An SVM Approach for Natural Language Learning

Michael Collins

MIT EECS/CSAIL

Joint work with Peter Bartlett, David McAllester, Ben Taskar

Supervised Learning in NLP

- Goal is to learn a function $F : \mathcal{X} \rightarrow \mathcal{Y}$,
  where $\mathcal{X}$ is a set of possible inputs,
  $\mathcal{Y}$ is a set of possible outputs.

- We have a training sample $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$
  where each $(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}$
  E.g., each x_i is a sentence, each y_i is a gold-standard parse

Global Linear Models

- Three components:

  GEN is a function from a string to a set of candidates

  $\mathbf{\Phi}$ maps a candidate to a feature vector

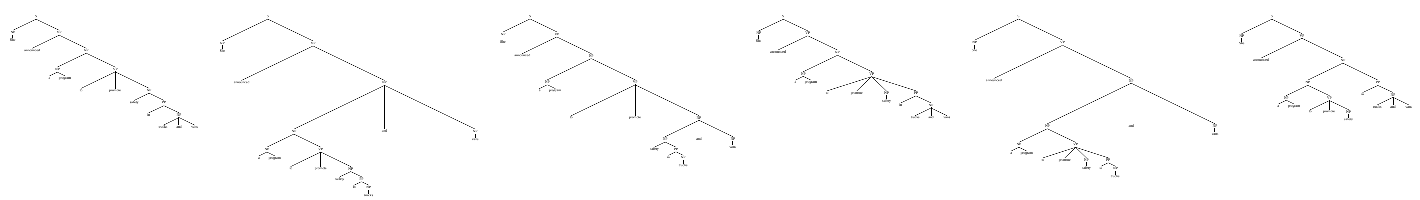
  $\mathbf{W}$ is a parameter vector

Component 1: GEN

- **GEN** enumerates a set of **candidates** for a sentence

She announced a program to promote safety in trucks and vans

$\Downarrow$ **GEN**

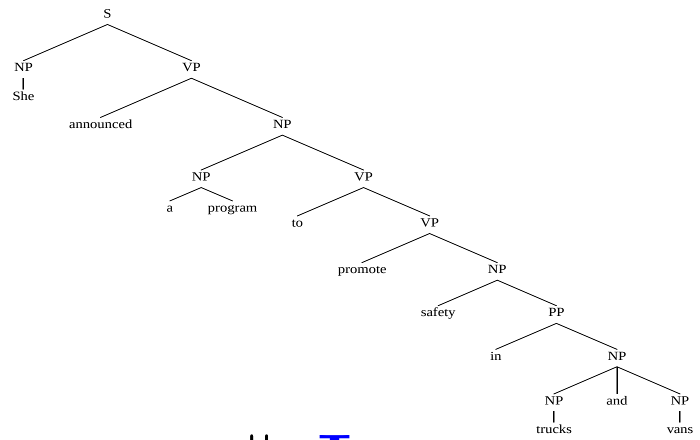


Component 2: $\mathbf{\Phi}$

- $\mathbf{\Phi}$ maps a candidate to a **feature vector** $\in \mathbb{R}^d$

- $\mathbf{\Phi}$ defines the **representation** of a candidate

---

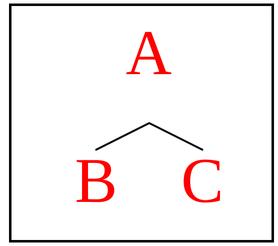


$$\Downarrow \mathbf{\Phi}$$

$$\langle 1, 0, 2, 0, 0, 15, 5 \rangle$$

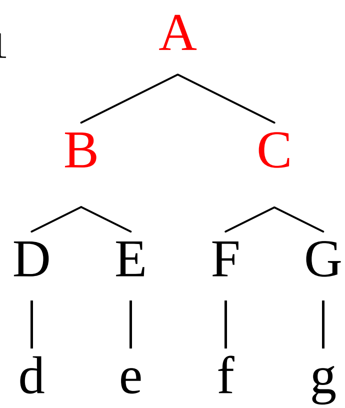
Features

- A "feature" is a function on a structure, e.g.,

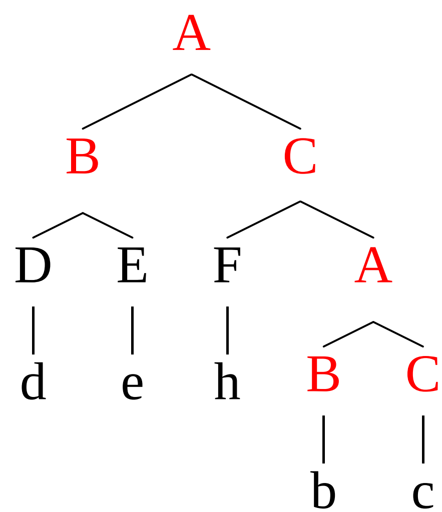
$h(x) =$ Number of times  is seen in x

---

T_1



T_2

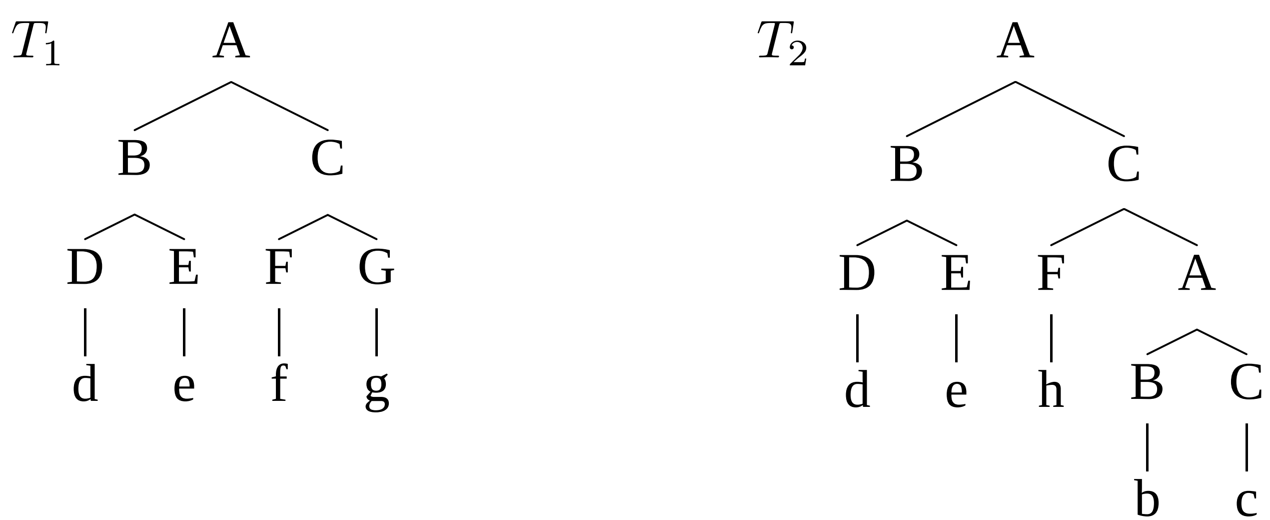


$$h(T_1) = 1 \qquad\qquad h(T_2) = 2$$

Feature Vectors

- A set of functions $h_1 \ldots h_d$ define a **feature vector**

$$\mathbf{\Phi}(x) = \langle h_1(x), h_2(x) \ldots h_d(x) \rangle$$

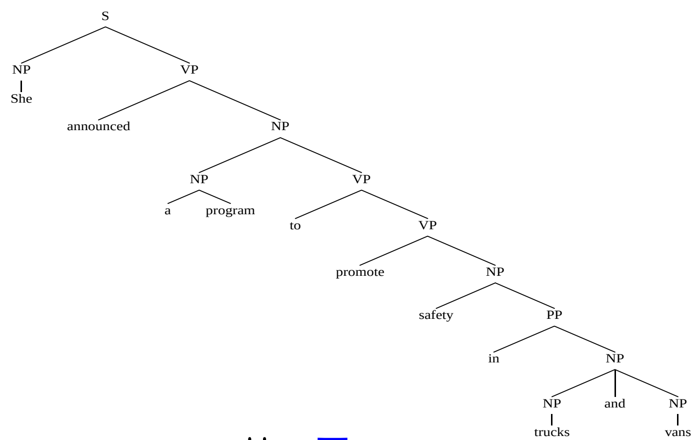


$$\mathbf{\Phi}(T_1) = \langle 1, 0, 0, 3 \rangle \qquad \mathbf{\Phi}(T_2) = \langle 2, 0, 1, 1 \rangle$$

Component 3: $\mathbf{W}$

- $\mathbf{W}$ is a **parameter vector** $\in \mathbb{R}^d$
- $\mathbf{\Phi}$ and $\mathbf{W}$ together map a candidate to a real-valued score

---



$$\Downarrow \mathbf{\Phi}$$

$$\langle 1, 0, 2, 0, 0, 15, 5 \rangle$$

$$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$$

