Announcements

- Last lecture next Tuesday: Repetition
  - Summary of all topics in the lecture
  - “Big picture” and current research directions
  - Opportunity to ask questions

- Please use this opportunity and prepare questions!

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

Recap: Junction Tree Algorithm

- Motivation
  - Exact inference on general graphs.
  - Works by turning the initial graph into a junction tree and then running a sum-product-like algorithm.
  - Intractable on graphs with large cliques.

- Main steps
  1. If starting from directed graph, first convert it to an undirected graph by moralization.
  2. Introduce additional links by triangulation in order to reduce the size of cycles.
  3. Find cliques of the moralized, triangulated graph.
  4. Construct a new graph from the maximal cliques.
  5. Remove minimal links to break cycles and get a junction tree.
     ⇒ Apply regular message passing to perform inference.

Recap: Junction Tree Example

- Without triangulation step
  - The final graph will contain cycles that we cannot break without losing the running intersection property!

- When applying the triangulation
  - Only small cycles remain that are easy to break.
  - Running intersection property is maintained.
Recap: MRF Structure for Images

- Basic structure
  - Noisy observations
  - "True" image content

- 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".

Recap: How to Set the Potentials?

- Unary potentials
  - E.g. color model, modeled with a Mixture of Gaussians
    - Learn color distributions for each label

$$\phi(x_i; y_i; \theta) = \log \sum_k \theta_k(x_i) p(k|x_i) N(y_i; \mu_k, \Sigma_k)$$

- Pairwise potentials
  - Potts Model
    - 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"
    - $\psi(x_i, x_j; g_{ij}(y); \theta) = \theta_{g_{ij}}(y) \delta(x_i \neq x_j)$
    - $g_{ij}(y) = e^{-\beta I_{ij}}$
    - Discourages label changes except in places where there is also a large change in the observations.

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

- 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)

[Boykov & Jolly, ICCV’01]
Simple Example of Energy

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

- unary potentials
- pairwise potentials

\[
D_p(t) = \exp \left( -\frac{\Delta p}{2\sigma^2} \right)
\]

Regional bias example

Suppose \( I \) and \( I' \) are given "expected" intensities of object and background

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

Object and background color distributions

EM-style optimization

Topics of This Lecture

- 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 \( E = \{(v_i, v_j) \ldots\} \)

Costs \( C = \{c_{ij} \ldots\} \)

Slide credit: Yuri Boykov
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 \(T\)

\[5 + 2 + 9 = 16\]

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 \(T\)

\[2 + 1 + 4 = 7\]

Slide credit: Pushmeet Kohli

---

**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.

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>1956</td>
<td>Ford &amp; Fulkerson</td>
<td>(O(</td>
</tr>
<tr>
<td>1957</td>
<td>Edmonds</td>
<td>(O(</td>
</tr>
<tr>
<td>1957</td>
<td>Edmonds &amp; Karp</td>
<td>(O(</td>
</tr>
<tr>
<td>1971</td>
<td>Dinic</td>
<td>(O(</td>
</tr>
<tr>
<td>1974</td>
<td>Korzysty</td>
<td>(O(</td>
</tr>
<tr>
<td>1979</td>
<td>Cherkassky &amp; Goldberg</td>
<td>(O(</td>
</tr>
<tr>
<td>1981</td>
<td>Tarjan &amp; Yannakakis</td>
<td>(O(</td>
</tr>
<tr>
<td>1984</td>
<td>Goldberg &amp; Tarjan</td>
<td>(O(</td>
</tr>
<tr>
<td>1987</td>
<td>Alon &amp; Galil</td>
<td>(O(</td>
</tr>
<tr>
<td>1987</td>
<td>Alon &amp; Yoda</td>
<td>(O(</td>
</tr>
<tr>
<td>1990</td>
<td>Goldberg &amp; Hagerup</td>
<td>(O(</td>
</tr>
<tr>
<td>1993</td>
<td>Goldberg &amp; Tarjan</td>
<td>(O(</td>
</tr>
<tr>
<td>1997</td>
<td>Goldberg &amp; Han</td>
<td>(O(</td>
</tr>
</tbody>
</table>

Slide credit: Andrew Goldberg

---

Maxflow Algorithms

**Flow = 0**

**Augmenting Path Based Algorithms**

1. Find path from source to sink with positive capacity
2. Push maximum possible flow through this path
3. Repeat until no path can be found

Algorithms assume non-negative capacity

Slide credit: Pushmeet Kohli

---

Maxflow Algorithms

**Flow = 0**

**Augmenting Path Based Algorithms**

1. Find path from source to sink with positive capacity
2. Push maximum possible flow through this path
3. Repeat until no path can be found

Algorithms assume non-negative capacity

Slide credit: Pushmeet Kohli
Maxflow Algorithms

Flow = 0 + 2

Augmenting Path Based Algorithms

1. Find path from source to sink with positive capacity
2. Push maximum possible flow through this path
3. Repeat until no path can be found

