CSE 659A: Advances in Computer Vision

Spring 2019: T-R: 2:30-4pm @ Cupples II/230

Instructor: Ayan Chakrabarti (ayan@wustl.edu).

http://www.cse.wustl.edu/~ayan/courses/cse659a/

Jan 31, 2019
Last time we talked about MRFs, learning despite the intractable partition function.

Then we talked about MRFs, and specifically pair-wise MRFs (although we said you could convert generic clique MRFs to a pair-wise factor graph).

We talked about Belief Propagation & Loopy BP as a way of doing inference (for both max-marginal and MAP estimation).
MRFS: APPLICATIONS

\[ P(X) \propto \exp \left( - \sum_i E_{ii}(X_i) - \sum_{ij} E_{ij}(X_i, X_j) \right) \]

Example Stereo \( X_i \in \{0, 1, \ldots, D\} \)

Unary Terms

Pairwise Terms

Come from Matching

Smoothness

Minimize Only Unary Term at Each Pixel

Result with MRF
MRFS: APPLICATIONS

\[ P(X) \propto \exp \left( - \sum_i E_{ii}(X_i) - \sum_{ij} E_{ij}(X_i, X_j) \right) \]

Example Segmentation

\[ X_i \in \{\text{road, grass, tree, bench, \ldots} \} \]

Comes from Classifier / CNN

Unary Terms

Pairwise Terms

Smoothness

Minimize Only Unary Term at Each Pixel

Result with MRF
TRW-* Algorithms: Represent a graph with cycles as a convex combination of overlapping trees (each tree is formed by dropping some edges).

- Do belief propagation on each tree, using input from other trees and appropriate weighting.
Gaussian Belief Propagation

- We've considered belief propagation for discrete-valued variables because messages are finite dimensional vectors.
- What if the variables are continuous-valued? The messages would be functions of continuous variables.

\[
m_{i \rightarrow j}(x_j) = \sum_{x_i} \left[ \Psi(x_i) \psi(x_i, x_j) \prod_{k: (i,k) \in E, k \neq j} m_{k \rightarrow i}(x_i) \right]
\]

becomes

\[
m_{i \rightarrow j}(x_j) = \int_{x_i} \left[ \Psi(x_i) \psi(x_i, x_j) \prod_{k: (i,k) \in E, k \neq j} m_{k \rightarrow i}(x_i) \right] dx_i
\]

- But if all our unary and pairwise potentials were uni-variate and bi-variate Gaussians, so would their products and marginals with updated means and co-variances.
- Pass messages that contain these means and co-variances.
Particle Belief Propagation: Ihler and McAllester, AISTATS 2009

\[ m_{i \rightarrow j}(x_j) = \int_{x_i} \left[ \Psi_i(x_i) \psi(x_i, x_j) \prod_{k:(i,k) \in E, k \neq j} m_{k \rightarrow i}(x_i) \right] dx_i \]

- The other option is to create a finite sample-set for each continuous variable \( \{x_i^s\}_s \), for all \( x_i \).
- Define \( m_{i \rightarrow j} \) as a continuous function of \( x_j \), but by averaging over the samples of \( x_i \).

\[ m_{i \rightarrow j}(x_j) = \frac{1}{S} \sum_{x_i^s} \left[ \Psi_i(x_i^s) \psi(x_i^s, x_j) \prod_{k:(i,k) \in E, k \neq j} m_{k \rightarrow i}(x_i^s) \right] \]

- Periodically re-sample all continuous variables based on the current estimate of their marginal distribution.
MRFS: PAIRWISE TERMS

\[
P(\{x_i \in V\}) = \frac{1}{Z} \prod_{x_i \in V} \Psi_i(x_i) \prod_{(x_i, x_j) \in E} \psi_{i,j}(x_i, x_j)
\]

\[
\Psi_i(x_i) = -\log E_{ii} x_i, \quad \psi_{i,j}(x_i, x_j) = -\log E_{ij}(x_i, x_j)
\]

- Its common to make the pairwise term factor into a term for labels scaled by a location dependent factor

\[
E_{ij}(x_i, x_j) = \mu_{ij} \times V(x_i, x_j)
\]

- Examples of label compatibility \( V \):

\[
V(x_i, x_j) = \delta[x_i \neq x_j]
\]

Called Pott's Model. Penalty if \( x_i \) not equal to \( x_j \).

