Lecture 10.

Note Title 2/8/2010

Alternative structure learning.

Output. $\underline{y} = (y_1, \ldots, y_M)$ input $\underline{d}$

Want to learn $\underline{\lambda}$ s.t. $\hat{\underline{y}}(\underline{\lambda}, \underline{d}) = \arg\max_{\underline{y}} \underline{\lambda} \cdot \underline{\varphi}(\underline{y}, \underline{d})$

is closed to the solution $\underline{y}(\underline{d})$ for training data $\{(\underline{y}_i, \underline{d}_i) : i = 1 \text{ to } N\}$

Error measure $\Delta(\underline{y} : \underline{y}^i) = \sum_{a=1}^{M} I(y_a \neq y_a^i)$

Define: $\Delta \underline{\varphi}_i(\underline{y}) = \underline{\varphi}(\underline{d}_i, \underline{y}_i) - \underline{\varphi}(\underline{d}_i, \underline{y})$

so $\underline{\lambda} \cdot \Delta \underline{\varphi}_i(\underline{y})$ is the difference of (negative) energy between the solution $\underline{y}_i$ and another state $\underline{y}$.

Minimize $\frac{1}{2} |\underline{\lambda}|^2 + C \sum_i \xi_i$

s.t. $\underline{\lambda} \cdot \Delta \underline{\varphi}_i(\underline{y}) \geq \Delta(\underline{y} : \underline{y}_i) - \xi_i \quad \forall i, y$

ξ_i are slack variables $\xi_i \geq 0$

like for standard binary SVM's.

i.e. make the difference of energy scale like the error. — there is a potential problem with this which we describe later.

Intuitively design the energy landscape so that the highest (negative) energy is at the correct solution and the energy falls off like $\Delta(\underline{y} : \underline{y}_i)$.

$$R(\underline{\lambda}) = \frac{1}{2} |\underline{\lambda}|^2 + C \sum_i \max_y \{ \Delta(\underline{y} : \underline{y}_i) - \underline{\lambda} \cdot \Delta \underline{\varphi}_i(\underline{y}) \}$$

$$= \frac{1}{2} |\underline{\lambda}|^2 + C \sum_i \max_y \{ \Delta(\underline{y} : \underline{y}_i) + \underline{\lambda} \cdot \underline{\varphi}(\underline{d}_i, \underline{y}) \} - \underline{\lambda} \cdot \underline{\varphi}(\underline{d}_i, \underline{y}_i)$$

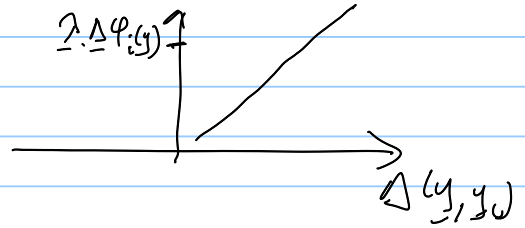
As before, same as what we obtained from the bound in the last lecture.

Problem $\max_{\underline{y}} \{ \Delta(\underline{y} : \underline{y}_i) + \underline{\lambda} \cdot \underline{\varphi}(\underline{d}_i, \underline{y}) \}$ may select states $\underline{y}$ which are a long way from the solution.

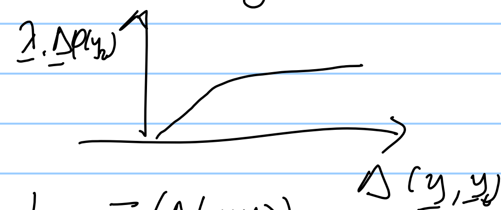
(2) <u>Problem</u>: we are imposing too strong a requirement on the energy fall-off.

$\underline{\lambda} \cdot \underline{\Delta\phi}_i(y) \geq \Delta(\underline{y}:\underline{y}_i)$ is too tough a criteria.

It pays too much attention to the behavior of the energy away from the solution $\underline{y}_i$



instead we only need



<u>So</u> can replace $\Delta(\underline{y}:\underline{y}_i)$ by $\sigma(\Delta(y:y_i))$ which reaches an asymptotic value for suff large $\Delta(y:y_i)$

<u>for simplicity</u> $\rightarrow$ require $\Delta(\underline{y}:\underline{y}_i)$ to have this asymptotic property (rescale if it doesn't).

<u>Primal</u> $\quad \frac{1}{2}|\underline{\lambda}|^2 + C\sum_i \max_y \{\Delta(y:y_i) - \underline{\lambda} \cdot \underline{\Delta\phi}_i(y)\}$

online learning: $\underline{\lambda}^{t+1} = \underline{\lambda}^t + C\{\underline{\phi}(\underline{d}_i, \hat{y}) - \underline{\phi}(\underline{d}_i, \underline{y}_i)\}$

$\hat{y} = \arg\max_y \{\Delta(y:y_i) + \underline{\lambda} \cdot \underline{\Delta\phi}_i(y)\}$

There is also a dual formulation

Introduce Lagrange parameters $\alpha_i(y)$ to deal with the inequality constraints

Obtain dual in the standard way - some dual in terms of Lagrange multipliers.

