

CSE 842

Natural Language Processing

Lecture 20: Expectation Maximization Algorithms in NLP

4/13/2009

CSE842, Spring 2009, MSU

1

Topics

- Maximum Likelihood Estimation
- Models with Hidden Variables
- EM Algorithms
- Some applications of EM Algorithms
 - Forward-backward algorithm related to HMM (covered earlier)
 - Inside-outside algorithm with PCFG
 - Estimation of weights in interpolation for language modeling
 - Parameter estimation for machine translation (to be covered later)

4/13/2009

CSE842, Spring 2009, MSU

2

A First Example: Coin Tossing

- A random variable X with two possible outcomes: $\{H, T\}$
- Parameter θ : the probability of coin coming up heads
- x_1, x_2, \dots, x_n is a sequence of observations:
HHTHTHTTHH
- What is θ ?

4/13/2009

CSE842, Spring 2009, MSU

3

Maximum Likelihood Estimation

- We have data points x_1, x_2, \dots, x_n , each data point represents an outcome of a random variable X .
- We have a parameter vector Θ and a parameter space Ω , $\Theta \in \Omega$.
- All data points are drawn independently and identically distributed from a distribution $P(x|\Theta^*)$ for some $\Theta^* \in \Omega$.
- The likelihood of data is:

$$Likelihood(\Theta) = P(x_1, x_2, \dots, x_n | \Theta) = \prod_{i=1}^n P(x_i | \Theta)$$

- The log-likelihood is:

$$L(\Theta) = \log Likelihood(\Theta) = \sum_{i=1}^n \log P(x_i | \Theta)$$

- The goal is to search Θ^* that maximize the log-likelihood of data.

$$\Theta_{ML} = \arg \max_{\Theta \in \Omega} L(\Theta) = \arg \max_{\Theta \in \Omega} \sum_{i=1}^n \log P(x_i | \Theta)$$

4/13/2009

CSE842, Spring 2009, MSU

4

Back to the Coin Example

- Suppose x_1, x_2, \dots, x_n has $\text{Count}(H)$ heads, and $(n - \text{Count}(H))$ tails

$$\begin{aligned} L(\Theta) &= \log(\Theta^{\text{Count}(H)} \times (1-\Theta)^{n-\text{Count}(H)}) \\ &= \text{Count}(H) \log \Theta + (n - \text{Count}(H)) \log(1-\Theta) \end{aligned}$$

- In this example, Θ is only a single parameter (not a vector). Identify Θ that maximizes the log likelihood data is:

$$\frac{\partial L(\Theta)}{\partial \Theta} = \text{Count}(H) \frac{1}{\Theta} - (n - \text{Count}(H)) \frac{1}{1-\Theta} = 0$$

- We now have:

$$\Theta_{ML} = \frac{\text{Count}(H)}{n}$$

Models with Hidden Variables

- Suppose we use two random variables X and Y describing our data
- Fully observed data: we can observe all (x_i, y_i) pairs, then

$$L(\Theta) = \sum_i \log P(x_i, y_i | \Theta)$$

- Partially observed data: only observe x_i , but not y_i , then

$$\begin{aligned} L(\Theta) &= \sum_i \log P(x_i | \Theta) \\ &= \sum_i \log \sum_{y \in Y} P(x_i, y | \Theta) \end{aligned}$$

- The EM (Expectation Maximization) algorithm is a method for finding

$$\Theta_{ML} = \arg \max_{\Theta} \sum_i \log \sum_{y \in Y} P(x_i, y | \Theta)$$

The Three Coins Example

- We have three coins:
 - Coin 0 has probability λ of heads;
 - Coin 1 has probability p_1 of heads;
 - Coin 2 has probability p_2 of heads

- For each trial we do the following:

First, toss Coin 0

If Coin 0 turns up heads, toss coin 1 three times

If Coin 0 turns up tails, toss coin 2 three times

I don't tell you whether Coin 0 came up heads or tails,

or whether Coin 1 or 2 was tossed three times,

but I do tell you how many heads/tails are seen at each trial

- You see the following sequence:

<HHH>, <TTT>, <HHH>, <TTT>, <HHH>

How do you estimate the values for λ , p_1 and p_2 ?

