

# CRF model

$$P(y | x) \propto \exp(\theta^T f(x, y))$$

Params  $\uparrow$  Feature Vector  $\rightarrow$   
Tagging  $\downarrow$  Text  $\downarrow$

- The idea:

- Global scoring function for output  $y$

$$G(y) = \theta^T f(x, y) = \sum_{j=1}^J \theta_j f_j(x, y)$$

- Global features from sum of local features

- Scoring function decomposes into local parts  
(therefore can use Viterbi)

$\rightarrow$  To predict  $y$  from  $x, \theta$

- Advantages vs. HMM

- Can add arbitrary observation features!

- Character ngrams
- Capitalization
- Word presence in dictionary...

- Discriminative learning: we use the **perceptron**

learn  $\theta$  from gold  $(x, y)$

$$\text{Global } f(x, y) = \sum_t f(y_{t-1}, y_t, x_t)$$

$$\text{score } G(y) = \theta' f(x, y)$$

$$= \theta^T \left( \sum_t f(y_{t-1}, y_t, x_t) \right)$$

$$= \sum_t \underbrace{\theta^T f(y_{t-1}, y_t, x_t)}_{g(y_{t-1}, y_t)}$$

$$= \sum_t A(y_{t-1}, y_t) + B_t(y_t)$$

↓

$\theta_{y_{t-1}, y_t}$

↓

$\theta_{\text{obs}: y_t, x_t}$

$$A(\text{Noun}, \text{Verb}) = \theta_{\text{Noun}, \text{Verb}}$$

Transition  
Feature

$$f_3(y_{t-1}, y_t, x_t) = \begin{cases} 1 & \text{if } y_{t-1} = v \text{ and} \\ & y_t = v \\ 0 & \text{otherwise} \end{cases}$$

obs.  
Feat

$$f_{1500}(y_{t-1}, y_t, x_t) = \begin{cases} 1 & \text{if } y_t = v \text{ and} \\ & x_t = get \\ 0 & \text{otherwise} \end{cases}$$

# CRF is from local features

$x =$	finna	get	good
$y =$	$s_0 = [START]$	V	A

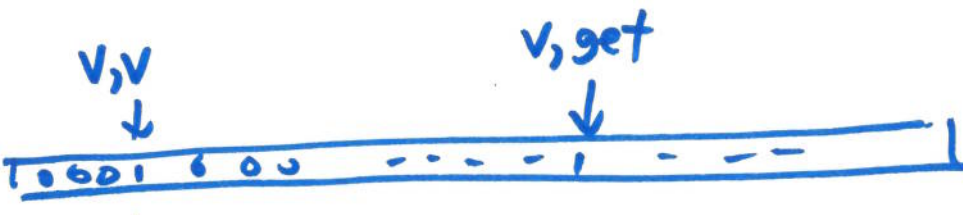
$$f(y_{t-1}, y_t, x_t, \lambda_t)$$

local  $f(y_{t-1}, y_t, x_t) =$  

$$f(s, v, finna)$$

binary  
features

$f(v, v, get) =$  

global  $f(x, y) =$  

$$G(y) = \theta^T f(x, y)$$

Count of  $v \Rightarrow v$  transitions in  $y$

$$= \sum_t f(y_{t-1}, y_t, \lambda_t)$$

gold  $y =$       finna      get      good  
                          V            V            A

Transition features

Observation features

$\theta$

Transition features							Observation features																
-0.6	-1.0	1.1	0.5	0.0	0.8	0.5	-1.3	-1.6	0.0	0.6	0.0	-0.2	-0.2	0.8	-1.0	0.1	-1.9	1.1	1.2	-0.1	-1.0	-0.1	-0.1

$f(x, y)$

1	0	2	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	3	0	0	0	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

$$f_{trans:V,A}(x, y) = \sum_{t=2}^N 1\{y_{t-1} = V, y_t = A\}$$

$$f_{obs:V,finna}(x, y) = \sum_{t=1}^N 1\{y_t = V, x_t = finna\}$$

$$\text{Goodness}(y) = \theta^T f(x, y)$$

# Structured/multiclass Perceptron

- Goal: learn model parameters  $\theta$  from labeled data
- For  $\sim 30$  iterations
  - For each  $(x, y)$  in dataset

*Handwritten notes:*  
 $y^* = N$     *finra*    *get*    *good*  
                   *N*            *✓*            *A*

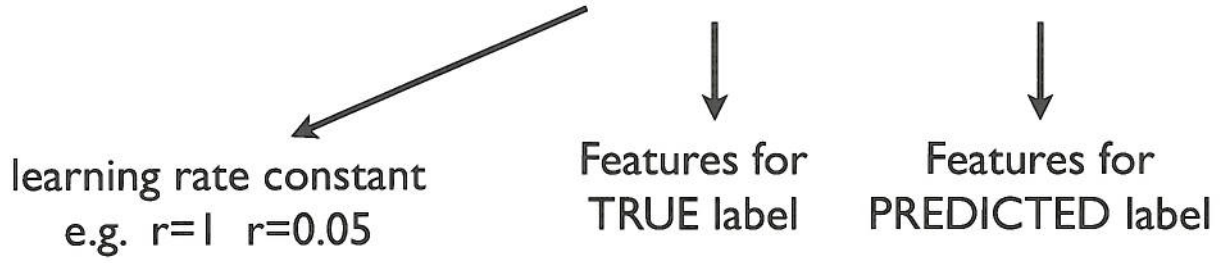
- PREDICT

$$y^* = \arg \max_{y'} \frac{\theta^T f(x, y')}{G(y)}$$

- IF  $y=y^*$ , do nothing
- ELSE update weights

$$\theta := \theta + r [f(x, y) - f(x, y^*)]$$

*Handwritten note:* *gold* with an arrow pointing to  $f(x, y)$  in the equation above.



# Update rule

learning rate  
e.g.  $r=1$

Features for  
TRUE label

Features for  
PREDICTED label

$$\theta := \theta + r[f(x, y) - f(x, y^*)]$$

For each feature  $j$  in true  $y$  but not predicted  $y^*$ :

$$\theta_j := \theta_j + (r)f_j(x, y)$$

For each feature  $j$  not in true  $y$ , but in predicted  $y^*$ :

$$\theta_j := \theta_j - (r)f_j(x, y)$$

finna      get      good  
 gold  $y =$     V      V      A  
 pred  $y^* =$     N      V      A  
                   error

$f(x, y)$   
 V, V: 1  
 V, A: 1  
 V, finna: 1  
 V, get: 1  
 A, good: 1

$f(x, y^*)$   
 N, V: 1  
 V, A: 1  
 N, finna: 1  
 V, get: 1  
 A, good: 1

$f(x, y) - f(x, y^*)$   
 V, V: +1  
 N, V: -1  
 N, finna: -1  
 V, finna: +1

$$\theta_j := \theta_j^{(old)} + r(f_j(x, y) - f_j(x, y^*))$$

Perceptron update rule:

$$\theta := \theta + r[f(x, y) - f(x, y^*)]$$