Graph theoretic clustering

• Represent distance between tokens using a weighted graph.
  – affinity matrix

• Cut up this graph to get subgraphs with strong interior links (and weak links between the subgraphs).
Graph for 9 tokens

Image representation of weight matrix
Measuring Affinity

Intensity

\[ aff(x, y) = \exp \left( \frac{1}{2 \sigma_i^2} \left( \| I(x) \oplus I(y) \|^2 \right) \right) \]

Distance

\[ aff(x, y) = \exp \left( \frac{1}{2 \sigma_d^2} \left( \| x \oplus y \|^2 \right) \right) \]

Texture

\[ aff(x, y) = \exp \left( \frac{1}{2 \sigma_t^2} \left( \| c(x) \oplus c(y) \|^2 \right) \right) \]
Scale affects affinity
Eigenvectors and cuts

- Simplest idea: we want a vector \( \mathbf{a} \) giving the association between each element and a cluster.
- We want elements within this cluster to, on the whole, have strong affinity with one another.
- We could maximize \( \mathbf{a}^T \mathbf{A} \mathbf{a} \).
- But need the constraint \( \mathbf{a}^T \mathbf{a} = 1 \).

- This is an eigenvalue problem - choose the eigenvector of \( \mathbf{A} \) with largest eigenvalue.
- This gives the cluster with greatest internal affinity.
Example eigenvector

points

matrix

best eigenvector
More than two segments

- Two options
  - Recursively split each side to get a tree, continuing till the eigenvalues are too small
  - Use the other eigenvectors
Normalized cuts

- Current criterion evaluates within cluster similarity, but not across cluster difference
- Instead, we’d like to maximize the within cluster similarity compared to the across cluster difference
- Write graph as $V$, one cluster as $A$ and the other as $B$

Maximize

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

- i.e. construct $A$, $B$ such that their within cluster similarity is high compared to their association with the rest of the graph
Normalized cuts

- Write a vector $y$ whose elements are 1 if item is in $A$, -b if it’s in $B$
- Write the matrix of the graph as $W$, and the matrix which has the row sums of $W$ on its diagonal as $D$, 1 is the vector with all ones.
- Criterion becomes $\min_y \frac{y^T (D \Box W) y}{y^T Dy}$
- And we have a constraint $y^T D1 = 0$

- This is hard to do, because $y$’s values are quantized
Normalized cuts

• Instead, solve the generalized eigenvalue problem

\[ \max_y \left( y^T (D \Box W)y \right) \text{ subject to } (y^T Dy = 1) \]

• which gives

\[ (D \Box W)y = \Box Dy \]

• Now look for a quantization threshold that maximizes the criterion --- i.e.
all components of \( y \) above that threshold go to one, all below go to \(-b\)
Figure from “Image and video segmentation: the normalised cut framework”, by Shi and Malik, copyright IEEE, 1998
Fitting

• Choose a parametric object/some objects to represent a set of tokens

• Most interesting case is when criterion is not local
  – can’t tell whether a set of points lies on a line by looking only at each point and the next.

• Three main questions:
  – what object represents this set of tokens best?
  – which of several objects gets which token?
  – how many objects are there?

(you could read line for object here, or circle, or ellipse or...)
Fitting and the Hough Transform

- Purports to answer all three questions
  - in practice, answer isn’t usually all that much help
- We do for lines only
- A line is the set of points \((x, y)\) such that \((\sin \theta)x + (\cos \theta)y + d = 0\)

- Different choices of \(\theta, d > 0\) give different lines
- For any \((x, y)\) there is a one parameter family of lines through this point, given by \((\sin \theta)x + (\cos \theta)y + d = 0\)

- Each point gets to vote for each line in the family; if there is a line that has lots of votes, that should be the line passing through the points
Mechanics of the Hough transform

• Construct an array representing \( q, d \)
• For each point, render the curve \((q, d)\) into this array, adding one at each cell
• Difficulties
  – how big should the cells be? (too big, and we cannot distinguish between quite different lines; too small, and noise causes lines to be missed)
• How many lines?
  – count the peaks in the Hough array
• Who belongs to which line?
  – tag the votes
• Hardly ever satisfactory in practice, because problems with noise and cell size defeat it