Spatial Ensemble Learning for Heterogeneous Geographic Data with Class Ambiguity

ZHE JIANG and ARPAN MAN SAINJU, Department of Computer Science, University of Alabama
YAN LI and SHASHI SHEKHAR, Department of Computer Science, University of Minnesota
JOSEPH KNIGHT, Department of Forest Resources, University of Minnesota

Class ambiguity refers to the phenomenon whereby similar features correspond to different classes at different locations. Given heterogeneous geographic data with class ambiguity, the spatial ensemble learning (SEL) problem aims to find a decomposition of the geographic area into disjoint zones such that class ambiguity is minimized and a local classifier can be learned in each zone. The problem is important for applications such as land cover mapping from heterogeneous earth observation data with spectral confusion. However, the problem is challenging due to its high computational cost. Related work in ensemble learning either assumes an identical sample distribution (e.g., bagging, boosting, random forest) or decomposes multi-modular input data in the feature vector space (e.g., mixture of experts, multimodal ensemble) and thus cannot effectively minimize class ambiguity. In contrast, we propose a spatial ensemble framework that explicitly partitions input data in geographic space. Our approach first preprocesses data into homogeneous spatial patches and uses a greedy heuristic to allocate pairs of patches with high class ambiguity into different zones. We further extend our spatial ensemble learning framework with spatial dependency between nearby zones based on the spatial autocorrelation effect. Both theoretical analysis and experimental evaluations on two real world wetland mapping datasets show the feasibility of the proposed approach.

CCS Concepts: • Information systems → Geographic information systems; Data mining; • Computing methodologies → Ensemble methods;

Additional Key Words and Phrases: Spatial classification, class ambiguity, spatial heterogeneity, spatial ensemble, local models

ACM Reference format:
Zhe Jiang, Arpan Man Sainju, Yan Li, Shashi Shekhar, and Joseph Knight. 2019. Spatial Ensemble Learning for Heterogeneous Geographic Data with Class Ambiguity. *ACM Trans. Intell. Syst. Technol.* 10, 4, Article 43 (August 2019), 25 pages.
https://doi.org/10.1145/3337798

1 INTRODUCTION

Classifying heterogeneous geographic data with class ambiguity, i.e., same feature values corresponding to different classes in different locations, is a fundamental challenge in machine learning [14, 15]. Figure 1 shows an example in a wetland mapping application. The goal is to classify

Authors' addresses: Z. Jiang (corresponding author) and A. M. Sainju, Computer Science Department, University of Alabama, Box 870290, Tuscaloosa, AL 35487; emails: zjiang@cs.ua.edu, asainju@crimson.ua.edu; Y. Li and S. Shekhar, 4-192, Keller Hall, 200 Union St. S.E., Minneapolis, MN 55455; emails: {lixx4266, shekhar}@umn.edu; J. Knight, 1530 Cleveland Avenue, North St. Paul, MN 55108; email: jknight@umn.edu.

2157-6904/2019/08-ART43 $15.00
https://doi.org/10.1145/3337798

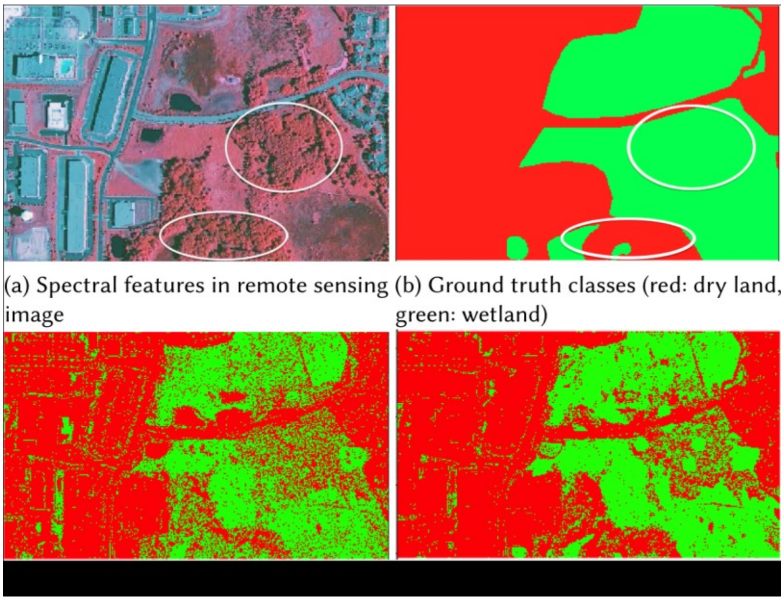

(a) Spectral features in remote sensing image
(b) Ground truth classes (red: dry land, green: wetland)

Fig. 1. Real-world example of heterogeneous geographic data: class ambiguity exists in two white circles.

remote sensing image pixels (Figure 1(a)) into wetland and dry land classes (Figure 1(b)). The two circled areas contain pixels that share very similar spectral values yet belong to two different classes (also called spectral confusion). As a result, decision tree and random forest classifiers learned from the entire image makes tremendous prediction errors as shown in Figure 1(c–d). The goal of spatial ensemble learning is to decompose the geographic area into zones to minimize class ambiguity and to learn a local model in each zone.

Motivations: Spatial ensemble learning can be used in many applications where geographic data is heterogeneous with class ambiguity. For example, in remote sensing image classification, spectral confusion is a challenging issue [18]. The issue is particularly important in countries where the type of auxiliary data that could reduce spectral confusion—such as elevation data or imagery of high temporal and spatial resolution—is not available. In economic study, it may happen that old house age indicates high price in rural areas but low price in urban areas [9]. Thus, *age* can be an effective coefficient to classify house price in individual zones but ineffective in a global model. In cultural study, 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 Great Britain [27]. In these cases, spatial ensemble learning can provide a tool that captures heterogeneous relationships between factors (e.g., house age, gestures) and target phenomena (e.g., house price, culture meanings).

Challenges: The SEL problem is computationally challenging. First, there are a large number of spatial samples (pixels) to partition. Second, the objective measure of class ambiguity is nondistributive, i.e., the degree of class ambiguity in a zone cannot be easily computed from the degrees of class ambiguity in its sub-zones. Finally, given a geographic data, the number of candidate partitions is exponential to the number of spatial samples. It can be proved that finding an optimal zone partition is NP-hard.

Related work: Spatial ensemble learning belongs to a general category of ensemble learning problems [5, 31, 38] in which a number of weak models are combined to boost prediction accuracy. Conventional ensemble methods, including bagging [2], boosting [10], and random forest [3], assume an identical distribution of samples. Thus, they cannot address heterogeneous geographic data with class ambiguity. Decomposition-based ensemble methods (also called divide-and-conquer), including mixture of experts [16, 37] and multimodal ensemble [24], 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 [35] or multiple local ellipsoids [29]. 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. Other methods such as adding spatial coordinates into feature vectors can be ineffective, since it creates geographic partitions whose zonal footprints are hard to interpret and can be too rigid to separate ambiguous zones with arbitrary shapes. There are other techniques for spatially heterogeneous data. A geographically weighted model [9] uses spatial kernel weighting functions to learn local models. However, it requires to learn a local model at every location, which is computationally very expensive, and it cannot allow arbitrary shapes of spatial zones for local models. Gaussian process [22] and multi-task learning [11] can also be used for heterogeneous geographic data, but they do not particularly focus on the class ambiguity issue. The mixture-of-experts approach has been used for *scene classification* on images via sub-blocks partitioning and learning local experts. But that problem is to classify an entire image (not individual pixels) [33].

Our contributions: To address limitations of related work, in our recent work [17], we formulate a spatial ensemble learning framework that explicitly partitions input data in geographic space. Our approach first preprocesses data into homogeneous patches and then uses a greedy heuristic to group patches into contiguous zones while minimizing class ambiguity. A local model is learned from each zone to make predictions on samples in the same zone. In our recent work, we make the following contributions: (1) we formulate a novel spatial ensemble learning problem to classify heterogeneous geographic data with class ambiguity; (2) we propose effective and efficient algorithms, including constraint-based hierarchical clustering for homogeneous patch generation, as well as a bisecting algorithm to group patches into contiguous zones via greedy heuristics; (3) we conduct experimental evaluations on the classification and computational performance of proposed approach on real-world wetland mapping datasets.

This article extends our recent work with the following additional contributions: (1) we provide theoretical analysis on the proposed algorithms, both on effectiveness and efficiency (time complexity); (2) we extend our previous spatial ensemble learning algorithm with spatial dependency constraint between adjacent-based classifiers to mitigate overfitting effect (when representative training samples do not exist within a zone but exist in another nearby zone); (3) we evaluate proposed extended method on a real-world dataset. We also add in full experiment results from our recent work.

Scope: This article focuses on the class ambiguity issue in heterogeneous geographic data. Other recent advances that do not address class ambiguity, such as spatial-spectral classifiers [7], objectbased image analysis [6, 25], metric learning, and active learning, fall outside the scope of this work.

Outline: The article is organized as follows: Section 2 defines basic concepts and formalizes the spatial ensemble learning problem. Section 3 introduces our approach. Experimental evaluations are in Section 5. Section 6 discusses some other relevant works. Section 7 concludes the article with future work.

Table 1. A List of Symbols and Descriptions

Symbol	Description
F	All samples in a raster framework
L	All labeled samples in F
U	All unlabeled samples in F
s_i	The *i*th spatial data sample
x_i	The vector of non-spatial features
l_i	The vector of two spatial coordinates
y_i	The class label of of sample s_i
$R(s_i,s_j)$	Spatial neighborhood relationship
P	A patch
Z	A zone
Lz	All labeled samples in Z
$N_k(s_i)$	Feature space neighborhood of s_i
$a(s_i)$	Per sample class ambiguity
$a(Z)$	Per zone class ambiguity

2 PROBLEM STATEMENT

This section formally defines the problem. Table 1 provides descriptions of the symbols used in our problem definition.

