Lecture 6: Clustering and Segmentation – Part 2

Professor Fei-Fei Li
Stanford Vision Lab
Recap: Gestalt Theory

• Gestalt: whole or group
  – Whole is greater than sum of its parts
  – Relationships among parts can yield new properties/features

• Psychologists identified series of factors that predispose set of elements to be grouped (by human visual system)

“I stand at the window and see a house, trees, sky. Theoretically I might say there were 327 brightnesses and nuances of colour. Do I have “327”? No. I have sky, house, and trees.”

Max Wertheimer
(1880-1943)

Untersuchungen zur Lehre von der Gestalt,
Psychologische Forschung, Vol. 4, pp. 301-350, 1923
http://psy.ed.asu.edu/~classics/Wertheimer/Forms/forms.htm
Recap: Gestalt Factors

- Not grouped
- Proximity
- Similarity
- Similarity
- Common Fate
- Common Region

- These factors make intuitive sense, but are very difficult to translate into algorithms.
Recap: Image Segmentation

• Goal: identify groups of pixels that go together
Recap: K-Means Clustering

• Basic idea: randomly initialize the $k$ cluster centers, and iterate between the two steps we just saw.

  1. Randomly initialize the cluster centers, $c_1, \ldots, c_k$
  2. Given cluster centers, determine points in each cluster
     • For each point $p$, find the closest $c_i$. Put $p$ into cluster $i$
  3. Given points in each cluster, solve for $c_i$
     • Set $c_i$ to be the mean of points in cluster $i$
  4. If $c_i$ have changed, repeat Step 2

• Properties
  – Will always converge to some solution
  – Can be a “local minimum”
    • Does not always find the global minimum of objective function:
      \[
      \sum_{\text{clusters } i} \sum_{\text{points } p \text{ in cluster } i} ||p - c_i||^2
      \]
Recap: Expectation Maximization (EM)

• Goal
  – Find blob parameters $\theta$ that maximize the likelihood function:
    \[
P(data|\theta) = \prod_x P(x|\theta)
    \]

• Approach:
  1. E-step: given current guess of blobs, compute ownership of each point
  2. M-step: given ownership probabilities, update blobs to maximize likelihood function
  3. Repeat until convergence
What we will learn today?

• Model free clustering
  – Mean-shift
• Graph theoretic segmentation
  – Normalized Cuts
  – Using texture features
• Segmentation as Energy Minimization
  – Markov Random Fields
    – Graph cuts for image segmentation (supp. materials)
    – s-t mincut algorithm (supp. materials)
    – Extension to non-binary case (supp. materials)
  – Applications

(Midterm materials)
What we will learn today?

• “Model free” clustering
  – Mean-shift

• Graph theoretic segmentation
  – Normalized Cuts
  – Using texture features

• Segmentation as Energy Minimization
  – Markov Random Fields
  – Graph cuts for image segmentation (supp. materials)
  – s-t mincut algorithm (supp. materials)
  – Extension to non-binary case (supp. materials)
  – Applications
Mean-Shift Segmentation

- An advanced and versatile technique for clustering-based segmentation

http://www.caip.rutgers.edu/~comanici/MSPAMI/msPamiResults.html

Mean-Shift Algorithm

• Iterative Mode Search
  1. Initialize random seed, and window W
  2. Calculate center of gravity (the “mean”) of W: \[ \sum_{x \in W} x H(x) \]
  3. Shift the search window to the mean
  4. Repeat Step 2 until convergence
Mean-Shift

Region of interest

Center of mass

Mean Shift vector

Slide by Y. Ukrainitz & B. Sarel
Mean-Shift

Region of interest
Center of mass

Mean Shift vector

Slide by Y. Ukrainitz & B. Sarel

Fei-Fei Li

Lecture 5 - 13 12-Oct-11
Mean-Shift
Mean-Shift

Region of interest

Center of mass

Mean Shift vector

Slide by Y. Ukrainitz & B. Sarel

Fei-Fei Li

Lecture 5 - 15  12-Oct-11
Mean-Shift

Region of interest
Center of mass
Real Modality Analysis

Tessellate the space with windows

Run the procedure in parallel
The blue data points were traversed by the windows towards the mode.
Mean-Shift Clustering

- Cluster: all data points in the attraction basin of a mode
- Attraction basin: the region for which all trajectories lead to the same mode
Mean-Shift Clustering/Segmentation

• Find features (color, gradients, texture, etc)
• Initialize windows at individual pixel locations
• Perform mean shift for each window until convergence
• Merge windows that end up near the same “peak” or mode
Mean-Shift Segmentation Results

http://www.caip.rutgers.edu/~comanici/MSPAMI/msPamiResults.html
More Results
More Results
Problem: Computational Complexity

