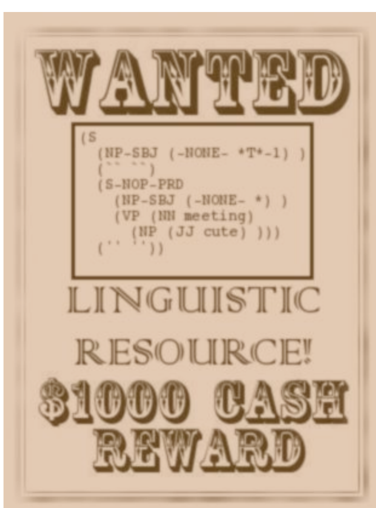
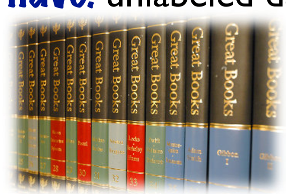
Rich Prior Knowledge in Learning for NLP

Gregory Druck, Kuzman Ganchev, João Graça

Why Incorporate Prior Knowledge?



have: unlabeled data



option: hire

linguist

annotators

Why Incorporate Prior Knowledge?

This approach does not scale to every task and domain of interest.

have: unlabeled data

option: hire

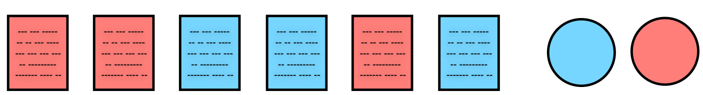
However, we already know a lot about most problems of interest.

linguist

annotators

Example: Document Classification

Documents   Labels



- **Prior Knowledge:**
  - <u>labeled features</u>: information about the label distribution when word ***w*** is present

sentiment polarity

positive	**negative**
memorable	terrible
perfect	boring
exciting	mess

newsgroups classification

baseball	**Mac**	**politics**	**...**
hit	Apple	senate	...
Braves	Macintosh	taxes	...
runs	Powerbook	liberal	...

Example: Information Extraction

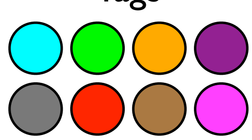
extraction from research papers:

W. H. Enright. Improving the efficiency of matrix operations in the numerical solution of stiff ordinary differential equations. ACM Trans. Math. Softw., 4(2), 127-136, June 1978.

- **Prior Knowledge:**
  - labeled features:
    - *the word* ***ACM*** *should be labeled either* ***journal*** *or* ***booktitle*** *most of the time*
  - non-Markov (long-range) dependencies:
    - *each reference has at most one segment of each type*

Example: Part-of-speech Induction

Tags

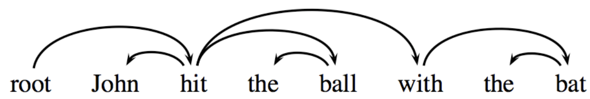


Text

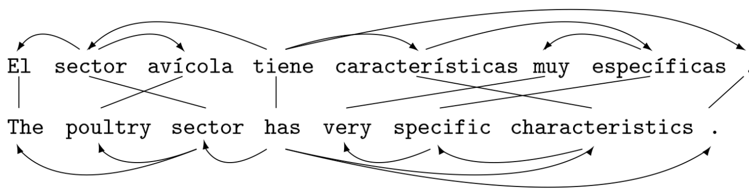
A career with the European institutions must become more attractive. Too many young, new...

- **Prior Knowledge:**
  - linguistic knowledge: *each sentence should have a* ***verb***
  - posterior sparsity: *the total number of different POS tags assigned to each word type should be small*

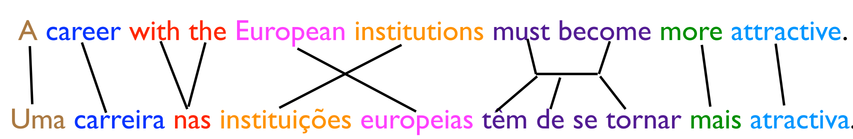
Example: Dependency Grammar Induction



- ***Prior Knowledge:***
  - <u>linguistic rules</u>: ***nouns*** *are usually dependents of* ***verbs***
  - <u>noisy labeled data</u>: *target language parses should be similar to aligned parses in a resource-rich source language*



Example: Word Alignment



- ***Prior Knowledge:***
  - <u>Bijectivity</u>: *alignment should be mostly one-to-one*
  - <u>Symmetry</u>: ***source→target*** *and* ***target→source*** *alignments should agree*

This Tutorial

In general, how can we leverage such knowledge and an unannotated corpus during learning?

Notation & Models

input variables (documents, sentences):	$\mathbf{x}$
structured output variables (parses, sequences):	$\mathbf{y}$
unstructured output variables (labels):	y
input / output variables for entire corpus:	$\mathbf{X}$ $\mathbf{Y}$
probabilistic model parameters:	θ
generative models:	$p_\theta(\mathbf{x}, \mathbf{y})$
discriminative models:	$p_\theta(\mathbf{y} \vert \mathbf{x})$
model feature function:	$\mathbf{f}(\mathbf{x}, \mathbf{y})$

Learning Scenarios

- **Unsupervised:**
  - unlabeled data + prior knowledge
- **Lightly Supervised:**
  - unlabeled data + "informative" prior knowledge
  - i.e. provides specific information about labels
- **Semi-Supervised:**
  - labeled data + unlabeled data + prior knowledge

Running Example #1: Document Classification

- **model:** Maximum Entropy Classifier (Logistic Regression)

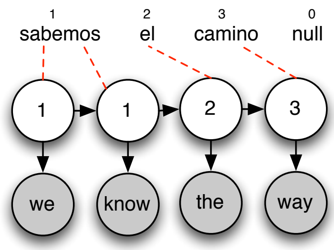
$$p_\theta(y|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \exp(\theta \cdot \mathbf{f}(\mathbf{x}, y))$$

- **setting:** lightly supervised; *no labeled data*
- **prior knowledge:**
  - labeled features: information about the label distribution when word ***w*** is present
  - label is *often* hockey or baseball when *game* is present

Running Example #2: Word Alignment

- **model:** first-order Hidden Markov Model (HMM)

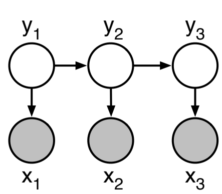
$$p_\theta(\mathbf{y}, \mathbf{x}) = p_\theta(y_0) \prod_{i=1}^{N} p_\theta(y_i | y_{i-1}) p_\theta(x_i | y_i)$$



- **setting:** unsupervised
- **prior knowledge:**
  - Bijectivity: *alignment should be mostly one-to-one*

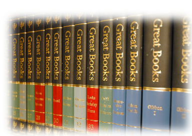
Problem

model



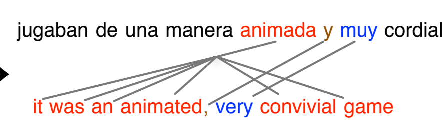
+

data



$\Rightarrow$

output

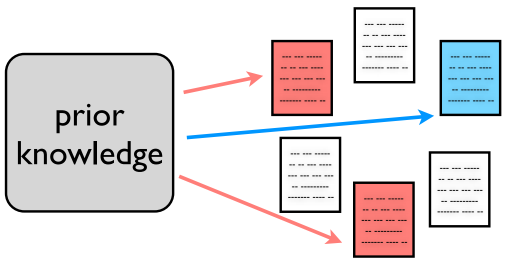


This output does not agree with prior knowledge!

- six target words align to source word animada
- five source words do not align with any target word

Limited Approach: Labeling Data

approach: *Convert prior knowledge to labeled data.*



Prototypes (+ cluster features):
- [Haghighi & Klein 06]

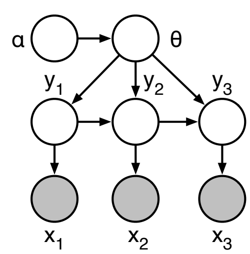
Others:
- [Raghavan & Allan 07]
- [Schapire et al. 02]

limitation: Often unclear how to do conversion

- **Example #1:** often (not always) *game* → {hockey,baseball}
- **Example #2:** *alignment should be mostly one-to-one*

Limited Approach: Bayesian Approach

approach: *Encode prior knowledge with a prior on parameters.*



specifying $p(\theta)$

natural: "θ should be *small* (or *sparse*)"
[Johnson 07], among many others

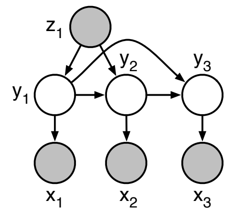
possible: "θ_i should be close to $\tilde{\theta}_i$"
(informative prior) [Dayanik et al. 06]

limitation: Our prior knowledge is not about parameters!
Parameters are difficult to interpret; hard to get desired effect.

- **Example #1:** often (not always) *game* → {hockey,baseball}
- **Example #2:** *alignment should be mostly one-to-one*

Limited Approach: Augmenting Model

approach: *Encode prior knowledge with additional variables and dependencies.*



limitation: can be difficult to get desired effect

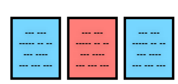
- **Example #1:** often (not always) *game* → {hockey,baseball}

limitation: may make exact inference intractable

- **Example #2:** *Bijectivity* makes inference #*P*-complete

This Tutorial

develop:

- a **language** for *directly* encoding prior knowledge
- **methods for learning** with knowledge in this language
  - (approximations to modeling this language directly)
- (*loosely*) these methods **perform mappings for us**:
  - encoded prior knowledge $\rightsquigarrow$ parameters θ
  - encoded prior knowledge $\rightsquigarrow$ labeling 

A Language for Encoding Prior Knowledge

Our prior knowledge is about **distributions over latent output variables**. (output variables are interpretable)

Specifically, we know some properties of this distribution:

- **Example #1:** often (not always) *game*→{hockey,baseball}

Formulation: know about the **expectations** of some functions under distribution over latent output variables

Constraint Features

- **constraint feature function:** $\phi(\mathbf{x}, \mathbf{y})$
- **Example #1:** $\phi_w(\mathbf{x}, y) = \mathbf{1}(y = l)1(w \in \mathbf{x})$
  - for document **x**, returns a vector with a 1 in the *l*th position if y is the *l*th label and the word *w* is in **x**
- **Example #2:** $\phi(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{N} \mathbf{1}(y_i = m)$
  - returns a vector with *mth* value = number of target words in sentence **x** that align with source word *m*

Expectations of Constraint Features

- **Example #1: Corpus expectation:**

$$\mathbf{E}_{p_\theta}[\phi(\mathbf{X},\mathbf{Y})] = \frac{1}{c_w}\sum_{\mathbf{x}}\sum_{y} p_\theta(y|\mathbf{x})\phi_w(\mathbf{x},y)$$

  - vector with expected distribution over labels for documents that contain *w* (c_w is the count of *w*)

- **Example #2: Per-example expectation:**

$$\mathbf{E}_{p_\theta}[\phi(\mathbf{x},\mathbf{y})] = \sum_{\mathbf{y}} p_\theta(\mathbf{y}|\mathbf{x})\phi(\mathbf{x},\mathbf{y})$$

  - vector with *mth* value = expected number of target words that align with source word *m*

