The EM algorithm

LING 572

Fei Xia

Week 10: 03/09/2010

What is EM?

- EM stands for “expectation maximization”.
- A **parameter estimation** method: it falls into the general framework of **maximum-likelihood estimation** (MLE).
- The general form was given in (Dempster, Laird, and Rubin, 1977), although essence of the algorithm appeared previously in various forms.

Outline

- MLE

- EM
  - Basic concepts
  - Main ideas of EM

- Additional slides:
  - EM for PM models
  - An example: Forward-backward algorithm

MLE

What is MLE?

- Given
  - A sample $X=\{X_1, \ldots, X_n\}$
  - A vector of parameters θ
- We define
  - Likelihood of the data: $P(X \mid \theta)$
  - Log-likelihood of the data: $L(\theta)=\log P(X|\theta)$
- Given X, find $\theta_{ML} = \arg\max_{\theta \in \Omega} L(\theta)$

MLE (cont)

- Often we assume that X_is are independently identically distributed (i.i.d.)

$$
\begin{aligned}
\theta_{ML} &= \arg\max_{\theta\in\Omega} \ L(\theta) \\
&= \arg\max_{\theta\in\Omega} \ \log P(X \mid \theta) \\
&= \arg\max_{\theta\in\Omega} \ \log P(X_1,\ldots,X_n \mid \theta) \\
&= \arg\max_{\theta\in\Omega} \ \log \prod_i \ P(X_i \mid \theta) \\
&= \arg\max_{\theta\in\Omega} \ \sum_i \log P(X_i \mid \theta)
\end{aligned}
$$

- Depending on the form of P(X | θ), solving this optimization problem can be easy or hard.

An easy case

- Assuming
  - A coin has a probability p of being heads, 1-p of being tails.
  - Observation: We toss a coin N times, and the result is a set of Hs and Ts, and there are m Hs.

- What is the value of p based on MLE, given the observation?

An easy case (cont)**

$$L(\theta) = \log P(X \mid \theta) = \log p^m (1-p)^{N-m}$$

$$= m \log p + (N-m)\log(1-p)$$

$$\frac{dL(\theta)}{dp} = \frac{d(m \log p + (N-m)\log(1-p))}{dp} = \frac{m}{p} - \frac{N-m}{1-p} = 0$$

p= m/N

EM: basic concepts

Basic setting in EM

- X is a set of data points: **observed** data
- θ is a parameter vector.
- EM is a method to find θ_{ML} where

$$\theta_{ML} = \arg\max_{\theta \in \Omega} L(\theta)$$

$$= \arg\max_{\theta \in \Omega} \log P(X \mid \theta)$$

- Calculating $P(X \mid \theta)$ directly is hard.
- Calculating $P(X,Y|\theta)$ is much simpler, where Y is “hidden” data (or “missing” data).

The basic EM strategy

- $Z = (X, Y)$
  - Z: complete data (“augmented data”)
  - X: observed data (“incomplete” data)
  - Y: hidden data (“missing” data)

The “missing” data Y

- Y need not necessarily be missing in the practical sense of the word.

- It may just be a conceptually convenient technical device to simplify the calculation of $P(X \mid \theta)$.

- There could be many possible Ys.

Examples of EM

	HMM	PCFG	MT	Coin toss
X (observed)	sentences	sentences	Parallel data	Head-tail sequences
Y (hidden)	State sequences	Parse trees	Word alignment	Coin id sequences
θ	a_{ij} b_{ijk}	$P(A \rightarrow BC)$	$t(f \mid e)$ $d(a_j \mid j, l, m)$, …	p1, p2, λ
Algorithm	Forward-backward	Inside-outside	IBM Models	N/A

The EM algorithm

- Consider a set of starting parameters
- Use these to “estimate” the missing data
- Use “complete” data to update parameters
- Repeat until convergence