Sometimes, you might have different compatibilities. For example, for the case of disparity:

\[
V(x_i, x_j) = \min(2, |x_i - x_j|)
\]

Or a general matrix for segmentation class labels (car and road has less penalty than car and water).
MRFS: PAIRWISE TERMS

\[ P(\{x_i \in V\}) = \frac{1}{Z} \prod_{x_i \in V} \Psi_i(x_i) \prod_{(x_i, x_j) \in E} \psi_{i,j}(x_i, x_j) \]

\[ \Psi_i(x_i) = -\log E_{ii}x_i, \quad \psi_{i,j}(x_i, x_j) = -\log E_{ij}(x_i, x_j) \]

- It's common to make the pairwise term factor into a term for labels scaled by a location dependent factor

\[ E_{ij}(x_i, x_j) = \mu_{ij} \times V(x_i, x_j) \]

The per-edge weight \( \mu_{ij} \) can also be based on different things:

- \( \mu_{ij} = 1 \), but graph has only edges in a neighborhood: \( E = \bigcup \forall i\{(i, j) : j \in \mathcal{N}_i\} \)

E.g., for each pixel \( i \), add an edge to all neighbors in a 5x5 neighborhood around that pixel \( i \).

- Same, but \( \mu_{ij} = \exp\left(-\frac{\|p_i-p_j\|^2}{2\sigma_s^2}\right) \). Here \( p_i \) is the \([x, y]\) location of pixel \( i \) and \( \sigma_s^2 \) is some chosen parameter.

- Same, but \( \mu_{ij} = \exp\left(-\frac{\|p_i-p_j\|^2}{2\sigma_s^2} - \frac{\|I_i-I_j\|^2}{2\sigma_i^2}\right) \). Add a term based on pixel intensities \( I_i \) in a reference image. Kind of like a bilateral filter, but used to define compatibility weights.
MRFS: PAIRWISE TERMS

\[ P(\{ x_i \in V \}) = \frac{1}{Z} \prod_{x_i \in V} \Psi_i(x_i) \prod_{(x_i, x_j) \in E} \psi_{i,j}(x_i, x_j) \]

\[ \Psi_i(x_i) = -\log E_{ii} x_i, \quad \psi_{i,j}(x_i, x_j) = -\log E_{ij}(x_i, x_j) \]

- Its common to make the pairwise term factor into a term for labels scaled by a location dependent factor

\[ E_{ij}(x_i, x_j) = \mu_{ij} \times V(x_i, x_j) \]

The per-edge weight \( \mu_{ij} \) can also be based on different things:

- Same, but \( \mu_{ij} = \exp \left( -\frac{\| p_i - p_j \|^2}{2\sigma_i^2} - \frac{\| I_i - I_j \|^2}{2\sigma_i^2} \right) \). Add a term based on pixel intensities \( I_i \) in a reference image. Kind of like a bilateral filter, but used to define compatibility weights.

- Use of weighted \( \mu_{ij} \) is common with much larger neighborhoods.

- Fully-connected MRFs: every pixel is connected to every other pixel, but with the above kind of \( \mu_{ij} \) which down-weights the contribution of pixels too far away (and too dissimilar nearby pixels in the reference).
MRF INFERENCE: MEAN FIELD ALGORITHM

\[ P(\{x_i \in V\}) = \frac{1}{Z} \prod_{x_i \in V} \Psi_i(x_i) \prod_{(x_i, x_j) \in E} \psi_{i,j}(x_i, x_j) \]

- Kind of like message passing as we'll see.
- Motivated as approximating the joint probability as a product of marginals.

\[ P(\{x_i \in V\}) = \prod_i Q_i(x_i) \]

And then compute \( x_i = \text{arg max} \ Q_i(x_i) \) for each node/pixel independently.
MRF INFERENCE: MEAN FIELD ALGORITHM

\[
P(\{x_i \in V\}) = \frac{1}{Z} \prod_{x_i \in V} \Psi_i(x_i) \prod_{(x_i, x_j) \in E} \psi_{i,j}(x_i, x_j)
\]

\[
P(\{x_i \in V\}) = \prod_i Q_i(x_i)
\]