2.1 Basic Concepts

Geographic raster framework: A geographic raster framework F is a tessellation of a 2-D plane into a regular grid. Each grid cell (or pixel) is a *spatial data sample*, defined as$s_i = (x_i,l_i,y_i)$, $1 \leq i \leq |F|$, where x_i is a non-spatial feature vector, l_i is a two-dimensional vector of spatial coordinates, and $y_i \in \{c_1,c_2,...,c_p\}$ is a class label among p categories. All the samples in F can be divided into two disjoint subsets, a *labeled sample set* $L = \{s_i = (x_i,l_i,y_i) \in F | y_i \text{ is known}\}$ and *unlabeled sample set* $U = \{s_i = (x_i,l_i,y_i) \in F | y_i \text{ is unknown}\}$. In the example of Figure 2(a), F has 64 samples, including 14 labeled samples (colored in "training labels") and 50 unlabeled samples. Each sample has a onedimensional feature x and a class label (*red* or *green*).

Geospatial neighborhood relationship: It is a Boolean function on two samples $R(s_i,s_j)$, whose value is *true* if and only if s_i and s_j are spatially adjacent (i.e., two cells share a boundary).

Patch: A patch P is a spatially contiguous subset of samples, formally, P ⊆F such that for any two sampless_i,s_j∈P, either R(s_i,s_j) is *true* or we can find a set of sampless_{p1},s_{p2}, . . .,s_{pL}∈P such that R(s_i,s_{p1}), R(s_{pk} ,s_{pk+1}), and R(s_{pL},s_j) are all *true* for $1 \le k \le L - 1$. For example, all samples with input feature value 3 in Figure 2(a) form a patch. A patch is *homogeneous* if its samples have similar feature vectors (e.g., by Euclidean distance) and its labeled samples, if they exist, belong to only one class. For example, there are seven homogeneous patches highlighted in different gray scales in the first map of Figure 2(a).

Zone: A zone Z is a number of homogeneous patches that are spatially contiguous with each other. It is a set of spatially contiguous samples in a raster framework Z ⊆F with both labeled samples L_Z =L ∩ Z and unlabeled samples U_Z =U ∩ Z. In the example of Figure 2(c), zone 1 consists of three homogeneous patches, while zone 2 consists of four homogeneous patches.

Class ambiguity refers to the phenomenon whereby samples with the same non-spatial feature vector belong to different classes due to spatial heterogeneity (e.g., heterogeneous terrains). For example, in Figure 2(a), the four samples labeled with feature value$x$ = 1 belong to different classes (two *red* and two *green*). A global decision tree model makes erroneous predictions (Figure 2(b)).

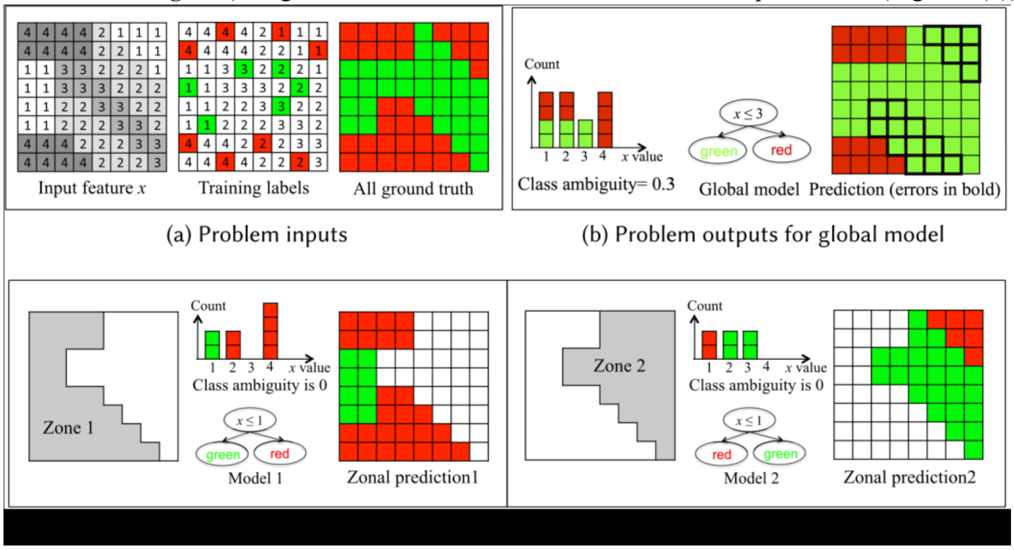


Fig. 2. Illustrative example of problem inputs and outputs (best viewed in color).

The degree of class ambiguity in a zone Z can be measured on its labeled samples L_Z. We define the following three concepts to quantify class ambiguity:

Feature space neighborhood: Feature space neighborhood of a sample s_i among all labeled samples L_Z in zone Z is defined as $N_k(s_i)$ = {s_j ∈ L_Z | s_j s_i,$d(x_i,x_j)$ is the k smallest}, where $d(x_i,x_j)$ is a metric function such as Euclidean distance. For example, for the red sample in the last column in the middle of Figure 2(a), its $N_2(s_i)$ can be any two labeled samples with x = 1 except the sample itself, including one *red* sample and two *green* samples. In this definition, we assume that labeled samples are locally dense in feature space to avoid the curse of dimensionality. In reality, this assumption is often satisfied due to the spatial autocorrelation effect (i.e., nearby training samples often resemble each other).

Per sample class ambiguity on a labeled sample s_i among all labeled samples L_Z in zone Z is defined as the ratio of labeled samples in different class from s_i in its neighborhood $N_k(s_i)$. Formal definition is in Equation 1, where $I(\cdot)$ is an indicator function. For example, the class ambiguity of the red sample in the last column of Figure 2(a) is █= 0.5 if one *red* sample and one *green* sample (with feature $x = 1$) are selected as $N_2(s_i)$. Its value can also be █= 1 if both *green* samples with feature value 1 happen to be selected as $N_2(s_i)$.

$$a(s_i) = \frac{1}{k} \quad {}_{j\,k} \quad {}_{i} I(y_j\ y_i) \qquad (1)$$
$$s \in N\ (s\ )$$

The *per zone class ambiguity* of a zone is defined as the average of *per sample class ambiguity* over all labeled samples. It is formally defined in Equation (2). For example, in Figure 2(a–b), the class ambiguity in the zone of the entire raster framework is (█ × 4 +█ × 4 + 0 × 2 + 0 × 4)/14 = 0.3. Similarly, the per zone class ambiguity of Z_1 or Z_2 in Figure 2(c) is0.

$$a(Z) = \frac{1}{|\quad|} \quad {}_{i} \quad a(s_i) \qquad (2)$$
$$L_Z\ s \in L_Z$$

A *spatial ensemble* is a decomposition of a raster framework F into m disjoint zones $\{Z_1, Z_2, \ldots, Z_m\}$ such that the average per zone class ambiguity is minimized. A local model can be learned in each zone Z_i based on its labeled (training) samples L_{Z_i} and then be used to classify unlabeled samples U_{Z_i} in the same zone. The concept of a local model in each zone can be generalized to a set of models (e.g., bagging, boosting, random forest) in the zone. In other words, spatial ensemble learning can be used together with traditional ensemble methods since they are orthogonal. Figure 2(c) shows an example of spatial ensemble with $m = 2$.

2.2 Problem Definition

The spatial ensemble learning problem is defined as follows: Input:

- A geographic raster framework F with labeled samples L and unlabeled samples U;
- The number of zones in the spatial ensemble: m;
- The parameter in feature space neighborhood: k.

Output: A spatial ensemble with m contiguous zones such that:

$$\arg\min_{Z_1, Z_2, \ldots, Z_m} \frac{}{m} \quad a(Z_i), \quad {}^{m}_{i=1}$$

$$1 \qquad \text{subject to} \qquad (1) Z_i \cap Z_j = \emptyset \text{ for} i \;\; j, \; m$$

$$(2)\; Z_i = F,$$

$$i=1$$

where $a(Z_i)$ is the per zone class ambiguity, and $f(Z_i)$ is the number of isolated patches.

Figure 2 shows a problem example. Inputs include a geographic data with 64 samples, 14 labeled (training) and 50 unlabeled, with one feature x and two classes (*red*, *green*) (Figure 2(a)). The class ambiguity of the entire framework is $a(F) = 0.3$, computed from the class histogram of training samples. A global decision tree makes prediction errors (Figure 2(b)). In contrast, a spatial ensemble with two zones in Figure 2(c) reduces per zone class ambiguity to zero. Predictions of local models show zero errors.

The spatial ensemble learning problem is formulated as a geographical partition problem, because we assume that the underlying causes of class ambiguity is spatial heterogeneity. This phenomenon is also known as "ecological fallacy," or spatial Simpson's Paradox. Individual zones in spatial ensemble are contiguous to avoid overfitting (spatial regularization) and also to conform to the first law of geography, "Everything is related to everything else, but nearby things are more relevant than distant things" [34]. There are several other assumptions in our problem formulation. First, we assume samples in the raster framework form homogeneous patches. This is often true due to the spatial autocorrelation effect, particularly when the pixel resolution is high. Second, we assume feature vectors of unlabeled (test) samples are given within the same raster framework of training samples. In other words, the problem belongs to transductive learning. This can limit the scope of the problem. Finally, we assume a pixel belongs to only one class, i.e., there is no class ambiguity within a pixel. The computational challenges of the problem are discussed in Theorem 2.1.

Theorem 2.1. *The spatial ensemble learning problem is NP-hard.*

Proof. Here, we only provide main ideas. First, our objective function of per zone class ambiguity is non-monotonic and non-distributive. Thus, we cannot compare one candidate zone partitioning against another without computing class ambiguity. Second, the number of possible zone partitioning is beyond polynomial. This can be derived from the NP-hardness of grid graph partitioning problems [8].

3 PROPOSED APPROACH

In this section, we present our algorithms to address computational challenges of the spatial ensemble learning problem. Our algorithms consist of two phases. First, input spatial data samples (both labeled and unlabeled) are clustered into homogeneous patches. We propose to use a constraint-based hierarchical spatial clustering approach (Section 3.1). After this, homogeneous patches are further grouped into contiguous zones through a recursive bisecting process (Section 3.2).

3.1 Preprocessing: Homogeneous Patches

Given geographic data with all labeled and unlabeled samples, generating homogeneous patches can be considered as image segmentation [12] but with the constraint that labeled samples in the same patch, if they exist, belong to the same class.