The Three Coins Example

- In this example,

$$Y = \{H, T\}$$

$$X = \{HHH, TTT, HTT, THH, HHT, TTH, HTH, THT\}$$

$$\Theta = \{\lambda, p_1, p_2\}$$

- And $P(x, y | \Theta) = P(y | \Theta)P(x | y, \Theta)$

where

$$P(y | \Theta) = \begin{cases} \lambda & \text{if } y = H \\ 1 - \lambda & \text{if } y = T \end{cases}$$

and

$$P(x | y, \Theta) = \begin{cases} p_1^h (1 - p_1)^{t-h} & \text{if } y = H \\ p_2^h (1 - p_2)^{t-h} & \text{if } y = T \end{cases}$$

where h = number of heads in x , t = number of tails in x

The Three Coins Example

- Various probabilities can be calculated, for example

$$P(x = THT, y = H | \Theta) = \lambda p_1 (1 - p_1)^2$$

The Three Coins Example

- Various probabilities can be calculated, for example

$$P(x = THT, y = H | \Theta) = \lambda p_1 (1 - p_1)^2$$

$$P(x = THT, y = T | \Theta) = (1 - \lambda) p_2 (1 - p_2)^2$$

The Three Coins Example

- Various probabilities can be calculated, for example

$$P(x = THT, y = H | \Theta) = \lambda p_1 (1 - p_1)^2$$

$$P(x = THT, y = T | \Theta) = (1 - \lambda) p_2 (1 - p_2)^2$$

$$\begin{aligned} P(x = THT | \Theta) &= P(x = THT, y = H | \Theta) + P(x = THT, y = T | \Theta) \\ &= \lambda p_1 (1 - p_1)^2 + (1 - \lambda) p_2 (1 - p_2)^2 \end{aligned}$$

The Three Coins Example

- Various probabilities can be calculated, for example

$$P(x = THT, y = H | \Theta) = \lambda p_1 (1 - p_1)^2$$

$$P(x = THT, y = T | \Theta) = (1 - \lambda) p_2 (1 - p_2)^2$$

$$\begin{aligned} P(x = THT | \Theta) &= P(x = THT, y = H | \Theta) + P(x = THT, y = T | \Theta) \\ &= \lambda p_1 (1 - p_1)^2 + (1 - \lambda) p_2 (1 - p_2)^2 \end{aligned}$$

$$\begin{aligned} P(y = H | x = THT, \Theta) &= \frac{P(x = THT, y = H | \Theta)}{P(x = THT | \Theta)} \\ &= \frac{\lambda p_1 (1 - p_1)^2}{\lambda p_1 (1 - p_1)^2 + (1 - \lambda) p_2 (1 - p_2)^2} \end{aligned}$$

The Three Coins Example

- Fully observed data may look like:
(H, <HHH>), (T, <TTT>), (H, <HHH>), (<T, <TTT>), (H, <HHH>)

Then easy to get maximum likelihood estimates as follows:

$$\lambda = \frac{3}{5}, p_1 = \frac{9}{9}, p_2 = \frac{0}{6}$$

- Partially observed data:
<HHH>, <TTT>, <HHH>, <TTT>, <HHH>
How to find the maximum likelihood parameters?

The Three Coins Example

Partially observed data:

<HHH>, <TTT>, <HHH>, <TTT>, <HHH>

If we know λ , p_1 , p_2 , then we can calculate the following:

if $\lambda = 0.3, p_1 = 0.3, p_2 = 0.6$:

$$P(y = H | x = \langle HHH \rangle) = \frac{P(\langle HHH \rangle, H)}{P(\langle HHH \rangle, H) + P(\langle HHH \rangle, T)}$$

$$= \frac{\lambda p_1^3}{\lambda p_1^3 + (1 - \lambda) p_2^3} = 0.0508$$

$$P(y = H | x = \langle TTT \rangle) = \frac{P(\langle TTT \rangle, H)}{P(\langle TTT \rangle, H) + P(\langle TTT \rangle, T)}$$

$$= \frac{\lambda(1 - p_1)^3}{\lambda(1 - p_1)^3 + (1 - \lambda)(1 - p_2)^3} = 0.6967$$