$$\langle 1, 0, 2, 0, 0, 15, 5 \rangle \cdot \langle 1.9, -0.3, 0.2, 1.3, 0, 1.0, -2.3 \rangle = 5.8$$

Putting it all Together

- $\mathcal{X}$ is set of sentences, $\mathcal{Y}$ is set of possible outputs (e.g. trees)

- Need to learn a function $\mathbf{F} : \mathcal{X} \rightarrow \mathcal{Y}$

- $\mathbf{GEN}$, $\mathbf{\Phi}$, $\mathbf{W}$ define

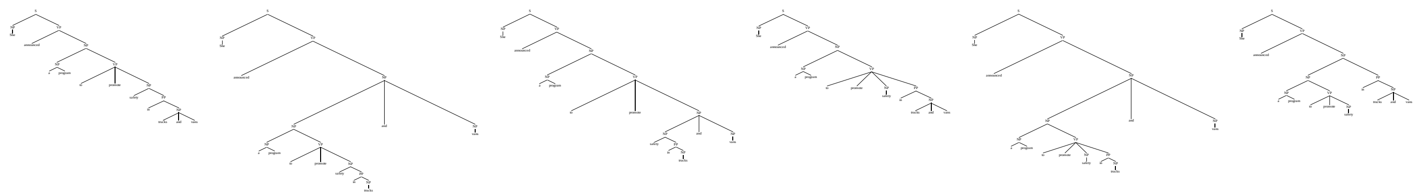
$$\mathbf{F}(x) = \arg\max_{y \in \mathbf{GEN}(x)} \mathbf{\Phi}(x, y) \cdot \mathbf{W}$$

Choose the highest scoring tree as the most plausible structure

- Given examples (x_i, y_i), how to set $\mathbf{W}$?

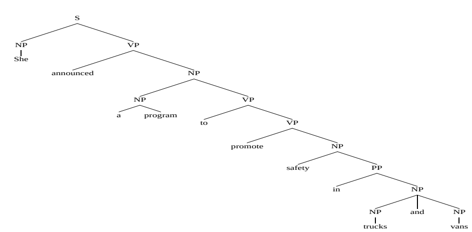
She announced a program to promote safety in trucks and vans

$\Downarrow$ **GEN**



$\Downarrow \mathbf{\Phi}$	$\Downarrow \mathbf{\Phi}$	$\Downarrow \mathbf{\Phi}$	$\Downarrow \mathbf{\Phi}$	$\Downarrow \mathbf{\Phi}$	$\Downarrow \mathbf{\Phi}$
$\langle 1, 1, 3, 5 \rangle$	$\langle 2, 0, 0, 5 \rangle$	$\langle 1, 0, 1, 5 \rangle$	$\langle 0, 0, 3, 0 \rangle$	$\langle 0, 1, 0, 5 \rangle$	$\langle 0, 0, 1, 5 \rangle$
$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$	$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$	$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$	$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$	$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$	$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$
13.6	12.6	12.1	3.3	9.4	11.1

$\Downarrow \arg\max$



Examples of Global Linear Models

- Parse Reranking, e.g., [Ratnaparkhi, Reynar and Roukos, 1994], [Johnson et. al, 1999], [Collins 2000], [Riezler et. al, 2004], [Shen, Sarkar and Joshi, 2003], [Charniak and Johnson, 2005]

- Conditional random fields for tagging problems [Lafferty, McCallum, and Pereira, 2001; Sha and Pereira, 2003]

- Speech recognition: estimating a discriminative n-gram model [Roark, Saraclar and Collins, 2004]

- Dependency parsing [McDonald, Pereira, Ribarov and Hajic, 2005]

- Reranking for machine translation [Shen and Joshi, 2005; Shen, Sarkar and Och, 2004]

- Alignments in MT [Taskar, Lacoste-Julien, and Klein, 2005]

Overview

- Margins, and the large margin solution
- An SVM algorithm
- Local feature vectors
  (what to do when **GEN** is large...)
- Justification for the algorithm
- Conclusions