- Need to shift many windows...
- Many computations will be redundant.
1. Assign all points within radius $r$ of end point to the mode.
2. Assign all points within radius \( r/c \) of the search path to the mode -> reduce the number of data points to search.
Summary Mean-Shift

- **Pros**
  - General, application-independent tool
  - Model-free, does not assume any prior shape (spherical, elliptical, etc.) on data clusters
  - Just a single parameter (window size h)
    - h has a physical meaning (unlike k-means)
  - Finds variable number of modes
  - Robust to outliers

- **Cons**
  - Output depends on window size
  - Window size (bandwidth) selection is not trivial
  - Computationally (relatively) expensive (~2s/image)
  - Does not scale well with dimension of feature space
Back to the Image Segmentation Problem...

• Goal: identify groups of pixels that go together

• Up to now, we have focused on ways to group pixels into image segments based on their appearance...
  – Segmentation as clustering.
• We also want to enforce region constraints.
  – Spatial consistency
  – Smooth borders
What we will learn today?

• Model free clustering
  – Mean-shift

• Graph theoretic segmentation
  – Normalized Cuts
  – Using texture features

• Segmentation as Energy Minimization
  – Markov Random Fields
  – Graph cuts for image segmentation (supp. materials)
  – s-t mincut algorithm (supp. materials)
  – Extension to non-binary case (supp. materials)
  – Applications
Images as Graphs

- **Fully-connected graph**
  - Node (vertex) for every pixel
  - Link between *every* pair of pixels, \((p,q)\)
  - Affinity weight \(w_{pq}\) for each link (edge)
    - \(w_{pq}\) measures similarity
    - Similarity is *inversely proportional* to difference
      (in color and position...)

Slide credit: Steve Seitz
Segmentation by Graph Cuts

• Break Graph into Segments
  – Delete links that cross between segments
  – Easiest to break links that have low similarity (low weight)
    • Similar pixels should be in the same segments
    • Dissimilar pixels should be in different segments

Slide credit: Steve Seitz
Measuring Affinity

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

- **Intensity**
  \[ \text{aff} \left( x, y \right) = \exp \left\{ -\frac{1}{2\sigma_d^2} \left\| I(x) - I(y) \right\|^2 \right\} \]

- **Color**
  \[ \text{aff} \left( x, y \right) = \exp \left\{ -\frac{1}{2\sigma_d^2} \left\| \text{dist} \left( c(x), c(y) \right) \right\|^2 \right\} \]
  (some suitable color space distance)

- **Texture**
  \[ \text{aff} \left( x, y \right) = \exp \left\{ -\frac{1}{2\sigma_d^2} \left\| f(x) - f(y) \right\|^2 \right\} \]
  (vectors of filter outputs)

Source: Forsyth & Ponce
Scale Affects Affinity

- Small $\sigma$: group only nearby points
- Large $\sigma$: group far-away points

Slide credit: Svetlana Lazebnik
Graph Cut: using Eigenvalues

- Extract a single good cluster
  - Where elements have high affinity values with each other

\[
\text{objective func.} = w_n^T A w_n
\]

\[
\|w_n\| = 1
\]

\[
A w_n = \lambda w_n
\]

\[
\{\text{association of element } i \text{ with cluster } n\} \times
\{\text{affinity between } i \text{ and } j\} \times
\{\text{association of element } j \text{ with cluster } n\}
\]

Fei-Fei Li
Graph Cut: using Eigenvalues

- Extract a single good cluster

Eigenvector associated w/ the largest eigenvalue
Graph Cut: using Eigenvalues

- Extract a single good cluster
- Extract weights for a set of clusters
Graph Cut: using Eigenvalues (effect of the scaling factor)
Algorithm 14.6: Clustering by Graph Eigenvectors

Construct an affinity matrix $A$
Compute the eigenvalues and eigenvectors of the affinity matrix
Until there are sufficient clusters
  Take the eigenvector corresponding to the
  largest unprocessed eigenvalue; zero all components corresponding
  to elements that have already been clustered, and threshold the
  remaining components to determine which element
  belongs to this cluster, choosing a threshold by
  clustering the components, or
  using a threshold fixed in advance.
If all elements have been accounted for, there are
sufficient clusters
end
Graph Cut

- Set of edges whose removal makes a graph disconnected
- Cost of a cut
  - Sum of weights of cut edges: $\text{cut}(A, B) = \sum_{p \in A, q \in B} w_{p,q}$
- A graph cut gives us a segmentation
  - What is a “good” graph cut and how do we find one?
Here, the cut is nicely defined by the block-diagonal structure of the affinity matrix.

