Structured Output Learning

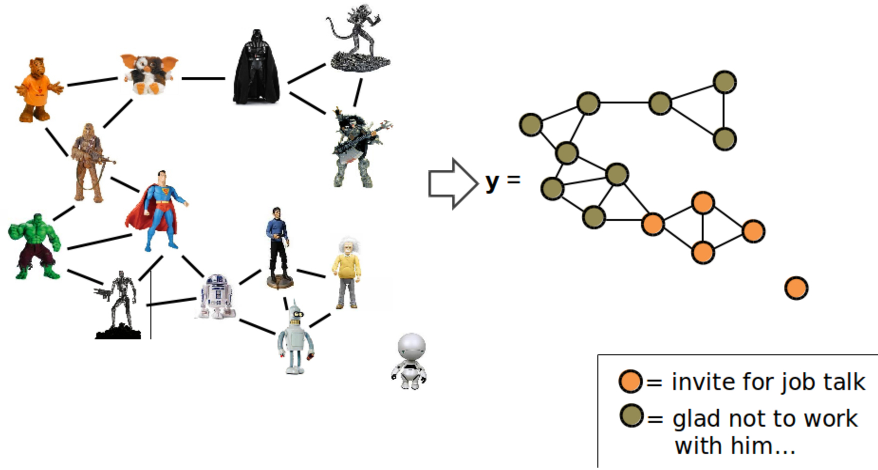
Nico Görnitz, TU Berlin, MPI Tuebingen
Ulf Brefeld, Yahoo! Research, Barcelona
Prof. Klaus R. Müller, TU Berlin

Overview

Examples

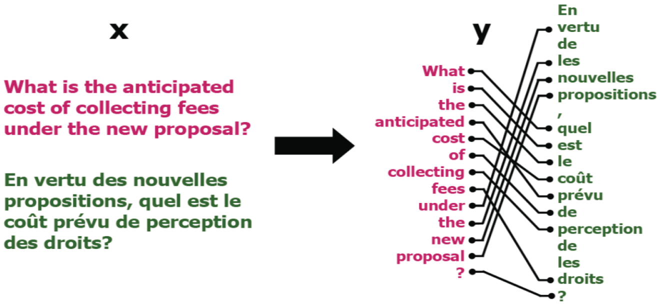
Collective Classification

- Task: Classify with respect to linkage
- input: graph
- output: graph



Bilingual Text Alignment

- Task: Align two sentences (source & target language)
- input: 2 sentences
- output: alignment

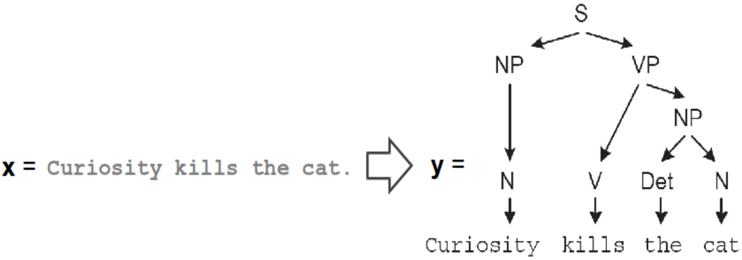


Combinatorial Structure

From Klein & Taskar, ACL'05 Tutorial

Natural Language Parsing

- Task: Predict the most probable parse tree for a given input sentence.
- input: sequence
- output: parse tree



Label Sequence Learning (1)

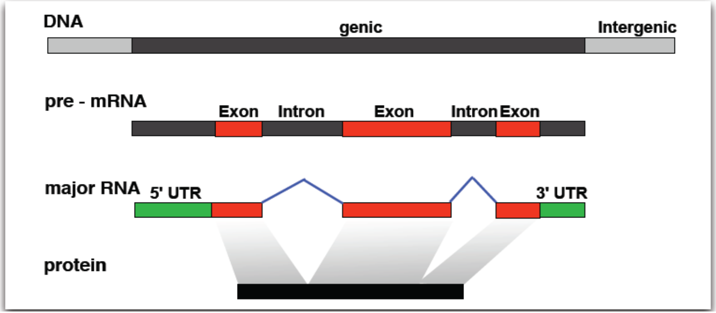
- Task: Part-of-speech (POS) tagging
- Related problems: Named entity recognition (NER)
- input: sequence
- output: sequence

x = Bello chases the cat.

y = < noun, verb, determiner, noun >

Label Sequence Learning (2)

- Task: Predict the most probable state sequence (gene finding)
- input: sequence
- output: sequence