Expressing Preferences

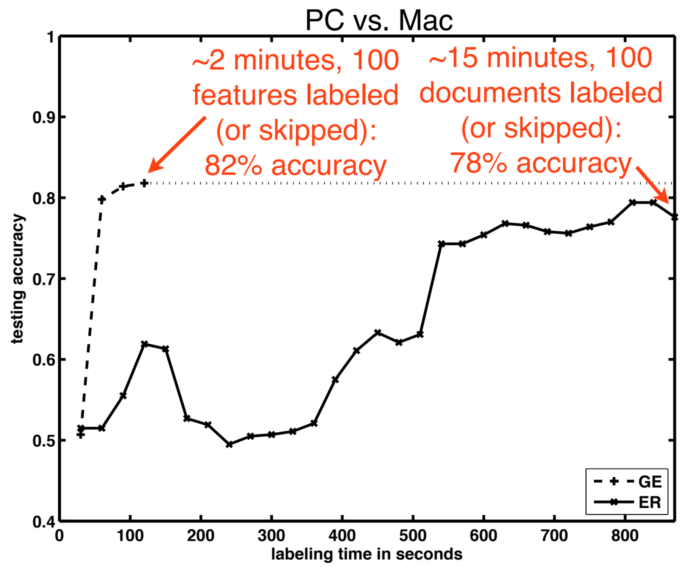
- express preferences using **target values**: $\tilde{\phi}$

- **Example #1:** $\mathbf{E}_{p_\theta}[\phi_w(\mathbf{X},\mathbf{Y})] \approx \tilde{\phi}$
  - *label distribution* for *game* is close to [40% 40% 20%]

- **Example #2:** $\mathbf{E}_{p_\theta}[\phi(\mathbf{x},\mathbf{y})] \leq \tilde{\phi}$
  - expected number of target words that align with each source word is at most *one*

Preview: Labeled Features

User Experiments [Druck et al. 08]



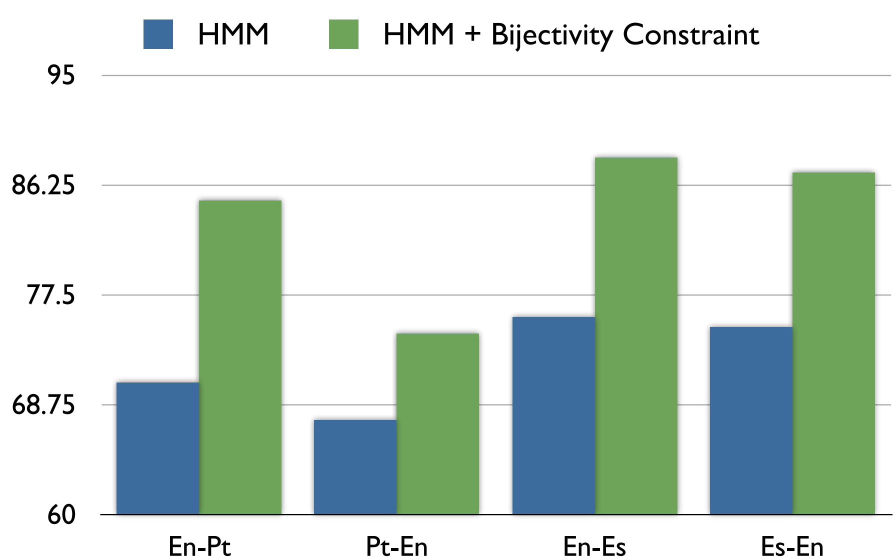
targets set with simple heuristic: majority label gets 90% of mass

complete set of labeled features

PC	Mac
dos	mac
ibm	apple
hp	quadra
dx	

Preview: Word Alignment

[Graça et al. 10]



Overview of the Frameworks

Running Example

Model Family: conditional exponential models

$$p_\theta(\mathbf{Y}|\mathbf{X}) = \frac{\exp(\theta \cdot \mathbf{f}(\mathbf{X}, \mathbf{Y}))}{\mathbf{Z}(\mathbf{X})}$$

$$\mathbf{Z}(\mathbf{X}) = \sum_{\mathbf{Y}} \exp(\theta \cdot \mathbf{f}(\mathbf{X}, \mathbf{Y}))$$

$\mathbf{f}(\mathbf{X}, \mathbf{Y})$ are *model* features

Choosing parameters θ

Model Family: conditional exponential models

$$p_\theta(\mathbf{Y}|\mathbf{X}) = \frac{\exp(\theta \cdot \mathbf{f}(\mathbf{X},\mathbf{Y}))}{\mathbf{Z}(\mathbf{X})}$$

Objective: maximize observed data likelihood

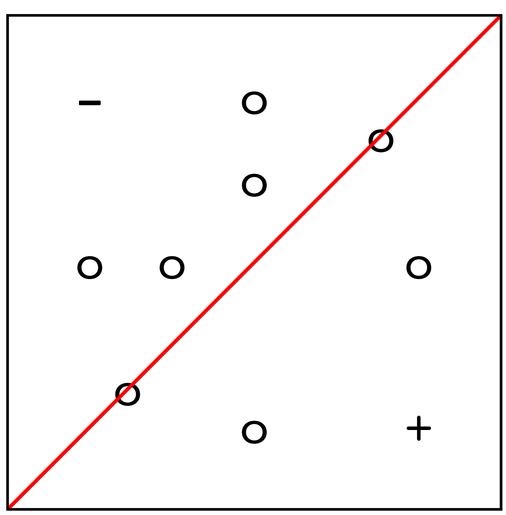
$$\max_\theta \quad \log p_\theta(\mathbf{Y}_L|\mathbf{X}_L) \ + \ \log p(\theta) \ \stackrel{\text{def}}{=} \ \mathcal{L}(\theta; D_L)$$

Note: Frameworks also suitable for generative models (no labeled data necessary)

Visual Example: Maximum Likelihood

Model: $p(\mathbf{Y}|\mathbf{X}) = \prod_i \frac{\exp(\mathbf{y}_i \mathbf{x}_i \cdot \theta)}{Z(\mathbf{x}_i)}$

Objective: $\max_\theta \ \log p_\theta(\mathbf{Y}_L|\mathbf{X}_L) - 0.1\|\theta\|_2^2$



A language for prior information

The expectations of user-defined *constraint* features $\phi(\mathbf{X}, \mathbf{Y})$ are close to some value $\tilde{\phi}$

$$\mathbf{E}[\phi(\mathbf{X}, \mathbf{Y})] \approx \tilde{\phi}$$

Running Example:

Want to ensure that 25% of unlabeled documents are about politics

- *constraint* features

$$\phi(\mathbf{x}, \mathbf{y}) = \begin{cases} 1 & \text{if } \mathbf{y} \text{ is ``politics''} \\ 0 & \text{otherwise} \end{cases}$$

- preferred expected value

$$\tilde{\phi} = 0.25$$

- Expectation w.r.t. unlabeled data

Constraint-Driven Learning

M. Chang, L. Ratinov, D. Roth (2007).

Motivation: Hard EM algorithm with preferences

Hard EM:

$$\text{E-Step: set } \hat{\mathbf{Y}} = \arg\max_{\mathbf{Y}} \ \log p_\theta(\mathbf{Y}|\mathbf{X})$$

$$\text{M-Step: set } \theta = \arg\max_{\theta} \ \log p_\theta(\hat{\mathbf{Y}}|\mathbf{X})$$

Constraint Driven Learning:

$$\text{E-Step: set } \hat{\mathbf{Y}} = \arg\max_{\mathbf{Y}} \ \log p_\theta(\mathbf{Y}|\mathbf{X}) - \text{penalty}(\mathbf{Y})$$

$$\text{M-Step: set } \theta = \arg\max_{\theta} \ \log p_\theta(\hat{\mathbf{Y}}|\mathbf{X})$$

Constraint-Driven Learning

Motivation: Hard EM algorithm with preferences

Constraint Driven Learning:

$$\text{E-Step: set } \hat{\mathbf{Y}} = \arg\max_{\mathbf{Y}} \ \log p_\theta(\mathbf{Y}|\mathbf{X}) - \text{penalty}(\mathbf{Y})$$

$$\text{M-Step: set } \theta = \arg\max_{\theta} \ \log p_\theta(\hat{\mathbf{Y}}|\mathbf{X})$$

- penalties encode similar information as $\mathbf{E}[\phi] \approx \tilde{\phi}$

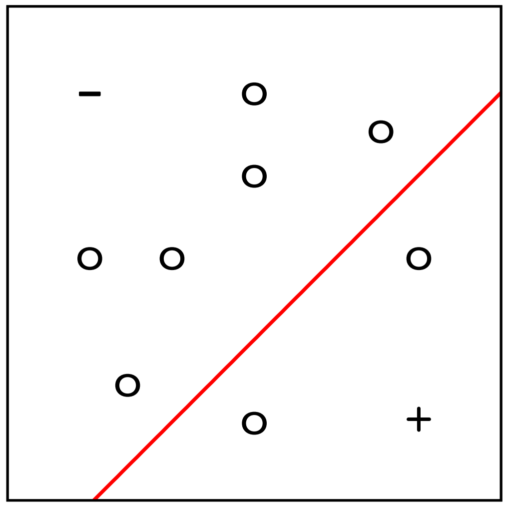
* more on this later *

- E-Step can be hard; use beam search

Visual Example: Constraint Driven Learning

$$\max_{\theta,\hat{\mathbf{Y}}} \quad \log p_\theta(\mathbf{Y}_L|\mathbf{X}_L) - 0.1\|\theta\|_2^2 \quad \text{s.t.} \quad \phi(\hat{\mathbf{Y}}) = 2$$

where $\hat{\mathbf{Y}}$ are "imagined" labels and $\phi[\hat{\mathbf{Y}}] = \text{count}(+, \hat{\mathbf{Y}})$



Posterior Regularization

J. Graça, K. Ganchev, B. Taskar (2007).

Motivation: EM algorithm with *sane* posteriors

EM:

$$\text{E-Step: set } q(\mathbf{Y}) = \arg\min_q \mathcal{D}_{\text{KL}}(q(\mathbf{Y})||p_\theta(\mathbf{Y}|\mathbf{X}))$$

$$\text{M-Step: set } \theta = \arg\max_\theta \mathbf{E}_{q(\mathbf{Y})}[p_\theta(\mathbf{Y}|\mathbf{X})]$$

Constrained EM:

$$\text{E-Step: set } q(\mathbf{Y}) = \arg\min_{q \in \mathcal{Q}} \mathcal{D}_{\text{KL}}(q(\mathbf{Y})||p_\theta(\mathbf{y}|\mathbf{x}))$$

$$\text{M-Step: set } \theta = \arg\max_\theta \mathbf{E}_{q(\mathbf{Y})}[p_\theta(\mathbf{Y}|\mathbf{X})]$$

Posterior Regularization

Motivation: EM algorithm with *sane* posteriors

Idea: $\mathbf{E}[\phi] \approx \tilde{\phi}$ provide constraints

define $\mathcal{Q}$: set of q such that $\mathbf{E}_q[\phi] \approx \tilde{\phi}$

run EM-like procedure but use proposal $q \in \mathcal{Q}$

Objective:

$$\max_{\theta} \quad \mathcal{L}(\theta; D_L) - \mathcal{D}_{\mathrm{KL}}(\mathcal{Q} \mid\mid p_\theta(\mathbf{Y}|\mathbf{X}))$$

where
$\mathcal{D}_{\mathrm{KL}}$ is Kullback-Leibler divergence
$\mathbf{X} = D_U$ are the input variables for unlabeled corpus
$\mathbf{Y}$ is label for *entire* unlabeled corpus

Posterior Regularization

Hard constraints:

$$\max_{\theta} \quad \mathcal{L}(\theta; D_L) \; - \; \min_{q \in \mathcal{Q}} \; \mathcal{D}_{\mathrm{KL}}(q(\mathbf{Y}) \mid\mid p_\theta(\mathbf{Y}|\mathbf{X}))$$

$$\mathcal{Q} = \left\{ q(\mathbf{Y}) : \left\| \mathbf{E}_q[\phi(\mathbf{Y})] = \tilde{\phi} \right\|_2^2 \leq \epsilon \right\}$$

