Advanced Machine Learning
Lecture 20

Structured Output Learning

21.01.2013

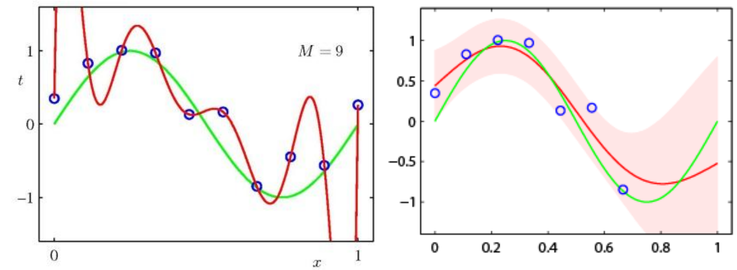
Bastian Leibe
RWTH Aachen
http://www.vision.rwth-aachen.de/

leibe@vision.rwth-aachen.de

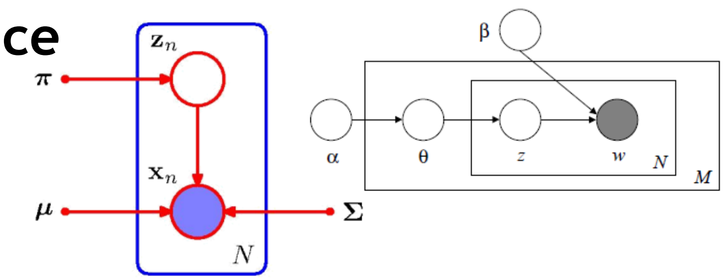
This Lecture: *Advanced Machine Learning*

- Regression Approaches
  - Linear Regression
  - Regularization (Ridge, Lasso)
  - Kernels (Kernel Ridge Regression)
  - Gaussian Processes

$$f : \mathcal{X} \rightarrow \mathbb{R}$$



- Bayesian Estimation & Bayesian Non-Parametrics
  - Prob. Distributions, Approx. Inference
  - Mixture Models & EM
  - Dirichlet Processes
  - Latent Factor Models
  - Beta Processes



- SVMs and Structured Output Learning
  - SVMs, SVDD, SV Regression
  - Structured Output Learning

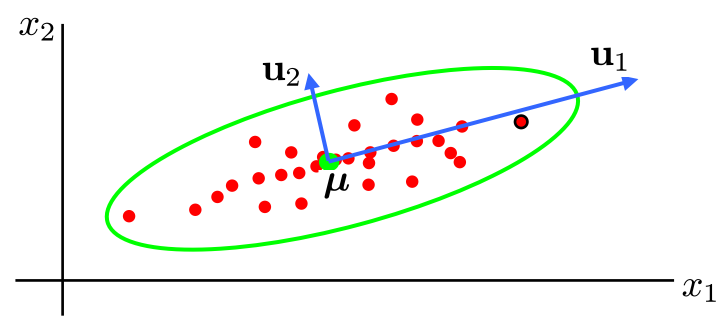
$$f : \mathcal{X} \rightarrow \mathcal{Y}$$

Topics of This Lecture

- Recap: Extensions to Support Vector Machines
  - Kernel PCA
  - Support Vector Data Description (1-class SVMs)
  - Support Vector Regression
- Structured Output Learning
  - From arbitrary inputs to arbitrary outputs
  - General structured prediction
  - Structured loss functions
  - Structured Output SVM
  - Cutting plane training

Recap: Kernel-PCA

- **Kernel-PCA procedure**
  - Given samples $\mathbf{x}_n \in \mathcal{X}$, kernel $\mathcal{X} \times \mathcal{X} \to \mathbb{R}$ with an implicit feature map ϕ: $\mathcal{X} \to \mathcal{H}$. Perform PCA in the Hilbert space $\mathcal{H}$.
  - Equivalently, we can use the eigenvectors $\mathbf{e}'_k$ and eigenvalues λ_k of the kernel matrix



$$\begin{aligned} K &= (\langle \phi(\mathbf{x}_m), \phi(\mathbf{x}_n) \rangle)_{m,n=1,\ldots,N} \\ &= (k(\mathbf{x}_m, \mathbf{x}_n))_{m,n=1,\ldots,N} \end{aligned}$$

  - Coordinate mapping: $\mathbf{x}_n \mapsto (\sqrt{\lambda_1}\mathbf{e}'_1, \ldots, \sqrt{\lambda_K}\mathbf{e}'_K)$