- Compute \(Q_i\) iteratively with a loopy BP-like algorithm.
- Update \(Q_i\) in each step based messages from all neighbors
  - In message from \(i \rightarrow j\), this will include \(i\)'s belief based on message from \(j \rightarrow i\).
MRF INFEERENCE: MEAN FIELD ALGORITHM

\[
P(\{x_i \in V\}) = \frac{1}{Z} \prod_{x_i \in V} \Psi_i(x_i) \prod_{(x_i, x_j) \in E} \psi_{i,j}(x_i, x_j)
\]

\[
P(\{x_i \in V\}) = \prod_i Q_i(x_i)
\]

Each \(x_i \in \mathcal{L}_i\).

- At iteration 0, set \(Q_i^0(x_i) = \frac{1}{Z_i} \Psi_i(x_i)\), with \(Z_i\) that \(\sum_{\mathcal{L}_i} Q_i^0(x_i) = 1\).

- At each iteration \(t\), compute for all \(i\):

  [Correction from class: the definition in terms of energies as discussed in later slides in class is the actual mean-field algorithm. In terms of potentials, updates in mean-field take expectation over log-potentials not potentials (i.e., NOT \(\sum Q(x_j) \psi(x_i, x_j)\)). The updated definition below is correct and now definitions in terms of \(\Psi, \psi\) and \(E_{ii}, E_{ij}\) are consistent.]

\[
Q_i^{t+1}(x_i) = \Psi_i(x_i) \prod_{j: (i,j) \in E} \exp \left( \sum_{x_j \in \mathcal{L}_j} Q_j^t(x_j) \log \psi_{i,j}(x_i, x_j) \right)
\]

Think of \(\sum_{x_j \in \mathcal{L}_j} Q_j^t(x_j) \log \psi_{i,j}(x_i, x_j) = \mathbb{E}_{Q_j(x_j)} \log \psi_{i,j}(x_i, x_j)\)

Kind of like \(m_{j \to i}\), except in log-domain and \(Q_j^t(x_j)\) includes belief from \(i\) to \(j\) in the previous iteration.
MRF INFERENCE: MEAN FIELD ALGORITHM

\[ P(\{x_i \in V\}) = \frac{1}{Z} \prod_{x_i \in V} \Psi_i(x_i) \prod_{(x_i, x_j) \in E} \psi_{i,j}(x_i, x_j) \]

\[ P(\{x_i \in V\}) = \prod_i Q_i(x_i) \]

Each \( x_i \in \mathcal{L}_i \).

- At iteration 0, set \( Q_i^0(x_i) = \frac{1}{Z_i} \Psi_i(x_i) \), with \( Z_i \) that \( \sum_{\mathcal{L}_i} Q_i^0(x_i) = 1 \).

- At each iteration \( t \), compute for all \( i \):

\[
Q_i^{t+1}(x_i) = \Psi_i(x_i) \prod_j \exp \left( \sum_{x_j \in L_j} Q_j^t(x_j) \log \psi_{i,j}(x_i, x_j) \right)
\]

- Normalize to get \( Q_i^{t+1}(x_i) = \frac{1}{Z_i} Q_i^{t+1}(x_i) \) with \( Z_i \) that \( \sum_{\mathcal{L}_i} Q_i^{t+1}(x_i) = 1 \).

Repeat iteratively till convergence.
MRF INFERENCE: MEAN FIELD ALGORITHM

\[ P(\{x_i \in V\}) = \frac{1}{Z} \prod_{x_i \in V} \Psi_i(x_i) \prod_{(x_i, x_j) \in E} \psi_{i,j}(x_i, x_j) \]

\[ Q''_{t+1}(x_i) = \Psi_i(x_i) \prod_{j: (i, j) \in E} \exp \left( \sum_{x_j \in L_j} Q'_j(x_j) \log \psi_{i,j}(x_i, x_j) \right) \]

Let's write this in terms of energies.

