Title

Batch Active Learning for Multispectral and Hyperspectral Image Segmentation Using Similarity Graphs

Authors

Chen, Bohan
Miller, Kevin
Bertozzi, Andrea L
et al.

Publication Date

2023

DOI

10.1007/s42967-023-00284-8

Peer reviewed

Batch active learning for multispectral and hyperspectral image segmentation using similarity graphs

Bohan Chen[1*], Kevin Miller[2], Andrea L. Bertozzi[1] and Jon Schwenk[3]

[1*]Department of Mathematics, University of California, Los Angeles, 520 Portola Plaza, Los Angeles, 90095, CA, USA.
[2]Oden Institute for Computational Engineering and Sciences, University of Texas at Austin, 201 E 24th St, Austin, 78712, TX, USA.
[3]Los Alamos National Laboratory, 87545, NM, USA.

*Corresponding author(s). E-mail(s): bhchenyz@g.ucla.edu;
Contributing authors: ksmiller@utexas.edu;
bertozzi@math.ucla.edu; jschwenk@lanl.gov;

Abstract

Graph learning, when used as a semi-supervised learning (SSL) method, performs well for classification tasks with a low label rate. We provide a graph-based batch active learning pipeline for pixel/patch neighborhood multi- or hyperspectral image segmentation. Our batch active learning approach selects a collection of unlabeled pixels that satisfy a local maximum constraint for the active learning acquisition function that determines the relative importance of each pixel to the classification. This work builds on recent advances in the design of novel active learning acquisition functions (e.g. the Model Change approach in arXiv:2110.07739) while adding important further developments including patch-neighborhood image analysis and batch active learning methods to further increase the accuracy and greatly increase the computational efficiency of these methods. In addition to improvements in accuracy, our approach can greatly reduce the number of labeled pixels needed to achieve the same level of the accuracy based on randomly selected labeled pixels.

Keywords: Image Segmentation, Graph Learning, Batch Active Learning, Hyperspectral Image

1 Introduction

Image segmentation is a basic problem in the field of machine learning and computer vision. One older approach involves partial differential equation (PDE)-based methods, which segment an image by solving a PDE on the image numerically, based on minimization of an energy functional [1–4]. More recently, graph-based methods have also been developed for both semi-supervised and unsupervised learning on image processing [5–15]. Another common choice is the neural network methods, including convolutional neural networks (CNN) [16] and graph convolutional networks (GCN) [17, 18], with trainable convolutional filters optimized by minimizing the difference between predicted and ground-truth labels.

We employ a graph-based learning method where the feature vectors of each pixel in an image are used to construct a graph whose edge weights are determined by feature vector similarity [19]. This approach has proven successful in noisy image recovery [7], studies using remotely-sensed images to combine LIDAR and optical images [20], and blind hyperspectral unmixing [21]. Graph learning is an approach that trains a classifier by minimizing a graph-based energy function to directly identify a function on the nodes of the graph. This is different from graph neural networks that train a convolutional kernel and evolve the corresponding convolutional operator on the graph.

Active learning is a branch of machine learning that judiciously selects a limited number of unlabeled data to query for labels, with the aim of maximally improving the underlying classifier's performance [22]. An acquisition function is used to quantify which data would be useful to label from the set of available unlabeled data. Active learning can significantly improve classifier performance at very low label rates and minimize the cost of labeling data by domain experts [22–25].

Traditional active learning selects labeled data sequentially; i.e., in each step, only the global maximum of the acquisition function is selected. Batch active learning selects a query set of multiple points in each step of the active learning process. Batch active learning provides new challenges compared to sequential active learning. Selecting data with similar information is redundant and does not fully utilize the acquisition function. Some prior methods for batch active learning imitate sequential active learning by selecting the batch through a greedy sequential process [26–28]. Some other batch active learning approaches segment the candidate set into several small subsets and select the batch samples as the collection of the maximum points of each small subset [29, 30]. Here, we develop a novel batch active learning approach, called LocalMax, to select a collection of unlabeled data that satisfy the graph local maximum condition in each step of the active learning process. Compared with other

batch active learning approaches, ours is more efficient while having almost identical performance as sequential active learning.

1.1 Our Contributions

In this paper, we develop a graph-based active learning pipeline for image segmentation tasks with very low label rates. Our novelties and major contributions are:

1. Inspired by [25], we apply active learning to select pixels to label for the graph Laplace learning classifier to process an image segmentation task. We extract a non-local means feature vector of each pixel in the target image and then build a similarity graph based on these feature vectors. Such a feature extraction allows us to include contextual (i.e. neighborhood) information for each pixel. Active learning reduces the number of labeled pixels required for this semi-supervised image segmentation method. With fewer than 0.5% labeled pixels selected by the active learning process, the classifier reaches a similar accuracy as that with 10% randomly sampled labeled pixels.
2. We introduce LocalMax, a novel batch active learning method. LocalMax allows the selection of a batch of unlabeled pixels in each step of the active learning process. LocalMax achieves nearly identical accuracies as sequential active learning while being more efficient since multiple query points can be sent to the domain expert at each iteration. In addition, LocalMax can be applied to any type of acquisition function.

2 Background for Graph-based Active Learning Model

In this section, we review basic graph learning and some active learning techniques applied to graph learning classifiers. We construct a similarity graph via a K-nearest neighbors approach [31]. We apply graph Laplace learning [32] with some labeled nodes to classify unlabeled nodes. The labeled nodes are selected through the active learning process.

2.1 Graph Construction

We generate a graph based on the dataset $X = \{x_1, x_2, \ldots, x_N\} \subset \mathbb{R}^d$ of d-dimensional feature vectors. X is indexed by the index set $Z = \{1, 2, \ldots, N\}$. Consider the graph $G(X, W)$ with vertex (node) set X and edge weight matrix $W \in \mathbb{R}^{N \times N}$, where W_{ij} denotes the edge weight between vertices $i \neq j$. The weight W_{ij} is chosen to be proportional to the similarity between corresponding feature vectors x_i and x_j. In our model, we choose

$$W_{ij} = \exp\left(-\frac{\angle(x_i, x_j)^2}{\sqrt{\tau_i \tau_j}}\right), \tag{1}$$

where $\angle(x_i, x_j) = \arccos\left(\frac{x_i^\top x_j}{\|x_i\|\|x_j\|}\right)$ is the angle between feature vectors x_i and x_j. The normalization constant τ_i is chosen according to the similarity to the K^{th} nearest neighbor of i (i.e., $\tau_i = \angle(x_i, x_{i_K})$, where x_{i_K} is the K^{th} nearest neighbor to x_i).

To improve computational efficiency, we require the $N \times N$ weight matrix W to be sparse. For each vertex x_i, we only consider edges between x_i and its K-nearest neighbors (KNN) according to the angle similarity stated above. This can be done by an approximate nearest neighbor search algorithm [31]. Let $x_{i_k},\ k = 1, 2, \ldots, K$ be the K-nearest neighbors of x_i (including x_i itself) according to angle similarity. Define a sparse weight matrix by

$$\bar{W}_{ij} = \begin{cases} W_{ij},\ j = i_1, i_2, \ldots, i_K, \\ 0,\ \text{otherwise.} \end{cases} \tag{2}$$

For practical purposes, K is chosen to ensure that the corresponding graph G is connected. We symmetrize the sparse weight matrix to obtain our final weight matrix by redefining $W_{ij} := (\bar{W}_{ij} + \bar{W}_{ji})/2$. Note that W is sparse, symmetric, and non-negative (i.e. $W_{ij} \geq 0$). In the following experiments of this paper, we choose the parameter $K = 50$ for the K-nearest neighbor search algorithm.