Label Sequence Learning: Hidden Markov Models

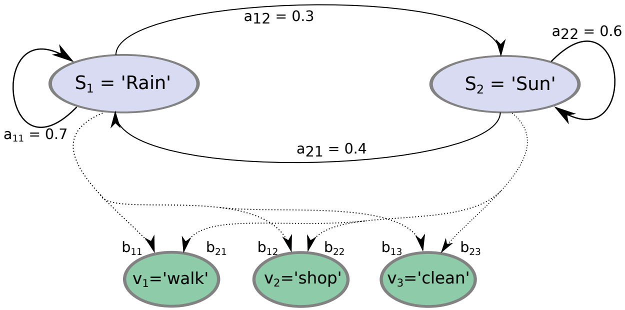
For some time around (see Rabiner, 'A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition', 1989)

- model: Ω
- observation sequences $\mathbf{o}_i$
- state sequences $\mathbf{q}_i$

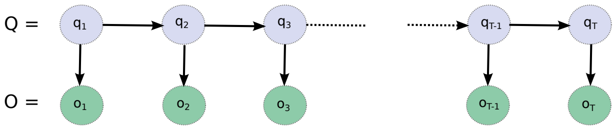
- Classical Tasks:
  1. predict the most probable state sequence $\mathbf{q}$ given Ω and $\mathbf{o}$
  2. calculate the probability of a observation $\mathbf{o}$ given the model Ω
  3. train the model Ω given observations $\mathbf{o}_i$

  (4. train the model Ω given observations $\mathbf{o}_i$ and corresponding state sequences $\mathbf{q}_i$)

State Model



Hidden Markov Model



$$P(Q, O|\lambda) = P(q_1, q_2, ..., q_T, o_1, o_2, ..., o_T|\lambda) = \prod_{i=1}^{T} p(q_i|q_{i-1}, \lambda)p(o_i|q_i, \lambda)$$

General Structured Output Learning

Hidden Markov Models apply to sequence structures (input and output), BUT what about tree's, graph, alignment problems, ...?

Joint Models in Input-Output Space (1)

Classical supervised learning

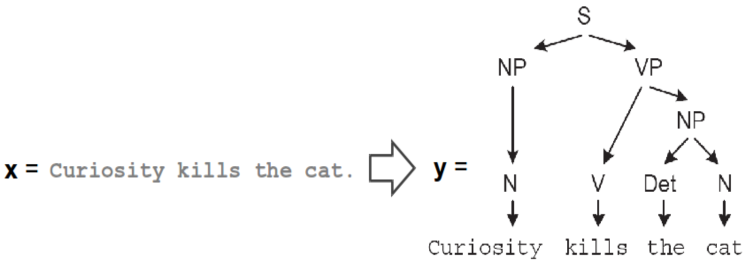
- Setting:
    - Observations-label pairs ($\mathbf{x}$,y).
    - $\mathbf{x} \in \Re^d$ and for
    - $y \in \{-1, 1\} \longrightarrow$ binary classification.
    - $y \in \Re \longrightarrow$ regression.
- Model f(x):
    - Classification: $y = \text{sign} f(\mathbf{x})$
    - Regression : $y = f(\mathbf{x})$
    - f shall generalize well on new and unseen data.

Joint Models in Input-Output Space (2)

- Structured Outputs:
  - Output variable y has an internal structure. (multiple variables with dependency structure)
  - Exponentially many possible values for y!
  - Model $f(\mathbf{x}) = y$ not appropriate to capture dependencies!
- Structured Approach:
  - Ranking model: $\mathbf{y} = \text{argmax}_{\bar{\mathbf{y}}} f(\mathbf{x}, \bar{\mathbf{y}})$
  - Model $f(\mathbf{x}, \mathbf{y}) = \langle \mathbf{w}, \Phi(\mathbf{x}, \mathbf{y}) \rangle$
  - Joint feature representation $\Phi(\mathbf{x}, \mathbf{y})$.

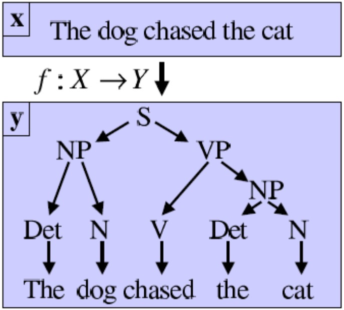
Natural Language Parsing (Remember?)