Margins

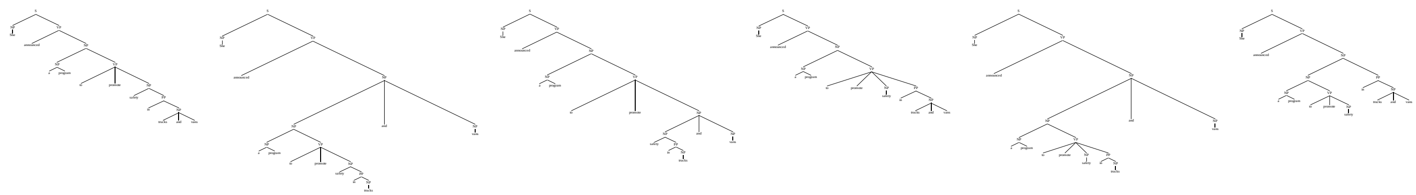
- Given parameter values $\mathbf{W}$, the *margin* on parse y for i'th training example is

$$M_{i,y} = \mathbf{\Phi}(x_i, y_i) \cdot \mathbf{W} - \mathbf{\Phi}(x_i, y) \cdot \mathbf{W}$$

This is **the difference in score between the correct parse, and parse** y

She announced a program to promote safety in trucks and vans

$\Downarrow$ **GEN**



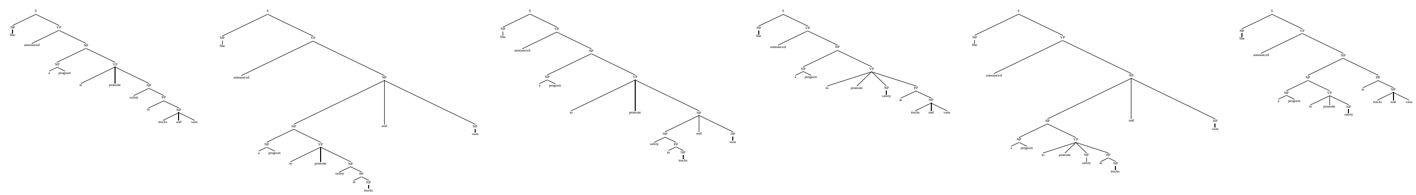
$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$	$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$	$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$	$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$	$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$	$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$
13.6	12.6	12.1	3.3	9.4	11.1

Margins (assuming first parse is correct):

—	1.0	1.5	10.3	4.2	2.5

She announced a program to promote safety in trucks and vans

$\Downarrow$ **GEN**



$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$	$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$	$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$	$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$	$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$	$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$
13.6	**14.8**	12.1	3.3	9.4	11.1

Margins (assuming first parse is correct):

—	**-1.2**	1.5	10.3	4.2	2.5

Support Vector Machines: The Large Margin Solution

Minimize

$$||\mathbf{W}||^2$$

under the constraints

$$\forall i, \forall y \neq y_i, \quad M_{i,y} \geq 1$$

(Note: a solution doesn't always exist)

$\|\mathbf{W}\|^2 = \sum_j \mathbf{W}_j^2$

Support Vector Machines: The Large Margin Solution

Minimize

$$||\mathbf{W}||^2$$

under the constraints

$$\forall i, \forall y \neq y_i, \quad M_{i,y} \geq 1$$

Statistical justifi cation:

- Assume there is a distribution $P(x, y)$ underlying training and test examples
- If $\frac{\|\mathbf{W}\|^2}{n}$ is small, with high probability $\mathbf{W}$ will have low error rate w.r.t. $P(x, y)$

Overview

- Margins, and the large margin solution
- An SVM algorithm
- Local feature vectors
  (what to do when **GEN** is large...)
- Justification for the algorithm
- Conclusions

Training an SVM: Dual Variables

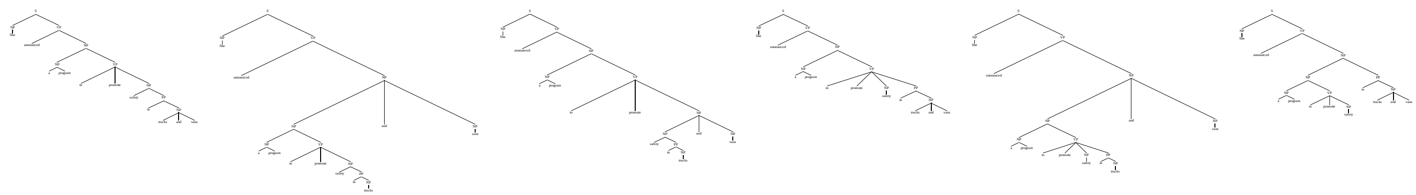
- For the perceptron, SVMs, and conditional random fields, the final parameter values can be expressed as:

$$\mathbf{W} = \sum_{i,y} \alpha_{i,y} \left[\mathbf{\Phi}(x_i, y_i) - \mathbf{\Phi}(x_i, y) \right]$$

where $\alpha_{i,y}$ are **dual variables**

She announced a program to promote safety in trucks and vans

$\Downarrow$ **GEN**



$$\Downarrow \quad \Downarrow \quad \Downarrow \quad \Downarrow \quad \Downarrow \quad \Downarrow$$

$$\mathbf{\Phi}_1 \quad \mathbf{\Phi}_2 \quad \mathbf{\Phi}_3 \quad \mathbf{\Phi}_4 \quad \mathbf{\Phi}_5 \quad \mathbf{\Phi}_6$$

Dual variables $\alpha_{i,y}$:

0.1 0.3 0.5 0.05 0.04 0.01

Assuming first parse is correct, contribution to **W** is

$$0.1[\mathbf{\Phi}_1 - \mathbf{\Phi}_1] + 0.3[\mathbf{\Phi}_1 - \mathbf{\Phi}_2] + 0.6[\mathbf{\Phi}_1 - \mathbf{\Phi}_3] + \ldots$$

Training an SVM

Inputs: Training set (x_i, y_i) for $i = 1 \ldots n$

Initialization: Set $\alpha_{i,y}$ to initial values,

Calculate $\mathbf{W} = \sum_{i,y} \alpha_{i,y} \left[\mathbf{\Phi}(x_i, y_i) - \mathbf{\Phi}(x_i, y) \right]$

Note: must have $\alpha_{i,y} > 0$, $\sum_y \alpha_{i,y} = 1$

Training an SVM: The Algorithm

(1) Calculate Margins:

$$\forall i, y, \quad M_{i,y} = \mathbf{\Phi}(x_i, y_i) \cdot \mathbf{W} - \mathbf{\Phi}(x_i, y) \cdot \mathbf{W}$$

(2) Update Dual Variables:

$$\forall i, y, \quad \alpha'_{i,y} \leftarrow \ldots$$

(More on this in a moment...)