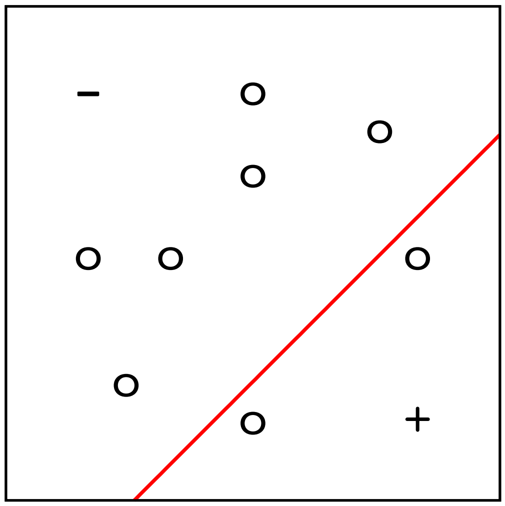
Soft constraints:

$$\max_{\theta} \quad \mathcal{L}(\theta; D_L) \; - \; \min_{q} \begin{pmatrix} \mathcal{D}_{\mathrm{KL}}(q(\mathbf{Y}) \mid\mid p_\theta(\mathbf{Y}|\mathbf{X})) + \\ \alpha \left\| \mathbf{E}_q[\phi(\mathbf{Y})] = \tilde{\phi} \right\|_2^2 \end{pmatrix}$$

Visual Example: Posterior Regularization

$$\max_{\theta} \quad \log p_\theta(\mathbf{Y}_L|\mathbf{X}_L) - 0.1\|\theta\|_2^2 \;-\; \mathcal{D}_{\mathrm{KL}}(\mathcal{Q}||p_\theta)$$

where: $\mathcal{D}_{\mathrm{KL}}(\mathcal{Q}||p_\theta) \;=\; \min_q \; \mathcal{D}_{\mathrm{KL}}(q||p_\theta) \;\text{ s.t. } \mathbf{E}_q[\phi] = 2$



Generalized Expectation Constraints

G. Mann, A. McCallum (2007).

Motivation: augment log-likelihood with cost for "bad" posteriors.

Objective:

$$\max_{\theta} \mathcal{L}(\theta; D_L) - \left\| \mathbf{E}_{p_\theta(\mathbf{Y}|\mathbf{X})}[\phi] \;-\; \tilde{\phi} \right\|_\beta$$

where $\mathbf{E}_{p_\theta(\mathbf{Y}|\mathbf{X})}[\phi] = \mathbf{E}_{p_\theta(\mathbf{Y}|\mathbf{X})}[\phi(\mathbf{X},\mathbf{Y})]$

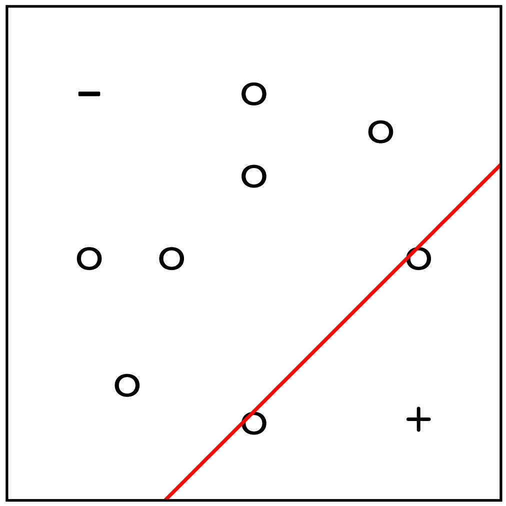
$= \sum_{\mathbf{Y}} p_\theta(\mathbf{Y}|\mathbf{X})\phi(\mathbf{X},\mathbf{Y})$ is short-hand

Optimization: gradient descent on θ

A visual comparison of the frameworks

Objective: Generalized Expectation Constraints

$$\max_{\theta} \quad \log p_\theta(\mathbf{Y}_L|\mathbf{X}_L) - 0.1\|\theta\|_2^2 - 500\|\mathbf{E}_{p_\theta}[\phi] - 2\|_2^2$$



Types of constraints

Constraint Driven Learning: Penalized Viterbi

$$\arg\max_{\mathbf{Y}} \quad \log p_\theta(\mathbf{Y}|\mathbf{X}) - \|\phi(\mathbf{X},\mathbf{Y}) - \tilde{\phi}\|_\beta$$

- Easy if $\|\phi(\mathbf{X},\mathbf{Y}) - \tilde{\phi}\|_\beta$ decompose as the model.

$$p(\mathbf{Y}|\mathbf{X}) = \prod_c p_c(\mathbf{y}_c|\mathbf{X}) \quad \text{and}$$

$$\|\phi(\mathbf{X},\mathbf{Y}) - \tilde{\phi}\|_\beta = \sum_c \delta_c(\mathbf{X},\mathbf{y}_c)$$

- Otherwise:
  - Beam search
  - Integer linear program

Types of constraints

Posterior Regularization: KL projection

$$\min_q \; \mathcal{D}_{\text{KL}}(q||p_\theta) \;\; \text{s.t.} \;\; \|\mathbf{E}_q[\phi] - \tilde{\phi}\|_\beta \leq \epsilon$$

- Usually easy if $\phi(\mathbf{Y}, \mathbf{X})$ decompose as the model:

$$p(\mathbf{Y}|\mathbf{X}) = \prod_c p_c(\mathbf{y}_c|\mathbf{X})$$

and $\Rightarrow$ $q(\mathbf{Y}|\mathbf{X}) = \prod_c q_c(\mathbf{y}_c|\mathbf{X})$

$$\phi(\mathbf{X}, \mathbf{Y}) = \sum_c \phi_c(\mathbf{X}, \mathbf{y}_c)$$

- Otherwise: Sample (e.g. K. Bellare, G. Druck, and A. McCallum, 2009)

Types of constraints

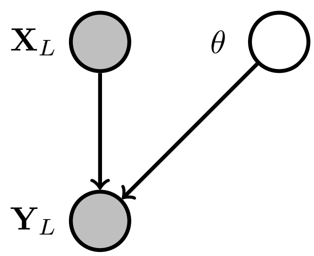
Generalized Expectation Constraints: Direct gradient

$$\max_\theta \mathcal{L}(\theta; D_L) - \left\| \mathbf{E}_{p_\theta(\mathbf{Y}|\mathbf{X})}[\phi] \; - \tilde{\phi} \right\|_\beta$$

- Usually easy if: $\phi(\mathbf{Y}, \mathbf{X})$
  - decomposes as the model $\phi(\mathbf{X}, \mathbf{Y}) = \sum_c \phi_c(\mathbf{X}, \mathbf{y}_c)$
  - Can compute $\mathbf{E}[\phi \times \mathbf{f}]$ * more on this later *
    - Unstructured
    - Sequence, Grammar (semiring trick)
- Otherwise: sample or approximate the gradient.

A Bayesian View: Measurements

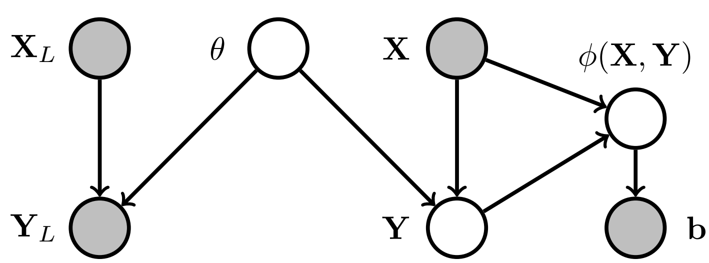
P. Liang, M. Jordan, D. Klein (2009)



Objective: mode of θ given observations

$$\max_{\theta} \quad \log p(\theta) + \sum_{(\mathbf{x},\mathbf{y})\in D_L} \log p_\theta(\mathbf{y}|\mathbf{x}) = \mathcal{L}(\theta; D_L)$$

A Bayesian View: Measurements



Objective: mode of θ given observations

$$\max_{\theta} \quad \mathcal{L}(\theta; D_L) + \log \mathbf{E}_{p_\theta(\mathbf{Y}|\mathbf{X})} \left[p(\tilde{\phi}|\phi(\mathbf{X}, \mathbf{Y})) \right]$$

What's wrong with this picture?

Objective: mode of θ given observations

$$\max_{\theta} \quad \mathcal{L}(\theta; D_L) + \log \mathbf{E}_{p_\theta(\mathbf{Y}|\mathbf{X})}\left[p(\tilde{\phi}|\phi(\mathbf{X},\mathbf{Y}))\right]$$

Example: Exactly 25% of articles are "politics"

$$p(\tilde{\phi}|\phi(\mathbf{X},\mathbf{Y})) = \mathbf{1}\left(\tilde{\phi} = \phi(\mathbf{X},\mathbf{Y})\right)$$

What is the probability exactly 25% of the articles are labeled ``politics"?

$$\mathbf{E}_{p_\theta(\mathbf{Y}|\mathbf{X})}\left[\mathbf{1}(\tilde{\phi} = \phi(\mathbf{X},\mathbf{Y}))\right]$$

How do we optimize this with respect to θ?

What's wrong with this picture?

Example: Compute prob: 25% of docs are "politics".

Article	p("politics")
1	0.2
2	0.4
3	0.1
4	0.6

Naively:

$$0.2 \times (1-0.4) \times (1-0.1) \times (1-0.6)$$
$$+ \ldots +$$
$$+(1-0.2) \times (1-0.4) \times (1-0.1) \times 0.6$$

in this case we can use a DP, but if there are many constraints, that doesn't work.

Easier: What is the expected number of "politics" articles?

$$0.2 + 0.4 + 0.1 + 0.6$$

Probabilities and Expectations

difficult to compute expectations of arbitrary functions *but...*

Usually: $\phi(\mathbf{X}, \mathbf{Y})$ decomposes as a sum

e.g. 25% of articles are "politics"

$$\phi(\mathbf{X}, \mathbf{Y}) = \sum_{\text{instances}} \phi(\mathbf{x}, \mathbf{y})$$

Idea: approximate

$$\mathbf{E}_{p_\theta(\mathbf{Y}|\mathbf{X})}\left[p\left(\tilde{\phi} \mid \phi(\mathbf{X}, \mathbf{Y})\right)\right] \approx p\left(\tilde{\phi} \mid \mathbf{E}_{p_\theta(\mathbf{Y}|\mathbf{X})}\left[\phi(\mathbf{X}, \mathbf{Y})\right]\right)$$

Probabilities and Expectations

Approximation: $\mathbf{E}_{p_\theta(\mathbf{Y}|\mathbf{X})}\left[p\left(\tilde{\phi} \mid \phi\right)\right] \approx p\left(\tilde{\phi} \mid \mathbf{E}_{p_\theta(\mathbf{Y}|\mathbf{X})}\left[\phi\right]\right)$

$$\Downarrow$$

Objective: $\max_\theta \quad \mathcal{L}(\theta; D_L) + \log p\left(\tilde{\phi} \mid \mathbf{E}_{p_\theta(\mathbf{Y}|\mathbf{X})}\left[\phi\right]\right)$

Example: $p\left(\tilde{\phi} \mid \mathbf{E}[\phi]\right)$ is Gaussian

$\Rightarrow \quad \log p\left(\tilde{\phi} \mid \mathbf{E}[\phi]\right)$ is $\left\|\mathbf{E}[\phi] - \tilde{\phi}\right\|_2^2$

so for appropriate $\log p\left(\tilde{\phi} \mid \mathbf{E}[\phi]\right)$ this is identical to GE!

