Machine Learning - Lecture 18

Inference & Applications

12.07.2016

Bastian Leibe
RWTH Aachen
http://www.vision.rwth-aachen.de
leibe@vision.rwth-aachen.de
Announcements

• Lecture evaluation
  ➢ Please fill out the evaluation forms...
Course Outline

• Fundamentals (2 weeks)
  ➢ Bayes Decision Theory
  ➢ Probability Density Estimation

• Discriminative Approaches (5 weeks)
  ➢ Linear Discriminant Functions
  ➢ Statistical Learning Theory & SVMs
  ➢ Ensemble Methods & Boosting
  ➢ Decision Trees & Randomized Trees

• Generative Models (4 weeks)
  ➢ Bayesian Networks
  ➢ Markov Random Fields
  ➢ Exact Inference
  ➢ Applications

B. Leibe
Topics of This Lecture

• Recap: Exact inference
  - Sum-Product algorithm
  - Max-Sum algorithm
  - Junction Tree algorithm

• Applications of Markov Random Fields
  - Application examples from computer vision
  - Interpretation of clique potentials
  - Unary potentials
  - Pairwise potentials

• Solving MRFs with Graph Cuts
  - Graph cuts for image segmentation
  - s-t mincut algorithm
  - Extension to non-binary case
  - Applications
Recap: Factor Graphs

- Joint probability
  - Can be expressed as product of factors: $p(x) = \frac{1}{Z} \prod_s f_s(x_s)$
  - Factor graphs make this explicit through separate factor nodes.

- Converting a directed polytree
  - Conversion to undirected tree creates loops due to moralization!
  - Conversion to a factor graph again results in a tree!
Recap: Sum-Product Algorithm

- Objectives
  - Efficient, exact inference algorithm for finding marginals.

- Procedure:
  - Pick an arbitrary node as root.
  - Compute and propagate messages from the leaf nodes to the root, storing received messages at every node.
  - Compute and propagate messages from the root to the leaf nodes, storing received messages at every node.
  - Compute the product of received messages at each node for which the marginal is required, and normalize if necessary.

\[ p(x) \propto \prod_{s \in \text{ne}(x)} \mu_{f_s \rightarrow x}(x) \]

- Computational effort
  - Total number of messages = 2 \cdot \text{number of graph edges}.
Recap: Sum-Product Algorithm

- Two kinds of messages
  - Message from factor node to variable nodes:
    - **Sum** of factor contributions
      \[ \mu_{f_s \rightarrow x}(x) \equiv \sum_{X_s} F_s(x, X_s) \]
      \[ = \sum_{X_s} f_s(x_s) \prod_{m \in \text{ne}(f_s) \setminus x} \mu_{x_m \rightarrow f_s}(x_m) \]
  - Message from variable node to factor node:
    - **Product** of incoming messages
      \[ \mu_{x_m \rightarrow f_s}(x_m) \equiv \prod_{l \in \text{ne}(x_m) \setminus f_s} \mu_{f_l \rightarrow x_m}(x_m) \]

⇒ Simple propagation scheme.
Recap: Sum-Product from Leaves to Root

Message definitions:

\[ \mu_{f_s \rightarrow x}(x) = \sum_{X_s} f_s(x_s) \prod_{m \in \text{ne}(f_s) \setminus x} \mu_{x_m \rightarrow f_s}(x_m) \]

\[ \mu_{x_m \rightarrow f_s}(x_m) = \prod_{l \in \text{ne}(x_m) \setminus f_s} \mu_{f_l \rightarrow x_m}(x_m) \]

\[ \mu_{x \rightarrow f}(x) = 1 \]

\[ \mu_{f \rightarrow x}(x) = f(x) \]
Recap: Sum-Product from Root to Leaves

Message definitions:

\[ \mu_{f_s \rightarrow x}(x) \equiv \sum_{X_s} f_s(x_s) \prod_{m \in \text{ne}(f_s) \setminus x} \mu_{x_m \rightarrow f_s}(x_m) \]

\[ \mu_{x_m \rightarrow f_s}(x_m) \equiv \prod_{l \in \text{ne}(x_m) \setminus f_s} \mu_{f_l \rightarrow x_m}(x_m) \]

\[ \mu_{x \rightarrow f}(x) = 1 \quad \mu_{f \rightarrow x}(x) = f(x) \]
Max-Sum Algorithm

- **Objective:** an efficient algorithm for finding
  - Value $x^{\text{max}}$ that maximises $p(x)$;
  - Value of $p(x^{\text{max}})$.