\[ P(\{x_i \in V\}) = \frac{1}{Z} \exp \left( - \sum_{x_i \in V} E_{ii}(x_i) - \sum_{(x_i, x_j) \in E} E_{i,j}(x_i, x_j) \right) \]

\[ Q''_{t+1}(x_i) = \exp \left( -E_{ii}(x_i) - \sum_{j: (i, j) \in E} \sum_{x_j \in L_j} Q'_j(x_j) E_{i,j}(x_i, x_j) \right) \]
MRF INFERENCE: MEAN FIELD ALGORITHM

\[ P(\{x_i \in V\}) = \frac{1}{Z} \exp \left( - \sum_{x_i \in V} E_{ii}(x_i) - \sum_{(x_i, x_j) \in E} E_{ij}(x_i, x_j) \right) \]

\[ Q''_{i+1}^{t}(x_i) = \exp \left( -E_{ii}(x_i) - \sum_{j: (i, j) \in E} \sum_{x_j \in L_i} Q_j^{t}(x_j) E_{ij}(x_i, x_j) \right) \]

Let's consider the case when \( E_{ij}(x_i, x_j) = \mu_{ij} V(x_i, x_j) \) and all pixels have the same set of possible labels.

\[ Q''_{i+1}^{t}(x_i) = \exp \left( -E_{ii}(x_i) - \sum_{j: (i, j) \in E} \sum_{l': E_i, l' \in L_i} Q_j^{t}(l') \mu_{ij} V(x_i, l') \right) \]

\[ Q''_{i+1}^{t}(x_i) = \exp \left( -E_{ii}(x_i) - \sum_{l' \in L} V(x_i, l') \sum_{j: (i, j) \in E} Q_j^{t}(l') \mu_{ij} \right) \]
MRF INFERENCE: MEAN FIELD ALGORITHM

\[
P(\{x_i \in V\}) = \frac{1}{Z} \exp \left( - \sum_{x_i \in V} E_{ii}(x_i) - \sum_{(x_i, x_j) \in E} E_{ij}(x_i, x_j) \right)
\]

\[
E_{ij}(x_i, x_j) = \mu_{ij} V(x_i, x_j)
\]

\[
Q'''_{t+1}(x_i) = \exp \left( -E_{ii}(x_i) - \sum_{l' \in L} V(x_i, l') \sum_{j; (i, j) \in E} Q'_j(l') \mu_{ij} \right)
\]

- Let's say our graph is over pixels in an image, and there at C classes in \( L \). Then,
  - \( E_{ii} \) is a \( H \times W \times C \) image. Represent as \( U[n] \in \mathbb{R}^C \) (for unary).
  - At each iteration, \( Q'''[n] \), \( Q'[n] \) are also \( H \times W \times C \) images.
  - \( V \) is a \( C \times C \) matrix

Let's consider the case where \( \mu_{ij} = k(-(n_j - n_i)) \).

Think of it as a kernel with support based on size of neighborhood \( \mathcal{N}_i \).

E.g., \( k[n] = \exp(-n^2/(2\sigma_s^2)) \)
MRF INFERENICE: MEAN FIELD ALGORITHM

\[ P(\{x_i \in V\}) = \frac{1}{Z} \exp \left( - \sum_{x_i \in V} E_{ii}(x_i) - \sum_{(x_i, x_j) \in E} E_{ij}(x_i, x_j) \right) \]

\[ E_{ij}(x_i, x_j) = \mu_{ij} V(x_i, x_j) \]

\[ Q''_{t+1}(x_i) = \exp \left( -E_{ii}(x_i) - \sum_{l' \in L} V(x_i, l') \sum_{j: (i, j) \in E} Q'_j(l') \mu_{ij} \right) \]

\[ \mu_{ij} = k(-(n_j - n_i)) \text{ e.g. } k[n] = \exp(-n^2/(2\sigma_s^2)) \]

\[ Q''_{t+1}[n] = \exp(-U[n] - (Q' \ast k)[n] \times V^T) \]

- The summation \( \sum_{j: (i,j) \in E} Q'_j(l') \mu_{ij} \) is a convolution on the previous marginal distribution "image"! Convolving all channels with the same kernel \( k \).
- The \( \times V^T \) implies doing a matrix-multiply to the \( C \)-dimensional vector at each pixel location \( (Q' \ast k)[n] \).
- Get \( Q'_{t+1}[n] \) from \( Q''_{t+1}[n] \) by doing a per-pixel normalization.
MRF INFERENCE: MEAN FIELD ALGORITHM

\[
P(\{x_i \in V\}) = \frac{1}{Z} \exp \left( - \sum_{x_i \in V} E_{ii}(x_i) - \sum_{(x_i, x_j) \in E} E_{ij}(x_i, x_j) \right)
\]