- **Subtle issue: Centering**
  - Subtracting the mean would require us to work in $\mathcal{H}$ with $\phi(\mathbf{x})$.
  - More elaborate procedure ($\to$ Bishop Ch. 12.3)

Recap: One-Class SVMs

- **Objective function**
  - Find the smallest ball (center $\mathbf{c} \in \mathcal{H}$, radius R) that contains "most" of the samples.
  - Solve

$$\min_{R\in\mathbb{R},\, \mathbf{c}\in\mathcal{H},\, \xi_n\in\mathbb{R}^+} R + \frac{1}{\nu N}\sum_{n=1}^{N}\xi_n$$

subject to

$$\|\phi(\mathbf{x}_n) - \mathbf{c}\|^2 \leq R^2 + \xi_n \qquad \text{for} \quad n = 1, \ldots, N$$

where $\nu \in (0,1)$ upper bounds the number of outliers.

⇒ Sparse solution, can be written entirely in terms of kernel functions $k(\mathbf{x}_n, \mathbf{x}_m)$.

⇒ Often used for outlier/anomaly detection.

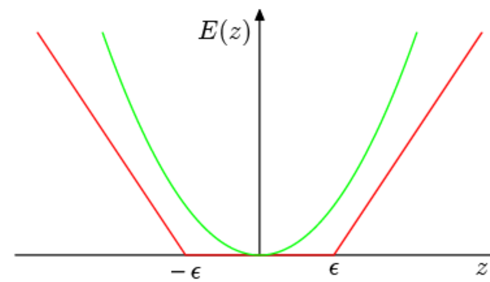
Slide adapted from Christoph Lampert

RWTH AACHEN UNIVERSITY

Recap: SV Regression

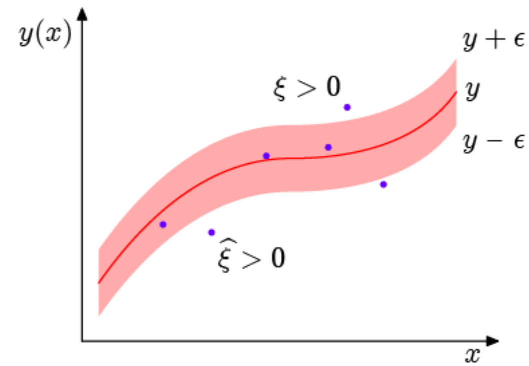
- **Obtaining sparse solutions**
  - **Define an ϵ-insensitive error function**

$$E_\epsilon(y(\mathbf{x}) - t) = \begin{cases} 0, & \text{if } |y(\mathbf{x}) - t| < \epsilon \\ |y(\mathbf{x} - t| - \epsilon, & \text{otherwise} \end{cases}$$



  - **Use for large-margin optimization**

$$C \sum_{n=1}^{N} [|y(\mathbf{x}_n) - t_n| - \epsilon]_+ + \frac{1}{2}||\mathbf{w}||^2$$



  - **Optimization with slack variables**

$$C \sum_{n=1}^{N} (\xi_n + \widehat{\xi}_n) + \frac{1}{2}||\mathbf{w}||^2$$

$$\begin{aligned} \xi_n &\geq 0 \\ \widehat{\xi}_n &\geq 0 \end{aligned}$$

$\Rightarrow$ **Support Vector Regression**

Recap: SV Regression - Primal Form

- **Lagrangian primal form**

$$L_p = C\sum_{n=1}^{N}(\xi_n + \widehat{\xi}_n) + \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{n=1}^{N}(\mu_n\xi_n + \widehat{\mu}_n\widehat{\xi}_n)$$

$$- \sum_{n=1}^{N} a_n\left(\epsilon + \xi_n + y_n - t_n\right) - \sum_{n=1}^{N}\widehat{a}_n\left(\epsilon + \widehat{\xi}_n - y_n + t_n\right)$$

- **Solving for the variables**

$$\frac{\partial L}{\partial \mathbf{w}} = 0 \;\Rightarrow\; \boxed{\mathbf{w} = \sum_{n=1}^{N}(a_n - \widehat{a}_n)\phi(\mathbf{x}_n)} \qquad \frac{\partial L}{\partial \xi_n} = 0 \;\Rightarrow\; \boxed{a_n + \mu_n = C}$$

$$\frac{\partial L}{\partial b} = 0 \;\Rightarrow\; \boxed{\sum_{n=1}^{N}(a_n - \widehat{a}_n) = 0} \qquad \frac{\partial L}{\partial \widehat{\xi}_n} = 0 \;\Rightarrow\; \boxed{\widehat{a}_n + \widehat{\mu}_n = C}$$