2.2 Graph Learning

With a graph $G(X, W)$ constructed as described in the previous section, we now describe a graph-based approach for semi-supervised learning and present previous work in this field. Assume we have some observations of the ground-truth labels on a subset of vertices $Z_0 \subset Z$. Let $y^\dagger : Z_0 \to \{1, 2, \ldots, n_c\}$ be the ground-truth labeling function that maps each index $j \in Z_0$ to exactly one class label $y_j^\dagger = y^\dagger(j) \in \{1, 2, \ldots, n_c\}$. The corresponding one-hot encoding mapping is $\mathbf{y}^\dagger : Z_0 \to \{e_1, e_2, \ldots, e_{n_c}\}$ defined by $\mathbf{y}^\dagger(j) = e_{y^\dagger(j)}$, where e_k is the k^{th} standard basis vector with all zeros except a 1 at the i^{th} entry. The goal for the semi-supervised learning task is to predict the labels of the *unlabeled* vertices $x_i \in X$, $i \in Z - Z_0$.

Important geometric information about the dataset X is encoded in graph Laplacian matrices [33, 34] defined on G. Define $d_j = \sum_{k \in Z} W_{jk}$ to be the degree of node j and let D be the diagonal matrix with diagonal entries $d_1, d_2, \ldots, d_N$. While there are various graph Laplacians one could define [34], we use the unnormalized graph Laplacian matrix $L_u = D - W$ in this paper.

The inferred classification of unlabeled vertices comes from thresholding a continuous-valued node function $\mathbf{u} : Z \to \mathbb{R}^{n_c}$. In particular, the predicted label of $x_i \in X$ is $y_i = \arg\max\{u_1(i), u_2(i), \ldots, u_{n_c}(i)\}$, where $u_k(i)$ is the k^{th} entry of $\mathbf{u}(i)$. Consider a $N \times n_c$ matrix U, whose i^{th} row is $\mathbf{u}(i)$; that is, each node function $\mathbf{u}$ can be identified by a matrix U whose ith represents the output of $\mathbf{u}$ at node i. The graph-based semi-supervised learning (SSL) model that we consider obtains an optimal $\hat{U}$ (i.e. optimal node function $\hat{\mathbf{u}}$)

by solving an optimization problem of the form:

$$\begin{aligned}\hat{U} &= \arg\min_{u} J_\ell(U, \mathbf{y}^\dagger) \\ &= \arg\min_{U \in \mathbb{R}^{N \times n_c}} \frac{1}{2}\langle U, LU\rangle_F + \sum_{j \in Z_0} \ell(\mathbf{u}(j), \mathbf{y}^\dagger(j)),\end{aligned} \tag{3}$$

where $\langle \cdot, \cdot \rangle_F$ is the Frobenius inner product for matrices.

The loss function $\ell : \mathbb{R}^{n_c} \times \mathbb{R}^{n_c} \to \mathbb{R}$ measures the difference between the prediction $\mathbf{u}(i)$ and the ground-truth $\mathbf{y}^\dagger(i)$ for i in the observation set Z_0. While there are several choices for the loss function, we simply apply a hard-constraint penalty

$$\ell_h(x, y) = \begin{cases} +\infty, & \text{if } x \neq y, \\ 0, & \text{if } x = y. \end{cases} \tag{4}$$

This hard-constraint penalty function ℓ_h forces the minimizer $\hat{U}$ to be exactly the same as the ground-truth $\mathbf{y}^\dagger$ on the observation set Z_0. This SSL scheme was introduced in [32] and we refer to it as Laplace learning. We can reorder the vertices to be able to write $U = \begin{bmatrix} U_l \\ U_u \end{bmatrix}$, where U_l corresponds to the submatrix of U whose rows are in the labeled (observed) index set Z_0 and U_u similarly corresponds to the submatrix of U whose rows are in $Z - Z_0$ (i.e. the unlabeled index set). Likewise, we can split the weight matrix W and degree matrix D into labeled and unlabeled submatrices as

$$W = \begin{bmatrix} W_{ll} & W_{lu} \\ W_{ul} & W_{uu} \end{bmatrix}, \quad D = \begin{bmatrix} D_{ll} & D_{lu} \\ D_{ul} & D_{uu} \end{bmatrix}. \tag{5}$$

As a result of the hard-constraint labeling of Laplace learning, $\hat{U}_l$ is fixed as the one-hot encodings of the observations on the labeled set Z_0. According to [32], the optimizer $\hat{U}_u$ of Laplace learning can be calculated explicitly as

$$\hat{U}_u = (D_{uu} - W_{uu})^{-1} W_{ul} \hat{U}_l. \tag{6}$$

The Laplace learning gives a harmonic solution $\hat{\mathbf{u}}$ on the graph G. It infers the sum-to-one property of the graph Laplace learning output node function $\hat{\mathbf{u}}$. If the ground truth labels are given in one-hot forms, for any node $i \in Z$, we have $\hat{u}_k(i) \geq 0$, $k = 1, 2, \ldots, n_c$ and $\sum_{k=1}^{n_c} \hat{u}_k(x) = 1$, where $\hat{\mathbf{u}}(i) = (\hat{u}_1(i), \hat{u}_2(i), \ldots, \hat{u}_{n_c}(i))$. With this property, at node $i \in Z$, $u_k(i)$ can be treated as the predicted probability that node i belongs to the class k.

There are various other graph SSL schemes based on the optimization problem (3). The main difference between them and the Laplace learning scheme is the choice of penalty function ℓ. In this paper we use the multiclass Gaussian regression (MGR) model [35, 36] which applies a L_2-norm penalty function

$\ell_\gamma(x,y) = \frac{1}{2\gamma^2}\|x-y\|_2^2$. The MGR model is an approximation of the graph Laplace learning model in the sense that $\gamma \to \infty$.

Denote by $\mathcal{G}(N)$ the computation cost of a Laplace leaning process on graph $G = (X, W)$ with the labeled set Z_0. Assume the graph is constructed with the KNN sparse similarity matrix W (Section 2.1) and that the size of labeled set is much smaller than the number of nodes, i.e. $|Z_0| \ll |Z|$. Recall the computational complexity of the conjugate gradient method to solve the linear equation $Ax = b$ (Chapter 10 of [37]) is $O(m\sqrt{\kappa})$, where m and κ are the number of non-zero entries and the condition number of matrix A respectively. If we solve the equation (6) by the conjugate gradient method, we have $\mathcal{G}(N) = O(KN\sqrt{\kappa_L})$, where κ_L is the condition number of the graph Laplacian L.

2.3 Bayesian Interpretation and Truncated Decomposition

A Bayesian interpretation of graph-based SSL models of the form $J_\ell(U, \mathbf{y}^\dagger)$ as in (3) provides further insight into the confidence of inferred classification on the unlabeled nodes[38–40]. The minimizer in (3) is equivalent to the *maximum a posteriori* (MAP) estimate of a posterior probability distribution with probability density function:

$$\mathbb{P}(U|\mathbf{y}^\dagger) \propto \exp(-J_\ell(U, \mathbf{y}^\dagger)), \tag{7}$$

The resulting form of the posterior $\mathbb{P}(U|\mathbf{y}^\dagger)$ depends on the choice of loss function ℓ. For example, when the MGR penalty function is applied, i.e. $\ell(x,y) = \ell_\gamma(x,y) = \frac{1}{2\gamma^2}\|x-y\|_2^2$, $\mathbb{P}(U|\mathbf{y}^\dagger)$ is a Gaussian distribution. The Bayesian interpretation serves as the fundamental of some acquisition functions we introduce later.

$L = L_u$ is a semi-positive definite matrix. By adjusting the number of nearest neighbors K considered at each vertex, we can guarantee that the graph G is connected. Further, the corresponding Laplacian matrix L has exactly one zero eigenvalue. We may order the eigenvalues of L as $0 = \lambda_1 < \lambda_2 \leq \ldots \leq \lambda_N$, and then consider the truncated decomposition of L with the smallest $M < N$ eigenvalues as $\hat{L} = V\Lambda V^\top$, where $\Lambda \in \mathbb{R}^{M\times M}$ is a diagonal matrix with diagonal entries $\lambda_1, \lambda_2, \ldots, \lambda_M$ and $V = [v^1, v^2, \ldots, v^M] \in \mathbb{R}^{N\times M}$ is the matrix of corresponding eigenvectors. v^i is the eigenvector of eigenvalue λ_i.

