AMRITA VISHWA VIDYAPEETHAM

22MAT230 MATHEMATICS FOR COMPUTING 4

Batch : AIE -C

Team C-11

Jovika N B	- CB.SC.U4AIE24223
P Poojitha	- CB.SC.U4AIE24237
Ramu Jahna Bindu	- CB.SC.U4AIE24249
Potnuru Varun	- CB.SC.U4AIE24244

Density-Adaptive Graph Laplacian Construction for Robust Spectral Clustering

1: Introduction :

Clustering satellite images is a challenging task due to high dimensionality, complex spatial patterns, and variations in data density. Traditional clustering methods such as K-Means fail to handle these challenges effectively. To overcome these limitations, this project employs **density-adaptive spectral clustering**, which combines deep feature extraction with graph-based clustering.

2:Data Representation and Feature Extraction :

To convert satellite images into numerical feature vectors suitable for mathematical analysis.

Mathematical Representation:

Let the dataset be defined as:

$$\mathcal{X} = \{x_1, x_2, \ldots, x_N\}$$

where N denotes the total number of images in the dataset.

Each image is transformed into a feature vector using a pretrained deep neural network:

$$x_i = f(\text{image}_i)$$

where $f(\cdot)$ denotes the feature extraction function.

The resulting feature vector lies in a d-dimensional space

$$x_i \in \mathbb{R}^d$$

In this work, the feature dimension is fixed as:

$$d = 2048$$

All extracted feature vectors are stacked row-wise to form a feature matrix:

$$X \in \mathbb{R}^{N \times d}$$

where $X = [x_1; x_2; \ldots; x_N]$.

Each row of the matrix X corresponds to one image, and each column represents a specific feature.

Explanation:

This transformation reduces the high-dimensional raw pixel representation of images into a compact and semantically meaningful feature space. Using deep features ensures that visually similar land-cover images produce similar feature vectors, which is essential for effective clustering.

The feature matrix X serves as the input for all subsequent steps, including distance computation, graph construction, and spectral clustering.

3:Distance Matrix Computation:

Compute the pairwise distances between all feature vectors obtained in previous step. These distances quantify the similarity between satellite images and form the basis for graph construction in subsequent steps.

Mathematical Representation:

After feature extraction, the dataset is represented by the feature matrix:

$$X = [x1; x2; \ldots; xN] \in \mathbb{R}^{N\times d}$$

Distance Measure :

The similarity between two images is measured using the Euclidean distance, defined as

$$dij = \| \ xi - xj \ \|_2$$

where:

-xi and xj are feature vectors of images i and j.

-d_{ij} represents the distance between them.

Distance Matrix Definition :

The pairwise distances between all images are stored in a distance matrix:

$$D = [d_{ij}] \in \mathbb{R}^{N\times N}$$

where each element is $D(i, j) = d_{ij}$

The distance matrix is computed by evaluating the Euclidean distance between every pair of rows in the feature matrix X.

The resulting matrix D is symmetric and satisfies:

$$D(i, i) = 0 \forall i$$

The distance matrix D provides a complete measure of similarity across the dataset. Smaller values of dij indicate that images i and j are visually similar, while larger values indicate dissimilarity. This information is essential for identifying local neighborhoods during graph construction

4: k-Nearest Neighbor Graph Construction :

This step is to construct a graph representation of the dataset that preserves local neighborhood relationships between satellite images. Instead of connecting every image to all other images, only the most relevant local connections are retained. This helps reduce noise and improves the robustness of spectral clustering.

Each image in the dataset is represented by a feature vector, and the pairwise distances between all images are stored in the distance matrix $D \in R^{N \times N}$.

Although this matrix contains complete similarity information, many of these relationships are not meaningful for clustering. In particular, distant and unrelated images should not influence the clustering process.

To address this issue, a k-nearest neighbor (k-NN) graph is constructed to capture only local geometric structure.

Mathematical Formulation :

Let x_i denote the feature vector corresponding to the i-th image. The set of k-nearest neighbors of x_i is defined as:

$N_k(i) = \{ \ j | x_j$ is among the k nearest neighbors of $x_i \}$

This set contains the indices of the k images that are closest to x_i based on the Euclidean distance computed in Step 2.