\[
E_{i,j}(x_i, x_j) = \mu_{i,j} V(x_i, x_j)
\]

\[
Q''_{t+1}(x_i) = \exp \left( -E_{ii}(x_i) - \sum_{l' \in L} V(x_i, l') \sum_{j:(i,j) \in E} Q'_j(l') \mu_{ij} \right)
\]

\[
\mu_{ij} = k(-(n_j - n_i)) \quad \text{e.g.} \quad k[n] = \exp(-n^2/(2\sigma_s^2))
\]

\[
Q''_{t+1}[n] = \exp \left( -U[n] - (Q' \ast k)[n] \times V^T \right)
\]

```python
def solve(U, k, V):
    C = U.shape[-1]
    Q = exp(-U) / sum(exp(-U), axis=-1)
    for it in range(MAXITER):
        for c in range(C):
            Q[:,:,c] = conv2(Q[:,:,c], k)
        Q = np.reshape(np.matmul(np.reshape(Q, [-1, C]), V.T), U.shape)
    Q = exp(-U-Q)
    Q = Q / sum(exp(Q), axis=-1)
```
MRF INFERENCE: MEAN FIELD ALGORITHM

\[
P(\{x_i \in V\}) = \frac{1}{Z} \exp\left( - \sum_{x_i \in V} E_{ii}(x_i) - \sum_{(x_i, x_j) \in E} E_{ij}(x_i, x_j) \right), E_{ii}(x_i, x_j) = \mu_{ij} V(x_i, x_j)
\]

\[
Q''^{t+1}(x_i) = \exp \left( -E_{ii}(x_i) - \sum_{l' \in L} V(x_i, l') \sum_{j: (i, j) \in E} Q_j(l') \mu_{ij} \right)
\]

Spatial \( \mu_{ij} \)

\[
\mu_{ij} = k(-(n_j - n_i)), \quad Q''^{t+1}[n] = \exp(-U[n] - (Q^t \ast k)[n] \times V^T)
\]

Intensity-based \( \mu_{ij} \)

\[
\mu_{ij} = \exp \left( -\frac{|n_i - n_j|^2}{2\sigma_s^2} - \frac{||I[n_i] - I[n_j]||^2}{2\sigma_i^2} \right)
\]

Replace the convolution with Bilateral filtering!

\[
Q''^{t+1}[n] = \exp(-U[n] - BFilt(Q^t[n]; I[n], \sigma_s^2, \sigma_i^2) \times V^T)
\]
MRF INFERENCE: MEAN FIELD ALGORITHM

\[ \begin{align*}
P(\{x_i \in V\}) &= \frac{1}{Z} \exp \left( - \sum_{x_i \in V} E_{ii}(x_i) - \sum_{(x_i, x_j) \in E} E_{ij}(x_i, x_j) \right), \quad E_{ij}(x_i, x_j) = \mu_{ij} V(x_i, x_j) \\
Q'^{t+1}(x_i) &= \exp \left( -E_{ii}(x_i) - \sum_{l' \in L} V(x_i, l') \sum_{j: (i, j) \in E} Q^t_j(l') \mu_{ij} \right) \\
\mu_{ij} &= \exp \left( - \frac{|n_i - n_j|^2}{2\sigma_s^2} - \frac{||I[n_i] - I[n_j]|^2}{2\sigma_i^2} \right) \\
Q'^{t+1}[n] &= \exp \left( -U[n] - \text{BFilter}(Q'[n]; I[n], \sigma_s^2, \sigma_i^2) \right) \times V^T
\end{align*} \]

- In the fully connected MRF case with "bilateral" weights, there are efficient data-structures you can use to do the bilateral filtering.

Krahenbuhl and Koltun, "Efficient Inference in Fully Connected CRFs with Gaussian Edge Potentials," NIPS 2011.

- These fully-connected MRFs are used in the successful DeepLab image segmentation method from Google!
Let's consider the binary case where each $x_i \in \{\alpha, \beta\}$.

