Message-passing algorithms (continued)

Nikos Komodakis
Ecole des Ponts ParisTech, LIGM

Traitement de l’information et vision artificielle
Graphs with loops

- We saw that Belief Propagation can exactly optimize MRFs that have tree-structured graphs.

- But what if the MRF graph contains loops?
Graphs with loops

- We saw that Belief Propagation can exactly optimize MRFs that have tree-structured graphs.

- But what if the MRF graph contains loops?

- Well, we will pretend it is a tree and keep passing messages until convergence.

- Resulting algorithm called Loopy Belief Propagation.
Loopy belief propagation (LBP)

- Messages from node $p$ to $q$ form a set $\{m_{pq}(x_q)\}_{x_q \in \mathcal{L}}$ with:

$$m_{pq}(x_q) = \min_{x_p \in \mathcal{L}} \left\{ V_{pq}(x_p, x_q) + V_p(x_p) + \sum_{r: r \neq q, (r,p) \in \mathcal{E}} m_{rp}(x_p) \right\}$$

- Messages are circulated around the network until they stabilize (fixed-point)
Loopy belief propagation (LBP)

- After convergence, compute min-marginals for each node by summing up all incoming messages to that node (min-marginals also often called beliefs).

- To each node, assign the label whose min-marginal has the smallest value (even if the graph is a tree, is this guaranteed to give you the optimal labeling?)
Loopy belief propagation (LBP)

- Message-passing schedule
  - Parallel
  - Sequential

- No guarantee that LBP computes the optimum
  - In some cases, it may not even converge

- Pseudo min-marginals

- Empirically, it works well in many cases

- There are some theoretical guarantees
  - But these are very weak in general
Generalizations of BP

- **Note:** there exist more advanced versions of message-passing algorithms than loopy BP:

  - Require more background knowledge
  - Work better for loopy graphs
  - Have better theoretical properties
  - E.g., convergence is guaranteed
  - Provide suboptimality bounds
  - In some cases, they can even compute the global optimum (e.g., when there are no ties in pseudo min-marginals)
Loopy Belief Propagation (LBP)

- Relation between BP and ICM

  - ICM can also be though of as using messages
    - What information do the messages contain in this case?

  - However, ICM messages are much weaker (i.e., contain much less information) than BP messages
Speeding up message-passing
Motivation: object recognition

- We will see an application of BP to object recognition in images
- Can be used for recognizing “objects” such as faces or human bodies
Object recognition using MRFs

- **Nodes**: correspond to object parts
- **Labels of a node**: its (x,y) location in the image
- **Graph edges**: determine which object parts are related to each other
- **Unary potentials**: appearance of an object part
- **Pair-wise potentials**: geometric relationships between interrelated object parts
Object recognition using MRFs
Object recognition using MRFs

As many other problems, 2 phases are required:

- **Training**: learn the parameters which define the unary and pair-wise potentials
  - A training set is required for this

- **Optimization**: given a new image, find the object
Object recognition using MRFs
Object recognition using MRFs
Object recognition using MRFs

- More sophisticated models for object recognition can be used as well
- Models for articulated bodies
Object recognition using MRFs

- More sophisticated models for object recognition can be used as well
- Models for articulated bodies
Speeding BP for certain types of MRFs

- What is the number of labels per node in the object recognition example?

- How does this affect the running time of the BP algorithm?
Speeding-up BP for certain classes of MRFs

- For certain types of pair-wise potentials, messages can be computed in linear time (w.r.t. the number of labels) and not in quadratic time.