ALGORITHM 1: Homogeneous Patch Generation

```
Input:
    • All samples in the raster framework: $F$
    • Spatial neighborhood relationship: $R(\cdot,\cdot)$
    • The number of output patches: $n$, $n$ $|F|$ Output:
    • A set of $n$ patches: $P = \{P_1, P_2, \ldots, P_n\}$
 1: Initialize a patch set $P = \{P_i = \{s_i\} | s_i \in F\}$
 2: while number of patches $|P| > n$ do
 3:     for each adjacent pair $P_i$ and $P_j$ do
 4:         if $d(P_i,P_j)$ has been computed then
 5:             Continue to next for iteration
 6:         if $L_{pi}, L_{pj}$ either empty or same class then
 7:             $d(P_i,P_j) \leftarrow \frac{1}{|P_i||P_j|} \sum_{s_i \in P_i, s_j \in P_j} d(x_i,x_j)$
 8:         else
 9:             $d(P_i,P_j) \leftarrow +\infty$
10: Find $P_i, P_j$ with minimum dissimilarity $d(P_i,P_j)$ 11: Merge
these two patches: $P_i \leftarrow P_i \cup P_j$, $P \leftarrow P \setminus P_j$ 12: return $P$.
```

Algorithm 1 shows our bottom-up hierarchical method to generate homogeneous patches. First, each data sample is initialized as a patch (step 1). The algorithm then repeatedly merges pairs of *adjacent patches* (patches with samples that are spatial neighbors) in a greedy manner. Only patch pairs whose labeled samples belong to the same class can be merged (step 6). The patch pair whose samples have the smallest feature dissimilarity (step 7) are merged first (steps 10–11). Merging continues until the number of patches is reduced to a given number n. In implementation, we can use a patch adjacency graph to efficiently find pairs of adjacent patches. The graph can be easily updated when two patches (nodes) are merged. Figure 3 shows a toy example. The input geographic data contains 64 samples with one feature and two classes (*red* and *green*). Adjacent samples with the same feature value are merged into a patch. For instance, all samples with feature value 4 in the upper left corner are merged into patch*A*. The final output is 7 homogeneous patches (shown by different shades: *A* to *G*).

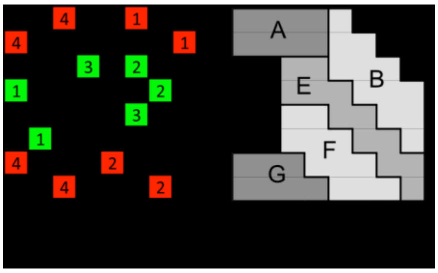


Fig. 3. Illustration of homogeneous patch generation.

Algorithm 1 has two major computational bottlenecks in its iterations: identification of the adjacent patch pair with the minimum dissimilarity on the entire map (step 10) and computation of dissimilarity values between new adjacent patch pairs (step 7). To address the first bottleneck, we propose to use a priority queue with adjacent patch pairs ordered by dissimilarity. To reduce the cost of patch dissimilarity computation, we reuse previously computed dissimilarity values when possible.

Details of these computational refinements are in Algorithm 2. The algorithm maintains a neighborhood graph where nodes are patches and edges are spatial adjacency between patches. Edge weights e_{ij} are dissimilarity values between adjacent patch pairs (v_i,v_j). Initially, the graph is a grid graph with each sample (pixel) as a node (patch) (steps 1–2). Then, the algorithm repeatedly merges two neighboring nodes with the minimum edge weight until the total number of nodes (patches) are reduced to a required number n. To quickly find neighboring nodes with the minimum edge weight, we maintain a priority queue of all neighboring node pairs ordered by their edge weights (step 3) and extract the minimum element from the queue in each iteration (step 6). After extracted, the pair of nodes v_i,v_j are merged into a new node v_n (step 10), and the corresponding edges are also updated. When computing the weights of edges connected to the new node v_n, we reuse the weights of edges connected to nodesv_i,v_j (step 12) to avoid redundant computation (see definition of $d(\mathsf{P}_i,\mathsf{P}_j)$ in steps 7 and 9 in Algorithm 1). The weights of new patch pairs are added to the priority queue (step 13). Once nodes v_i,v_j are merged, their corresponding elements in the priority queue become obsolete. Thus, we maintain a hash set of all obsolete nodes (steps 1 and 9) to ignore their elements in the priority queue (steps 7–8).

3.2 Group Homogeneous Patches into Zones

After samples are clustered into homogeneous patches, the second phase of our spatial ensemble learning method aims to divide these patches into several contiguous groups (zones) to minimize class ambiguity within each group (zone). This can be considered as a planar graph partition problem where nodes are patches and edges are spatial adjacency. To group patches (nodes) into multiple zones, we propose a bisecting algorithm (Algorithm 3). The algorithm starts with one zone containing the set of all patches (steps 1–2), and then keeps breaking down the current most ambiguous zone into two until the number of zones reaches a required number (steps 3–7). The critical question now becomes how to divide a zone (set of patches) into two to minimize class ambiguity. This is done via another subroutine called TwoZoneSpatialEnsemble (Algorithm 4) whose details are introduced below.

Since graph partitioning problems are generally computationally hard [8], in Algorithm 4, we propose a greedy heuristic that assign patches (graph nodes) into two zones maximizing *inter-zone*

class ambiguity while minimizing *intra-zone* class ambiguity. To do that, the algorithm uses a seedgrowing process to expand two zones on a patch adjacency graph. At the beginning, all patches
ALGORITHM 2: Faster Homogeneous Patch Generation

Input:

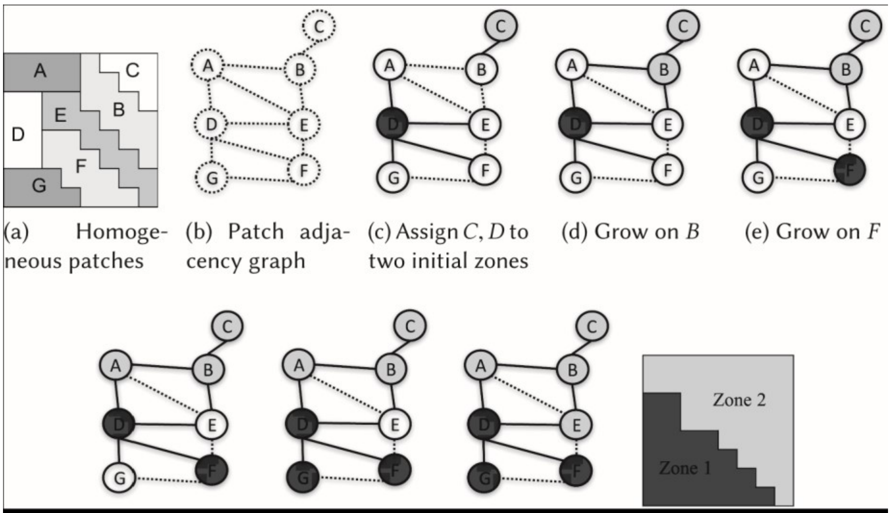
- All samples in the raster framework: F
- Spatial neighborhood relationship: R(·, ·)
- The number of output patches: n, n |F| Output:
- A set of n patches: P= {P$_1$,P$_2$, . . .,P$_n$}

```
1: Initialize a patch set P= {P_i = {s_i}|s_i ∈ F}
2: Initialize a neighborhood graph G(V,E) with each patch as a node: v_i =P_i
   = {s_i} for 1 ≤ i ≤ |F|

          e_ij = ⎧⎪⎨⎪⎩ d∞(x_i,x_j)        if s_i,s_j are neighbors, same class or unlabeled

                              otherwise

3: Create a priority queue PQ with all neighbor pairs (v_i,v_j,e_ij)
4: Initialize a set of obsolete nodes O ← ∅
5: while |V| > n and PQ not empty do
6:     (v_i,v_j,e_ij) ← ExtractMin(PQ)
7:     if v_i ∈ O or v_j ∈ O then
8:         Continue to next while iteration
9:     O ← O ∪{v_i} ∪{v_j}
10:    Create a new node v_n merging v_i,v_j in G (P_n ← P_i ∪ P_j)
11:    for each other neighbor node v_k of v_i or v_j do
12:
              ||||⎧|||⎨ e_k,ik|P_ki|·|k P_ki |+ie_ki ,j |jP_kk,|·|j P_kj |        j      if v_k neighbors both v_i,v_j

e_k,n = ||||⎩⎩ de_k(,P_i |,PP|kP|·|))|P||·|P_kP_k|·|i(||·|P+(PdP|+i(i||P|++P_ke|,P|P)jjj|))|||PP_k |·||·|PP_j ||   ifif vv_kk
neighborsneighbors vv_ij onlyonly

                    |P |·|(P |+|P |)
```

```
13:         Add edge ($v_k$,$v_n$,$e_{k,n}$) to graph $G$
14: Add ($v_k$,$v_n$,$e_{k,n}$) into priority queue $PQ$ 15: Remove
obsolete nodes $v_i$,$v_j$ and their edges 16: return P= $V$ .
```

are marked as *unassigned* (step 2), and the class ambiguity of all patch pairs (whether adjacent or not) are computed (step 3). The algorithm finds the two patches with the highest class ambiguity as initial seeds and assigns one patch to each zone, respectively (steps 4–5). The algorithm also maintains a set of frontier nodes (unassigned spatially adjacent nodes) F_1,F_2 for each zone (steps 6–7). Next, the algorithm iteratively grows a zone by adding a node from its frontier until all nodes are *assigned* (i.e., two frontiers are empty).

When selecting a node from the frontier of a zone, we use a greedy heuristic that maximizes *inter-zone* class ambiguity while minimizing *intra-zone* class ambiguity. This is shown in the formula of $A^1{}_k$ and $A^2{}_k$ (steps 10 and 14). In the formula of $A^1{}_k$, the numerator is the maximum class ambiguity between the candidate patch P_k and patches in the other zone Z_2, reflecting inter-zone class ambiguity, while the denominator is the maximum class ambiguity between P_k and patches in its corresponding zone Z_1, reflecting intra-zone class ambiguity. We add a value 1 in the formula for normalization. To avoid the case in which most patches are assigned to one single zone, we also add a size-balance factor $B_k{}^1$ ($B_k{}^2$) to our heuristic. Size balance factor across two zones can be



(a) Homogeneous patches (b) Patch adjacency graph (c) Assign C, D to two initial zones (d) Grow on B (e) Grow on F

(f) Grow on A (g) Grow on G (h) Grow on E (i) Final 2 zones

Fig. 4. A running example of Algorithm 3.