Optimizing GE objective

GE Objective:

$$\mathcal{O}_{\text{GE}} = \max_{\theta} \quad \mathcal{L}(\theta; D_L) - \left\| \mathbf{E}_{p_\theta(\mathbf{Y}|\mathbf{X})}[\phi(\mathbf{X}, \mathbf{Y})] - \tilde{\phi} \right\|_{\beta}$$

- Gradient involves covariance

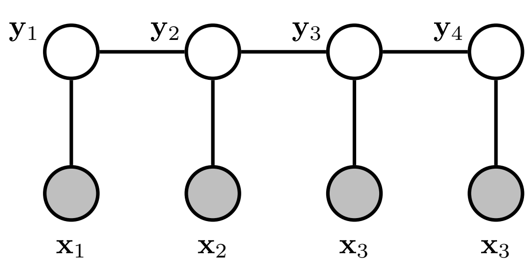
$$\text{Cov}(\phi, \mathbf{f}) = \mathbf{E}[\phi \times \mathbf{f}] - \mathbf{E}[\phi] \times \mathbf{E}[\mathbf{f}]$$

this can be hard because

$$\mathbf{E}[\phi \times \mathbf{f}] = \sum_{\mathbf{Y}} p(\mathbf{Y})\phi(\mathbf{Y}) \times \mathbf{f}(\mathbf{Y})$$

and the usual dynamic programs (inside outside, forward backward) can't compute this.

Optimizing GE Objective



$$\mathbf{E}[\phi \times \mathbf{f}] = \sum_{\mathbf{Y}} p(\mathbf{Y})\phi(\mathbf{Y}) \times \mathbf{f}(\mathbf{Y})$$

$$\phi(\mathbf{Y}) \times \mathbf{f}(\mathbf{Y}) = \left[\sum_i \phi(\mathbf{y}_i) \right] \times \left[\sum_j \mathbf{f}(\mathbf{y}_j) \right]$$

Maintaining both $\mathbf{y}_i$ and $\mathbf{y}_j$ in the DP is expensive

* Semiring trick can help for some problems *
E.g. if inference is a hypergraph problem.

A Variational Approximation

GE Objective:

$$\mathcal{O}_{\text{GE}} = \max_{\theta} \quad \mathcal{L}(\theta; D_L) - \left\| \tilde{\phi} - \mathbf{E}_{p_\theta(\mathbf{Y}|\mathbf{X})}[\phi(\mathbf{X}, \mathbf{Y})] \right\|_\beta$$

- Can be hard to compute $\text{Cov}(\phi, \mathbf{f})$ in gradient.

Idea: use variational approximation $q(\mathbf{Y}) \approx p_\theta(\mathbf{Y}|\mathbf{X})$

$$\max_{\theta, q(\mathbf{Y})} \quad \mathcal{L}(\theta; D_L) - \mathcal{D}_{\text{KL}}\left(q(\mathbf{Y}) \;||\; p_\theta(\mathbf{Y}|\mathbf{X})\right) - \left\| \mathbf{E}_q[\phi(\mathbf{X}, \mathbf{Y})] \; - \; \tilde{\phi} \right\|_\beta$$

* Note: this is the PR objective *

Approximating with the mode

PR Objective:

$$\max_{\theta, q(\mathbf{Y})} \quad \mathcal{L}(\theta; D_L) - \mathcal{D}_{\text{KL}}\left(q(\mathbf{Y}) \;||\; p_\theta(\mathbf{Y}|\mathbf{X})\right) - \left\| \mathbf{E}_q[\phi(\mathbf{X}, \mathbf{Y})] \; - \; \tilde{\phi} \right\|_\beta$$

sometimes minimizing the KL is hard.

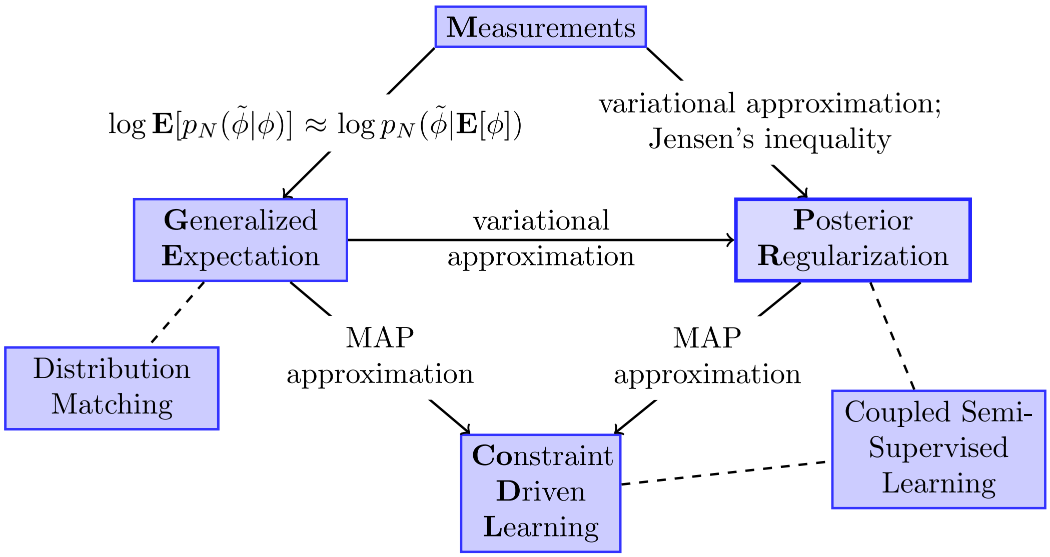
Idea: use hard assignment $q(\mathbf{Y}) \approx \mathbf{1}(\mathbf{Y} = \hat{\mathbf{Y}})$:

- $\mathcal{D}_{\text{KL}}\left(q(\mathbf{Y}) \;||\; p_\theta(\mathbf{Y}|\mathbf{X})\right)$ becomes $\log p(\hat{\mathbf{Y}})$
- $\left\| \mathbf{E}_q[\phi(\mathbf{X}, \mathbf{Y})] \; - \; \tilde{\phi} \right\|_\beta$ becomes $\log p(\tilde{\phi} \mid \phi(\mathbf{X}, \hat{\mathbf{Y}}))$
- use EM-like procedure to optimize

Constraint Driven Learning Objective:

$$\max_{\theta, \hat{\mathbf{Y}}} \quad \mathcal{L}(\theta; D_L) + \log p_\theta(\hat{\mathbf{Y}}) + \log p(\tilde{\phi} | \phi(\mathbf{X}, \hat{\mathbf{Y}}))$$

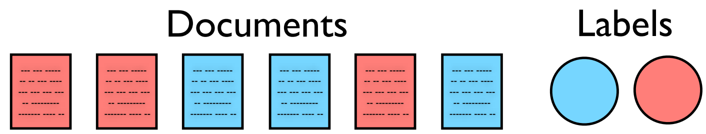
Visual Summary



Applications

- **Unstructured problems:**
  - Document Classification
- **Sequence problems:**
  - Information Extraction
  - Pos-Induction
  - Word Alignment
- **Tree problems:**
  - Grammar Induction

Document Classification



- **Model:** Max. Entropy Classifier (Logistic Regression)

$$p_\theta(y|\mathbf{x}) = \frac{\exp(\theta \cdot \mathbf{f}(\mathbf{x}, y))}{\sum_y \exp(\theta \cdot \mathbf{f}(\mathbf{x}, y))}$$

- **Challenge:** What if we have no labeled data?
- cannot use standard unsupervised learning: $\sum_y p_\theta(y|\mathbf{x}) = 1$

Labeled Features

- often we can still provide some light supervision
- **prior knowledge:** labeled features

sentiment polarity

positive	**negative**
memorable	terrible
perfect	boring
exciting	mess

newsgroups classification

baseball	**Mac**	**politics**	**...**
hit	Apple	senate	...
Braves	Macintosh	taxes	...
runs	Powerbook	liberal	...

- **formally:** have an estimate of the distribution over labels for documents that contain word *w*: $\tilde{\phi}_w$

Leveraging Labeled Features with GE

[Mann & McCallum 07], [Druck et al. 08]

- **constraint feature:** $\phi_w(\mathbf{x}, y) = \mathbf{1}(y = l)1(w \in \mathbf{x})$
  - for a document $\mathbf{x}$, returns a vector with a 1 in the *l*th position if y is the *l*th label and the word *w* is in $\mathbf{x}$
- **expectation:** label distribution for docs that contain *w*

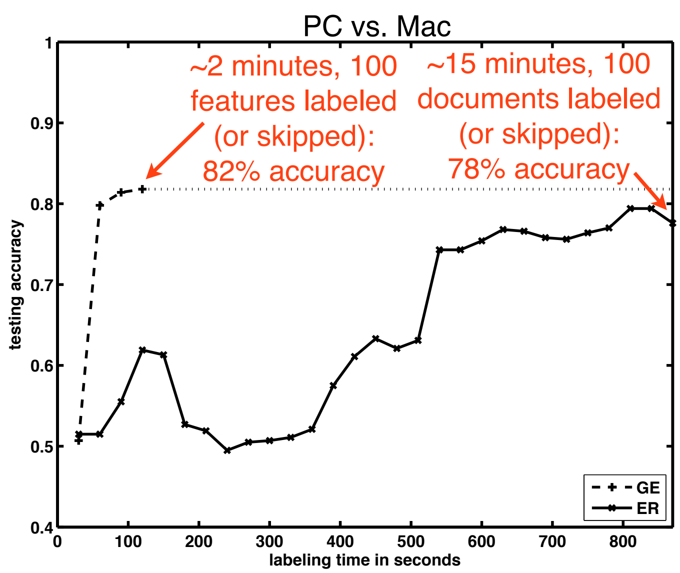
$$\frac{1}{c_w} \sum_{\mathbf{x}} \mathrm{E}_{p_\theta(y|\mathbf{x})}[\phi_w(\mathbf{x}, y)]$$

- **GE penalty:** KL divergence from target distribution

$$\mathcal{D}_{KL}\left(\tilde{\phi}_w || \frac{1}{c_w} \sum_{\mathbf{x}} \mathrm{E}_{p_\theta(y|\mathbf{x})}[\phi_w(\mathbf{x}, y)]\right)$$

User Experiments with Labeled Features

[Druck et al. 08]



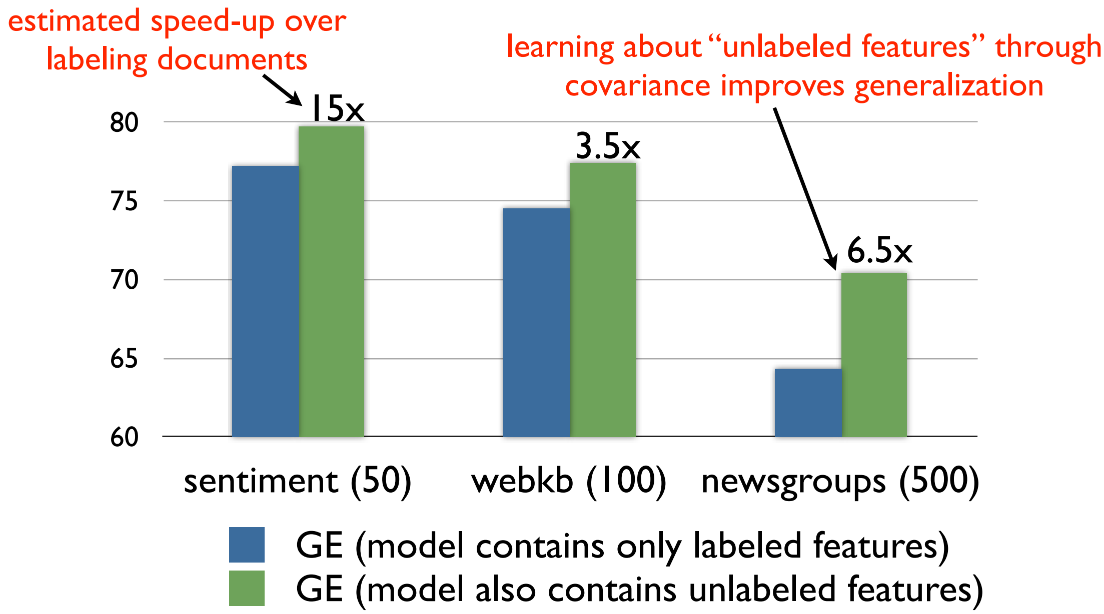
targets set with simple heuristic: majority label gets 90% of mass

complete set of labeled features

PC	Mac
dos	mac
ibm	apple
hp	quadra
dx	

Experiments with Labeled Features

[Druck et al. 08]



Information Extraction: Example Tasks

- **citation extraction:**

Cousot, P. and Cousot, R. 1978. Static determination of dynamic properties of recursive procedures. In Proceedings of the IFIP Conference on Programming Concepts, E. Neuhold, Ed. North-Holland Pub. Co., 237-277.