- Task: Predict the most probable parse tree for a given input sentence.
- input: sequence
- output: parse tree



Natural Language Parsing (Joint Feature Map)

- Task: Predict the most probable parse tree for a given input sentence.
- input: sequence
- output: parse tree



$$\Psi(\mathbf{x},\mathbf{y}) = \begin{pmatrix} 1 \\ 0 \\ 2 \\ 1 \\ \vdots \\ 0 \\ 2 \\ 1 \\ 1 \\ 1 \end{pmatrix} \begin{array}{c} S \to NP\ VP \\ S \to NP \\ NP \to Det\ N \\ VP \to V\ NP \\ \\ Det \to dog \\ Det \to the \\ N \to dog \\ V \to chased \\ N \to cat \end{array}$$

Problem Setting

- Given: n structured input-output pairs $(\mathbf{x}_1, \mathbf{y}_1), \ldots (\mathbf{x}_n, \mathbf{y}_n) \in \mathcal{X} \times \mathcal{Y}$.
  - E.g., $\mathbf{x}_i$ is a sentence and $\mathbf{y}_i$ the sequence of part-of-speech (POS) tags.
  - Let $|\mathbf{x}|$ denote the #words in $\mathbf{x}$ and let Ω be the set of POS-tags.
  - Possible output space $\mathcal{Y}$ for a given input $\mathbf{x}$ has $\Omega^{|\mathbf{x}|}$ elements.
- Loss function $\Delta : \mathcal{Y} \times \mathcal{Y} \to \Re$.
  - E.g., Hamming distance for sequences $\Delta(\mathbf{y}, \mathbf{y}') = \sum_{j=1}^{|\mathbf{x}|} [[y_j \neq y'_j]]$.

- Task: Find joint model $f : \mathcal{X} \times \mathcal{Y} \to \Re$ that minimizes the expected risk (the generalization error)

$$R[f] = \int_{\mathcal{X} \times \mathcal{Y}} \Delta(\mathbf{y}, \mathrm{argmax}_{\mathbf{y}'} f(\mathbf{x}, \mathbf{y}')) p(\mathbf{x}, \mathbf{y}) d\mathbf{x} d\mathbf{y}.$$

Structured Output Support Vector Machine

Structured Output Support Vector Machines

- Given $\mathbf{x}$='Bello chases the cat'
- We want: $\mathbf{y} = \text{argmax}_{\mathbf{y}'} \mathbf{w}^T \phi(\mathbf{x}, \mathbf{y}') = \langle N, V, D, N \rangle$

- Explicit representation:
  $\mathbf{w}^T \phi(\mathbf{x}, \langle N, V, D, N \rangle) \geq \mathbf{w}^T \phi(\mathbf{x}, \langle N, N, N, N \rangle)$
  $\mathbf{w}^T \phi(\mathbf{x}, \langle N, V, D, N \rangle) \geq \mathbf{w}^T \phi(\mathbf{x}, \langle N, N, N, V \rangle)$
  $\mathbf{w}^T \phi(\mathbf{x}, \langle N, V, D, N \rangle) \geq \mathbf{w}^T \phi(\mathbf{x}, \langle N, N, V, N \rangle)$
  $\mathbf{w}^T \phi(\mathbf{x}, \langle N, V, D, N \rangle) \geq \mathbf{w}^T \phi(\mathbf{x}, \langle N, V, N, N \rangle)$
  $\mathbf{w}^T \phi(\mathbf{x}, \langle N, V, D, N \rangle) \geq \mathbf{w}^T \phi(\mathbf{x}, \langle V, N, N, N \rangle)$
  ...

SO-SVM: Primal Problem

- Large margin approach

$$\begin{aligned} \min_{\mathbf{w},\xi} \quad & \frac{1}{2}\|\mathbf{w}\| + C\sum_{i=1}^{n}\xi_i \\ \text{s.t.} \quad & \forall_{i=1}^{n} \forall_{\bar{\mathbf{y}}\neq\mathbf{y}_i} : \langle \mathbf{w}, \Phi(\mathbf{x}_i,\mathbf{y}_i) - \Phi(\mathbf{x}_i,\bar{\mathbf{y}})\rangle \geq 1-\xi_i \\ & \forall_{i=1}^{n} : \xi_i \geq 0 \end{aligned}$$