  ⇒ Application of dynamic programming in graphical models.

- **In general, maximum marginals ≠ joint maximum.**
  - Example:

    $\begin{array}{c|cc}
    & x = 0 & x = 1 \\
    \hline
    y = 0 & 0.3 & 0.4 \\
    y = 1 & 0.3 & 0.0 \\
    \end{array}$

    $\arg \max_x p(x, y) = 1 \quad \arg \max_x p(x) = 0$
Max-Sum Algorithm - Key Ideas

• Key idea 1: Distributive Law (again)

\[
\begin{align*}
\max(ab, ac) &= a \max(b, c) \\
\max(a + b, a + c) &= a + \max(b, c)
\end{align*}
\]

⇒ Exchange products/summations and max operations exploiting the tree structure of the factor graph.

• Key idea 2: Max-Product → Max-Sum

➢ We are interested in the maximum value of the joint distribution

\[
p(x_{\text{max}}) = \max_x p(x)
\]

⇒ Maximize the product \( p(x) \).

➢ For numerical reasons, use the logarithm.

\[
\ln \left( \max_x p(x) \right) = \max_x \ln p(x).
\]

⇒ Maximize the sum (of log-probabilities).
Max-Sum Algorithm

- Maximizing over a chain (max-product)

\[ p(x_{\text{max}}) = \max_x p(x) = \max_{x_1} \ldots \max_{x_M} p(x) \]

\[ = \frac{1}{Z} \max_{x_1} \ldots \max_{x_N} [\psi_{1,2}(x_1, x_2) \ldots \psi_{N-1,N}(x_{N-1}, x_N)] \]

\[ = \frac{1}{Z} \max_{x_1} \left[ \max_{x_2} \left[ \psi_{1,2}(x_1, x_2) \left[ \ldots \max_{x_N} \psi_{N-1,N}(x_{N-1}, x_N) \right] \ldots \right] \right] \]

- Exchange max and product operators

- Generalizes to tree-structured factor graph

\[ \max_x p(x) = \max_{x_n} \prod_{f_s \in \text{ne}(x_n)} \max_{X_s} f_s(x_n, X_s) \]

Slide adapted from Chris Bishop

Image source: C. Bishop, 2006
Max-Sum Algorithm

- Initialization (leaf nodes)

\[ \mu_{x \rightarrow f}(x) = 0 \quad \mu_{f \rightarrow x}(x) = \ln f(x) \]

- Recursion

  - Messages

  \[ \mu_{f \rightarrow x}(x) = \max_{x_1, \ldots, x_M} \left[ \ln f(x, x_1, \ldots, x_M) + \sum_{m \in \text{ne}(f_s) \setminus x} \mu_{x_m \rightarrow f}(x_m) \right] \]

  \[ \mu_{x \rightarrow f}(x) = \sum_{l \in \text{ne}(x) \setminus f} \mu_{f_l \rightarrow x}(x) \]

  - For each node, keep a record of which values of the variables gave rise to the maximum state:

\[ \phi(x) = \arg \max_{x_1, \ldots, x_M} \left[ \ln f(x, x_1, \ldots, x_M) + \sum_{m \in \text{ne}(f_s) \setminus x} \mu_{x_m \rightarrow f}(x_m) \right] \]
Max-Sum Algorithm

• Termination (root node)
  - Score of maximal configuration
    \[ p^{\text{max}} = \max_x \left[ \sum_{s \in \text{ne}(x)} \mu_{f_s \rightarrow x}(x) \right] \]
  - Value of root node variable giving rise to that maximum
    \[ x^{\text{max}} = \arg \max_x \left[ \sum_{s \in \text{ne}(x)} \mu_{f_s \rightarrow x}(x) \right] \]
  - Back-track to get the remaining variable values
    \[ x_{n-1}^{\text{max}} = \phi(x_n^{\text{max}}) \]

Slide adapted from Chris Bishop
Visualization of the Back-Tracking Procedure

- Example: Markov chain

$\Rightarrow$ Same idea as in Viterbi algorithm for HMMs...