(3) Update Parameters: $\mathbf{W} = \sum_{i,y} \alpha'_{i,y} \left[\mathbf{\Phi}(x_i, y_i) - \mathbf{\Phi}(x_i, y)\right]$

(4) If not converged, return to Step (1)

Updating the Dual Variables

$$\forall i, y, \quad \alpha'_{i,y} \leftarrow \frac{\alpha_{i,y} e^{\eta \nabla i,y}}{\sum_y \alpha_{i,y} e^{\eta \nabla i,y}}$$

where

$$\begin{array}{ll} \nabla_{i,y} = 0 & \text{for } y = y_i \\ \nabla_{i,y} = 1 - M_{i,y} & \text{for } y \neq y_i \end{array}$$

Intuition:

- if $M_{i,y} > 1$, $\alpha_{i,y}$ decreases
- if $M_{i,y} < 1$, $\alpha_{i,y}$ increases
- if $M_{i,y} = 1$, $\alpha_{i,y}$ stays the same
- The learning rate $\eta > 0$ controls the magnitude of the updates

$$\forall i, y, \quad \alpha'_{i,y} \leftarrow \frac{\alpha_{i,y} e^{\eta \nabla i,y}}{\sum_y \alpha_{i,y} e^{\eta \nabla i,y}}$$

where

$$\nabla i,y = 0 \quad \text{for } y = y_i$$
$$\nabla i,y = 1 - M_{i,y} \quad \text{for } y \neq y_i$$

	$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$	$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$	$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$	$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$
	13.6	13.0	14.8	3.3
Margins:	—	0.6	-1.2	10.3

$$\forall i, y, \quad \alpha'_{i,y} \leftarrow \frac{\alpha_{i,y} e^{\eta \nabla_{i,y}}}{\sum_y \alpha_{i,y} e^{\eta \nabla_{i,y}}}$$

where

$$\begin{array}{ll} \nabla_{i,y} = 0 & \text{for } y = y_i \\ \nabla_{i,y} = 1 - M_{i,y} & \text{for } y \neq y_i \end{array}$$

	$\Downarrow$ $\mathbf{\Phi} \cdot \mathbf{W}$	$\Downarrow$ $\mathbf{\Phi} \cdot \mathbf{W}$	$\Downarrow$ $\mathbf{\Phi} \cdot \mathbf{W}$	$\Downarrow$ $\mathbf{\Phi} \cdot \mathbf{W}$
	13.6	13.0	14.8	3.3
Margins:	—	0.6	-1.2	10.3
Values for $\nabla_{i,y}$:	0.0	0.4	2.2	-9.3
Values for $e^{\eta \nabla_{i,y}}$: (with $\eta = 1$)	1.0	1.49	9.03	0.00001

$$\forall i, y, \quad \alpha'_{i,y} \leftarrow \frac{\alpha_{i,y} e^{\eta \nabla_{i,y}}}{\sum_y \alpha_{i,y} e^{\eta \nabla_{i,y}}}$$

where $\nabla_{i,y} = 0$ for $y = y_i$
$\nabla_{i,y} = 1 - M_{i,y}$ for $y \neq y_i$

	$\Downarrow$ **Φ · W**	$\Downarrow$ **Φ · W**	$\Downarrow$ **Φ · W**	$\Downarrow$ **Φ · W**
	13.6	13.0	14.8	3.3
Margins:	—	0.6	-1.2	10.3
Values for $\nabla_{i,y}$:	0.0	0.4	2.2	-9.3
Values for $e^{\eta \nabla_{i,y}}$: (with $\eta = 1$)	1.0	1.49	9.03	0.00001
Old dual values $\alpha_{i,y}$:	0.1	0.3	0.5	0.1
New dual values $\alpha'_{i,y}$:	0.02	0.088	0.89	0.0

Training an SVM: The Algorithm

(1) Calculate Margins:

$$\forall i, y, \quad M_{i,y} = \mathbf{\Phi}(x_i, y_i) \cdot \mathbf{W} - \mathbf{\Phi}(x_i, y) \cdot \mathbf{W}$$

(2) Update Dual Variables:

$$\forall i, y, \quad \alpha'_{i,y} \leftarrow \frac{\alpha_{i,y} e^{\eta \nabla_{i,y}}}{\sum_y \alpha_{i,y} e^{\eta \nabla_{i,y}}}$$

where

$$\begin{array}{ll} \nabla_{i,y} = 0 & \text{for } y = y_i \\ \nabla_{i,y} = 1 - M_{i,y} & \text{for } y \neq y_i \end{array}$$

(3) Update Parameters: $\mathbf{W} = \sum_{i,y} \alpha'_{i,y} \left[\mathbf{\Phi}(x_i, y_i) - \mathbf{\Phi}(x_i, y)\right]$

(4) If not converged, return to Step (1)

Theory

- Algorithm converges to the minimum of

$$\sum_i \max_y \left(1 - M_{i,y}\right)_+ + \frac{1}{2}||\mathbf{W}||^2$$

where

$$(1 - M_{i,y})_+ = \begin{cases} (1 - M_{i,y}) & \text{if } (1 - M_{i,y}) > 0 \\ 0 & \text{otherwise} \end{cases}$$

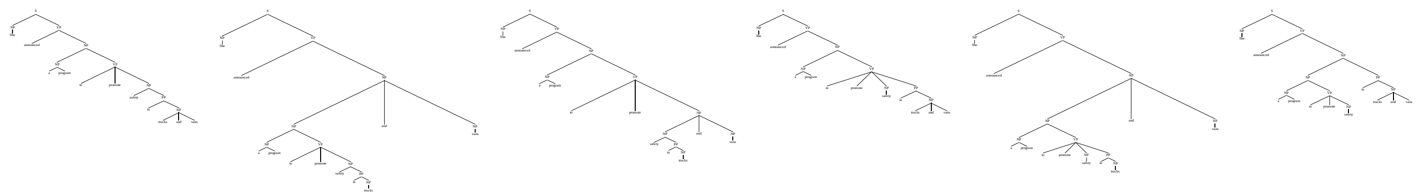
This is the **hinge loss**: penalizes values for $M_{i,y}$ that are < 1

Note, as before:

$$M_{i,y} = \mathbf{\Phi}(x_i, y_i) \cdot \mathbf{W} - \mathbf{\Phi}(x_i, y) \cdot \mathbf{W}$$

She announced a program to promote safety in trucks and vans

$\Downarrow$ **GEN**



$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$	$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$	$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$	$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$	$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$	$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$
13.6	12.6	12.1	3.3	9.4	11.1

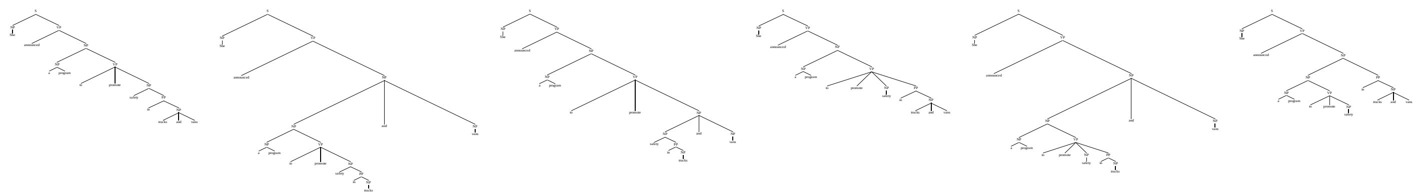
Margins (assuming first parse is correct):