⇒ How can this be generalized?
Minimum Cut

• We can do segmentation by finding the *minimum cut* in a graph
  – a *minimum cut* of a graph is a cut whose cutset has the smallest number of elements (unweighted case) or smallest sum of weights possible.
  – Efficient algorithms exist for doing this

• Drawback:
  – Weight of cut proportional to number of edges in the cut
  – Minimum cut tends to cut off very small, isolated components

Cuts with lesser weight than the ideal cut

Ideal Cut
Normalized Cut (NCut)

• A minimum cut penalizes large segments
• This can be fixed by normalizing for size of segments
• The normalized cut cost is:

$$ Ncut(A, B) = \frac{cut(A, B)}{assoc(A,V)} + \frac{cut(A, B)}{assoc(B,V)} $$

$$ assoc(A,V) = \text{sum of weights of all edges in } V \text{ that touch } A $$

$$ = cut(A, B) \left[ \frac{1}{\sum_{p \in A} w_{p,q}} + \frac{1}{\sum_{q \in B} w_{p,q}} \right] $$

• The exact solution is NP-hard but an approximation can be computed by solving a \textit{generalized eigenvalue} problem.

Interpretation as a Dynamical System

- Treat the links as springs and shake the system
  - Elasticity proportional to cost
  - Vibration “modes” correspond to segments
    - Can compute these by solving a generalized eigenvector problem
NCuts as a Generalized Eigenvector Problem

• Definitions

\( W : \text{the affinity matrix, } W(i, j) = w_{i,j}; \)

\( D : \text{the diag. matrix, } D(i, i) = \sum_j W(i, j); \)

\( x : \text{a vector in } \{1, -1\}^N, x(i) = 1 \Leftrightarrow i \in A. \)

• Rewriting Normalized Cut in matrix form:

\[
NCut(A,B) = \frac{cut(A,B)}{assoc(A,V)} + \frac{cut(A,B)}{assoc(B,V)}
\]

\[
= \frac{(1 + x)^T (D - W)(1 + x)}{k1^T D1} + \frac{(1 - x)^T (D - W)(1 - x)}{(1 - k)1^T D1}; \quad k = \frac{\sum_{x_i > 0} D(i,i)}{\sum_i D(i,i)}
\]

\[
= \ldots
\]
We see again this is an unbiased measure, which reflects how tightly on average nodes within the group are connected to each other.

A further important property of this definition of association and disassociation of a partition is that they are naturally related:

\[
N_{\text{int}}(A, B) = \frac{\text{aa}(A, B) + \text{aa}(B, A)}{\text{aa}(A, V) + \text{aa}(B, V) - \text{aa}(A, B)}
\]

\[
= \frac{\text{aa}(A, B)}{\text{aa}(A, V) - \text{aa}(A, B)} + \frac{\text{aa}(B, A)}{\text{aa}(B, V) - \text{aa}(B, A)}
\]

\[
= 2 - \frac{\text{aa}(A, B) + \text{aa}(B, A)}{\text{aa}(A, V) + \text{aa}(B, V)}
\]

Hence the two partition criteria that we seek in our grouping algorithms, minimizing the disassociation between the groups and maximizing the association within the group, are in fact identical, and can be satisfied simultaneously. In our algorithms, we will use this normalized cut as the partition criterion.

Having defined the graph partition criteria that we want to optimize, we will show how such an optimal partition can be computed efficiently.

2.1 Computing the optimal partition

Given a partition of nodes of a graph, \( V \), into two sets \( A \) and \( B \), let \( \delta(A) \) be an \( N = |V| \) dimensional indicator vector, \( \delta(A)_i = 1 \) if node \( i \) is in \( A \), and \( 0 \) otherwise. Let \( d(i) = \sum_j \delta(i, j) \), be the total connection from node \( i \) to all other nodes. With the definitions as and \( d \) we can rewrite \( N_{\text{int}}(A, B) \) as:

\[
N_{\text{int}}(A, B) = \frac{\text{aa}(A, B) + \text{aa}(B, A)}{\text{aa}(A, V) + \text{aa}(B, V) - \text{aa}(A, B)}
\]

\[
= \frac{\sum_{(i,j) \in \delta(A) \times \delta(B) \cap \delta(A^c) \times \delta(B^c)} \delta(i, j)}{\sum_{i,j \in \delta(A) \times \delta(B) \cap \delta(A^c) \times \delta(B^c)} \delta(i, j)}
\]

