Problem Set 13 Solutions

Computer Vision 2021
University of Bern

1 Segmentation

1. Lets derive the Mean shift iteration.
   Solution
   The key to mean shift is a technique for efficiently finding peaks in high-dimensional data distribution without ever computing the complete function explicitly. Consider the image pixels, which can be thought of as having been drawn from some probability density function. If we could compute this density function. we could find its major peaks (modes) and identify regions of the input space that climb to the same peak as being part of the same region. The first question, then, is how to estimate the density function given a sparse set of samples. One of the simplest approaches is to just smooth the data, e.g., by convolving it with a fixed kernel.

$$f(x) = \frac{1}{N} \sum_i K(x - x_i)$$

   Once we have computed $f(x)$, we can find its local maximum using gradient ascent or some other optimization technique. Let us compute the gradient of $f(x)$ w.r.t x.

$$\nabla f(x) = \frac{1}{N} \sum_{i=1}^{N} \nabla K(x - x_i)$$

   here we make the assumption that $K(x-x_i)$ is a radially symmetric kernel, i.e., its value depends on the distance between x and x_i. For this reason, we can define the following equality:

$$k(\|x - x_i\|^2) = K(x - x_i)$$

and then

$$\nabla f(x) = \frac{2}{N}\sum_{i=1}^{N}(x - x_i)\nabla k(\|x - x_i\|^2)$$

let us define $g(r) = -\nabla k(r)$. This yields to

$$\nabla f(x) = \frac{2}{N}\sum_{i=1}^{N}(x_i - x)g(\|x - x_i\|^2)$$

$$\nabla f(x) = \frac{2}{N}\sum_{i=1}^{N}x_i g(\|x - x_i\|^2) - \sum_{i=1}^{N}x g(\|x - x_i\|^2)$$

$$\nabla f(x) = \frac{2}{N}\sum_{i=1}^{N}g(\|x - x_i\|^2)\left[\frac{\sum_{i=1}^{N}x_i g(\|x - x_i\|^2)}{\sum_{i=1}^{N}g(\|x - x_i\|^2)} - x\right]$$

$$\nabla f(x) = \left[\frac{2}{N}\sum_{i=1}^{N}g(\|x - x_i\|^2)\right]m(x)$$

where the vector $m(x) = \frac{\sum_{i=1}^{N}x_i g(\|x-x_i\|^2)}{\sum_{i=1}^{N}g(\|x-x_i\|^2)} - x$ is called **mean shift**.

from the above equation we can express mean shift as:

$$m(x) = \frac{\nabla f(x)}{\frac{2}{N}\sum_{i=1}^{N}g(\|x - x_i\|^2)}$$

we observe that mean shift is the gradient of the probability density function in the non-parametric form.

2. The Normalized-Cut segmentation algorithm computes the eigenvalues and eigenvectors of the normalized affinity matrix, $D^{-1/2}AD^{1/2}$. Describe the meaning of the two eigenvectors corresponding to the two smallest eigenvalues.
Solution

Let us write normalized cut weights

$$\frac{w(A,B)}{w(A,V)} + \frac{w(A,B)}{w(B,V)}$$

where $w(A,B)$ is the sum of weights of all edges between $A \subset V$ and $B \subset V$. Note that V represents all the nodes in graph. More formally,

$$w(A,B) = \sum_{i \in A, j \in B} w_{i,j} \quad \text{and} \quad w(A,V) = \sum_{i \in A, j \in V} w_{i,j}$$

The optimal cut is obtained as

$$\min_{A,B} \frac{w(A,B)}{w(A,V)} + \frac{w(A,B)}{w(B,V)}$$

obtaining optimal A and B is a NP-hard problem.

Instead we will find a relaxed solution yielding to a generalized eigenvalue problem. For detailed explanations see also Link 1 and Link 2.

Let the matrix W be defined by $W[i,j] = w_{i,j}$; let also $d[i] = \sum_{j \in V} w_{i,j}$ and the matrix $D = diag(d)$.

Let also x be such that $x[i] = +1$ if $i \in A$ and $x[i] = -1$ if $i \in B$.

After detailed derivations our objective becomes

$$\min_{y} = \frac{y^T A y}{y^T D y} \quad \text{s.t} \quad y^T D \mathbf{1} = 0$$

where y is a vector with entries in $\{1, -b\}$, $b = k/(1-k)$, $k = \sum_{i:x[i]>0} d[i] / \sum_i d[i]$ and $A = D - W$.

One can optimize the above objective by solving the generalized eigenvalue problem

$$Ay = \lambda D y$$

Not that the solution automatically satisfies $y^T D \mathbf{1} = 0$ constraint.

By plugging it in the original problem we then look for the smallest eigenvalue λ.

