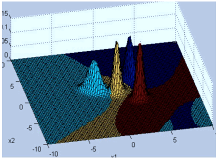
Log-Linear Models, Logistic Regression and Conditional Random Fields

February 21, 2013

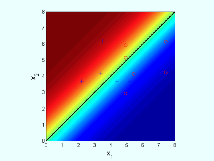
Generative, Conditional and Discriminative

- Given $\mathcal{D} = (x_t, y_t)_{t=1}^{T}$ sampled *iid* from unknown $P(x, y)$
- Generative Learning (maximum likelihood Gaussians)
  - Choose family of functions $p_{\boldsymbol{\theta}}(x, y)$ parametrized by $\boldsymbol{\theta}$
  - Find $\boldsymbol{\theta}$ by maximizing likelihood: $\prod_{t=1}^{T} p_{\boldsymbol{\theta}}(x_i, y_i)$
  - Given x, output $\hat{y} = \arg\max_y \frac{p_{\boldsymbol{\theta}}(x,y)}{\sum_y p_{\boldsymbol{\theta}}(x,y)}$
- Conditional Learning (logistic regression)
  - Choose family of functions $p_{\boldsymbol{\theta}}(y|x)$ parametrized by $\boldsymbol{\theta}$
  - Find $\boldsymbol{\theta}$ by maximizing conditional likelihood: $\prod_{t=1}^{T} p_{\boldsymbol{\theta}}(y_i|x_i)$
  - Given x, output $\hat{y} = \arg\max_y p_{\boldsymbol{\theta}}(y|x)$
- Discriminative Learning (support vector machines)
  - Choose family of functions $y = f_{\boldsymbol{\theta}}(x)$ parametrized by $\boldsymbol{\theta}$
  - Find $\boldsymbol{\theta}$ by minimizing classification error $\sum_{t=1}^{T} \ell(y_i, f_{\boldsymbol{\theta}}(x_i))$
  - Given x, output $\hat{y} = f_{\boldsymbol{\theta}}(x)$

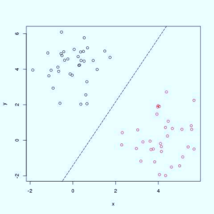
Generative, Conditional and Discriminative



Generative



Conditional



Discriminative

Generative: Maximum Entropy

Maximum entropy (or generally) minimum relative entropy $\mathcal{RE}(p\|h) = \sum_y p(y) \ln \frac{p(y)}{h(y)}$ subject to linear constraints

$$\min_p \mathcal{RE}(p\|h) \text{ s.t. } \sum_y p(y)\mathbf{f}(y) = \mathbf{0}, \sum_y p(y)\mathbf{g}(y) \geq \mathbf{0}$$

Solution distribution looks like an exponential family model

$$p(y) \quad = \quad h(y) \exp\left(\boldsymbol{\theta}^\top \mathbf{f}(y) + \boldsymbol{\vartheta}^\top \mathbf{g}(y)\right) / Z(\boldsymbol{\theta}, \boldsymbol{\vartheta})$$

Maximize the dual (the negative log-partition) to get $\boldsymbol{\theta}, \boldsymbol{\vartheta}$.

$$\max_{\theta, \boldsymbol{\vartheta} \geq \mathbf{0}} - \ln Z(\boldsymbol{\theta}, \boldsymbol{\vartheta}) \quad = \quad \max_{\theta, \boldsymbol{\vartheta} \geq \mathbf{0}} - \ln \sum_y h(y) \exp\left(\boldsymbol{\theta}^\top \mathbf{f}(y) + \boldsymbol{\vartheta}^\top \mathbf{g}(y)\right)$$

Generative: Exponential Family and Maximum Likelihood

All maximum entropy models give an exponential family form:

$$p(y) \quad = \quad h(y)\exp(\boldsymbol{\theta}^\top \mathbf{f}(y) - a(\boldsymbol{\theta}))$$

This is also a *log-linear model* over discrete $y \in \Omega$ where $|\Omega| = n$

$$p(y|\boldsymbol{\theta}) \quad = \quad \frac{1}{Z(\boldsymbol{\theta})} h(y)\exp\left(\boldsymbol{\theta}^\top \mathbf{f}(y)\right)$$

