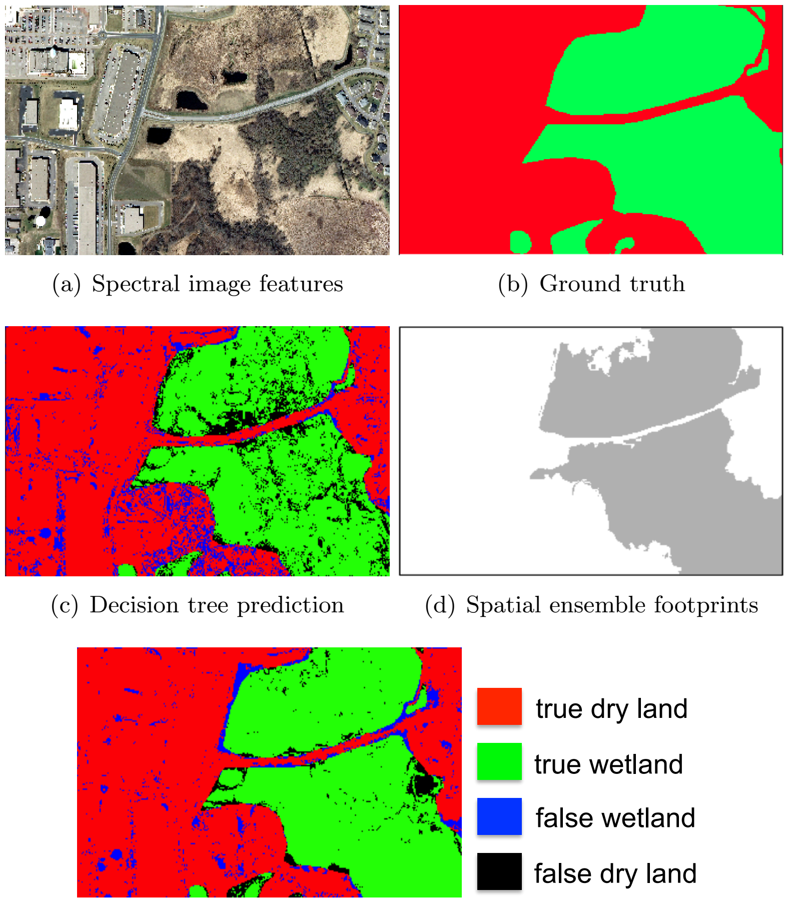
Figure 4.12: Comparison with related work adding spatial coordinate features in decision tree, random forests, and mixture of experts (black and blue are errors, best viewed in color)

with vegetation in the aerial imagery but in the real world it is very different compared to the regular tree canopy feature. Spatial ensemble footprints (Figure 4.13(d)) separated ambiguous areas into different local decision tree models, so there was less spectral confusion in each local model.

Similar situation happens for the case studies of Swan Lake and Big Stone areas where two different features (water with adjacent vegetation) can be seen in the same place but with different physical properties in the real world. Thus, a potential solution to this type of spectral confusion would be the use of topography data for better separation with the type of SE model used in this chapter. Topography plays a key role because the wetlands types found in this areas that tends to take place in topographic depressions where spectral data cannot discriminate well by itself.

4.5 Discussion

Our spatial ensemble approach addresses class ambiguity issue that is due to unknown heterogeneous geographical factors. We also assume that we have sufficient representative training samples to identify ambiguous subareas. There are other relevant



(a) Spectral image features (b) Ground truth

(c) Decision tree prediction (d) Spatial ensemble footprints

(e) Spatial ensemble prediction

Figure 4.13: Real world case study in Chanhassen (best viewed in color)

work to address spatial heterogeneity, including geographically weighted model such as GWR [232], Gaussian process [256], multi-task learning [257]. These methods do not focus on class ambiguity issues. One recent work [258] addresses class ambiguity in object classification, but not pixel-wise classification. There are also other ensemble learning methods [259] [260] that do not consider reducing class ambiguity in input data partitioning. These methods do not focus on class ambiguity issue either.

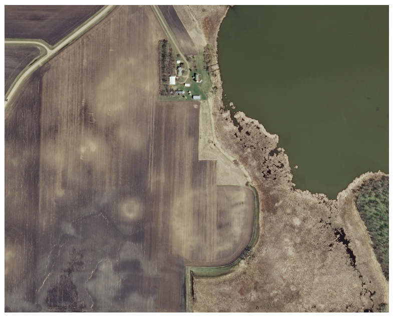
4.6 Summary

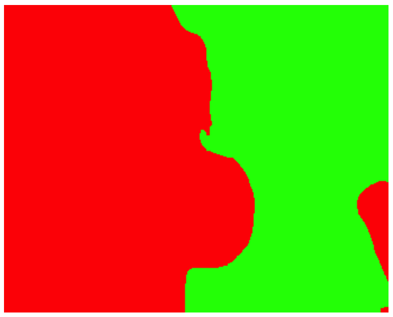
This chapter investigates spatial ensemble learning problem for geographical data with class ambiguity. The problem is important in many applications such as land cover classification in remote sensing, but is challenging due to unknown flexible footprint shapes, as well as potentially exponential number of possible partitions. To address these challenges, we proposed a spatial ensemble approach that first decomposes the space into homogeneous patches, and then groups patches using bottom-up greedy heuristic to separate out ambiguous pairs. Experimental evaluations on three real world remote sensing datasets show that our spatial ensemble approach outperforms other approaches in classification accuracy.

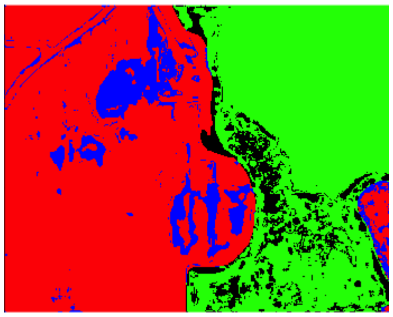
In future work, we plan to conduct more theoretical analysis on computational properties of the problem, e.g., NP-hardness. We also plan to explore other computational strategies, e.g., top-down spatial partitioning. We will also explore spatial ensemble methods for more than two footprints.