The Three Coins Example

After filling in hidden variables for each example, partially observed data may look like the following:

<HHH>, H)	$P(y = H HHH) = 0.0508$
<HHH>, T)	$P(y = T HHH) = 0.9492$
<TTT>, H)	$P(y = H TTT) = 0.6967$
<TTT>, T)	$P(y = T TTT) = 0.3033$
<HHH>, H)	$P(y = H HHH) = 0.0508$
<HHH>, T)	$P(y = T HHH) = 0.9492$
<TTT>, H)	$P(y = H TTT) = 0.6967$
<TTT>, T)	$P(y = T TTT) = 0.3033$
<HHH>, H)	$P(y = H HHH) = 0.0508$
<HHH>, T)	$P(y = T HHH) = 0.9492$

The Three Coins Example

After filling in hidden variables for each example, partially observed data may look like the following:

<HHH>, H)	$P(y = H HHH) = 0.0508$	<div style="border: 1px solid blue; padding: 5px;"> <p>Expected # of heads for coin 0: $3 * 0.0508 + 2 * 0.6967 = 1.5458$</p> </div>
<HHH>, T)	$P(y = T HHH) = 0.9492$	
<TTT>, H)	$P(y = H TTT) = 0.6967$	
<TTT>, T)	$P(y = T TTT) = 0.3033$	
<HHH>, H)	$P(y = H HHH) = 0.0508$	
<HHH>, T)	$P(y = T HHH) = 0.9492$	
<TTT>, H)	$P(y = H TTT) = 0.6967$	
<TTT>, T)	$P(y = T TTT) = 0.3033$	
<HHH>, H)	$P(y = H HHH) = 0.0508$	
<HHH>, T)	$P(y = T HHH) = 0.9492$	

The Three Coins Example

After filling in hidden variables for each example, partially observed data may look like the following:

(<HHH>, H)	$P(y = H HHH) = 0.0508$	Expected # of heads for coin 1: $0.0508 * 3 * 3 = 0.4572$
(<HHH>, T)	$P(y = T HHH) = 0.9492$	
(<TTT>, H)	$P(y = H TTT) = 0.6967$	
(<TTT>, T)	$P(y = T TTT) = 0.3033$	Expected # of flips for coin 1: $0.0508 * 3 * 3 + 0.6967 * 3 * 2 = 4.6374$
(<HHH>, H)	$P(y = H HHH) = 0.0508$	
(<HHH>, T)	$P(y = T HHH) = 0.9492$	
(<TTT>, H)	$P(y = H TTT) = 0.6967$	
(<TTT>, T)	$P(y = T TTT) = 0.3033$	
(<HHH>, H)	$P(y = H HHH) = 0.0508$	
(<HHH>, T)	$P(y = T HHH) = 0.9492$	

The Three Coins Example

After filling in hidden variables for each example, partially observed data may look like the following:

(<HHH>, H)	$P(y = H HHH) = 0.0508$	Expected # of heads for coin 2: $0.9492 * 3 * 3 = 8.5428$
(<HHH>, T)	$P(y = T HHH) = 0.9492$	
(<TTT>, H)	$P(y = H TTT) = 0.6967$	
(<TTT>, T)	$P(y = T TTT) = 0.3033$	Expected # of flips for coin 1: $0.9492 * 3 * 3 + 0.3033 * 3 * 2 = 10.3626$
(<HHH>, H)	$P(y = H HHH) = 0.0508$	
(<HHH>, T)	$P(y = T HHH) = 0.9492$	
(<TTT>, H)	$P(y = H TTT) = 0.6967$	
(<TTT>, T)	$P(y = T TTT) = 0.3033$	
(<HHH>, H)	$P(y = H HHH) = 0.0508$	
(<HHH>, T)	$P(y = T HHH) = 0.9492$	

The Three Coins Example

Estimate new parameters

$$\lambda = \frac{\text{expected\# of heads by Coin 0}}{\text{total \# of flips of Coin 0}} = \frac{1.5458}{5} = 0.3092$$

$$p_1 = \frac{\text{expected \# of heads by Coin 1}}{\text{expected \# of flips by Coin 1}} = \frac{0.4572}{4.6374} = 0.0986$$

$$p_2 = \frac{\text{expected \# of heads by Coin 2}}{\text{expected \# of flips by Coin 2}} = \frac{8.5428}{10.3626} = 0.8244$$

The Three Coins Example

- Iterations:

- Use λ , p_1 , p_2 learned from the previous iteration to estimate $P(y|x)$
- Re-estimate λ , p_1 , p_2 using maximum likelihood estimation based on observed data and predicted $P(y|x)$.