ALGORITHM 3: Bisecting Multi-zone Spatial Ensemble

Input:

- A set of homogeneous patches: $P= \{P_1,P_2, . . .,P_n\}$
- The number of zones: m (m n)
- The parameter in class ambiguity measure: k

- The balancing parameter in our greedy heuristic: α Output:
- A spatial ensemble of m zones: $Z = \{Z_1, \ldots, Z_m\}$ 1: Initialize a zone with all input

```
patches: $Z_1 \leftarrow P$
2: Initialize a set of zones for outputs: $Z \leftarrow \{Z_1\}$
3: while $|Z| < m$ do
4:     Find the zone with max class ambiguity:
       $Z_0 = \arg\max_{Z_i \in Z} a(Z_i)$
5:     Remove zone $Z_0$ from result set: $Z \leftarrow Z \setminus Z_0$ 6:
       [redacted] $=$ TwoZoneSpatialEnsemble($Z_0$, $k$,
$\alpha$)
7: $Z \leftarrow Z$ [redacted] $Z$
8: return Z
```

measured via the entropy $-r_1 \log r_1 - r_2 \log r_2$, where r_1 and r_2 are the ratio of the sizes (number of samples) of zone 1 and zone 2 to their total size. A higher entropy value indicates more sizebalanced zones. We use a parameterα to weight the influence of two factors in our heuristic (step 12 and 16). The node with the maximum overall score P_{k0} is selected and is added to its corresponding zone Z_{f0} (step 18). The node is then removed from frontiers. Its original frontier is expanded with the node's unassigned neighbors. Finally, all nodes are *assigned*, the frontiers become empty, and the two zones are returned (step 21).

Running example: Figure 4 shows a running example of Algorithm 4 with the same input data as the example in Figure 3. Assume $k = 2$, $\alpha = 0.5$, and $m = 2$. The adjacency graph of patches is

Table 2. Patch Pairs with Non-zero Class Ambiguity

Patch P_i	Patch P_j	$a(P_i \cup P_j)$
B	*F*	0.5
C	*D*	0.5
D	*C*	0.5
F	*B*	0.5

ALGORITHM 4: Two Zone Spatial Ensemble

Input:

- A set of homogeneous patches: $P = \{P_1, P_2, \ldots, P_n\}$
- The parameter in class ambiguity measure: k
- The weight parameter in our greedy heuristic: α Output:
- A spatial ensemble of two zones: $\{Z_1, Z_2\}$

```
1: Create a spatial adjacency graph with patches as nodes
```

```
2: Initialize all nodes as *unassigned*
3: Compute class ambiguity $a_{ij} = a(P_i \cup P_j)$ for any $i$ $j$
4: Find $P_i, P_j$ with max class ambiguity $a_{ij}$
5: Initialize $Z_1 \leftarrow \{P_i\}$, $Z_2 \leftarrow \{P_j\}$, mark $P_i, P_j$ as *visited*
6: Initialize $F_1$ with all *unassigned* neighboring patches of $Z_1$
7: Initialize $F_2$ with all *unassigned* neighboring patches of $Z_2$
8: while F[redacted]do
9:         for each [redacted]
10:          $A$[redacted]$_{k1+\sup}$ $a(P_k,P_o)$ //class ambiguity avoidance
                  $P_o \in Z_1$
11:  $B_k^1 \leftarrow$ SizeBalance(Z,Z$_2$)   [redacted]   //zone size balance
12:    Compute overall score:          $S_k \leftarrow \alpha A$[redacted]
13:        for each[redacted]
                                       k   o
14:         $A$[redacted]                        //class ambiguity avoidance
                  $P_o \in Z_2$
15:            $B_k^2 \leftarrow$ SizeBalance($Z_1$,$Z_2 \cup \{P_k\}$) //zone size balance
16:          Compute overall score: $S_k^2 \leftarrow \alpha A$[redacted]
17:        Find the $P_{k0} \in F_{f0}$ ($f_0 \in \{1, 2\}$) with max overall score
18: $Z_{f0} \leftarrow Z_{f0} \cup \{P_{k0}\}$, mark $P_{k0}$ as *visited*
19: $F_1 \leftarrow F_1 \setminus \{P_{k0}\}$, $F_2 \leftarrow F_2 \setminus \{P_{k0}\}$
20: Expand $F_{f0}$ with all *unassigned* neighboring patches of $P_{k0}$
21: return {$Z_1$, $Z_2$}
```

shown in Figure 4(b). Patch pairwise class ambiguity is shown in Table 2. The two zones are shown by two different colors. Frontiers are shown by solid edges connected to zones. Initially, $Z_1 = \{C\}$ and $Z_2 = \{D\}$ (Figure 4(b)). The frontier of Z_1 is $\{B\}$, while the frontier of Z_2 is $\{A,E,F,G\}$. In the next iteration, all candidate nodes from the frontiers have zero class ambiguity avoidance score, but node B has the highest size balance score, so it is selected to grow Z_1. Nodes F,A,G,E are then selected consecutively. The final output two zones are shown in Figure 4(i). This output is slightly different from our problem example in Figure 2, but both reduce class ambiguity to zero.

Algorithm 4 will face the situation whereby some patches do not have (or have only few) training samples. The algorithm still can work in this case, since it always starts with two patches with the largest class ambiguity (each contains sufficient training samples in an opposite class). In zone expansion steps, patches without training samples can appear in the frontier to be merged. For such patches, we will still use the same selection measures (i.e., class ambiguity avoidance, size balance).

It is just that adding a patch without training samples will not change class ambiguity of a zone; it can only impact the size balance.

3.3 Extension with Spatial Dependency Across Zones

In our current spatial ensemble framework, a base classifier is learned based on training samples in a zone itself and is only used to classify the samples within the same zone. In other words, base classifiers in different zones work independently. One potential limitation of this framework is that when the number of zones is large, the amount of training samples falling into each zone tends to reduce, posing a potential risk for overfitting. For example, there can be sub-areas in a zone whose representative training samples are partitioned into another zone due to our recursive zone partition process based on greedy strategy. In this case, these sub-areas may be misclassified due to lack of representative training samples in the zone.

To address the challenge, we propose to use the multi-task learning framework to joint-learn base classifiers in different zones together. Each task is the process of learning a base classifier in a specific zone based on training samples in it. One main question is how to determine the relatedness between different tasks (zones). According to the first law of geography, "everything is related to everything else, but near things are more related than distance things." Thus, zones that are adjacent to each other tend to be similar, and thus their classifiers should be related. However, in our spatial ensemble learning, we should also consider class ambiguity between zones, which indicates "negative" relatedness. If two zones have high class ambiguity, then we should avoid learning similar models between the two zones, because such models tend to misclassify ambiguous samples. Therefore, we define relatedness between different tasks based on both zone spatial adjacency and zone class ambiguity. Specifically, we define relatedness between zone Z_i and zone Z_j, i.e.,$W_{i,j}$, in Equation (3), where $a(Z_i \cup Z_j)$ is the class ambiguity, and $\sigma > 0$ is a parameter to control the impact of class ambiguity on task relatedness (a small σ value means more negative impact of class ambiguity on task relatedness).

$$W_{i,j} = \begin{cases} e^{a(Z_i \cup Z_j)} & \text{if } Z_i \text{ and } Z_j \text{ are spatially adjacent} \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

Based on the definition of task (zone) relatedness, we use multi-task learning with spatial dependency constraint across adjacent zones. The constraint is added to the objective (loss) function of individual models. Similar ideas have been studied before on feature vector space ensemble [23], but the effect on spatial ensemble has not been explored. Here, we use logistic regression as an example. The idea can be generalized to all base classification models with differentiable loss functions. The overall objective function can be specified as Equation (4):

$$L(\beta_1, \beta_2, \ldots, \beta_m) = -\sum_{k=1}^{m} \sum_{i \in L_k} \log P(y_i \mid x_i) + \lambda \sum W_{i,j} (\beta_i - \beta_j)^T (\beta_i - \beta_j)$$

$$= \quad {}_{m\,k}\log \prod\prod \quad {}_{-\beta\,x} \quad {}_{1-y} \quad 1 \quad {}_{y+\lambda} \quad W_{i,j} \quad {}^{T}(\beta_i$$
$$\beta_j).\ ^{e}$$
$$(\beta_i \quad \beta_j)\ {}_{k=1\ i\in L}\ 1 + e{-\beta\,x}\ \ 1 + e{-\beta\,x}\ {}_{i,j}$$

(4)

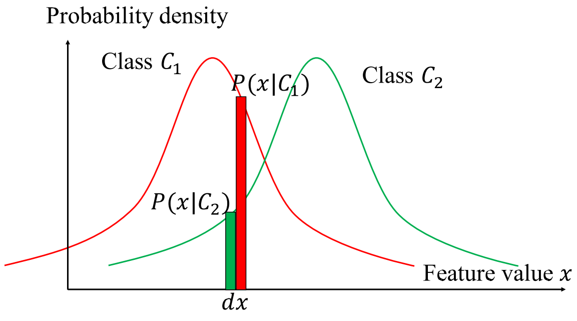


Fig. 5. Proof of class ambiguity measure as an upper bound of Bayesian error.

The objective is convex but non-linear. Thus, the Newton Ralphson [36] method can be used to estimate optimal parameter values. After parameters β_k for all zone k are learned, each base classifier can be used to classify samples within its own zone. Two base classifiers with high relatedness (both being spatially adjacent and with low class ambiguity) will share similar model parameters (their training samples are mutually utilized). This helps avoid overfitting when representative training samples of one zone happen to exist in nearby zones only.