Slide adapted from Chris Bishop

Image source: C. Bishop, 2006
Topics of This Lecture

- Factor graphs
  - Construction
  - Properties

- Sum-Product Algorithm for computing marginals
  - Key ideas
  - Derivation
  - Example

- Max-Sum Algorithm for finding most probable value
  - Key ideas
  - Derivation
  - Example

- Algorithms for loopy graphs
  - Junction Tree algorithm
  - Loopy Belief Propagation
**Junction Tree Algorithm**

- **Motivation**
  - **Exact** inference on general graphs.
  - Works by turning the initial graph into a **junction tree** with one node per clique and then running a sum-product-like algorithm.
  - **Intractable** on graphs with large cliques.
Loopy Belief Propagation

- Alternative algorithm for loopy graphs
  - Sum-Product on general graphs.
  - Strategy: simply ignore the problem.
  - Initial unit messages passed across all links, after which messages are passed around until convergence
    - Convergence is not guaranteed!
    - Typically break off after fixed number of iterations.
  - Approximate but tractable for large graphs.
  - Sometime works well, sometimes not at all.
Topics of This Lecture

• Recap: Exact inference
  - Sum-Product algorithm
  - Max-Sum algorithm
  - Junction Tree algorithm

• Applications of Markov Random Fields
  - Application examples from computer vision
  - Interpretation of clique potentials
  - Unary potentials
  - Pairwise potentials

• Solving MRFs with Graph Cuts
  - Graph cuts for image segmentation
  - s-t mincut algorithm
  - Extension to non-binary case
  - Applications
Markov Random Fields (MRFs)

• What we’ve learned so far...
  ➢ We know they are undirected graphical models.
  ➢ Their joint probability factorizes into clique potentials,
    \[ p(x) = \frac{1}{Z} \prod_C \psi_C(x_C) \]
    which are conveniently expressed as energy functions.
    \[ \psi_C(x_C) = \exp\{-E(x_C)\} \]
  ➢ We know how to perform inference for them.
    - Sum/Max-Product BP for exact inference in tree-shaped MRFs.
    - Loopy BP for approximate inference in arbitrary MRFs.
    - Junction Tree algorithm for converting arbitrary MRFs into trees.

• But what are they actually good for?
  ➢ And how do we apply them in practice?
Markov Random Fields

- Allow rich probabilistic models.
  - But built in a local, modular way.
  - Learn local effects, get global effects out.
- Very powerful when applied to regular structures.
  - Such as images...

Observed evidence

Hidden “true states”

Neighborhood relations

Slide adapted from William Freeman
Applications of MRFs

• Movie “No Way Out” (1987)
Applications of MRFs

- Many applications for low-level vision tasks
  - Image denoising

Results by [Roth & Black, CVPR'05]
Applications of MRFs

- Many applications for low-level vision tasks
  - Image denoising

```
Observation process

yi
xi

“Smoothness constraints”

Noisy observations

“True” image content
```

Results by [Roth & Black, CVPR’05]

B. Leibe
Applications of MRFs

- Many applications for low-level vision tasks
  - Image denoising
  - Inpainting

Since 1699, when French explorers landed at the great bend of the Mississippi River and celebrated the first Mardi Gras in North America, New Orleans has brewed a fascinating melange of cultures. It was French, then Spanish, then French again, then sold to the United States. Through all these years, and even into the 1900s, others arrived from everywhere: Acadians (Cajuns), Africans, indige–
Applications of MRFs

- Many applications for low-level vision tasks
  - Image denoising
  - Inpainting
  - Image restoration

Results by [Roth & Black, CVPR’05]
Applications of MRFs

- Many applications for low-level vision tasks
  - Image denoising
  - Inpainting
  - Image restoration
  - Image segmentation
Applications of MRFs

- Many applications for low-level vision tasks
  - Image denoising
  - Inpainting
  - Image restoration
  - Image segmentation

Convert a low-res image into a high-res image!

Image source: [Freeman et al., CG&A’03]
Applications of MRFs

- Many applications for low-level vision tasks
  - Image denoising
  - Inpainting
  - Image restoration
  - Image segmentation
  - Super-resolution

Image source: [Freeman et al., CG&A'03]
Applications of MRFs

- Many applications for low-level vision tasks
  - Image denoising
  - Inpainting
  - Image restoration
  - Image segmentation
  - Super-resolution
  - Optical flow

Image patches

Scene patches

Image pair

Scene

Image source: William Freeman
Applications of MRFs

- Many applications for low-level vision tasks
  - Image denoising
  - Inpainting
  - Image restoration
  - Image segmentation
  - Super-resolution
  - Optical flow
  - Stereo depth estimation

Stereo image pair

Disparity map
Applications of MRFs

- Many applications for low-level vision tasks
  - Image denoising
  - Inpainting
  - Image restoration
  - Image segmentation
  - Super-resolution
  - Optical flow
  - Stereo depth estimation

- MRFs have become a standard tool for such tasks.
  - Let’s look at how they are applied in detail...
MRF Structure for Images

- **Basic structure**

- **Two components**
  - Observation model
    - How likely is it that node $x_i$ has label $L_i$ given observation $y_i$?
    - This relationship is usually learned from training data.
  - Neighborhood relations
    - Simplest case: 4-neighborhood
    - Serve as smoothing terms.
    - Discourage neighboring pixels to have different labels.
    - This can either be learned or be set to fixed “penalties”.

