GENERAL

- Homework reviews due Thursday! (No extensions)
- There was a slight bug in the mean-field derivation for potentials last time (version in posted slides is correct).

\[
Q^{t+1}_i(x_i) = \Psi_i(x_i) \prod_{j \in \mathcal{N}_i} \exp \left( \sum_{k \in \mathcal{V}_i} Q^{t}_j(x_j) \log \psi_{ij}(x_i, x_j) \right)
\]

and NOT

\[
Q^{t+1}_i(x_i) = \Psi_i(x_i) \prod_{j \in \mathcal{N}_i} \left( \sum_{x_j} Q^{t}_j(x_j) \psi_{ij}(x_i, x_j) \right)
\]

- This is consistent with the ‘energy’-based version that we finally used for efficient computation.
- So, message passing, but computing ‘expectations’ in the log-space. \(Q()\) aren’t “true marginals”, but computed so that their product best approximates the joint distribution.

MRF INFERENCE: GRAPH CUTS

- Last time, talked about find MAP estimate using graph-cuts.
- Exact solution in polynomial time for binary labels.
- Talked about approximate inference for multi-labels.
- Discussed ‘swap’ moves.

MRF INFERENCE: GRAPH CUTS

Generalization to Multi-labels

\[
X = \arg \min \sum_i E_i(x_i) + \sum_{(i,j)} E_{ij}(x_i, x_j)
\]

- \(\alpha \) Expansion Move: Given a current labeling \(\{x'_i\}\) and a possible label \(\alpha\) to “expand”
- Assumes \(E_{ij}\) is a metric.

- Construct a new graph with all variables, but only two labels \(\alpha\) and \(\bar{\alpha}\) (not \(\alpha\))
- At the end of the cut, we will set variables \(\alpha\) to \(\alpha\) and \(\bar{\alpha}\) to their value from previous iteration.
- If a variable \(x_i\) is not currently \(\alpha\), set \(E'_u(\alpha) = E_u(\alpha), E'_u(\bar{\alpha}) = E_u(x'_i)\)
- If a variable \(x_i\) is currently \(\alpha\), set \(E'_u(\alpha) = 0, E'_u(\bar{\alpha}) = \infty\)
  - This is an expansion, we won’t change the label of something that’s already \(\alpha\) to \(\bar{\alpha}\).
- If two neighbors \(x_i, x_j\) currently have the same label \(\gamma\) (including possibly \(\alpha\)), then set \(E_{ij}(\alpha, \bar{\alpha}) = E_{ij}(\gamma, \alpha)\)
- If two neighbors \(x_i, x_j\) do NOT currently have the same label (including if one of them is \(\alpha\)), create a new node \(a\), and replace the edge between \(i\) and \(j\) with two edges between \((i, a), (b, j)\).

\[
E'_{ia} = E_{ij}(x'_i, \alpha), E'_{ia} = E_{ij}(x'_j, \alpha)
\]
and

\[
E'_{ba} = E_{ij}(x'_j, \alpha), E'_{ba} = E_{ij}(x'_i, \alpha)
\]
MRF INFERENCE: GRAPH CUTS

Generalization to Multi-labels

\[ X = \arg \min \sum_i E_d(x_i) + \sum_{i,j} E_g(x_i, x_j) \]

- \( \alpha \) Expansion Move: Given a current labeling \( \{x_i^j\} \) and a possible label \( \alpha \) to "expand". Metric \( E_d \).

  if \( x_i^j \neq \alpha \), \( E_d^\alpha(\alpha) = E_d(\alpha), E_d^\alpha(\bar{\alpha}) = E_d(x_i^j) \)
  if \( x_i^j = \alpha \), \( E_d^\alpha(\alpha) = 0, E_d^\alpha(\bar{\alpha}) = \infty \)
  if \( x_i^j = x_i^\gamma \), \( E_d^\alpha(\alpha, \bar{\alpha}) = E_d(\gamma, \alpha) \)
  if \( x_i^j \neq x_i^\gamma \), \( E_d^\alpha(\alpha, \bar{\alpha}) = E_d(x_i^j, \alpha), E_d^\alpha(\alpha, \bar{\alpha}) = E_d(x_i^\gamma) \), \( E_d^\alpha(\alpha) = 0 \)

- Graph cut on this can’t swap to \( \bar{\alpha} \).
  - If I swap, I’m paying a unary cost differential (from the original energy) of \( E_d(x_i^\gamma, \alpha) \) to \( E_d(\alpha) \).
  - If two neighbors are the same and equal to \( \alpha \), no question of swapping.
  - If two neighbors are the same, and I swap one to \( \alpha \), I’m paying the cost of \( E_d(\gamma, \alpha) \). (Note \( E_d(\gamma, \gamma) = 0 \).)
  - If two neighbors are the same, and I swap neither, no added pairwise cost.