Let \( D \) be an \( N \times N \) diagonal matrix with \( d \) on its diagonal, \( W \) be an \( N \times N \) symmetrical matrix with \( W(i,j) = \delta(i,j) \), and \( \mathbf{1} \) be an \( N \times 1 \) vector of all ones. Using the facts \( \frac{\partial}{\partial \theta} \mathbf{1} = \mathbf{1} \) and \( \frac{\partial}{\partial \theta} \delta = \mathbf{1} \), we can rewrite \( \frac{\partial}{\partial \theta} [N_{\text{int}}(A, B)] \):

\[
\frac{\partial}{\partial \theta} [N_{\text{int}}(A, B)] = \frac{2 \delta(A) - \delta(A^c) \cdot \delta(B)}{\sum_{i,j \in \delta(A) \times \delta(B) \cap \delta(A^c) \times \delta(B^c)} \delta(i, j)}
\]

Let \( d(A) = \sum_{i,j} D_{i,j} \delta(A)_i \delta(A)_j \), \( d(A) = \sum_{i,j} D_{i,j} \delta(A^c)_i \delta(A^c)_j \), \( M = \sum_{i,j} D_{i,j} \), and \( \tau = 1 - d(A) \), we can then further expand the above equation as:

\[
\frac{\partial}{\partial \theta} [N_{\text{int}}(A, B)] = \frac{2 \delta(A) - \delta(A^c) \cdot \delta(B)}{\sum_{i,j \in \delta(A) \times \delta(B) \cap \delta(A^c) \times \delta(B^c)} \delta(i, j)}
\]
NCuts as a Generalized Eigenvalue Problem

• After simplification, we get

\[ NCut(A, B) = \frac{y^T (D-W)y}{y^T Dy}, \quad \text{with } y_i \in \{1, -b\}, \quad y^T D1 = 0. \]

• This is a Rayleigh Quotient
  – Solution given by the “generalized” eigenvalue problem
    \[ (D - W)y = \lambda Dy \]
  – Solved by converting to standard eigenvalue problem
    \[ D^{-\frac{1}{2}} (D - W)D^{-\frac{1}{2}} z = \lambda z, \quad \text{where } z = D^{-\frac{1}{2}} y \]

• Subtleties
  – Optimal solution is second smallest eigenvector
  – Gives continuous result—must convert into discrete values of \( y \)

This is hard, as \( y \) is discrete!

Relaxation: continuous \( y \).
NCuts Example

NCuts segments

Smallest eigenvectors

Image source: Shi & Malik

Fei-Fei Li

Lecture 6 - 48
12-Oct-11
Discretization

- **Problem:** eigenvectors take on continuous values
  - How to choose the splitting point to binarize the image?

- **Possible procedures**
  a) Pick a constant value (0, or 0.5).
  b) Pick the median value as splitting point.
  c) Look for the splitting point that has the minimum *NCut* value:
     1. Choose \( n \) possible splitting points.
     2. Compute *NCut* value.
     3. Pick minimum.
NCuts: Overall Procedure

1. Construct a weighted graph $G=(V,E)$ from an image.
2. Connect each pair of pixels, and assign graph edge weights,
   $$W(i,j) = \text{Prob. that } i \text{ and } j \text{ belong to the same region.}$$
3. Solve $(D-W)y = \lambda Dy$ for the smallest few eigenvectors. This yields a continuous solution.
4. Threshold eigenvectors to get a discrete cut
   – This is where the approximation is made (we’re not solving NP).
5. Recursively subdivide if NCut value is below a pre-specified value.

NCuts Matlab code available at http://www.cis.upenn.edu/~jshi/software/
Color Image Segmentation with NCuts

Image Source: Shi & Malik
Using Texture Features for Segmentation

- Texture descriptor is vector of filter bank outputs

Using Texture Features for Segmentation

- Texture descriptor is a vector of filter bank outputs.
- **Textons** are found by clustering.
Using Texture Features for Segmentation

- Texture descriptor is vector of filter bank outputs.
- *Textons* are found by clustering.
- Affinities are given by similarities of texton histograms over windows given by the “local scale” of the texture.
Results with Color & Texture
Summary: Normalized Cuts

• **Pros:**
  – Generic framework, flexible to choice of function that computes weights ("affinities") between nodes
  – Does not require any model of the data distribution

• **Cons:**
  – Time and memory complexity can be high
    • Dense, highly connected graphs ⇒ many affinity computations
    • Solving eigenvalue problem for each cut
  – Preference for balanced partitions
    • If a region is uniform, NCuts will find the modes of vibration of the image dimensions
What we will learn today?