4 THEORETICAL ANALYSIS

Theorem 4.1. *The expectation of per zone class ambiguity measure is an upper bound of Bayesian error.*

Proof. Without the loss of generalizability, we provide proof for the case of binary classification (two classes). Assume that the prior probability of class 1 and class 2 in a zone Z are $P_Z(C_1)$ and $P_Z(C_2)$, respectively. Also assume that the conditional probability of a feature vector x in two classes are $P_Z(x|C_1)$ and $P_Z(x|C_2)$, respectively, as shown in Figure 5. The Bayesian error in Z can be expressed below for binary classes. The expression is based on the definition that Bayesian error rate is the lowest possible error rate for any classifier. The optimal classifier is the one that classifies a sample into a class with a higher probability.

$$BayesianError(Z) = \int_x min(P_Z(C_1)P_Z(x|C_1), P_Z(C_2)P_Z(x|C_2))dx$$

Assume that samples are locally dense in feature space (this assumption is often true when training samples are in the form of spatially contiguous patches and the spatial autocorrelation effect is high). The proposed per sample class ambiguity measure ($a(s)$) is an estimation of the percentage of opposite-class samples against the checked sample in a small neighborhood $dx = \{x\ |d$ [redacted]

$\epsilon\}$ in the feature vector space. This small neighborhood is approximated by our feature space neighborhood $N_k(s)$ in per sample class ambiguity measure. Thus, the probability ratio at feature value x can be approximated by the corresponding class ratio in $N_k(s)$:

$$E[a(s)] = E \Bigg| \Bigg| \Bigg| \_1_{k} \quad I(y_j \; y) \Bigg| \Bigg| \Bigg| \Bigg| \Bigg| \Bigg| = \frac{P^z \,|\, \sout{(C_2)P_Z(x\,|\,C_2)} \quad |}{} ,$$

$$\Big| \Big\lfloor k_{\;j(s)}\, P_Z(C_1)P_Z(x\; C_1) + P_Z(C_2)P_Z(x\; C_2)\; {}_{s\in N}$$

where x is the feature value of s.

$$E[a(Z)] \quad = Pz\,(C_1)Pz\,(x\,|\,C_1) \cdot Pz\,|(C_2)Pz\,(x\,|\,C_2)\;|\; dx,\; {}_x P_Z(C_1)P_Z(x$$

$$C_1) + P_Z(C_2)P_Z(x\; C_2)$$

$$+Pz\,(C_2)Pz\,(x\,|\,C_2) \sout{\cdot\; Pz\,|(C_1)Pz\,(x\,|\,C_1)\;|\; dx,\; {}_x P_Z(C_1)P_Z}(x$$

$$\sout{C_1) + P_Z(C_2)P_Z(x\; C_2)}$$

$$\sout{2P\,{}_x P_Z(C_1)P_Z(x\,|\,C_1) + P_Z}$$

$$(C_2)P_Z(x\,|\,C_2)$$

$$={}_Z(C_1)P(x\,|\,C_1)P_Z(C_2)P_Z(x\,|\,C_2)\; dx,$$

$$1$$

$$=^{x}\, Pz\,(C_1)P1z\,(x\;|C_1) + Pz\,(C_2)P1z\,(x\;|C_2)\; dx.$$

From the fact that the minimum of two values is smaller than their harmonic mean, we can get $BayesianError(Z) \leq E[a(Z)]$. In multi-class scenarios, the neighbor samples of an$s$ can be grouped into two parts, i.e., those in the same class as s and those in other classes, which simplify the problem into a binary case. Therefore, proposed per zone class ambiguity measure is an upper bound of Bayesian error.

Theorem 4.1 is important, because Bayesian error rate is generally considered as the lowest possible error rate for any classifier in statistical classification. The fact that class ambiguity is an upper bound of Bayesian error theoretically justifies that minimizing class ambiguity through spatial ensemble learning (zone partitioning) can help reduce Bayesian error rate and thus improve classification performance in each zone.

Theorem 4.2. *The time complexity of baseline homogeneous patch generation (Algorithm 1) is* $O((N - n)(N + ep^2))$, *where N is the total number of samples, n is the number of output patches, e* *and p are the maximum number of neighbors and the maximum number of samples on each patch, respectively.*

Proof. The loop runs $O(N - n)$ iterations. In each iteration, the algorithm needs to compute feature distance for any neighboring patch pair if its value does not exist already. This only happens when a patch is recently created through merging other patches. Thus, there are at most $O(e)$ distance computations in each iteration, each costing $O(p^2)$. Together with a linear scan to select minimum distance pair, each iteration costs $O(N + ep^2)$. So, total cost is $O((N - n)(N + ep^2)$.

Theorem 4.3. *The time complexity of faster homogeneous patch generation (Algorithm 2) is* $O((N - n)(\log N + ep^2))$*, where N is the total number of samples, n is the number of output patches, e and p are the maximum number of neighbors and the maximum number of samples on each patch, respectively.*

Proof. The main difference between Algorithm 2 and Algorithm 1 is that the former uses a priority queue to extract the patch pair with the minimum distance, costing $O(\log N)$ instead of $O(N)$ for each operation. Step 12 of Algorithm 2 also prunes out some redundant computation but does not improve the worst-case time complexity. So, the total cost is $O((N - n)(\log N + ep^2)$.

Theorem 4.4. *The time complexity of two zone spatial ensemble (Algorithm 4) and bisecting spatial ensemble (Algorithm 3) is* $O(n^2 l^2 \log k)$ *and* $O(mn^2 l^2 \log k)$*, respectively, where m is the number of output zones, n is the number of input patches, l is the maximum number of labeled samples in each patch, and k is the class ambiguity parameter.*

Proof. We first analyze the time complexity of Algorithm 4. Step 3 computes class ambiguity of $O(n^2)$ patch pairs, each costing $O(l^2 \log k)$ if we use a size k priority queue to maintain the current k-nearest-neighbors in feature space. This is the most time-consuming part. After this, in each iteration, the algorithm selects one node from two frontiers with $O(n)$ candidates. Evaluating each candidate costs $O(n)$. So, the total cost is $O(n^2 l^2 \log k + n^2) = O(n^2 l^2 \log k)$. For Algorithm 3, in each "while" loop, it computes the zone with maximum class ambiguity, costing$O((nl)^2 \log k) = O(n^2 l^2 \log k)$, and then calls Algorithm 4, costing $O(n^2 l^2 \log k)$. Thus, the total time complexity of Algorithm 3 is $O(mn^2 l^2 \log k)$.

5 EXPERIMENTAL EVALUATION

The goal of the experiments was to:

- Evaluate the classification accuracy of spatial ensemble learning.
- Test the sensitivity of spatial ensemble to its parameters.
- Evaluate the proposed extension with spatial dependency across zones.
- Evaluate the computational costs of spatial ensemble algorithms.

5.1 Experiment Setup

In the classification accuracy experiment, we compared the following methods:

- Global model learning (i.e., learning models and making predictions on the entire study area), including single model, and bagging, boosting, and random forest [1].
- Decomposition-based ensemble via feature clustering, e.g., K means or hierarchical clustering (HC). Specifically, all samples are clustered into groups, and a base classifier (or bagging, boosting, and random forest of base classifiers) is learned in each group and makes predictions.
- Hierarchical mixture of expert method [16, 37].
- Multimodal ensemble based on K means clustering from Reference [24].
- Spatial ensemble method with base classifiers as either a single model or model ensemble (bagging, boosting, and random forest).

We also tested the sensitivity of spatial ensemble learning to its parameters, including the number of zones m, the base classifier type, class ambiguity measure parameter k, and balancing parameter α in greedy heuristic. The number of patches$n$ in preprocessing was determined via trying different values and visualizing the output homogeneous patches. For computational performance comparison, we compared our baseline and refined homogeneous patch generation algorithms (Algorithms 1 and 2). We also evaluated computational performance of the bisecting spatial ensemble algorithm (Algorithm 3). Codes for spatial ensemble and global models were implemented in Java and base classifiers were from Weka toolbox [1]. Codes for decomposition-based ensemble via feature clustering, as well as multimodal ensemble, were implemented in R. Codes for the hierarchical mixture of experts were in Matlab [26] (the code has only logistic regression as base classifiers). We conducted computational experiments on an iMac desktop with 4GHz Intel Core i7 processor and 32GB DDR3 main memory.

Dataset description: Our datasets were collected from two areas in Minnesota: *Chanhassen* and *Big Stone* [30]. We used eight explanatory features, including four spectral bands (red, green, blue, near-infrared) in high resolution (3m by 3m) aerial photos from the National Agricultural Imagery Program during leaf-off season, and four corresponding textures on homogeneity [28]. Class labels (wetland and dry land) were collected from the updated National Wetland Inventory. The Chanhassen scene contains 221 by 374 pixels, and the BigStone scene contains 718 by 830 pixels. We used systematic clustered sampling to select the training set and used remaining pixels as the test set (details in Table 3).

Table 3. Dataset Description

	Training Samples		Test Samples	
Scene	Dry	Wet	Dry	Wet
Chanhassen	6,715	4,323	40,362	31,254
Big Stone	45,483	27,138	345,557	177,762

Evaluation metric: We evaluated the classification performance with confusion matrices and F-score (harmonic mean of precision and recall) on the wetland class (wetland class is of more interest).

5.2 Classification Performance Evaluation

5.2.1 Comparison on Classification Accuracy. In Chanhassen data, we set parameter values as $n = 100$, $m = 6$, $k = 10$, $\alpha = 0.9$. In BigStone data, we set parameter values as $n = 800$, $m = 20$, $k = 10$, $\alpha = 0.9$ (n and m were set higher in BigStone than in Chanhassen, because BigStone is larger). The classification accuracy results for the two datasets are summarized in Table 4 and Table 5, respectively. In the confusion matrix displayed in each table, the first and second rows show *true* dry land and wetland samples, respectively, and the first and second columns show *predicted* dry land and wetland samples, respectively. We can see that in global models, bagging, boosting, and random forest slightly improve a single decision tree (overall F-score increases by 0.03), but significant errors remain. Decomposition-based ensemble methods via feature space clustering—e.g., K means or hierarchical clustering—improve global models from up to 0.79 to up to 0.84 in Chanhassen data, and from up to 0.78 to 0.84 in BigStone data. Note that we did not run hierarchical clustering on BigStone data due to the high computational cost on a large number of samples. Mixture of experts do not perform well on either dataset, even worse than global models, probably due to its logistic regression (generalized linear model) base classifiers. We also tried multimodal ensemble [24] with K-means clustering. The best results were achieved (F-scores of around 0.71 for Chanhassen dataset and around 0.81 for BigStone dataset) when we set up the number of clusters in each class as 2. But its performance quickly gets worse when we increase the number of clusters in each class. In summary, improvements of these methods over global models are largely due to their ensemble of local models in feature space. However, since training samples with class ambiguity can easily fall into the same cluster in feature space (due to highly similar feature values) though their classes are different, these methods cannot separate samples with class ambiguity. In contrast, the spatial ensemble of models achieve the best F-scores (over 0.9 on Chanhassen data, and up to 0.86 on BigStone data). The improvements can be seen in the reduction on the number of false negatives in the confusion matrices (lower left corner) (around 80% reduction on Chanhassen data, and around 50% reduction on BigStone data, compared with global models).

