Part 5: Structured Support Vector Machines

Sebastian Nowozin and Christoph H. Lampert

Colorado Springs, 25th June 2011

Reminder: learning by regularized risk minimization

For compatibility function $g(x, y; w) := \langle w, \phi(x, y) \rangle$ find w^* that minimizes

$$\mathbb{E}_{(x,y) \sim d(x,y)} \, \Delta(\, y, \mathrm{argmax}_y \, g(x, y; w) \,).$$

Two major problems:

- $d(x, y)$ is unknown
- $\mathrm{argmax}_y \, g(x, y; w)$ maps into a discrete space
  $\rightarrow \Delta(\, y, \mathrm{argmax}_y \, g(x, y; w))$ is discontinuous, piecewise constant

Task:

$$\min_{w} \quad \mathbb{E}_{(x,y)\sim d(x,y)} \, \Delta(\, y, \operatorname{argmax}_y g(x,y;w) \,).$$

Problem 1:

- $d(x,y)$ is unknown

Solution:

- Replace $\mathbb{E}_{(x,y)\sim d(x,y)}\big(\,\cdot\,\big)$ with *empirical estimate* $\frac{1}{N}\sum_{(x^n,y^n)}\big(\,\cdot\,\big)$
- To avoid overfitting: add a *regularizer*, e.g. $\lambda\|w\|^2$.

New task:

$$\min_{w} \quad \lambda\|w\|^2 + \frac{1}{N}\sum_{n=1}^{N} \Delta(\, y^n, \operatorname{argmax}_y g(x^n,y;w) \,).$$

Task:

$$\min_w \quad \lambda\|w\|^2 + \frac{1}{N}\sum_{n=1}^{N} \Delta(\ y^n, \mathrm{argmax}_y\, g(x^n, y; w)\ ).$$

Problem:

- $\Delta(\ y, \mathrm{argmax}_y\, g(x, y; w)\ )$ discontinuous w.r.t. w.

Solution:

- Replace $\Delta(y, y')$ with *well behaved* $\ell(x, y, w)$
- Typically: ℓ *upper bound* to Δ, *continuous* and *convex* w.r.t. w.

New task:

$$\min_w \quad \lambda\|w\|^2 + \frac{1}{N}\sum_{n=1}^{N} \ell(x^n, y^n, w))$$

Regularized Risk Minimization

$$\min_{w} \qquad \lambda\|w\|^2 \quad + \quad \frac{1}{N}\sum_{n=1}^{N} \ell(x^n, y^n, w))$$

Regularization + *Loss on training data*

Hinge loss: maximum margin training

$$\ell(x^n, y^n, w) := \max_{y\in\mathcal{Y}} \left[\Delta(y^n, y) + \langle w, \phi(x^n, y)\rangle - \langle w, \phi(x^n, y^n)\rangle \right]$$

Alternative:

Logistic loss: probabilistic training

$$\ell(x^n, y^n, w) := \log \sum_{y\in\mathcal{Y}} \exp\big(\langle w, \phi(x^n, y)\rangle - \langle w, \phi(x^n, y^n)\rangle\big)$$

Structured Output Support Vector Machine

$$\min_w \frac{1}{2}\|w\|^2 + \frac{C}{N}\sum_{n=1}^{N}\Big[\max_{y\in\mathcal{Y}} \Delta(y^n,y) + \langle w,\phi(x^n,y)\rangle - \langle w,\phi(x^n,y^n)\rangle\Big]$$

Conditional Random Field

$$\min_w \frac{\|w\|^2}{2\sigma^2} + \sum_{n=1}^{N}\Big[\log\sum_{y\in\mathcal{Y}}\exp\big(\langle w,\phi(x^n,y)\rangle - \langle w,\phi(x^n,y^n)\rangle\big)\Big]$$

CRFs and SSVMs have more in common than usually assumed.