Define $A = V^\top U \in \mathbb{R}^{M\times n_c}$ and $\hat{A} = V^\top \hat{U} \in \mathbb{R}^{M\times n_c}$, where $\hat{U} = \arg\min_{U\in\mathbb{R}^{N\times n_c}} \tilde{J}_\ell(A, \mathbf{y}^\dagger)$. Let A_m and $\hat{A}_m$ be the respective m^{th} columns of A and $\hat{A}$. When the MGR penalty function is applied, the posterior probability distribution for $A_m|\mathbf{y}^\dagger$ is given by the Gaussian distribution:

$$\begin{aligned} &A|\mathbf{y}^\dagger \sim \mathcal{N}(\hat{A}_m, C_{\text{MGR}}), \\ &C_{\text{MGR}} = \left(\Lambda + V^\top\left(\frac{1}{\gamma^2}P^\top P\right)V\right)^{-1}, \end{aligned} \tag{8}$$

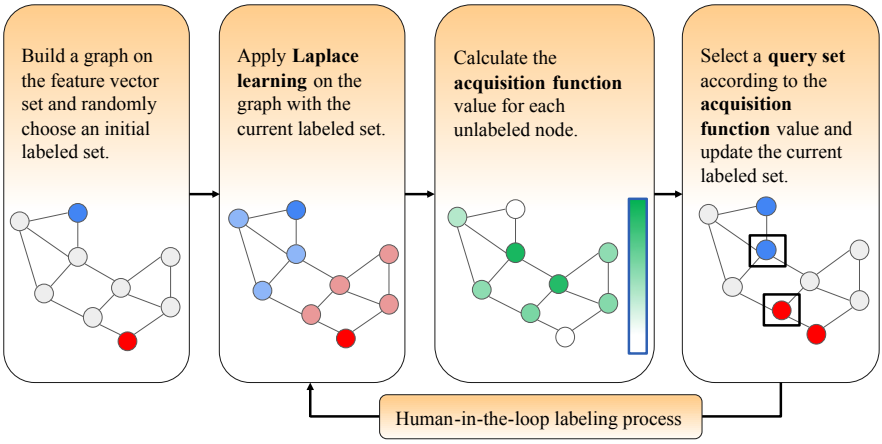


Fig. 1: Flowchart of the active learning process. The active learning loop is based on a fixed graph. In each step, we apply Laplace learning on the graph and update the labeled set with a query set selected based on the current acquisition function values. It should be noticed that it might need the human-in-the-loop process to obtain the label of the selected query set in each step of the active learning process.

where $P \in \mathbb{R}^{|Z_0| \times N}$ is a projection matrix onto the indices corresponding to the labeled set Z_0. When the graph G is connected, then the matrix C_{MGR} is guaranteed to exist, i.e. the matrix $(\Lambda + V^\top(\frac{1}{\gamma^2}P^\top P)V)$ is invertible.

2.4 Active Learning: Acquisition Functions

Active learning improves the performance of the underlying semi-supervised learning (SSL) methods by carefully selecting unlabeled points to hand-label via the use of an oracle or human in the loop. The aim of active learning is to identify which unlabeled inputs ($x_i \in X$ with $i \in Z - Z_0$) for which it would be the "most helpful" to have a human in the loop observe and obtain labels. The core of active learning is the *acquisition function* $\mathcal{A} : Z - Z_0 \to \mathbb{R}$, which evaluates the benefit of obtaining the label of each unlabeled datapoint. The *query set* $\mathcal{Q} \subset Z - Z_0$ of unlabeled points that are to be labeled is chosen via the optimization of an acquisition function. Note that in this work, we use Laplace learning [32] as the underlying semi-supervised classifier. Figure 1 is the flowchart of our active learning process based on the graph Laplace learning classifier. The acquisition functions we introduce are designed for the graph learning classifier, including the Uncertainty (UC)[38–40], Model-Change (MC)[25, 39], Variance Minimization (VOpt)[26], and Model-Change Variance Optimal (MCVOpt) acquisition functions [23].

The UC acquisition function $\mathcal{A}_{\mathrm{UC}}$ quantifies the uncertainty of the classifier $\mathbf{u}$ on each unlabeled node [38–40] by the current classifier's output value for that unlabeled node, (i). Uncertainty sampling thus prioritizes querying points that are close to the current classifier's decision boundaries. Various methods can be applied to quantify the uncertainty based on $\hat{\mathbf{u}}$. Here we consider the **smallest-margin** uncertainty acquisition function:

$$\mathcal{A}_{\mathrm{UC}}(i) = 1 - \left(u_{k_0}(i) - \max_{k=1,2,\ldots,n_c;\ k\neq k_0} u_k(i) \right), \tag{9}$$

where $i \in Z$, $k_0 = \arg\max_{j=1,2,\ldots,n_c} u_k(x)$.

For the VOpt, MC and MCVOpt acquisition functions, in the interest of numerical stability and similar to [23], we replace the hard-constraint penalty function ℓ_h with the MGR penalty function $\ell_\gamma(x,y) = \frac{1}{2\gamma^2}\|x-y\|_2^2$ for MC acquisition function calculations (but not for the underlying SSL model). The MGR penalty ℓ_γ is a numerically stable perturbation of the hard-constraint penalty function ℓ_h. When $\gamma \to 0^+$, $\ell_\gamma \to \ell_h$.

The VOpt acquisition function $\mathcal{A}_{\mathrm{VOpt}}$ is developed to minimize the expected error of the prediction results [26]. If we acquire the label of the unlabeled node $i \in Z \setminus Z_0$ and use labels of $Z \cup \{i\}$ to process the graph learning, then the expected prediction error on the set $Z \setminus (Z_0 \cup \{i\})$ can be computed as follows:

$$\mathbb{E}\left(\sum_{i\in Z\setminus(Z_0\cup\{k\})} \|\mathbf{u}(i) - \mathbf{y}_i^\dagger\|^2 \right) = \mathrm{Tr}(L_k^{-1}), \tag{10}$$

where L_k^{-1} is the submatrix of the graph Laplacian L with both row and column indices $Z\setminus(Z_0\cup\{k\})$. Approximating the matrix L_k^{-1} by the truncated decomposition, we have the VOpt acquisition function:

$$\mathcal{A}_{\mathrm{Vopt}}(k) = \frac{1}{\gamma^2 + v_k^\top C_{\mathrm{MGR}} v_k} \|C_{\mathrm{MGR}} v_k\|_2^2, \tag{11}$$

where C_{MGR} is given by (8) and v_k is the k^{th} column of $V^\top$.

The MC and MCVOpt acquisition function [23, 25, 39] is developed based on the look-ahead model with the objective energy:

$$J_\ell^{k,\hat{\mathbf{y}}_k}(U, \mathbf{y}^\dagger;\ \hat{\mathbf{y}}_k) = \frac{1}{2}\langle U, LU\rangle_F + \sum_{i\in Z_0} \ell(\mathbf{u}(i), \mathbf{y}^\dagger(i)) + \ell(\mathbf{u}(k), \hat{\mathbf{y}}_k). \tag{12}$$

where $\hat{\mathbf{y}}_k$ is the one-hot pseudo-label for the *currently unlabeled* node $k \in Z - Z_0$. Let $\hat{U} = \arg\min(J_\ell(U, \mathbf{y}^\dagger)$ and $\hat{U}^{k,\hat{\mathbf{y}}_k} = \arg\min J_\ell^{k,\hat{\mathbf{y}}_k}(U, \mathbf{y}^\dagger;\ \hat{\mathbf{y}}_k)$. Practically, $\hat{\mathbf{y}}_k$ is the one-hot thresholding vector of $\hat{\mathbf{u}}(k)$ (the k^{th} column of

$\hat{U}$). With the spectral truncation, the MC acquisition function is given by:

$$\begin{aligned}\mathcal{A}_{\mathrm{MC}}(k) &= \|\hat{U}^{k,\hat{\mathbf{y}}_k} - \hat{U}\|_F \\ &= \frac{1}{\gamma^2 + v_k^\top C_{\mathrm{MGR}} v_k} \|C_{\mathrm{MGR}} v_k\|_2 \|\hat{A}^\top v_k - \hat{\mathbf{y}}_k\|_2,\end{aligned} \tag{13}$$

where $\hat{A} = V^\top \hat{U} \in \mathbb{R}^{M \times n_c}$ to be the projection of the matrix U onto the eigenvectors of the graph Laplacian. Similarly, the MCVOpt acquisition function can be written as:

$$\mathcal{A}_{\mathrm{MCVOpt}}(k) = \frac{1}{\gamma^2 + v_k^\top C_{\mathrm{MGR}} v_k} \|C_{\mathrm{MGR}} v_k\|_2^2 \|\hat{A}^\top v_k - \hat{\mathbf{y}}_k\|_2, \tag{14}$$

3 Batch Active Learning Pipeline for Image Segmentation

In this section, we introduce our pipeline with the graph Laplace learning classifier and batch active learning. Given an image for the segmentation task, we extract a feature vector for each pixel and construct a similarity graph based on the cosine similarity between these feature vectors according to Section 2.1. Then we apply the graph Laplace learning (Section 2.2) with labeled pixels selected by our batch active learning approach, LocalMax. The node classification on the graph gives a segmentation of the given image.