• Model free clustering
  – Mean-shift

• Graph theoretic segmentation
  – Normalized Cuts
  – Using texture features

• Segmentation as Energy Minimization
  – Markov Random Fields
  – Graph cuts for image segmentation (supp. materials)
  – s-t mincut algorithm (supp. materials)
  – Extension to non-binary case (supp. materials)
  – Applications
Markov Random Fields

- Allow rich probabilistic models for images
- But built in a local, modular way
  - Learn local effects, get global effects out

Observed evidence

Hidden “true states”

Neighborhood relations
MRF Nodes as Pixels

Original image

Degraded image

Reconstruction from MRF modeling pixel neighborhood statistics
MRF Nodes as Patches

\[ \Phi(x_i, y_i) \]

\[ \Psi(x_i, x_j) \]

Image patches

Scene patches
Network Joint Probability

\[ P(x, y) = \prod_i \Phi(x_i, y_i) \prod_{i,j} \Psi(x_i, x_j) \]
Energy Formulation

- Joint probability
  \[ P(x, y) = \prod_i \Phi(x_i, y_i) \prod_{i,j} \Psi(x_i, x_j) \]

- Taking the log \( p(.) \) turns this into an Energy optimization problem

  \[ \log P(x, y) = \sum_i \log \Phi(x_i, y_i) + \sum_{i,j} \log \Psi(x_i, x_j) \]

  \[ E(x, y) = \sum_i \varphi(x_i, y_i) + \sum_{i,j} \psi(x_i, x_j) \]

- This is similar to free-energy problems in statistical mechanics (spin glass theory). We therefore draw the analogy and call \( E \) an energy function.

- \( \varphi \) and \( \psi \) are called potentials.
Energy Formulation

- Energy function
  \[ E(x, y) = \sum_{i} \varphi(x_i, y_i) + \sum_{i,j} \psi(x_i, x_j) \]

  - Single-node potentials \( \varphi \)
    - Encode local information about the given pixel/patch
    - How likely is a pixel/patch to belong to a certain class (e.g. foreground/background)?

  - Pairwise potentials \( \psi \)
    - Encode neighborhood information
    - How different is a pixel/patch’s label from that of its neighbor? (e.g. based on intensity/color/texture difference, edges)
Energy Minimization

• Goal:
  – Infer the optimal labeling of the MRF.

• Many inference algorithms are available, e.g.
  – Gibbs sampling, simulated annealing
  – Iterated conditional modes (ICM)
  – Variational methods
  – Belief propagation

• Recently, Graph Cuts have become a popular tool
  – Only suitable for a certain class of energy functions
  – But the solution can be obtained very fast for typical vision problems (~1MPixel/sec).
What we will learn today?

• Graph theoretic segmentation
  – Normalized Cuts
  – Using texture features
  – Extension: Multi-level segmentation

• Segmentation as Energy Minimization
  – Markov Random Fields
  – Graph cuts for image segmentation
  – s-t mincut algorithm
  – Extension to non-binary case
  – Applications
GrabCut: live demo

- Included in MS Office 2010 (let’s try it)
GrabCut: live demo

* Included in MS Office 2010 (let’s try it)
GrabCut: live demo

• Included in MS Office 2010 (let’s try it)
GraphCut Image Synthesis Results
Application: Texture Synthesis in the Media

- Currently, still done manually...

Slide credit: Kristen Grauman
Improving Efficiency of Segmentation

• Problem: Images contain many pixels
  – Even with efficient graph cuts, an MRF formulation has too many nodes for interactive results.

• Efficiency trick: Superpixels
  – Group together similar-looking pixels for efficiency of further processing.
  – Cheap, local oversegmentation
  – Important to ensure that superpixels do not cross boundaries

• Several different approaches possible
  – Superpixel code available here

Image source: Greg Mori
Superpixels for Pre-Segmentation

Graph structure

<table>
<thead>
<tr>
<th>Image</th>
<th>Dimension</th>
<th>Nodes Ratio</th>
<th>Edges Ratio</th>
<th>Lag with Pre-segmentation</th>
<th>Lag without Pre-segmentation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boy</td>
<td>(408, 600)</td>
<td>10.7</td>
<td>16.8</td>
<td>0.12s</td>
<td>0.57s</td>
</tr>
<tr>
<td>Ballet</td>
<td>(440, 800)</td>
<td>11.4</td>
<td>18.3</td>
<td>0.21s</td>
<td>1.39s</td>
</tr>
<tr>
<td>Twins</td>
<td>(1024, 768)</td>
<td>20.7</td>
<td>32.5</td>
<td>0.25s</td>
<td>1.82s</td>
</tr>
<tr>
<td>Girl</td>
<td>(768, 1147)</td>
<td>23.8</td>
<td>37.6</td>
<td>0.22s</td>
<td>2.49s</td>
</tr>
<tr>
<td>Grandpa</td>
<td>(1147, 768)</td>
<td>19.3</td>
<td>30.5</td>
<td>0.22s</td>
<td>3.56s</td>
</tr>
</tbody>
</table>

Speedup
Summary: Graph Cuts Segmentation

• **Pros**
  – Powerful technique, based on probabilistic model (MRF).
  – Applicable for a wide range of problems.
  – Very efficient algorithms available for vision problems.
  – Becoming a de-facto standard for many segmentation tasks.