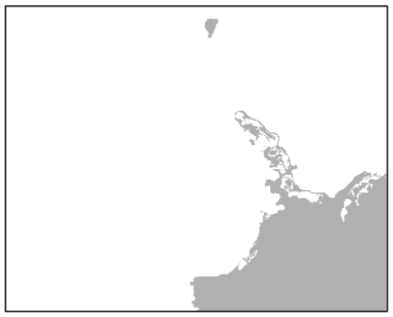
Algorithm 8 Patch Grouping

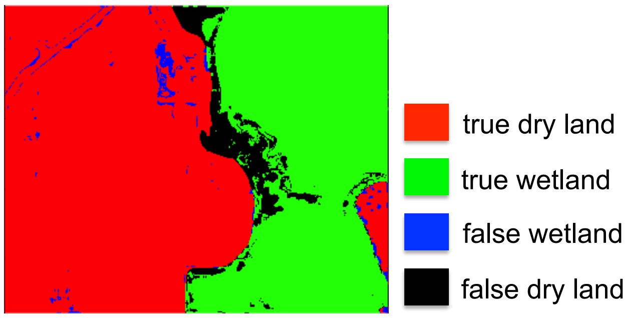
```
Input:
    • $P = \{p_i\} \cup \{n_j\} \cup \{u_k\}$: positively labeled patches $p_i$, negatively labeled patches
    $n_j$, and unlabeled patches $u_k$
    • $G$: spatial adjacency graph on patches
    • $\delta$: threshold of max patchiness of footprints
Output:
    • two sets $S_1, S_2$ such that $S_1 \cap S_2 = \emptyset, S_1 \cup S_2 = P$
 1: Initialize two footprints: $S_1 \leftarrow \emptyset$, $S_2 \leftarrow \emptyset$
 2: Part I: Seed Assignment
 3: Compute $a_{i,j} \leftarrow ambiguity(p_i \cup n_j)$ for any $i, j$
 4: Identify ambiguous pairs $(p_i, n_j)$ whose $a_{i,j} > 0$
 5: for each $(p_i, n_j)$ ordered by decreasing $a_{i,j}$ do
 6:   if $S_1 = \emptyset$ and $S_2 = \emptyset$ then
 7:     $S_1 \leftarrow \{p_i\}$, $S_2 \leftarrow \{n_j\}$
 8:   else if $p_i \in (S_1 \cup S_2)$ or $n_j \in (S_1 \cup S_2)$ then
 9:     if $p_i \in S_1$ and $n_j \notin S_2$ then
10:       $S_2 \leftarrow S_2 \cup \{n_j\}$
11:     else if $p_i \notin S_1$ and $n_j \in S_2$ then
12:       $S_1 \leftarrow S_1 \cup \{p_i\}$
13:     else if $n_j \in S_1$ and $p_i \notin S_2$ then
14:       $S_2 \leftarrow S_2 \cup \{p_i\}$
15:     else if $n_j \notin S_1$ and $p_i \in S_2$ then
16:       $S_1 \leftarrow S_1 \cup \{n_j\}$
17:   else
18:     $k \leftarrow \underset{k\in\{1,2\}}{\arg\min}\ max(d(p_i, S_k), d(n_j, S_{3-k}))$
19:     $S_k \leftarrow S_k \cup \{p_i\}$, $S_{3-k} \leftarrow S_{3-k} \cup \{n_j\}$
20: Part II: Seed Growing
21: $R \leftarrow P \setminus (S_1 \cup S_2)$
22: while $R \neq \emptyset$ do
23:   for $p \in R$ and $p$ adjacent to any $S_{k,k\in\{1,2\}}$ do
24:     $\Delta_p \leftarrow ClassBalance(S_k \cup p) - ClassBalance(S_k)$
25:   Find $p$ and its adjacent $S_k$ with max $\Delta_p$
26:   $S_k \leftarrow S_k \cup p$, $R \leftarrow R \setminus p$
27: Part III: Spatial Adjustment
28: while $Patchines(S_1, S_2, G) > \delta$ do
29:   for every $p \in S_{k,k\in\{1,2\}}$ do
30:     $LSA_p \leftarrow LocalSpatialAutocorrelation(p, S_1, S_2, G)$
31:     if $LSA_p < 0$ then
32:       $\Delta_p \leftarrow$ ambiguity$(S_{3-k} \cup \{p\})-$ambiguity$(S_{3-k})$
33:   find $p$ with smallest $\Delta_p$ among $LSA_p < 0$
34:   $S_k \leftarrow S_k \setminus \{p\}$, $S_{3-k} \leftarrow S_{3-k} \cup \{p\}$
```


(a) Spectral image


(b) Ground truth

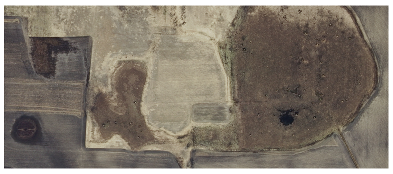

(c) Decision tree prediction

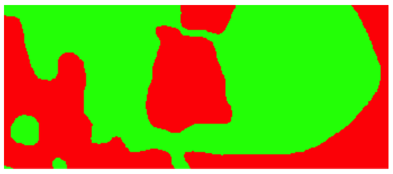

(d) Spatial ensemble footprints

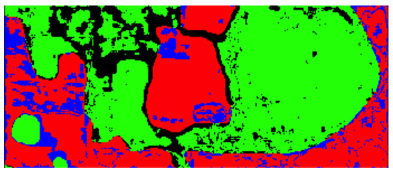


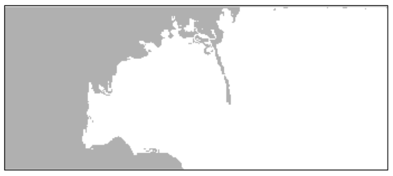
(e) Spatial ensemble prediction

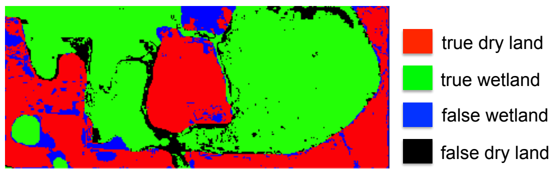
Figure 4.14: Real world case study in Swan Lake (best viewed in color)


(a) Spectral image features


(b) Ground truth


(c) Decision tree prediction


(d) Spatial ensemble footprints



(e) Spatial ensemble prediction

Figure 4.15: Real world case study in Big Stone (best viewed in color)

Chapter 5

Conclusion and Discussion

This thesis investigates novel spatial classification techniques to address unique challenges of spatial big data. Spatial big data, e.g., earth observation image, GPS trajectories, geo-referenced event reports, has potential to transform society in many applications such as precision agriculture, disease outbreak detection, and food-water-energy nexus, etc. However, spatial big data poses uniques challenges such as spatial autocorrelation, anisotropy, and heterogeneity. The thesis first surveys current techniques in spatial and spatiotemporal data mining. Compared with other surveys in literature, it focuses on spatial statistical foundations and categorizes computational approaches according to output pattern families. The thesis also introduces a novel spatial decision tree classification model to address the challenge of spatial autocorrelation and anisotropy, as well as a spatial ensemble learning framework to address the challenge of spatial heterogeneity. Evaluations on real world remote sensing datasets in wetland mapping applications show that proposed techniques outperform related work, e.g., in classification accuracy, and salt-and-pepper noise level.

Several directions should be explored in future work. First, novel classification techniques need be investigated for spatial big data with various spatial scales and resolutions, e.g., remote sensing imagery with resolutions from sub-meters to hundreds of meters. Utilizing all the data together can potentially improve predictive performance. Second, it is also important to develop other techniques to address the challenge of spatial heterogeneity, particularly for spatial variability in high resolution spatial big data for applications such as precision agriculture. High resolution aerial photos from unmanned aerial vehicles provide unique opportunities for early diagnosis for crop disease and nutrient adoption at sub-plot levels in agriculture. Finally, current big data analytics techniques are mostly empirical or data-driven, heavily focusing on identifying patterns or learning predictive models from data, but they often do not incorporate the laws of physics and common sense understanding. Thus, these techniques are prone to generate spurious patterns that require elimination using domain knowledge. Another future direction is to investigate the fusion of data-driven analytics and physics constraints to develop physics-aware spatial big data analytics that improve pattern interpretability and reduce spurious patterns.

References

[1] Bert Little, Michael Schucking, Brandon Gartrell, Bing Chen, Kenton Ross, and Rodney McKellip. High granularity remote sensing and crop production over space and time: Ndvi over the growing season and prediction of cotton yields at the farm field level in texas. In *ICDM Workshops*, pages 426–435, 2008.

[2] Shashi Shekhar, Pusheng Zhang, Yan Huang, and Ranga Raju Vatsavai. Trends in spatial data mining. *Data mining: Next generation challenges and future directions*, pages 357–380, 2003.

[3] S. Banerjee, B.P. Carlin, and A.E. Gelfand. *Hierarchical modeling and analysis for spatial data.* Chapman & Hall, 2004.

[4] O. Schabenberger and C.A. Gotway. *Statistical Methods for Spatial Data Analysis.* Chapman and Hall, 2005.

[5] Alan E Gelfand, Peter Diggle, Peter Guttorp, and Montserrat Fuentes. *Handbook of spatial statistics.* CRC Press, 2010.

[6] N. A. C. Cressie. *Statistics for Spatial Data.* Wiley, 1993.

[7] N.A. Cressie. *Statistics for Spatial Data (Revised Edition).* Wiley, New York, 1993.

[8] Peter W Gething, Peter M Atkinson, AM Noor, PW Gikandi, Simon I Hay, and Mark S Nixon. A local space–time kriging approach applied to a national outpatient malaria data set. *Computers & geosciences*, 33(10):1337–1350, 2007.

[9] C. E. Warrender and M. F. Augusteijn. Fusion of image classifications using Bayesian techniques with Markov rand fields. *International Journal of Remote Sensing*, 20(10):1987–2002, 1999.

[10] L. Anselin. Local indicators of spatial association-lisa. *Geographical Analysis*, 27(2):93–155, 1995.

[11] Stan Openshaw. *The modifiable areal unit problem.* ISBN 0860941345. OCLC, 1983.

[12] Brian D Ripley. Modelling spatial patterns. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 172–212, 1977.