- both do regularized risk minimization
- $\log\sum_y \exp(\cdot)$ can be interpreted as a *soft-max*

Solving the Training Optimization Problem Numerically

Structured Output Support Vector Machine:

$$\min_{w} \; \frac{1}{2}\|w\|^2 + \frac{C}{N}\sum_{n=1}^{N}\Big[\max_{y\in\mathcal{Y}} \; \Delta(y^n, y) + \langle w, \phi(x^n, y)\rangle - \langle w, \phi(x^n, y^n)\rangle\Big]$$

Unconstrained optimization, convex, non-differentiable objective.

Structured Output SVM (equivalent formulation):

$$\min_{w,\xi} \quad \frac{1}{2}\|w\|^2 + \frac{C}{N}\sum_{n=1}^{N}\xi^n$$

subject to, for $n = 1, \dots, N$,

$$\max_{y\in\mathcal{Y}} \Big[\Delta(y^n, y) + \langle w, \phi(x^n, y)\rangle - \langle w, \phi(x^n, y^n)\rangle\Big] \le \xi^n$$

N non-linear contraints, convex, differentiable objective.

Structured Output SVM (also equivalent formulation):

$$\min_{w,\xi} \quad \frac{1}{2}\|w\|^2 + \frac{C}{N}\sum_{n=1}^{N} \xi^n$$

subject to, for $n = 1, \ldots, N$,

$$\Delta(y^n, y) + \langle w, \phi(x^n, y)\rangle - \langle w, \phi(x^n, y^n)\rangle \leq \xi^n, \quad \text{for all } y \in \mathcal{Y}$$

$N|\mathcal{Y}|$ linear constraints, convex, differentiable objective.

Summary – S-SVM Learning

Given:

- training set $\{(x^1, y^1), \ldots, (x^n, y^n)\} \subset \mathcal{X} \times \mathcal{Y}$
- loss function $\Delta : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$.

Task: learn parameter w for $f(x) := \operatorname{argmax}_y \langle w, \phi(x, y) \rangle$ that minimizes expected loss on future data.

S-SVM solution derived by *maximum margin* framework:

- enforce correct output to be better than others by a margin:

$$\langle w, \phi(x^n, y^n) \rangle \geq \Delta(y^n, y) + \langle w, \phi(x^n, y) \rangle \quad \text{for all } y \in \mathcal{Y}.$$

- convex optimization problem, but non-differentiable
- many equivalent formulations $\to$ different training algorithms
- training needs repeated argmax prediction, no probabilistic inference

Solving the Training Optimization Problem Numerically

We can solve an S-SVM like a linear SVM:

One of the equivalent formulations was:

$$\min_{w\in\mathbb{R}^D,\xi\in\mathbb{R}^n_+} \|w\|^2 \;+\; \frac{C}{N}\sum_{n=1}^{N}\xi^n$$

subject to, for $i = 1, \dots n$,

$$\langle w, \phi(x^n, y^n)\rangle - \langle w, \phi(x^n, y)\rangle \geq \Delta(y^n, y) \;-\; \xi^n, \quad \text{for all } y \in \mathcal{Y}‘.$$

Introduce feature vectors $\delta\phi(x^n, y^n, y) := \phi(x^n, y^n) - \phi(x^n, y)$.

Solve

$$\min_{w \in \mathbb{R}^D, \xi \in \mathbb{R}^n_+} \|w\|^2 \ + \ \frac{C}{N} \sum_{n=1}^{N} \xi^n$$

subject to, for $i = 1, \ldots n$, for all $y \in \mathcal{Y}$,

$$\langle w, \delta\phi(x^n, y^n, y) \rangle \geq \Delta(y^n, y) \ - \ \xi^n.$$

This has the same structure as an ordinary SVM!