Algorithms assume non-negative capacity

Slide credit: Pushmeet Kohli
B. Leibe

Maxflow Algorithms

Flow = 2

Algorithm

1. Find path from source to sink with positive capacity
2. Push maximum possible flow through this path
3. Repeat until no path can be found

Algorithms assume non-negative capacity

Slide credit: Pushmeet Kohli
B. Leibe

Maxflow Algorithms

Flow = 2 + 4

Augmenting Path Based Algorithms

1. Find path from source to sink with positive capacity
2. Push maximum possible flow through this path
3. Repeat until no path can be found

Algorithms assume non-negative capacity

Slide credit: Pushmeet Kohli
B. Leibe

Maxflow Algorithms

Flow = 6

Augmenting Path Based Algorithms

1. Find path from source to sink with positive capacity
2. Push maximum possible flow through this path
3. Repeat until no path can be found

Algorithms assume non-negative capacity

Slide credit: Pushmeet Kohli
B. Leibe
Maxflow Algorithms

Flow = 6

Augmenting Path Based Algorithms

1. Find path from source to sink with positive capacity
2. Push maximum possible flow through this path
3. Repeat until no path can be found

Algorithms assume non-negative capacity

Slide credit: Pushmeet Kohli

B. Leibe

Maxflow Algorithms

Flow = 6 + 1

Augmenting Path Based Algorithms

1. Find path from source to sink with positive capacity
2. Push maximum possible flow through this path
3. Repeat until no path can be found

Algorithms assume non-negative capacity

Slide credit: Pushmeet Kohli

B. Leibe

Applications: Maxflow in Computer Vision

- Specialized algorithms for vision problems
  - Grid graphs
  - Low connectivity (m – 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

Slide credit: Pushmeet Kohli

B. Leibe

When Can s-t Graph Cuts Be Applied?

\[ E(L) = \sum_p E_p(L_p) + \sum_{p_1,p_2} E_{p_1,p_2}(L_{p_1},L_{p_2}) \]

- s-t graph cuts can only globally minimize binary energies that are submodular.
  \[ E(L) \text{ can be minimized by s-t graph cuts} \iff E(s,t) \leq E(s,t) + E(t,s) \]

Submodularity (“convexity”)

- Submodularity is the discrete equivalent to convexity.
  - Implies that every local energy minimum is a global minimum.
  \[ \Rightarrow \text{Solution will be globally optimal.} \]
Recap: Simple Binary Image Denoising Model

- **MRF Structure**

  Observation process

  "True" image content

  "Smoothness constraints" \( x_i, y_i \in \{-1, 1\} \)

  Example: simple energy function

  \[
  E(x, y) = h \sum_i x_i + \beta \sum_{(i,j)} \delta(x_i \neq x_j) - \eta \sum_i x_i y_i
  \]

  - Smoothness term: fixed penalty \( \beta \) if neighboring labels disagree.
  - Observation term: fixed penalty \( \eta \) if label and observation disagree

  B. Leibe
  Image source: C. Bishop, 2006

Converting an MRF into an s-t Graph

- **Conversion:**

  Conversion:

  \[
  L_4 = -1
  \]

  \[
  L_4 = 1
  \]

- **Energy:**

  \[
  E(x, y) = h \sum_i x_i + \beta \sum_{(i,j)} \delta(x_i \neq x_j) - \eta \sum_i x_i y_i
  \]

  - Unary potentials are straightforward to set. How?
  - Pairwise potentials are more tricky, since we don’t know \( x_i \)!

  Tricks:
  - the pairwise energy only has an influence if \( x_i \neq x_j \).
  - (Only!) in this case, the cut will go through the edge \( \{i,j\} \).

  B. Leibe

Topics of This Lecture

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

- **Dealing with Non-Binary Cases**

  - Limitation to binary energies is often a nuisance.
  - E.g. binary segmentation only...
  - We would like to solve also multi-label problems.
    - The bad news: Problem is NP-hard 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 (\( \beta \)-approx) and converges in a few iterations.

  B. Leibe

\( \alpha \)-Expansion Move

- **Basic idea:**

  - Break multi-way cut computation into a sequence of binary s-t cuts.
**α-Expansion Algorithm**

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

**Example: Stereo Vision**

- **Original pair of “stereo” images**
- **Depth map**
- **Ground truth**

**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: \( \Rightarrow \) binary optimization problem

**Topics of This Lecture**

- 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.

**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 credits:**
- Yuri Boykov
- Matthieu Bray
- Carsten Rother
Iterated Graph Cuts

Energy after each iteration

Result

Color model (Mixture of Gaussians)

GrabCut: Example Results

This is included in the newest version of MS Office!

Applications: Interactive 3D Segmentation

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