STRUCTURAL SVMS

COMP 4680/8650
Justin Domke
comp8650@anu.edu.au

Reading: Murphy, §19.7

What to do for inference?

- Suppose we have some distribution $p(y|x)$.
  - E.g. x = images, y = segmentations.

- Given an input x, what to predict?

$$y^* = \arg\max_{y \in \mathcal{Y}} p(y|x)$$

$$y_i^* = \arg\max_{y_i \in \mathcal{Y}_i} p(y_i|x)$$

- What's the right thing to do?

Utility Functions

- $U(y, a)$ says how happy you are to take action a if true output is y.

- E.g., driving a car.

$$a \in \{\text{left, center, right}\}$$

$$U(y, a) = \begin{cases} 1 & \text{if on road} \\ -10 & \text{if off road} \\ -100000 & \text{if crashed} \end{cases}$$

Utility Functions

- Pick action a to maximize expected reward.

$$\sum_y p(y|x)U(y,a)$$

- Sometimes "actions" are the states themselves.

- If $U(y,y') = I[y = y']$, then optimal approach is

$$y' = \arg\max_{y \in \mathcal{Y}} p(y|x)$$

- If $U(y,y') = \sum_i I[y_i = y_i']$, then optimal approach is

$$y_i' = \arg\max_{y_i \in \mathcal{Y}_i} p(y_i|x)$$

MAP Inference

- If we are only going to use $p(y|x;w)$ to do MAP inference, why fit a full probability distribution?
- Alternative Approach. Define the "mapping"

$$y^*(x;w) := \arg\max_{y \in \mathcal{Y}} p(y|x;w)$$

and just find w to make this mapping accurate.

- Why do this?
  - Model mis-specification. (Similar to conditional vs. joint training.)
  - Computational Issues.

Computational Issues

- MAP inference is NP-complete
- Marginal inference is #P-complete

- Tons of works exists on great MAP solvers, often for specific problems (proteins, image segmentations)

- Often, MAP solvers can certify that they found the optimal solution in a given instance, even if this is not possible in general.

SSVMs – Basic idea

- Take some probabilistic model

$$p(y|x;w) = \exp(w^T \phi(y,x) - A(w))$$

- Instead of being probabilistic, define a loss directly on the inference procedure

$$y^*(x;w) := \arg\max_{y \in \mathcal{Y}} p(y|x;w)$$

$$y^*(x;w) = \arg\max_{y \in \mathcal{Y}} w^T \phi(y,x)$$

Structured Perceptron

$$\min_w \sum_i Q(w, x_i, y_i)$$

$$Q(w, x_i, y_i) = \max_{\hat{y}_i \in \mathcal{Y}} w^T \phi(\hat{y}_i, x_i) - w^T \phi(y_i, x_i)$$

- What needs to happen to have no loss?

$$w^T \phi(y_i, x_i) = \max_{\hat{y}_i \in \mathcal{Y}} w^T \phi(\hat{y}_i, x_i)$$

Structured Perceptron

Stochastic subgradient descent

- For $k = 1, 2, \ldots$
  - Pick a random i
  - Find $\hat{y}_i = \arg\max_{\hat{y}_i \in \mathcal{Y}} w^T \phi(\hat{y}_i, x_i)$
  - Set $w \leftarrow w - \eta_k \cdot (\phi(\hat{y}_i, x_i) - \phi(y_i, x_i))$

Gradually decrease η_k over time.

Structured Perceptron

- Advantages:
  - Only need to be able to perform MAP inference

- Disadvantages
  - Only need score of correct output to slightly beat all incorrect output
  - Loss is not tuned for application priorities.

Structured SVM

$$\min_w \sum_i Q(w, x_i, y_i)$$

$$Q(w, x_i, y_i) = \max_{\hat{y}_i \in \mathcal{Y}} L(y_i, \hat{y}_i) + w^T \phi(\hat{y}_i, x_i) - w^T \phi(y_i, x_i)$$

- What needs to happen to have no loss?

$$w^T \phi(y_i, x_i) = \max_{\hat{y}_i \in \mathcal{Y}} L(y_i, \hat{y}_i) + w^T \phi(\hat{y}_i, x_i)$$

Structural SVMs

- $L(y_i, \hat{y}_i)$ measures how much you dislike predicting $\hat{y}_i$ when true output is y_i.

- Common example: Hamming distance

$$L(y_i, \hat{y}_i) = \sum_n I[y_{in} \neq \hat{y}_{in}]$$

Structured SVMs

Stochastic subgradient descent

- For $k = 1, 2, \ldots$
  - Pick a random i
  - Find $\hat{y}_i = \arg\max_{\hat{y}_i \in \mathcal{Y}} L(y_i, \hat{y}_i) + w^T \phi(\hat{y}_i, x_i)$
  - Set $w \leftarrow w - \eta_k \cdot (\phi(\hat{y}_i, x_i) - \phi(y_i, x_i))$

Gradually decrease η_k over time.