3.1 Batch method

In the active learning process illustrated by Figure 1, we select a query set according to a prescribed acquisition function $\mathcal{A}$. In the sequential active learning process, the query set $Q \subset Z - Z_0$ is selected by

$$Q = \{k\}, \ k = \underset{k \in Z - Z_0}{\arg\max}\, \mathcal{A}(k). \tag{15}$$

For batch active learning with batch size B, simply selecting the top-B maximizers of the acquisition function likely includes nodes that are connected in the graph. As an inductive bias, the graph Laplace learning would have similar outputs with neighboring labeled nodes. Therefore, it is redundant to sample neighbors in the graph.

We propose a batch active learning method named LocalMax. This method was originally developed for the classification task of synthetic aperture radar (SAR) datasets [41]. We define the local maximum of a certain node function $\mathcal{A} : Z \to \mathbb{R}$ on a KNN-generated graph G according to Definition 1.

Definition 1 (**Local Max of a Graph Node Function**) Consider a KNN-generated graph $G = (X, W)$, where X is the set of nodes indexed by Z and W is

the edge weight matrix. For a graph node function $\mathcal{A} : Z \to \mathbb{R}$, $k \in Z$ is a *local maximum node* if and only if for any j, $\mathcal{A}(k) \geq \mathcal{A}(j)$, if there is an edge between x_j and x_k. Equivalently, $k \in Z$ is a local maximum if and only if:

$$\mathcal{A}(k) \geq \mathcal{A}(j), \ \forall J \text{ s.t. } W_{jk} > 0. \tag{16}$$

The LocalMax batch active learning method selects the batch query set to be the top-B local maximums of the acquisition function $\mathcal{A}$ in the graph G. Algorithm 1 shows the process of the LocalMax batch active learning method. It should be noted that the batch size B can not be extremely large as there might not be enough local maximums of the discrete set of acquisition function values at a given iteration. Algorithm 1 has the maximal computational complexity $O(KN)$, where K is the KNN parameter (Section 2.1) and N is the number of nodes in the graph G. Usually $K \ll N$, the computational complexity is $O(N)$.

Algorithm 1 LocalMax Batch Active Learning

Require: A KNN graph $G = (X, W)$, X is indexed by Z. A labeled index set Z_0. An acquisition function $\mathcal{A} : Z - Z_0 \to \mathbb{R}^+$. Batch size B.
Ensure: The query set $\mathcal{Q}$.

Extend the domain of $\mathcal{A}$ to Z by defining $\mathcal{A}(j) = 0, \ \forall j \in Z_0$
$S \leftarrow Z - Z_0$; $\mathcal{Q} = \emptyset$
while $S \neq \emptyset$ and $|\mathcal{Q}| < B$ **do**
    $k \leftarrow \arg\max_{k \in S} \mathcal{A}(k)$
    $N(k) = \{i \in Z : W_{jk} > 0\}$
    if $\mathcal{A}(k) \geq \mathcal{A}(i), \ \forall i \in N(k)$ **then**
        $Q \leftarrow Q \cup \{k\}$
    end if
    $S \leftarrow S - N(k)$
end while

This method has some important advantages. Practically, regions of high acquisition value will only have a small number of local maxima, so the LocalMax method obtains a batch of nodes from multiple regions of high acquisition. Due to the complicated structure of data, there are often many regions with high acquisition, so batches can be relatively large. This method also selects what the model predicts to be the most important point from the high-acquisition region.

According to Section 2.2, the computational cost of the graph Laplacian learning is $\mathcal{G}(N) = O(KN\sqrt{\kappa_L})$. If we want to sample a total number of M query nodes from the whole active learning process, the computational complexity of the sequential active learning process is $M\mathcal{G}(N)$ while the LocalMax batch active learning with the batch size B has the computational complexity

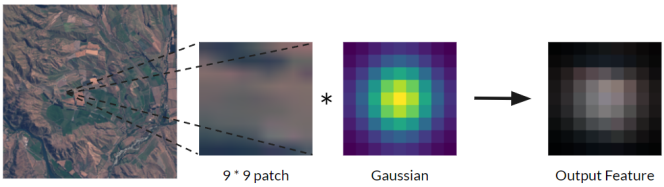


Fig. 2: The feature extraction process for a single pixel. The feature vector is a Gaussian-weighted patch centered on the pixel.

$M/B[O(N)+\mathcal{G}(N)]$. This implies that the LocalMax batch active learning process is much more efficient than the sequential active learning, proportionally to the batch size B. This result is verified by experiments in Section 4.1.

3.2 Image Segmentation Pipeline

We develop an image segmentation pipeline with graph learning and batch active learning approaches mentioned in Section 2 and 3.1. The first step for pixel classification is to associate each pixel with a feature vector. One can consider simply using the pixel values of all channels as the corresponding feature vector. In this case, the dimension of a feature vector is the same as the number of channels of the image. While this simple construction is straightforward, it is useful to include neighborhood information of each pixel for feature extraction.

For pixel i, consider a $(2k+1)\times(2k+1)$ neighborhood patch P_i centered at pixel i. If pixel i is near the boundary of the image, apply reflection padding to expand the image before taking the neighborhood patch. Inspired by the non-local means method [42], we consider a $(2k+1)\times(2k+1)$ discrete Gaussian kernel G with $\sigma = k/2$. Specifically,

$$G(i,j) = \frac{\alpha}{2\pi\sigma^2}\exp\left(-\frac{(i-k-1)^2+(j-k-1)^2}{2\sigma^2}\right),$$

where α is a constant such that $\sum_{i,j=1}^{2k+1} G(i,j) = 1$. The weighted patch is then defined by

$$P_i^w(i,j) = P_i(i,j)G(i,j),$$

for each pair of pixels $i,j = 1,2,\ldots,2k+1$. This feature process is illustrated by figure 2.

We apply this non-local means weighting process to each of the C channels in the image. Flattening these weighted patches and concatenating them together gives the non-local means feature vector for pixel i. The dimension of the resulting non-local means feature vector for a given pixel is $d = C(2k+1)^2$.

For a given image, we extract the non-local means feature vector for each of its pixels to get the feature vector set X. Then we build a similarity graph $G = (X, W)$ based on the feature vector set S and with the sparse similarity weight matrix W via the KNN method based on these according to Section 2.1.