- Parameters are vector $\boldsymbol{\theta} \in \mathbb{R}^d$
- Features are $\mathbf{f} : \Omega \mapsto \mathbb{R}^d$ mapping each y to some vector
- Prior is $h : \Omega \mapsto \mathbb{R}^+$ a fixed non-negative measure
- Partition function ensures that $p(y|\boldsymbol{\theta})$ normalizes

$$Z(\boldsymbol{\theta}) \quad = \quad \sum_y h(y)\exp(\boldsymbol{\theta}^\top \mathbf{f}(y))$$

Generative: Exponential Family and Maximum Likelihood

We are given some *iid* data $y_1, \ldots, y_T$ where $y \in \{0, 1\}$. If we wanted to find the best parameters of an exponential family distribution known as the Bernouilli distribution:

$$\begin{aligned} p(y|\boldsymbol{\theta}) &= h(y) \exp(\boldsymbol{\theta}^\top \mathbf{f}(y) - a(\boldsymbol{\theta})) \\ &= \theta^y (1-\theta)^{1-y} \end{aligned}$$

This is unsupervised generative learning

We simply find the $\boldsymbol{\theta}$ that maximizes the likelihood

$$L(\boldsymbol{\theta}) = \prod_{t=1}^{T} p(y_t|\boldsymbol{\theta}) = \theta^{\sum_t y_t} (1-\theta)^{T - \sum_t y_t}$$

Taking log then derivatives and setting to zero gives $\theta = \frac{1}{T} \sum_t y_t$.

Conditional: Logistic Regression

Given input-output *iid* data $(x_1, y_1), \ldots, (x_T, y_T)$ where $y \in \{0, 1\}$.
Binary logistic regression computes a probability for $y = 1$ by

$$p(y = 1|x, \boldsymbol{\vartheta}) = \frac{1}{1 + \exp(-\boldsymbol{\vartheta}^\top \phi(x))}.$$

And the probability for $p(y = 0|x, \boldsymbol{\theta}) = 1 - p(y = 1|x, \boldsymbol{\theta})$.
This is supervised conditional learning.
We find the $\boldsymbol{\theta}$ that maximizes the *conditional* likelihood

$$L(\boldsymbol{\vartheta}) = \prod_{t=1}^{T} p(y_t|x_t, \boldsymbol{\vartheta})$$

We can maximize this by doing gradient ascent.
Logistic regression is an example of a *log-linear model*.

Conditional: Log-linear Models

Like an exponential family, but allow Z, h and $\mathbf{f}$ also depend on x

$$p(y|x, \boldsymbol{\theta}) = \frac{1}{Z(x, \boldsymbol{\theta})} h(x, y) \exp\left(\boldsymbol{\theta}^\top \mathbf{f}(x, y)\right)$$

- Parameters are just one long vector $\boldsymbol{\theta} \in \mathbb{R}^d$
- Functions $\mathbf{f} : \Omega_x \times \Omega_y \mapsto \mathbb{R}^d$ map x, y to a vector
- Prior is $h : \Omega_x \times \Omega_y \mapsto \mathbb{R}^+$ a fixed non-negative measure
- Partition function ensures that $p(y|x, \boldsymbol{\theta})$ normalizes

To make a prediction, we simply output

$$\hat{y} = \arg\max_y p(y|x, \boldsymbol{\theta}).$$

Let's mimic (multi-class) logistic regression with this form.

Conditional: Log-linear Models

In multi-class logistic regression, we have $y \in \{1, \ldots, n\}$.

$$p(y|x, \boldsymbol{\theta}) = \frac{1}{Z(x, \boldsymbol{\theta})} h(x, y) \exp\left(\boldsymbol{\theta}^\top \mathbf{f}(x, y)\right)$$

If $\phi(x) \in \mathbb{R}^k$, then $\mathbf{f}(x, y) \in \mathbb{R}^{kn}$.
Choose the following for the feature function

$$\mathbf{f}(x, y) = \left[\delta[y = 1]\phi(x)^\top \;\; \delta[y = 2]\phi(x)^\top \ldots \delta[y = n]\phi(x)^\top\right]^\top.$$

If $n = 2$ and $h(x, y) = 1$, get traditional binary logistic regression!