- Huge speed-up if number of labels is large.
Fast message updates

- Pairwise term V measuring label difference
- Sum product
  - Express as a convolution
  - $O(k \log k)$ algorithm using the FFT
  - Linear-time approximation algorithms for Gaussian models
- Min sum (max product)
  - Express as a min convolution
  - Linear time algorithms for common models using distance transforms and lower envelopes
Fast message updates for sum-product

- When $V(x_i, x_j) = \rho(x_i - x_j)$ can write message update as convolution
  
  $$m_{j \rightarrow i}(x_i) = \sum_{x_j} (\rho(x_j - x_i) \cdot h(x_j))$$

  $$= \rho \ast h$$

  - Where $h(x_j) = D_j(x_j) \prod_{k \in \mathcal{N}(j) \setminus i} m_{k \rightarrow j}(x_j)$

- Thus FFT can be used to compute in $O(k \log k)$ time for $k$ values
  - Still somewhat large constants

- For $\rho$ a (mixture of) Gaussian(s) do faster
Fast message updates for min-sum

- Can write message update as
  \[ m'_{j \rightarrow i}(x_i) = \min_{x_j} (\rho'(x_j-x_i) + h'(x_j)) \]
  - Where \( h'(x_j) = D'_j(x_j) + \sum_{k \in \mathcal{N}(j) \setminus i} m'_{k \rightarrow j}(x_j) \)

- Convolution-like operation over min,+ rather than \( \sum, \times \) [FH00,FHK03]
  - No general fast algorithm like FFT
  - Certain important special cases in linear time
Commonly used pair-wise costs

- Potts model $\rho'(x) = \begin{cases} 0 & \text{if } x=0 \\ d & \text{otherwise} \end{cases}$
- Linear model $\rho'(x) = c|x|$
- Quadratic model $\rho'(x) = cx^2$
- Truncated models
  - Truncated linear $\rho'(x) = \min(d,c|x|)$
  - Truncated quadratic $\rho'(x) = \min(d,cx^2)$
- Min convolution can be computed in linear time for any of these cost functions
Potts pair-wise potential

- Substituting in to min convolution
  \[
  m'_{j\rightarrow i}(x_i) = \min_{x_j}(\rho'(x_j-x_i) + h'(x_j))
  \]
  can be written as
  \[
  m'_{j\rightarrow i}(x_i) = \min(h'(x_i), \min_{x_j}h'(x_j)+d)
  \]
Potts pair-wise potential

- Substituting in to min convolution
  \[ m'_{j \rightarrow i}(x_i) = \min_{x_j} (\rho'(x_j - x_i) + h'(x_j)) \]
  can be written as
  \[ m'_{j \rightarrow i}(x_i) = \min(h'(x_i), \min_{x_j} h'(x_j) + d) \]

- No need to compare pairs \( x_i, x_j \)
  - Compute min over \( x_j \) once, then compare result with each \( x_i \)

- \( O(k) \) time for \( k \) labels
  - No special algorithm, just rewrite expression to obtain alternative (fast) computation
Linear pair-wise potential

- Substituting in to min convolution yields
  \[ m'_{j\rightarrow i}(x_i) = \min_{x_j}(c|x_j - x_i| + h'(x_j)) \]
Linear pair-wise potential

- Substituting in to min convolution yields
  \[ m'_{j \rightarrow i}(x_i) = \min_{x_j} (c|x_j-x_i| + h'(x_j)) \]
- Similar form to the \( L_1 \) distance transform
  \[ \min_{x_j} (|x_j-x_i| + 1(x_j)) \]
  - Where \( 1(x) = \begin{cases} 0 \text{ when } x \in P \\ \infty \text{ otherwise} \end{cases} \)
  is an indicator function for membership in \( P \)
- Distance transform measures \( L_1 \) distance to nearest point of \( P \)
  - Can think of computation as lower envelope of cones, one for each element of \( P \)
(Opening parenthesis on Distance Transforms)
Distance transforms

- Set of points, $P$, and measure of distance
  \[ DT(P)[x] = \min_{y \in P} \text{dist}(x,y) \]
- For each location $x$ distance to nearest point $y$ in $P$
  - Can think of “cones” rooted at each $y \in P$
  - Min over all the cones (lower envelope)
Using different distances