[13] Eric Marcon, Florence Puech, et al. Generalizing ripley's k function to inhomogeneous populations. Technical report, mimeo, 2003.

[14] Martin Kulldorff. A spatial scan statistic. *Communications in Statistics-Theory and methods*, 26(6):1481–1496, 1997.

[15] Sung Nok Chiu, Dietrich Stoyan, Wilfrid S Kendall, and Joseph Mecke. *Stochastic geometry and its applications.* John Wiley & Sons, 2013.

[16] Xavier Guyon. *Random fields on a network: modeling, statistics, and applications.* Springer Science & Business Media, 1995.

[17] A. Okabe, H. Yomono, and M. Kitamura. Statistical analysis of the distribution of points on a network. *Geographical Analysis*, 27, 1995.

[18] Atsuyuki Okabe and Kokichi Sugihara. *Spatial analysis along networks: statistical and computational methods.* John Wiley & Sons, 2012.

[19] Atsuyuki Okabe, KeiIchi Okunuki, and Shino Shiode. The sanet toolbox: New methods for network spatial analysis. *Transactions in GIS*, 10(4):535–550, 2006.

[20] Waldo R Tobler. A computer movie simulating urban growth in the detroit region. *Economic geography*, 46:234–240, 1970.

[21] I. Vainer, Sarit Kraus, G.A. Kaminka, and H. Slovin. Scalable classification in large scale spatiotemporal domains applied to voltage-sensitive dye imaging. In *Data Mining, 2009. ICDM '09. Ninth IEEE International Conference on*, pages 543–551, Dec 2009.

[22] Michelangelo Ceci, Annalisa Appice, and Donato Malerba. Spatial associative classification at different levels of granularity: A probabilistic approach. In *PKDD*, pages 99–111, 2004.

[23] Wei Ding, Tomasz F Stepinski, and Josue Salazar. Discovery of geospatial discriminating patterns from remote sensing datasets. In *SDM*, pages 425–436. SIAM, 2009.

[24] Richard Frank, Martin Ester, and Arno J. Knobbe. A multi-relational approach to spatial classification. In *KDD*, pages 309–318, 2009.

[25] Michael D. Twa, Srinivasan Parthasarathy, Thomas W. Raasch, and Mark Bullimore. Decision tree classification of spatial data patterns from videokeratography using zernicke polynomials. In *SDM*, pages 3–12, 2003.

[26] Jin Li and Andrew D Heap. A review of comparative studies of spatial interpolation methods in environmental sciences: performance and impact factors. *Ecological Informatics*, 6(3):228–241, 2011.

[27] Shrutilipi Bhattacharjee, Pabitra Mitra, and Soumya K Ghosh. Spatial interpolation to predict missing attributes in GIS using semantic kriging. *IEEE Transactions on Geoscience and Remote Sensing*, 52(8):4771–4780, 2014.

[28] Avit Kumar Bhowmik and Pedro Cabral. Statistical evaluation of spatial interpolation methods for small-sampled region: a case study of temperature change phenomenon in bangladesh. In *Computational Science and Its Applications-ICCSA 2011*, pages 44–59. Springer, 2011.

[29] Sudipto Banerjee, Bradley P. Carlin, and Alan E. Gelfrand. *Hierarchical Modeling and Analysis for Spatial Data.* CRC Press, 2003.

[30] S. Li. A Markov Random Field Modeling. *Computer Vision (Publisher: Springer Verlag*, 1995.

[31] S. Shekhar, Paul R. Schrater, Ranga R. Vatsavai, Weili Wu, and S. Chawla. Spatial Contextual Classification and Prediction Models for Mining Geospatial Data. *IEEE Transactions on Multimedia*, 4(2), 2002.

[32] Chi-Hoon Lee, Russell Greiner, and Osmar R. Zaïane. Efficient spatial classification using decoupled conditional random fields. In *PKDD*, pages 272–283, 2006.

[33] L Anselin. *Spatial Econometrics: methods and models.* Kluwer, Dordrecht, Netherlands, 1988.

[34] S Chawla, S. Shekhar, W.-L. Wu, and U. Ozesmi. modeling spatial dependencies for mining geospatial data. In *ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery 2000: 70-77*, 2000.

[35] S. Chawla, S. Shekhar, W. Wu, and U. Ozesmi. Modeling spatial dependencies for mining geospatial data. *1st SIAM International Conference on Data Mining*, 2001.

[36] Alexander Liu, Goo Jun, and Joydeep Ghosh. Spatially cost-sensitive active learning. In *SDM*, pages 814–825. SIAM, 2009.

[37] AS Fotheringham, C. Brunsdon, and M. Charlton. *Geographically weighted regression: the analysis of spatially varying relationships.* Wiley, 2002.

[38] Karthik Subbian and Arindam Banerjee. Climate multi-model regression using spatial smoothing. In *SDM*, pages 324–332, 2013.

[39] P. Stolorz, H. Nakamura, E. Mesrobian, R.R. Muntz, E.C. Shek, J.R. Santos, J. Yi, K. Ng, S.Y. Chien, R. Mechoso, and J.D. Farrara. Fast Spatio-Temporal Data Mining of Large Geophysical Datasets. In *Proceedings of the First International Conference on Knowledge Discovery and Data Mining, AAAI Press, 300-305*, 1995.

[40] Shashi Shekhar and Sanjay Chawla. *Spatial Databases: A Tour.* Prentice Hall, 2003.

[41] Gary Marcus and Ernest Davis. Eight (no, nine!) problems with big data. *The New York Times*, 6(04):2014, 2014.

[42] Peter M Caldwell, Christopher S Bretherton, Mark D Zelinka, Stephen A Klein, Benjamin D Santer, and Benjamin M Sanderson. Statistical significance of climate sensitivity predictors obtained by data mining. *Geophysical Research Letters*, 41(5):1803–1808, 2014.

[43] Krzysztof Koperski, Junas Adhikary, and Jiawei Han. Spatial data mining: progress and challenges survey paper. In *Proc. ACM SIGMOD Workshop on Research Issues on Data Mining and Knowledge Discovery, Montreal, Canada*, pages 1–10. Citeseer, 1996.

[44] M. Ester, H.-P. Kriegel, and J. Sander. Spatial Data Mining: A Database Approach. In *Proc. Fifth Symposium on Rules in Geographic Information Databases*, 1997.

[45] Shashi Shekhar, Michael R Evans, James M Kang, and Pradeep Mohan. Identifying patterns in spatial information: A survey of methods. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 1(3):193–214, 2011.

[46] Shashi Shekhar, Zhe Jiang, Reem Y. Ali, Emre Eftelioglu, Xun Tang, Venkata M. V. Gunturi, and Xun Zhou. Spatiotemporal data mining: A computational perspective. *ISPRS International Journal of Geo-Information*, 4(4):2306, 2015.

[47] Harvey J. Miller and Jiawei Han. *Geographic Data Mining and Knowledge Discovery.* Taylor & Francis, Inc., Bristol, PA, USA, 2001.

[48] Harvey J Miller and Jiawei Han. *Geographic data mining and knowledge discovery.* CRC Press, 2009.

[49] Slava Kisilevich, Florian Mansmann, Mirco Nanni, and Salvatore Rinzivillo. *Spatio-temporal clustering.* Springer, 2010.

[50] Charu C Aggarwal. *Outlier analysis.* Springer Science & Business Media, 2013.

[51] Xun Zhou, Shashi Shekhar, and Reem Y Ali. Spatiotemporal change footprint pattern discovery: an inter-disciplinary survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 4(1):1–23, 2014.

[52] Michael Worboys and Matt Duckham. *GIS: A Computing Perspective. Second Edition.* CRC, 2004. ISBN: 978-0415283755.

[53] Zhilin Li, Jun Chen, and Emmanuel Baltsavias. *Advances in photogrammetry, remote sensing and spatial information sciences: 2008 ISPRS congress book*, volume 7. CRC Press, 2008.

[54] May Yuan. Temporal gis and spatio-temporal modeling. In *Proceedings of Third International Conference Workshop on Integrating GIS and Environment Modeling, Santa Fe, NM*, 1996.

[55] James F. Allen. Towards a general theory of action and time. *Artif. Intell.*, 23(2):123–154, 1984.