5.2.2 Effect of the Number of Zones m. To test the effect of the number of zones m in spatial ensemble learning, we fixed the other parameters the same as Section 5.2, but varying the number of zones m from 2 to 10 in Chanhassen data, and from 2 to 40 in BigStone data. We measured the overall F-score of the four spatial ensemble learning models over m. Results are summarized in Figure 6(d) and Figure 7(d). As can be seen, as the number of zones m increases, the classification accuracy of all spatial ensemble learning models improves and then reaches a plateau. Similar trends are shown on both datasets. It is worth noting that we have sufficient training samples falling into each zone in this case. For the case in which we have insufficient training samples falling into a zone, we can leverage the method proposed in Section 3.3, and its results are shown in Section 5.3. In practice, the parameter m can be determined based on the size and homogeneity

Table 4. Results on Chanhassen ("SE" for Spatial Ensemble)

Ensemble Method	Confusion Matrix		F score
Global Single Model	36,734	3,628	0.76
	9,640	21,614	
Global Bagging	36,497	3,865	0.79
	8,272	22,982	
Global Boosting	35,506	4,856	0.79
	7,646	23,608	

Global Random Forest	36,867	3,495	0.79
	8,349	22,905	
Kmeans with Single Model	35,664	4,698	0.79
	9,655	21,599	
Kmeans with Bagging	36,430	3,932	0.83
	7,868	23,386	
Kmeans with Boosting	36,286	4,076	0.83
	7,475	23,779	
Kmeans with Random Forest	36,824	3,538	0.84
	7,675	23,579	
HC with Single Model	14,627	2,236	0.79
	3,728	9,409	
HC with Bagging	15,124	1,702	0.83
	3,124	10,050	
HC with Boosting	15,059	1,767	0.83
	3,065	10,109	
HC with Random Forest	15,208	1,618	0.84
	3,160	10,014	
Mixture of Experts	33,963	6,399	0.71
	10,326	20,928	
SE with Single Model	37,407	2,955	0.92
	2,073	29,181	
SE with Bagging	37,565	2,797	0.93
	1,871	29,383	
SE with Boosting	37,527	2,835	0.93
	1,851	29,403	
SE with Random Forest	37,609	2,753	0.93
	1,688	29,566	

of the study area. The bigger and more heterogeneous a study area is, a larger *m* value is needed. We also tested the sensitivity of other feature space ensemble methods to the number of clusters. Results were shown in Figure 6(a–c) and Figure 7(a–b). We did not run hierarchical clustering on BigStone data due to the high computational cost on a large number of samples. In decompositionbased ensemble via feature clustering (e.g., K-means and hierarchical clustering), F-scores only increase with *m* for single decision tree base classifiers and generally do not improve with *m* for bagging, boosting, and random forest base classifiers. From the results, feature space ensemble do not need a large number of clusters if base classifiers in each cluster are bagging, boosting, or random forest instead of single decision trees. The reason can be that bagging, boosting, and

Table 5. Results on BigStone ("SE" for Spatial Ensemble)

Ensemble Method	Confusion Matrix		F score
Global Single Model	305,172	40,385	0.75

	46,760	131,002	
Global Bagging	313,625	31,932	0.77
	45,586	132,176	
Global Boosting	307,866	37,691	0.76
	44,813	132,949	
Global Random Forest	317,777	27,780	0.78
	46,263	131,499	
Kmeans with Single Model	315,721	39,943	0.82
	38,705	128,950	
Kmeans with Bagging	312,173	33,296	0.83
	44,126	133,724	
Kmeans with Boosting	313,313	32,156	0.84
	43,769	134,081	
Kmeans with Random Forest	316,663	28,806	0.84
	44,299	133,551	
Mixture of Experts	329,400	16,157	0.60
	93,589	84,173	
SE with Single Model	316,201	29,356	0.85
	25,300	152,462	
SE with Bagging	316,908	28,649	0.86
	23,162	154,600	
SE with Boosting	315,817	29,740	0.85
	23,397	154,365	
SE with Random Forest	318,009	27,548	0.86
	22,926	154,836	

random forest are more complex models compared with single decision trees and thus do not need a large number of clusters in feature space.

5.2.3 Effect of Base Classifier Type. The parameters were the same as Section 5.2. We chose several different base classifier types including decision tree (DT), SVM, neural network (NN), and logistic regression (LR). Results on two datasets are shown in Figure 8. We can see that spatial ensemble learning consistently outperforms global model learning on different base classifier types. The gap is narrower when neural network is used as a base classifier, probably due to its greater model complexity.

5.2.4 Effect of the Parameter k in Class Ambiguity Measure. We fixed the same parameters as Section 5.2 except that we varied the parameter k from 5 to 25. Results of our four spatial ensemble models on different parameter k are summarized in Figure 9. From the results, we can see that the spatial ensemble learning algorithm is not sensitive to the value of k. This is partly because we test our algorithm to a relatively modest range of k. Since k is the number of nearest neighboring labeled samples in the feature space, a very big value of k will make the nearest neighbor samples not a good approximation of local distribution and thus degrade performance. In addition, class ambiguity measure based on a very large k may not be computable when the number of training

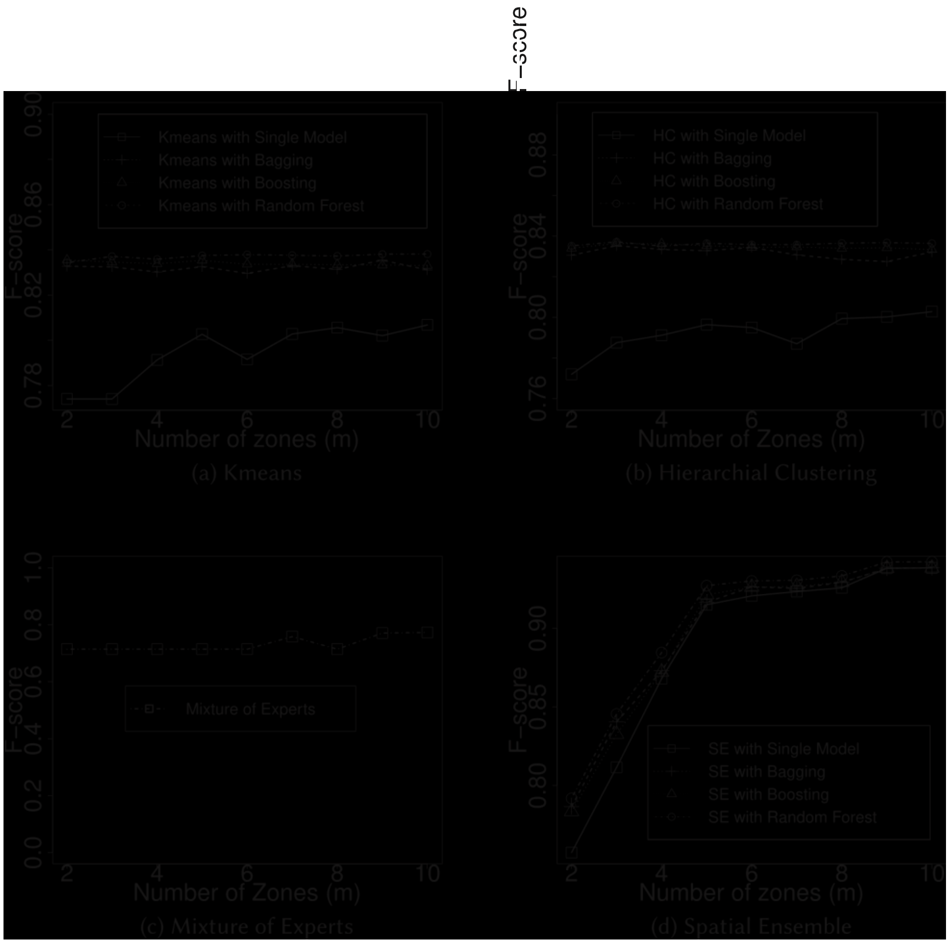


Fig. 6. Effect of the number of zones *m* on Chanhassen data.

samples in a patch pair is much smaller than *k*. In practice, we can select a modest value of *k*, e.g., *k* = 5 or *k* = 10.

5.2.5 Effect of the Parameter α in Greedy Heuristic. We fixed the same parameters as Section 5.2 except that we varied the balancing parameter in our greedy heuristic *α* from 0 to 1. A higher *α* means a higher weight on class ambiguity avoidance than on zone size balance. Results are summarized in Figure 10. We can see that the spatial ensemble learning results are generally stable. In Chanhassen data, classification performance slightly improves as *α* increases. In BigStone data, classification performance stays stable except for the extreme cases *alpha* = 0 and *α* = 1. In practice, we can determine the value of *α* based on cross-validation.