Highlights

- General algorithm for missing data problems
- Requires “specialization” to the problem at hand
- Examples of EM:
  - Forward-backward algorithm for HMM
  - Inside-outside algorithm for PCFG
  - EM in IBM MT Models

EM: main ideas

Idea #1: find θ that maximizes the likelihood of training data

$$\theta_{ML} = \arg\max_{\theta \in \Omega} \; L(\theta)$$

$$= \arg\max_{\theta \in \Omega} \; \log P(X \mid \theta)$$

Idea #2: find the θ^t sequence

No analytical solution ➔ iterative approach, find

$$\theta^0, \theta^1, ..., \theta^t, ...$$

s.t.

$$l(\theta^0) < l(\theta^1) < ... < l(\theta^t) < ....$$

Idea #3: find θ^{t+1} that maximizes a tight lower bound of $l(\theta) - l(\theta^t)$

$$l(\theta) - l(\theta^t) \geq \boxed{\sum_{i=1}^{n} E_{P(y|x_i,\theta^t)}[\log \frac{P(x_i, y \mid \theta)}{P(x_i, y \mid \theta^t)}]}$$

a tight lower bound

Idea #4: find θ^{t+1} that maximizes the Q function**

Lower bound of $l(\theta) - l(\theta^t)$

$$\theta^{(t+1)} = \arg\max_{\theta} \boxed{\sum_{i=1}^{n} E_{P(y|x_i,\theta^t)}[\log \frac{p(x_i, y \mid \theta)}{p(x_i, y \mid \theta^t)}]}$$

$$= \arg\max_{\theta} \boxed{\sum_{i=1}^{n} E_{P(y|x_i,\theta^t)}[\log P(x_i, y \mid \theta)]}$$

The Q function

The Q-function**

- Define the Q-function (a function of θ):

$$Q(\theta,\theta^t) = E_{P(Y|X,\theta^t)}[\log P(X,Y|\theta)]$$

$$= \sum_{i=1}^{n} E_{P(y|x_i,\theta^t)}[\log P(x_i,y|\theta)]$$

$$= \sum_{i=1}^{n} \sum_{y} P(y|x_i,\theta^t) \log P(x_i,y|\theta)$$

  - Θ^t is the current parameter estimate and is a constant (vector).
  - Θ is the normal variable (vector) that we wish to adjust.

- The Q-function is the expected value of the complete data log-likelihood P(X,Y|θ) with respect to Y given X and θ^t.

Calculating model expectation in MaxEnt

$$E_p f_j = \sum_{x \in X, y \in Y} p(x, y) f_j(x, y)$$

$$= \sum_{x \in X, y \in Y} p(x) p(y \mid x) f_j(x, y) \qquad \approx \sum_{x \in X, y \in Y} \tilde{p}(x) p(y \mid x) f_j(x, y)$$

$$= \sum_{x \in X} \tilde{p}(x) \sum_{y \in Y} p(y \mid x) f_j(x, y) \qquad = \sum_{x \in S} \tilde{p}(x) \sum_{y \in Y} p(y \mid x) f_j(x, y)$$

$$= \frac{1}{N} \sum_{i=1}^{N} \sum_{y \in Y} p(y \mid x_i) f_j(x_i, y)$$

The EM algorithm

- Start with initial estimate, θ^0
- Repeat until convergence
  - E-step: calculate

$$Q(\theta,\theta^t) = \sum_{i=1}^{n}\sum_{y} P(y \mid x_i, \theta^t) \log P(x_i, y \mid \theta)$$

  - M-step: find

$$\theta^{(t+1)} = \arg\max_{\theta} Q(\theta,\theta^t)$$

Important classes of EM problem

- Products of multinomial (PM) models
- Exponential families
- Gaussian mixture
- …

Summary

- EM falls into the general framework of **maximum-likelihood estimation** (MLE).

- Calculating $P(X \mid \theta)$ directly is hard.