[56] B. George, S. Kim, and S. Shekhar. Spatio-temporal Network Databases and Routing Algorithms: A Summary of Results . In *Proceedings of International Symposium on Spatial and Temporal Databases (SSTD'07)*, Boston, MA, USA, July 2007.

[57] B. George and S. Shekhar. Time Aggregated Graphs: A model for spatio-temporal network. In *Proceedings of the Workshops (CoMoGIS) at the 25th International Conference on Conceptual Modeling (ER2006)*, Tucson, AZ, USA, 2006.

[58] Claudio EC Campelo and Brandon Bennett. *Representing and reasoning about changing spatial extensions of geographic features.* Springer, 2013.

[59] P.N. Tan, M. Steinbach, V. Kumar, et al. *Introduction to data mining.* Pearson Addison Wesley Boston, 2006.

[60] Paul Bolstad. *GIS Fundamentals: A First Text on GIS.* Eider Press, 2002.

[61] Auroop R. Ganguly and Karsten Steinhaeuser. Data mining for climate change and impacts. In *ICDM Workshops*, pages 385–394, 2008.

[62] Martin Erwig, Markus Schneider, and Fernuniversitat Hagen. Spatio-temporal predicates. *IEEE Transactions on Knowledge and Data Engineering*, 14:881–901, 2002.

[63] Jinyang Chen, Rangding Wang, Liangxu Liu, and Jiatao Song. Clustering of trajectories based on hausdorff distance. In *Electronics, Communications and Control (ICECC), 2011 International Conference on*, pages 1940–1944. IEEE, 2011.

[64] Zhang Zhang, Kaiqi Huang, and Tieniu Tan. Comparison of similarity measures for trajectory clustering in outdoor surveillance scenes. In *Pattern Recognition, 2006. ICPR 2006. 18th International Conference on*, volume 3, pages 1135–1138. IEEE, 2006.

[65] Pusheng Zhang, Yan Huang, Shashi Shekhar, and Vipin Kumar. Correlation analysis of spatial time series datasets: A filter-and-refine approach. In *Advances in Knowledge Discovery and Data Mining*, pages 532–544. Springer, 2003.

[66] Jaya Kawale, Snigdhansu Chatterjee, Dominick Ormsby, Karsten Steinhaeuser, Stefan Liess, and Vipin Kumar. Testing the significance of spatio-temporal teleconnection patterns. In *KDD*, pages 642–650, 2012.

[67] Mete Celik, Shashi Shekhar, James P. Rogers, James A. Shine, and Jin Soung Yoo. Mixed-drove spatio-temporal co-occurence pattern mining: A summary of results. In *ICDM '06: Proceedings of the Sixth International Conference on Data Mining*, pages 119–128, Washington, DC, USA, 2006. IEEE Computer Society.

[68] Karthik Ganesan Pillai, Rafal A. Angryk, and Berkay Aydin. A filter-and-refine approach to mine spatiotemporal co-occurrences. In *SIGSPATIAL/GIS*, pages 104–113, 2013.

[69] Pradeep Mohan, Shashi Shekhar, James A. Shine, and James P. Rogers. Cascading spatio-temporal pattern discovery. *IEEE Trans. Knowl. Data Eng.*, 24(11):1977–1992, 2012.

[70] Pradeep Mohan, Shashi Shekhar, James A. Shine, and James P. Rogers. Cascading spatio-temporal pattern discovery: A summary of results. In *SDM*, pages 327–338, 2010.

[71] Yan Huang, Liqin Zhang, and Pusheng Zhang. A framework for mining sequential patterns from spatio-temporal event data sets. *IEEE Transactions on Knowledge and Data Engineering*, 20(4):433–448, 2008.

[72] Yan Huang, Liqin Zhang, and Pusheng Zhang. Finding sequential patterns from a massive number of spatio-temporal events. In *SDM*, pages 634–638, 2006.

[73] Jeremy Mennis, Roland Viger, and C Dana Tomlin. Cubic map algebra functions for spatio-temporal analysis. *Cartography and Geographic Information Science*, 32(1):17–32, 2005.

[74] Daniel G Brown, Rick Riolo, Derek T Robinson, Michael North, and William Rand. Spatial process and data models: Toward integration of agent-based models and gis. *Journal of Geographical Systems*, 7(1):25–47, 2005.

[75] J. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers, 1993.

[76] V. Varnett and T. Lewis. *Outliers in Statistical Data*. John Wiley, 1994.

[77] T; Agarwal, R;Imielinski and A Swami. Mining association rules between sets of items in large databases. In *Proc. of the ACM SIGMOD Conference on Management of Data, Washington, D.C.*, may 1993.

[78] R Agrawal and R Srikant. Fast algorithms for Mining Association Rules. In *Proc. of Very Large Databases*, may 1994.

[79] A. Jain and R. Dubes. *Algorithms for Clustering Data*. Prentice Hall, 1988.

[80] Noel Cressie and Christopher K Wikle. *Statistics for spatio-temporal data*. John Wiley & Sons, 2011.

[81] Robert H Shumway and David S Stoffer. *Time series analysis and its applications: with R examples*. Springer Science & Business Media, 2010.

[82] Phaedon C Kyriakidis and André G Journel. Geostatistical space–time models: a review. *Mathematical geology*, 31(6):651–684, 1999.

[83] Noel A.C.Cressie. *Statistics for Spatial Data* . Wiley-Interscience, 1993. ISBN: 978-0471002550.

[84] V Barnett and T Lewis. *Outliers in Statistical Data*. John Wiley, 3rd edition edition, 1994.

[85] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM Comput. Surv.*, 41(3):15:1–15:58, July 2009.

[86] S. Shekhar, C.T. Lu, and P. Zhang. A unified approach to detecting spatial outliers. *GeoInformatica*, 7(2), 2003.

[87] J. Haslett, R. Bradley, P. Craig, A. Unwin, and G. Wills. Dynamic graphics for exploring spatial data with application to locating global and local anomalies. *American Statistician*, pages 234–242, 1991.

[88] Anselin Luc. Exploratory Spatial Data Analysis and Geographic Information Systems. In M. Painho, editor, *New Tools for Spatial Analysis*, pages 45–54, 1994.

[89] Dechang Chen, Chang-Tien Lu, Yufeng Kou, and Feng Chen. On detecting spatial outliers. *GeoInformatica*, 12(4):455–475, 2008.

[90] Chang-Tien Lu, Dechang Chen, and Yufeng Kou. Detecting spatial outliers with multiple attributes. In *ICTAI '03: Proceedings of the 15th IEEE International Conference on Tools with Artificial Intelligence*, page 122, Washington, DC, USA, 2003. IEEE Computer Society.

[91] Yufeng Kou, Chang-Tien Lu, and Dechang Chen. Spatial weighted outlier detection. In *SDM*, pages 614–618, 2006.

[92] Xutong Liu, Feng Chen, and Chang-Tien Lu. On detecting spatial categorical outliers. *GeoInformatica*, 18(3):501–536, 2014.

[93] Erich Schubert, Arthur Zimek, and Hans-Peter Kriegel. Local outlier detection reconsidered: a generalized view on locality with applications to spatial, video, and network outlier detection. *Data Min. Knowl. Discov.*, 28(1):190–237, 2014.

[94] Mingxi Wu, Chris Jermaine, Sanjay Ranka, Xiuyao Song, and John Gums. A model-agnostic framework for fast spatial anomaly detection. *TKDD*, 4(4):20, 2010.

[95] James M. Kang, Shashi Shekhar, Christine Wennen, and Paige Novak. Discovering Flow Anomalies: A SWEET Approach. In *International Conference on Data Mining*, 2008.

[96] Yan Huang, Shashi Shekhar, and Hui Xiong. Discovering co-location patterns from spatial datasets: A general approach. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 16(12):1472–1485, 2004.

[97] Mete Celik, Shashi Shekhar, James P. Rogers, and James A. Shine. Mixed-drove spatiotemporal co-occurrence pattern mining. *IEEE Transactions on Knowledge and Data Engineering*, 20(10):1322–1335, 2008.

[98] Y. Chou. *Exploring Spatial Analysis in Geographic Information System.* Onward Press, 1997.