<u>Dual</u>, $\quad \sum_i\sum_y \alpha_i(y)\Delta(y;y_i) - \frac{1}{2}\left|\sum_{i,y}\alpha_i(y)\Delta\phi_i(y)\right|^2$

with constraints $\sum_y \alpha_i(y) = C, \ \forall i \qquad \alpha_i(y) \geq 0, \ \forall i,y$

like prob. dist.

<u>For</u> many problems the Δ & $\Delta\phi$ have simple form

$\Delta(y,y_i) = \sum_a I(y_a \neq y_a^i)$ <u>unary</u> terms.

$\Delta\phi_i(y) = \sum_{(a,b)\in\mathcal{E}} \Delta\phi(y_a, y_b)$ $\quad$ Here dual depends only on pseudomarginals $\alpha_i(y_a), \alpha_i(y_a,y_b)$

Like Bethe

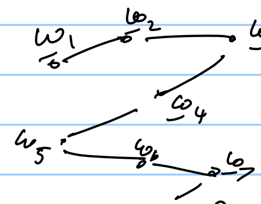
$\max \sum_i \sum_{a,y_a} \alpha_i(y_a) I(y_a \neq y_a^i) - \frac{1}{2}\sum_{i,j}\sum_{\substack{a,b\\ y_a y_b}}\sum_{\substack{c,d\\ y_c y_d}} \alpha_i(y_a,y_b)\alpha_j(y_c,y_d)\Delta\phi_i(y_a,y_b)\Delta\phi_j(y_c,y_d)$

+ Consistency $\sum_{y_b}\alpha_i(y_a,y_b) = \alpha_i(y_a)$.

(3)

Graphical Models of Objects.

Hand Model.



$G = (\mathcal{V}, \mathcal{E})$

nodes edges.

state variables $\omega_i \in \mathcal{V}$.

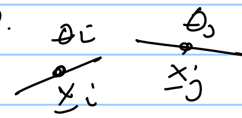
$$E[\{\omega_\mu\}; I] = \sum_{\mu \in \mathcal{V}} \lambda_\mu \cdot \phi_\mu(\omega_\mu; I) + \sum_{(\mu,\nu) \in \mathcal{E}} \lambda_{\mu\nu} \cdot \phi_{\mu\nu}(\omega_\mu, \omega_\nu).$$

$$P[\{\omega_\mu\} | I] = \frac{1}{Z} e^{-E[\{\omega_\mu\}: I]}$$

If Graph Structure is linear — tree — then you can use dynamic programming to estimate $\{\hat{\omega}_\mu\} = \arg\min_{\{\omega_\mu\}} E[\{\omega_\mu\}: I]$

Quadratic in state space & linear in number of nodes

But: state space is very large.



$\omega_\mu = (x_\mu, \theta_\mu)$

position orientation

To make this practical:

(i) prune DP paths if the energy is above a threshold.

(ii) surround suppression — eliminate solutions which are too close.

What are the energy terms $\lambda_\mu \cdot \phi_\mu(\omega_\mu; I)$ & $\lambda_{\mu\nu} \cdot \phi_{\mu\nu}(\omega_\mu, \omega_\nu)$?

For the "Hand" application the unary terms $\lambda_\mu \cdot \phi_\mu(\omega_\mu; I)$ "attract" the nodes to edges in the image and orientate them perpendicular to the edge gradient

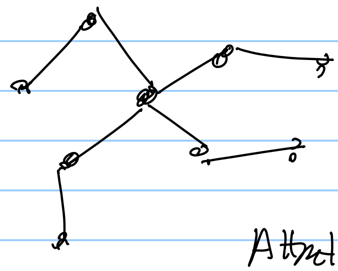
For the "Hand" application the binary term $\lambda_{\mu\nu} \cdot \phi_{\mu\nu}(\omega_\mu, \omega_\nu)$ imposes spatial constraints

→ predict the position of node ν relative to node μ using a prototype hand but allow for some uncertainty. → Coughlan et al.

Note: if there is training data then you can learn the λ's by the methods described in the last few lectures

(4) Pictorial Structures:

- Felzenszwalb & Huttenlocher
- Perona.



Tree-Structure: enables DP

Data terms: $\sum_\mu \phi_\mu(\underline{\omega}_\mu; I)$

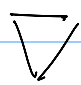
Attract to (i) edges

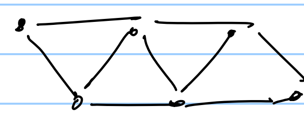
(ii) corners

(iii) interest points → e.g. SIFT

Alternatively

Triangles





closed loops, but can convert to DP using triangles

geometric priors — enable invariance to rotation, scale, position

Δ—Δ—Δ—Δ

linear in triangles.

data terms —

enable us to use appearance cues within the object, not just at the nodes.

Other models

→ require other inference algorithms (e.g. BP).