Iteration	λ	p_1	p_2
0	0.3000	0.3000	0.6000
1	0.3092	0.0986	0.8244
2	0.3940	0.0012	0.9893
3	0.4000	0.0000	1.0000

- Depending on the initial values of , the algorithm may converge to a local maximum instead of the global maximum.

The EM Algorithm

- Θ^t is the parameter vector at the t 'th iteration
- Iterative procedure is defined as
$$\Theta^t = \arg \max_{\Theta} Q(\Theta, \Theta^{t-1})$$
where
$$Q(\Theta, \Theta^{t-1}) = \sum_i \sum_{y \in Y} P(y | x_i, \Theta^{t-1}) \log P(x_i, y | \Theta)$$
- Two steps:
 - E-step: fixed model (Θ^t), estimate posterior of the hidden variable for each data point.
 - M-step: re-estimate the parameters (i.e., the model) that maximize the data with hidden variable.
- EM is guaranteed to converge to a local maximum, or saddle-point, of the likelihood function.

Topics

- Maximum Likelihood Estimation
- Models with Hidden Variables
- EM Algorithms
- Some applications of EM Algorithms
 - Forward-backward algorithm related to HMM (covered earlier)
 - **Inside-outside algorithm with PCFG**
 - **Estimation of weights in interpolation for language modeling**
 - Parameter estimation for machine translation (to be covered later)

Linear Interpolation

$$P_{LI}(w_i | w_{i-2}, w_{i-1}) = \lambda_1 \times P_{ML}(w_i | w_{i-2}, w_{i-1}) \\ + \lambda_2 \times P_{ML}(w_i | w_{i-1}) \\ + \lambda_3 \times P_{ML}(w_i)$$

Where $\lambda_1 + \lambda_2 + \lambda_3 = 1$, and $\lambda_i \geq 0$ for all i .

How to Set the Lambdas?

- Use a **held-out, or development**, corpus
- Choose lambdas which maximize the probability of some held-out data
 - i.e. fix the N -gram probabilities, search for lambda values that when plugged into previous equation, give largest probability for held-out set
 - use EM to do this search

How to Set the Lambdas?

- Find λ_1 , λ_2 , and λ_3 to maximize the probability of the held-out data.

$$L(\lambda_1, \lambda_2, \lambda_3) = \sum_{i=1}^N \log \tilde{P}(w_i | w_{i-2}, w_{i-1})$$

or equivalently,

$$L(\lambda_1, \lambda_2, \lambda_3) = \sum_{w_i, w_{i-1}, w_{i-2} \in V} \text{Count}(w_{i-2}, w_{i-1}, w_i) \log \tilde{P}(w_i | w_{i-2}, w_{i-1})$$

where $\text{Count}(w_{i-2}, w_{i-1}, w_i)$ is the number of times (w_{i-2}, w_{i-1}, w_i) occurs in the held-out data

such that: $\lambda_1 + \lambda_2 + \lambda_3 = 1$ and $\lambda_i \geq 0$, where

$$\begin{aligned} \tilde{P}(w_i | w_{i-2}, w_{i-1}) &= \lambda_1 \times P_{ML}(w_i | w_{i-2}, w_{i-1}) + \\ &= \lambda_2 \times P_{ML}(w_i | w_{i-1}) + \lambda_3 \times P_{ML}(w_i) \end{aligned}$$

Use EM to Set the Lambdas?