- Dual representation: $\mathbf{w} = \sum_i \sum_{\mathbf{y}'} \alpha_{i(\mathbf{y})}(\phi(\mathbf{x}_i,\mathbf{y}_i) - \phi(\mathbf{x}_i,\mathbf{y}'))^T$
- Optimization leads to sparse models.
- Use Working set approach: incrementally add and remove constraints.
  $\forall_{i=1}^{n} \forall_{\bar{\mathbf{y}}\neq\mathbf{y}_i} \longrightarrow \forall_{i=1}^{n}\, \bar{y} = \text{argmax}_{\mathbf{y}'\neq\mathbf{y}_i} \mathbf{w}^T\Phi(\mathbf{x}_i,\mathbf{y}')$
  Computation of **argmax** depends on the application at hand.
  E.g., Viterbi algorithm (sequential outp.) or chart parser (tree struct. outp)
- Unconstrained version:
  $\min_{\mathbf{w}} \frac{1}{2}||w||^2 + C\sum_i \max(0, \max_{\mathbf{y}'} 1 + \langle \mathbf{w}, \Phi(\mathbf{x},\mathbf{y}')\rangle - \langle \mathbf{w}, \Phi(\mathbf{x},\mathbf{y})\rangle)$

- Dual formulation

$$\max_{\boldsymbol{\alpha}} \quad \sum_{i=1}^{n} \sum_{\bar{\mathbf{y}} \neq \mathbf{y}_i} \alpha_i(\bar{\mathbf{y}}) - \frac{1}{2} \sum_{i=1}^{n} \sum_{\bar{\mathbf{y}} \neq \mathbf{y}_i} \sum_{j=1}^{n} \sum_{\bar{\mathbf{y}}' \neq \mathbf{y}_j} \alpha_i(\bar{\mathbf{y}}) \alpha_j(\bar{\mathbf{y}}') K(i, \bar{\mathbf{y}}, j, \bar{\mathbf{y}}')$$

$$s.t. \quad \forall_{i=1}^{n} \forall_{\bar{\mathbf{y}} \neq \mathbf{y}_i} \; \alpha_i(\bar{\mathbf{y}}) \geq 0$$

- where

$$\begin{aligned} K(i, \bar{\mathbf{y}}, j, \bar{\mathbf{y}}') &= \langle \Phi(\mathbf{x}_i, \mathbf{y}_i) - \Phi(\mathbf{x}_i, \bar{\mathbf{y}}), \Phi(\mathbf{x}_j, \mathbf{y}_j) - \Phi(\mathbf{x}_j, \bar{\mathbf{y}}') \rangle \\ &= \langle \Phi(\mathbf{x}_i, \mathbf{y}_i), \Phi(\mathbf{x}_j, \mathbf{y}_j) \rangle - \langle \Phi(\mathbf{x}_i, \bar{\mathbf{y}}), \Phi(\mathbf{x}_j, \mathbf{y}_j) \rangle \\ &\quad - \langle \Phi(\mathbf{x}_i, \bar{\mathbf{y}}), \Phi(\mathbf{x}_j, \mathbf{y}_j) \rangle + \langle \Phi(\mathbf{x}_i, \bar{\mathbf{y}}), \Phi(\mathbf{x}_j, \bar{\mathbf{y}}') \rangle \end{aligned}$$

- Structured SVM minimizes hinge loss
  $\ell(f, \mathbf{x}, \mathbf{y}) = \max(0, \max_{\mathbf{y}'} 1 + \langle \mathbf{w}, \Phi(\mathbf{x}, \mathbf{y}') \rangle - \langle \mathbf{w}, \Phi(\mathbf{x}, \mathbf{y}) \rangle)$
- Upper bounds 0/1 loss
- BUT: 0/1 loss not appropriate in structured predictions
  True output: $\mathbf{y} = \langle N, V, D, N \rangle$
  Predictions: $\mathbf{y}_1 = \langle N, V, D, V \rangle$ and $\mathbf{y}_2 = \langle P, P, P, P \rangle$.
- Measure error by structured loss function: $\Delta : \mathcal{Y} \times \mathcal{Y} \rightarrow \Re$
  Incorporate structural loss either by rescaling the margin (confidence) or the slack variables (error).