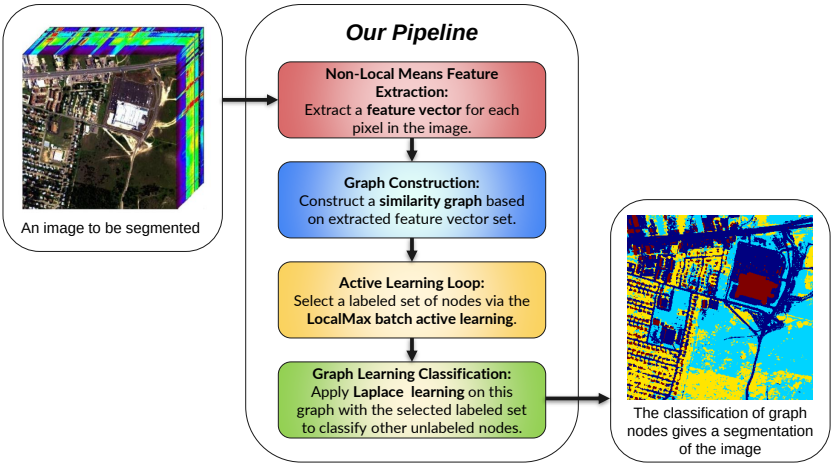


Fig. 3: Our graph-based active learning pipeline for the image segmentation task. Red box: feature extraction (Section 3.2); Blue box: Graph Construction (Section 2.1); Yellow box: Batch Active Learning (Section 2.4 and 3.1); Green box: Graph Learning (Section 2.2).

On the graph G, we randomly initialize a labeled node set and apply the LocalMax batch active learning to select a labeled set of nodes Z_0 according to Section 2.4 and 3.1. Finally, we predict the labels of unlabeled nodes $Z - Z_0$ in graph G with the graph Laplace learning classifier based on the selected labeled node set Z_0. This node classification on G gives a segmentation on the given image. The flowchart of our pipeline is Figure 3.

4 Experiments and Results

This section shows the experiments and results of our graph-based active learning pipeline on the image segmentation tasks. Based on our contributions in Section 1.1, we run two types of experiments: the comparison between LocalMax and sequential active learning process and the application of our pipeline on image segmentation tasks with low label rates.

4.1 Comparison between LocalMax and Sequential Active Learning: Accuracy and Efficiency

We apply our graph-based active learning method to perform image segmentation on the Urban dataset. The Urban dataset was recorded in October 1995 by the Hyperspectral Digital Imagery Collection Experiment (HYDICE) over the urban area in Copperas Cove, TX, U.S.. The ground truth labels are provided by Zhu. et al. [43] and include four classes: asphalt, grass, tree, and roof. This dataset contains a hyperspectral image with a size 307×307 pixels,

each of which corresponds to a 2 square meters area. The raw image includes 210 channels, but we use the clean version with 162 channels after removing certain channels due to dense water vapor and atmospheric effects. Figure 4 shows the raw image and ground truth labels of the Urban dataset.

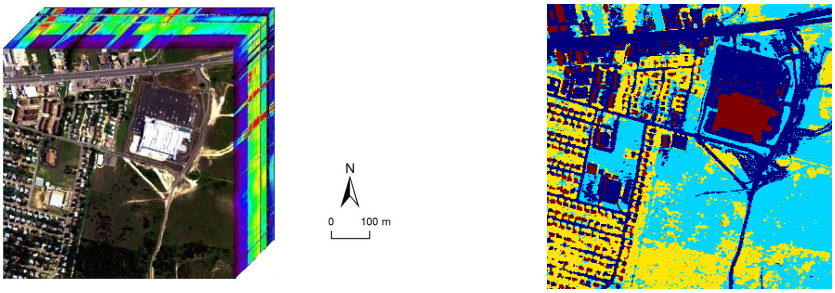
Here we simply apply the hyperspectral pixel values as the feature vector, i.e. for each pixel, the feature vector corresponding to it is the vector of 162 channels. With four acquisition functions, we sample up to 134 pixels (0.15% of all pixels) with different active learning sampling methods. We initialize the labeled set with 10 random pixels in each class (in total 40 pixels) and sample extra 94 pixels according to the active learning methods. For a certain acquisition function $\mathcal{A}$, we consider four sampling methods: Sequential, Random, Top-Max, and LocalMax, the last three of which are batch active learning with a batch size B.

- **Sequential sampling** selects the global maximum node of $\mathcal{A}$ to update the current labeled set. The query set $\mathcal{Q} = \{k^*\}$ and $k^* = \arg\max_{k \in Z - Z_0} \mathcal{A}(k)$.
- **Random sampling** selects a batch of B unlabeled nodes according to the uniform distribution on the unlabeled node set $Z - Z_0$.
- **Top-Max sampling** selects a batch of B unlabeled nodes as the top-B maximum of $\mathcal{A}$, i.e. the query set $\mathcal{Q} = \{i_1, i_2, \ldots, i_B\} \subset Z - Z_0$ where $i_1 = \arg\max_{i \in Z - Z_0} \mathcal{A}(i)$ and $i_b = \arg\max_{i \in Z - Z_0 - \{i_1, \ldots, i_{b-1}\}} \mathcal{A}(i)$ for $b = 2, 3, \ldots, B$.
- **LocalMax sampling** is the method we proposed in Section 3.1.

Figure 5 shows the curves between the accuracy and the number of labeled pixels for the four acquisitions and Table 1 shows the time consumption and accuracy values for label rates $0.1\%, 0.15\%$. From these experiments, we conclude the following:

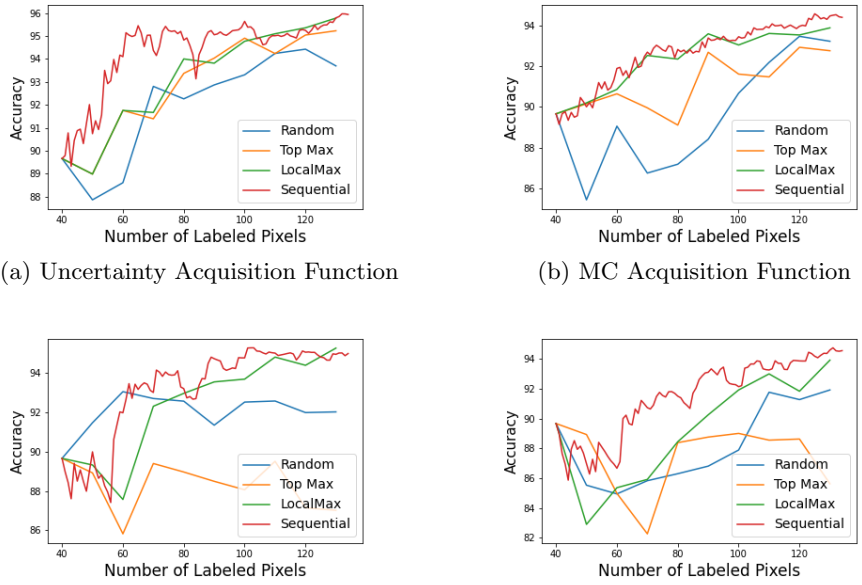
1. **Accuracy Performance:** Sequential active learning has the best accuracy performance according to Figure 5 which shows its higher accuracy for almost all numbers of labeled pixels. Our batch active learning method LocalMax has the second-best accuracy values and performs almost identically as the sequential one, especially for larger numbers of labeled pixels (i.e. $[80, 130]$). This is also verified by Table 1 which shows in bold the top-2 accuracy values. LocalMax consistently shows accuracies in the top 2 and sometimes performs better than Sequential.
2. **Efficiency Performance:** According to the timings in Table 1, LocalMax takes approximately the same time as the Random and Top-Max sampling methods while the Sequential active learning takes around 8 times longer. The time multiplier 8 is close to the theoretical multiplier 10 (same as the batch size B) according to Section 3.1.

In summary, LocalMax batch active learning is much more efficient than Sequential active learning without significantly sacrificing accuracy.



(a) Raw image (b) Ground Truth Labels

Fig. 4: Urban Dataset. Panel(a) shows the raw hyperspectral image we used for experiments. Panel(b) shows the ground-truth labels. Label information: asphalt (navy blue), grass (light blue), trees (yellow), roof (red).



(a) Uncertainty Acquisition Function (b) MC Acquisition Function

(c) MCVOpt Acquisition Function. (d) VOpt Acquisition Function

Fig. 5: Comparison between batch active learning methods and sequential active learning for four acquisition functions. Each panel includes four curves, of which the X-axis is the number of labeled pixels and the Y-axis is the accuracy. The blue, yellow, green, and red curves correspond to the Random, Top-Max, LocalMax, and Sequential sampling method respectively for the active learning process. More details on accuracy values and time consumption are shown in Table 1. Descriptions of each sampling method are in Section 4.1.