Noisy observations

“True” image content
MRF Nodes as Pixels

Original image

Degraded image

Reconstruction from MRF modeling pixel neighborhood statistics

These neighborhood statistics can be learned from training data!

Slide adapted from William Freeman
MRF Nodes as Patches

More general relationships expressed by potential functions $\Phi$ and $\Psi$. 

Slide credit: William Freeman
Network Joint Probability

- Interpretation of the factorized joint probability

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

- Scene
- Image
- Image-scene compatibility function
- Scene-scene compatibility function
- Local observations
- Neighboring scene nodes
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  
Pairwise potentials

• Single-node (unary) 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)

B. Leibe
How to Set the Potentials? Some Examples

- **Unary potentials**
  - E.g., color model, modeled with a Mixture of Gaussians
    \[
    \phi(x_i, y_i; \theta_\phi) = \log \sum_k \theta_\phi(x_i, k)p(k|x_i)\mathcal{N}(y_i; \bar{y}_k, \Sigma_k)
    \]
  
  ⇒ Learn color distributions for each label
How to Set the Potentials? Some Examples

- **Pairwise potentials**
  - **Potts Model**
    \[ \psi(x_i, x_j; \theta_\psi) = \theta_\psi \delta(x_i \neq x_j) \]
    - Simplest discontinuity preserving model.
    - Discontinuities between any pair of labels are penalized equally.
    - Useful when labels are unordered or number of labels is small.

  - **Extension: “contrast sensitive Potts model”**
    \[ \psi(x_i, x_j, g_{ij}(y); \theta_\psi) = \theta_\psi g_{ij}(y) \delta(x_i \neq x_j) \]
    where
    \[ g_{ij}(y) = e^{-\beta \|y_i - y_j\|^2} \]
    \[ \beta = 2 \cdot \text{avg} \left( \|y_i - y_j\|^2 \right) \]
    - Discourages label changes except in places where there is also a large change in the observations.
Extension: Conditional Random Fields (CRF)

- **Idea:** Model conditional instead of joint probability

  \[ \phi(D | x_i, x_j) \]

  \[ \phi(D | x_i) \]

- **Energy formulation**

  \[
  E(x) = \sum_{i \in S} \left( \phi(D | x_i) + \sum_{j \in N_i} \left( \phi(D | x_i, x_j) + \psi(x_i, x_j) \right) \right) + \text{const}
  \]

  **Unary likelihood**

  **Contrast Term**

  **Uniform Prior (Potts Model)**

Slide credit: Phil Torr
Example: MRF for Image Segmentation

- MRF structure

Pairwise potential
\[ \phi(D | x_i, x_j) \]

Unary potential
\[ \phi(D | x_i) \]

Pixels
Labels
Prior Potts model

Data (D)
Unary likelihood
Pair-wise Terms
MAP Solution

Slide credit: Phil Torr
Energy Minimization

- **Goal:**
  - Infer the optimal labeling of the MRF.

- **Many inference algorithms are available, e.g.**
  - Simulated annealing
  - Iterated conditional modes (ICM)
  - Belief propagation
  - Graph cuts
  - Variational methods
  - Monte Carlo sampling

- **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).
Topics of This Lecture

• Recap: Exact inference
  - Factor Graphs
  - Sum-Product/Max-Sum Belief Propagation
  - Junction Tree algorithm

• Applications of Markov Random Fields
  - Application examples from computer vision
  - Interpretation of clique potentials
  - Unary potentials
  - Pairwise potentials

• Solving MRFs with Graph Cuts
  - Graph cuts for image segmentation
  - s-t mincut algorithm
  - Extension to non-binary case
  - Applications
Graph Cuts for Binary Problems

- Idea: convert MRF into source-sink graph

Minimum cost cut can be computed in polynomial time
(max-flow/min-cut algorithms)

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

[Boykov & Jolly, ICCV’01]

Slide credit: Yuri Boykov
Simple Example of Energy

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

- **Unary potentials**
- **Pairwise potentials**
- **t-links**
- **n-links**

**Diagram:**
- \( D_p(t) \)
- \( D_p(s) \)
- \( L_p \in \{s, t\} \) (binary object segmentation)

*Slide adapted from Yuri Boykov*
Adding Regional Properties

Regional bias example

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

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

NOTE: hard constrains are not required, in general.

