Machine Learning - Lecture 18
Inference & Applications
12.07.2016

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

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.
    \( p(x) \propto \prod_{x \in \omega(x)} \mu_{f_s \rightarrow x}(x) \)
• Computational effort
  » Total number of messages \( = 2 \times \text{number of graph edges} \).
Two kinds of messages
- Message from factor node to variable nodes:
  - Sum of factor contributions
    \[ \mu_{f \to x}(x) \equiv \sum_{X_{\setminus x}} f(x, X_{\setminus x}) \]
    \[ = \sum_{X_{\setminus x}} \phi_{x}(x) \prod_{m \in \text{e}(f) \setminus x} \mu_{x_m \to f}(x_{m}) \]
  - Message from variable node to factor node:
    - Product of incoming messages
      \[ \mu_{x_{m} \to f}(x_{m}) \equiv \prod_{i \in \text{e}(x_{m}) \setminus f} \mu_{i \to x_{m}}(x_{m}) \]

⇒ Simple propagation scheme.

Key idea 1: Distributive Law (again)
- \( \max(ab, ac) = a \max(b, c) \)
- \( \max(a+b, a+c) = a + \max(b, c) \)

⇒ 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
  \[ p(x^{\text{max}}) = \max_{x} p(x) \]
  \[ = \max_{x} \ln \max p(x) = \max_{x} \ln p(x) \]
- For numerical reasons, use the logarithm.
  \[ \ln \max p(x) = \max \ln p(x) \]

⇒ Maximize the sum of log-probabilities.

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 are joint maximum.
  - Example:
    \[ \begin{align*}
    y &= 0 \quad x &= 1 \\
    y &= 1 \quad 0.3 & 0.0
    \end{align*} \]
    \[ \arg \max_x p(x, y) = 1 \quad \arg \max_x p(x) = 0 \]

Max-Sum Algorithm
- Maximizing over a chain (max-product)
Max-Sum Algorithm

- Initialization (leaf nodes)
  \[ \psi_{x \rightarrow f}(x) = 0 \quad \mu_{f \rightarrow x}(x) = \ln f(x) \]

- Recursion
  - Messages
    \[ \mu_{f \rightarrow x}(x) = \mathop{\operatorname{max}}_{x_{\chi \setminus f}} \left[ \ln f(x) + \sum_{m \in \chi(f) \setminus x} \mu_{m \rightarrow f}(x_m) \right] \]
    \[ \mu_{x \rightarrow f}(x) = \sum_{l \in \text{dom}(x)f} \mu_{f \rightarrow x}(x) \]
  - For each node, keep a record of which values of the variables gave rise to the maximum state:
    \[ \phi(x) = \mathop{\operatorname{arg\ max}}_{x_{\chi \setminus x}} \left[ \ln f(x) + \sum_{m \in \chi(f) \setminus x} \mu_{m \rightarrow f}(x_m) \right] \]

Visualization of the Back-Tracking Procedure

- Example: Markov chain

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...

Applications of MRFs

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

Applications of MRFs

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

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

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

- Optical flow

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
  - Stereo depth estimation

MRF Structure for Images

- Basic structure
  - 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”.

MRF Nodes as Patches

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

Network Joint Probability

- Interpretation of the factorized joint probability

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

- **Energy function**
  \[ E(x, y) = \sum_x \phi(x, y) + \sum_{x,j} \psi(x, x_j) \]

  - Single-node potentials \( \phi \)
    - 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)

- **Single-node (unary) potentials** \( \phi \)
  - E.g., color model, modeled with a Mixture of Gaussians
  \[ \phi(x_i, y; \theta_k) = \log \sum_k \theta_k p(k|x_i) N(y_i; \bar{y}_k, \Sigma_k) \]

  \[ \Rightarrow \text{Learn color distributions for each label} \]

  - Pairwise potentials \( \psi \)
    - Discourages label changes except in places where there is also a large change in the observations.

How to Set the Potentials? Some Examples

- **Pairwise potentials**
  - Potts Model
    \[ \psi(x_i, x_j; \theta) = \theta \delta(x_i \neq x_j) \]

    - 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”
    \[ \psi(x_i, x_j; g_{ij}(y); \theta_y) = \theta g_{ij}(y) \delta(x_i \neq x_j) \]

    where
    \[ g_y(y) = e^{-\beta |y-y_i|} \]

    - Discourages label changes except in places where there is also a large change in the observations.