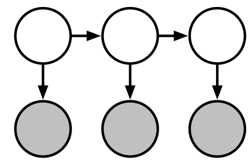
- **apartment listing extraction:**

Detached single family house. 3 bedrooms 1 1/2 baths. Almost 1000 square feet in living area. 1 car garage. New pergo floor and tile kitchen floor. New interior/exterior paint. Close to shopping mall and bus stop. Near 101/280. Available July 1, 2004. If you are interested, email for more details.

Information Extraction: Markov Models

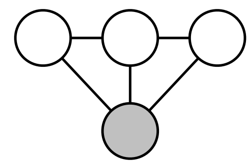
- models for **sequence labeling** based IE
- **Hidden Markov Model (HMM):**

$$p_\theta(\mathbf{y}, \mathbf{x}) = p_\theta(y_0) \prod_{i=1}^{N} p_\theta(y_i | y_{i-1}) p_\theta(x_i | y_i)$$



- **Conditional Random Field (CRF):**

$$p_\theta(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \exp(\sum_{i=1}^{N} \theta \cdot \mathbf{f}(\mathbf{x}, y_{i-1}, y_i))$$



Information Extraction: Labeled Features

[Mann & McCallum 08], [Liang et al. 09]

apartments example labeled features:

ROOMMATES	respectful
CONTACT	*phone*
FEATURES	laundry

constraint features:

$$\phi_q(\mathbf{x}, y_i, i) = \mathbf{1}(y_i = l) q(\mathbf{x}, i)$$

vector with a 1 in the *l*th position if *y* is the *l*th label and predicate *q* is true (i.e. *w* is present at *i*)

expectation:

$$\frac{1}{c_q} \sum_{\mathbf{x}} \sum_{i} \mathrm{E}_{p_\theta(y_i|\mathbf{x})}[\phi_q(\mathbf{x}, y_i, i)]$$

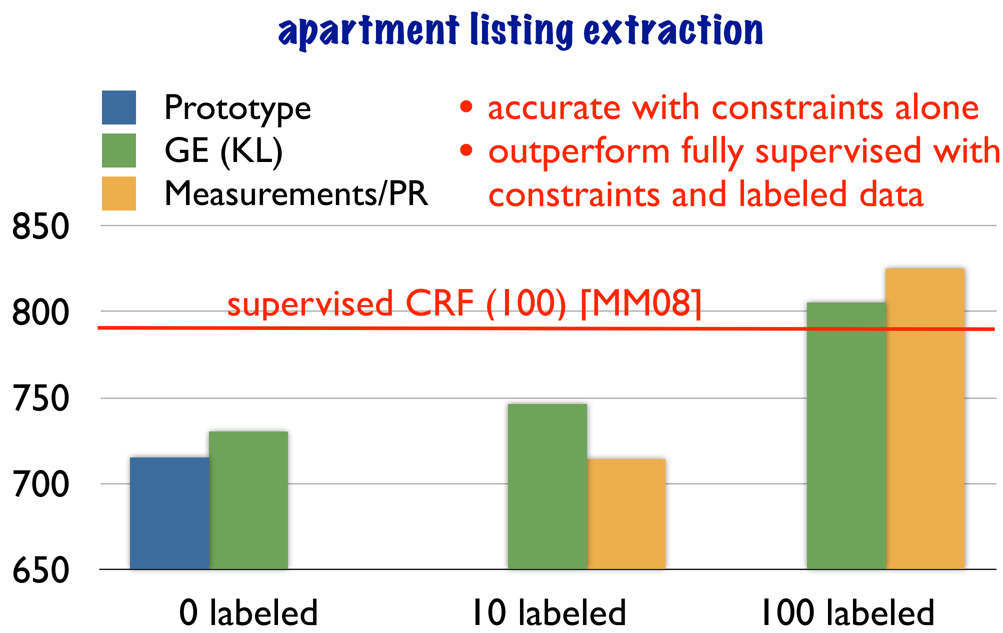
label distribution when *q* is true

model: Linear Chain CRF

note: Semiring trick makes GE O(L^2) instead of O(L^3) as in [Mann & McCallum 08]

Information Extraction: Labeled Features

[Haghighi & Klein 06], [Mann & McCallum 08], [Liang et al. 09]



Limitations of Markov Models

- ***predicted:*** Cousot, P. and Cousot, R. 1978. Static determination of dynamic properties of recursive procedures. In Proceedings of the IFIP Conference on Programming Concepts, E. Neuhold, Ed. North-Holland Pub. Co., 237-277.
- prediction has two **author** and two **title** segments:
  - error #1: Neuhold, Ed. should be **editor**
  - error #2: North-Holland Pub. Co., should be **publisher**
- *A Markov model cannot represent that at most one segment of each type appears in each reference.*

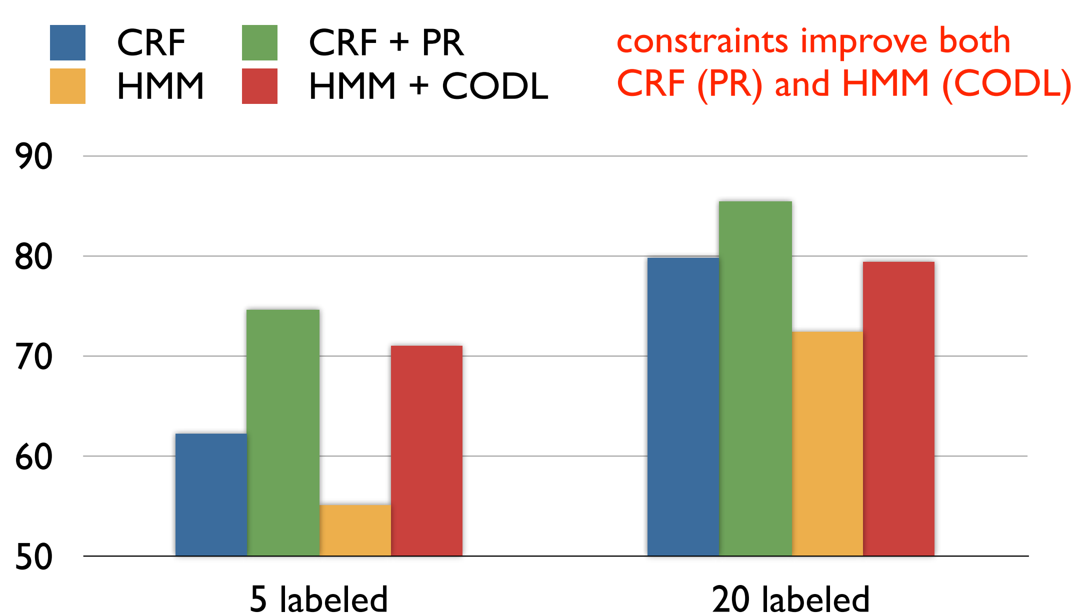
Long-Range Constraints

[Chang et al. 07] [Bellare et al. 09]

- *"Each field is a contiguous sequence of tokens and appears at most once in a citation."*
- **constraint feature:** counts the number of segments of each type
- constrained to be ≤ 1 using **PR** or **CODL**
- **additional constraints:** 10 labeled features such as:
    - *pages*→**pages**
    - *proc.*→**booktitle**

Long-Range Constraints

[Chang et al. 07] [Bellare et al. 09]

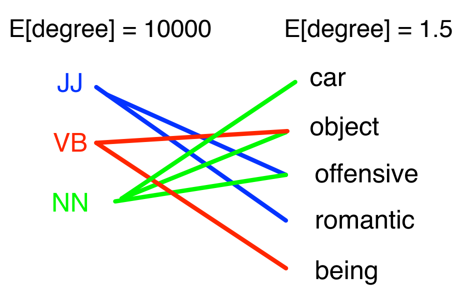


Other Applications in Information Extraction

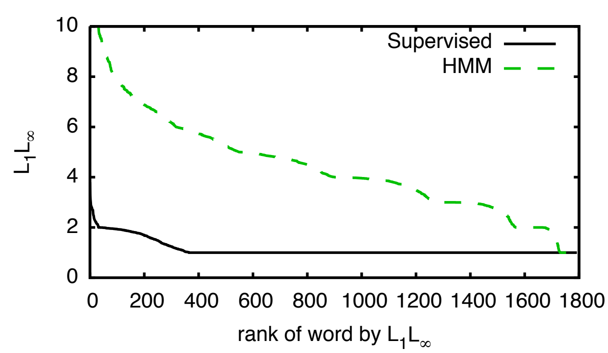
citation	model	method	description
[Mann et al. 07]	MaxEnt	GE	constraints on label marginals
[Druck et al. 09]	CRF	GE	actively labeled features
[Bellare & McCallum 09]	alignment CRF	GE	labeled features
[Singh et al. 10]	semi-Markov CRF	PR	labeled gazetteers
[Druck et al. 10]	HMM	PR	constraints derived from labeled data

Pos Induction Low Tag Ambiguity

[Graça et al. 09]



Distribution of word ambiguity



Measuring Tag Ambiguity

[Graça et al. 09]

	N	V	ADJ	Prep	ADV	Sum
a run into town.	0.9	0.1	0	0	0	1
of the mile run.	0.7	0.1	0.1	0	0.1	1
run gold.	0.1	0.3	0	0.6	0	1
run errands.	0.3	0.6	0	0	0.1	1
run for mayor.	0.3	0.7	0	0	0	1
Max	0.9	0.7	0.1	0.6	0.2	

Sum

2.5

- Pick a particular word type: run
  - Stack all occurrences
- Calculate posterior probability
- Take the maximum for each tag
- Sum the maxes

ϕ_{wti} :Word type ***w*** has hidden state ***t*** at occurrence ***i***

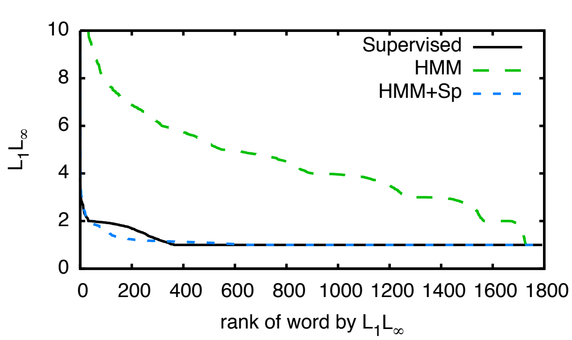
$$\min_{c_{wt}} E_{q(y)}[\phi_{wti}] \leq c_{wt}$$

$$\ell_1/\ell_\infty = \sum_{wt} c_{wt}$$

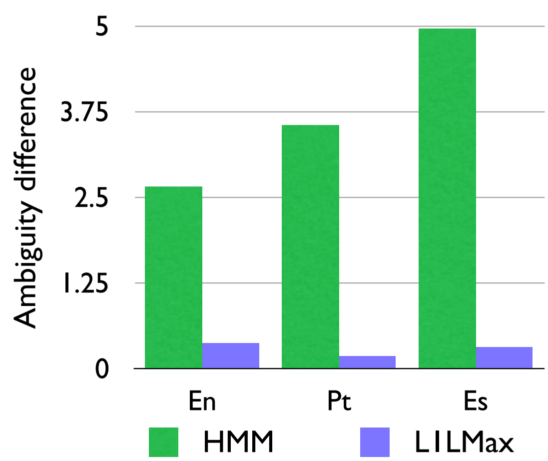
Tag Sparsity

[Graça et al. 09]