MRF INFERENCE: GRAPH CUTS

Generalization to Multi-labels

\[ X = \arg \min \sum_i E_d(x_i) + \sum_{i,j} E_g(x_i, x_j) \]

- \( \alpha \) Expansion Move: Given a current labeling \( \{x_i^j\} \) and a possible label \( \alpha \) to "expand". Metric \( E_g \).

  if \( x_i^j \neq \alpha \), \( E_g^\bar{\alpha}(\alpha) = E_g(\alpha), E_g^\bar{\alpha}(\bar{\alpha}) = E_g(x_i^j) \)
  if \( x_i^j = \alpha \), \( E_g^\bar{\alpha}(\alpha) = 0, E_g^\bar{\alpha}(\bar{\alpha}) = \infty \)
  if \( x_i^j = x_i^\gamma \), \( E_g^\bar{\alpha}(\alpha, \bar{\alpha}) = E_g(\gamma, \alpha) \)
  if \( x_i^j \neq x_i^\gamma \), \( E_g^\bar{\alpha}(\alpha, \bar{\alpha}) = E_g(x_i^j, \alpha), E_g^\bar{\alpha}(\alpha, \bar{\alpha}) = E_g(x_i^\gamma) \), \( E_g^\bar{\alpha}(\alpha) = 0 \)

If two neighbors are not the same: Case 1b: Neither originally \( \alpha \) and I swap neither.

- Since (cost \( a = \bar{\alpha} \)) \( E_d(x_i^\gamma, \gamma) \leq E_d(x_i^\gamma, \alpha) + E_d(\alpha, x_i^\gamma) \) (cost \( a = \alpha \)), it makes sense to set \( a = \bar{\alpha} \).
  - No additional cost.

If two neighbors are not the same: Case 1a: One originally \( \alpha \) (and I don’t swap the other).

- \( a = \alpha \rightarrow E_g^\alpha = E_g(\beta, \alpha) \), \( \alpha = \bar{\alpha} \rightarrow E_g^\bar{\alpha} = E_g(\beta, \alpha) + E_g(\beta, \alpha) \)

So set \( a = \alpha \), incur no additional cost from original labeling.
If two neighbors are not the same: Case 2b: Neither originally $(\alpha, \beta)$, $(\gamma, \delta)$.

- If I swap both to $\alpha$, it makes sense to also set $\alpha = a$. Gain / subtract the cost $E_y(\alpha, \gamma')$.

**MRF INFERENCE: GETTING DIVERSE SOLUTIONS**

$$X = \arg \max P(X)$$

- We’re extracting the most likely labeling from a probability distribution.
- But the probability distribution can suggest other labelings of $X$ that are just as likely (perhaps with only a very slightly lower probability).
- The most likely solution may not be the best:
  - Modeling error (sub-optimal unaries)
  - Bayes error (there are in fact many plausible solutions)
  - Optimization error (our algorithm finds a sub-optimal minima)
- So in some cases, we would like to get multiple solutions for $X$:
  - Interactive Segmentation: show different likely solutions to user, let them pick.
  - Stereo: get different likely depths from each frame of a video stereo camera, and resolve across frames.
  - Use a second classifier on top to identify the best solution.

**MRF INFERENCE: GRAPH CUTS**

Generalization to Multi-labels

$$X = \arg \min \sum_i E_i(x_i) + \sum_{i \neq j} E_{ij}(x_i, x_j)$$

- $\alpha$ Expansion Move: Given a current labeling $\{x_i^j\}$ and a possible label $\alpha$ to “expand”. Metric $E_y$.
  - if $x_i^j \neq \alpha$, $E_y(\alpha) = E_y(x_i^j, \alpha), \ E_{y, a} = E_y(x_i^j, \alpha), \ E_{a, a} = 0$
  - if $x_i^j = \alpha$, $E_y(\alpha) = 0, E_{y, a}(\alpha) = \infty$
  - if $x_i^j = x_i^j = \gamma$, $E_{y, a}(\gamma, \alpha) = E_y(\gamma, \alpha)$
- If two neighbors are not the same: Case 2b: Neither originally $\beta(\alpha, \beta), \gamma(\delta, \gamma)$