Recap: SV Regression – Dual Form

- From this, we can derive the dual form
  - Maximize

$$L_d(\mathbf{a}, \widehat{\mathbf{a}}) = -\frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N}(a_n - \widehat{a}_n)(a_m - \widehat{a}_m)k(\mathbf{x}_n, \mathbf{x}_m)$$

$$-\epsilon\sum_{n=1}^{N}(a_n + \widehat{a}_n) + \sum_{n=1}^{N}(a_n - \widehat{a}_n)t_n$$

  - under the conditions

$$0 \leq a_n \leq C$$

$$0 \leq \widehat{a}_n \leq C$$

  - Predictions for new inputs are then made using

$$y(\mathbf{x}) = \sum_{n=1}^{N}(a_n - \widehat{a}_n)k(\mathbf{x}, \mathbf{x}_n) + b$$

Topics of This Lecture

- Recap: Extensions to Support Vector Machines
  - Kernel PCA
  - Support Vector Data Description (1-class SVMs)
  - Support Vector Regression
- Structured Output Learning
  - From arbitrary inputs to arbitrary outputs
  - General structured prediction
  - Structured loss functions
  - Structured Output SVM
  - Cutting plane training

From Arbitrary Inputs to Arbitrary Outputs

- **With kernels, we can handle "arbitrary" input spaces:**
  - We only need a pairwise similarity measure for objects:
    - Images: e.g., χ^2 kernel
    - Gene sequences: e.g., string kernels
    - Graphs: e.g., random walk kernels

- **We can learn mappings**

$$f : \mathcal{X} \to \{-1, 1\} \quad \text{or} \quad f : \mathcal{X} \to \mathbb{R}$$

- **What about arbitrary *output* spaces?**
  - We know: kernels correspond to feature maps: $\phi : \mathcal{X} \to \mathcal{H}$.
  - But: we cannot invert ϕ, there is no $\phi^{-1} : \mathcal{H} \to \mathcal{X}$.
  - Kernels do not readily help us to construct

$$f : \mathcal{X} \to \mathcal{Y} \quad \text{with} \quad \mathcal{Y} \neq \mathbb{R}$$

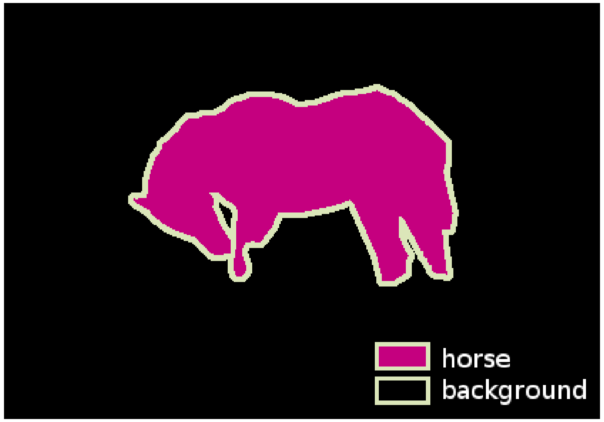
What Would We Like to Predict?

- Natural Language Processing:
  - Automatic Translation (output: sentences)
  - Sentence Parsing (output: parse trees)
- Bioinformatics:
  - Secondary Structure Prediction (output: bipartite graphs)
  - Enzyme Function Prediction (output: path in a tree)
- Robotics:
  - Planning (output: sequence of actions)
- Computer Vision
  - Image Segmentation (output: segmentation mask)
  - Human Pose Estimation (output: positions of body parts)
  - Image Retrieval (output: ranking of images in database)

Slide credit: Christoph Lampert

Example: Semantic Image Segmentation

$\mapsto$



Input: images **Output: segmentation masks**

- **Problem formulation**
  - **Input space :** $\mathcal{X} = \{\textbf{images}\} \equiv [0{,}255]^{3 \cdot M \cdot N}$
  - **Output space:** $\mathcal{Y} = \{\textbf{segmentation masks}\} \equiv \{0{,}1\}^{M \cdot N}$
  - **(Structured) prediction function:** $f : \mathcal{X} \to \mathcal{Y}$

$$f(\mathbf{x}) := \arg\min_{\mathbf{y} \in \mathcal{Y}} E(\mathbf{x}, \mathbf{y})$$