Conditional: Log-linear Models

Rewrite binary logistic regresion $p(y = 1|x, \boldsymbol{\vartheta}) = \frac{1}{1+\exp(-\boldsymbol{\vartheta}^\top \phi(x))}$ as a log-linear model with $n = 2$, $h(x, y) = 1$ and $\mathbf{f}(x, y)$ as before

$$
\begin{aligned}
p(y|x, \boldsymbol{\theta}) &= \frac{h(x, y) \exp\left(\boldsymbol{\theta}^\top \mathbf{f}(x, y)\right)}{Z(x, \boldsymbol{\theta})} \\
&= \frac{\exp\left(\mathbf{f}(x, y)^\top \boldsymbol{\theta}\right)}{\sum_{y=0}^{1} \exp\left(\mathbf{f}(x, y)^\top \boldsymbol{\theta}\right)} \\
p(y = 1|x, \boldsymbol{\theta}) &= \frac{\exp\left([\mathbf{0} \;\; \phi(x)^\top]\boldsymbol{\theta}\right)}{\exp\left([\phi(x)^\top \;\; \mathbf{0}]\boldsymbol{\theta}\right) + \exp\left([\mathbf{0} \;\; \phi(x)^\top]\boldsymbol{\theta}\right)} \\
&= \frac{1}{1 + \exp\left([\phi(x)^\top \;\; \mathbf{0}]\boldsymbol{\theta} - [\mathbf{0} \;\; \phi(x)^\top]\boldsymbol{\theta}\right)}
\end{aligned}
$$

Can you see how to write $\boldsymbol{\vartheta}$ in terms of $\boldsymbol{\theta}$?

Conditional Random Fields (CRFs)

- Conditional random fields generalize maximum entropy
- Trained on *iid* data $\{(x_1, y_1), \ldots, (x_t, y_t)\}$
- A CRF is just a log-linear model with big n

$$p(y|x_j, \boldsymbol{\theta}) \quad = \quad \frac{1}{Z(x_j, \boldsymbol{\theta})} h(x_j, y) \exp(\boldsymbol{\theta}^\top \mathbf{f}(x_j, y))$$

- Maximum conditional log-likelihood objective function is

$$J(\boldsymbol{\theta}) = \sum_{j=1}^{t} \ln \frac{h(x_j, y_j)}{Z(x_j, \boldsymbol{\theta})} + \boldsymbol{\theta}^\top \mathbf{f}(x_j, y_j) \tag{1}$$

- Regularized conditional maximum likelihood is

$$J(\boldsymbol{\theta}) = \sum_{j=1}^{t} \ln \frac{h(x_j, y_j)}{Z(x_j, \boldsymbol{\theta})} + \boldsymbol{\theta}^\top \mathbf{f}(x_j, y_j) - \tfrac{t\lambda}{2} \|\boldsymbol{\theta}\|^2 \tag{2}$$

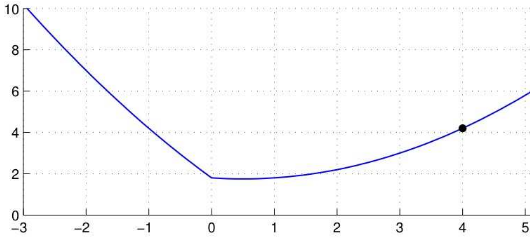
Conditional Random Fields (CRFs)

- To train a CRF, we maximize (regularized) conditional likelihood
- Traditionally, maximum entropy, log-linear models and CRFs were trained using *majorization* (the EM algorithm is a majorization method)
- The algorithms were called *improved iterative scaling (IIS)* or *generalized iterative scaling (GIS)*
    - Maximum entropy [Jaynes '57]
    - Conditional random fields [Lafferty, et al. '01]
    - Log-linear models [Darroch & Ratcliff '72]