SO-SVM: Margin Rescaling

- Structured SVM minimizes hinge loss with margin rescaling $\ell(f, \mathbf{x}, \mathbf{y}) = \max(0, \max_{\mathbf{y}'} \Delta(\mathbf{y}, \mathbf{y}') + \langle \mathbf{w}, \Phi(\mathbf{x}, \mathbf{y}')\rangle - \langle \mathbf{w}, \Phi(\mathbf{x}, \mathbf{y})\rangle)$
- Primal constraints $\forall_{i=1}^{n} \forall_{\bar{\mathbf{y}} \neq \mathbf{y}_i} : \langle \mathbf{w}, \Phi(\mathbf{x}_i, \mathbf{y}_i) - \Phi(\mathbf{x}_i, \bar{\mathbf{y}})\rangle \geq \boxed{\Delta(\mathbf{y}_i, \bar{\mathbf{y}})} - \xi_i$
- Dual Objective (first term) $\max_{\boldsymbol{\alpha}} \sum_{i=1}^{n} \sum_{\bar{\mathbf{y}} \neq \mathbf{y}_i} \alpha_i(\bar{\mathbf{y}}) \boxed{\Delta(\mathbf{y}_i, \bar{\mathbf{y}})} - \frac{1}{2} \dots$
- Still decomposable (necessary for e.g. Viterbi)
- But Δ may dominate the loss!

SO-SVM: Slack Rescaling

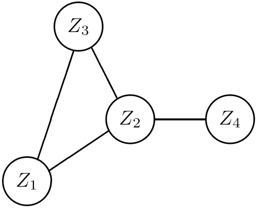
- Structured SVM minimizes hinge loss with margin rescaling
  $\ell(f, \mathbf{x}, \mathbf{y}) = \max(0, \max_{\mathbf{y}'} \Delta(\mathbf{y}, \mathbf{y}')(1 - \langle \mathbf{w}, \Phi(\mathbf{x}, \mathbf{y}')\rangle - \langle \mathbf{w}, \Phi(\mathbf{x}, \mathbf{y})\rangle))$
- Primal constraints $\forall_{i=1}^{n} \forall_{\bar{\mathbf{y}} \neq \mathbf{y}_i} : \langle \mathbf{w}, \Phi(\mathbf{x}_i, \mathbf{y}_i) - \Phi(\mathbf{x}_i, \mathbf{y})\rangle \geq 1 - \boxed{\frac{\xi_i}{\Delta(\mathbf{y}_i, \bar{\mathbf{y}})}}$
- Addition dual constraints occur $\boxed{\forall_{i=1}^{n} \sum_{\bar{\mathbf{y}} \neq \mathbf{y}_i} \frac{\alpha_i(\bar{\mathbf{y}})}{\Delta(\mathbf{y}_i, \bar{\mathbf{y}}} \leq C}$
- Empirically a good choice
- But argmax may be hard to compute (non-linear)!

SO-SVM: Optimization

- Input $(\mathbf{x}_1,\mathbf{y}_1),\ldots,(\mathbf{x}_n,\mathbf{y}_n)$, $C>0$, $\varepsilon>0$
- $S_i = \varnothing$ for $i=1,\ldots,n$
- Repeat
  - For $i = 1,\ldots,n$ do
    - $H(\bar{\mathbf{y}}) = \begin{cases} 1 - \langle \mathbf{w}, \Phi(\mathbf{x}_i,\mathbf{y}_i) - \Phi(\mathbf{x}_i,\bar{\mathbf{y}})\rangle & \text{SVM}^{0/1} \\ \Delta(\mathbf{y}_i,\mathbf{y})(1 - \langle \mathbf{w}, \Phi(\mathbf{x}_i,\mathbf{y}_i) - \Phi(\mathbf{x}_i,\mathbf{y})\rangle) & \text{SVM}^{slack} \\ \Delta(\mathbf{y}_i,\bar{\mathbf{y}}) - \langle \mathbf{w}, \Phi(\mathbf{x}_i,\mathbf{y}_i) - \Phi(\mathbf{x}_i,\bar{\mathbf{y}})\rangle & \text{SVM}^{margin} \end{cases}$
    - Compute cutting plane: $\hat{\mathbf{y}} = \operatorname{argmax}_{\bar{\mathbf{y}}} H(\bar{\mathbf{y}})$
    - Determine actual slack: $\xi_i = \max\{0, \max_{\mathbf{y}' \in S_i} H(\mathbf{y}')\}$
    - If $H(\hat{\mathbf{y}}) > \xi_i + \varepsilon$ then
      - Add constraint to working set $S_i \leftarrow S_i \cup \{\hat{\mathbf{y}}\}$
      - Optimize α_S (solve dual optimization problem)
    - end
  - end
- Until no S_i has changed during iteration.

Markov Random Fields & Conditional Random Fields

Conditional Independence

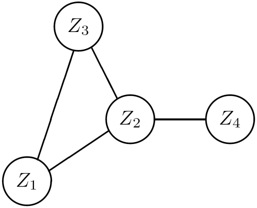


- Encode dependency structure of a given problem by a graph.
- Two random variables are connected with an edge if they directly depend on each other.
- Two unconnected variables are independent given the value of all other variables.