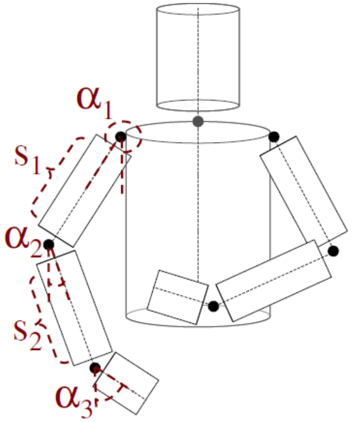
  - **Energy function**

$$E(\mathbf{x}, \mathbf{y}) = \sum_i \mathbf{w}_i^\top \phi_{unary}(\mathbf{x}_i, \mathbf{y}_i) + \sum_i \sum_j \mathbf{w}_{ij}^\top \phi_{pairwise}(\mathbf{y}_i, \mathbf{y}_j)$$

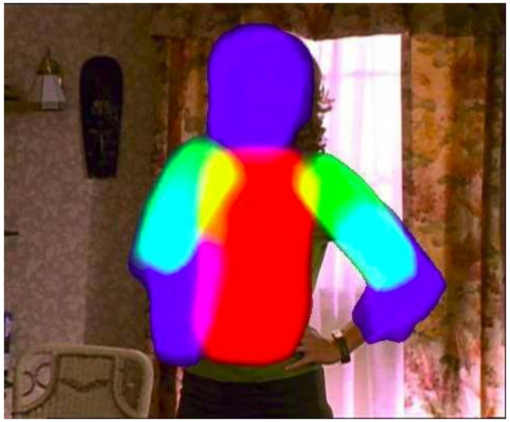
Images from [M. Everingham et al., IJCV'10]

Example: Human Pose Estimation

Input: image



Body model



Output: model fit

- **Problem formulation**
  - Input space : $\mathcal{X} = \{\text{images}\}$
  - Output space: $\mathcal{Y} = \{\text{pos./angles of body parts}\} \equiv \mathbb{R}^{4K}$
  - (Structured) prediction function: $f : \mathcal{X} \to \mathcal{Y}$

$$f(\mathbf{x}) := \arg\min_{\mathbf{y} \in \mathcal{Y}} E(\mathbf{x}, \mathbf{y})$$

  - Energy function

$$E(\mathbf{x}, \mathbf{y}) = \sum_i \mathbf{w}_i^\top \phi_{fit}(\mathbf{x}_i, \mathbf{y}_i) + \sum_i \sum_j \mathbf{w}_{ij}^\top \phi_{pose}(\mathbf{y}_i, \mathbf{y}_j)$$

Example: Object Localization

Input: image

Output: object position
(*left*, *top*, *right*, *bottom*)

- **Problem formulation**
  - Input space : $\mathcal{X} = \{\text{images}\}$
  - Output space: $\mathcal{Y} = \{\text{bounding box coordinates}\} \equiv \mathbb{R}^4$
  - (Structured) prediction function: $f : \mathcal{X} \to \mathcal{Y}$

$$f(\mathbf{x}) := \arg\max_{\mathbf{y}\in\mathcal{Y}} F(\mathbf{x}, \mathbf{y})$$

  - Scoring function $F(\mathbf{x}, \mathbf{y}) = \mathbf{w}^\top \phi(\mathbf{x}, \mathbf{y})$, where $\phi(\mathbf{x},\mathbf{y}) = h(\mathbf{x}|_{\mathrm{y}})$ is a feature vector for an image region, e.g., bag-of-words.

Computer Vision Examples: Summary

- **Image Segmentation**

$$\mathbf{y} = \operatorname*{argmin}_{\mathbf{y}\in\{0,1\}^N} E(\mathbf{x},\mathbf{y})$$

$$E(\mathbf{x},\mathbf{y}) = \sum_i \mathbf{w}_i^\top \phi_{unary}(\mathbf{x}_i,\mathbf{y}_i) + \sum_{i,j} \mathbf{w}_{ij}^\top \phi_{pairwise}(\mathbf{y}_i,\mathbf{y}_j)$$

- **Pose Estimation**

$$\mathbf{y} = \operatorname*{argmin}_{\mathbf{y}\in\mathbb{R}^{4K}} E(\mathbf{x},\mathbf{y})$$

$$E(\mathbf{x},\mathbf{y}) = \sum_i \mathbf{w}_i^\top \phi_{fit}(\mathbf{x}_i,\mathbf{y}_i) + \sum_{i,j} \mathbf{w}_{ij}^\top \phi_{pose}(\mathbf{y}_i,\mathbf{y}_j)$$