Majorization

If cost function $\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} C(\boldsymbol{\theta})$ has no closed form solution Majorization uses with a surrogate Q with closed form update to monotonically minimize the cost from an initial $\boldsymbol{\theta}_0$

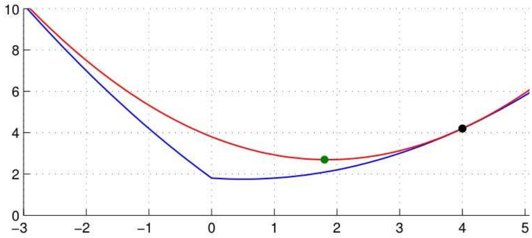
- Find bound $Q(\boldsymbol{\theta}, \boldsymbol{\theta}_i) \geq C(\boldsymbol{\theta})$ where $Q(\boldsymbol{\theta}_i, \boldsymbol{\theta}_i) = C(\boldsymbol{\theta}_i)$
- Update $\boldsymbol{\theta}_{i+1} = \arg\min_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}, \boldsymbol{\theta}_i)$
- Repeat until converged



Majorization

If cost function $\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} C(\boldsymbol{\theta})$ has no closed form solution Majorization uses with a surrogate Q with closed form update to monotonically minimize the cost from an initial $\boldsymbol{\theta}_0$

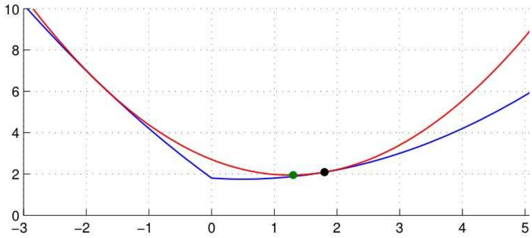
- Find bound $Q(\boldsymbol{\theta}, \boldsymbol{\theta}_i) \geq C(\boldsymbol{\theta})$ where $Q(\boldsymbol{\theta}_i, \boldsymbol{\theta}_i) = C(\boldsymbol{\theta}_i)$
- Update $\boldsymbol{\theta}_{i+1} = \arg\min_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}, \boldsymbol{\theta}_i)$
- Repeat until converged



Majorization

If cost function $\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} C(\boldsymbol{\theta})$ has no closed form solution
Majorization uses with a surrogate Q with closed form update
to monotonically minimize the cost from an initial $\boldsymbol{\theta}_0$

- Find bound $Q(\boldsymbol{\theta}, \boldsymbol{\theta}_i) \geq C(\boldsymbol{\theta})$ where $Q(\boldsymbol{\theta}_i, \boldsymbol{\theta}_i) = C(\boldsymbol{\theta}_i)$
- Update $\boldsymbol{\theta}_{i+1} = \arg\min_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}, \boldsymbol{\theta}_i)$
- Repeat until converged



Majorization

IIS and GIS were preferred until [Wallach '03, Andrew & Gao '07]

Method	Iterations	LL Evaluations	Time (s)
IIS	≥ 150	≥ 150	≥ 188.65
Conjugate gradient (FR)	19	99	124.67
Conjugate gradient (PRP)	27	140	176.55
L-BFGS	22	22	29.72

Gradient descent appears to be faster

But newer majorization methods are faster still

Gradient Ascent for CRFs

We have the following model

$$p(y|x,\boldsymbol{\theta}) = \frac{1}{Z(x,\boldsymbol{\theta})} h(x,y) \exp\left(\boldsymbol{\theta}^\top \mathbf{f}(x,y)\right)$$

We want to maximize the conditional (log) likelihood:

$$\begin{aligned}
\log L(\boldsymbol{\theta}) &= \sum_{t=1}^{T} \log p(y_t|x_t,\boldsymbol{\theta}) \\
&= \sum_{t=1}^{T} -\log Z(x_t,\boldsymbol{\theta}) + \log(h(x_t,y_t)) + \boldsymbol{\theta}^\top \mathbf{f}(x_t,y_t) \\
&= const - \sum_{t=1}^{T} \log Z(x_t,\boldsymbol{\theta}) + \boldsymbol{\theta}^\top \sum_{t=1}^{T} \mathbf{f}(x_t,y_t)
\end{aligned}$$

Same as minimizing the sum of log partition functions plus linear!