- Initialize λ_1 , λ_2 , and λ_3
- Iteration

Step 1: Estimate the following quantity

$$c_1 = \sum_{i=1}^N \frac{\lambda_1 P_{ML}(w_i | w_{i-2}, w_{i-1})}{\lambda_1 P_{ML}(w_i | w_{i-2}, w_{i-1}) + \lambda_2 P_{ML}(w_i | w_{i-1}) + \lambda_3 P_{ML}(w_i)} = \sum_{w_{i-2}, w_{i-1}, w_i \in V} \frac{\text{Count}(w_{i-2}, w_{i-1}, w_i) \lambda_1 P_{ML}(w_i | w_{i-2}, w_{i-1})}{\lambda_1 P_{ML}(w_i | w_{i-2}, w_{i-1}) + \lambda_2 P_{ML}(w_i | w_{i-1}) + \lambda_3 P_{ML}(w_i)}$$

$$c_2 = \sum_{i=1}^N \frac{\lambda_2 P_{ML}(w_i | w_{i-1})}{\lambda_1 P_{ML}(w_i | w_{i-2}, w_{i-1}) + \lambda_2 P_{ML}(w_i | w_{i-1}) + \lambda_3 P_{ML}(w_i)} = \sum_{w_{i-2}, w_{i-1}, w_i \in V} \frac{\text{Count}(w_{i-2}, w_{i-1}, w_i) \lambda_2 P_{ML}(w_i | w_{i-1})}{\lambda_1 P_{ML}(w_i | w_{i-2}, w_{i-1}) + \lambda_2 P_{ML}(w_i | w_{i-1}) + \lambda_3 P_{ML}(w_i)}$$

$$c_3 = \sum_{i=1}^N \frac{\lambda_3 P_{ML}(w_i)}{\lambda_1 P_{ML}(w_i | w_{i-2}, w_{i-1}) + \lambda_2 P_{ML}(w_i | w_{i-1}) + \lambda_3 P_{ML}(w_i)} = \sum_{w_{i-2}, w_{i-1}, w_i \in V} \frac{\text{Count}(w_{i-2}, w_{i-1}, w_i) \lambda_3 P_{ML}(w_i)}{\lambda_1 P_{ML}(w_i | w_{i-2}, w_{i-1}) + \lambda_2 P_{ML}(w_i | w_{i-1}) + \lambda_3 P_{ML}(w_i)}$$

Step 2: Re-estimate the parameters

$$\lambda_1 = \frac{c_1}{c_1 + c_2 + c_3}, \lambda_2 = \frac{c_2}{c_1 + c_2 + c_3}, \lambda_3 = \frac{c_3}{c_1 + c_2 + c_3}$$

- Repeat step 1 and 2 until λ_i 's converge

EM for PCFG - Inside-outside algorithm

Review: PCFG

- Probabilistic Context Free Grammar, G
 - A set of terminals, $\{w^k\}, k=1, \dots, V$
 - A set of nonterminals, $\{N^i\}, i=1, \dots, n$
 - A designated start symbol, N^1
 - A set of rules, $\{N^i \rightarrow \zeta^j\}$
 - A corresponding set of probabilities on rules

$$\forall i, \sum_j P(N^i \rightarrow \zeta^j) = 1$$

Review: Chomsky Normal Form

A context free grammar G in Chomsky Normal Form is as follows

$$N^i \rightarrow N^j N^k$$

$$N^i \rightarrow w^j$$

$$\forall j = 1, \dots, n$$

$$\sum_{r,s} P(N^j \rightarrow N^r N^s) + \sum_k P(N^j \rightarrow w^k) = 1$$

Main Questions for PCFG

- Calculate the probability of a sentence $w_{1:m} (w_1 \dots w_m)$ according to a grammar G:

$$P(w_{1:m} | G)$$

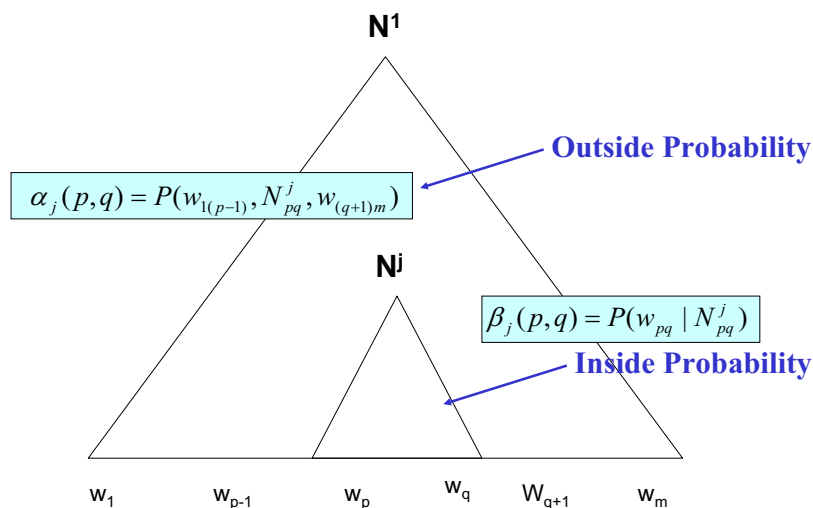
- Identify the most likely parse for a sentence:

$$\arg \max_t P(t | w_{1:m}, G)$$

- Learn probabilities associated with grammar rules in G that maximize the probability of training sentences:**

$$\arg \max_G P(w_{1:m} | G)$$

Inside-outside Probabilities



Inside-outside Probabilities

- Inside probability: total prob of generating words $w_p \dots w_q$ from non-terminal N^j .

$$\beta_j(p, q) = P(w_{pq} | N^j_{pq})$$

- Outside probability: total prob of beginning with the start symbol N^1 and generating N^j_{pq} and all the words outside $w_p \dots w_q$

$$\alpha_j(p, q) = P(w_{1(p-1)}, N^j_{pq}, w_{(q+1)m})$$

Inside Probability

Induction:

- Base case: $\beta_j(k, k) = P(w_k | N_{kk}^j) = P(N^j \rightarrow w_k)$
- Induction:

$$\beta_j(p, q) = P(w_{pq} | N_{pq}^j)$$

$$= \sum_{r,s} \sum_{d=p}^{q-1} P(w_{pd}, N_{pd}^r, w_{(d+1)q}, N_{(d+1)q}^s | N_{pq}^j)$$

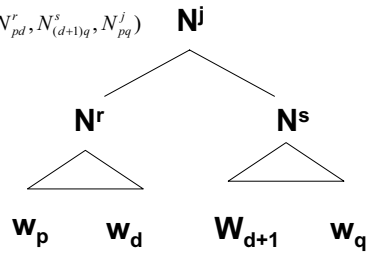
$$= \sum_{r,s} \sum_{d=p}^{q-1} P(w_{(d+1)q} | w_{pd}, N_{pd}^r, N_{(d+1)q}^s, N_{pq}^j) \times P(w_{pd} | N_{pd}^r, N_{(d+1)q}^s, N_{pq}^j)$$

$$\times P(N_{pd}^r, N_{(d+1)q}^s | N_{pq}^j)$$

$$= \sum_{r,s} \sum_{d=p}^{q-1} P(w_{(d+1)q} | N_{(d+1)q}^s) \times P(w_{pd} | N_{pd}^r)$$

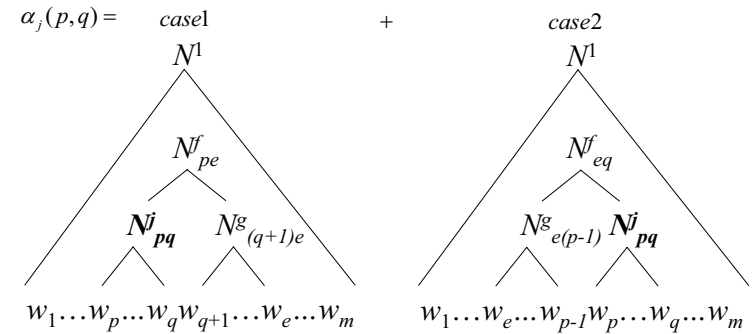
$$\times P(N_{pd}^r, N_{(d+1)q}^s | N_{pq}^j)$$

$$= \sum_{r,s} \sum_{d=p}^{q-1} P(N^j \rightarrow N^r N^s) \beta_r(p, d) \beta_s(d+1, q)$$