**MRF INFERENCE: GRAPH CUTS**

Generalization to Multi-labels

$$X = \arg \min \sum_i E_i(x_i) + \sum_{i \neq j} E_{ij}(x_i, x_j)$$

- $\alpha$ Expansion Move: Given a current labeling $\{x_i^j\}$ and a possible label $\alpha$ to “expand”. Metric $E_y$.
  - if $x_i^j \neq \alpha$, $E_y(\alpha) = E_y(x_i^j, \alpha), \ E_{y, a} = E_y(x_i^j, \alpha), \ E_{a, a} = 0$
  - if $x_i^j = \alpha$, $E_y(\alpha) = 0, E_{y, a}(\alpha) = \infty$
  - if $x_i^j = x_i^j = \gamma$, $E_{y, a}(\gamma, \alpha) = E_y(\gamma, \alpha)$
- If two neighbors are not the same: Case 2b: Neither originally $\beta(\alpha, \beta), \gamma(\delta, \gamma)$

- if I swap one, say $x_i^j$ to $\alpha$ and keep the other $x_i$ to $\beta = \delta$
  - $a = \alpha \rightarrow E_y(\beta, \alpha), \ a = \beta \rightarrow E_y(\alpha, \beta) + E_y(\beta, \delta)$
  - So set $a = \alpha$ again. Subtract $E_y(x_i^j, \gamma')$ but add $E_y(\alpha, x_j^j)$.

So we’re saying that given the MAP solution $X$, we want to find a different $X'$ that also has high $P(X')$.

- What does different mean? If say that $X' \neq X$, there’s probably a solution very close to $X$ (change one pixel’s label) that has pretty much the same value of probability.
- So this means we want to have a measure of difference or “diversity” $\Delta(X, X')$ and say that we want $\Delta(X, X') > C$.

Given $X_0$, now find $X_1$:

$$X_1 = \arg \min \sum_i E_u(x_i) + \sum_{ij} E_q(x_i, x_j)$$

such that $\Delta(X_1, X_0) > C$.

Typical diversity measures are sums over pixels: $\Delta(X, X') = \sum \delta(x_i, x'_i)$

Can be as simple as $\delta(x_i, x'_i) = 0$ if both equal, and 1 otherwise.

### Extension to $M$-modes:

Given $X_0, X_1, \ldots, X_{M-1}$, now find $X_M$:

$$X_M = \arg \min \sum_i E_u(x_i) + \sum_{ij} E_q(x_i, x_j)$$

such that $\sum \delta(x_m, x_{m+1}) > C, \forall m \in \{0, 1, \ldots, M-1\}$.

$X(\lambda_0, \lambda_1, \ldots, \lambda_{M-1}) = \arg \min \sum_i (E_u(x_i) - \lambda \delta(x_i, x_{0})) + \sum_{ij} E_q(x_i, x_j)$

- Now an $M$-dimensional search over the $\lambda$s.
- But again, in practice, you select on set of values based on a validation set.
Consider the problem of semantic segmentation. $I[n]$: RGB $\Rightarrow L[n] \in C$ per-pixel label map.

We train a CNN to give us a per-pixel probability distribution over the $C$ classes.

$$I[n] \Rightarrow \text{CNN} \rightarrow U[n] \rightarrow \text{Per-pixel-softmax} \rightarrow \Psi[n]$$

- $\Psi[n]$ is a $C$-dimensional vector interpreted as class probabilities for pixel $n$.
- $U[n]$ is their log (+ a constant because it’s un-normalized)
- Remember, $\text{SoftMax}(U) = \frac{\exp(U)}{\sum \exp(U)}$
- Often, just selecting $L[n] = \arg \max \Psi[n]$ doesn’t do well.

Add an MRF for smoothing:

$$L = \arg \min \sum_n -U[n](l_n) + \sum_{ij} E_{ij}(l_i, l_j)$$

Here $U[n](l_n)$ is the entry in $U[n]$ for label $l_n \in C$.

---

**CNNS + MRF**

- Consider the case where $E_{ij}(l_i, l'_j) = \mu(l_i, l'_j)k(-(p_i - p_j))$
- Remember how we said we can use mean-field inference to solve this.