—	1.0	1.5	10.3	4.2	2.5

In this case $\max_y (1 - M_{i,y})_+ = 0$

She announced a program to promote safety in trucks and vans

$\Downarrow$ **GEN**



$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$	$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$	$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$	$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$	$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$	$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$
13.6	12.2	12.1	3.3	9.4	11.1

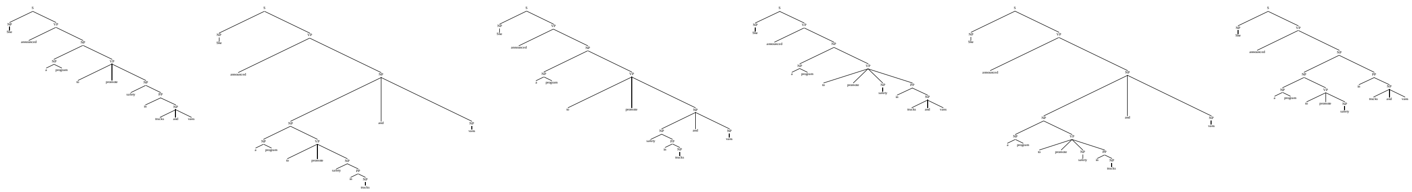
Margins (assuming first parse is correct):

—	1.4	1.5	10.3	4.2	2.5

In this case $\max_y (1 - M_{i,y})_+ = 0$

She announced a program to promote safety in trucks and vans

$\Downarrow$ **GEN**



$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$	$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$	$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$	$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$	$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$	$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$
13.6	13.0	12.1	3.3	9.4	11.1

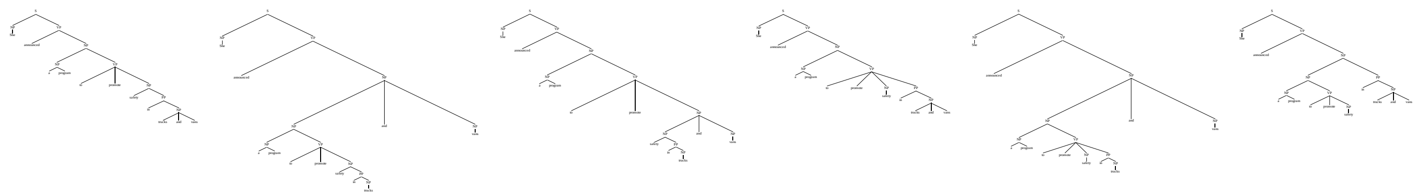
Margins (assuming first parse is correct):

—	**0.6**	1.5	10.3	4.2	2.5

In this case $\max_y (1 - M_{i,y})_+ = 0.4$

She announced a program to promote safety in trucks and vans

$\Downarrow$ **GEN**



$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$	$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$	$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$	$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$	$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$	$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$
13.6	13.0	14.8	3.3	9.4	11.1

Margins (assuming first parse is correct):

—	**0.6**	**-1.2**	10.3	4.2	2.5

In this case $\max_y (1 - M_{i,y})_+ = 2.2$

Theory

- Algorithm converges to the minimum of

$$\underbrace{\sum_i \max_y \left(1 - M_{i,y}\right)_+}_{\text{Penalizes margins less than } 1} \quad + \quad \underbrace{\frac{1}{2}||\mathbf{W}||^2}_{\text{Penalizes large parameter values}}$$

Theory

- Algorithm converges to the minimum of

$$\underbrace{\sum_i \max_y (1 - M_{i,y})_+}_{\text{Penalizes margins less than } 1} \quad + \quad \underbrace{\frac{1}{2}||\mathbf{W}||^2}_{\text{Penalizes large parameter values}}$$

- Note: it's trivial to modify the algorithm to minimize

$$C \sum_i \max_y (1 - M_{i,y})_+ + \frac{1}{2}||\mathbf{W}||^2$$

for some $C > 0$

- As $C \rightarrow \infty$ we get closer to the large margin solution

Optimizing Other Loss Functions

- Suppose for each incorrect parse tree, we have a "loss"

$$L_{i,y}$$

E.g., $L_{i,y}$ is number of parsing errors in y for sentence x_i

- New updates:

$$\forall i, y, \quad \alpha'_{i,y} = \frac{\alpha_{i,y} e^{\eta(L_{i,y} - M_{i,y})}}{\sum_y \alpha_{i,y} e^{\eta(L_{i,y} - M_{i,y})}}$$

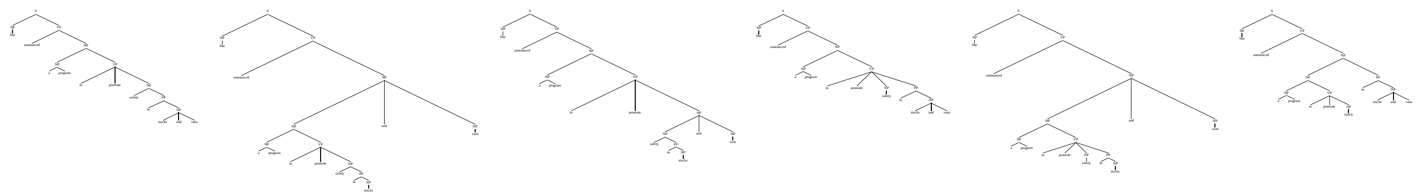
- Algorithm converges to the minimum of

$$\sum_i \max_y \left(L_{i,y} - M_{i,y}\right)_+ + \frac{1}{2}||\mathbf{W}||^2$$

(Loss function from [Taskar, Guestrin, Koller, 2003])

She announced a program to promote safety in trucks and vans

$\Downarrow$ **GEN**



$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$	$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$	$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$	$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$	$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$	$\Downarrow \mathbf{\Phi} \cdot \mathbf{W}$
13.6	13.0	14.8	3.3	9.4	11.1

Margins (assuming first parse is correct):