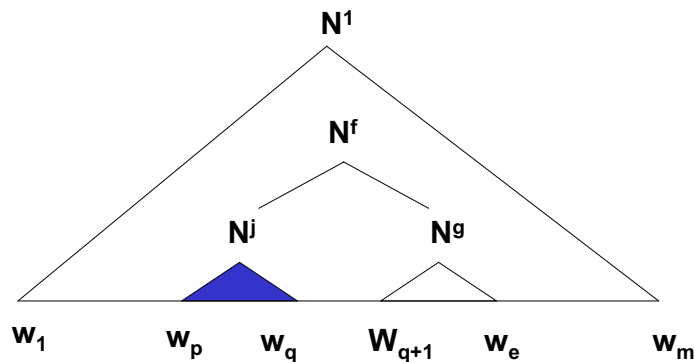


Outside Probability

- Base case $\alpha_j(1, m) = \begin{cases} 1 & j=1 \\ 0 & j \neq 1 \end{cases}$
- Induction

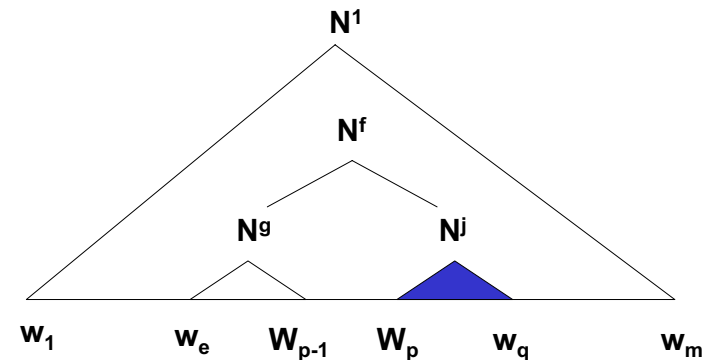


Case 1



$$\alpha_j(p, q) = \sum_{f, g \neq j} \sum_{e=q+1}^m \alpha_f(p, e) P(N^f \rightarrow N^j N^g) \beta_g(q+1, e)$$

Case 2



$$\alpha_j(p, q) = \sum_{f, g} \sum_{e=1}^{p-1} \alpha_f(e, q) P(N^f \rightarrow N^g N^j) \beta_g(e, p-1)$$

Outside probability

Base case:

$$\alpha_j(1, m) = \begin{cases} 1 & \text{if } j = 1 \\ 0 & \text{otherwise} \end{cases}$$

Induction

$$\begin{aligned} \alpha_j(p, q) = & \sum_{f, g \neq j} \sum_{e=q+1}^m \alpha_f(p, e) P(N^f \rightarrow N^j N^g) \beta_g(q+1, e) \\ & + \sum_{f, g} \sum_{e=1}^{p-1} \alpha_f(e, q) P(N^f \rightarrow N^g N^j) \beta_g(e, p-1) \end{aligned}$$

Summary of Probabilities

Using inside probability, bottom up,

$$P(w_{1m}) = P(w_{1m} | N_{1m}^1) = \beta_1(1, m)$$

Using outside probability, top down

$$\begin{aligned} P(w_{1m}) &= \sum_j P(w_{1(k-1)}, w_{kk}, w_{(k+1)m}, N_{kk}^j) \\ &= \sum_j P(w_{kk} | w_{1(k-1)}, w_{(k+1)m}, N_{kk}^j) \times P(w_{1(k-1)}, w_{(k+1)m}, N_{kk}^j) \\ &= \sum_j \alpha_j(k, k) P(N^j \rightarrow w_k) \quad \text{for } \forall k \end{aligned}$$

Product of inside and outside probability

$$P(w_{1m}, N_{pq}^j) = \alpha_j(p, q) \beta_j(p, q)$$

Learning a PCFG

- Inside-outside algorithm
- A special case of the EM algorithm
 - X (observed data): each data point is a sentence w_{1m} .
 - Y (hidden data): parse tree Tr.
 - Θ (parameters):