---

**CNNS + MRF**

- Input $I$ which is $H \times W \times 3$
- $I \rightarrow \text{CNN} \rightarrow U$ which is $H \times W \times C$
- $Q_0 = \exp(U)$ followed by normalization
CNNS + MRF

\[ L = \arg \min \sum_n -U[n](l_n) + \sum_{ij} E_g(l_i, l_j) \]

- Input \( I \) which is \( H \times W \times 3 \)
- \( I \to \text{CNN} \to U \) which is \( H \times W \times C \)
- \( Q^0 = \text{SoftMax} (U) \)
- For \( t = 0 : T \)
  \[ Q^{t+1}[n] \propto \exp \{ U[n] - (Q^t \ast k)[n] \times \mu^T \} \]
- Select \( L[n] = \arg \max Q^{T+1}[n] \)

One option is to train a CNN with softmax+cross-entropy loss between \( U \) and GT labels.
At inference time, run the CNN, and then solve MRF with hand-chosen \( \mu \) and \( k \)

But this may be sub-optimal.
- During training, the CNN isn’t aware of the fact that its outputs will be smoothed.
- Hand-chosen \( \mu \) and \( k \) may not be correct.

Instead, train end-to-end, by turning mean-field iterations into layers that share weights across iterations.
CNNS + MRF

\[ L = \arg \min \sum_{n} -U(n,i_o) + \sum_{ij} E_g(i,j) \]

- Input \( I \)
- \( U \leftarrow \text{CNN}(I) \)
- Add \( T \) layer groups with common learnable parameters \( \mu, k \)
  - \( Q \leftarrow \text{SoftMax}(U) \)
  - \( S \leftarrow \text{DepthWise_Conv}(Q,k) \)
  - \( A \leftarrow \text{Conv}_{1x1}(S,k) \)
  - \( U \leftarrow U + A \)
- Loss \( = \text{SoftMax-CrossEntropy}(U, L) \)

We are back-propagating through the mean-field layers to the multiple layers of the CNN.

- Common sets of weights for layers for all iterations: like an RNN.
- Can pick a lower value of \( T \) for training, and use many more iterations at test time.

STEREO

- In 559A, we talked about solving stereo with matching + pair-wise smoothness costs

\[ d[n] = \arg \min \sum_{n} M_n[d[n]] + \sum_{n_1, n_2} V_{n_1,n_2}(d[n_1], d[n_2]) \]

Here, \( M_n \) came from hamming-distance (was \( D \) numbers for each pixel). Hand chosen \( V \).
- We talked about SGM as a way of minimizing. But now you know of more energy minimization methods.
- But this is limiting. The only thing that pair-wise potentials can express is that two neighboring pixels should be close. That depth-discontinuities are rare.
- But we might want “higher-order” priors on disparity/depth maps.
- In particular, we might want to say that natural scenes are piece-wise planar. i.e., most groups of neighboring pixels lie on the same plane, with plane boundaries being rare.

Remember from 559, if a set of points \( \{(x,y)\} \in P \) lie on the same plane \( p \):

\[ d(x,y) = \alpha_p x + \beta_p y + \gamma_p \]

with common parameters \( \alpha_p, \beta_p, \gamma_p \).

STEREO

\[ d[n] = \arg \min \sum_{n} M_n[d[n]] + \sum_{n_1, n_2} V_{n_1,n_2}(d[n_1], d[n_2]) \]

\[ d(x,y) = \alpha_p x + \beta_p y + \gamma_p \]

- But it’s hard to relate a planarity constraint as a pair-wise MRF! You can fit any two (or three) points to a plane, so that doesn’t impose a constraint.

Possible Solution:

- Use a regular pair-wise cost to get an initial disparity map \( d[n] \).
- Segment the image into super-pixels. (Use SLIC on the left image).
  - Assumption is that the points within each super-pixel lie on the same plane.
Possible Solution:

- Use a regular pair-wise cost to get an initial disparity map $d[n]$.
- Segment the image into super-pixels. (Use SLIC on the left image).
  - Assumption is that the points within each super-pixel lie on the same plane.

- Inside each super-pixel, do a least-squares fit to the initial disparity values to get $\alpha$, $\beta$, $\gamma$
- Replace $d(x, y) = \alpha x + \beta y + \gamma$ based on the per-pixel estimated plane parameters.

Improved Solution:

- Define an MRF over super-pixels instead of individual pixels.
- Form cliques that contain pairs of adjoining super-pixels.
- Reason about the pair as:
  - Both being co-planar.
  - Not co-planar, but having a hinge (i.e., no discontinuity at the boundary)
  - Discontinuity at boundary: and which one "occludes" the other.