[Boykov & Jolly, ICCV’01]

Slide credit: Yuri Boykov
Adding Regional Properties

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

$D_p(s) \propto \exp \left( -\|I_p - I^s\|^2 / 2\sigma^2 \right)$

$D_p(t) \propto \exp \left( -\|I_p - I^t\|^2 / 2\sigma^2 \right)$

EM-style optimization

Slide credit: Yuri Boykov

[Boykov & Jolly, ICCV’01]
Topics of This Lecture

• Recap: Exact inference
  - Factor Graphs
  - Sum-Product/Max-Sum Belief Propagation
  - Junction Tree algorithm

• Applications of Markov Random Fields
  - Application examples from computer vision
  - Interpretation of clique potentials
  - Unary potentials
  - Pairwise potentials

• Solving MRFs with Graph Cuts
  - 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\}

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 an st-cut?
Sum of cost of all edges going from \(S\) to \(T\)

\[
5 + 2 + 9 = 16
\]
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

\[2 + 1 + 4 = 7\]
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
## 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>$O(n^2 mU)$</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>$O(nm \log U)$</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>$O(nm \log^2 n)$</td>
</tr>
<tr>
<td>1983</td>
<td>Sleator &amp; Tarjan</td>
<td>$O(nm \log n)$</td>
</tr>
<tr>
<td>1986</td>
<td>Goldberg &amp; Tarjan</td>
<td>$O(nm \log(n^2/m))$</td>
</tr>
<tr>
<td>1987</td>
<td>Ahuja &amp; Orlin</td>
<td>$O(nm + n^2 \log U)$</td>
</tr>
<tr>
<td>1987</td>
<td>Ahuja et al.</td>
<td>$O(nm \log(n\sqrt{\log U/m}))$</td>
</tr>
<tr>
<td>1989</td>
<td>Cheriyan &amp; Hagerup</td>
<td>$E(nm + n^2 \log^2 n)$</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>$O(nm + n^{8/3} \log n)$</td>
</tr>
<tr>
<td>1992</td>
<td>King et al.</td>
<td>$O(nm + n^{2+\epsilon})$</td>
</tr>
<tr>
<td>1993</td>
<td>Phillips &amp; Westbrook</td>
<td>$O(nm (\log_{m/n} n + \log^{2+\epsilon} n))$</td>
</tr>
<tr>
<td>1994</td>
<td>King et al.</td>
<td>$O(nm \log_{m/(n \log n)} n)$</td>
</tr>
<tr>
<td>1997</td>
<td>Goldberg &amp; Rao</td>
<td>$O(n^{3/2} \log(n^2/m) \log U)$</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
Applications: 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.cs.ucl.ac.uk/staff/V.Kolmogorov/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**.  
  \[ E(s,s) + E(t,t) \leq E(s,t) + E(t,s) \]

  \[ E(L) \text{ can be minimized by } s-t \text{ graph cuts} \iff \text{Submodularity ("convexity")} \]

- **Submodularity** is the discrete equivalent to convexity.
  - Implies that every local energy minimum is a global minimum.
  \[ \Rightarrow \text{Solution will be globally optimal.} \]
Topics of This Lecture

• Recap: Exact inference
  - Factor Graphs
  - Sum-Product/Max-Sum Belief Propagation
  - Junction Tree algorithm

• Applications of Markov Random Fields
  - Application examples from computer vision
  - Interpretation of clique potentials
  - Unary potentials
  - Pairwise potentials

• Solving MRFs with Graph Cuts
  - Graph cuts for image segmentation
  - s-t mincut algorithm
  - Extension to non-binary case
  - Applications
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
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
Iterated Graph Cuts

Result

Energy after each iteration

Color model (Mixture of Gaussians)

Slide credit: Carsten Rother
GrabCut: Example Results

- This is included in the newest versions of MS Office!

Image source: Carsten Rother
Applications: Interactive 3D Segmentation

Slide credit: Yuri Boykov
References and Further Reading

• A gentle introduction to Graph Cuts can be found in the following paper:

• Try the GraphCut implementation at [http://www.cs.ucl.ac.uk/staff/V.Kolmogorov/software.html](http://www.cs.ucl.ac.uk/staff/V.Kolmogorov/software.html)