$$P(N^j \rightarrow N^r N^s)$$

$$P(N^j \rightarrow w_k)$$

Learning a PCFG

- Choose an initial Θ^0

$$P(N^j \rightarrow N^r N^s),$$

$$P(N^j \rightarrow w^k)$$

for all $j, r, s = 1, \dots, n$ and $k = 1, \dots, V$

$$\forall j = 1, \dots, n$$

$$\sum_{r,s} P(N^j \rightarrow N^r N^s) + \sum_k P(N^j \rightarrow w^k) = 1$$

Probabilities for Binary Rules

- E-step: estimate expected count using Θ^{t-1}

$$E(N^j | w_{1m}) = \sum_{p=1}^m \sum_{q=p}^m P(N_{pq}^j | w_{1m}) = \sum_{p=1}^m \sum_{q=p}^m \frac{P(N_{pq}^j, w_{1m})}{P(w_{1m})} = \sum_{p=1}^m \sum_{q=p}^m \frac{\alpha_j(p, q) \beta_j(p, q)}{\beta_1(1, m)} \quad (1)$$

$$\begin{aligned} E(N^j \rightarrow N^r N^s | w_{1m}) &= \sum_{p=1}^{m-1} \sum_{q=p+1}^m P(N^j \rightarrow N^r N^s, N_{pq}^j | w_{1m}) \\ &= \sum_{p=1}^{m-1} \sum_{q=p+1}^m \frac{P(N^j \rightarrow N^r N^s, N_{pq}^j, w_{1m})}{P(w_{1m})} = \frac{\sum_{p=1}^{m-1} \sum_{q=p+1}^m \sum_{d=p}^{q-1} \alpha_j(p, q) P(N^j \rightarrow N^r N^s) \beta_r(p, d) \beta_s(d+1, q)}{\beta_1(1, m)} \quad (2) \end{aligned}$$

- M-step: re-estimate Θ^t

$$P(N^j \rightarrow N^r N^s | w_{1m}) = \frac{P(N^j \rightarrow N^r N^s, N^j | w_{1m})}{P(N^j | w_{1m})} = \frac{(2)}{(1)} = \frac{\sum_{p=1}^m \sum_{q=p}^{q-1} \alpha_j(p, q) P(N^j \rightarrow N^r N^s) \beta_r(p, d) \beta_s(d+1, q)}{\sum_{p=1}^m \sum_{q=p}^m \alpha_j(p, q) \beta_j(p, q)}$$

Probabilities for Unary Rules

- E-step: estimate expected count using Θ^{t-1}

$$E(N^j | w_{1m}) = \sum_{p=1}^m \sum_{q=p}^m P(N_{pq}^j | w_{1m}) = \sum_{p=1}^m \sum_{q=p}^m \frac{P(N_{pq}^j, w_{1m})}{P(w_{1m})} = \sum_{p=1}^m \sum_{q=p}^m \frac{\alpha_j(p, q) \beta_j(p, q)}{\beta_1(1, m)} \quad (1)$$

$$E(N^j \rightarrow w^k | w_{1m}) = \frac{P(N^j \rightarrow w^k, w_{1m})}{P(w_{1m})} = \frac{\sum_{h=1}^m \alpha_j(h, h) \beta_j(h, h) \delta(w_h = w^k)}{\beta_1(1, m)} \quad (3)$$

- M-step: re-estimate Θ^t

$$\begin{aligned} P(N^j \rightarrow w^k | N^j, w_{1m}) &= \frac{P(N^j \rightarrow w^k, N^j | w_{1m})}{P(N^j | w_{1m})} \\ &= \frac{(3)}{(1)} = \frac{\sum_{h=1}^m \alpha_j(h, h) \beta_j(h, h) \delta(w_h = w^k)}{\sum_{p=1}^m \sum_{q=p}^m \alpha_j(p, q) \beta_j(p, q)} \end{aligned}$$

Multiple Training Sentences

- E-step can be done in parallel for each sentence
- M-step

$$\hat{P}(N^j \rightarrow N^r N^s) = \frac{\sum_{i=1}^{\omega} E(N^j \rightarrow N^r N^s | W_i)}{\sum_{i=1}^{\omega} E(N^j | W_i)}$$

$$\hat{P}(N^j \rightarrow w^k) = \frac{\sum_{i=1}^{\omega} E(N^j \rightarrow w^k | W_i)}{\sum_{i=1}^{\omega} E(N^j | W_i)}$$

Analogy with HMM

	HMM	PCFG
Evaluating	Forward / backward algorithm	Inside / outside algorithm
Decoding	Viterbi algorithm	CYK algorithm
Learning	Forward-backward algorithm	Inside-outside algorithm