Gradient Ascent for CRFs

$$
\begin{aligned}
\frac{\partial \log L}{\partial \boldsymbol{\theta}} &= \frac{\partial}{\partial \boldsymbol{\theta}} \left(\boldsymbol{\theta}^\top \sum_{t=1}^{T} \mathbf{f}(x_t, y_t) - \sum_{t=1}^{T} \log Z(x_t, \boldsymbol{\theta}) \right) \\
&= \sum_{t=1}^{T} \mathbf{f}(x_t, y_t) - \sum_{t=1}^{T} \frac{1}{Z(x_t, \boldsymbol{\theta})} \sum_{y} h(x_t, y) \frac{\partial}{\partial \boldsymbol{\theta}} \exp\left(\boldsymbol{\theta}^\top \mathbf{f}(x_t, y) \right) \\
&= \sum_{t=1}^{T} \mathbf{f}(x_t, y_t) - \sum_{t=1}^{T} \sum_{y} \frac{h(x_t, y)}{Z(x_t, \boldsymbol{\theta})} \exp\left(\boldsymbol{\theta}^\top \mathbf{f}(x_t, y) \right) \mathbf{f}(x_t, y) \\
&= \sum_{t=1}^{T} \mathbf{f}(x_t, y_t) - \sum_{t=1}^{T} \sum_{y} \mathbf{f}(x_t, y) p(y | x_t, \boldsymbol{\theta})
\end{aligned}
$$

The gradient is the difference between the feature vectors at the true labels minus the expected feature vectors under the current distribution. To update, $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \eta \frac{\partial \log L}{\partial \boldsymbol{\theta}}$.

Stochastic Gradient Ascent for CRFs

Given current $\boldsymbol{\theta}$, update by taking a small step along the gradient

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \eta \frac{\partial \log L}{\partial \boldsymbol{\theta}}.$$

We can use the full derivative:

$$\frac{\partial \log L}{\partial \boldsymbol{\theta}} = \sum_{t=1}^{T} \mathbf{f}(x_t, y_t) - \sum_{t=1}^{T} \sum_{y} \mathbf{f}(x_t, y) p(y|x_t, \boldsymbol{\theta})$$

Or do stochastic gradient with only a single random datapoint t:

$$\frac{\partial \log L}{\partial \boldsymbol{\theta}} = \mathbf{f}(x_t, y_t) - \sum_{y} \mathbf{f}(x_t, y) p(y|x_t, \boldsymbol{\theta})$$

Better Majorization for CRFs

Recall log-linear model over discrete $y \in \Omega$ where $|\Omega| = n$

$$p(y|\boldsymbol{\theta}) = \frac{1}{Z(\boldsymbol{\theta})} h(y) \exp\left(\boldsymbol{\theta}^\top \mathbf{f}(y)\right)$$

- Parameters are vector $\boldsymbol{\theta} \in \mathbb{R}^d$
- Features are $\mathbf{f} : \Omega \mapsto \mathbb{R}^d$ mapping each y to some vector
- Prior is $h : \Omega \mapsto \mathbb{R}^+$ a fixed non-negative measure
- Partition function ensures that $p(y|\boldsymbol{\theta})$ normalizes

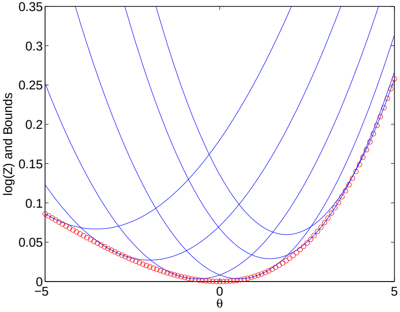
$$Z(\boldsymbol{\theta}) = \sum_y h(y) \exp(\boldsymbol{\theta}^\top \mathbf{f}(y))$$

Problem: it's ugly to minimize (unlike a quadratic function)