- **Unary potentials**
  - E.g., color model, modeled with a Mixture of Gaussians

  \[ \phi(x_i, y; \theta_k) = \log \sum_k \theta_k p(k|x_i) N(y_i; \bar{y}_k, \Sigma_k) \]

  \[ \Rightarrow \text{Learn color distributions for each label} \]

Example: MRF for Image Segmentation

- **MRF structure**
  - Pairwise potential \( \phi(D|x_i, x_j) \)
  - Unary potential \( \phi(D|x_i) \)

  - Pixels
  - Labels

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

Extension: Conditional Random Fields (CRF)

- **Idea:** Model conditional instead of joint probability
  \[ \psi(D|x_i) \]

- **Energy formulation**
  \[ E(x) = \sum_i \phi(D|x_i) + \sum_{i,j} (\phi(D|x_i, x_j) + \psi(x_i, x_j)) + \text{const} \]

  - Uniform Prior
  - Contrast Term
  - Unary Likelihood

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)

**Simple Example of Energy**

\[ E(L) = \sum_p D_p(L_p) + \sum_{p,q} w_{pq} \delta(L_p \neq L_q) \]

Unary potentials \( D_p \), pairwise potentials \( w_{pq} \). Links \( L_p \in \{s,t\} \)

(binary object segmentation)

**Adding Regional Properties**

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

Regional bias example

\[ D_p(t) = \exp \left( -\frac{||I_p - I'||^2}{2\sigma^2} \right) \]

EM-style optimization

**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

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

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

Source

\( \{v_1, v_2 \ldots v_n\} \)

\( \{(v_1, v_2) \ldots\} \)

Costs \( \{c_{ij}\} \)

Sink

\( 2\ + \ 1\ + \ 4\ = \ 7 \)

Slide credit: Pushmeet Kohli

The s-t-Mincut Problem

Source

\( \{v_1, v_2 \ldots v_n\} \)

\( \{(v_1, v_2) \ldots\} \)

Costs \( \{c_{ij}\} \)

Sink

\( 5\ + \ 2\ + \ 9\ = \ 16 \)

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

\( n: \) nodes
\( m: \) edges
\( U: \) maximum edge weight

<table>
<thead>
<tr>
<th>Year</th>
<th>Discoverer(s)</th>
<th>Library</th>
</tr>
</thead>
<tbody>
<tr>
<td>1956</td>
<td>Dantzig</td>
<td>O(nm)</td>
</tr>
<tr>
<td>1956</td>
<td>Ford &amp; Fulkerson</td>
<td>O(nm)</td>
</tr>
<tr>
<td>1959</td>
<td>Edmonds &amp; Karp</td>
<td>O(nm^2)</td>
</tr>
<tr>
<td>1959</td>
<td>Dantzig</td>
<td>O(nm^2)</td>
</tr>
<tr>
<td>1973</td>
<td>Edmonds</td>
<td>O(n^3)</td>
</tr>
<tr>
<td>1974</td>
<td>Lawler</td>
<td>O(n^3)</td>
</tr>
<tr>
<td>1977</td>
<td>Cherkassky</td>
<td>O(nm^2)</td>
</tr>
<tr>
<td>1977</td>
<td>Cull &amp; Sklansky</td>
<td>O(nm^2)</td>
</tr>
<tr>
<td>1983</td>
<td>Stoer &amp; Tijdeman</td>
<td>O(nm^2)</td>
</tr>
<tr>
<td>1985</td>
<td>Goldberg &amp; Tarjan</td>
<td>O(nmlog^2(n))</td>
</tr>
<tr>
<td>1987</td>
<td>Ahuja &amp; Orlin</td>
<td>O(nm + n^2 log(n))</td>
</tr>
<tr>
<td>1987</td>
<td>Ahuja et al</td>
<td>O(nm + n^2 log(n))</td>
</tr>
<tr>
<td>1988</td>
<td>Charpan &amp; Hagimag</td>
<td>O(nm + n^2 log(n))</td>
</tr>
<tr>
<td>1988</td>
<td>Charpan &amp; Vazirani</td>
<td>O(nm + n^2 log(n))</td>
</tr>
<tr>
<td>1990</td>
<td>Charpan &amp; Vazirani</td>
<td>O(nm + n^2 log(n))</td>
</tr>
<tr>
<td>1990</td>
<td>Ahuja et al</td>
<td>O(nm + n^2 log(n))</td>
</tr>
<tr>
<td>1992</td>
<td>Ahuja &amp; Orlin</td>
<td>O(nm + n^2 log(n))</td>
</tr>
<tr>
<td>1993</td>
<td>Ahuja et al</td>
<td>O((n+m)log(n))</td>
</tr>
<tr>
<td>1994</td>
<td>Ahuja et al</td>
<td>O(n^2 log^2(n))</td>
</tr>
<tr>
<td>1997</td>
<td>Goldberg &amp; Ries</td>
<td>O(n^2 log^2(n))</td>
</tr>
<tr>
<td>1997</td>
<td>Goldberg &amp; Ries</td>
<td>O(n^2 log^2(n))</td>
</tr>
</tbody>
</table>

 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?

\[ E(L) = \sum_{t\text{-links}} E_p(L_p) + \sum_{n\text{-links}} E(L_p, L_q) \]

- s-t graph cuts can only globally minimize binary energies that are submodular.
- Submodularity is the discrete equivalent to convexity.
  - Implies that every local energy minimum is a global minimum.
  - Solution will be globally optimal.

Submodularity is the discrete equivalent to convexity.

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

Submodularity ("convexity")

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.

User segmentation cues

GrabCut: Data Model

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

GrabCut: Example Results

- This is included in the newest versions 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](http://www.cs.ucl.ac.uk/staff/V.Kolmogorov/software.html)