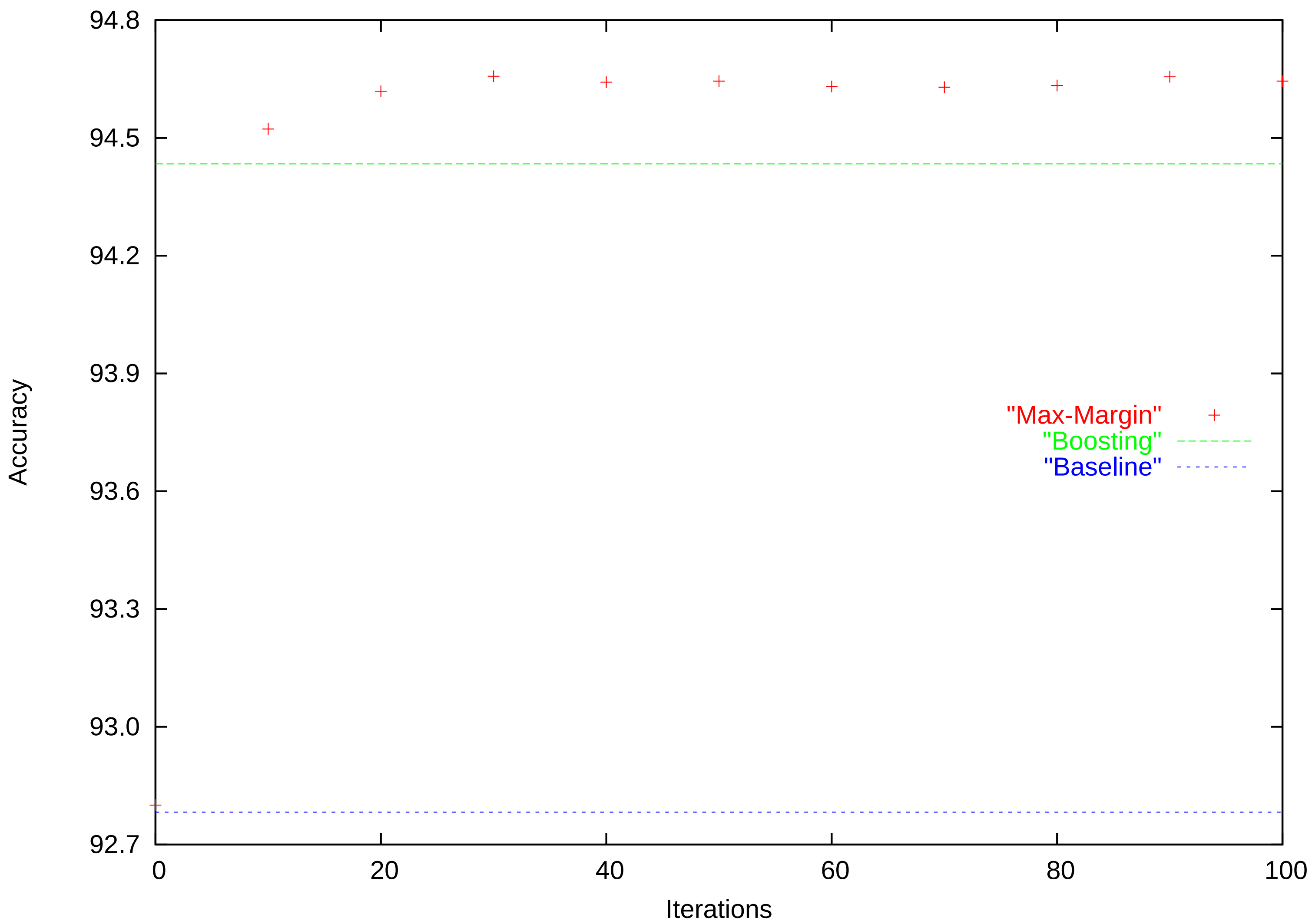
—	**0.6**	**-1.2**	10.3	4.2	2.5

Values for $L_{i,y}$:

0	5.0	1.0	2.3	1.7	2.5

In this case $\max_y (L_{i,y} - M_{i,y})_+ = 4.4$

Accuracy on a Parse Reranking Task



- $\approx 36,000$ training examples, 1 million trees total

Overview

- Margins, and the large margin solution
- An SVM algorithm
- Local feature vectors
  (what to do when **GEN** is large...)
- Justification for the algorithm
- Conclusions

Local Representations: What to do when $\mathbf{GEN}$ is large

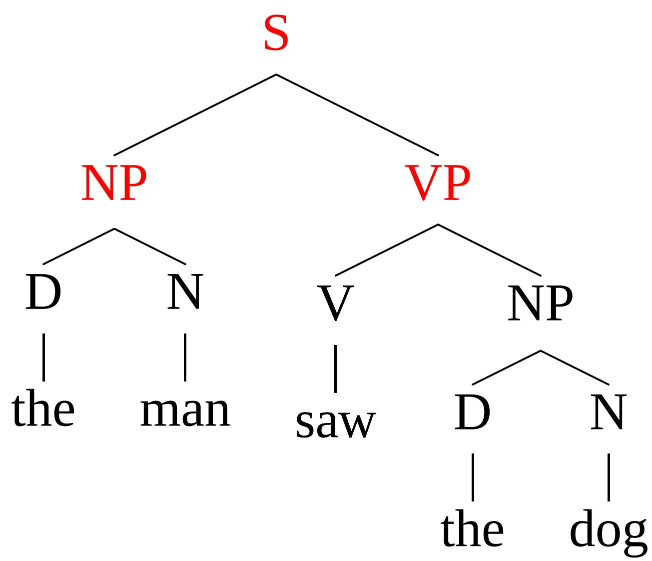
- Suppose $\mathbf{GEN}(x)$ is **all** parses for x under a context-free grammar

- We now have an **exponential** number of parses

- We have an exponential number of dual variables $\alpha_{i,y}$, margins $M_{i,y}$, feature vectors $\mathbf{\Phi}(x_i, y)$, error terms $L_{i,y}$ etc.

Local Representations

A tree:



Its context-free productions:

$$\langle \texttt{S} \rightarrow \texttt{NP VP}, \; 1, \; 2, \; 5\rangle$$
$$\langle \texttt{NP} \rightarrow \texttt{D N}, \; 1, \; 1, \; 2\rangle$$
$$\langle \texttt{VP} \rightarrow \texttt{V NP}, \; 3, \; 3, \; 5\rangle$$
$$\langle \texttt{NP} \rightarrow \texttt{D N}, \; 4, \; 4, \; 5\rangle$$