[99] K. Koperski and J. Han. Discovery of Spatial Association Rules in Geographic Information Databases. In *Proc. Fourth International Symposium on Large Spatial Databases, Maine. 47-66*, 1995.

[100] Y Morimoto. Mining frequent neighboring class sets in spatial databases. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2001.

[101] Hui Xiong, Shashi Shekhar, Yan Huang, Vipin Kumar, Xiaobin Ma, and Jin Soung Yoo. A framework for discovering co-location patterns in data sets with extended spatial objects. In *SDM*, pages 78–89, 2004.

[102] Yan Huang, Jian Pei, and Hui Xiong. Mining co-location patterns with rare events from spatial data sets. *GeoInformatica*, 10(3):239–260, 2006.

[103] Song Wang, Yan Huang, and Xiaoyang Sean Wang. Regional co-locations of arbitrary shapes. In *SSTD*, pages 19–37, 2013.

[104] Wei Ding, Christoph F. Eick, Xiaojing Yuan, Jing Wang, and Jean-Philippe Nicot. A framework for regional association rule mining and scoping in spatial datasets. *GeoInformatica*, 15(1):1–28, 2011.

[105] Sajib Barua and Jörg Sander. Mining statistically significant co-location and segregation patterns. *IEEE Trans. Knowl. Data Eng.*, 26(5):1185–1199, 2014.

[106] Jin Soung Yoo and Shashi Shekhar. A joinless approach for mining spatial colocation patterns. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 18(10), 2006.

[107] Huiping Cao, Nikos Mamoulis, and David W. Cheung. Discovery of collocation episodes in spatiotemporal data. In *ICDM*, pages 823–827, 2006.

[108] Huiping Cao, Nikos Mamoulis, and David W. Cheung. Mining frequent spatio-temporal sequential patterns. In *ICDM*, pages 82–89, 2005.

[109] Florian Verhein. Mining complex spatio-temporal sequence patterns. In *SDM*, pages 605–616, 2009.

[110] Lu An Tang, Yu Zheng, Jing Yuan, Jiawei Han, Alice Leung, Wen-Chih Peng, and Thomas F. La Porta. A framework of traveling companion discovery on trajectory data streams. *ACM TIST*, 5(1):3, 2013.

[111] Amy McGovern, Nathaniel Troutman, Rodger A. Brown, John K. Williams, and Jennifer Abernethy. Enhanced spatiotemporal relational probability trees and forests. *Data Min. Knowl. Discov.*, 26(2):398–433, 2013.

[112] Jae-Gil Lee, Jiawei Han, Xiaolei Li, and Hong Cheng. Mining discriminative patterns for classifying trajectories on road networks. *Knowledge and Data Engineering, IEEE Transactions on*, 23(5):713–726, May 2011.

[113] Anastasios Noulas, Salvatore Scellato, Neal Lathia, and Cecilia Mascolo. Mining user mobility features for next place prediction in location-based services. In *ICDM*, pages 1038–1043, 2012.

[114] Josh Jia-Ching Ying, Wang-Chien Lee, and Vincent S. Tseng. Mining geographic-temporal-semantic patterns in trajectories for location prediction. *ACM TIST*, 5(1):2, 2013.

[115] Hong Cheng, Jihang Ye, and Zhe Zhu. What's your next move: User activity prediction in location-based social networks. In *SDM*, pages 171–179, 2013.

[116] Jia-Dong Zhang and Chi-Yin Chow. iGSLR: personalized geo-social location recommendation: a kernel density estimation approach. In *SIGSPATIAL/GIS*, pages 324–333, 2013.

[117] Bin Liu, Yanjie Fu, Zijun Yao, and Hui Xiong. Learning geographical preferences for point-of-interest recommendation. In *KDD*, pages 1043–1051, 2013.

[118] Yu Zheng and Xing Xie. Learning travel recommendations from user-generated GPS traces. *ACM TIST*, 2(1):2, 2011.

[119] Hao Wang, Manolis Terrovitis, and Nikos Mamoulis. Location recommendation in location-based social networks using user check-in data. In *SIGSPATIAL/GIS*, pages 364–373, 2013.

[120] Jie Bao, Yu Zheng, and Mohamed F. Mokbel. Location-based and preference-aware recommendation using sparse geo-social networking data. In *SIGSPATIAL/GIS*, pages 199–208, 2012.

[121] J. Han, M. Kamber, and A. K. H. Tung. Spatial Clustering Methods in Data Mining: A Survey. In *Geographic Data Mining and Knowledge Discovery*. Taylor and Francis, 2001.

[122] George Karypis, Eui-Hong Han, and Vipin Kumar. Chameleon: Hierarchical clustering using dynamic modeling. *IEEE Computer*, 32(8):68–75, 1999.

[123] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd*, volume 96, pages 226–231, 1996.

[124] Raymond A Jarvis and Edward A Patrick. Clustering using a similarity measure based on shared near neighbors. *Computers, IEEE Transactions on*, 100(11):1025–1034, 1973.

[125] M.F. Worboys. *GIS: A Computing Perspective.* Taylor and Francis, 1995.

[126] Deepti Joshi, Ashok Samal, and Leen-Kiat Soh. A dissimilarity function for clustering geospatial polygons. In *Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 384–387. ACM, 2009.

[127] Sujing Wang and Christoph F Eick. A polygon-based clustering and analysis framework for mining spatial datasets. *GeoInformatica*, 18(3):569–594, 2014.

[128] Robert M Haralick and Linda G Shapiro. Image segmentation techniques. In *1985 Technical Symposium East*, pages 2–9. International Society for Optics and Photonics, 1985.

[129] KwangSoo Yang, Apurv Hirsh Shekhar, Dev Oliver, and Shashi Shekhar. Capacity-constrained network-voronoi diagram. *IEEE Trans. Knowl. Data Eng.*, 27(11):2919–2932, 2015.

[130] George Karypis. Multi-constraint mesh partitioning for contact/impact computations. In *Proceedings of the 2003 ACM/IEEE conference on Supercomputing*, page 56. ACM, 2003.

[131] Deepti Joshi, Ashok Samal, and Leen-Kiat Soh. Spatio-temporal polygonal clustering with space and time as first-class citizens. *GeoInformatica*, 17(2):387–412, 2013.

[132] Derya Birant and Alp Kut. St-dbscan: An algorithm for clustering spatial–temporal data. *Data & Knowledge Engineering*, 60(1):208–221, 2007.

[133] Min Wang, Aiping Wang, and Anbo Li. Mining spatial-temporal clusters from geo-databases. In *Advanced Data Mining and Applications*, pages 263–270. Springer, 2006.

[134] T Warren Liao. Clustering of time series data—a survey. *Pattern recognition*, 38(11):1857–1874, 2005.

[135] Jae-Gil Lee, Jiawei Han, and Kyu-Young Whang. Trajectory clustering: a partition-and-group framework. In *Proceedings of the 2007 ACM SIGMOD international conference on Management of data*, pages 593–604. ACM, 2007.

[136] Zhenjie Zhang, Yin Yang, Anthony KH Tung, and Dimitris Papadias. Continuous k-means monitoring over moving objects. *Knowledge and Data Engineering, IEEE Transactions on*, 20(9):1205–1216, 2008.

[137] Christian S Jensen, Dan Lin, and Beng Chin Ooi. Continuous clustering of moving objects. *Knowledge and Data Engineering, IEEE Transactions on*, 19(9):1161–1174, 2007.

[138] Anthony JT Lee, Yi-An Chen, and Weng-Chong Ip. Mining frequent trajectory patterns in spatial–temporal databases. *Information Sciences*, 179(13):2218–2231, 2009.

[139] Varun Chandola and Vipin Kumar. Summarization–compressing data into an informative representation. *Knowledge and Information Systems*, 12(3):355–378, 2007.

[140] Dev Oliver, Shashi Shekhar, James M Kang, Renee Laubscher, Veronica Carlan, and Abdussalam Bannur. A k-main routes approach to spatial network activity summarization. *Knowledge and Data Engineering, IEEE Transactions on*, 26(6):1464–1478, 2014.