Loss-augmented inference

Loss-Augmented Inference

$$\hat{y}_i = \arg\max_{\hat{y}_i \in \mathcal{Y}} L(y_i, \hat{y}_i) + w^T \phi(\hat{y}_i, x_i)$$

- Typically, L can be "folded into" into the features, so that

$$\hat{y}_i = \arg\max_{\hat{y}_i \in \mathcal{Y}} w'^T \phi(\hat{y}_i, x_i)$$

- Example:

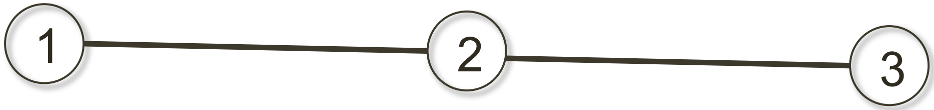
$$\phi(y, x) = \{x \cdot I[y_i = y_i^*]\} \cup \{I[y_i = y_i^*, y_j = y_j^*]\}$$

$$L(y_i, \hat{y}_i) = \sum_i I[y_i \neq \hat{y}_i]$$

Demo – Multi-Label Prediction

$x \in \mathbb{R}^N$ $\quad y \in \{-1, +1\}^N$ $\quad L(y_i, \hat{y}_i) = \sum_i I[y_i \neq \hat{y}_i]$

With two variables: $\phi(y, x) = (x_1 y_1, x_2 y_2, y_1 y_2)$



With N variables: $\phi(y, x) = \{x_i y_i\} \cup \{y_i y_{i+1}\}$

Optimization Theory

$$\min_w \frac{\lambda}{2}||w||^2 + \sum_{i=1}^{N} \max_y w^T(\phi(y, x_i) - \phi(y_i, x_i)) + L(y_i, y)$$

$$\min_w \frac{1}{2}||w||^2 + C\sum_{i=1}^{N} \max_y w^T(\phi(y, x_i) - \phi(y_i, x_i)) + L(y_i, y)$$

$$\min_w \frac{1}{2}||w||^2 + C\sum_{i=1}^{N} \xi_i$$

$$\text{s.t.} \quad \forall i, \forall y, \; \xi_i \geq L(y_i, y) + w^T(\phi(y, x_i) - \phi(y_i, x_i))$$

Optimization Theory

$$\min_{w,\xi} \frac{1}{2}||w||^2 + C\sum_{i=1}^{N}\xi_i$$
$$\text{s.t.} \quad \forall i, \forall y, \;\; \xi_i \geq L(y_i, y) + w^T(\phi(y, x_i) - \phi(y_i, x_i))$$

- This is a quadratic objective, with linear constraints, known as a quadratic program (QP).

- However, there are $N|\mathcal{Y}|$ constraints!
  - E.g., with binary variables, $|\mathcal{Y}| = 2^N$.

Max-Margin Markov Networks (2003)

- Convert optimization to "dual form".

Primal formulation (6)

$$\min \quad \frac{1}{2}||\mathbf{w}||^2 + C\sum_{\mathbf{x}} \xi_{\mathbf{x}} \; ;$$

$$\text{s.t.} \;\; \mathbf{w}^\top \Delta\mathbf{f}_{\mathbf{x}}(\mathbf{y}) \geq \Delta\mathbf{t}_{\mathbf{x}}(\mathbf{y}) - \xi_{\mathbf{x}}, \; \forall \mathbf{x}, \mathbf{y}.$$

Dual formulation (7)

$$\max \sum_{\mathbf{x},\mathbf{y}} \alpha_{\mathbf{x}}(\mathbf{y})\Delta\mathbf{t}_{\mathbf{x}}(\mathbf{y}) - \frac{1}{2}\left|\left|\sum_{\mathbf{x},\mathbf{y}} \alpha_{\mathbf{x}}(\mathbf{y})\Delta\mathbf{f}_{\mathbf{x}}(\mathbf{y})\right|\right|^2;$$

$$\text{s.t.} \sum_{\mathbf{y}} \alpha_{\mathbf{x}}(\mathbf{y}) = C, \forall \mathbf{x}; \quad \alpha_{\mathbf{x}}(\mathbf{y}) \geq 0\,, \forall \mathbf{x}, \mathbf{y}.$$

- Make change of variables, optimize over <u>marginals</u>.

$$\begin{aligned}
\mu_{\mathbf{x}}(y_i, y_j) &= \textstyle\sum_{\mathbf{y}\sim[y_i,y_j]} \alpha_{\mathbf{x}}(\mathbf{y}), \quad \forall\, (i,j) \in E, \forall y_i, y_j, \;\; \forall\, \mathbf{x}; \\
\mu_{\mathbf{x}}(y_i) &= \textstyle\sum_{\mathbf{y}\sim[y_i]} \alpha_{\mathbf{x}}(\mathbf{y}), \qquad \forall\, i, \; \forall y_i, \;\; \forall\, \mathbf{x};
\end{aligned}$$

(Only works for treelike graphs.)