• **Cons/Iissues**
  – Graph cuts can only solve a limited class of models
    • Submodular energy functions
    • Can capture only part of the expressiveness of MRFs
  – Only approximate algorithms available for multi-label case
What we have learned today

• Model free clustering
  – Mean-shift

• Graph theoretic segmentation
  – Normalized Cuts
  – Using texture features

• Segmentation as Energy Minimization
  – Markov Random Fields
    – Graph cuts for image segmentation (supp. materials)
    – s-t mincut algorithm (supp. materials)
    – Extension to non-binary case (supp. materials)
  – Applications

(Midterm materials)
Supplementary materials
Supplementary materials

• Segmentation as Energy Minimization
  – Markov Random Fields
  – Graph cuts for image segmentation (supp. materials)
  – s-t mincut algorithm (supp. materials)
  – Extension to non-binary case (supp. materials)
  – Applications
Graph Cuts for Optimal Boundary Detection

• Idea: convert MRF into source-sink graph

Minimum cost cut can be computed in polynomial time

(max-flow/min-cut algorithms)

Slide credit: Yuri Boykov
Simple Example of Energy

\[ E(L) = \sum_p D_p(L_p) + \sum_{pq \in N} w_{pq} \cdot \delta(L_p \neq L_q) \]

Regional term \( t \)-links

Boundary term \( n \)-links

\[ w_{pq} = \exp\left\{-\frac{\Delta I_{pq}}{2\sigma^2}\right\} \]

\( L_p \in \{s, t\} \) (binary object segmentation)
Adding Regional Properties

Regional bias example

Suppose $I^s$ and $I^t$ are given “expected” intensities of object and background

\[ D_p(s) \propto \exp \left( - \frac{\| I_p - I^s \|^2}{2\sigma^2} \right) \]
\[ D_p(t) \propto \exp \left( - \frac{\| I_p - I^t \|^2}{2\sigma^2} \right) \]

NOTE: hard constrains are not required, in general.

Slide credit: Yuri Boykov
Adding Regional Properties

“expected” intensities of **object** and **background** $I^s$ and $I^t$ can be re-estimated

\[
D_p(s) \propto \exp\left(-\frac{\|I_p - I^s\|^2}{2\sigma^2}\right)
\]
\[
D_p(t) \propto \exp\left(-\frac{\|I_p - I^t\|^2}{2\sigma^2}\right)
\]

EM-style optimization

**Slide credit:** Yuri Boykov
Adding Regional Properties

• More generally, regional bias can be based on any intensity models of object and background

\[ D_p(L_p) = -\log \Pr(I_p | L_p) \]

given object and background intensity histograms

Slide credit: Yuri Boykov
How to Set the Potentials? Some Examples

- Color potentials
  - e.g. modeled with a Mixture of Gaussians
    \[
    \pi(x_i, y_i; \theta_\pi) = \log \sum_k \theta_\pi(x_i, k) P(k | x_i) N(y_i; \bar{y}_k, \Sigma_k)
    \]

- Edge potentials
  - e.g. a “contrast sensitive Potts model”
    \[
    \phi(x_i, x_j, g_{ij}(y); \theta_\phi) = -\theta_\phi^T g_{ij}(y) \delta(x_i \neq x_j)
    \]
    where
    \[
    g_{ij}(y) = e^{-\beta \|y_i - y_j\|^2} \quad \beta = 2 \cdot \text{avg} \left( \|y_i - y_j\|^2 \right)
    \]

- Parameters $\theta_\pi, \theta_\phi$ need to be learned, too!

[Shotton & Winn, ECCV’06]
Other Applications: Texture Synthesis

Graph-cut textures
(Kwatra, Schodl, Essa, Bobick 2003)

Similar to “image-quilting” (Efros & Freeman, 2001)
Basic Idea

- Input texture
- Random placement of blocks
- Neighboring blocks constrained by overlap
- Minimal error boundary cut

Slide from Alyosha Efros
Minimal Error Boundary

Overlapping blocks

Vertical boundary

Overlap error

min. error boundary

Fei-Fei Li

Lecture 6 - 12-Oct-11
GraphCut Texture Synthesis Results

Original fragments

Results (with perspective correction)

Source: Vivek Kwatra
Application: Texture Synthesis in the Media

• Currently, still done manually...
How Do we Know?