- **Object Localization**

$$\mathbf{y} = \operatorname*{argmax}_{\mathbf{y}\in\mathbb{R}^4} F(\mathbf{x},\mathbf{y})$$

$$F(\mathbf{x},\mathbf{y}) = \mathbf{w}^\top \phi(\mathbf{x},\mathbf{y})$$

Slide credit: Christoph Lampert

Grand Unified View

Predict structured output by maximization

$$\mathbf{y} = \arg\max_{\mathbf{y} \in \mathcal{Y}} F(\mathbf{x}, \mathbf{y})$$

of a compatibility function

$$F(\mathbf{x}, \mathbf{y}) = \langle \mathbf{w}, \phi(\mathbf{x}, \mathbf{y}) \rangle$$

that is linear in a parameter vector $\mathbf{w}$.

Slide credit: Christoph Lampert

Generic Structured Prediction

- A generic structured prediction problem
  - $\mathcal{X}$: arbitrary input domain
  - $\mathcal{Y}$: structured output domain, decompose $\mathbf{y} = (y_1,\ldots,y_K)$
  - Prediction function $f : \mathcal{X} \rightarrow \mathcal{Y}$ given by

$$f(\mathbf{x}) = \arg\max_{\mathbf{y}\in\mathcal{Y}} F(\mathbf{x}, \mathbf{y})$$

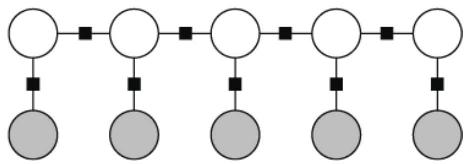
  - Compatibility function (or negative of "energy")

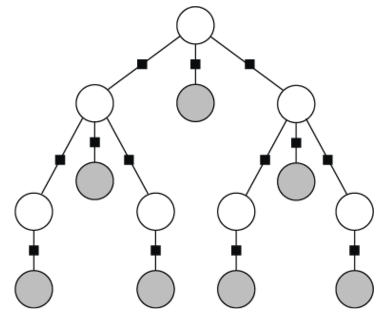
$$\begin{aligned} F(\mathbf{x}, \mathbf{y}) &= \langle \mathbf{w}, \phi(\mathbf{x}, \mathbf{y}) \rangle \\ &= \sum_{i=1}^{K} \mathbf{w}_i^\top \phi_i(y_i, \mathbf{x}) && \text{unary terms} \\ &+ \sum_{i,j=1}^{K} \mathbf{w}_{ij}^\top \phi_{ij}(y_i, y_j, \mathbf{x}) && \text{binary terms} \\ &+ \ldots && \text{higher-order terms} \end{aligned}$$

Generic Structured Prediction

- Machine Learning lecture: How to solve $\mathrm{argmax}_{\mathbf{y}}\ F(\mathbf{x},\mathbf{y})$?
  - Loop-free graphs: Viterbi algorithm, max-sum BP

Chain

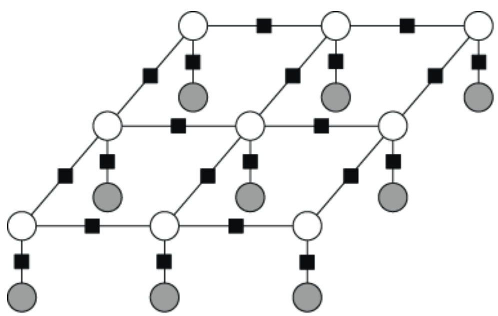


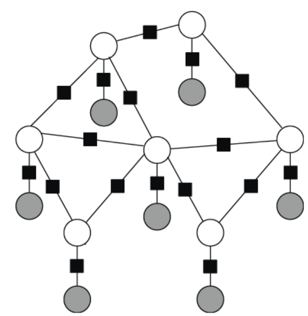


Tree

  - Loopy graphs: Graph Cuts, Loopy BP

Grid





Arbitrary graph

- This lecture
  - *How to learn a good function $F(\mathbf{x},\mathbf{y})$ from training data?*

Parameter Learning in Structured Models

- **Problem statement**
  - Given: parametric model (family): $F(\mathbf{x},\mathbf{y}) = \langle \mathbf{w}, \phi(\mathbf{x},\mathbf{y}) \rangle$
  - Given: prediction method: $f(\mathbf{x}) = \operatorname{argmax}_{\mathbf{y}\in\mathcal{Y}} F(\mathbf{x},\mathbf{y})$
  - Not given: parameter vector $\mathbf{w}$ (high-dimensional)

- **Supervised Training**
  - Given: example pairs $\{(\mathbf{x}_1,\mathbf{y}_1), \ldots, (\mathbf{x}_n, \mathbf{y}_n)\} \subset \mathcal{X} \times \mathcal{Y}$.
  - Typical inputs with "the right" outputs for them.