We're going to convert this into a min-cut problem on a weighted graph $G$. 

\[
X = \arg \max_X P(X) = \arg \max_X \exp \left( - \sum_i E_{ii}(x_i) - \sum_{(i,j)} E_{ij}(x_i, x_j) \right)
\]

\[
X = \arg \min_X \sum_i E_{ii}(x_i) + \sum_{(i,j)} E_{ij}(x_i, x_j)
\]
{p, q, ... s} = \arg \min \sum_i E_{ii}(x_i) + \sum_{(i,j)} E_{ij}(x_i, x_j)

- Create a node for each variable. Also create nodes for the two possible labels \( \alpha, \beta \).
- Copy over all edges from the MRF. Also add an edge from each variable to \( \alpha \) and to \( \beta \).
Without loss of generality, set \( E_{ij}(x_i, x_j) = e_{i,j} \delta(x_i \neq x_j) \) such that \( e_{i,j} \) is positive.

- Can do this because there are only two labels: remaining to the unary terms.

\[
E_{ij}(x_i, x_j) = C(x_i = \alpha) + C(x_i = \beta) + e_{i,j} \delta(x_i \neq x_j) + C
\]

- Express \( E_{ii}(x_i) = t_i^\alpha \delta(x_i \neq \alpha) + t_i^\beta \delta(x_i \neq \beta) \)

- Make sure these terms are positive (by adding a constant if necessary).
MRF INFERENCE: GRAPH CUTS

Treat these as weights on the edge.

Now solve the min-cut between $\alpha$ and $\beta$ in this graph: Delete edges to separate $\alpha$ and $\beta$ so that sum of weights on deleted edges is minimized.

From the cut graph, assign variables based on whether they remain connected to $\alpha$ or $\beta$.

This corresponds to minimizing the original MAP energy.

$$\{p, q, \ldots s\} = \arg \min \sum_{i} t_i^\alpha \delta(x_i \neq \alpha) + \sum_{i} t_i^\beta \delta(x_i \neq \beta) + \sum_{(i,j)} e_{i,j} \delta(x_i \neq x_j)$$
MRF INFERENCE: GRAPH CUTS

Generalization to Multi-labels

• NP Hard, but approximate iterative algorithms that calls the binary graph-cut solver multiple times.
• At each iteration consider two kinds of moves.
  • \( \alpha, \beta \) swap
    ▪ For all pairs of labels \( \alpha, \beta \)
      ○ Consider the subset of nodes which have labels \( \alpha \) or \( \beta \).
      ○ Solve a binary cut to figure out whether to swap labels for some pixels in this sub-set.
      ○ See if this decreases the original energy. If so, keep, otherwise stick with original.
  • \( \alpha \) expansion
    ▪ For all labels \( \alpha \)
      ○ Solve a binary cut to figure out whether to set some pixels that are not \( \alpha \) to \( \alpha \).
      ○ See if this decreases the original energy. If so, keep, otherwise stick with original.
• Requires conditions on \( E_{ij}(x_i, x_j) \):

\[
E_{ij}(p, q) = 0 \iff p = q; E_{ij}(p, q) = E_{ij}(q, p) \geq 0; E_{ij}(p, q) \leq E_{ij}(p, q) + E_{ij}(q, r)
\]
• Satisfies all three: metric. Satisfies only the last two: semi-metric.
MRF INFERENCE: GRAPH CUTS

Generalization to Multi-labels

- \(\alpha, \beta\) swap move.

Given a current labeling, and a candidate pair of \(\alpha\) and \(\beta\) to swap:

- Consider the set of variables \(V_\alpha\) and \(V_\beta\) that currently have these labels.
- Consider the union of these two sets, and set up a binary segmentation problem on these variables.
- Edge weights remain the same. But change the unary entries of the node as

\[
E'_{ii}(x_i = \alpha) = E_{ii}(x_i = \alpha) + \sum_{(i,j) \in E, x_j \notin V_\alpha \cup V_\beta} E_{ij}(\alpha, x_j)
\]

\[
E'_{ii}(x_i = \beta) = E_{ii}(x_i = \beta) + \sum_{(i,j) \in E, x_j \notin V_\alpha \cup V_\beta} E_{ij}(\beta, x_j)
\]

where \(x_j\) is the current value of the variable (which is not \(\alpha, \beta\))

- Solve a graph-cut to re-label variables in \(V_\alpha, V_\beta\) to either \(\alpha\) or \(\beta\).