Graph Construction:

Using the neighborhood information, a graph $G = (V, E)$ is constructed, where:

- Each vertex $v_i \in V$ represents an image,
- An edge $(i, j) \in E$ exists if image j belongs to the k-nearest neighbor set of image i.

This graph structure is represented using an adjacency matrix $A \in \mathbb{R}^{N \times N}$, defined as:

$$A_{ij} = \begin{cases} 1, & \text{if } j \in \mathcal{N}_k(i) \\ 0, & \text{otherwise} \end{cases}$$

Symmetric k-NN Graph :

Since the k-nearest neighbor relationship is not necessarily symmetric, the adjacency matrix is symmetrized to form an undirected graph:

$$A = \frac{1}{2}(A + A^\top)$$

This ensures that an edge exists between two nodes if either node considers the other as a nearest neighbor.

The k-NN graph is constructed by sorting each row of the distance matrix D, selecting the indices corresponding to the smallest k distances, and updating the adjacency matrix accordingly.

The resulting adjacency matrix is sparse, which significantly reduces computational complexity and memory usage in later steps.

Explanation :

The k-nearest neighbor graph plays a crucial role in the overall clustering pipeline. By retaining only local connections, the graph preserves the intrinsic structure of the data while suppressing the influence of distant and unrelated points.

This local connectivity is essential for accurate estimation of local density in the subsequent density-adaptive affinity computation.

If the graph were fully connected, spectral clustering would be dominated by noise and global density variations.

The k-NN graph prevents this issue and forms a reliable foundation for constructing the affinity matrix and graph Laplacian.

5: Density-Adaptive Affinity Matrix Construction :

This step is to compute a similarity (affinity) matrix that accurately reflects the relationships between images while adapting to variations in local data density. Unlike global similarity measures, the proposed approach assigns a locally scaled similarity to each data point, making the clustering process more robust.

From the Distance matrix computation we observe that dataset is represented as a k-nearest neighbor graph that preserves local neighborhood relationships. However, graph connectivity alone does not quantify how strongly two images are related. To perform spectral clustering, it is necessary to assign a weigh**t** to each edge that reflects the similarity between connected nodes.

Traditional spectral clustering methods use a single global scaling parameter for all data points. This approach performs poorly when the dataset contains regions of varying density, which is common in satellite imagery. To overcome this limitation, a density-adaptive (self-tuning) affinity measure is employed.

Local Scaling Parameter:

For each data point x_i, a local scaling parameter is defined based on its neighborhood:

$$\sigma i = d(xi, xi, k)$$

where $x_{i,k}$ denotes the kth nearest neighbor of x_i , and $d(\cdot, \cdot)$ represents the Euclidean distance.

This parameter provides an estimate of the local data density around x_i:

- A small value of σi indicates a dense region.
- A large value of σi indicates a sparse region.

Affinity Matrix Definition:

Using the local scaling parameters, the affinity between two data points x_i and x_j is computed as:

$$W_{ij} = \exp\left(-\frac{d_{ij}^2}{\sigma_i \sigma_j}\right)$$

where d_{ij} is the Euclidean distance between x_i and x_j

The resulting affinity values are stored in the affinity matrix:

$$W \in \mathbb{R}^{N \times N}$$

The affinity matrix is computed using element-wise operations on the distance matrix and the local scaling vector.

The resulting matrix W is symmetric and sparse, as affinity values are computed only for pairs of points connected in the k-nearest neighbor graph.

Explanation :

The density-adaptive affinity matrix plays a central role in the proposed method. By incorporating local scaling, similarity computation becomes sensitive to neighborhood density rather than relying on a single global parameter. This ensures that points in dense regions are not overly connected and points in sparse regions are not under-connected.

As a result, the affinity matrix provides a faithful representation of the underlying data structure, which is essential for constructing a meaningful graph Laplacian in the next step.

6: Graph Laplacian Construction :

The objective of this step is to normalize the affinity graph constructed in the previous step and prepare it for spectral analysis. The graph Laplacian provides a mathematical representation of the graph structure and plays a central role in spectral clustering.

Degree Matrix Construction:

The degree of a node represents the total connection strength of that node in the graph. The degree matrix is defined as a diagonal matrix:

$$D_{ii} = \sum_{j=1}^{N} W_{ij}$$

$$D = diag(D_{11}, D_{22}, \dots, D_{NN})$$

Each diagonal entry D_{ii} corresponds to the sum of affinities connected to node i.