$\{$  $, \ldots\}$

  - Task: determine „good" $\mathbf{w}$.

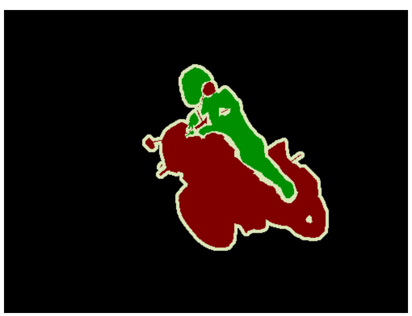
Loss Function

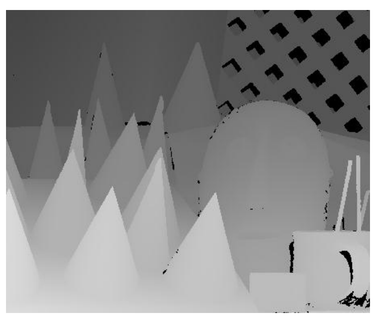
- What make a solution "good"?
  - Define a loss function

$$\Delta : \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R}^{+}$$

such that $\Delta(\mathbf{y}, \mathbf{y}')$ measures the loss/cost incurred by predicting $\mathbf{y}'$ when $\mathbf{y}$ is correct.

- The loss function is application dependent:


Number of mislabeled pixels

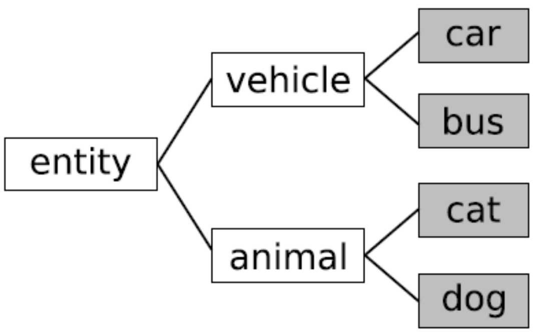

Total depth error


Number of wrong body parts

Bounding box area overlap

Some Popular Structured Loss Functions

- **Zero-one loss**
  - Definition: $\Delta(\mathbf{y}, \mathbf{y}') = \delta(\mathbf{y} \neq \mathbf{y}')$
  - *"Every prediction that is not identical to the intended one is considered a mistake, and all mistakes are penalized equally."*
  - Most common loss for multi-class problems.
  - Less frequently used for structured prediction tasks.



- **Hierarchical multi-class loss**
  - Definition: $\Delta(\mathbf{y}, \mathbf{y}') = \frac{1}{2} dist_H(\mathbf{y}, \mathbf{y}')$
    where H is a hierarchy over the classes in $\mathcal{Y}$ and $dist_H(\mathbf{y},\mathbf{y}')$ measures the distance of $\mathbf{y}$ and $\mathbf{y}'$.
  - Common way to incorporate information about label hierarchies in multi-class prediction problems

Image from [Nowozin & Lampert, FTCGV'11]

Some Popular Structured Loss Functions

- **Hamming loss**
  - Definition: $\Delta(\mathbf{y},\mathbf{y}') = \frac{1}{m}\sum_{i=1}^{m}\delta(\mathbf{y}_i \neq \mathbf{y}'_i)$
  - Frequently used loss for image segmentation and other tasks in which the output $\mathbf{y}$ consists of multiple part labels $\mathbf{y}_1,\ldots,\mathbf{y}_m$.
  - Each part label is judged independently and the average number of labeling errors is determined.

- **Area overlap loss**
  - Definition: $\Delta(\mathbf{y},\mathbf{y}') = \frac{area(\mathbf{y}\cap\mathbf{y}')}{area(\mathbf{y}\cup\mathbf{y}')}$
  - Standard loss in object localization, e.g., the PASCAL VOC detection challenges.
  - $\mathbf{y}$ and $\mathbf{y}'$ are bounding box coordinates, and $\mathbf{y} \cap \mathbf{y}'$ and $\mathbf{y} \cup \mathbf{y}'$ are their intersection and union, respectively.