[141] Bei Pan, Ugur Demiryurek, Farnoush Banaei-Kashani, and Cyrus Shahabi. Spatiotemporal summarization of traffic data streams. In *Proceedings of the ACM SIGSPATIAL International Workshop on GeoStreaming*, pages 4–10. ACM, 2010.

[142] Michael R Evans, Dev Oliver, Shashi Shekhar, and Francis Harvey. Summarizing trajectories into k-primary corridors: a summary of results. In *Proceedings of the 20th International Conference on Advances in Geographic Information Systems*, pages 454–457. ACM, 2012.

[143] Zhe Jiang, Michael Evans, Dev Oliver, and Shashi Shekhar. Identifying k primary corridors from urban bicycle gps trajectories on a road network. *Information Systems (to appear)*, pages –, 2015.

[144] M Kulldorff. Satscan user guide for version 9.0, 2011.

[145] Ned Levine. *CrimeStat 3.0: A Spatial Statistics Program for the Analysis of Crime Incident Locations*. Ned Levine & Associatiates: Houston, TX / National Institute of Justice: Washington, DC, 2004.

[146] Emre Eftelioglu, Shashi Shekhar, Dev Oliver, Xun Zhou, Michael R. Evans, Yiqun Xie, James M. Kang, Renee Laubscher, and Christopher Farah. Ring-shaped hotspot detection: A summary of results. In *2014 IEEE International Conference on Data Mining, ICDM 2014, Shenzhen, China, December 14-17, 2014*, pages 815–820, 2014.

[147] Toshiro Tango, Kunihiko Takahashi, and Kazuaki Kohriyama. A space–time scan statistic for detecting emerging outbreaks. *Biometrics*, 67(1):106–115, 2011.

[148] Daniel B Neill and Andrew W Moore. A fast multi-resolution method for detection of significant spatial disease clusters. In *Advances in Neural Information Processing Systems*, page None, 2003.

[149] Jerry Ratcliffe. Crime mapping: spatial and temporal challenges. In *Handbook of quantitative criminology*, pages 5–24. Springer, 2010.

[150] Anselin Luc. Local Indicators of Spatial Association: LISA. *Geographical Analysis*, 27(2):93–115, 1995.

[151] Nakarin Chaikaew, Nitin K Tripathi, and Marc Souris. International journal of health geographics. *International Journal of Health Geographics*, 8:36, 2009.

[152] Sudarshan S Chawathe. Organizing hot-spot police patrol routes. In *Intelligence and Security Informatics, 2007 IEEE*, pages 79–86. IEEE, 2007.

[153] Mete Celik, Shashi Shekhar, Betsy George, James P. Rogers, and James A. Shine. Discovering and quantifying mean streets: A summary of results. Technical Report 025, University of Minnesota, 07 2007.

[154] S. Shiode and A. Okabe. Network variable clumping method for analyzing point patterns on a network. In *Unpublished paper presented at the Annual Meeting of the Associations of American Geographers*, Philadelphia, Pennsylvania, 2004.

[155] Wei Chang, Daniel Zeng, and Hsinchun Chen. Prospective spatio-temporal data analysis for security informatics. In *Intelligent Transportation Systems, 2005. Proceedings. 2005 IEEE*, pages 1120–1124. IEEE, 2005.

[156] D.B. Neill, A.W. Moore, M. Sabhnani, and K. Daniel. Detection of emerging space-time clusters. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 218–227. ACM.

[157] V. Chandola, D. Hui, L. Gu, B. Bhaduri, and R.R. Vatsavai. Using time series segmentation for deriving vegetation phenology indices from modis ndvi data, 2010.

[158] M. Worboys and M. Duckham. *GIS: A computing perspective.* CRC, ISBN:0415283752., 2004.

[159] Florentin Bujor, Emmanuel Trouvé, Lionel Valet, J-M Nicolas, and J-P Rudant. Application of log-cumulants to the detection of spatiotemporal discontinuities in multitemporal sar images. *Geoscience and Remote Sensing, IEEE Transactions on*, 42(10):2073–2084, 2004.

[160] Yukio Kosugi, Mitsuteru Sakamoto, Munenori Fukunishi, Wei Lu, Takeshi Doihara, and Shigeru Kakumoto. Urban change detection related to earthquakes using an adaptive nonlinear mapping of high-resolution images. *Geoscience and Remote Sensing Letters, IEEE*, 1(3):152–156, 2004.

[161] Gerardo Di Martino, Antonio Iodice, Daniele Riccio, and Giuseppe Ruello. A novel approach for disaster monitoring: fractal models and tools. *Geoscience and Remote Sensing, IEEE Transactions on*, 45(6):1559–1570, 2007.

[162] R.J. Radke, S. Andra, O. Al-Kofahi, and B. Roysam. Image change detection algorithms: a systematic survey. *Image Processing, IEEE Transactions on*, 14(3):294–307, 2005.

[163] R. Thoma and M. Bierling. Motion compensating interpolation considering covered and uncovered background. *Signal Processing: Image Communication*, 1(2):191–212, 1989.

[164] T. Aach and A. Kaup. Bayesian algorithms for adaptive change detection in image sequences using markov random fields. *Signal Processing: Image Communication*, 7(2):147–160, 1995.

[165] Gang Chen, Geoffrey J Hay, Luis MT Carvalho, and Michael A Wulder. Object-based change detection. *International Journal of Remote Sensing*, 33(14):4434–4457, 2012.

[166] Baudouin Desclee, Patrick Bogaert, and Pierre Defourny. Forest change detection by statistical object-based method. *Remote Sensing of Environment*, 102(1):1–11, 2006.

[167] J Im, JR Jensen, and JA Tullis. Object?based change detection using correlation image analysis and image segmentation. *International Journal of Remote Sensing*, 29(2):399–423, 2008.

[168] Til Aach, André Kaup, and Rudolf Mester. Statistical model-based change detection in moving video. *Signal processing*, 31(2):165–180, 1993.

[169] Eric JM Rignot and Jacob J van Zyl. Change detection techniques for ers-1 sar data. *Geoscience and Remote Sensing, IEEE Transactions on*, 31(4):896–906, 1993.

[170] J. Im and J.R. Jensen. A change detection model based on neighborhood correlation image analysis and decision tree classification. *Remote Sensing of Environment*, 99(3):326–340, 2005.

[171] Y. Yakimovsky. Boundary and object detection in real world images. *Journal of the ACM (JACM)*, 23(4):599–618, 1976.

[172] David H Douglas and Thomas K Peucker. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *Cartographica: The International Journal for Geographic Information and Geovisualization*, 10(2):112–122, 1973.

[173] M. Kulldorff, WF Athas, EJ Feurer, BA Miller, and CR Key. Evaluating cluster alarms: a space-time scan statistic and brain cancer in los alamos, new mexico. *American Journal of Public Health*, 88(9):1377–1380, 1998.

[174] M. Kulldorff. Prospective time periodic geographical disease surveillance using a scan statistic. *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, 164(1):61–72, 2001.

[175] Daniel J Isaak, Erin E Peterson, Jay M Ver Hoef, Seth J Wenger, Jeffrey A Falke, Christian E Torgersen, Colin Sowder, E Ashley Steel, Marie-Josee Fortin, Chris E Jordan, et al. Applications of spatial statistical network models to stream data. *Wiley Interdisciplinary Reviews: Water*, 1(3):277–294, 2014.

[176] Dev Oliver, Abdussalam Bannur, James M Kang, Shashi Shekhar, and Renee Bousselaire. A k-main routes approach to spatial network activity summarization: A summary of results. In *Data Mining Workshops (ICDMW), 2010 IEEE International Conference on*, pages 265–272. IEEE, 2010.

[177] Venkata M. V. Gunturi and Shashi Shekhar. Lagrangian xgraphs: A logical data-model for spatio-temporal network data: A summary. In *Advances in Conceptual Modeling - ER 2014 Workshops, ENMO, MoBiD, MReBA, QMMQ, SeCoGIS, WISM, and ER Demos, Atlanta, GA, USA, October 27-29, 2014. Proceedings*, pages 201–211, 2014.