  ➔ Introduce Y (“hidden” data), and calculating $P(X,Y|\theta)$ instead.

- Start with initial estimate, repeat E-step and M-step until convergence.

- Closed-form solutions for some classes of models.

Strengths of EM

- Numerical stability: in every iteration of the EM algorithm, it increases the likelihood of the observed data.

- The EM handles parameter constraints gracefully.

Problems with EM

- Convergence can be very slow on some problems and is intimately related to the amount of missing information.

- It guarantees to improve the probability of the training corpus, which is different from reducing the errors directly.

- It cannot guarantee to reach global maxima (it could get struck at the local maxima, saddle points, etc)

  ➔ The initial estimate is important.

Additional slides

The EM algorithm for PM models

PM models

$$p(x, y \mid \theta) = \prod_{i} p_i^{\,c(i,x,y)} = \prod_{i \in \Omega_1} p_i^{\,c(i,x,y)} \times \ldots \times \prod_{i \in \Omega_m} p_i^{\,c(i,x,y)}$$

Where $(\Omega_1, \ldots, \Omega_m)$ is a partition of all the parameters, and for any j

$$\sum_{i \in \Omega_j} p_i = 1$$

HMM is a PM

$$p(x, y \mid \theta)$$

$$= \prod_{s_i, s_j} p(s_i \Rightarrow s_j)^{c(s_i \Rightarrow s_j, x, y)} \times$$

$$\prod_{s_i, s_j, w_k} p(s_i \overset{w_k}{\Rightarrow} s_j)^{c(s_i \overset{w_k}{\Rightarrow} s_j, x, y)}$$

$$\sum_j a_{ij} = 1$$

$$\sum_k b_{ijk} = 1$$

PCFG

- PCFG: each sample point (x,y):
  - x is a sentence
  - y is a possible parse tree for that sentence.

$$P(x, y \mid \theta) = \prod_{i=1}^{n} P(A_i \to \beta_i \mid A_i)$$

$$P(x, y \mid \theta) =$$

$$P(S \to NP\,VP \mid S) \times$$

$$P(NP \to Jim \mid NP) \times$$

$$P(VP \to sleeps \mid VP)$$

PCFG is a PM

$$p(x, y \mid \theta)$$

$$= \prod_{A,\beta} p(A \Rightarrow \beta)^{c(A \Rightarrow \beta, x, y)}$$

$$\sum_{A} p(A \Rightarrow \beta) = 1$$

Q-function for PM

$$
\begin{aligned}
&Q(\theta,\theta^t) \\
&= \sum_{i=1}^{n}\sum_{y} P(y \mid x_i,\theta^t)\log P(x_i, y \mid \theta) \\
&= \sum_{i=1}^{n}\sum_{y} P(y \mid x_i,\theta^t)\log\Big(\prod_{k}\Big(\prod_{j\in\Omega_k} p_j^{\,C(j,x,y)}\Big)\Big) \\
&= \Big(\sum_{i=1}^{n}\sum_{y} P(y \mid x_i,\theta^t)\log\prod_{j\in\Omega_1} p_j^{\,C(j,x_i,y)}\Big) + \ldots + \Big(\sum_{i=1}^{n}\sum_{y} P(y \mid x_i,\theta^t)\log\prod_{j\in\Omega_k} p_j^{\,C(j,x_i,y)}\Big) \\
&= \boxed{\Big(\sum_{i=1}^{n}\sum_{y} P(y \mid x_i,\theta^t)\sum_{j\in\Omega_1} C(j,x_i,y)\log p_j\Big)} + \ldots + \Big(\sum_{i=1}^{n}\sum_{y} P(y \mid x_i,\theta^t)\sum_{j\in\Omega_k} C(j,x_i,y)\log p_j\Big)
\end{aligned}
$$

$$Q_1(\theta,\theta^t)$$

Maximizing the Q function

Maximize

$$Q_1(\theta,\theta^t)=\sum_{i=1}^{n}\sum_{y}P(y\mid x_i,\theta^t)\sum_{j\in\Omega_1}C(j,x_i,y)\log p_j$$