Better Majorization for CRFs

The bound $\ln Z(\boldsymbol{\theta}) \leq \ln z + \frac{1}{2}(\boldsymbol{\theta} - \tilde{\boldsymbol{\theta}})^\top \boldsymbol{\Sigma}(\boldsymbol{\theta} - \tilde{\boldsymbol{\theta}}) + (\boldsymbol{\theta} - \tilde{\boldsymbol{\theta}})^\top \boldsymbol{\mu}$ is tight at $\tilde{\boldsymbol{\theta}}$ and holds for parameters given by

Input $\tilde{\boldsymbol{\theta}}, \mathbf{f}(y), h(y)\ \forall y \in \Omega$
Init $z \to 0^+, \boldsymbol{\mu} = \mathbf{0}, \boldsymbol{\Sigma} = z\mathbf{I}$ For each $y \in \Omega$ { $\alpha = h(y)\exp(\tilde{\boldsymbol{\theta}}^\top \mathbf{f}(y))$ $\mathbf{l} = \mathbf{f}(y) - \boldsymbol{\mu}$ $\boldsymbol{\Sigma} += \frac{\tanh(\frac{1}{2}\ln(\alpha/z))}{2\ln(\alpha/z)} \mathbf{l}\mathbf{l}^\top$ $\boldsymbol{\mu} += \frac{\alpha}{z+\alpha}\mathbf{l}$ $z += \alpha$ }
Output $z, \boldsymbol{\mu}, \boldsymbol{\Sigma}$



Better Majorization for CRFs

Bound Proof.

1) Start with bound $\log(e^{\theta} + e^{-\theta}) \leq c\theta^2$ [Jaakkola & Jordan '99]
2) Prove scalar bound via Fenchel dual using $\theta = \sqrt{\vartheta}$
3) Make bound multivariate $\log(e^{\boldsymbol{\theta}^\top \mathbf{1}} + e^{-\boldsymbol{\theta}^\top \mathbf{1}})$
4) Handle scaling of exponentials $\log(h_1 e^{\boldsymbol{\theta}^\top \mathbf{f}_1} + h_2 e^{-\boldsymbol{\theta}^\top \mathbf{f}_2})$
5) Add one term $\log(h_1 e^{\boldsymbol{\theta}^\top \mathbf{f}_1} + h_2 e^{-\boldsymbol{\theta}^\top \mathbf{f}_2} + h_3 e^{-\boldsymbol{\theta}^\top \mathbf{f}_3})$
6) Repeat extension for n terms $\square$

Better Majorization for CRFs (Bound also Finds Gradient)

Init $z \to 0^+, \boldsymbol{\mu} = \mathbf{0}, \boldsymbol{\Sigma} = z\mathbf{I}$ For each $y \in \Omega$ { $\alpha \;\; = h(y)\exp(\tilde{\boldsymbol{\theta}}^\top \mathbf{f}(y))$ $\mathbf{l} \;\; = \mathbf{f}(y) - \boldsymbol{\mu}$ $\boldsymbol{\Sigma} += \frac{\tanh(\frac{1}{2}\ln(\alpha/z))}{2\ln(\alpha/z)}\mathbf{l}\mathbf{l}^\top$ $\boldsymbol{\mu} += \frac{\alpha}{z+\alpha}\mathbf{l}$ $z \;+= \alpha$ }
Output $z, \boldsymbol{\mu}, \boldsymbol{\Sigma}$

Recall gradient $$\frac{\partial \log L}{\partial \boldsymbol{\theta}} = \sum_{t=1}^{T} \mathbf{f}(x_t, y_t) - \sum_{t=1}^{T} \sum_{y} \mathbf{f}(x_t, y) p(y|x_t, \boldsymbol{\theta})$$

The bound's $\boldsymbol{\mu}$ give part of gradient (can skip $\boldsymbol{\Sigma}$ updates).

$$\boldsymbol{\mu} = \sum_{y} \mathbf{f}(x_t, y) p(y|x_t, \boldsymbol{\theta})$$

Better Majorization for CRFs

Input x_j, y_j and functions $h_{x_j}, \mathbf{f}_{x_j}$ for $j=1,\ldots,t$ Input regularizer $\lambda \in \mathbb{R}^+$