[178] Venkata M.V. Gunturi, Ernesto Nunes, KwangSoo Yang, and Shashi Shekhar. A critical-time-point approach to all-start-time lagrangian shortest paths: A summary of results. In Dieter Pfoser, Yufei Tao, Kyriakos Mouratidis, MarioA. Nascimento, Mohamed Mokbel, Shashi Shekhar, and Yan Huang, editors, *Advances in Spatial and Temporal Databases*, volume 6849 of *Lecture Notes in Computer Science*, pages 74–91. Springer Berlin Heidelberg, 2011.

[179] V.M.V. Gunturi, S. Shekhar, and K. Yang. A critical-time-point approach to all-departure-time lagrangian shortest paths. *IEEE Transactions on Knowledge and Data Engineering*, PP(99):1–1, 2015.

[180] J. Speed. Iot for v2v and the connected car. `www.slideshare.net/JoeSpeed/aw-megatrends-2014-joe-speed`.

[181] Reem Y. Ali, Venkata M.V. Gunturi, A Kotz, S. Shekhar, and W. Northrop. Discovering non-compliant window co-occurrence patterns: A summary of results. In *Accepted in 14th Intl. Symp. on Spatial and Temporal Databases*. 2015.

[182] ESRI. Breathe Life into Big Data: ArcGIS Tools and Hadoop Analyze Large Data Stores. `http://www.esri.com/esri0news/arcnews/summer13articles/breathe0life0into0big0data!`

[183] ESRI. ESRI: GIS and Mapping Software. `www.esri.com`.

[184] Ablimit Aji, Fusheng Wang, Hoang Vo, Rubao Lee, Qiaoling Liu, Xiaodong Zhang, and Joel Saltz. Hadoop gis: a high performance spatial data warehousing system over mapreduce. *Proceedings of the VLDB Endowment*, 6(11):1009–1020, 2013.

[185] Ahmed Eldawy and Mohamed F Mokbel. Spatialhadoop: A mapreduce framework for spatial data. In *Proceedings of the IEEE International Conference on Data Engineering (ICDE'15). IEEE*, 2015.

[186] Ching Avery. Giraph: Large-scale graph processing infrastructure on hadoop. *Proceedings of the Hadoop Summit. Santa Clara*, 2011.

[187] Yucheng Low, Joseph E Gonzalez, Aapo Kyrola, Danny Bickson, Carlos E Guestrin, and Joseph Hellerstein. Graphlab: A new framework for parallel machine learning. *arXiv preprint arXiv:1408.2041*, 2014.

[188] Grzegorz Malewicz, Matthew H Austern, Aart JC Bik, James C Dehnert, Ilan Horn, Naty Leiser, and Grzegorz Czajkowski. Pregel: a system for large-scale graph processing. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*, pages 135–146. ACM, 2010.

[189] J. R. Quinlan. *C4.5: programs for machine learning.* Morgan kaufmann, 1993.

[190] Zhe Jiang, Shashi Shekhar, Pradeep Mohan, Joseph Knight, and Jennifer Corcoran. Learning spatial decision tree for geographical classification: a summary of results. In *SIGSPATIAL/GIS*, pages 390–393, 2012.

[191] Mark A Friedl and Carla E Brodley. Decision tree classification of land cover from remotely sensed data. *Remote sensing of environment*, 61(3):399–409, 1997.

[192] Jennifer M Corcoran, Joseph F Knight, and Alisa L Gallant. Influence of multi-source and multi-temporal remotely sensed and ancillary data on the accuracy of random forest classification of wetlands in northern minnesota. *Remote Sensing*, 5(7):3212–3238, 2013.

[193] Joseph F Knight, Bryan P TOLCSER, Jennifer M CORCORAN, and Lian P RAMPI. The effects of data selection and thematic detail on the accuracy of high spatial resolution wetland classifications. *Photogrammetric engineering and remote sensing*, 79(7):613–623, 2013.

[194] Bryan Walsh. How wetlands worsen climate change. http://www.time.com/time/health/article/0,8599,1953751,00.html, January 2010.

[195] A. Deschamps, D. Greenlee, TJ Pultz, and R. Saper. Geospatial data integration for applications in flood prediction and management in the red river basin. In *International Geoscience and Remote Sensing Symposium, Toronto, Canada. Symposium, Geomatics in the Era of RADARSAT (GER'97), Ottawa, Canada*, 2002.

[196] R.R. Hearne. Evolving water management institutions in the red river basin. *Environmental Management*, 40(6):842–852, Springer, 2007.

[197] CJ Van Westen. Remote sensing for natural disaster management. *International Archives of Photogrammetry and Remote Sensing*, 33(B7/4; PART 7):1609–1617, 2000.

[198] A. Akselrod-Ballin, M. Galun, R. Basri, A. Brandt, M.J. Gomori, M. Filippi, and P. Valsasina. An integrated segmentation and classification approach applied to multiple sclerosis analysis. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 1, pages 1122–1129. IEEE, 2006.

[199] M.E. Celebi, H.A. Kingravi, Y.A. Aslandogan, and W.V. Stoecker. Detection of blue-white veil areas in dermoscopy images using machine learning techniques. In *Proc. of SPIE Vol*, volume 6144, pages 61445T–1. Citeseer, 2006.

[200] D Bazell and David W Aha. Ensembles of classifiers for morphological galaxy classification. *The Astrophysical Journal*, 548(1):219, 2001.

[201] Tao Yuan and Way Kuo. Spatial defect pattern recognition on semiconductor wafers using model-based clustering and bayesian inference. *European Journal of Operational Research*, 190(1):228–240, 2008.

[202] J. Quinlan. Induction of decision trees. *Machine learning*, 1(1):81–106, Springer, 1986.

[203] Leo Breiman, Jerome Friedman, Charles J Stone, and Richard A Olshen. *Classification and regression trees*. Chapman & Hall/CRC, 1984.

[204] X. Li and C. Claramunt. A spatial Entropy-Based decision tree for classification of geographical information. *Transactions in GIS*, 10(3):451–467, Blackwell Publishing Ltd, 2006.

[205] D. Stojanova, M. Ceci, A. Appice, D. Malerba, and S. Džeroski. Global and local spatial autocorrelation in predictive clustering trees. In *Discovery Science*, pages 307–322. Springer, 2011.

[206] Daniela Stojanova, Michelangelo Ceci, Annalisa Appice, Donato Malerba, and Saso Dzeroski. Dealing with spatial autocorrelation when learning predictive clustering trees. *Ecological Informatics, Elsevier*, 2012.

[207] Zhe Jiang, Shashi Shekhar, Xun Zhou, Joseph Knight, and Jennifer Corcoran. Focal-test-based spatial decision tree learning: a summary of result. In *Data Mining (ICDM), 2013 IEEE 13th International Conference on*. IEEE, 2013.

[208] Zhe Jiang, Shashi Shekhar, Xun Zhou, Joseph Knight, and Jennifer Corcoran. Focal-test-based spatial decision tree learning. *Knowledge and Data Engineering, IEEE Transactions on*, 2015.

[209] Anne HS Solberg, Torfinn Taxt, and Anil K Jain. A markov random field model for classification of multisource satellite imagery. *Geoscience and Remote Sensing, IEEE Transactions on*, 34(1):100–113, 1996.

[210] Mete Celik, Baris M Kazar, Shashi Shekhar, Daniel Boley, and David J Lilja. Spatial dependency modeling using spatial auto-regression. In *Workshop on Geospatial Analysis and Modeling with Geoinformation Connecting Societies (GICON), International Cartography Association (ICA)*, 2006.

[211] Charles Boncelet. Image noise models. In Alan C. Bovik, editor, *Handbook of Image and Video Processing*, chapter 4.5. Academic Press, 2 edition, 2005.

[212] Luc Anselin. Local indicators of spatial association—lisa. *Geographical analysis*, 27(2):93–115, 1995.

[213] Russell G Congalton. A review of assessing the accuracy of classifications of remotely sensed data. *Remote sensing of Environment*, 37(1):35–46, 1991.

[214] John D Bossler, John R Jensen, Robert B McMaster, and Chris Rizos. *Manual of geospatial science and technology*. CRC Press, 2004.

[215] Russell G Congalton and Kass Green. *Assessing the accuracy of remotely sensed data: principles and practices*. CRC press, 2008.

[216] DRK Brownrigg. The weighted median filter. *Communications of the ACM*, 27(8):807–818, 1984.