Table 1: Comparison between different Active Learning Sampling Methods

Label Percentage			*0.1%*		*0.15%*	
Acq	***Sampling***	***B***	*Time*(s)	*Acc*(%)	*Time*(s)	*Acc*(%)
UC	Sequential	1	1290.13	**94.90**	2576.86	**95.93**
	Random	10	180.42	92.86	358.84	93.69
	Top-Max	10	169.59	**94.03**	343.24	95.22
	LocalMax	10	165.30	93.81	326.56	**95.78**
MC	Sequential	1	1250.22	**92.70**	2505.60	**94.40**
	Random	10	166.59	88.41	327.95	93.22
	Top-Max	10	165.07	92.68	327.35	92.71
	LocalMax	10	165.28	**93.59**	334.10	**93.88**
MCVOpt	Sequential	1	1131.92	**93.70**	2257.06	**94.99**
	Random	10	161.83	91.34	322.44	92.02
	Top-Max	10	160.61	88.49	323.99	87.02
	LocalMax	10	164.79	**93.55**	321.92	**95.27**
MC	Sequential	1	1289.92	**92.60**	2576.86	**94.54**
	Random	10	175.99	86.80	346.61	91.91
	Top-Max	10	169.91	88.75	339.96	85.62
	LocalMax	10	175.61	**90.24**	357.60	**93.91**

This table shows the efficiency and accuracy performance of active learning sampling methods with different acquisition functions. The first row 'Acq' refers to the acquisition function. The second row 'Sampling' refers to the choice of active learning sampling methods, including Sequential, Random, Top-Max, and LocalMax-the last three of which are batch active learning. The third row B is the batch size. We sample up to 0.1% and 0.15% labeled pixels in the Urban dataset and show the timings and accuracies. The top two accuracy values are bolded for each acquisition function. Descriptions of each sampling method are in Section 4.1

4.2 Semi-Supervised Image Segmentation with Low Label Rates

Here we perform segmentation experiments on three datasets evaluated on overall accuracy as a function of the amount of labeled data used in training. Our method is compared with the **graph-based semi-supervised method**, abbr. **GL-SSL**, and **MBO unsupervised method**, abbr. **MBO-UN**, proposed in [14]. There are some differences between our graph learning method and the GL-SSL proposed in [14], the foremost being that our method uses the KNN approach to build a sparse similarity graph while the GL-SSL is based on a fully connected graph and uses the Nyström extension method to approximate the graph Laplacian matrix.

In the following experiments, we consider the overall accuracy (OA) at different percentages of labeled pixels. The definition of overall accuracy is:

$$\text{OA} = \frac{\text{Number of Correctly Classified Pixels}}{\text{Total Number of Pixels}}. \tag{17}$$

4.2.1 Landsat-7 Dataset

We select a Landsat-7 multispectral image of the Colville River (Alaska, USA) from the RiverPIXELS dataset [44], which provides paired Landsat and water-and-sediment labeled patches of size $256 \times 256 \times 6$, where the 6 multispectral channels correspond to Blue, Green, Red, Near IR, Shortwave IR 1, and Shortwave IR 2. Each pixel in the image covers roughly 900 m^2. We aim to segment them into three classes: land, water, and bare sediment as provided by RiverPIXELS.

We use the non-local means feature vector of each pixel to build the similarity graph. The neighborhood patch size is 7×7, which leads to a 294 dimensional feature vector for each pixel. We sample up to 200 pixels (0.3% of all pixels) based on the random initialization of one pixel in each class (three pixels in total for the initialization) via the LocalMax batch active learning approach of batch size 20. Table 2 shows the overall accuracy results. In addition, we sample up to 3300 (around 5% of all pixels) labeled pixels based on the random initialization of 10 labeled pixels in each class (30 in total) and batch size 100. Both the UC and MCVOpt acquisition functions reach a better accuracy with 0.3% labeled pixels than randomly selecting 5% labeled pixels. Figure 6 shows the segmentation result of our graph-based batch active learning method with UC acquisition functions.

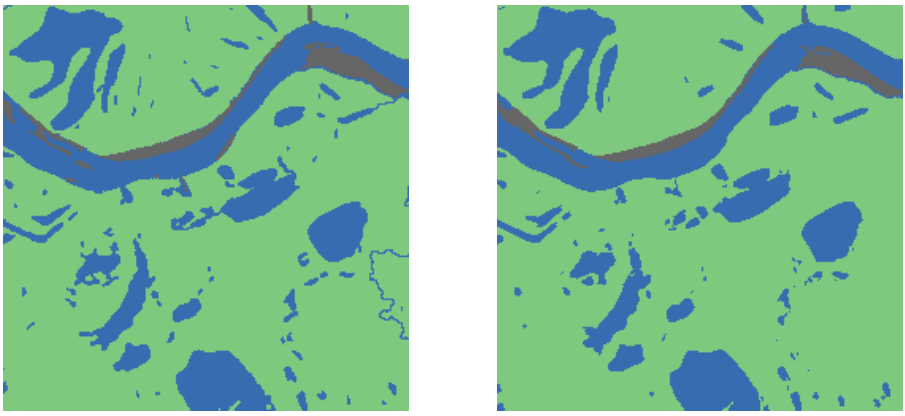
Table 2: Overall Accuracy of a Landsat-7 Multispectral Image (65536 pixels)

Labeled Percentages		*0.1%*	*0.2%*	*0.3%*	*5%*
Method	***Sampling***				
Ours	LocalMax UC	25.17%	**95.92%**	96.23%	**98.65%**
Ours	LocalMax MC	91.75%	95.04%	95.80%	97.50%
Ours	LocalMax MCVOpt	**94.81%**	95.23%	**96.25%**	97.74%
Ours	LocalMax VOpt	93.61%	94.4%	94.78%	96.44%
Ours	Random	88.43%	91.38%	92.55%	96.12%

This table shows the overall accuracy of the Landsat-7 multispectral image among different methods and under different label rates. 'Sampling' shows how we selected training pixels for this dataset. LocalMax batch active learning method has a batch size of 20 for columns $0.1\%, 0.2\%, 0.3\%$, while the batch size is 100 for column 5%.

4.2.2 Urban Dataset

The Urban dataset is introduced in Section 4.1. For these experiments, we use the hyperspectral pixel values as the feature vector for each pixel. We sample up to 286 pixels (0.3% of all pixels) based on the random initialization of one pixel in each class (four pixels in total for the initialization) with LocalMax with batch size 10. In addition, we sample 4700 (around 5% of all pixels) with LocalMax with batch size 100 based on the random initialization of 10 pixels in each class. Overall accuracies of these two sampling processes together



(a) Ground-truth labels (b) OA: 96.23%

Fig. 6: The ground-truth and segmentation result of the a Landsat-7 multispectral image from the RiverPIXELS dataset. Panel (a): ground-truth labels; Panel (b): Segmentation result with 0.3% labeled pixels sampled according to LocalMax with a batch size of 20 and the UC acquistion function. The similarity graph is based on non-local means feature vectors with the neighborhood patch- of size 7×7.

with the randomly selected labeled set and methods in the other paper are shown in Table 3. It can be seen that LocalMax batch active learning with the uncertainty (UC) acquisition function attains an accuracy of 97.30% with only 0.3% labeled pixels, which is similar to the accuracy 97.76% with 10% randomly selected labeled pixels. In addition, our graph learning classifier performs better than the GL-SSL method with the same 10% randomly sampled labeled pixels. Figure 7 shows two sampled segmentation results of the UC and MCVOpt acquisition functions respectively.

4.2.3 Kennedy Space Center (KSC) Dataset

The Kennedy Space Center Dataset is a hyperspectral image at the Kennedy Space Center (KSC) in Florida, acquired by the NASA AVIRIS (Airborne Visible/Infrared Imaging Spectrometer) instrument. This hyperspectral image has size 512×614. The raw image includes 224 channels, but we are using the clean version with 176 channels after removing water absorption and low SNR channels.