Normalized Graph Laplacian:

Using the degree matrix, the normalized graph Laplacian is constructed as:

$$L = D^{-1/2} W D^{-1/2}$$

This formulation normalizes the influence of each node by its degree and ensures balanced contribution from both dense and sparse regions of the graph.

The degree matrix is computed by summing each row of the affinity matrix W and placing the resulting values along the diagonal of a square matrix. The inverse square root of the degree matrix is then computed, and matrix multiplication is used to obtain the normalized Laplacian L.

Explanation :

The normalized graph Laplacian removes the dominance of highly connected nodes and ensures that the clustering process is influenced by relative, rather than absolute, connectivity. This normalization is essential when dealing with real-world data, such as satellite images, where local densities vary significantly.

The Laplacian matrix encapsulates the global structure of the graph while preserving local neighborhood relationships. It serves as the input to the eigen decomposition step, where meaningful low-dimensional representations of the data are extracted.

7: Eigen Decomposition :

The objective of this step is to extract meaningful low-dimensional representations of the data by performing eigen decomposition on the normalized graph Laplacian. These representations reveal the intrinsic cluster structure of the data.

The dataset is represented by the normalized graph Laplacian :

$$L \in \mathbb{R}^{N \times N}$$

This matrix encodes both local neighborhood relationships and global graph structure. To identify clusters, it is necessary to analyze the spectral properties of this matrix.

Spectral clustering is based on the principle that the eigenvectors of the graph Laplacian contain information about connected components and cluster separability.

Eigenvalue Problem

The eigen decomposition of the normalized graph Laplacian is defined by the following equation:

$$Lu = \lambda u$$

where:

- λ denotes an eigenvalue,
- uuu denotes the corresponding eigenvector.

Each eigenvector represents a direction in which the graph structure varies smoothly.

Eigen Decomposition Result :

Solving the eigenvalue problem yields a set of eigenvalues and eigenvectors:

$$U = [u_1, u_2, \ldots, u_N]$$

where $U \in \mathbb{R}^{N \times N}$ is the matrix of eigenvectors and each column u_i corresponds to an eigenvalue λ_i

Selection of Dominant Eigenvectors :

To perform clustering into K groups, the eigenvectors corresponding to the largest K eigenvalues are selected:

$$U_K = [u_1, u_2, \ldots, u_K]$$

$$U_K \in R^{N \times K}$$

Each row of U_K provides a new representation of a data point in a reduced K-dimensional space.

Explanation :

Eigen decomposition transforms the original graph-based representation into a spectral space where clusters become more distinguishable. Points belonging to the same cluster tend to have similar values in the selected eigenvectors, while points from different clusters become separable.

This step effectively reduces dimensionality while preserving the most important structural information required for clustering.

8: Clustering :

This step is to assign final cluster labels to the data points using the low-dimensional spectral representations obtained from eigen decomposition. This step produces the final grouping of satellite images into meaningful clusters.

Each data point is represented in a reduced K-dimensional spectral space using the matrix

$$U_K \in \mathbb{R}^{N \times K}$$

Although this representation reveals the cluster structure, explicit cluster labels have not yet been assigned.

A clustering algorithm is therefore required to group these spectral embeddings.

Spectral Embedding Normalization:

Before clustering, each row of the eigenvector matrix is normalized to unit length:

$$y_i = \frac{u_i}{\|u_i\|_2}$$

where:

- u_i is the i-th row of U_K
- y_i is the normalized spectral embedding.

The normalized embeddings are collected into a matrix:

$$Y = [y_1; y_2; \ldots ; y_N] \in \mathbb{R}^{N \times K}$$

Clustering in Spectral Space:

K-Means clustering is applied to the rows of matrix Y to partition the data into K clusters:

$$\mathscr{C} = \{C_1, C_2, \ldots , C_K\}$$

where each cluster $C_k$contains data points with similar spectral representations.

Explanation :

Clustering in the spectral space is significantly more effective than clustering in the original feature space. The spectral embedding separates data points based on graph connectivity rather than raw distances, allowing complex, non-convex cluster structures to be identified.

As a result, images belonging to the same land-cover category tend to be grouped together, even when they are not well separated in the original feature space.