Initialize $\boldsymbol{\theta}_0$ anywhere and set $\tilde{\boldsymbol{\theta}} = \boldsymbol{\theta}_0$
While not converged
  For $j = 1$ to t compute bound for $\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j$ from $h_{x_j}, \mathbf{f}_{x_j}, \tilde{\boldsymbol{\theta}}$
  Set $\tilde{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \sum_j \frac{1}{2}(\boldsymbol{\theta} - \tilde{\boldsymbol{\theta}})^\top(\boldsymbol{\Sigma}_j + \lambda \mathbf{I})(\boldsymbol{\theta} - \tilde{\boldsymbol{\theta}})$
$+ \sum_j \boldsymbol{\theta}^\top(\boldsymbol{\mu}_j - \mathbf{f}_{x_j}(y_j) + \lambda\tilde{\boldsymbol{\theta}})$

Output $\hat{\boldsymbol{\theta}} = \tilde{\boldsymbol{\theta}}$

Theorem

If $\|\mathbf{f}(x_j, y)\| \leq r$ *get* $J(\hat{\boldsymbol{\theta}}) - J(\boldsymbol{\theta}_0) \geq (1-\epsilon)\max_{\boldsymbol{\theta}}(J(\boldsymbol{\theta}) - J(\boldsymbol{\theta}_0))$ *within* $\left\lceil \ln(1/\epsilon) / \ln\left(1 + \frac{\lambda \log n}{2r^2 n}\right)\right\rceil$ *steps*

Convergence Proof

Proof.

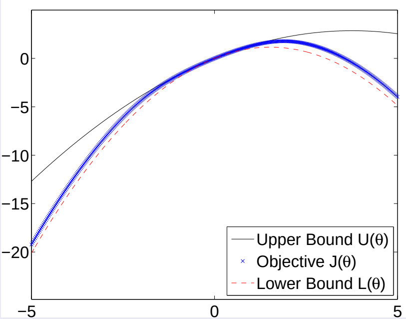


Figure: Quadratic bounding sandwich. Compare upper and lower bound curvatures to bound maximum # of iterations.

□

Experiments - Multi-Class Classification & Linear Chains

Data-set	SRBCT		Tumors		Text		SecStr		CoNLL		PennTree	
Size	$n=4$ $t=83$ $d=9236$ $\lambda=10^1$		$n=26$ $t=308$ $d=390260$ $\lambda=10^1$		$n=2$ $t=1500$ $d=23922$ $\lambda=10^2$		$n=2$ $t=83679$ $d=632$ $\lambda=10^1$		$m=9$ $t=1000$ $d=33615$ $\lambda=10^1$		$m=45$ $t=1000$ $d=14175$ $\lambda=10^1$	
Algorithm	time	iter	time	iter	time	iter	time	iter	time	iter	time	iter
LBFGS	6.10	42	3246.83	8	15.54	7	881.31	47	25661.54	17	62848.08	7
Grad	7.27	43	18749.15	53	153.10	69	1490.51	79	93821.72	12	156319.31	12
Congrad	40.61	100	14840.66	42	57.30	23	667.67	36	88973.93	23	76332.39	18
Bound	**3.67**	**8**	**1639.93**	**4**	**6.18**	**3**	**27.97**	**9**	**16445.93**	**4**	**27073.42**	**2**

Table: Time in seconds and iterations to match LBFGS solution for multi-class logistic regression (on SRBCT, Tumors, Text and SecStr data-sets where n is the number of classes) and Markov CRFs (on CoNLL and PennTree data-sets, where m is the number of classes). Here, t is the number of samples, d is the dimensionality of the feature vector and λ is the cross-validated regularization setting.

Experiments - Linear Chains

Model	Error	oov Error
Hidden Markov Model	5.69%	45.59%
Maximum Entropy Markov Model	6.37%	54.61%
Conditional Random Field	5.55%	48.05%

Table: Accuracy on Penn tree-bank data-set for parts-of-speech tagging with training on half of the 1.1 million word corpus. Note, the oov rate is the error rate on out-of-vocabulary words.