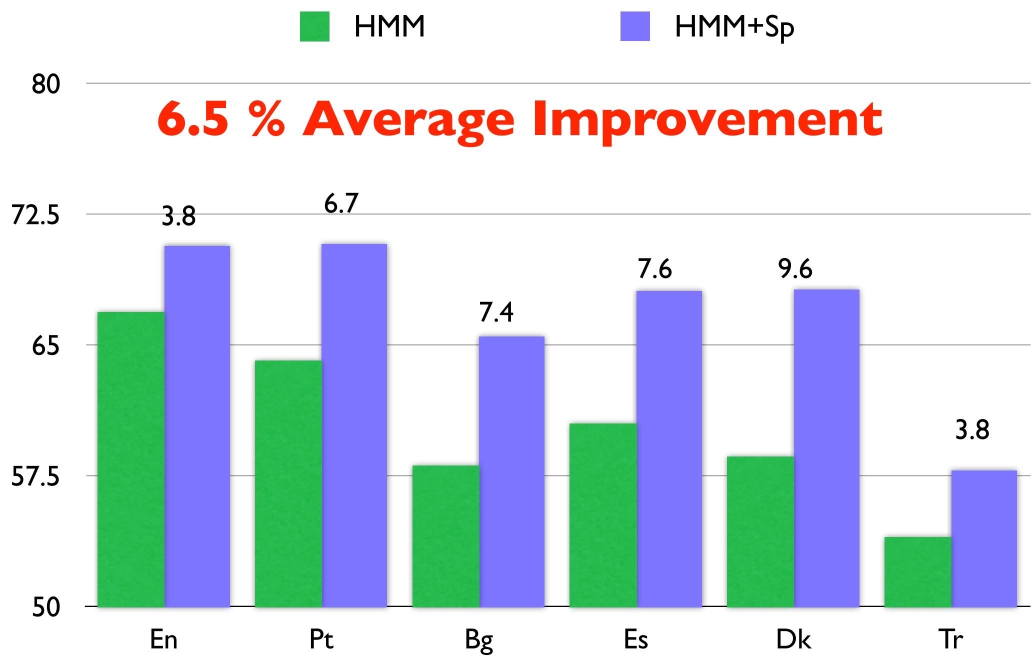
Distribution of word ambiguity



Average ambiguity difference



Results
[Graça et al. 09]

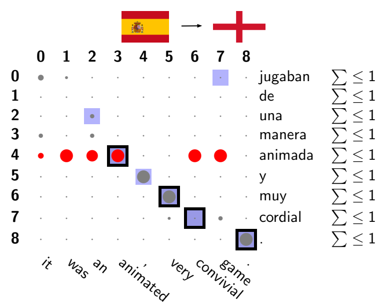


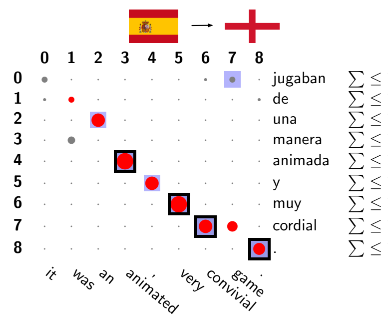
Word Alignments
[Graça et al. 10]

- **Bijectivity constraints**:
  - Each word should align to at most one other word
- **Symmetry constraints:**
  - Directional models should agree

Bijectivity Constraints

[Graça et al. 10]



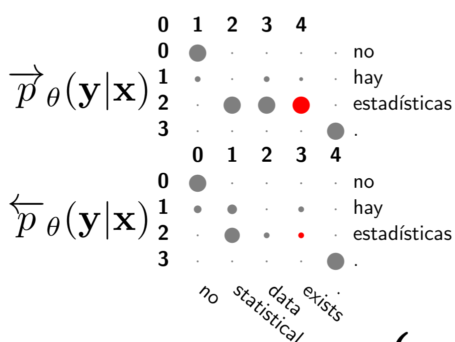


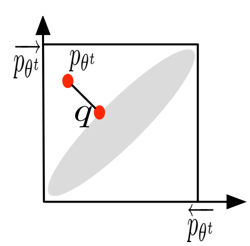
Feature: $\phi(\mathbf{x},\mathbf{y}) = \sum_{i=1}^{N} \mathbf{1}(y_i = m)$

Constraint: $\mathbf{E}_q[\phi(\mathbf{x},\mathbf{y})] \leq 1$

Symmetry Constraints

[Graça et al. 10]





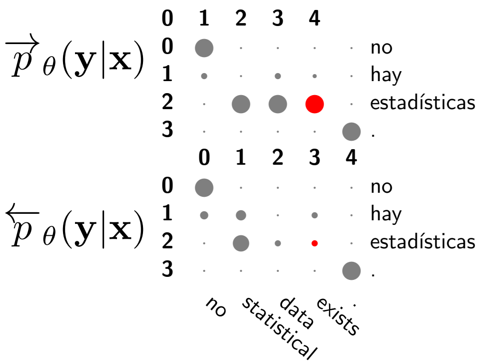
Feature: $\phi(\mathbf{x},\mathbf{y}) = \begin{cases} +1 & \mathbf{y} \in \overrightarrow{y} \text{ and } \overrightarrow{y}_i = j \\ -1 & \mathbf{y} \in \overleftarrow{y} \text{ and } \overleftarrow{y}_j = i \\ 0 & \text{otherwise} \end{cases}$

Constraint: $\mathbf{E}_q[\phi(\mathbf{x},\mathbf{y})] = 0$

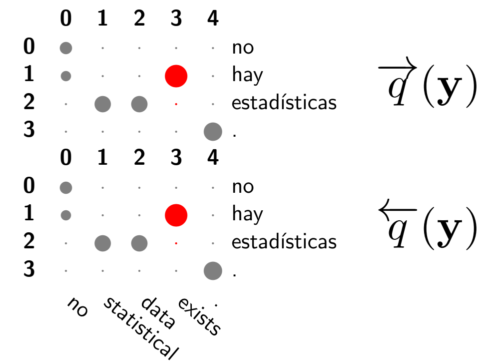
Symmetry Constraints

[Graça et al. 10]

Before projection:

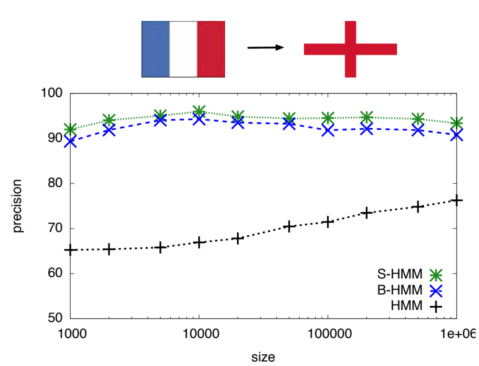


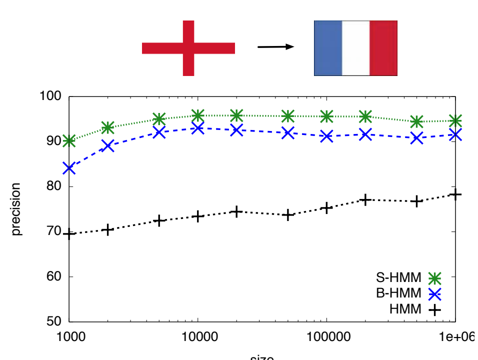
After projection:



Results

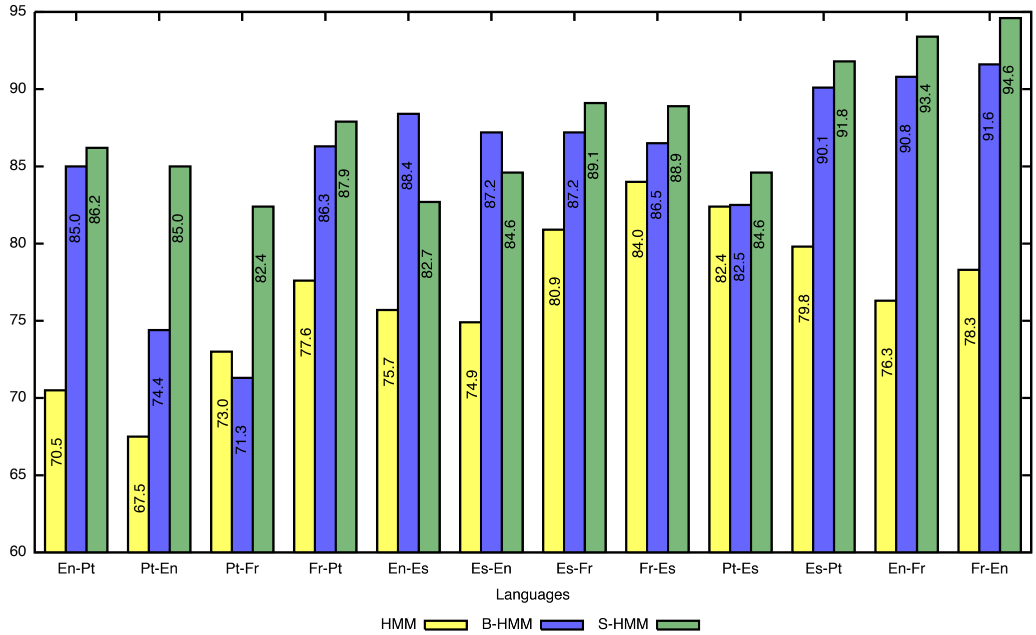
[Graça et al. 10]





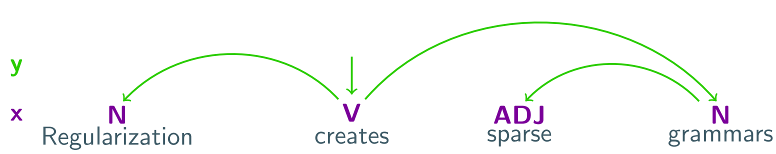
Results

[Graça et al. 10]



Dependency Parsing
DMV Model

[Graça et al. 04]

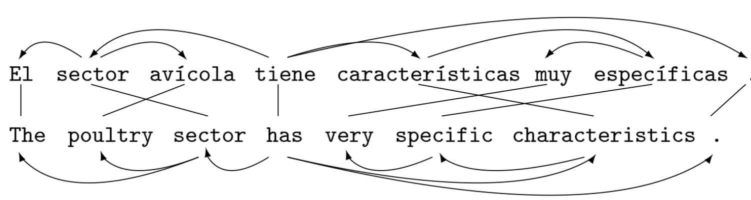


$$p_\theta(\mathbf{x},\mathbf{y}) = \theta_{root(V)}$$
$$\cdot\theta_{stop(nostop|V,right,false)} \cdot \theta_{child(N|V,right)}$$
$$\cdot\theta_{stop(stop|V,right,true)} \cdot \theta_{stop(nostop|V,left,false)} \cdot \theta_{child(N|V,left)}$$

Dependency Parsing

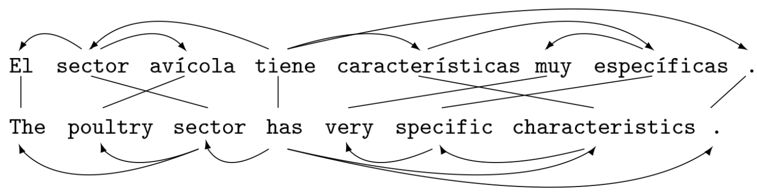
- Transfer annotations from another language
  - [Ganchev et al. 09]
- Constrain the number of child/parent relations
  - [Gillenwater et al. 11]
- Use linguistic rules
  - [Druck et al. 09] [Naseem et al. 10]