Subject to the constraint $\sum_{j\in\Omega_1}p_j=1$

Use Lagrange multipliers

$$\hat{Q}_1(\theta,\theta^t)=(\sum_{i=1}^{n}\sum_{y}P(y\mid x_i,\theta^t)\sum_{j\in\Omega_1}C(j,x_i,y)\log p_j)-\lambda\sum_{j\in\Omega_1}p_j$$

Optimal solution

$$\hat{Q}_1(\theta,\theta^t) = \left(\sum_{i=1}^{n}\sum_{y} P(y \mid x_i,\theta^t)\sum_{j\in\Omega_1} C(j,x_i,y)\log p_j\right) - \lambda \sum_{j\in\Omega_1} p_j$$

$$\frac{\partial \hat{Q}_1(\theta,\theta^t)}{\partial p_j} = \left(\sum_{i=1}^{n}\sum_{y} p(y \mid x_i,\theta^t) C(j,x_i,y) / p_j\right) - \lambda = 0$$

Expected count

$$p_j = \frac{\sum_{i=1}^{n}\sum_{y} p(y \mid x_i,\theta^t) C(j,x_i,y)}{\lambda}$$

Normalization factor

PCFG example

- Calculate expected counts

$$\overline{Count}(S \rightarrow NP\ VP)) = \sum_{i=1}^{m} \sum_{y} P(y|x^i, \Theta^{t-1}) Count(x^i, y, S \rightarrow NP\ VP)$$

- Update parameters

$$P(S \rightarrow NP\ VP|S) = \frac{\overline{Count}(S \rightarrow NP\ VP)}{\sum_{S \rightarrow \beta \in R} \overline{Count}(S \rightarrow \beta)}$$

EM: An example

Forward-backward algorithm

- HMM is a PM (Product of Multi-nominal) Model

- Forward-backward algorithm is a special case of the EM algorithm for PM Models.

- X (observed data): each data point is an O_{1T}.
- Y (hidden data): state sequence X_{1T}.
- Θ (parameters): a_{ij}, b_{ijk}, $\pi_{i.}$

Expected counts

$$count(s_i \to s_j) = \sum_{Y} P(Y \mid X, \theta) * count(X, Y, s_i \to s_j)$$

$$= \sum_{X_{1T}} P(X_{1T} \mid O_{1T}, \theta) * count(O_{1T}, X_{1T}, s_i \to s_j)$$

$$= \sum_{t=1}^{T} P(X_t = i, X_{t+1} = j \mid O_{1T})$$

$$= \sum_{t=1}^{T} \xi_{ij}(t)$$

Expected counts (cont)

$$count(s_i \xrightarrow{w_k} s_j) = \sum_{Y} P(Y \mid X, \theta) * count(X, Y, s_i \xrightarrow{w_k} s_j)$$

$$= \sum_{X_{1T}} P(X_{1T} \mid O_{1T}, \theta) * count(O_{1T}, X_{1T}, s_i \xrightarrow{w_k} s_j)$$

$$= \sum_{t=1}^{T} P(X_t = i, X_{t+1} = j \mid O_{1T}, \theta) * \delta(O_k, w_k)$$

$$= \sum_{t=1}^{T} \xi_{ij}(t) \delta(O_k, w_k)$$

The inner loop for forward-backward algorithm

Given an input sequence and (S, K, Π, A, B)