Conditional Independence

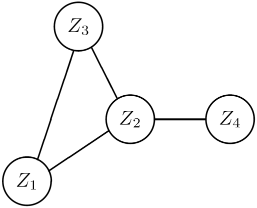
Given (sets of) random variables A, B, C: We say A is conditionally independent of B given C, and write $A \perp B | C$, if for any valid assignment $B = b$ and $C = c$ the relation $P(A|B = b, C = c) = P(A|C = c)$ holds.

Conditional Independence



- Consider the set of discrete random variables $V = \{Z_1, \ldots, Z_4\}$.
- Let $\mathcal{G} = (V, E)$ encode pairwise dependencies between variables V.
- Knowing the actual value of Z_2, variable Z_4 is independent of Z_1 and Z_3.
- We write $Z_4 \perp \{Z_1, Z_3\} | Z_2$.
- The joint probability can be written as

$$p(V) = p(Z_1, Z_3 | Z_2) p(Z_4 | Z_2) p(Z_2). \tag{1}$$

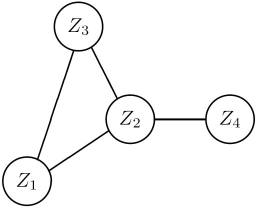


Markov Random Field

A collection V of random variables over a finite domain with joint probability P and fulfilling Equation 2 with respect to an undirected graph $\mathcal{G}$ is said to be a Markov random field (MRF).

$$\forall i, j,\ e_{ij} \notin E : Z_i \perp Z_j | V \backslash \{Z_i, Z_j\}. \tag{2}$$

Hammersley & Clifford Theorem



- Every MRF $V = (Z_1, \ldots, Z_n)$ has a Gibbs distribution wrt $\mathcal{G}$:

$$p(Z_1 = z_1, \ldots, Z_n = z_n) = \exp\left\{ \sum_{C \in \mathcal{C}} \langle \boldsymbol{\lambda}_C, \Phi_C(\mathbf{z}_C) \rangle - \log Z \right\}$$

- $\mathbf{z}_C$ denotes the restriction of a valid assignment $\mathbf{z} = (z_1, \ldots, z_n)$ on the maximal cliques $C \in \mathcal{C}$ of $\mathcal{G}$
- Φ_C are feature functions defined on maximal cliques.
- Partition function $Z = \sum_{\mathbf{z}} \exp\{\sum_{C \in \mathcal{C}} \langle \boldsymbol{\lambda}_C, \Phi_C(\mathbf{z}_C) \rangle\}$ (normalization).

- MRFs can be written as a member in the exponential family

$$p(\mathbf{z}|\boldsymbol{\lambda}) = \exp\{\langle \boldsymbol{\lambda}, \Phi(\mathbf{z})\rangle - g(\boldsymbol{\lambda})\}, \quad \boldsymbol{\lambda} \in \Lambda.$$

- $\Phi(\mathbf{z})$ denotes the sufficient statistics.
- $\boldsymbol{\lambda} \in \Lambda$ is the natural parameter.
- The domain Λ consists of all $\boldsymbol{\lambda}$ having the log-partition function

$$g(\boldsymbol{\lambda}) = \log \sum_{\mathbf{z}} \exp\{\langle \boldsymbol{\lambda}, \Phi(\mathbf{z})\rangle\} < \infty.$$

- The log-partition function is also the moment generating function of the exponential family:

$$\frac{\partial}{\partial \boldsymbol{\lambda}} g(\boldsymbol{\lambda}) = \mathbf{E}_{p(\mathbf{z}|\boldsymbol{\lambda})}[\Phi(\mathbf{z})], \quad \frac{\partial^2}{\partial \boldsymbol{\lambda} \partial \boldsymbol{\lambda}} g(\boldsymbol{\lambda}) = \mathbf{Cov}_{p(\mathbf{z}|\boldsymbol{\lambda})}[\Phi(\mathbf{z})], \quad \ldots$$