A part is a $\langle$`rule`, start-point, mid-point, end-point$\rangle$ tuple

Assumption 1: Local Feature-Vector Representations

- If x is a sentence, r is a part, then

$$\phi(x, r)$$

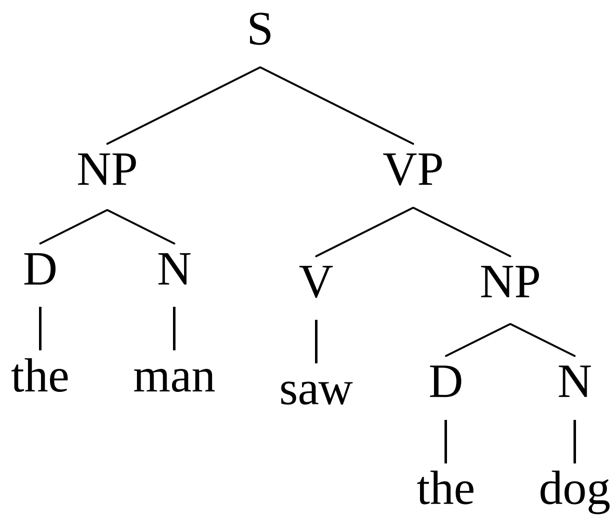
  is a *local feature-vector*

- For any parse tree y, we define

$$\mathbf{\Phi}(x, y) = \sum_{r \in y} \phi(x, r)$$

Local Feature Vectors

$(x, y) =$



$$
\begin{aligned}
\mathbf{\Phi}(x, y) = \\
& \phi(\text{the man saw the dog}, \langle \texttt{S} \rightarrow \texttt{NP}\ \texttt{VP}, 1, 2, 5 \rangle) \\
& + \phi(\text{the man saw the dog}, \langle \texttt{NP} \rightarrow \texttt{D}\ \texttt{N}, 1, 1, 2 \rangle) \\
& + \phi(\text{the man saw the dog}, \langle \texttt{VP} \rightarrow \texttt{V}\ \texttt{NP}, 3, 3, 5 \rangle) \\
& + \phi(\text{the man saw the dog}, \langle \texttt{NP} \rightarrow \texttt{D}\ \texttt{N}, 4, 4, 5 \rangle)
\end{aligned}
$$

Can find $\arg\max_y \mathbf{W} \cdot \mathbf{\Phi}(x, y)$ using CKY

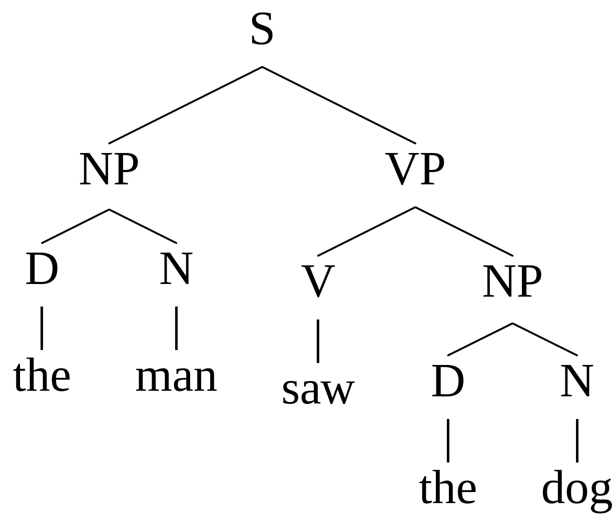
Assumption 2: Local Error Functions

- For any example i, assume $l_{i,r}$ is "cost" of proposing rule r in parse tree for x_i
- For example: $l_{i,r} = 1$ if rule r is not in the correct parse y_i, 0 otherwise
- Define

$$L_{i,y} = \sum_{r \in y} l_{i,r}$$

Local Error Functions

$(x_i, y) =$



$$
\begin{aligned}
L_{i,y} = \\
&\; l(i, \langle \mathtt{S} \;\rightarrow\; \mathtt{NP}\;\; \mathtt{VP}, 1, 2, 5\rangle) \\
&+ l(i, \langle \mathtt{NP} \;\rightarrow\; \mathtt{D}\;\; \mathtt{N}, 1, 1, 2\rangle) \\
&+ l(i, \langle \mathtt{VP} \;\rightarrow\; \mathtt{V}\;\; \mathtt{NP}, 3, 3, 5\rangle) \\
&+ l(i, \langle \mathtt{NP} \;\rightarrow\; \mathtt{D}\;\; \mathtt{N}, 4, 4, 5\rangle)
\end{aligned}
$$

The EG Algorithm under Local Assumptions

- The updates:

$$\forall i, y, \quad \alpha'_{i,y} = \frac{\alpha_{i,y} e^{\eta(L_{i,y} - M_{i,y})}}{\sum_y \alpha_{i,y} e^{\eta(L_{i,y} - M_{i,y})}}$$

- But now, we have

$$L_{i,y} - M_{i,y} = \sum_{r \in y} (l_{i,r} + \mathbf{W} \cdot \phi_{i,r}) - \mathbf{W} \cdot \mathbf{\Phi}(x_i, y_i)$$

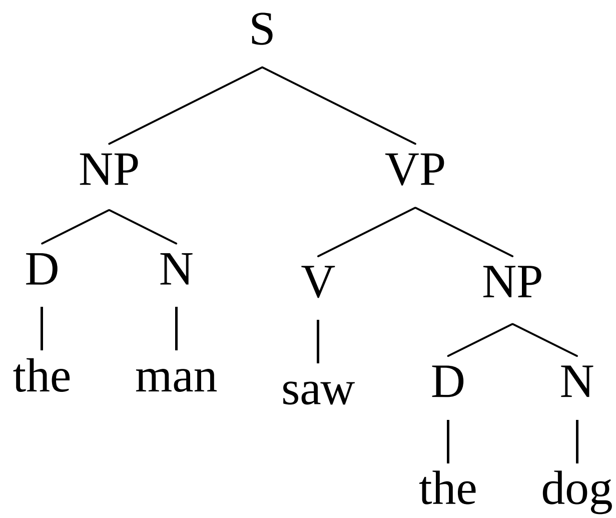
- We can represent $\alpha_{i,y}$ variables *compactly*:

$$\alpha_{i,y} = \frac{e^{\sum_{r \in y} \theta_{i,r}}}{\sum_y e^{\sum_{r \in y} \theta_{i,r}}}$$

- The updates are implemented as $\theta'_{i,r} \leftarrow \theta_{i,r} + \eta(l_{i,r} + \mathbf{W} \cdot \phi_{i,r})$

Local Dual Variables

$(x_i, y) =$



$$\alpha_{i,y} = \frac{e^{S_{i,y}}}{Z}$$

$$
\begin{aligned}
S_{i,y} = &\\
& \theta(i, \langle \mathtt{S} \rightarrow \mathtt{NP}\ \mathtt{VP}, 1, 2, 5\rangle) \\
+ & \theta(i, \langle \mathtt{NP} \rightarrow \mathtt{D}\ \mathtt{N}, 1, 1, 2\rangle) \\
+ & \theta(i, \langle \mathtt{VP} \rightarrow \mathtt{V}\ \mathtt{NP}, 3, 3, 5\rangle) \\
+ & \theta(i, \langle \mathtt{NP} \rightarrow \mathtt{D}\ \mathtt{N}, 4, 4, 5\rangle)
\end{aligned}
$$

There are an exponential number of $\alpha_{i,y}$ variables,
but there are a polynomial number of $\theta(i, rule)$ variables

Overview

- Margins, and the large margin solution
- An SVM algorithm
- Local feature vectors
  (what to do when **GEN** is large...)
- Justification for the algorithm
- Conclusions

How did we Derive the Algorithm?