Another Example

Segmentation: Caveats

• We’ve looked at bottom-up ways to segment an image into regions, yet finding meaningful segments is intertwined with the recognition problem.

• Often want to avoid making hard decisions too soon

• Difficult to evaluate; when is a segmentation successful?
  – Often depends on the rest of the recognition pipeline.
References and Further Reading

• Background information on Normalized Cuts can be found in Chapter 14 of

• Try the NCuts Matlab code at

• Try the GraphCut implementation at
  [http://www.adastral.ucl.ac.uk/~vladkolm/software.html](http://www.adastral.ucl.ac.uk/~vladkolm/software.html)
Supplementary materials

- Segmentation as Energy Minimization
  - Markov Random Fields
  - Graph cuts for image segmentation
  - s-t mincut algorithm
  - Extension to non-binary case
  - Applications
How Does it Work? The s-t-Mincut Problem

Graph \((V, E, C)\)

Vertices \(V = \{v_1, v_2 \ldots v_n\}\)

Edges \(E = \{(v_1, v_2) \ldots\}\)

Costs \(C = \{c_{(1, 2)} \ldots\}\)
The s-t-Minicut Problem

What is an s-t-cut?
An s-t-cut (S,T) divides the nodes between source and sink.

What is the cost of an s-t-cut?
Sum of cost of all edges going from S to T

5 + 2 + 9 = 16

Slide credit: Pushmeet Kohli
The s-t-Mincut Problem

What is an st-cut?
An st-cut (S,T) divides the nodes between source and sink.

What is the cost of a st-cut?
Sum of cost of all edges going from S to T

What is the st-mincut?
st-cut with the minimum cost

2 + 1 + 4 = 7

Slide credit: Pushmeet Kohli
## History of Maxflow Algorithms

### Augmenting Path and Push-Relabel

<table>
<thead>
<tr>
<th>Year</th>
<th>Discoverer(s)</th>
<th>Bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>1951</td>
<td>Dantzig</td>
<td>$O(n^2 m U)$</td>
</tr>
<tr>
<td>1955</td>
<td>Ford &amp; Fulkerson</td>
<td>$O(m^2 U)$</td>
</tr>
<tr>
<td>1970</td>
<td>Dinitz</td>
<td>$O(n^2 m)$</td>
</tr>
<tr>
<td>1972</td>
<td>Edmonds &amp; Karp</td>
<td>$O(m^2 \log U)$</td>
</tr>
<tr>
<td>1973</td>
<td>Dinitz</td>
<td>$O(n m \log U)$</td>
</tr>
<tr>
<td>1974</td>
<td>Karzanov</td>
<td>$O(n^3)$</td>
</tr>
<tr>
<td>1977</td>
<td>Cherkassky</td>
<td>$O(n^2 m^{1/2})$</td>
</tr>
<tr>
<td>1980</td>
<td>Galil &amp; Naamad</td>
<td>$O(n m \log^2 n)$</td>
</tr>
<tr>
<td>1983</td>
<td>Sleator &amp; Tarjan</td>
<td>$O(n m \log n)$</td>
</tr>
<tr>
<td>1986</td>
<td>Goldberg &amp; Tarjan</td>
<td>$O(n m \log(n^2/m))$</td>
</tr>
<tr>
<td>1987</td>
<td>Ahuja &amp; Orlin</td>
<td>$O(n m + n^2 \log U)$</td>
</tr>
<tr>
<td>1987</td>
<td>Ahuja et al.</td>
<td>$O(n m \log(n \sqrt{\log U}/m))$</td>
</tr>
<tr>
<td>1989</td>
<td>Cheriyan &amp; Hagerup</td>
<td>$E(n m + n^2 \log^2 n)$</td>
</tr>
<tr>
<td>1990</td>
<td>Cheriyan et al.</td>
<td>$O(n^3/\log n)$</td>
</tr>
<tr>
<td>1990</td>
<td>Alon</td>
<td>$O(n m + n^{8/3} \log n)$</td>
</tr>
<tr>
<td>1992</td>
<td>King et al.</td>
<td>$O(n m + n^{2+\epsilon})$</td>
</tr>
<tr>
<td>1993</td>
<td>Phillips &amp; Westbrook</td>
<td>$O(n m (\log_{m/n} n + \log^{2+\epsilon} n))$</td>
</tr>
<tr>
<td>1994</td>
<td>King et al.</td>
<td>$O(n m \log_{m/(n \log n)} n)$</td>
</tr>
<tr>
<td>1997</td>
<td>Goldberg &amp; Rao</td>
<td>$O(m^{3/2} \log (n^2/m) \log U)$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$O(n^{2/3} m \log (n^2/m) \log U)$</td>
</tr>
</tbody>
</table>