Cutting Plane Optimization (2009)

- Repeat: 1) Solve this optimization:

$$\min_{w,\xi} \frac{1}{2}||w||^2 + C\sum_{i=1}^{N}\xi_i$$
$$\text{s.t.} \quad \forall(i,y) \in A, \quad \xi_i \geq L(y_i,y) + w^T(\phi(y,x_i) - \phi(y_i,x_i))$$

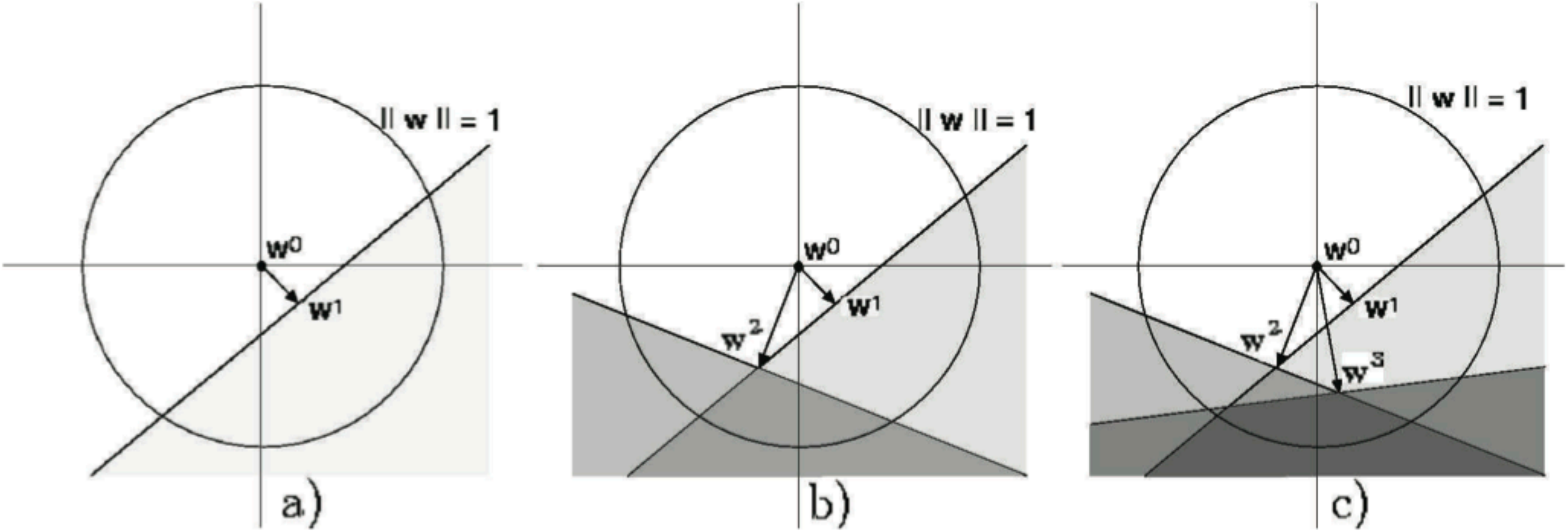
- 2) For all i, find most violated constraint:

$$\max_y L(y_i,y) + w^T(\phi(y,x_i) - \phi(y_i,x_i))$$

- If more than ξ_i, add (i,y) to A.

- 3) If don't need to add any constraints, we have the global optimum!

Cutting Plane Optimization (2009)



Linear Complexity: SVM-Struct (2009)

$$\min_{w,\xi} \frac{1}{2}||w||^2 + C\sum_{i=1}^{N}\xi_i$$
$$\text{s.t.} \quad \forall (i,y) \in A, \quad \xi_i \geq L(y_i, y) + w^T(\phi(y, x_i) - \phi(y_i, x_i))$$

- Reformulate with a <u>single</u> ξ

$$\min_{w,\xi} \frac{1}{2}||w||^2 + C\xi$$
$$\text{s.t.} \quad \forall (\bar{y}_1, ..., \bar{y}_N) \in A, \quad \xi \geq \sum_{i=1}^{N} L(y_i, \bar{y}_i) + w^T(\phi(\bar{y}_i, x_i) - \phi(y_i, x_i))$$

- Same strategy: iteratively add to A. But, can show:
  - Optimizing objective takes O(N) time.
  - Total number of iterations is a constant.
  - Thus, solve entire optimization in O(N) time, and O(N) oracle calls.