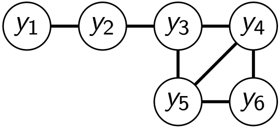
Parts of speech data-set where there are 45 labels per word, e.g.

PRP	VBD	DT	NN	IN	DT	NN
|	|	|	|	|	|	|
I	saw	the	man	with	the	telescope

$$p(y|\mathbf{x},\boldsymbol{\theta}) = \frac{1}{Z}\psi(y_1,y_2)\psi(y_2,y_3)\psi(y_3,y_4)\psi(y_4,y_5)\psi(y_5,y_6)\psi(y_6,y_7)$$

How big is y? Recall graphical models for large spaces...

Bounding Graphical Models with Large n

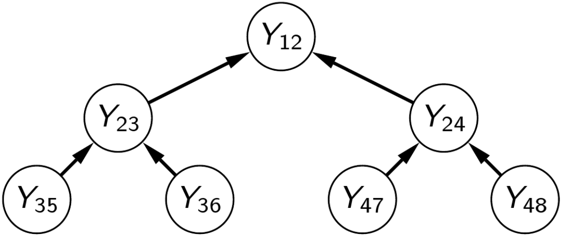


- Each iteration is $\mathcal{O}(tn)$, but what if n is large?
- Graphical model: an undirected graph G representing a distribution $p(Y)$ where $Y = \{y_1, \ldots, y_n\}$ and $y_i \in \mathbb{Z}$
- $p(Y)$ factorizes as product of $\{\psi_1, \ldots, \psi_C\}$ functions over $\{Y_1, \ldots, Y_C\}$ subsets of variables over the maximal cliques of G

$$p(y_1, \ldots, y_n) = \frac{1}{Z} \prod_{c \in C} \psi_c(Y_c)$$

- E.g. $p(y_1, \ldots, y_6) = \psi(y_1, y_2)\psi(y_2, y_3)\psi(y_3, y_4, y_5)\psi(y_4, y_5, y_6)$

Bounding Graphical Models with Large n



- Instead of enumerating over all n, exploit graphical model
- Build junction tree and run a *Collect* algorithm
- Useful for computing $Z(\boldsymbol{\theta})$, $\frac{\partial \log Z(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}$ and $\boldsymbol{\Sigma}$ efficiently
- Bound needs $\mathcal{O}(t \sum_c |Y_c|)$ rather than $\mathcal{O}(tn)$
- For an HMM, this is $\mathcal{O}(TM^2)$ instead of $\mathcal{O}(M^T)$

Bounding Graphical Models with Large n

for $c = 1, \ldots, m$ {

$Y_{both} = Y_c \cap Y_{pa(c)}$; $Y_{solo} = Y_c \setminus Y_{pa(c)}$

for each $u \in Y_{both}$ {

initialize $z_{c|x} \leftarrow 0^+$, $\boldsymbol{\mu}_{c|x} = \mathbf{0}$, $\boldsymbol{\Sigma}_{c|x} = z_{c|x}\mathbf{I}$

for each $v \in Y_{solo}$ {

$$w = u \otimes v; \quad \alpha_w = h_c(w) e^{\tilde{\boldsymbol{\theta}}^\top \mathbf{f}_c(w)} \prod_{b \in ch(c)} z_{b|w}$$

$$\mathbf{l}_w = \mathbf{f}_c(w) - \boldsymbol{\mu}_{c|u} + \sum_{b \in ch(c)} \boldsymbol{\mu}_{b|w}$$

$$\boldsymbol{\Sigma}_{c|u} += \sum_{b \in ch(c)} \boldsymbol{\Sigma}_{b|w} + \frac{\tanh(\frac{1}{2}\ln(\frac{\alpha_w}{z_{c|u}}))}{2\ln(\frac{\alpha_w}{z_{c|u}})} \mathbf{l}_w \mathbf{l}_w^\top$$

$$\boldsymbol{\mu}_{c|u} += \frac{\alpha_w}{z_{c|u} + \alpha_w} \mathbf{l}_w; \quad z_{c|u} += \alpha_w \quad \}\}\}$$