[217] Humor Hwang and Richard A Haddad. Adaptive median filters: new algorithms and results. *Image Processing, IEEE Transactions on*, 4(4):499–502, 1995.

[218] Raymond H Chan, Chung-Wa Ho, and Mila Nikolova. Salt-and-pepper noise removal by median-type noise detectors and detail-preserving regularization. *Image Processing, IEEE Transactions on*, 14(10):1479–1485, 2005.

[219] S Esakkirajan, T Veerakumar, Adabala N Subramanyam, and CH PremChand. Removal of high density salt and pepper noise through modified decision based unsymmetric trimmed median filter. *Signal Processing Letters, IEEE*, 18(5):287–290, 2011.

[220] Jo Wijnant and Thérèse Steenberghen. Per-parcel classification of urban ikonos imagery. In *Proceedings of 7th AGILE Conference on Geographic Information Science*, pages 447–455, 2004.

[221] Yuliya Tarabalka, Jón Atli Benediktsson, and Jocelyn Chanussot. Spectral–spatial classification of hyperspectral imagery based on partitional clustering techniques. *Geoscience and Remote Sensing, IEEE Transactions on*, 47(8):2973–2987, 2009.

[222] Anne Puissant, Jacky Hirsch, and Christiane Weber. The utility of texture analysis to improve per-pixel classification for high to very high spatial resolution imagery. *International Journal of Remote Sensing*, 26(4):733–745, 2005.

[223] GJ Hay and G Castilla. Geographic object-based image analysis (geobia): A new name for a new discipline. In *Object-based image analysis*, pages 75–89. Springer, 2008.

[224] Tin Kam Ho and Mitra Basu. Complexity measures of supervised classification problems. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(3):289–300, 2002.

[225] Ester Bernadó-Mansilla and Tin Kam Ho. Domain of competence of xcs classifier system in complexity measurement space. *Evolutionary Computation, IEEE Transactions on*, 9(1):82–104, 2005.

[226] Tin Kam Ho, Mitra Basu, and Martin Hiu Chung Law. Measures of geometrical complexity in classification problems. In *Data complexity in pattern recognition*, pages 1–23. Springer, 2006.

[227] Dengsheng Lu and Qihao Weng. A survey of image classification methods and techniques for improving classification performance. *International journal of Remote sensing*, 28(5):823–870, 2007.

[228] Lian P Rampi, Joseph F Knight, and Keith C Pelletier. Wetland mapping in the upper midwest united states. *Photogrammetric Engineering & Remote Sensing*, 80(5):439–448, 2014.

[229] Stacy L Ozesmi and Marvin E Bauer. Satellite remote sensing of wetlands. *Wetlands ecology and management*, 10(5):381–402, 2002.

[230] Kali E Sawaya, Leif G Olmanson, Nathan J Heinert, Patrick L Brezonik, and Marvin E Bauer. Extending satellite remote sensing to local scales: land and water resource monitoring using high-resolution imagery. *Remote Sensing of Environment*, 88(1):144–156, 2003.

[231] Martin Herold, Dar A Roberts, Margaret E Gardner, and Philip E Dennison. Spectrometry for urban area remote sensing—development and analysis of a spectral library from 350 to 2400 nm. *Remote Sensing of Environment*, 91(3):304–319, 2004.

[232] A Stewart Fotheringham, Chris Brunsdon, and Martin Charlton. *Geographically weighted regression: the analysis of spatially varying relationships*. John Wiley & Sons, 2003.

[233] Barbara Pease and Allan Pease. *The Definitive Book of Body Language*. Bantam, July 2006.

[234] Thomas G Dietterich. Ensemble methods in machine learning. In *Multiple classifier systems*, pages 1–15. Springer, 2000.

[235] Zhi-Hua Zhou. *Ensemble methods: foundations and algorithms*. CRC Press, 2012.

[236] Gavin Brown, Jeremy Wyatt, Rachel Harris, and Xin Yao. Diversity creation methods: a survey and categorisation. *Information Fusion*, 6(1):5–20, 2005.

[237] Ye Ren, Le Zhang, and PN Suganthan. Ensemble classification and regression-recent developments, applications and future directions [review article]. *Computational Intelligence Magazine, IEEE*, 11(1):41–53, 2016.

[238] Leo Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996.

[239] Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, 55(1):119–139, 1997.

[240] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[241] Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87, 1991.

[242] Léon Bottou and Vladimir Vapnik. Local learning algorithms. *Neural computation*, 4(6):888–900, 1992.

[243] Michael I Jordan and Robert A Jacobs. Hierarchical mixtures of experts and the em algorithm. *Neural computation*, 6(2):181–214, 1994.

[244] Seniha Esen Yuksel, Joseph N Wilson, and Paul D Gader. Twenty years of mixture of experts. *Neural Networks and Learning Systems, IEEE Transactions on*, 23(8):1177–1193, 2012.

[245] Anuj Karpatne, Ankush Khandelwal, and Vipin Kumar. Ensemble learning methods for binary classification with multi-modality within the classes. In *Proceedings of the 2015 SIAM International Conference on Data Mining, Vancouver, BC, Canada, April 30 - May 2, 2015*, pages 730–738. SIAM, 2015.

[246] Anuj Karpatne and Vipin Kumar. Adaptive heterogeneous ensemble learning using the context of test instances. In *2015 IEEE International Conference on Data Mining, ICDM 2015, Atlantic City, NJ, USA, November 14-17, 2015*, pages 787–792, 2015.

[247] Lei Xu, Michael I Jordan, and Geoffrey E Hinton. An alternative model for mixtures of experts. *Advances in neural information processing systems*, pages 633–640, 1995.

[248] Viswanath Ramamurti and Joydeep Ghosh. Advances in using hierarchical mixture of experts for signal classification. In *Acoustics, Speech, and Signal Processing, 1996. ICASSP-96. Conference Proceedings., 1996 IEEE International Conference on*, volume 6, pages 3569–3572. IEEE, 1996.

[249] Martin Szummer and Rosalind W Picard. Indoor-outdoor image classification. In *Content-Based Access of Image and Video Database, 1998. Proceedings., 1998 IEEE International Workshop on*, pages 42–51. IEEE, 1998.

[250] Markus Enzweiler and Dariu M Gavrila. A multilevel mixture-of-experts framework for pedestrian classification. *Image Processing, IEEE Transactions on*, 20(10):2967–2979, 2011.

[251] Anna Bosch, Xavier Muñoz, and Robert Martí. Which is the best way to organize/classify images by content? *Image and vision computing*, 25(6):778–791, 2007.

[252] Ronald K Pearson. Surveying data for patchy structure. In *SDM*, pages 20–31. SIAM, 2005.

[253] David R. Martin and Charless C. Fowlkes. Matlab codes for multi-class hierarchical mixture of experts model. "`http://www.ics.uci.edu/~fowlkes/software/hme/`", 2002.

[254] Weka 3: Data mining software in java. "`http://www.cs.waikato.ac.nz/ml/weka/`", 2016.

[255] "updated national wetland invenctory, u.s. fish and wildlife service". "`https://www.fws.gov/wetlands/Data/Mapper.html`", 2016.

[256] Goo Jun and Joydeep Ghosh. Semisupervised learning of hyperspectral data with unknown land-cover classes. *Geoscience and Remote Sensing, IEEE Transactions on*, 51(1):273–282, 2013.

[257] André R Gonçalves, Fernando J Von Zuben, and Arindam Banerjee. Multi-label structure learning with ising model selection. In *Proceedings of the 24th International Conference on Artificial Intelligence*, pages 3525–3531. AAAI Press, 2015.

[258] Jian Dong, Wei Xia, Qiang Chen, Jianshi Feng, Zhongyang Huang, and Shuicheng Yan. Subcategory-aware object classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 827–834, 2013.

[259] Nitesh V Chawla, Thomas E Moore, Lawrence O Hall, Kevin W Bowyer, W Philip Kegelmeyer, and Clayton Springer. Distributed learning with bagging-like performance. *Pattern recognition letters*, 24(1):455–471, 2003.

[260] Robert E Banfield, Lawrence O Hall, Kevin W Bowyer, and W Philip Kegelmeyer. Ensembles of classifiers from spatially disjoint data. In *Multiple Classifier Systems*, pages 196–205. Springer, 2005.