Today: latent variables

**SVMstruct** + CCCP

**Deep learning**

**Latent variables**

Motivation: semantic segmentation \(\rightarrow\) find boundary of different objects

Segmentation is expensive \(\rightarrow\) \(Z\) "latent variable"

Perhaps only have class labels \(\rightarrow\) \(y\)

Also: [Felzenszwalb & al TPAMI 2010]
"deformable part models" for object recognition

\(Z\) there were many part configurations

Before, we had \(s(x, y; w) = \langle w, y(x, y) \rangle\)

Now, consider \(s(x, y, z; w) = \langle w, y(x, y, z) \rangle\)

As before, could predict with argmax \(s(x, y, z; w)\)

\(y^{*}, z^{*} \in Z\)

\(\bigcirc\) CRF \((p(y|x))\) \(\rightarrow\) hidden CRF \(p(y|x)\)

Similar to latent variable modeling with graph model

ML \(\rightarrow\) EM (expectation-maximization)

No analog for latent SVMstruct is CCCP

\[ l(y, (y^{*}, z^{*})) \]

Generalize structured hinge loss:

\[ \ell(x, y; w) = \max_{\tilde{y}, \tilde{z}} -\langle w, y(x, y, z) \rangle + \ell(y, (y^{*}, z^{*})) \]

\[ \text{max}_{w} \langle w, y(x, y, z) \rangle \geq \ell(y, (y^{*}, z^{*})) \]

\((1.3)\)
\[ f(x, y, \omega) = \max_{\tilde{y}, \tilde{z}} <w, y \tilde{x}, y \tilde{z}> + \ell(y, \tilde{y}, \tilde{z}) \]

\[ - \min_{\tilde{y}, \tilde{z}} <w, y \tilde{x}, y \tilde{z}> + \ell(y, \tilde{y}, \tilde{z}) \]

Here \( g(x, y, \omega) = u(w) - v(w) \) where \( u \) and \( v \) are convex f. of \( w \)

"difference of convex functions"

\[ \rightarrow \text{CCCP procedure is to approximate minimize this} \]

**CCCP procedure:**

- Linearize \( v(w) \) at \( w \) to get an upper-bound.
- \( w_{k+1} \) is obtained by minimizing this upper-bound.
- Repeat a majorization-minimization procedure (EM is an example)
  
  \[ f_t(w) = u(w) - \left[ v(w_{k}) + (v(w_{k}))-v(w), w-w_{k} \right] \geq g(w) \quad \forall w \]
  
  (for subgradient) \[ f_{t}(w_{k}) = g(w_{k}) \]

**Properties of procedure:**

- Like EM, descent procedure i.e. \( g(w_{k+1}) \leq g(w_{k}) \)

\[ g_{t}(w_{k}) = f_{t}(w_{k}) \geq f_{t}(w_{k+1}) \geq g(w_{k}) \]

- Local linear convergence to a stationary point [see references for latent sums-net]

\[ \ast \text{for sums-net:} \quad v(w) = \min_{z} <w, (x, y, z)> + \ell_{z}(x, y, z) \]

\[ \hat{v}(w_{k}) = \ell(y, \hat{y}, \hat{z}(x, y, w_{k})) \]

\[ \Rightarrow f_{t}(w_{k}) = \max_{z} <w, (x, y, z)> + \ell(y, \hat{y}, \hat{z} > + \ell_{z}(x, y, w_{k}) \]

\[ \rightarrow \text{like sums-net objective} \]

**CCCP algorithm for latent sums-net:**

repeat:

\[ \text{find } \omega \in \tilde{z}^{(i)} \text{ for all ground truth } y^{(i)} \text{ using } w_{k} \]
Deep learning

I) "deep learning" features in a structural prediction model

Example: OCR

\[ \text{OCR} \]

so far \( l_t(x_t, y^t) = \begin{cases} 0 & \text{if } y_t = y^t \\ 1 & \text{otherwise} \end{cases} \)

instead \( l_t(x_t, y^t) = \) \text{(learned an image-net e.g.)}

\( \text{Example: } [\text{Vu et al. ICCV 2015}]: \) "content-aware cuts for person head detection"

II) "end-to-end" training: \text{structured prediction energy networks (SPENs)}

III) \text{recurrent neural networks (RNN)}

\text{motivation: } p(y|x) = \prod_{t=1}^{T} p(y_t|y_{t-1}, x_t)

\text{graphical model approach: } \prod_{t=1}^{T} p(y_t|y_{t-1}, x_t)

\text{structured parameterization of } p(y_t|y_{t-1}, x_t)

\begin{align*}
\text{using NN: } & \quad h_{t+1} = f(h_t, x_t, y_t, w) \\
& \quad h_t = f(f(\ldots, y_1, \ldots, y_{t-1}, x_t, y_{t-1}, w))
\end{align*}

define \( p(y_t|y_{t-1}, x_t) \propto \exp(c(y_t)^T W h_t) \) e.g., \( p(y_t|y_{t-1}, x_t) \propto \exp(c(y_t)^T W h_t) \) given by \( c(y_t) \)

\text{Standard learning: use ML,
Standard learning: use ML.
\[
\text{I.e. } \min_{\mathbf{W}} \frac{1}{N} \sum_{i=1}^{N} \log \frac{p(y_i^t | x_i^{(t)})}{\hat{p}(y_i^t | y_{i:t-1}^1, x_i^{(t)})} \geq \log \frac{p(y_i^t | y_{i:t-1}^1, x_i^{(t)})}{\hat{p}(y_i^t | y_{i:t-1}^1, x_i^{(t)})} \quad \text{output of a deep NN}
\]

For ML, do SGD objective
\[
\text{gradient of } \log \frac{p(y_i^t | y_{i:t-1}^1, x_i^{(t)})}{\hat{p}(y_i^t | y_{i:t-1}^1, x_i^{(t)})} \rightarrow \text{use backpropagation}
\]

Decoding: \( \arg \max_{y^t \in Y} \sum_{i=1}^{N} \log p(y_i^t | y_{i:t-1}^1, x_i^{(t)}) \) \( \rightarrow \) use hard\( \hat{y}_t^t \) (decider).

Need approximation \( \rightarrow \) greedy decoding \( \hat{y}_t^t = \arg \max_{y_t \in \mathbb{Y}_t} \sum_{i=1}^{N} \log p(y_i^t | y_{i:t-1}^1, x_i^{(t)}) \) beam search (greedy decoding with memory of size \( k \)).

Beam search: construct \( \hat{y}_t^t, \ldots, \hat{y}_t^L \) beam of size \( L \) (memory)

- at step \( t \), you have \( L \) candidate solution prefixes \( \hat{y}_t^1, \ldots, \hat{y}_t^L \)

- expand possible next choice: \( \hat{y}_t^t \rightarrow \hat{y}_t^{t+1} \) \( \vdots \)

- score them (e.g. \( \log p(y_t^t | \hat{y}_t^t) + \log p(y_t^{t+1} | \hat{y}_t^{t+1}) \))

- then keep \( \frac{L}{k} \) candidates as \( \hat{y}_t^{(k)}_{t+1} \) \( k \in \{1, \ldots, L \} \)

Viterbi alg. which does "backtracking" to correct past mistakes

seq2seq decoder architecture

L = useful way to get \( p(y_t^t | y_{i:t-1}^1, x_i^{(t)}) \) for a NN
Issues:

a) Variable length output?

b) Long input sequence $x$?

Problem: Need to summarize input sentence in context of fixed length.

Solution: "Attention mechanism"

c) Vanishing gradient?

- LSTM
- Gated recurrent unit (GRU)
- etc..