- quadratic objective ☺
- linear constraints ☺

Solve

$$\min_{w\in\mathbb{R}^D,\xi\in\mathbb{R}^n_+} \|w\|^2 \; + \; \frac{C}{N}\sum_{n=1}^{N}\xi^n$$

subject to, for $i = 1, \dots n$, for all $y \in \mathcal{Y}$,

$$\langle w, \delta\phi(x^n, y^n, y)\rangle \geq \Delta(y^n, y) \; - \; \xi^n.$$

This has the same structure as an ordinary SVM!

- quadratic objective ☺
- linear constraints ☺

Question: Can't we use a ordinary SVM/QP solver?

Solve

$$\min_{w\in\mathbb{R}^D,\xi\in\mathbb{R}^n_+} \|w\|^2 \; + \; \frac{C}{N}\sum_{n=1}^{N}\xi^n$$

subject to, for $i = 1,\dots n$, for all $y \in \mathcal{Y}$,

$$\langle w, \delta\phi(x^n, y^n, y)\rangle \geq \Delta(y^n, y) \; - \; \xi^n.$$

This has the same structure as an ordinary SVM!

- quadratic objective ☺
- linear constraints ☺

Question: Can't we use a ordinary SVM/QP solver?

Answer: Almost! We could, if there weren't $N|\mathcal{Y}|$ constraints.

- E.g. 100 binary 16×16 images: 10^{79} constraints

Solution: working set training

- It's enough if we enforce the **active constraints**.
  The others will be fulfilled automatically.
- We don't know which ones are active for the optimal solution.
- But it's likely to be only a small number $\leftarrow$ can of course be formalized.

Keep a set of potentially active constraints and update it iteratively:

Solution: working set training

- It's enough if we enforce the **active constraints**.
  The others will be fulfilled automatically.
- We don't know which ones are active for the optimal solution.
- But it's likely to be only a small number $\leftarrow$ can of course be formalized.

Keep a set of potentially active constraints and update it iteratively:

Working Set Training

- Start with working set $S = \emptyset$ (no contraints)
- Repeat until convergence:
  - Solve S-SVM training problem with constraints from S
  - Check, if solution violates any of the *full* constraint set
    - if no: we found the optimal solution, *terminate*.
    - if yes: add most violated constraints to S, *iterate*.

Solution: working set training

- It's enough if we enforce the **active constraints**.
  The others will be fulfilled automatically.
- We don't know which ones are active for the optimal solution.
- But it's likely to be only a small number $\leftarrow$ can of course be formalized.

Keep a set of potentially active constraints and update it iteratively:

Working Set Training

- Start with working set $S = \emptyset$ (no contraints)
- Repeat until convergence:
  - Solve S-SVM training problem with constraints from S
  - Check, if solution violates any of the *full* constraint set
    - if no: we found the optimal solution, *terminate*.
    - if yes: add most violated constraints to S, *iterate*.

Good *practical performance* and *theoretic guarantees*:

- polynomial time convergence ϵ-close to the global optimum

[Tsochantaridis et al. "Large Margin Methods for Structured and Interdependent Output Variables", JMLR, 2005.]

Working Set S-SVM Training

input training pairs $\{(x^1, y^1), \dots, (x^n, y^n)\} \subset \mathcal{X} \times \mathcal{Y}$,
input feature map $\phi(x, y)$, loss function $\Delta(y, y')$, regularizer C

$S \leftarrow \emptyset$
repeat
    $(w, \xi) \leftarrow$ *solution to QP only with constraints from* S
    for i=1,. . . ,n **do**
        $\hat{y} \leftarrow \mathrm{argmax}_{y \in \mathcal{Y}} \Delta(y^n, y) + \langle w, \phi(x^n, y) \rangle$
        if $\hat{y} \neq y^n$ **then**
            $S \leftarrow S \cup \{(x^n, \hat{y})\}$
        end if
    end for
until S doesn't change anymore.

output prediction function $f(x) = \mathrm{argmax}_{y \in \mathcal{Y}} \langle w, \phi(x, y) \rangle$.

Observation: each update of w needs 1 argmax-prediction per example.
(but we solve globally for next w, not by local steps)