Computer Vision II

• Last time: Computer vision tasks as massive search problems

\[ f(x) = y^* = \arg\max_{y \in Y} g(x, y) \]
• Detection: $f: \mathcal{X} \to \mathcal{Y} \quad \mathcal{Y} = \text{all possible positions (and scales) of object}$

• Foreground/background segmentation: $f: \mathcal{X} \to \mathcal{Y} \quad \mathcal{Y} = \text{all possible 0/1 labelings of image } \{0,1\}^n$

• Labeling: $f: \mathcal{X} \to \mathcal{Y} \quad \mathcal{Y} = \text{all possible labelings of image } \{1, \ldots, L\}^n$

• Pose estimation: $f: \mathcal{X} \to \mathcal{Y} \quad \mathcal{Y} = \text{all possible poses } (u, v, \theta, s) \text{ of image } \{1, \ldots, P\}^K$

$$f(x) = y^* = \arg\max_{y \in \mathcal{Y}} g(x, y)$$

• Detection: $f: \mathcal{X} \to \mathcal{Y} \quad \mathcal{Y} = \text{all possible positions (and scales) of object}$

$$g(x, y) = w \cdot \varphi(x, y)$$

• Foreground/background segmentation: $f: \mathcal{X} \to \mathcal{Y} \quad \mathcal{Y} = \text{all possible 0/1 labelings of image } \{0,1\}^n$

• Labeling: $f: \mathcal{X} \to \mathcal{Y} \quad \mathcal{Y} = \text{all possible labelings of image } \{1, \ldots, L\}^n$

• Pose estimation: $f: \mathcal{X} \to \mathcal{Y} \quad \mathcal{Y} = \text{all possible poses } (u, v, \theta, s) \text{ of image } \{1, \ldots, P\}^K$

Energy models:

$$g(x, y) = \sum_{i=1}^{n} g_i(x, y_i) + \sum_{i,j \in N(i)} g_{i,j}(y_i, y_j, x)$$

Probabilistic models:

$$p(y|x) = \frac{1}{Z} \exp(g(x, y)) = \frac{1}{Z} \prod_{i} \psi_i(x, y_i) \prod_{i,j \text{ linked}} \psi_{ij}(y_i, y_j)$$
Reminder of key results

• Exact algorithms on tree-structured graphs
  – Message passing
  – Max-product: compute $y^*$
  – Sum-product: estimate marginals $p(y_i|x)$

• Today:
  – Details of max/sum for tree-structured models
    • Detection with parts
    • Pose estimation
  – Efficient search for detection (in some special cases)
• Next:
  – Details of general cases for segmentation and labeling

Richer description is needed to capture the variation in appearance of typical visual classes

\[ g(x, y) = w \cdot \varphi(x, y) \]
Representation by parts:
\[ y = y_1, \ldots, y_K \]

Possible (bad) model:
Find each part independently
\[ y_i^* = \arg\max_{y_i} g_i(x, y_i) \]
• For each individual part:

\[ g_i(x, y_i) = w_i \cdot \varphi(x, y_i) \]

Feature at \( y_i \) (e.g., HoG)

\[ w_1 \]

\[ w_2 \]

\[ w_3 \]

\[ w_4 \]

Likely

Unlikely
\[ g_{ij}(y_i, y_j) = w_{ij} \cdot \varphi_{ij}(y_i, y_j) \]

Feature vector describing the location of \( y_i \) with respect to \( y_j \)

For example: \( \varphi_{ij} = \begin{bmatrix} -(u_1 - u_2)^2 \\ -(v_1 - v_2)^2 \end{bmatrix} \)
\[ g(x, y) = w_1^T \cdot \varphi_1(x, y_1) + w_2^T \cdot \varphi_2(x, y_2) + w_3^T \cdot \varphi_3(x, y_3) + w_{12} \cdot \varphi_{12}(x, y_{12}) + w_{13} \cdot \varphi(x, y_{13}) \]

\[ g(x, y) = w_1^T \varphi_1(x, y_1) + w_2^T \varphi_2(x, y_2) + w_3^T \varphi_3(x, y_3) + w_{12} \varphi_{12}(y_1, y_2) + w_{13} \varphi_{13}(y_2, y_3) \]

\[
\max_y g(x, y) = \max_{y_1} \max_{y_2} \max_{y_3} w_1^T \varphi_1(x, y_1) + w_2^T \varphi_2(x, y_2) + w_3^T \varphi_3(x, y_3) + w_{12} \varphi_{12}(y_1, y_2) + w_{13} \varphi_{13}(y_1, y_3) \]

Usual max sum trick: \( \max(a+b, a+c) = a + \max(b,c) \)

\[
\max(w_1^T \varphi_1(x, y_1) + \max(w_2^T \varphi_2(x, y_2) + w_{12} \varphi_{12}(y_1, y_2)) + \max(w_3^T \varphi_3(x, y_3) + w_{13} \varphi_{13}(y_1, y_3)))
\]
General case

- Message passing (DP):
  \[
  \text{score}(y_j) = w_j^T \varphi_j(x, y_j) + \sum_{k \text{ descendant}(j)} m_k(y_j)
  \]

  \[
  m_a(y_b) = \max_{y_a} \text{score}(y_a) + w_{ab}^T \varphi_{ab}(y_a, y_b)
  \]
Estimating the marginals

\[ p(y|x) = \frac{1}{Z} \exp(g(x, y)) = \frac{1}{Z} \prod \psi_i(x, y_i) \prod_{i,j \text{ linked}} \psi_{ij}(y_i, y_j) \]