- Euclidean distance (L_2 norm)
  \[ \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + ...} \]
- City block distance (L_1 norm)
  \[ |x_1 - y_1| + |x_2 - y_2| + ... \]
- Chessboard distance (L_\infty norm)
  \[ \max(|x_1 - y_1|, |x_2 - y_2|, ...) \]
Grid formulation of distance transforms

- Commonly computed on a grid $\Gamma$, for set of points $P \subseteq \Gamma$
  \[
  DT(P)[x] = \min_{y \in \Gamma} (\text{dist}(x,y) + 1_P(y))
  \]
- Where $1_P(y)$ indicator function for $P$
  - Value of 0 when $y \in P$, $\infty$ otherwise
  - Can think of cone rooted at each point of grid, rather than of $P$
  - Cones not at points of $P$ are infinitely large so don’t figure into minimum
Naïve way of computing distance transforms

- For each point on the grid, explicitly consider each point of P and minimize
  - For n grid points and m points in P take time $O(mn)$
  - Note that m is $O(n)$, so $O(n^2)$ method

- Not very practical even for moderate size grids such as images
  - Even a low-resolution video frame has about 300K pixels
    - About 100 billion distance computations
L₁ Distance Transform (1D case)

- 1D case, L₁ norm: $|x₁ - y₁| + |x₂ - y₂|$
L₁ Distance Transform (1D case)

- 1D case, L₁ norm: \(|x₁ - y₁| + |x₂ - y₂|
  - Two passes:
    - Find closest point on left
    - Find closest on right if closer than one on left
**L₁ Distance Transform (1D case)**

- **1D case, L₁ norm:** $|x_1 - y_1| + |x_2 - y_2|$
  - Two passes:
    - Find closest point on left
    - Find closest on right if closer than one on left
  - Incremental:
    - Moving left-to-right, closest point on left either previous closest point or current point
    - Analogous for moving right-to-left
  - Can keep track of closest point as well as distance to it
    - Will illustrate distance only, less book-keeping
L₁ Distance Transform (1D case)

- Two pass O(n) algorithm for 1D L₁ norm (just distance and not source point)
  1. **Initialize**: For all j
     \[
     D[j] \leftarrow 1_p[j]
     \]
  2. **Forward**: For j from 1 up to n-1
     \[
     D[j] \leftarrow \min(D[j], D[j-1]+1)
     \]
  3. **Backward**: For j from n-2 down to 0
     \[
     D[j] \leftarrow \min(D[j], D[j+1]+1)
     \]
\( L_1 \) Distance Transform (2D case)

- 2D case analogous to 1D
  - Initialization
  - Forward and backward pass
    - Forward pass adds one to closest above and to left, takes min with self
    - Backward pass analogous below and to right
Generalization of distance transform

- DT of arbitrary functions: $\min_y \| x-y \| + f(y)$
  - Exact same algorithms apply
  - Combination of cost function $f(y)$ at each location and distance function

- This is exactly the form that the messages in BP have for the special class of pairwise potentials that we saw earlier
(Closing parenthesis on Distance Transforms)
Quadratic pairwise potentials

- Substituting in to min convolution yields
  \[ m'_{j \rightarrow i}(x_i) = \min_{x_j} (c(x_j-x_i)^2 + h'(x_j)) \]
- Again similar form to distance transform
- Compute lower envelope of parabolas
  - Each value of \( x_j \) defines a quadratic constraint, parabola rooted at \((x_j, h(x_j))\)
  - In general can be done in \(O(k \log k)\) [DG95]
  - Here parabolas are same shape and ordered, so \(O(k)\)
Combinations of pairwise potentials

- **Truncated models**
  - Compute un-truncated message $m'$
  - Truncate using Potts-like computation on $m'$ and original function $h'$
    $$\min(m'(x_i), \min_{x_j} h'(x_j) + d)$$

- **More general combinations**
  - Min of any constant number of linear and quadratic functions, with or without truncation
    - E.g., multiple “segments”