$n$: #nodes  
$m$: #edges  
$U$: maximum edge weight

Algorithms assume non-negative edge weights

Slide credit: Andrew Goldberg

Fei-Fei Li  
Lecture 6 -  
96  
12-Oct-11
How to Compute the s-t-Mincut?

Solve the dual maximum flow problem

Compute the maximum flow between Source and Sink

Constraints
Edges: Flow < Capacity
Nodes: Flow in = Flow out

Min-cut/Max-flow Theorem
In every network, the maximum flow equals the cost of the st-mincut
Maxflow in Computer Vision

- Specialized algorithms for vision problems
  - Grid graphs
  - Low connectivity \( (m \sim O(n)) \)
- Dual search tree augmenting path algorithm
  [Boykov and Kolmogorov PAMI 2004]
  - Finds approximate shortest augmenting paths efficiently
  - High worst-case time complexity
  - Empirically outperforms other algorithms on vision problems
  - Efficient code available on the web
    [http://www.adastral.ucl.ac.uk/~vladkolm/software.html](http://www.adastral.ucl.ac.uk/~vladkolm/software.html)
When Can s-t Graph Cuts Be Applied?

\[
E(L) = \sum_p E_p(L_p) + \sum_{pq \in N} E(L_p, L_q)
\]

- s-t graph cuts can only globally minimize binary energies that are submodular. \([\text{Boros & Hummer, 2002, Kolmogorov & Zabih, 2004}]\)

- Non-submodular cases can still be addressed with some optimality guarantees.
  - Current research topic

\[
E(s,s) + E(t,t) \leq E(s,t) + E(t,s)
\]

Submodularity ("convexity")

Slide credit: Bastian Leibe
Dealing with Non-Binary Cases

• For image segmentation, the limitation to binary energies is a nuisance.
  ⇒ Binary segmentation only
• We would like to solve also multi-label problems.
  – NP-hard problem with 3 or more labels
• There exist some approximation algorithms which extend graph cuts to the multi-label case
  – $\alpha$-Expansion
  – $\alpha\beta$-Swap
• They are no longer guaranteed to return the globally optimal result.
  – But $\alpha$-Expansion has a guaranteed approximation quality (2-approx) and converges in a few iterations.
α-Expansion Move

• Basic idea:
  – Break multi-way cut computation into a sequence of binary s-t cuts.
$\alpha$-Expansion Algorithm

1. Start with any initial solution
2. For each label “$\alpha$” in any (e.g. random) order
   1. Compute optimal $\alpha$-expansion move (s-t graph cuts)
   2. Decline the move if there is no energy decrease

• Stop when no expansion move would decrease energy
α-Expansion Moves

- In each α-expansion a given label “α” grabs space from other labels

For each move we choose the expansion that gives the largest decrease in the energy: binary optimization problem
GraphCut Applications: “GrabCut”

• **Interactive Image Segmentation** [Boykov & Jolly, ICCV’01]
  – Rough region cues sufficient
  – Segmentation boundary can be extracted from edges

• **Procedure**
  – User marks foreground and background regions with a brush.
  – This is used to create an initial segmentation which can then be corrected by additional brush strokes.

Slide credit: Matthieu Bray
Iterated Graph Cuts

Result

Color model (Mixture of Gaussians)

Energy after each iteration

Slide credit: Carsten Rother
GrabCut: Data Model

- Obtained from interactive user input
  - User marks foreground and background regions with a brush
  - Alternatively, user can specify a bounding box

Slide credit: Carsten Rother
GrabCut: Coherence Model

- An object is a coherent set of pixels:

\[ \psi(x, y) = \gamma \sum_{(m,n) \in C} \delta[x_n \neq x_m] e^{-\beta \| y_m - y_n \|^2} \]

How to choose \( \gamma \)?

Error (%) over training set:

Slide credit: Carsten Rother
Graph Cuts for Image Segmentation

- **Segmentation by s-t mincut**
  - *Cut*: separating source and sink
  - *Min Cut*: Global minimal energy in polynomial time (1MPixel/sec)
  - s-t MinCut Problem is equal to MaxFlow Problem [Fulkerson 56]

Slide credit: Carsten Rother