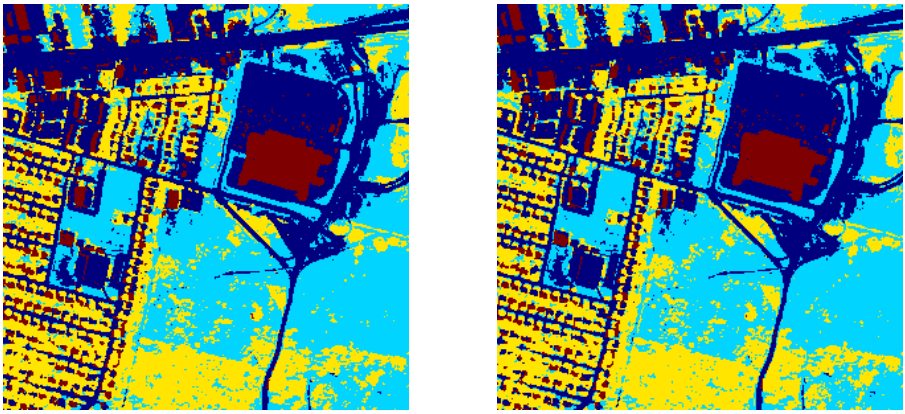
There are 314368 pixels in this dataset while only 5211 (around 1.66%) of them have ground-truth labels. The ground-truth labels include 13 classes of different kinds of land coverings in this region. It is visualized in Figure 8a. We thus segment this hyperspectral image into 13 classes. Our results are calculated based only on the 5211 pixels with ground-truth labels.

Table 3: Overall Accuracy of Urban Dataset (94249 pixels)

Labeled Percentages		*0.1%*	*0.2%*	*0.3%*	*5%*	*10%*
Method	***Sampling***					
Ours	LM UC	**94.96%**	**95.56%**	**97.30%**	**99.71%**	N\A
Ours	LM MC	90.71%	93.31%	94.84%	98.60%	N\A
Ours	LM MCVOpt	94.13%	95.33%	95.35%	98.94%	N\A
Ours	LM VOpt	91.32%	94.27%	94.52%	97.44%	N\A
Ours	Random	87.20%	91.93%	93.30%	97.20%	**97.76%**
GL-SSL[1]	Random	N\A	N\A	N\A	N\A	93.48%
UN-MBO[1]	Unsupervised	92.35% with **No Labeled Pixel**				

This table shows the overall accuracy of the Urban dataset among different methods and under different label rates. 'Sampling' describes how training pixels were selected. LocalMax batch active learning has batch size 10 for columns $0.1\%, 0.2\%, 0.3\%$ while the batch size is 100 for column 5%.

[1]Method proposed in [14]

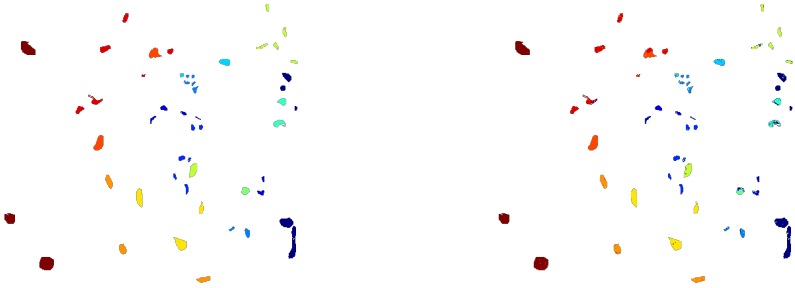


(a) OA: 97.30% (b) OA: 95.35%

Fig. 7: The segmentation result of the Urban dataset with 0.3% labeled pixels sampled according to LocalMax batch active learning with batch size 10. Panel(a): UC acquisition function; Panel(b): MCVOpt acquistion function. Label information: asphalt (navy blue), grass (light blue), trees (yellow), roof (red). The ground-truth labels are in Figure 4b.

We sample up to 325 pixels (6% of all pixels with ground-truth labels) based on the random initialization of 1 pixel in each class (13 pixels in total for the initialization).

Table 4 shows the overall accuracy of the KSC dataset. LocalMax batch active learning with the uncertainty (UC) acquisition function attains an accuracy of 89.83% with 6% labeled pixels, outperforming random selection of 20%



(a) Ground-truth labels (b) OA: 89.83%

Fig. 8: The ground-truth and segmentation result of the KSC dataset. Panel (a): ground-truth labels of 5211 pixels, including 13 classes; Panel (b): Segmentation result with 6% labeled pixels sampled according to LocalMax batch active learning with batch size 10 and the UC acquistion function.

pixels. Figure 8b shows the segmentation result of the our graph-based batch active learning method with UC acquisition functions.

Table 4: Overall Accuracy of KSC Dataset (5122 pixels with ground-truth labels)

Labeled Percentages		*2.5%*	*4%*	*6%*	*20%*
Method	***Sampling***				
Ours	LocalMax UC	**85.47%**	**88.66%**	**89.83%**	N\A
Ours	LocalMax MC	82.04%	85.86%	87.85%	N\A
Ours	LocalMax MCVOpt	85.12%	85.86%	87.13%	N\A
Ours	LocalMax VOpt	83.22%	84.68%	86.66%	N\A
Ours	Random	82.22%	85.22%	85.83%	**88.91%**
GL-SSL[1]	Random	80.37%	N\A	N\A	N\A

This table shows the overall accuracy of the KSC dataset among different methods and under different label rates. 'Sampling' describes how we selected training pixels. LocalMax batch active learning is performed with a batch size of 5. Although the KSC dataset includes 314368 pixels, the accuracy values and the percentage of labeled pixels in this table are calculated based on 5211 labeled pixels.

[1]Method proposed in [14]

5 Conclusion

We propose a graph-based batch active learning pipeline for multi- and hyperspectral image segmentation . Our method showed excellent image segmentation skill using very low percentages of available training data. Compared

with a similar graph-based image segmentation method proposed in [14], our method requires fewer labeled pixels to achieve better overall accuracy. This suggests that careful selection points to label through active learning is beneficial for this application of graph-based semi-supervised classification. In addition, we introduced a batch active learning approach, LocalMax, to select a batch of pixels in each step of the active learning process. According to our experiments, LocalMax batch active learning not only accelerates the process of sampling pixels but also retains similar accuracies as sequential active learning using the same acquisition function.

Acknowledgments. Bohan Chen is supported by the UC-National Lab In-Residence Graduate Fellowship Grant L21GF3606. Kevin Miller was supported by a DOD National Defense Science and Engineering Graduate (NDSEG) Research Fellowship. Jon Schwenk is supported by the Laboratory Directed Research and Development program of Los Alamos National Laboratory under project numbers 20170668PRD1 and 20210213ER. Andrea Bertozzi is supported by the NGA under Contract No. HM04762110003. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NGA.

References

[1] Mumford, D.B., Shah, J.: Optimal approximations by piecewise smooth functions and associated variational problems. Communications on pure and applied mathematics (1989)

[2] Kass, M., Witkin, A., Terzopoulos, D. International Journal of Computer Vision, 321–331 (1988)

[3] Chan, T.F., Vese, L.A.: Active contours without edges. IEEE Transactions on image processing **10**(2), 266–277 (2001)

[4] Bresson, X., Esedoḡlu, S., Vandergheynst, P., Thiran, J.-P., Osher, S.: Fast global minimization of the active contour/snake model. Journal of Mathematical Imaging and Vision **28**(2), 151–167 (2007)

[5] Gilboa, G., Osher, S.: Nonlocal operators with applications to image processing. Multiscale Modeling & Simulation **7**(3), 1005–1028 (2009)

[6] Merkurjev, E., Sunu, J., Bertozzi, A.L.: Graph mbo method for multiclass segmentation of hyperspectral stand-off detection video. In: 2014 IEEE International Conference on Image Processing (ICIP), pp. 689–693 (2014). IEEE

[7] Merkurjev, E., Kostic, T., Bertozzi, A.L.: An mbo scheme on graphs for classification and image processing. SIAM Journal on Imaging Sciences **6**(4), 1903–1930 (2013)