- Given structured data $\mathcal{D} = \{(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_n, \mathbf{y}_n)\}$, find parameters $\boldsymbol{\lambda}$ by maximizing the likelihood $\mathcal{L}$,

$$\mathcal{L}(\boldsymbol{\lambda}) = \prod_{i=1}^{n} p(\mathbf{y}_i | \mathbf{x}_i, \boldsymbol{\lambda}) = \prod_{i=1}^{n} \exp\left\{\langle \boldsymbol{\lambda}, \Phi(\mathbf{x}_i, \mathbf{y}_i)\rangle - g(\boldsymbol{\lambda}|\mathbf{x}_i)\right\}$$

  with $g(\boldsymbol{\lambda}|\mathbf{x}_i) = \log \sum_{\mathbf{y}} \exp\{\langle \boldsymbol{\lambda}, \Phi(\mathbf{x}_i, \mathbf{y})\rangle\}$.
- The log-likelihood (that has to maximized wrt $\boldsymbol{\lambda}$) is given by

$$\log \mathcal{L}(\boldsymbol{\lambda}) = \sum_{i=1}^{n} \langle \boldsymbol{\lambda}, \Phi(\mathbf{x}_i, \mathbf{y}_i)\rangle - g(\boldsymbol{\lambda}|\mathbf{x}_i). \tag{3}$$

- The gradient wrt parameter vector $\boldsymbol{\lambda}$ is

$$\frac{\partial}{\partial \boldsymbol{\lambda}} \log \mathcal{L} = \mathbf{E}_{\hat{p}(X,Y)}[\Phi(X, Y)] - \sum_{i=1}^{n} \mathbf{E}_{p(Y|\mathbf{x}_i;\boldsymbol{\lambda})}[\Phi(Y, \mathbf{x}_i)].$$

- BUT: Maximum likelihood $\rightarrow$ bad generalization performance for high-dimensional problems.
- Remedy: Incorporate prior on the weights to...
  - ... express beliefs about parameters before looking at the data.
  - ... promote sparse models, having zero weights for redundant features.
- Apply a zero mean Gaussian prior with variance σ^2 on $\boldsymbol{\lambda}$, such that

$$\boldsymbol{\lambda} \sim N(\mathbf{0}, \mathbb{1}\sigma^2).$$

- Bayes Theorem says

$$\text{posterior} = \frac{\text{likelihood} \times \text{prior}}{\text{marginal likelihood}}.$$

- We obtain

$$\log p(\boldsymbol{\lambda}|\mathcal{D}) = \sum_{i=1}^{n} \left[\langle \boldsymbol{\lambda}, \Phi(\mathbf{x}_i, \mathbf{y}_i) \rangle - g(\boldsymbol{\lambda}|\mathbf{x}_i) \right] - \log 2\pi\sigma - \frac{\boldsymbol{\lambda}^{\mathsf{T}}\boldsymbol{\lambda}}{2\sigma^2}. \quad (4)$$

Interpretation

- KCRF: Maximize (log) posterior distribution of parameters:

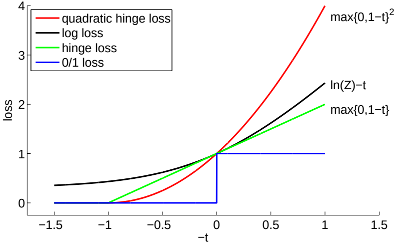
$$\log p(\boldsymbol{\lambda}|\mathcal{D}) = \sum_{i=1}^{n} \left[\langle \boldsymbol{\lambda}, \Phi(\mathbf{x}_i, \mathbf{y}_i)\rangle - g(\boldsymbol{\lambda}|\mathbf{x}_i)\right] - \underbrace{\log 2\pi\sigma}_{\text{constant}} - \frac{\boldsymbol{\lambda}^\mathsf{T}\boldsymbol{\lambda}}{2\sigma^2} \to \max$$

- Rewriting and omitting constant terms leads to

$$-\log p(\boldsymbol{\lambda}|\mathcal{D}) \propto \underbrace{\sigma^2 \sum_{i=1}^{n} \left[g(\boldsymbol{\lambda}|\mathbf{x}_i) - \langle \boldsymbol{\lambda}, \Phi(\mathbf{x}_i, \mathbf{y}_i)\rangle\right]}_{\text{empirical risk (log-loss)}} + \underbrace{\frac{1}{2}\|\boldsymbol{\lambda}\|^2}_{\text{regularization}} \to \min$$

- Maximizing the (conditional) likelihood = minimizing log-loss!
- Incorporating priors = regularization!
- Variance σ^2 acts as a trade-off parameter!

Log-loss vs. Hinge-loss



Exemplary loss functions with $t = f(\mathbf{x}, \mathbf{y}) - \max_{\hat{\mathbf{y}})} f(\mathbf{x}, \hat{\mathbf{y}})$.
Log-loss is shifted to pass through the $(0, 1)$ *coordinate.*

- Both, log-loss and hinge-loss upper bound 0/1-loss.
- Log-loss assigns penalties to all training examples
  - $\Rightarrow$ non-sparse solutions!
- Hinge-loss depends only on misclassified instances
  - $\Rightarrow$ sparse solutions!