1. Calculate forward probability:
   - Base case $\alpha_i(1) = \pi_i$
   - Recursive case: $\alpha_j(t+1) = \sum_i \alpha_i(t) a_{ij} b_{ijo_t}$
2. Calculate backward probability:
   - Base case: $\beta_i(T+1) = 1$
   - Recursive case: $\beta_i(t) = \sum_j \beta_j(t+1) a_{ij} b_{ijo_t}$
3. Calculate expected counts: $$\xi_{ij}(t) = \frac{\alpha_i(t) a_{ij} b_{ijo_t} \beta_j(t+1)}{\sum_{m=1}^{N} \alpha_m(t) \beta_m(t)}$$
4. Update the parameters:

$$a_{ij} = \frac{\sum_{t=1}^{T} \xi_{ij}(t)}{\sum_{j=1}^{N} \sum_{t=1}^{T} \xi_{ij}(t)} \qquad b_{ijk} = \frac{\sum_{t=1}^{T} \delta(o_t, w_k) \xi_{ij}(t)}{\sum_{t=1}^{T} \xi_{ij}(t)}$$

HMM

- A HMM is a tuple (S, Σ, Π, A, B) :
  - A set of states $S=\{s_1, s_2, \ldots, s_N\}$.
  - A set of output symbols $\Sigma=\{w_1, \ldots, w_M\}$.
  - Initial state probabilities $\Pi = \{\pi_i\}$
  - State transition prob: $A=\{a_{ij}\}$.
  - Symbol emission prob: $B=\{b_{ijk}\}$

- State sequence: $X_1 \ldots X_{T+1}$
- Output sequence: $o_1 \ldots o_T$

Forward probability

The probability of producing $o_{i,t-1}$ while ending up in state s^i:

$$\alpha_i(t) \stackrel{def}{=} P(O_{1,t-1}, X_t = i)$$

Calculating forward probability

Initialization: $$\alpha_i(1) = \pi_i$$

Induction:

$$\alpha_j(t+1) = P(O_{1,t}, X_{t+1} = j)$$

$$= \sum_i P(O_{1,t}, X_t = i, X_{t+1} = j)$$

$$= \sum_i P(O_{1,t-1}, X_t = i) * P(o_t, X_{t+1} = j \mid O_{1,t-1}, X_t = i)$$

$$= \sum_i P(O_{1,t-1}, X_t = i) * P(o_t, X_{t+1} = j \mid X_t = i)$$

$$= \sum_i \alpha_i(t) a_{ij} b_{ijo_t}$$

Backward probability

- The probability of producing the sequence $O_{t,T}$, given that at time t, we are at state s^i.

$$\beta_i(t) \overset{def}{=} P(O_{t,T} \mid X_t = i)$$

Calculating backward probability

Initialization: $\beta_i(T+1) = 1$

Induction:

$$\beta_i(t) \overset{def}{=} P(O_{t,T} \mid X_t = i)$$

$$= \sum_j P(o_t, O_{(t+1),T}, X_{t+1} = j \mid X_t = i)$$

$$= \sum_j P(o_t, X_{t+1} = j \mid X_t = i) * P(O_{(t+1),T} \mid X_t = i, X_{t+1} = j, o_t)$$

$$= \sum_j P(o_t, X_{t+1} = j \mid X_t = i) * P(O_{t+1,T} \mid X_{t+1} = j)$$

$$= \sum_j \beta_j(t+1) a_{ij} b_{ijo_t}$$

Calculating the prob of the observation

$$P(O) = \sum_{i=1}^{N} \alpha_i (T+1)$$

$$P(O) = \sum_{i=1}^{N} \pi_i \beta_i (1)$$

$$P(O) = \sum_{i=1}^{N} P(O, X_t = i)$$

$$= \sum_{i=1}^{N} \alpha_i (t) \beta_i (t)$$

Estimating parameters

- The prob of traversing a certain arc at time t given O: (denoted by $p_t(i, j)$ in M&S)

$$\xi_{ij}(t) = P(X_t = i, X_{t+1} = j \mid O)$$

$$= \frac{P(X_t = i, X_{t+1} = j, O)}{P(O)}$$

$$= \frac{\alpha_i(t) a_{ij} b_{ijo_t} \beta_j(t+1)}{\sum_{m=1}^{N} \alpha_m(t) \beta_m(t)}$$