D is a diagonal matrix; it adds the weights of edges connected to the $i-th$ node. The affinity matrix $A = D - W$ is also called the Laplacian matrix.

Example: for a triangle with 3 nodes and 3 edges, the corresponding

$$D = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 2 \end{bmatrix} \quad W = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix} \quad A = D - W = \begin{bmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix}$$

In the above example the wights between nodes are all 1. For arbitrary weights $w_{1,2}, w_{1,3}, w_{2,3}$. In that case A will be

$$A = \begin{bmatrix} w_{1,2} + w_{1,3} & -w_{1,2} & -w_{1,3} \\ -w_{2,1} & w_{2,1} + w_{2,3} & -w_{2,3} \\ -w_{3,1} & -w_{3,2} & w_{3,1} + w_{3,2} \end{bmatrix}$$

We can simplify the above objective by substituting $z = D^{1/2}y$. Thus $z_0 = D^{1/2}\mathbf{1} = 0$ is the eigenvector of L with eigenvalue 0.

Then we have the following equivalent objective

$$\min_z = \frac{z^L z}{z^T z} \quad \text{s.t} \quad zTz_0 = 0$$

L has following properties:

- L is a symmetric positive semidefinite; orthogonal eigenvectors and eigenvalues $\lambda \geq 0$
- we look for the second smallest eigenvalue of L to optimize the objective function
- the eigenvector corresponding to the second smallest eigenvalue is used to partition the image into two sets (its elements should converge to two values each identifying one of the sets)
- we can repeat the process separately on each of the two sets

Conclusion: The smallest eigenvalue is always 0 and its corresponding eigenvector is all 1s, corresponding to all pixels in the image. The second smallest eigenvalue has an eigenvector that divides the pixels into two sets, which we'll use to bi-partition the image into two regions.

3. Define an affinity measure, $A(i, j)$, which measures the similarity of pixels i and j and depends on the similarity of their intensities, $f(i)$ and $f(j)$, and inversely on their distance apart, $d(i, j)$. Briefly explain your definition.
Solution

$$A(i,j) = e^{-\frac{\|f(i)-f(j)\|}{s_1^2}} \cdot \begin{cases} e^{-\frac{d(i,j)}{s_2^2}} & d(i,j) < T \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

This definition combines the two measures scaling each by their standard deviation in order to make them comparable. Also, only pixels that are within distance T of each other are combined so that the affinity between distant pixels in the image is always 0.

4. To minimise the computational burden of NCut, which requires computing a very large affinity matrix, suppose we first divide an input image into small sub-images, say each of size 50×50. Using these sub-images,

describe the main steps of a procedure to use NCut in a two-stage fashion (first with the sub-images, and second with the results of the first stage) to segment an image.
Solution

Step 1: For each 50×50 sub-image, use the affinity function defined in the previous question and apply NCut to compute its bi-partition.

Step 2: Compute the connected components of the binary image constructed by merging the results from Step 1.

Step 3: Define a new graph in which there is a node for each connected component form Step 2, and an arc for each pair of adjacent connected components. Define a new affinity function between pairs of regions based on their size, length of common boundary, average brightness, etc. If two regions are not adjacent, make their affinity 0. Otherwise, make it depend on the similarity of the mean brightness values for the two regions, and their sizes, for example. Apply NCut to this graph, which will compute a bi-partition of the original image. This two-stage approach will have benefits of both saving space and also reducing the tendency of NCut to over-segment images.

5. Consider an image that is empty except for two sets of points. One is a set of points distributed roughly uniformly on a circle of radius r centered at point C1, which is near the center of the image. The other set of points is distributed on a circle of radius $2r$, which is centred at point $C2$, which is located inside the other circle. Assume the points in each set are distributed densely enough so that the distances between points on the same circle are smaller than the distances between points on different circles.

   (a) Describe what segmentation the k-means clustering algorithm would produce for this example, and briefly explain why.
   Solution
   Assuming $k = 2$ is given and the initial cluster centers are $C1$ and $C2$, the k-means algorithm will not find the correct two circular clusters. This is because it attempts to reassign points based on their distance to the class centers. The class separation boundary in this example will be the perpendicular bisector between the estimated calls centers, which is initially $C1$ and $C2$. hence all points on one side of this line will be assigned to $C1$ and all points on the other side of this line will be assigned to $C2$. At the following iterations the updated class centers will move further aaway from the center of the image, resulting in a final partitioning that corresponds to hal-circles, i.e., half points from each of the two circles will be grouped together.

   (b) Describe what segmentation the normalized-cut algorithm would produce for this example, and briefly explain why.

Solution

Using an affinity measure that is based on the distance between pairs of points, and because the distance between several points within a given circle is less than the distance between points in different circles, the normalized-cut algorithm will compute a bi-partition that correctly segments this image into the two sets of circular points.