- Maximizing Equation 3 (CRF) and 4 (kCRF) is expensive because of the computation of the partition function $g(\boldsymbol{\lambda}|\mathbf{x})$.
- Possible optimization strategies have been proposed:
  - Linear programming
  - Iterative scaling
  - Conjugate gradients
  - Gauss-Newton subspace optimizations
  - Gradient tree boosting
  - Stochastic meta descent

Empirical Results + Summary + References

Part-of-speech tagging (1)

- Annotate input sentence with part-of-speech tags:
  $\mathbf{x} = \langle \text{Bello chases the cat} \rangle \longrightarrow \mathbf{y} = \langle N, V, Det, N \rangle$
  Penn treebank corpus.
- Experimental setup:
  Traning sets: 500, 1000, 2000, 4000, and 8000 sentences.
  10% of training set used as validation set.
  Independent test set of 1600 sentences.
- Methods:
  HMM, CRF, structured perceptron, structured SVM
- Features:
  HMM: transition and emission probabilities.
  All others: transition counts and 450.000 lexical emission features
  e.g., [[previous word ends with 'action' $\wedge\ y_t = \sigma$]]

Part-of-speech tagging (2)

Train Size	500	1000	2000	4000	8000
HMM	23.46	19.95	17.96	17.58	15.87
CRF	16.53	12.51	9.84	7.76	6.38
Perceptron	10.16	7.79	6.38	5.39	4.49
SVM	8.37	6.58	5.75	4.71	4.08

(Nguyen & Guo, 07)

- discriminative methods outperform generative methods
- SO-SVM outperforms CRF

Named-Entity-Recognition (1)

- Task: detect named entities in text.
- Example:

„Der FIFA Praesident Sepp Blatter und Franz Beckenbauer haben am letzten Samstag in Muenchen ...“

⇩

„Der FIFA Praesident Sepp Blatter und Franz Beckenbauer haben am letzten Samstag in Muenchen ...“

- Entities: organization, person, location, timex, …
- BIO-encoding: beginning (B), inside (I), outside (O).
- **y** = „O ORG-B O PER-B PER-I O PER-B PER-I O O O TIM-B O LOC-B...“

Named-Entity-Recognition (2)

- CoNLL2002 data set:

  300 sentences from a Spanish News wire article corpus.

  9 Labels in BIO encoding:

  - Person (beginning/inside).

  - Organization (beginning/inside).

  - Location (beginning/inside).

  - Miscellaneous names (beginning/inside).

  - Outside

- Algorithms:

  HMM, CRF, structured perceptron, structured $SVM_{0/1}$

- Features:

  HMM: transition and emission probabilities

  All other methods: transitions + emission counts

- Inference/Decoding:

  All methods: Viterbi algorithm

Named-Entity-Recognition (3)

Method	HMM	CRF	Perceptron	SVM
Error	9.36	5.17	5.94	5.08

(Altun et al., 03)

- discriminative methods outperform generative methods
- SO-SVM and CRF almost equal

Summary

- Structured prediction models
  - Allow to address naturally arising problems!
- Structural SVMs
  - 0/1 loss: $SVM_{0/1}$
  - Arbitrary loss functions: SVM_{margin}, SVM_{slack}
  - Sparse models, convergence in polynomial time.
  - Only joint feature mapping and argmax need to be adapted to problem at hand!
- Empirical Results
  - Discriminative models outperform generative ones.
  - Structural SVM is state-of-the-art for structural problems.

References

- I. Tsochantaridis, T. Joachims, T. Hofmann, Y. Altun. Large margin methods for structured and interdependent output variables, JMLR 6, 2005.
- B. Taskar, C. Guestrin, and D. Koller. Max-margin Markov networks, Advances in Neural Information Processing Systems 16, 2004.
- Y. Altun, I. Tsochantaridis, and T. Hofmann. Hidden Markov support vector machines, Proceedings of the International Conference on Machine Learning, 2003.
- M. Collins and N. Duffy. Convolution kernels for natural language. In Advances in Neural Information Processing Systems 14, 2002.
- J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data, Proceedings of the International Conference on Machine Learning, 2001.
- N. Nguyen and Y. Guo. Comparisons of Sequence Labeling Algorithms and Extensions. Proceedings of the International Conference on Machine Learning, 2007.