Lecture 22 - latent SVMstruct; RNN

Tuesday, April 9, 2024 9:30 AM

today: • latent variable SVMstruct ~ CCCP
• deep learning ~ RNN

latent variables

motivation: semantic segmentation

segmentation is expensive → z "latent variable"

perhaps only have class labels → y

also: [Felzenszwalb et al TPAMI 2010
"deformable parts models" for object recognition
↳ z there was an object part configuration

before, we had $s(x,y;w) = \langle w, \varphi(x,y)\rangle$

now, consider $s(x,y,z;w) = \langle w, \varphi(x,y,z)\rangle$

as before, could predict with $\arg\max_{y\in\mathcal{Y}, z\in\mathcal{Z}} s(x,y,z;w)$

learning — CRF approach $p(y|x)$ —generalize→ hidden CRF $p(y,z|x)$

similarly to latent variable model with graph. model → marginalize z out

ML → EM (expectation-maximization)

SVMstruct (surrogate loss minimization)

analog for latent SVMstruct is CCCP (majorization minimization procedure)

latent SVMstruct

$\ell(y,(\tilde{y},\tilde{z}))$

generalized structured hinge loss

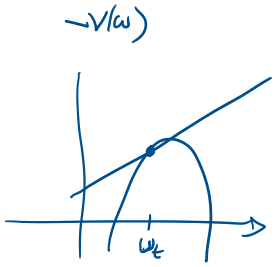
$$\xi(x,y,w) \geq \max_{\tilde{y},\tilde{z}} \langle w, \varphi(x,\tilde{y},\tilde{z})\rangle + \ell(y,(\tilde{y},\tilde{z})) \;\underbrace{-\max_{z'\in\mathcal{Z}} \langle w, \varphi(x,y,z')\rangle}_{\text{best score for ground truth}} \geq \ell(y,h_w(x))$$

$\geq u(w)$ $\quad$ $\geq v(w)$ $\quad$ $(\tilde{y},\tilde{z})$

here, $\xi(x,y,w) = u(w) - v(w)$ where u & v are convex fct. of w

"difference of convex functions"

↳ CCCP procedure is to approx. minimize this



CCCP procedure:

- linearize $v(w)$ at w_t to get an upper bound on $-v(w)$
- w_{t+1} is obtained by minimizing the upperbound $u(w)$ + linearized piece (w)
- repeat ↳ a majorization-minimization procedure (EM is another example)

$$\begin{cases} g_t(w) = u(w) - [v(w_t) + \langle \nabla v(w_t), w - w_t \rangle] \geq g(w) \quad \forall w \\ w_{t+1} = \operatorname{argmin}_w g_t(w) \end{cases}$$

(∇ or subgradient) and $g_t(w_t) = g(w_t)$

properties of procedure: • like EM, descent procedure i.e. $g(w_{t+1}) \leq g(w_t)$

$$g(w_t) = g_t(w_t) \geq g_t(w_{t+1}) \geq g(w_{t+1})$$

($\geq$ upper bound) //

- local linear convergence to a station. pt. for latent SVMstruct [see NeurIPS OPT workshop 2012 paper]

---

* CCCP for SVMstruct: $v(w) = \max_{z'} \langle w, \varphi(x,y,z') \rangle \rightarrow \hat{=} \operatorname{argmax}_{z'} \langle w_t, \varphi(x,y,z') \rangle$

$\hat{z}(x,y,w_t)$

$$\partial v(w_t) = \varphi(x,y,\hat{z}(x,y,w_t))$$

$$\Rightarrow g_t(w) = \max_{(\tilde{y},\tilde{z})} \langle w, \varphi(x,\tilde{y},\tilde{z}) \rangle + \ell(y,\tilde{y},\tilde{z}) - \langle w, \varphi(x,y,\hat{z}_t) \rangle + \text{cst.}$$

↝ like SVMstruct objective

CCCP algorithm for latent SVMstruct:

repeat:
- fill in $\hat{z}_t^{(i)}$ for all ground truth $y^{(i)}$ using w_t
- solve a standard SVMstruct to get w_{t+1}
- repeat

10h16

Deep learning

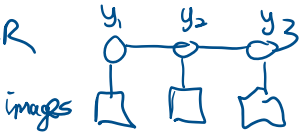
go from $\langle w, \varphi(x,y) \rangle$ to $\langle w, \varphi(x,y;\theta) \rangle$ (can learn)

↳ plug in "deep learning" features in a structured prediction model

go from $\langle w, \varphi(x,y)\rangle$ to $\langle w, \psi(x,y;\theta)\rangle$

I) plug in "deep learning" features in a structured pred. model

example: OCR



example:
[Vu & al: ICCV 2015]
"context aware CNNs for person head detection"

so far $\varphi_t(x_t, y_t) = \begin{pmatrix} 0 \\ x_t \\ 0 \end{pmatrix} \leftarrow y^{th}$

instead $\varphi_t(x_t, y_t) = \begin{pmatrix} 0 \\ NN_\theta(x_t) \\ 0 \end{pmatrix} \leftarrow y^{th}$ — pre-trained on imagenet e.g.