Dependency Parsing
Transfer annotations
[Ganchev et al. 09]



- Use information from a resource rich language
- Make the annotation transfer robust
- Preserve n % of the edges

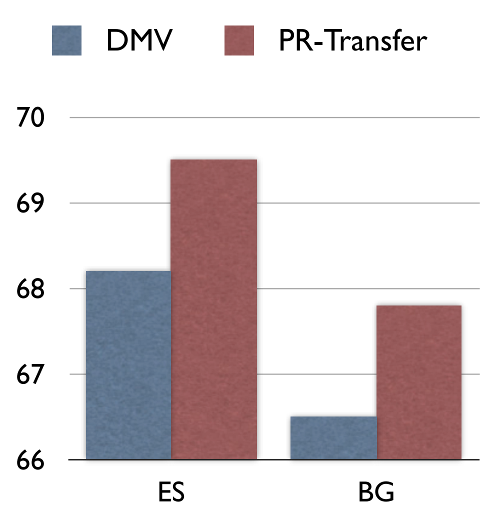
Dependency Parsing
Transfer annotations
[Ganchev et al. 09]



$$\mathbf{E}_q[\phi(\mathbf{x}, \mathbf{y})] = \frac{1}{|C_{\mathbf{x}}|} \sum_{y \in C_{\mathbf{x}}} q(y|\mathbf{x})$$

$$\mathbf{E}_q[\phi(\mathbf{x}, \mathbf{y})] \geq b$$

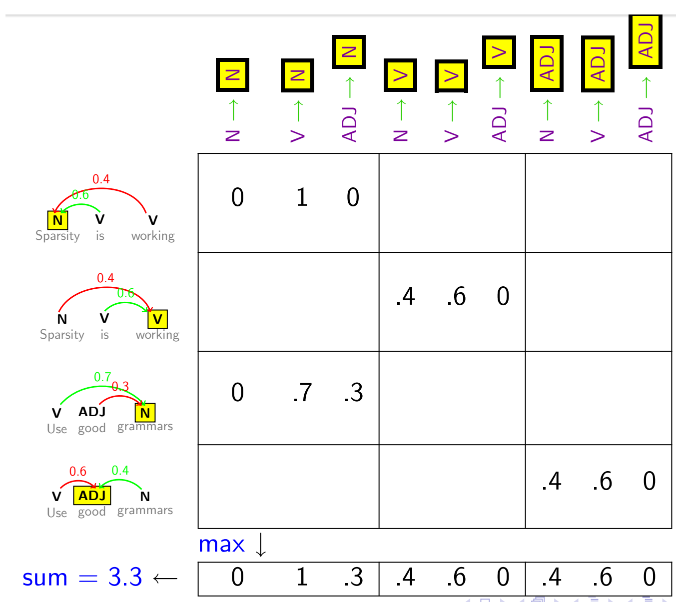
Dependency Parsing
Transfer annotations
[Ganchev et al. 09]



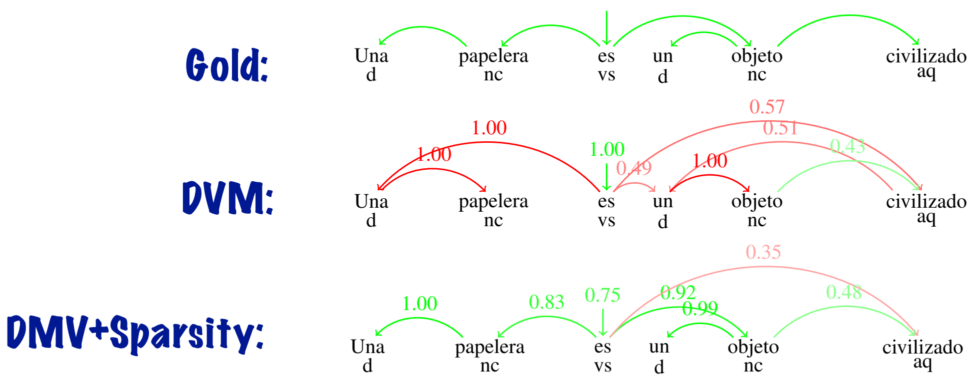
Dependency Parsing
Posterior Sparsity
[Graça et al. 10]

- ML learns very ambiguous grammars
  - all productions have some probability
  - constrain the number of possible productions

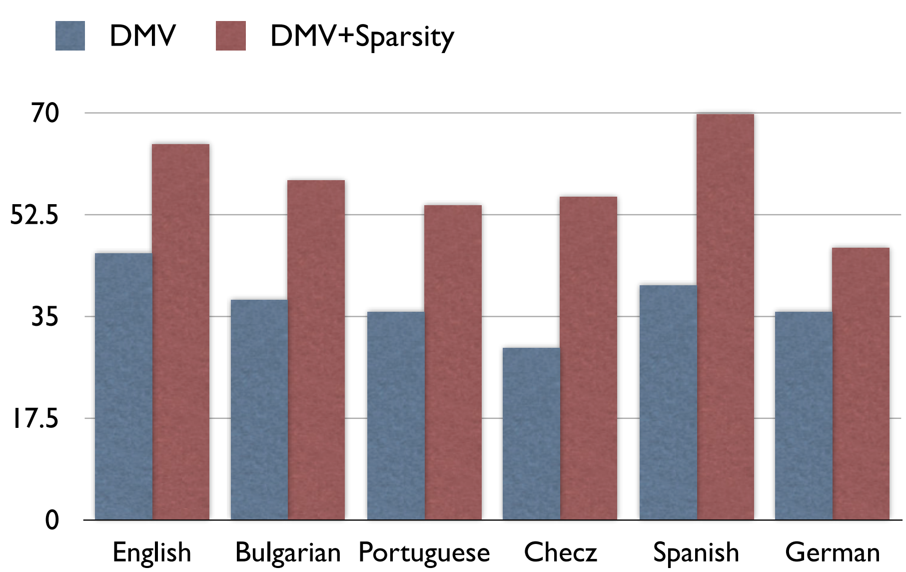
Dependency Parsing
Posterior Sparsity
[Gillenwater et al. 11]



Dependency Parsing
Posterior Sparsity
[Gillenwater et al. 11]



Dependency Parsing
Posterior Sparsity
[Gillenwater et al. 11]



Dependency Parsing
Linguistic Rules
[Naseem et al. 10]

Small set of universal rules

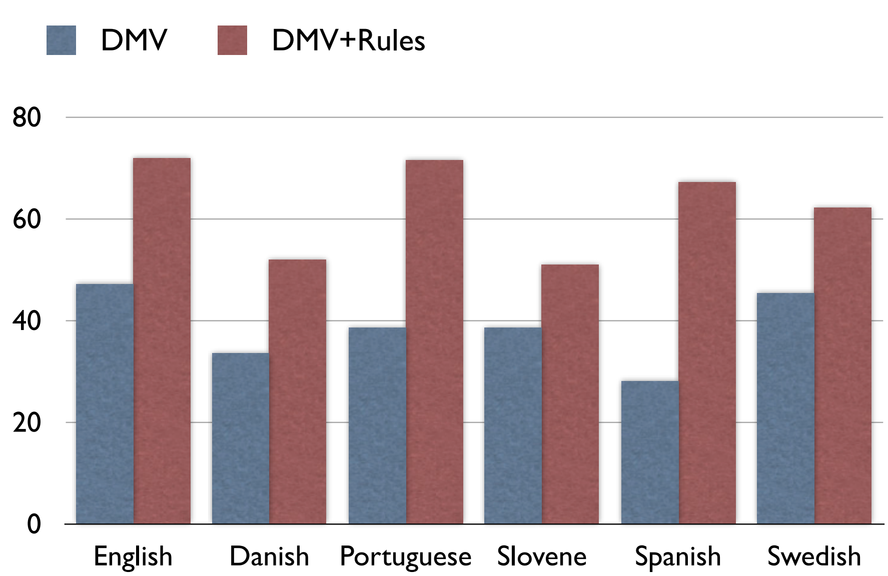
Root → Auxiliary	Noun → Adjective
Root → Verb	Noun → Article
Verb → Noun	Noun → Noun
Verb → Pronoun	Noun → Numeral
Verb → Adverb	Preposition → Noun
Verb → Verb	Adjective → Adverb
Auxiliary → Verb	

$\phi(\mathbf{x}, \mathbf{y})$

= 1 if edge in rule set

$$\mathbf{E}_q[\phi(\mathbf{x}, \mathbf{y})] \geq b$$

Dependency Parsing
Linguistic Rules
[Naseem et al. 10]



Dependency Parsing: Applications using Other Models

- **Tree CRF**
  - [Druck et al. 09]
- **MST Parser**
  - [Ganchev et al. 09]

Other Applications

- **Multi view learning:**
  - [Ganchev et al. 08]
- **Relation extraction:**
  - [Chen et al. 11]

Implementation Tips and Tricks

Off-the-Shelf Tools: MALLET

http://mallet.cs.umass.edu

- *off-the-shelf* support for **labeled features**
- **models:** *MaxEnt Classifier*, *Linear Chain CRF* (*one* and *two label* constraints)
- **methods:** *GE* and *PR*
- **constraints** on label distributions for input features
- **GE penalties:** KL divergence, ℓ_2^2 (+ soft inequalities)
- **PR penalties:** ℓ_2^2 (+ soft inequalities)
- **in development:** Tree CRF, ℓ_1 and other penalties

Off-the-Shelf Tools: MALLET

http://mallet.cs.umass.edu

- **import data** in *SVMLight-like or CoNLL03-like* formats

```
positive interesting:2 film:1 ...
negative tired:1 sequel:1 ...
positive best:1 recommend:2 ...
```

```
U.N.        NNP   B-NP   B-ORG
official    NN    I-NP   O
heads       VBZ   B-VP   O
```

- **import constraints** in a simple text format:

```
tired negative:0.8 positive:0.2
best positive:0.9 negative:0.1
```

```
U.N. B-ORG:0.7,0.9
B-VP O:0.95,
```

- easily **specify method options** (*i.e. SimpleTagger*):

```
java cc.mallet.fst.semi_supervised.tui.SemiSupSimpleTagger \
--train true --test lab --loss l2 --learning ge \
unlabeled.txt test.txt constraints.txt
```

New GE Constraints: MALLET

http://mallet.cs.umass.edu

- *Java Interfaces* for implementing **new** GE constraints
- covariance computation implemented (MaxEnt, CRF)
- *primarily* need to write methods to:

compute constraint features and expectations
compute GE objective value
compute GE objective gradient (but not covariance)

- **restriction:** constraints must factor with model
- **restriction:** GE objective must be differentiable

New PR Constraints: MALLET

http://mallet.cs.umass.edu

- *Java Interfaces* for implementing **new** PR constraints
- inference algorithms implemented (MaxEnt, CRF)
- *primarily* need to write methods for E-step (projection):

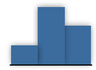
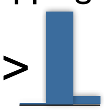
`compute constraint features and expectations`
`compute scores under q for E-step`
`compute objective function for E-step`
`compute gradient for E-step`

- **restriction:** constraints must factor with model

GE Implementation Advice

- **computing covariance** (required for gradient):
  - **trick:** compute cov. of composite constraint feature
    - **example:** ℓ_2^2 penalty: $\phi_c(\mathbf{x},\mathbf{y}) = \sum_{\phi} 2(\tilde{\phi} - \mathbf{E}[\phi])\phi(\mathbf{x},\mathbf{y})$
    - **result:** only need to store vectors of size $\dim(\mathbf{f})$ in computation, rather than covariance matrix
  - **trick:** efficient gradient computation in *hypergraphs*
    - use semiring algorithms of [Li & Eisner 09]
    - **result:** same time complexity as supervised (w. both)