- We want to find the $\mathbf{W}$ that minimizes:

$$\underbrace{\sum_i \max_y \left(1 - M_{i,y}\right)_+}_{\text{Penalizes margins less than } 1} \quad + \quad \underbrace{\frac{1}{2}||\mathbf{W}||^2}_{\text{Penalizes large parameter values}}$$

The Dual Optimization Problem for the SVM

Choose $\alpha_{i,y}$ values to maximize

$$Q(\bar{\alpha}) = \sum_{i,y \neq y_i} \alpha_{i,y} - \frac{1}{2} \|\mathbf{W}\|^2$$

where

$$\mathbf{W} = \sum_{i,y} \alpha_{i,y} \left[\mathbf{\Phi}(x_i, y_i) - \mathbf{\Phi}(x_i, y) \right]$$

Constraints:

$$\forall i, \forall y, \quad \alpha_{i,y} \geq 0$$

$$\forall i, \quad \sum_{y} \alpha_{i,y} = 1$$

The Dual Optimization Problem for the SVM

- We want to maximize $Q(\bar{\alpha})$
- It can be shown that

$$\frac{dQ(\bar{\alpha})}{d\alpha_{i,y}} = \nabla_{i,y} \qquad \begin{array}{ll} \nabla_{i,y} = 0 & \text{for } y = y_i \\ \nabla_{i,y} = 1 - M_{i,y} & \text{for } y \neq y_i \end{array}$$

- Gradient ascent:

$$\alpha_{i,y} \leftarrow \alpha_{i,y} + \eta \nabla_{i,y}$$

- Exponentiated Gradient:

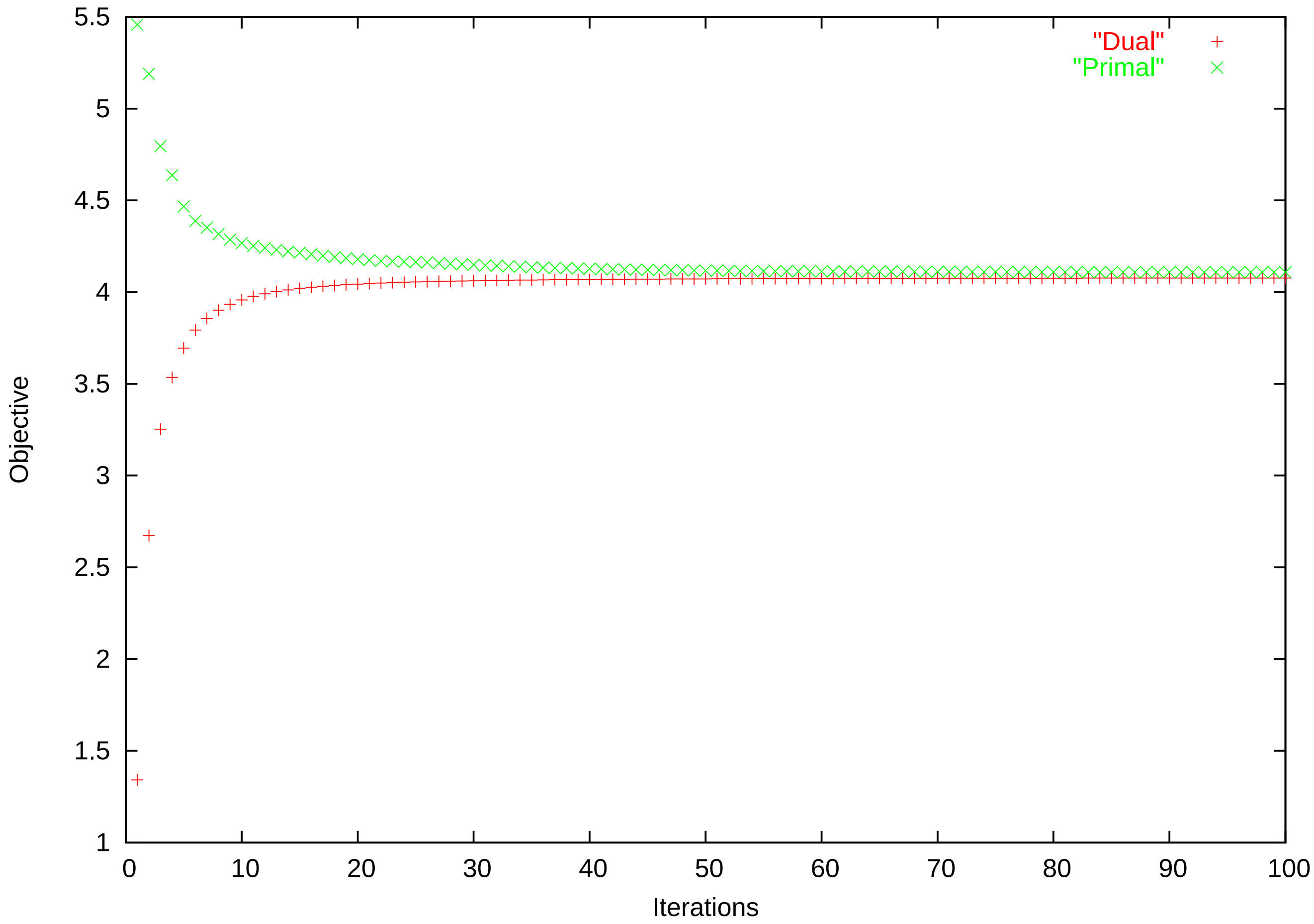
$$\alpha_{i,y} \leftarrow \frac{\alpha_{i,y} e^{\eta \nabla_{i,y}}}{\sum_y \alpha_{i,y} e^{\eta \nabla_{i,y}}}$$

(Motivation: $\alpha_{i,y}$'s remain positive, $\sum_y \alpha_{i,y} = 1$)

- The *exponentiated gradient* method is an example of *multiplicative updates*: central to AdaBoost (Freund and Schapire), online learning algorithms such as Winnow (Warmuth), several applications to combinatorial optimization, linear programming, problems in game theory, etc. etc. (survey article by Arora, Hazan and Kale)

- Analysis of the algorithm builds on work by Warmuth and collaborators in online learning

Convergence on a Parse Reranking Task



- $\approx 36,000$ training examples, 1 million trees total
- $\approx 500,000$ sparse features

Overview

- Margins, and the large margin solution
- An SVM algorithm
- Local feature vectors
  (what to do when **GEN** is large...)
- Justification for the algorithm
- Conclusions

Contributions

- A simple algorithm for finding the SVM solution
- Relies on close connections between margins, dual variables, dual problem for the SVM
- Experiments show good performance on reranking tasks
- The algorithm has a convenient *compact* form for context-free grammars with local feature–vectors