Structured Output SVM

- **Two criteria for decision function f:**
  1. **Correctness: Ensure $f(\mathbf{x}_n) = \mathbf{y}_n$ for training data, $n = 1, \ldots, N$.**
  2. **Robustness: f should also work if $\mathbf{x}_n$ are perturbed.**

- **Translated to structured prediction, this means**
  - **With $f(\mathbf{x}) = \operatorname{argmax}_{\mathbf{y}\in\mathcal{Y}} \langle \mathbf{w}, \phi(\mathbf{x}, \mathbf{y})\rangle$:**
  1. **Ensure for $n = 1, \ldots, N$,**

$$\operatorname*{argmax}_{\mathbf{y}\in\mathcal{Y}} \langle \mathbf{w}, \phi(\mathbf{x}_n, \mathbf{y})\rangle = \mathbf{y}_n$$

$$\Leftrightarrow \quad \langle \mathbf{w}, \phi(\mathbf{x}_n, \mathbf{y}_n)\rangle \geq \epsilon + \langle \mathbf{w}, \phi(\mathbf{x}_n, \mathbf{y})\rangle \qquad \text{for all } \mathbf{y} \in \mathcal{Y} \setminus \{\mathbf{y}_n\}$$

  2. **Enforce large margin, minimize $\|\mathbf{w}\|^2$.**

Structured Output SVM

- **Slack formulation** of S-SVM
  - Solve
$$\min_{\mathbf{w}\in\mathbb{R}^D,\, \xi_n\in\mathbb{R}^+} \frac{1}{2}\|\mathbf{w}\|^2 + \frac{C}{N}\sum_{n=1}^{N}\xi_n$$
    subject to
$$\langle \mathbf{w}, \phi(\mathbf{x}_n, \mathbf{y}_n)\rangle \geq \Delta(\mathbf{y}_n, \mathbf{y}) + \langle \mathbf{w}, \phi(\mathbf{x}_n, \mathbf{y})\rangle - \xi_n$$
$$\text{for all } \mathbf{y} \in \mathcal{Y} \setminus \{\mathbf{y}_n\}$$

- **Interpreting the constraint terms:**

$\langle \mathbf{w}, \phi(\mathbf{x}_n, \mathbf{y}_n)\rangle$	Score for output $\mathbf{y}_n$
$\langle \mathbf{w}, \phi(\mathbf{x}_n, \mathbf{y})\rangle$	Score for any other output $\mathbf{y}$
$\Delta(\mathbf{y}_n, \mathbf{y}) \geq 0$	Loss for predicting $\mathbf{y}$ when $\mathbf{y}_n$ would be correct
ξ_n	Slack, outliers may violate criterion

Structured Output SVM

- **Slack formulation of S-SVM**
  - Solve
$$\min_{\mathbf{w}\in\mathbb{R}^D,\, \xi_n\in\mathbb{R}^+} \frac{1}{2}\|\mathbf{w}\|^2 + \frac{C}{N}\sum_{n=1}^{N}\xi_n$$
  subject to
$$\langle \mathbf{w}, \phi(\mathbf{x}_n,\mathbf{y}_n)\rangle \;\geq\; \Delta(\mathbf{y}_n,\mathbf{y}) + \langle \mathbf{w}, \phi(\mathbf{x}_n,\mathbf{y})\rangle - \xi_n$$
$$\text{for all } \mathbf{y} \in \mathcal{Y} \setminus \{\mathbf{y}_n\}$$

- **Optimization problem very similar to normal SVM**
  - Quadratic in $\mathbf{w}$, linear in ξ.
  - Constraints linear in $\mathbf{w}$ and ξ.
  - Convex!

- **But there are $N(|\mathcal{Y}|-1)$ constraints!**

  $\Rightarrow$ Numeric optimization needs some tricks, will be expensive.

Discussion

- S-SVM formulation with slack variables
  - The constrained optimization problem has an elementary form and is jointly convex.
  - However, this advantage comes at the price of a large number of constraints: $|\mathcal{Y}|$ inequalities per training sample!

  $\Rightarrow$ For most structured prediction problems, this is much larger than what software packages for constrained convex optimization can process in reasonable time.

  $\Rightarrow$ Often not even possible to *store* all the constraints in memory!

- However...
  - Weight vector has only D degrees of freedom.
  - Slack variables have only N degrees of freedom.

  $\Rightarrow$ $D+N$ constraints suffice to determine the optimal solution.