GE Implementation Advice

- **parameter regularization:**
  - ℓ_2^2 regularization encourages bootstrapping by penalizing very large parameter values:  > 
- **optimization:** non-convex
  - usually *L-BFGS* still preferable (use "restart trick")
  - *zero initialization* usually works well
  - other init: supervised, MaxEnt, GE in simpler model

Off-the-Shelf Tools: PR Toolkit

http://code.google.com/p/pr-toolkit/

- off-the-shelf support for **PR**
- **models:**
  - MaxEnt Classifier, HMM,DMV
- **applications:**
  - Word Alignment, Pos Induction, Grammar Induction
- **constraints:** posterior sparsity, bijectivity, agreement
- No command line mode
- Smaller support base

PR Implementation example:
Word Alignment - Bijectivity

- **Learning:** EM, PR
  - `void eStep(counts, lattices);`
  - `void mStep(counts);`
  - `lattice constraint.project(lattice);`
- **Model:** HMM
  - `lattice computePosteriors(lattice);`
  - `void addCount(lattice, counts);`
  - `void updateParameters(counts);`
- **Constraints:** Bijectivity
  - `lattice project(lattice);`

PR Implementation example:
EM

```
class EM {

 model;


void em(n){
 lattices= model.getLattices();
 counts = model.counts();
 for(i=0; i< n; i++) {
   eStep(counts, lattices);
   mStep(counts);
 }
}
```

```
void eStep(counts, lattices) {
   counts.clear();
   for(l : lattices)  {
    model.computePosterior(l);
    model.addCount(l,counts);
   }
}


void mStep(counts) {
   model.updateParameters(counts);
}

   ......

}
```

PR Implementation example: PR

```
class PR {

 model;
 constraint;


void em(n){
 lattices= model.getLattices();
 counts = model.counts();
 for(i=0; i< n; i++) {
   eStep(counts, lattices);
   mStep(counts);
 }
}
```

```
void eStep(counts, lattices) {
   counts.clear();
   for(l : lattices){
    model.computePosterior(l);
    constraint.project(l);
    model.addCount(l,counts);
   }
}


void mStep(counts) {
   model.updateParameters(counts);
}

   ......

}
```

PR Implementation example: HMM

```
class HMM {

 obsProb, transProbs,initProbs;


lattice computerPosteriors(lattice){
 "Run forward backward"
}

void addCount(lattice,counts){
 "Add posteriors to count table"
}

void updateParams(counts){
 "Normalize counts"
 "Copy counts to params table"
}
```

```
void getCounts(){
 "return copy of params structures"
}

void getLattices(){
 "return structure of all lattices
in the corpus"
}

   ......

}
```

PR Implementation example: Bijective constraints

- **Constraint:** returns a vector with *mth* value = number of target words in sentence x that align with source word *m*

$$\phi(\mathbf{x},\mathbf{y}) = \sum_{i=1}^{N} \mathbf{1}(y_i = m) \qquad \mathcal{Q} = \{q : \mathbf{E}_q[\phi(\mathbf{x},\mathbf{y})] \leq 1\}$$

- **Primal:** Hard

$$\mathcal{D}_{\mathrm{KL}}(\mathcal{Q}|p_\theta) = \arg\min_q \mathcal{D}_{\mathrm{KL}}(q|p_\theta)$$

- **Dual:** Easy

$$\arg\max_{\lambda \geq 0} \; -b^T \cdot \lambda - \log Z(\lambda) - ||\lambda||_2$$

$$Z(\lambda) = \sum_y p_\theta(\mathbf{y}|\mathbf{x}) \exp(-\lambda \cdot \phi(\mathbf{x},\mathbf{y}))$$

PR Implementation example: Bijective Constraints

```
class BijectiveConstraints {
model;

lattice project(lattice){
 obj = BijectiveObj(model,lattice);
 Optimizer.optimize(obj);
}

}


class BijectiveObj {
  lattice;

void updateModel(newLambda){
 lattice_ = lattice*exp(newLambda);
 computerPosteriors(lattice)
}


double getObj(){
   obj = -dot(lambda,b);
   obj -= lattice.likelihood;
   obj -= l2Norm(lambda);
}

double[] getGrad(){
 grad = lattice.posteriors - b;
 grad -= norm(lambda);
 return grad;
}
```

Other Software Packages

- **Learning Based Java:**
  - http://cogcomp.cs.illinois.edu/page/software_view/11
  - support for Constraint-Driven Learning
- **Factorie:**
  - http://code.google.com/p/factorie/
  - support for GE and PR in development

Rich Prior Knowledge in Learning for Natural Language Processing

Bibliography

For a more up-to-date bibliography as well as additional information about these methods, point your browser to: `http://sideinfo.wikkii.com/`

1 Constraint-Driven Learning

Constraint driven learning (CoDL) was first introduced in Chang et al. [2007], and has been used also in Chang et al. [2008]. A further paper on the topic is in submission [Chang et al., 2010].

2 Generalized Expectation

Generalized Expectation (GE) constraints were first introduced by Mann and McCallum [2007][1] and were used to incorporate prior knowledge about the label distribution into semi-supervised classification. GE constraints have also been used to leverage "labeled features" in document classification [Druck et al., 2008] and information extraction [Mann and McCallum, 2008, Druck et al., 2009b, Bellare and McCallum, 2009], and to incorporate linguistic prior knowledge into dependency grammar induction [Druck et al., 2009a].

3 Posterior Regularization

The most clearly written overview of Posterior Regularization (PR) is Ganchev et al. [2010]. PR was first introduced in Graca et al. [2008], and has been applied to dependency grammar induction [Ganchev et al., 2009, Gillenwater et al., 2009, 2011, Naseem et al., 2010], part of speech induction [Graça et al., 2009a], multi-view learning [Ganchev et al., 2008], word alignment [Graca et al., 2008, Ganchev et al., 2009, Graça et al., 2009b], and cross-lingual semantic alignment [Platt et al., 2010]. The framework was independently discovered by Bellare et al. [2009] as an approximation to GE constraints, under the name Alternating Projections, and used under that name also by Singh et al. [2010] and Druck and McCallum [2010] for information extraction. The framework was also independently discovered by Liang et al. [2009] as an approximation to

[1] In Mann and McCallum [2007] the method was called *Expectation Regularization*.

a Bayesian model motivated by modeling prior information as measurements, and applied to information extraction.

4 Closely related frameworks

Quadrianto et al. [2009] introduce a distribution matching framework very closely related to GE constraints, with the idea that the model should predict the same feature expectations on labeled and undlabeled data for a set of features, formalized as a kernel.

Carlson et al. [2010] introduce a framework for semi-supervised learning based on constraints, and trained with an iterative update algorithm very similar to CoDL, but introducing only confident constraints as the algorithm progresses.

Gupta and Sarawagi [2011] introduce a framework for agreement that is closely related to the PR-based work in Ganchev et al. [2008], with a slightly different objective and a different training algorithm.

References

K. Bellare, G. Druck, and A. McCallum. Alternating projections for learning with expectation constraints. In *Proc. UAI*, 2009.

Kedar Bellare and Andrew McCallum. Generalized expectation criteria for bootstrapping extractors using record-text alignment. In *EMNLP*, pages 131–140, 2009.

Andrew Carlson, Justin Betteridge, Richard C. Wang, Estevam R. Hruschka Jr., and Tom M. Mitchell. Coupled Semi-Supervised Learning for Information Extraction. In *Proceedings of the Third ACM International Conference on Web Search and Data Mining (WSDM)*, 2010.

M. Chang, L. Ratinov, and D. Roth. Guiding semi-supervision with constraint-driven learning. In *Proc. ACL*, 2007.

Ming-Wei Chang, Lev Ratinov, and Dan Roth. Structured learning with constrained conditional models. 2010. In submission.

M.W. Chang, L. Ratinov, N. Rizzolo, and D. Roth. Learning and inference with constraints. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*. AAAI, 2008.

G. Druck, G. Mann, and A. McCallum. Learning from labeled features using generalized expectation criteria. In *Proc. SIGIR*, 2008.

G. Druck, G. Mann, and A. McCallum. Semi-supervised learning of dependency parsers using generalized expectation criteria. In *Proc. ACL-IJCNLP*, 2009a.

Gregory Druck and Andrew McCallum. High-performance semi-supervised learning using discriminatively constrained generative models. In *Proceedings of the International Conference on Machine Learning (ICML 2010)*, pages 319–326, 2010.

Gregory Druck, Burr Settles, and Andrew McCallum. Active learning by labeling features. In *EMNLP*, pages 81–90, 2009b.

K. Ganchev, J. Graça, J. Blitzer, and B. Taskar. Multi-view learning over structured and non-identical outputs. In *Proc. UAI*, 2008.

K. Ganchev, J. Gillenwater, and B. Taskar. Dependency grammar induction via bitext projection constraints. In *Proc. ACL-IJCNLP*, 2009.

Kuzman Ganchev, Joo Graa, Jennifer Gillenwater, and Ben Taskar. Posterior sparsity in unsupervised dependency parsing. *Journal of Machine Learning Research*, 11:2001–2049, July 2010. URL `http://jmlr.csail.mit.edu/papers/v11/ganchev10a.html`.

Jennifer Gillenwater, Kuzman Ganchev, Joo Graa, Ben Taskar, and Fernando Pereira. Sparsity in grammar induction. In *NIPS Workshop on Grammar Induction, Representation of Language and Language Learning*, 2009.

Jennifer Gillenwater, Kuzman Ganchev, Joo Graa, Fernando Pereira, and Ben Taskar. Posterior sparsity in unsupervised dependency parsing. *Journal of Machine Learning Research*, 12:455–490, February 2011. URL `http://jmlr.csail.mit.edu/papers/v12/gillenwater11a.html`.

Joao Graca, Kuzman Ganchev, and Ben Taskar. Expectation maximization and posterior constraints. In J.C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 569–576. MIT Press, Cambridge, MA, 2008.

J. Graça, K. Ganchev, F. Pereira, and B. Taskar. Parameter vs. posterior sparisty in latent variable models. In *Proc. NIPS*, 2009a.

J. Graça, K. Ganchev, and B. Taskar. Postcat - posterior constrained alignment toolkit. In *The Third Machine Translation Marathon*, 2009b.

Rahul Gupta and Sunita Sarawagi. Joint training for open-domain extraction on the web: exploiting overlap when supervision is limited. In *Proceedings of the Fourth ACM International Conference on Web Search and Data Mining (WSDM)*, 2011.

P. Liang, M. I. Jordan, and D. Klein. Learning from measurements in exponential families. In *Proc. ICML*, 2009.

G. S. Mann and A. McCallum. Simple, robust, scalable semi-supervised learning via expectation regularization. In *Proc. ICML*, 2007.

G. S. Mann and A. McCallum. Generalized expectation criteria for semi-supervised learning of conditional random fields. In *Proc. ACL*, 2008.

Tahira Naseem, Harr Chen, Regina Barzilay, and Mark Johnson. Using universal linguistic knowledge to guide grammar induction. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1234–1244, Cambridge, MA, October 2010. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/D10-1120`.

John Platt, Kristina Toutanova, and Wen-tau Yih. Translingual document representations from discriminative projections. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 251–261, Cambridge, MA, October 2010. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/D10-1025`.

Novi Quadrianto, James Petterson, and Alex Smola. Distribution matching for transduction. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 1500–1508. MIT Press, 2009.

Sameer Singh, Dustin Hillard, and Chris Leggetter. Minimally-supervised extraction of entities from text advertisements. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 73–81, Los Angeles, California, June 2010. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/N10-1009`.