II) "end-to-end" training

structured prediction energy networks (SPENs)

$\varphi(x,y,\theta) = \langle w, \psi(x,y,\theta)\rangle$

III) recurrent neural networks (RNN)

motivation: $p(y|x) = \prod_{t \le T} p(y_t | y_{1:t-1}, x)$ — chain rule

graphical modeling approaches:

- $y_t \perp y_{1:t-1} \mid y_{\pi_t}$ [directed] $\prod_t p(y_t | y_{\pi_t}, x)$
- $\frac{1}{Z(x)} \prod_{c \in C} \psi_c(y_c, x)$ [undirected]

RNN → "structured parameterization" with no cond. indep. assumptions of $p(y_t | y_{1:t-1}, x)$ using a NN

↳ usually loose exact decoding

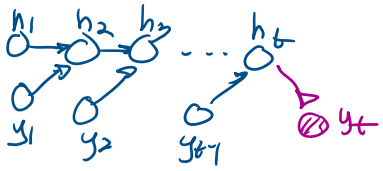
$h_{t+1} \triangleq f(h_t, x, y_t, W)$

$h_t = f(f(\dots f(\underbrace{f(h_1, x, y_1, W)}_{h_2}, x, y_2)) \dots), x, y_{t-1}, W)$ ($\underbrace{f(h_1, x, y_1, W)}_{h_2} \to h_3$)

define

$p(y_t | y_{1:t-1}, x) \propto \exp(c(y_t)^T \tilde{W} h_t)$

"code" for y_t, e.g.



$p(y_t | y_{1:t-1}, x)$

↳ given by a deep NN architecture

e.g. word embedding in NMT and can be fine tuned

standard learning: using MCL

$\sum_{i=1}^{n}$

standard learning : using MCL

i.e. $\min_{w,\tilde{w}} -\frac{1}{n}\sum_{i=1}^{n} \log p(y^{(i)} \mid x^{(i)})$

$\sum_t \log p(y_t^{(i)} \mid y_{1:t-1}^{(i)}, x^{(i)})$ ← output of a deep NN

"teacher forcing" → yield "exposure problem" i.e. don't know $p(\tilde{y}_t \mid \text{unseen } (\tilde{y}_{1:t-1}), z)$

for ML, do SGD on obj.

gradient of $\log p(y_t^{(i)} \mid y_{1:t-1}^{(i)}, x^{(i)}; w, \tilde{w})$

→ use back propagation

decoding : $\operatorname{argmax}_{\tilde{y} \in \mathcal{Y}} \sum_t \log p(\tilde{y}_t \mid \tilde{y}_{1:t-1}, x)$ → NP hard?

→ need approximation
- greedy decoding $\hat{y}_t = \operatorname{argmax}_{y_t \in \mathcal{Y}_t} p(y_t \mid \hat{y}_1, \dots, \hat{y}_{t-1}, x)$
- beam search "greedy decoding with memory of size K" ↳ size of beam

beam search: goal: construct $\hat{y}_1, \dots \hat{y}_T$

beam of size L (memory)

- at step t, you have L candidate solution prefixes $y_{1:t}^{(1)}, \dots, y_{1:t}^{(L)}$
- expand possible next choice $|\mathcal{Y}_{t+1}| \cdot L$

score them (e.g. $\log p(y_{t+1} \mid y_{1:t}^{(l)}, x) + \log p(y_{1:t}^{(l)}, x)$)

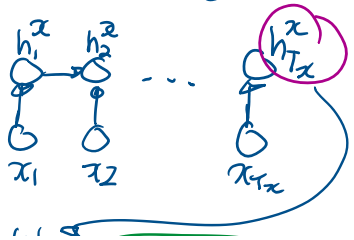
then keep top L candidates as $\{y_{1:t+1}^{(l)}\}_{l=1}^{L}$

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

seq2seq a.k.a. encoder/decoder architecture

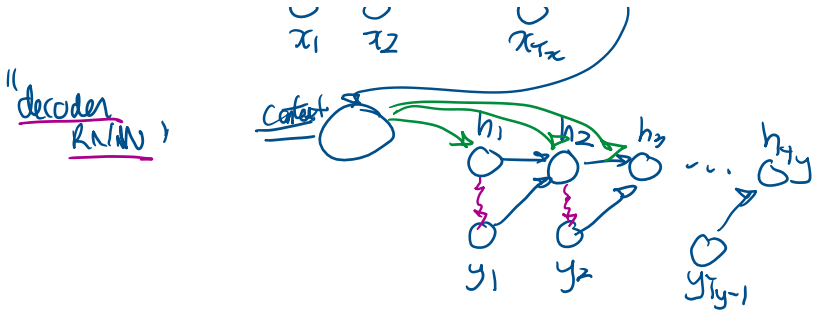
↳ useful way to get $p(y_t \mid y_{1:t-1}, x)$ for a RNN when x has variable length

"encoder RNN"



→ fixed size representation

"decoder



Issues:

a) variable length output? → end-of-sequence symbol

b) how to handle long input sequence x?

problem: need to summarize input sentence in one context vector of fixed length

solution: "attention mechanism"

c) vanishing gradient?

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

d) sequential processing

→ fixed by "transformer" architecture "attention is all you need"

"self-attention"

↳ quadratic cost

→ need tricks to handle long context / sequence

→ GPT & al.