

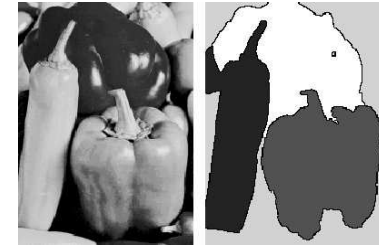
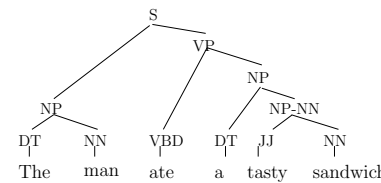
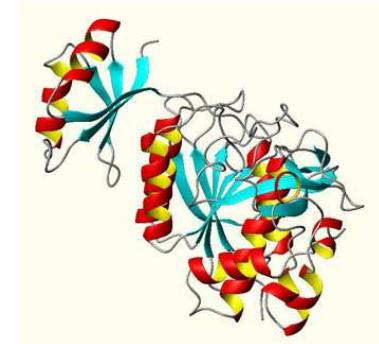
Structured Prediction

Hal Daumé III

CS5350: Machine Learning

02 Dec 2008

What are complex structures?



Problem formulation

Informally, given:

1. A bunch of inputs
2. Correct corresponding outputs

Induce a function that maps novel inputs to corresponding outputs.

Formally, given:

1. An input space \mathcal{X}
2. An output space \mathcal{Y} (that is “hairy”)
3. A distribution \mathcal{D} over $\mathcal{X} \times \mathcal{Y}$
4. A loss function $\ell : \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R}^+$

Induce function $f : \mathcal{X} \rightarrow \mathcal{Y}$ with low expected loss (with respect to \mathcal{D})

Example 1: Sequence labeling

| | | | | | | | |
|--------|-----|------|------|-----|--------|-------|----------|
| Input | The | man | ate | the | really | tasty | sandwich |
| Output | DET | NOUN | VERB | DET | ADV | ADJ | NOUN |

Inputs are sequences; outputs are (equal-length) sequences of labels drawn from a label set.

Three approaches:

- ▶ Independent prediction: Ignore the structure and predict each label independently
- ▶ Global prediction: Learn a score $f : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}^+$ so that good ys have high score
- ▶ Productive prediction: Learn a function that generates the output sequentially

Independent Prediction

Method 1: Independent prediction

| | | | | | | | |
|--------|---------|---------|---------|---------|---------|---------|----------|
| | $t = 1$ | $t = 2$ | $t = 3$ | $t = 4$ | $t = 5$ | $t = 6$ | $t = 7$ |
| Input | The | man | ate | the | really | tasty | sandwich |
| Output | DET | NOUN | VERB | DET | ADV | ADJ | NOUN |

Features may depend on any of the input, but must *decompose entirely over the output*:

$$f(x, y) = \sum_{t=1}^T \mathbf{w}^\top \Phi(x, y_t)$$

One may use any classifier, for instance:

- ▶ Logistic regression (aka maximum entropy) classifiers
- ▶ Support vector machines:
- ▶ ...

Pros:

- ▶ Really efficient at training and test time
- ▶ Can use any off the shelf multiclass classifier

Independent prediction: Enforcing Constraints

For instance:

| | | | | | | |
|--------|--------|-------|-------|----|----------|-------|
| Input | George | Bush | spoke | to | Congress | today |
| Output | B-PER | I-PER | O | O | B-ORG | O |

Must ensure that I-PER always follows B-PER or I-PER.

- ▶ At test time, constraints can be enforced by (eg) integer programming
- ▶ Incurs additional test-time complexity
- ▶ See [Punyakanok and Roth; IJCAI 2005]

Global prediction

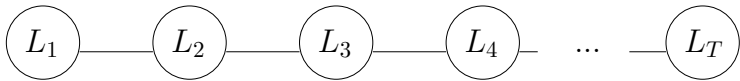
Goal: Learn a score $f : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}^+$ so that good y s have high score

Three issues:

- ▶ How to represent f
- ▶ How to learn f from a finite sample
- ▶ Given f and a new input x , how to find best y

Global prediction: function representation

Convenient representation: y is a graph.



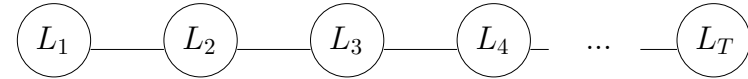
Define **features** over the **cliques** of the graph: $\Phi(x, c) \in \mathbb{R}^D$

Represent f as linear with weights \mathbf{w} :

$$f(x, y) = \sum_{c \in \mathcal{Y}} \mathbf{w}^\top \Phi(x, c)$$

Function representation: sequence labeling

| | | | | | | | |
|---------------|-----|------|------|-----|--------|-------|----------|
| Input | The | man | ate | the | really | tasty | sandwich |
| Output | DET | NOUN | VERB | DET | ADV | ADJ | NOUN |



Choose cliques as adjacent labels.