Estimating the marginals

- Propagate partial sums from leaves

\[ P(y_j|y_i, x) = P(y_j|y_i)m_j(y_j) \]

\[ m_j(y_j) \propto e^{w_j^T \varphi_j(x,y_j)} \prod_{k \text{ child of } j} \sum_{y_k} P(y_k|y_j, x) \]
Estimating the marginals

- Propagate partial sums from root

\[ P(y_j, y_i|x) = P(y_j|y_i, x)P(y_i|x) \]

\[ P(y_j|x) = \sum_{y_i} P(y_j, y_i|x) \]

Example from Deva Ramanan
Parenthesis: Efficiency issues in search for a detection in an image

- Detection: \( f: \mathcal{X} \rightarrow \mathcal{Y} \) \( \mathcal{Y} = \) all possible positions (and scales) of object \( g(x, y) = w \cdot \varphi(x, y) \)
- Need to evaluate all possible boxes = all possible positions and sizes = \( N^4 \)
- Is it possible to do this more efficiently?
- Yes, for some special cases

Branch-and-bound

- if \( \overline{g}_k < \underline{g}_l \) for all \( l \) then there is no point in exploring \( A_k \)
Branch-and-bound

- if \( \overline{g_k} < \overline{g_l} \) for all \( l \) then there is no point in exploring \( A_k \)
- if \( g(\{y\}) = g(\{y\}) = g(\{y\}) \) for all \( y \)
- Notes:
  - Worst case complexity remains the same
  - Lower is trivial: Pick any \( a \) in \( A_k \)

Branch-and-bound for detection

- \( A \) = set of boxes
- Each box parameterized by \([T, B, L, R]\)
- Each set \( A \) parameterized by \( T_{\text{min}}, T_{\text{max}}, B_{\text{min}}, B_{\text{max}}, L_{\text{min}}, L_{\text{max}}, R_{\text{min}}, R_{\text{max}} \)

One example box \( y \)
Branch-and-bound for detection

\[ y_n = \bigcap_{y \in A} y \]

\[ y_0 = \bigcup_{y \in A} y \]

Subwindow search

- Depth first search: split current set of windows \( A \) by splitting one of the intervals \([T_{\min}, T_{\max}], [B_{\min}, B_{\max}], [L_{\min}, L_{\max}], [R_{\min}, R_{\max}]\) in the middle
- Next question: How to evaluate \( g(A), g(A) \)

\[ R'_{\max} = \frac{R_{\min} + R_{\max}}{2} \]

\[ L'_{\min} = \frac{L_{\min} + L_{\max}}{2} \]
Additive scores

• We consider scores of the form:

\[ g(x, y) = \sum_{x_i \text{ occurs in } y} w(x_i) \]

Example BoW:

• \( h(x_i) \) = number of times word \( x_i \) occurs in box \( y \)
• \( w(x_i) \) = entry of weight vector \( w \) for word \( x_i \)

\[ g(x, y) = \sum_{x_i} w(x_i)h(x_i) = \sum_{x_i \text{ occurs in } y} w(x_i) \]

Branch-and-bound for detection

• If score is additive, then simple upper bound:

\[ \bar{g}(x, y) = \sum_{x_i \text{ occurs in } y} \max(w(x_i), 0) + \sum_{x_i \text{ occurs in } y} \min(w(x_i), 0) \]

\[ \bar{g}(x, y) = \sum_{x_i \text{ occurs in } y} w(x_i)^+ + \sum_{x_i \text{ occurs in } y} w(x_i)^- \]
Efficient sliding windows (ESS)

• Branch:
  – Depth first search: split current set of windows A by
    splitting one of the intervals
    \([T_{min}, T_{max}], [B_{min}, B_{max}], [L_{min}, L_{max}], [R_{min}, R_{max}]\)
    in the middle

• Bound:
  \(\bar{g}(x, y) = \sum_{x_i \text{ occurs in } y_u} w(x_i)^+ + \sum_{x_i \text{ occurs in } y_n} w(x_i)^-\)

Efficient sliding windows (ESS)

• Branch:
  – Depth first search: split current set of windows A by splitting
    one of the intervals
    \([T_{min}, T_{max}], [B_{min}, B_{max}], [L_{min}, L_{max}], [R_{min}, R_{max}]\) in the middle

• Bound:
  \(\bar{g}(x, y) = \sum_{x_i \text{ occurs in } y_u} w(x_i)^+ + \sum_{x_i \text{ occurs in } y_n} w(x_i)^-\)

• \(O(1)\) computation of bound

• Can be extended to non-linear operations on histogram
  representations: histogram intersection, \(\chi^2\), pyramid
  kernels

• Later: BB idea can be applied to other problems, e.g., non-tree
  inference
Reminder of key results

• Exact algorithms on tree-structured graphs
  – Message passing
  – Max-product: compute $y^*$
  – Sum-product: estimate marginals $p(y_i|x)$

• Today:
  – Details of max/sum for tree-structured models
    • Detection with parts
    • Pose estimation
  – Efficient search for detection (in some special cases)

• Next:
  – Details of general cases for segmentation and labeling