[8] Boyd, Z.M., Porter, M.A., Bertozzi, A.L.: Stochastic block models are a discrete surface tension. Journal of Nonlinear Science **30**(5), 2429–2462 (2020)

[9] Boyd, Z.M., Bae, E., Tai, X.-C., Bertozzi, A.L.: Simplified energy landscape for modularity using total variation. SIAM Journal on Applied Mathematics **78**(5), 2439–2464 (2018)

[10] Hu, H., Sunu, J., Bertozzi, A.L.: Multi-class graph mumford-shah model for plume detection using the mbo scheme. Proceedings of the EMMCVPR conference in Hong Kong 2015 **8932**, 209–222. X. -C. Tai et al (Eds), Springer Lecture Notes in Computer Science

[11] Hu, H., Laurent, T., Porter, M.A., Bertozzi, A.L.: A method based on total variation for network modularity optimization using the mbo scheme. SIAM Journal on Applied Mathematics **73**(6), 2224–2246 (2013)

[12] Bertozzi, A.L., Flenner, A.: Diffuse interface models on graphs for classification of high dimensional data. Multiscale Modeling & Simulation **10**(3),

1090–1118 (2012)

[13] Garcia-Cardona, C., Merkurjev, E., Bertozzi, A.L., Flenner, A., Percus, A.G.: Multiclass data segmentation using diffuse interface methods on graphs. IEEE Transactions on Pattern Analysis and Machine Intelligence **36**(8), 1600–1613 (2014)

[14] Meng, Z., Merkurjev, E., Koniges, A., Bertozzi, A.L.: Hyperspectral image classification using graph clustering methods. Image Processing On Line **7**, 218–245 (2017)

[15] Ciurte, A., Bresson, X., Cuisenaire, O., Houhou, N., Nedevschi, S., Thiran, J.-P., Cuadra, M.B.: Semi-supervised segmentation of ultrasound images based on patch representation and continuous min cut. PLoS ONE **9**(7) (2014)

[16] O'Shea, K., Nash, R.: An introduction to convolutional neural networks. arXiv preprint arXiv:1511.08458 (2015)

[17] Wu, F., Souza, A., Zhang, T., Fifty, C., Yu, T., Weinberger, K.: Simplifying graph convolutional networks. In: International Conference on Machine Learning, pp. 6861–6871 (2019). PMLR

[18] Thorpe, M., Nguyen, T.M., Xia, H., Strohmer, T., Bertozzi, A., Osher, S., Wang, B.: Grand++: Graph neural diffusion with a source term. In: International Conference on Learning Representations (2021)

[19] Bertozzi, A.L., Merkurjev, E.: Graph-based optimization approaches for machine learning, uncertainty quantification and networks **20**, 503–531 (2019)

[20] Iyer, G., Chanussot, J., Bertozzi, A.L.: A Graph-Based Approach for Data Fusion and Segmentation of Multimodal Images. IEEE Transactions on Geoscience and Remote Sensing **59**(5), 4419–4429 (2021). https://doi.org/10.1109/TGRS.2020.2971395

[21] Qin, J., Lee, H., Chi, J.T., Drumetz, L., Chanussot, J., Lou, Y., Bertozzi, A.L.: Blind hyperspectral unmixing based on graph total variation regularization. IEEE Transactions on Geoscience and Remote Sensing **59**(4), 3338–3351 (2021). https://doi.org/10.1109/TGRS.2020.3020810

[22] Settles, B.: Active Learning vol. 6, pp. 1–114. Morgan & Claypool Publishers LLC, Carnegie Mellon University, USA (2012). https://doi.org/10.2200/s00429ed1v01y201207aim018. https://doi.org/10.2200/s00429ed1v01y201207aim018

[23] Miller, K., Mauro, J., Setiadi, J., Baca, X., Shi, Z., Calder, J., Bertozzi,

A.L.: Graph-based active learning for semi-supervised classification of SAR data (2022). https://arxiv.org/abs/2204.00005

[24] Dasgupta, S.: Two faces of active learning. Theoretical Computer Science **412**(19), 1767–1781 (2011). https://doi.org/10.1016/j.tcs.2010.12.054

[25] Miller, K., Bertozzi, A.L.: Model-change active learning in graph-based semi-supervised learning. arXiv preprint arXiv:2110.07739 (2021)

[26] Ji, M., Han, J.: A variance minimization criterion to active learning on graphs. In: Artificial Intelligence and Statistics, pp. 556–564 (2012). PMLR

[27] Ma, Y., Garnett, R., Schneider, J.G.: Σ-optimality for active learning on Gaussian random fields. In: NIPS, pp. 2751–2759 (2013)

[28] Cai, W., Zhang, Y., Zhou, J.: Maximizing expected model change for active learning in regression. In: 2013 IEEE 13th International Conference on Data Mining, pp. 51–60 (2013). IEEE

[29] Gal, Y., Islam, R., Ghahramani, Z.: Deep Bayesian active learning with image data. In: International Conference on Machine Learning, pp. 1183–1192 (2017). PMLR

[30] Kushnir, D., Venturi, L.: Diffusion-based deep active learning. arXiv preprint arXiv:2003.10339 (2020)

[31] Arya, S., Mount, D.M., Netanyahu, N.S., Silverman, R., Wu, A.Y.: An optimal algorithm for approximate nearest neighbor searching fixed dimensions. Journal of the ACM (JACM) **45**(6), 891–923 (1998)

[32] Zhu, X., Ghahramani, Z., Lafferty, J.D.: Semi-supervised learning using Gaussian fields and harmonic functions. In: Proceedings of the 20th International Conference on Machine Learning (ICML-03), pp. 912–919 (2003)

[33] Belkin, M., Niyogi, P., Sindhwani, V.: Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. Journal of machine learning research **7**(11) (2006)

[34] Von Luxburg, U.: A tutorial on spectral clustering. Statistics and computing **17**(4), 395–416 (2007)

[35] Ma, Y., Huang, T.-K., Schneider, J.G.: Active search and bandits on graphs using sigma-optimality. In: UAI, vol. 542, p. 551 (2015)

[36] Bertozzi, A.L., Hosseini, B., Li, H., Miller, K., Stuart, A.M.: Posterior consistency of semi-supervised regression on graphs. Inverse Problems

37(10), 105011 (2021)

[37] Shewchuk, J.R., et al.: An introduction to the conjugate gradient method without the agonizing pain. Carnegie-Mellon University. Department of Computer Science Pittsburgh (1994)

[38] Bertozzi, A.L., Luo, X., Stuart, A.M., Zygalakis, K.C.: Uncertainty quantification in the classification of high dimensional data. SIAM/ASA J. Uncertainty Quantification **6**(2), 568–595 (2018)

[39] Miller, K., Li, H., Bertozzi, A.L.: Efficient graph-based active learning with probit likelihood via Gaussian approximations (2020). arXiv: 2007.11126

[40] Qiao, Y., Shi, C., Wang, C., Li, H., Haberland, M., Luo, X., Stuart, A.M., Bertozzi, A.L.: Uncertainty quantification for semi-supervised multi-class classification in image processing and ego-motion analysis of body-worn videos. Electronic Imaging **2019**(11), 264–1 (2019)

[41] Chapman, J., Chen, B., Tan, Z., Calder, J., Miller, K., Bertozzi, A.L.: Novel batch active learning approach and its application on the synthetic aperture radar datasets (2023)

[42] Buades, A., Coll, B., Morel, J.-M.: A non-local algorithm for image denoising. In: 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), vol. 2, pp. 60–65 (2005). IEEE

[43] Zhu, F., Wang, Y., Xiang, S., Fan, B., Pan, C.: Structured sparse method for hyperspectral unmixing. ISPRS Journal of Photogrammetry and Remote Sensing **88**, 101–118 (2014)

[44] Schwenk, J., Rowland, J.C.: RiverPIXELS: paired Landsat images and expert-labeled sediment and water pixels for a selection of rivers v1.0. a global, high-resolution river network model for improved flood risk prediction. Technical report, ESS-DIVE Deep Insight for Earth Science Data (2022). https://data.ess-dive.lbl.gov/view/doi:10.15485/1865732