5.3 The Effect of Extension with Spatial Dependency Across Zones

Here, we evaluate the effect of extending spatial ensemble learning framework with additional spatial dependency between base classifiers. We used the Bigstone dataset, since it is larger and more prone to overfitting for when training samples are sparsely distributed on the map (some zones in spatial ensemble may lack representative training samples that exist in other nearby zones). We selected training samples that are in the form of 104 circular clusters, with 43 clusters (6,347 samples) in the wetland class and 61 clusters (9,149 samples) in the dry land class. We used logistic regression model as base model. We compared three candidate methods, including

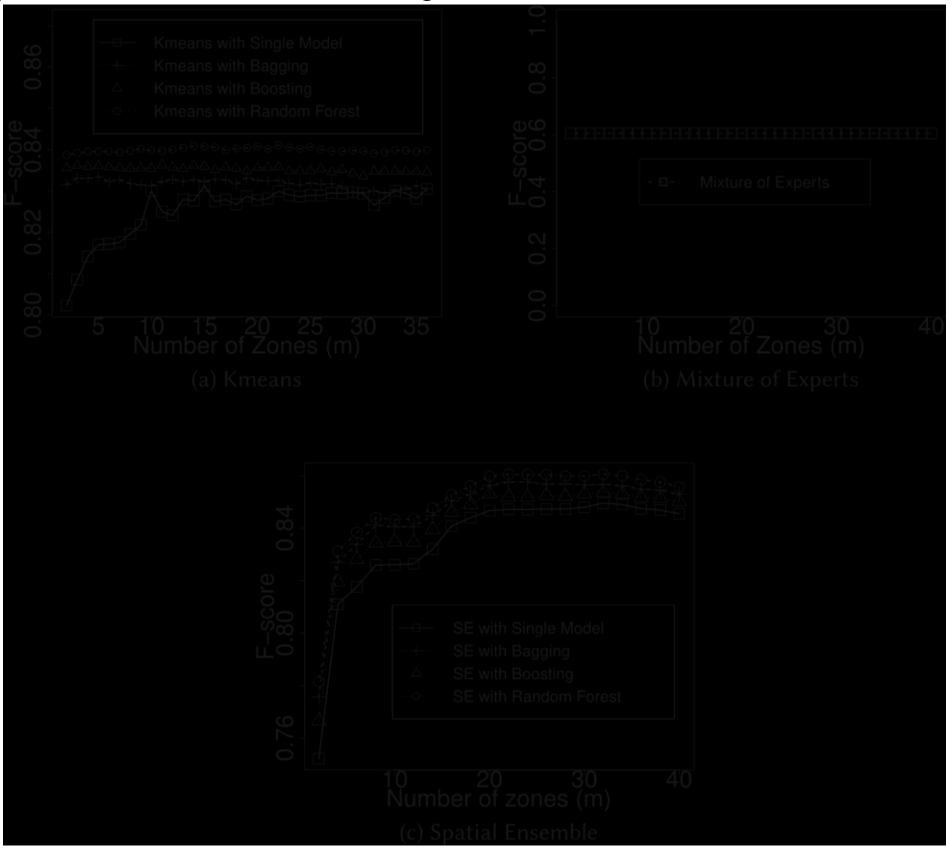


Fig. 7. Effect of the number of zones *m* on BigStone data.

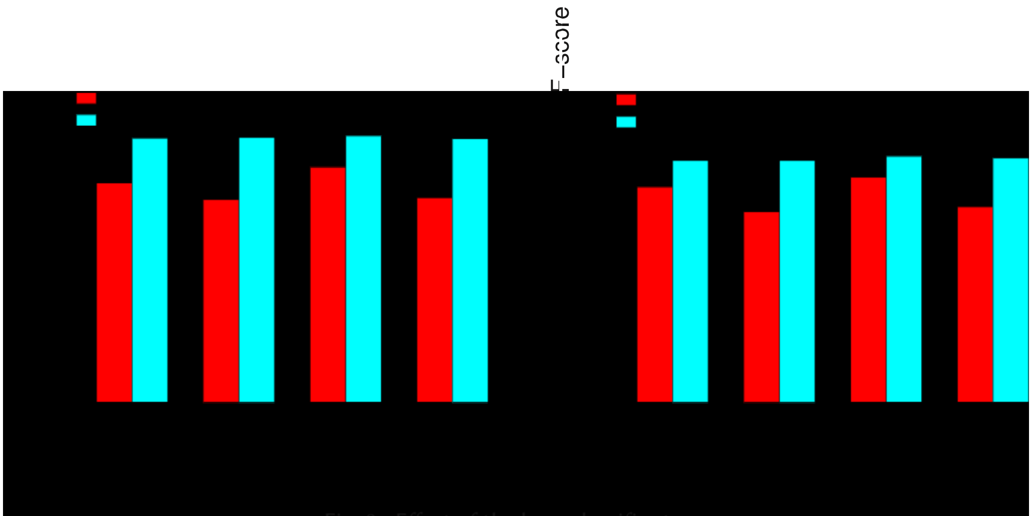
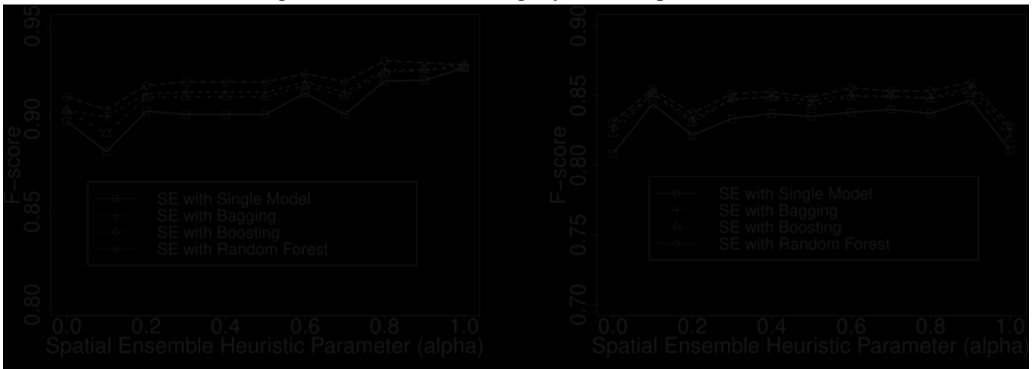


Fig. 8. Effect of the base classifier type.

global logistic regression model, spatial ensemble of logistic regression models without spatial dependency, as well as spatial ensemble of logistic regression models with spatial dependency between base classifiers. The parameter λ was set as 100 in extended spatial ensemble method.

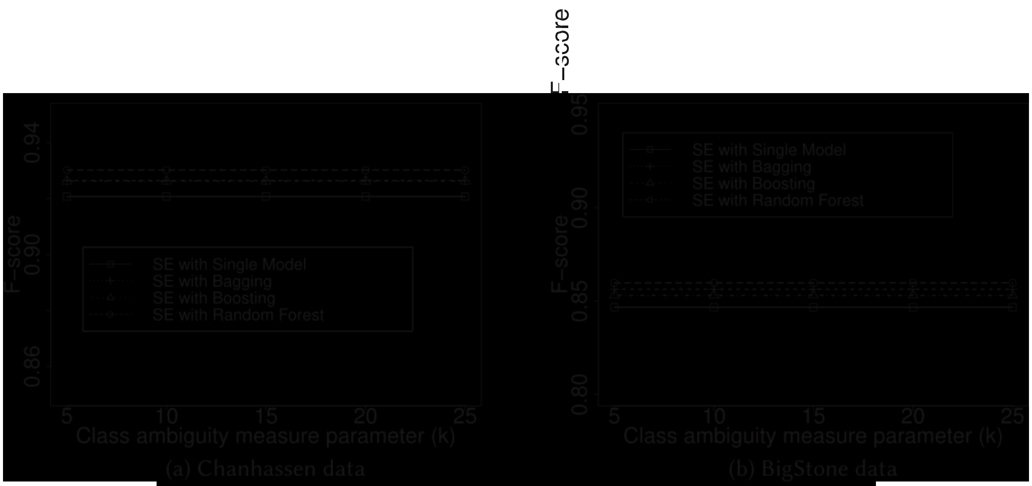
Results are summarized in Figure 11. As can be seen, the global model has an F-score below 0.65 (around 0.645). Spatial ensemble proposed in our conference paper without spatial dependency has

Fig. 9. Effect of class ambiguity measure parameter *k*.



(a) Results on Chanhassen (b) Results on BigStone

Fig. 10. Effect of balancing parameter α in spatial ensemble.



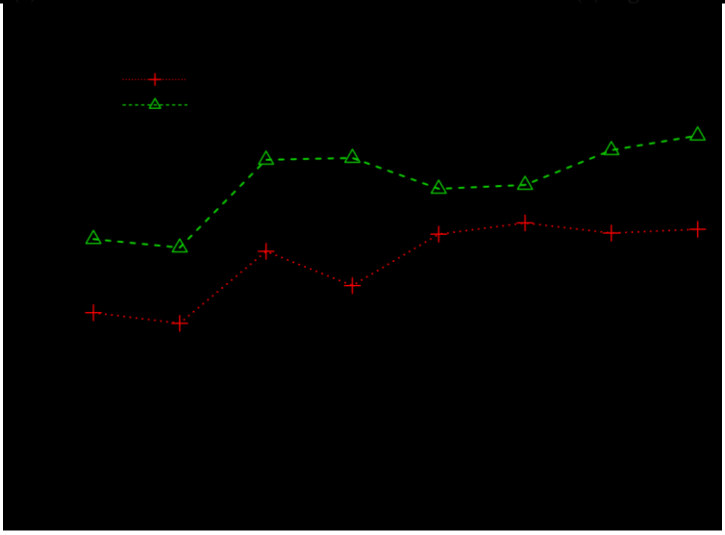


Fig. 11. Evaluate the effect of spatial dependency between base classifiers in spatial ensemble.

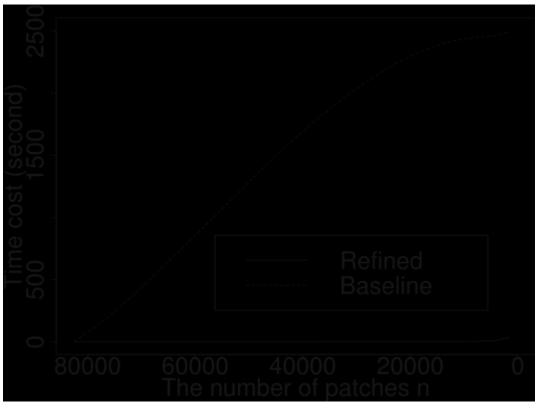


Fig. 12. Time costs of baseline and refined algorithms in homogeneous patch generation.

F-score slightly increasing with the number of zones. Its F-scores are all above that of the global model. We did not show more zones because the algorithm stopped partitioning a zone further when no class ambiguity can be seen based on training samples within the zone. In contrast, our spatial ensemble with spatial dependency between base classifiers showed the best classification performance. The results show that adding spatial dependency can help improve spatial ensemble learning framework when training samples are sparse in the region. It is worth noting that adding spatial dependency may not necessarily improve classification performance in all cases. Spatial dependency constraint is helpful when representative training samples do not exist within a zone but exist in another nearby zone.

5.4 Computational Performance Evaluation

We now discuss the computational time costs of our spatial ensemble learning algorithms, including the homogeneous patch generation phase and the bisecting spatial ensemble phase.