  $\Rightarrow$ *The question is only which of them are the essential ones...*

Solving S-SVM Training

- Solving the S-SVM optimization
  - If we knew the set of relevant constraints in advance, we could solve the optimization efficiently.
  - ⇒ Approximate the solution iteratively.
  - ⇒ Cutting Plane training algorithm.
- Cutting Plane training
  - Delayed constraint generation technique
  - Search for the best weight vector and the set of active constraints simultaneously in an iterative manner.
  - Approximate solution with much faster runtime.

Cutting Plane Training

- Cutting Plane algorithm
  1. Start from an empty working set.
  2. In each iteration, solve the optimization problem for $(\mathbf{w}^*, \xi^*)$ with only the constraints in the working set.
  3. Check for each sample if any of the $|\mathcal{Y}|$ constraints are violated.
  4. If not, we have found the optimal solution.
  5. Otherwise, add most violated constraints to the working set.

- Speed-ups
  - To achieve faster convergence, choose a tolerance $\epsilon > 0$ and require a constraint to be violated by at least ϵ.

  $\Rightarrow$ Possible to prove convergence after $\mathcal{O}(\frac{1}{\epsilon^2})$ steps with the guarantee that objective value at the solution differs only at most by ϵ from the global minimum.

Cutting Plane Algorithm

Algorithm 15 Cutting Plane S-SVM Training

$w^* = \textsc{CuttingPlane}(\varepsilon)$
Input:
    ε tolerance
Output:
    $w^* \in \mathbb{R}^D$ learned weight vector
Algorithm:
$S \leftarrow \emptyset$
repeat
    $(w_{cur}, \xi_{cur}) \leftarrow$ solution to (6.7) with constraints (6.8) from S
    for n=1,. . . ,N **do**
        $y^* \leftarrow \operatorname{argmax}_{y \in \mathcal{Y}} H_n(y; w_{cur}, \xi_{cur})$
        if $H_n(y^*; w_{cur}, \xi_{cur}) > \varepsilon$ **then**
            $S \leftarrow S \cup \{(x^n, y^*)\}$
        end if
    end for
until S did not change in this iteration
$w^* \leftarrow w_{cur}$

where $H_n(y; w, \xi) := g(x^n, y, w) - g(x^n, y^n, w) + \Delta(y, y^n) - \xi^n$.

Source: [Nowozin & Lampert, FTCGV'11]

Cutting Plane: Most Expensive Steps

- Solving the quadratic optimization problem
  - As long as the working set size is reasonable, this can be solved using general purpose quadratic program solvers, either in the primal or in the dual form.
  - Also possible to adapt existing SVM training methods (typically leads to much higher performance).

- Identifying the most violated constraint
  - Loss-augmented prediction step
  - Need to solve N optimization problems of the form
$$\mathrm{argmax}_{\mathbf{y}} \;\; \Delta(\mathbf{y}_n, \mathbf{y}) + \langle \mathbf{w}, \phi(\mathbf{x}_n, \mathbf{y}) \rangle$$
  - Strong resemblance to the evaluation of
$$f(\mathbf{x}) = \mathrm{argmax}_{\mathbf{y}} \;\; \langle \mathbf{w}, \phi(\mathbf{x}_n, \mathbf{y}) \rangle$$
  - In many cases, Δ can be rewritten to look like additional terms of the inner product. $\Rightarrow$ reuse MAP prediction routines.

Summary for Today

- We have motivated Structured Prediction
  - Ability to use a large set of more general loss functions...
  - ...while keeping the large-margin learning idea.
  - ⇒ Possibility to design a loss function that directly optimizes the scoring function the final approach will be evaluated on.
- Introduction to Structured SVMs
  - Formulation with slack variables
  - Cutting-plane training
- What is still missing?
  - How to incorporate kernels?
  - How is this used in applications?
  - ⇒ *Next lecture...*

References and Further Reading

- Structured SVMs were first introduced here
  - I. Tsochantaridis, T. Joachims, T. Hofmann, Y. Altun, Large Margin Methods for Structured and Interdependent Output Variables, Journal of Machine Learning Research, Vol. 6, pp. 1453-1484, 2005.

- Additional details on Structured SVMs can be found in Chapter 6 of the following tutorial on Structured Learning
  - S. Nowozin, C. Lampert, Structured Learning and Prediction in Computer Vision, Foundations and Trends in Computer Graphics and Vision, Vol. 6(3-4), pp. 185-365, 2011.