► Allowed features:

- Is the tag “VERB” associated with the word “ate”?
- Do we assign “DET” to a single-letter word?
- Does y contain the sequence “DET NOUN”?

► Disallowed features:

- Does y contain the sequence “DET NOUN VERB”?
- How many VERBs are there in the label sequence?
- Are all instances of the word “the” assigned the same label?

Global prediction: How to predict

Prediction problem: given new point \hat{x} and weights \mathbf{w} , compute:

$$\hat{y} = \arg \max_{y \in \mathcal{Y}} \sum_{c \in \mathcal{Y}} \mathbf{w}^\top \Phi(x, c)$$

Under what circumstances can we solve this exactly?

- Roughly, if Φ only models *local* structure
- Formally, scales exponentially in the *tree-width* of the graph
- For linear chains, Viterbi search in time $\mathcal{O}(TK^{|C|})$
- For Ising models, NP-hard (graph cut)
- For bipartite matchings, polynomial time (Hungarian algorithm)

Global prediction: how to learn (I)

Conditional random fields: model outputs *probabilistically*

$$p(y | x; \mathbf{w}) = \frac{1}{Z(x, \mathbf{w})} \exp \left[\sum_{c \in \mathcal{Y}} \mathbf{w}^\top \Phi(x, c) \right]$$

$$Z(x, \mathbf{w}) = \sum_{y' \in \mathcal{Y}} \exp \left[\sum_{c \in \mathcal{Y}'} \mathbf{w}^\top \Phi(x, c) \right]$$

See [Lafferty, McCallum and Pereira; ICML 2001]

Can be optimized using, eg., limited memory Hessian methods, (stochastic) gradient descent, etc.

Intuition: make correct output look good, all others look bad

Requires efficient computation of normalizer!

Global prediction: how to learn (II)

Instead of forcing **all** other outputs to look bad, just look at the best!

Leads to **maximum margin Markov networks**

$$\min_w \frac{1}{2} \|\mathbf{w}\|^2$$

ensures large margin

$$\text{subject to } \underbrace{f(x_n, y_n) - f(x_n, y')}_{\text{predicted scores}} \geq \underbrace{\ell(y_n, y')}_{\text{cost of error}} \quad (\forall n, y')$$

See [Taskar, Guestrin and Koller; NIPS 2002]

Looks like exponentially many constraints, but can be reduced (if clever) to depend on the tree-width.

(Technically, must introduce slack variables.)

Can we trade structure for features?

| | | | | | | | |
|--------|-----|------|------|-----|--------|-------|----------|
| Input | The | man | ate | the | really | tasty | sandwich |
| Output | DET | NOUN | VERB | DET | ADV | ADJ | NOUN |

Idea: instead of using VERB-DET as a feature, just add extra *input features* and use independent predictions.

Problem:

- ▶ This introduces a ton new features, whose weights we need to estimate
- ▶ Probably don't have enough data to do this reliably
- ▶ CRFs are **computationally complex** but **statistically simple**; IRL is **computationally simple** but **statistically complex**

Solution:

- ▶ "Structure Compilation" [Liang, Daumé and Klein, ICML 2008]

Global prediction: Summary

Your task:

- ▶ Represent output structure as a graph
- ▶ Define features over cliques

Prediction can be solved efficiently for many reasonable problems:

- ▶ chains, trees, bipartite matchings, graph cuts (sort of)

CRF normalization efficient for many problems:

- ▶ chains, trees

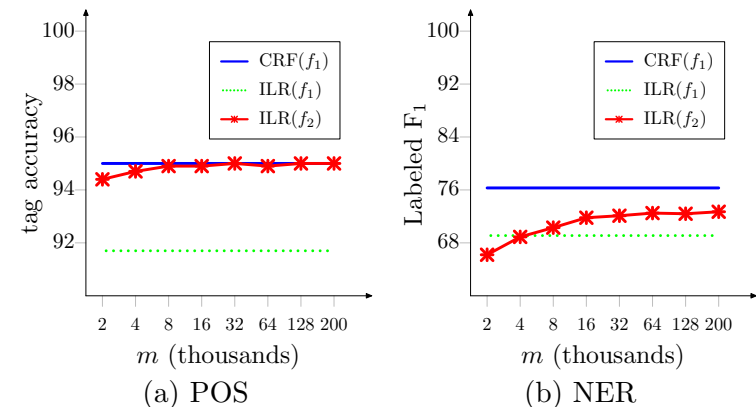
M³N constraints polynomial for more problems:

- ▶ chains, trees, bipartite matchings

Structure Compilation

Simple idea/algorithm:

- ▶ Train a CRF on your labeled data
- ▶ Run this CRF on a large amount of unlabeled data
- ▶ Train an IRL on the data labeled by the CRF, using more features



Productive prediction: Striking a middle ground

Productive prediction: adding history to classifiers

Idea: Follow independent classifier methodology, but:

- ▶ Predict variables in a prescribed order
- ▶ Allow features to depend on *any* past decision

| | | | | | | | |
|--------|-----|------|------|-----|--------|-------|----------|
| Input | The | man | ate | the | really | tasty | sandwich |
| Output | DET | NOUN | VERB | DET | ADV | ADJ | NOUN |

▶ **At training time:**

1. Make a classification example for DET
2. Make an example for NOUN, knowing DET
3. Make an example for VERB, knowing DET NOUN
4. Make an example for DET, knowing DET . . . VERB
5. And so on . . .

▶ **At test time:**

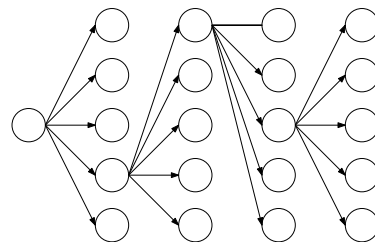
1. Predict the first label y_1
2. Predict the second y_2 , knowing y_1
3. Predict y_3 , knowing y_1, y_2
4. And so on . . .

What could go wrong in this picture?

Structured prediction via search

Key Idea: view prediction task as search

- ▶ A **path** corresponds to a full output
- ▶ Each **decision** is over a small set of options
- ▶ Train a **classifier** to make search predictions



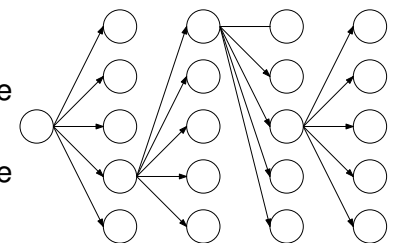
How can we train this classifier?

- ▶ **Train on every node in the search space**
This is effectively what CRFs do
- ▶ **Train only on the correct path**
This incurs label-bias problem
See [Daumé & Marcu; ICML 2005] and [Xu and Fern, ICML 2007]
- ▶ **Train on the subset of states that we are likely to reach!**
How do we know what states we are likely to reach?
Chicken-and-egg problem!

SEARN: Search-based structured prediction

Idea: Train on the subset of states that we are likely to reach

- ▶ Iterative algorithm
- ▶ First train on the correct path $\Rightarrow h^{(1)}$
- ▶ Then on an interpolation between the best path and $h^{(1)} \Rightarrow h^{(2)}$
- ▶ Then on an interpolation between the best path and $h^{(1)}$ and $h^{(2)} \Rightarrow h^{(3)}$
- ▶ And so on . . .



Guaranteed to converge in a polynomial number of iterations to a model with regret:

$$R(\mathcal{D}, h_{\text{last}}) \leq 2T \ln T \ell_{\text{avg}} + \frac{1 + \ln T}{T} C_{\text{max}}$$

See [Daumé III, Langford and Marcu; MLJ 2007]

Experimental comparison

Summary of experimental results

Part of Speech Tagging: Similar performance; speed is differentiator
[Daumé III and Marcu; ICML 2005], [Punyakanok and Roth; IJCAI 2005]

Named-entity Recognition: Independent classifiers worse
[McCallum and Li; CoNLL 2003], [Tsochantaridis, Hoffmann, Joachims & Altun; JMLR 2005],
[Liang, Daumé III & Klein; ICML 2008], [Daumé III, Langford & Marcu; MLJ 2007]

Machine translation: Independent classifiers & global impossible
[Liang, Bouchard-Côté, Klein & Taskar; ACL 2006]

Coreference resolution/clustering: Global/predictive best
[McCallum and Wellner; NIPS 2004], [Daumé III and Marcu; NAACL 2005],
[Finley and Joachims; ICML 2005]

Bipartite matching: CRFs impossible; M^3 Ns outperform classifiers
[Taskar, Lacoste-Julien & Jordan; JMLR 2006]

Image segmentation: CRFs & M^3 Ns state of the art performance
[Kumar and Herbert; NIPS 2004], [Wang and Ji, CVPR 2005]

Summary

Problem: Learn to predict complex structures from examples

Three solutions:

1. Enumerate over all possible structures
Useful for simple problems, or small data sets
2. Ignore structure entirely
Useful when structure is not particularly useful
Can recover somewhat through “structure compilation”
3. Compute structure piecewise
Useful for large data when structure can be sequentialized