To evaluate the time costs of homogeneous patch generation phase, we compared our baseline algorithm (Algorithm 1) and refined algorithm (Algorithm 2) on different parameter values of n (the number of patches). We used the Chanhassen data with 82,654 total input samples. We varied the values of parameter n from 82,000 to 100. Results are shown in Figure 12. We can see that as n decreases, the time costs of both algorithms increase (due to more merging operations), but the cost of baseline is far higher than the refined algorithm. The growth rate of time cost in the baseline algorithm gradually gets lower with decreasing n. The reason is that as the patch adjacency graph gets smaller, the cost of finding the best patch pair with the minimum dissimilarity is also lower. In contrast, the growth rate in the refined algorithm gets higher. The reason is that as patches get larger, the cost computing dissimilarity is more expensive (finding the patch pair with the minimum dissimilarity from a priority queue is very fast).

We measured the time costs of homogeneous patch generation (the refined algorithm) and bisecting spatial ensemble on the two datasets we used. The parameter settings were the same as Section 5.2. Results are summarized in Table 6. The time costs are averages of five runs. The reported time does not include local model learning time. Our algorithms can process over half a million samples within several minutes.

Regarding computational costs of various ensemble methods, we did not provide quantitative comparison, since methods are implemented in different programming languages. Based on our experience, feature space clustering based on K-means is the fastest. Spatial ensemble and feature space clustering based on hierarchical clustering are much slower due to time-expensive patch generation and feature clustering.

Table 6. Computational Time Costs of Spatial Ensemble

	Chanhassen	BigStone
Number of samples	82,654	595,940
Patch Generation	45s	238s
Bisecting Spatial Ensemble	6s	190s

6 DISCUSSION

There are other relevant works to our problem. Geographic Object Based Image Analysis (GEOBIA) [13] is a popular technique for earth imagery classification. GEOBIA first segments earth imagery into objects and then treats objects as minimum classification units. Segmentation can be done by software tools (e.g., eCognition) based on feature similarity (e.g., color, texture) often semi-automatically with human in the loop. Results are promising (e.g., reducing salt-andpepper noise) particularly on high-resolution earth imagery. The main difference from our work is on the goal of space partition: GEOBIA partitions image based on feature similarity to recognize objects, while our spatial ensemble approach partitions space into zones to minimize class ambiguity. To consider class ambiguity in existing GEOBIA, extra manual efforts are often needed such as adding object features like "distance to roads." In fact, image segmentation in GEOBIA can be used in the preprocessing step of our approach (Algorithms 1–2) to generate homogeneous patches. After this, our spatial ensemble algorithms (Algorithms 3–4) can be applied to assign patches (or image segments) into different zones to minimize class ambiguity. There are other spatial classification methods that address spatial autocorrelation, including spatial decision trees [19, 20, 21]. These methods are orthogonal and complementary to spatial ensemble learning. In recent years, deep-learning methods (e.g., Deeplab [4], U-net [32]) have achieved great success in image segmentation applications. In our spatial ensemble problem, the input training set only consists of isolated labeled pixels (or pixel groups), not complete image segments. Thus, it is hard to apply a deep-learning approach. However, if sufficient training data with complete labeled image segments are available, deep-learning methods can be very promising solutions.

Our problem requires two important parameters: the number of zones m and the number of feature space neighbors k. According to our results in sensitivity analysis, our algorithm is not quite sensitive to the number k. In practice, we can select a modest number such as 10. The number of zones m can be determined based on generally two factors in a specific application: the overall study area size and the homogeneity of landscape in the area. A bigger study area and a more geographically heterogeneous area require a bigger m. For example, in urban areas where landscape often changes, the number of zones should be bigger; in rural areas where landscape is often largely homogeneous, the number of zones can be small.

7 CONCLUSION

This article investigates the spatial ensemble learning problem for heterogeneous geographic data with class ambiguity. We proposed spatial ensemble learning algorithms that consist of two phases: generating homogeneous patch from input spatial data samples and grouping homogeneous patches into different zones to reduce class ambiguity via a greedy heuristic. We also extended our spatial ensemble learning framework with spatial dependency constraint between nearby base classifiers. We analyzed the theoretical properties of proposed algorithms both on effectiveness and on computational efficiency. Evaluations on real-world datasets show that our spatial ensemble approach outperforms global models in classification accuracy, and incorporating spatial dependency in spatial ensemble can improve classification performance. Computational experiments also show that proposed computational refinements are effective in reducing time cost.

In future work, we plan to evaluate proposed algorithms on other applications such as spatial modeling in economic data. We also plan to investigate inductive spatial ensemble learning, in which test samples can be from a different spatial framework from the training samples.

REFERENCES

[1] 2016. Weka 3: Data mining software in Java. Retrieved on September 1, 2018 from http://www.cs.waikato.ac.nz/ml/weka/.

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

[3] Leo Breiman. 2001. Random forests. *Machine Learn.* 45, 1 (2001), 5–32.

[4] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. 2018. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Trans. Patt. Anal. Machine Intell.* 40, 4 (2018), 834–848.

[5] Thomas G. Dietterich. 2000. Ensemble methods in machine learning. In *Multiple Classifier Systems*. Springer, 1–15.

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

[7] Mathieu Fauvel, Yuliya Tarabalka, Jón Atli Benediktsson, Jocelyn Chanussot, and James C. Tilton. 2013. Advances in spectral-spatial classification of hyperspectral images. *Proc. IEEE* 101, 3 (2013), 652–675.

[8] Andreas Emil Feldmann. 2013. Fast balanced partitioning is hard even on grids and trees. *Theoret. Comput. Sci.* 485 (2013), 61–68.

[9] A. Stewart Fotheringham, Chris Brunsdon, and Martin Charlton. 2003. *Geographically Weighted Regression*. John Wiley & Sons, Limited.

[10] Yoav Freund and Robert E. Schapire. 1997. A decision-theoretic generalization of on-line learning and an application to boosting. *J. Comput. Syst. Sci.* 55, 1 (1997), 119–139.

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

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

[13] G. J. Hay and G. Castilla. 2008. Geographic object-based image analysis (GEOBIA): A new name for a new discipline. In *Object-based Image Analysis*. Springer, 75–89.

[14] Tin Kamo Ho and Mitra Basu. 2002. Complexity measures of supervised classification problems. *IEEE Trans. Patt. Anal. Machine Intell.* 24, 3 (2002), 289–300.

[15] Tin Kam Ho, Mitra Basu, and Martin Hiu Chung Law. 2006. Measures of geometrical complexity in classification problems. In *Data Complexity in Pattern Recognition*. Springer, 1–23.

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

[17] Zhe Jiang, Yan Li, Shashi Shekhar, Lian Rampi, and Joseph Knight. 2017. Spatial ensemble learning for heterogeneous geographic data with class ambiguity: A summary of results. In *Proceedings of the 25th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. ACM, 23.

[18] Zhe Jiang and Shashi Shekhar. 2017. *Spatial Big Data Science: Classification Techniques for Earth Observation Imagery*. Springer.

[19] Zhe Jiang, Shashi Shekhar, Pradeep Mohan, Joseph Knight, and Jennifer Corcoran. 2012. Learning spatial decision tree for geographical classification: A summary of results. In *Proceedings of the 20th International Conference on Advances in Geographic Information Systems*. ACM, 390–393.

[20] Zhe Jiang, Shashi Shekhar, Xun Zhou, Joseph Knight, and Jennifer Corcoran. 2013. Focal-test-based spatial decision tree learning: A summary of results. In *Proceedings of the 13th IEEE International Conference on Data Mining (ICDM'13)*. IEEE, 320–329.

[21] Zhe Jiang, Shashi Shekhar, Xun Zhou, Joseph Knight, and Jennifer Corcoran. 2015. Focal-test-based spatial decision tree learning. *IEEE Trans. Knowl. Data Eng.* 27, 6 (2015), 1547–1559.

[22] Goo Jun and Joydeep Ghosh. 2013. Semisupervised learning of hyperspectral data with unknown land-cover classes. *IEEE Trans. Geosci. Remote Sens.* 51, 1 (2013), 273–282.

[23] Anuj Karpatne, Ankush Khandelwal, Shyam Boriah, and Vipin Kumar. 2014. Predictive learning in the presence of heterogeneity and limited training data. In *Proceedings of the SIAM International Conference on Data Mining*. SIAM, 253–261.

[24] Anuj Karpatne, Ankush Khandelwal, and Vipin Kumar. 2015. Ensemble learning methods for binary classification with multi-modality within the classes. In *Proceedings of the SIAM International Conference on Data Mining*. SIAM, 730–738.

[25] Dengsheng Lu and Qihao Weng. 2007. A survey of image classification methods and techniques for improving classification performance. *Int. J. Remote Sens.* 28, 5 (2007), 823–870.

[26] David R. Martin et al. 2002. Matlab codes for multi-class hierarchical mixture of experts model. Retrieved from: http://www.ics.uci.edu/~fowlkes/software/hme/.

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

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

[29] Viswanath Ramamurti and Joydeep Ghosh. 1996. Advances in using hierarchical mixture of experts for signal classification. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, Vol. 6. IEEE, 3569–3572.

[30] Lian P. Rampi, Joseph F. Knight, and Keith C. Pelletier. 2014. Wetland mapping in the upper midwest United States. *Photogram. Eng. & Remote Sens.* 80, 5 (2014), 439–448.

[31] Ye Ren, Le Zhang, and P. N. Suganthan. 2016. Ensemble classification and regression-recent developments, applications and future directions. *Comput. Intell. Mag. IEEE* 11, 1 (2016), 41–53.

[32] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015. U-net: Convolutional networks for biomedical image segmentation. In *Proceedings of the International Conference on Medical Image Computing and Computer-assisted Intervention*. Springer, 234–241.

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

[34] Waldo R. Tobler. 1970. A computer movie simulating urban growth in the Detroit region. *Econ. Geo.* 46 (1970), 234–240.

[35] Lei Xu, Michael I. Jordan, and Geoffrey E. Hinton. 1995. An alternative model for mixtures of experts. In *Proceedings of the Conference on Neural Information Processing Systems* (1995), 633–640.

[36] Tjalling J. Ypma. 1995. Historical development of the Newton–Raphson method. *SIAM Rev.* 37, 4 (1995), 531–551.

[37] Seniha Esen Yuksel, Joseph N. Wilson, and Paul D. Gader. 2012. Twenty years of mixture of experts. *IEEE Trans. Neural Netw. Learn. Syst.* 23, 8 (2012), 1177–1193.

[38] Zhi-Hua Zhou. 2012. *Ensemble Methods: Foundations and Algorithms*. CRC Press.

Received March 2018; revised September 2018; accepted May 2019