The prob of being at state i at time t given O:

$$\gamma_i(t) = P(X_t = i \mid O) = \sum_{j=1}^{N} P(X_t = i, X_{t+1} = j \mid O)$$

$$\gamma_i(t) = \sum_{j=1}^{N} \xi_{ij}(t)$$

Expected counts

Sum over the time index:

- Expected # of transitions from state i to j in O:

$$\sum_{t=1}^{T} \xi_{ij}(t)$$

- Expected # of transitions from state i in O:

$$\sum_{t=1}^{T} \gamma_i(t) = \sum_{t=1}^{T} \sum_{j=1}^{N} \xi_{ij}(t) = \sum_{j=1}^{N} \sum_{t=1}^{T} \xi_{ij}(t)$$

Update parameters

$$\hat{\pi}_i = \exp ected \;\; frequency \; in \; state \; i \; at \; time \; t = 1 \quad = \gamma_i(1)$$

$$a_{ij} = \frac{\exp ected \;\; \# \, of \;\; transitions \;\; from \; state \; i \; to \;\; j}{\exp ected \;\; \# \, of \;\; transitions \;\; from \; state \; i} = \frac{\sum_{t=1}^{T} \xi_{ij}(t)}{\sum_{t=1}^{T} \gamma_i(t)} = \frac{\sum_{t=1}^{T} \xi_{ij}(t)}{\sum_{j=1}^{N} \sum_{t=1}^{T} \xi_{ij}(t)}$$

$$b_{ijk} = \frac{\exp ected \;\; \# \, of \;\; transitions \;\; from \; state \; i \; to \;\; j \; with \; k \; observed}{\exp ected \;\; \# \, of \;\; transitions \;\; from \; state \; i \; to \;\; j} \quad = \quad \frac{\sum_{t=1}^{T} \delta(o_t, w_k) \xi_{ij}(t)}{\sum_{t=1}^{T} \xi_{ij}(t)}$$

Final formulae

$$\xi_{ij}(t) = \frac{\alpha_i(t) a_{ij} b_{ijo_t} \beta_j(t+1)}{\sum_{m=1}^{N} \alpha_m(t) \beta_m(t)}$$

$$a_{ij} = \frac{\sum_{t=1}^{T} \xi_{ij}(t)}{\sum_{j=1}^{N} \sum_{t=1}^{T} \xi_{ij}(t)}$$

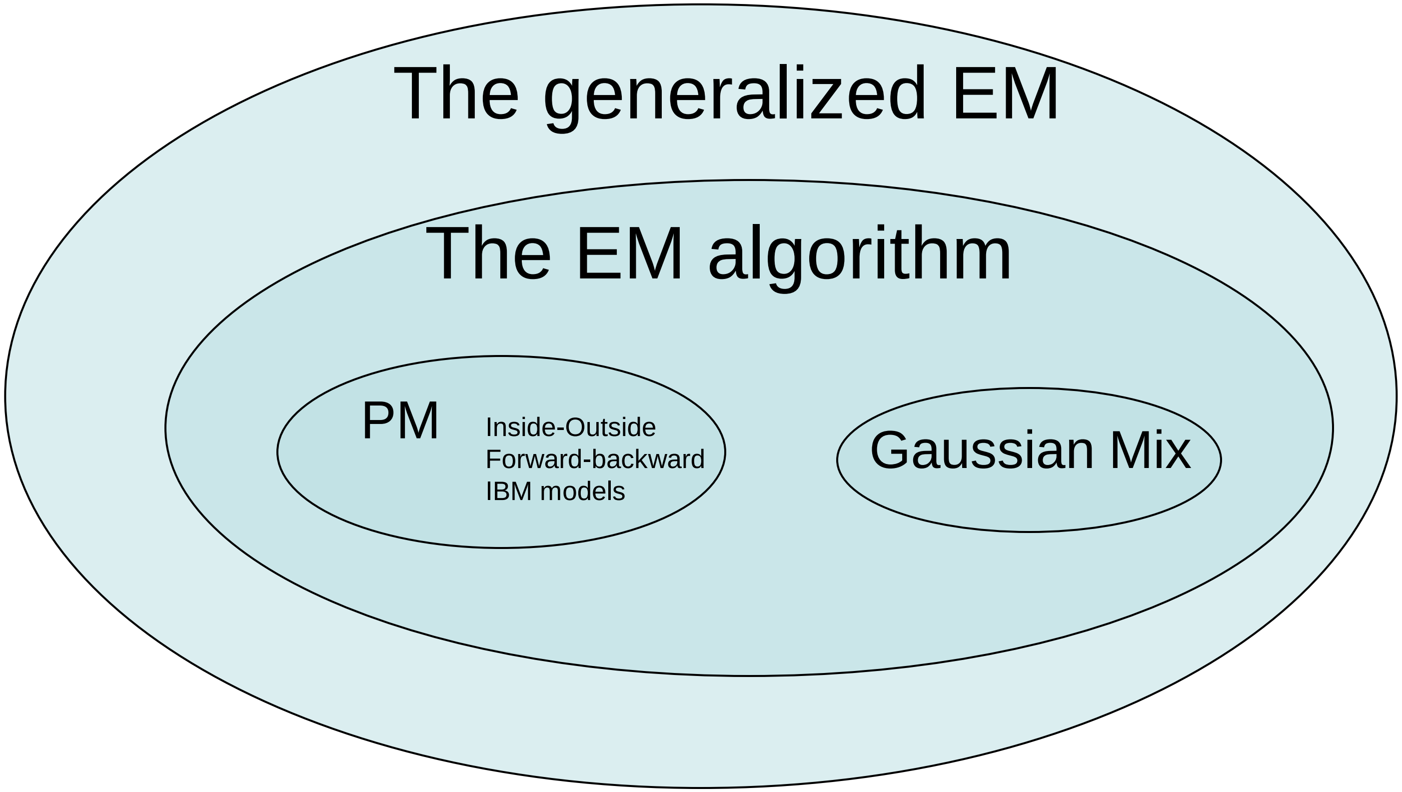
$$b_{ijk} = \frac{\sum_{t=1}^{T} \delta(o_t, w_k) \xi_{ij}(t)}{\sum_{t=1}^{T} \xi_{ij}(t)}$$

Summary

- The EM algorithm
  - An iterative approach
  - L(θ) is non-decreasing at each iteration
  - Optimal solution in M-step exists for many classes of problems.

- The EM algorithm for PM models
  - Simpler formulae
  - Three special cases
    - Inside-outside algorithm
    - Forward-backward algorithm
    - IBM Models for MT

Relations among the algorithms



The EM algorithm for PM models

Algorithm: For $t = 1 \ldots T$, // for each iteration

- For $r = 1 \ldots |\Theta|$, set $\overline{Count}(r) = 0$
- For $i = 1 \ldots m$, // for each training example x_i
  - For all y, calculate $t_y = P(x^i, y|\Theta^{t-1})$ // for each possible y
  - Set $sum = \sum_y t_y$
  - For all y, set $u_y = t_y/sum$ (note that $u_y = P(y|x^i, \Theta^{t-1})$)
  - For all $r = 1 \ldots |\Theta|$, set // for each parameter

$$\overline{Count}(r) = \overline{Count}(r) + \sum_y u_y Count(x^i, y, r)$$

- For all $r = 1 \ldots |\Theta|$, set // for each parameter

$$\Theta_r^t = \frac{\overline{Count}(r)}{Z}$$

where Z is a normalization constant that ensures that the multinomial distribution of which Θ_r^t is a member sums to 1.