The EM Algorithm

Vincent Ng
Cornell University

Slides based on notes by:
Lillian Lee, Ted Pedersen, and Roni Rosenfeld

The Statistical Modeling Framework

◆ Task
  → Given data x and a model parameterized by \( \theta \), find the \( \theta \) that maximizes the likelihood of x.

\[
\theta^* = \arg \max_{\theta} P_\theta(x) = \arg \max_{\theta} \log(P_\theta(x))
\]

◆ Why using EM for ML estimation?
  → Suppose \( P_\theta(x) \) hard to maximize, but there exists hidden data \( h \) such that \( \arg \max_{\theta} P_\theta(x, h) \) is easy.
  → EM (Dempster, Laird, Rubin, 1977) enables us to make MLE in the presence of hidden data.

Combining Language Models

◆ Given two language models \( M_A \) and \( M_B \), create a hybrid model \( M_H \) that, every time it is consulted, stochastically chooses which of the two models to use, with probability \( \lambda \) of choosing \( M_A \).

◆ Now, given a sample \( D = \{s_1, s_2, ..., s_n\} \) generated by \( M_H \), find an ML estimate for \( \lambda \).

Combining Language Models

◆ Given two language models \( M_A \) and \( M_B \), create a hybrid model \( M_H \) that, every time it is consulted, stochastically chooses which of the two models to use, with probability \( \lambda \) of choosing \( M_A \).

◆ Now, given a sample \( D = \{s_1, s_2, ..., s_n\} \) generated by \( M_H \), find an ML estimate for \( \lambda \).

◆ Observable data: D
◆ Hidden data: M (which model generated \( s_i \))
Maximizing Log Likelihood

Goal: \( \arg \max_{\lambda} \log(P_\lambda(D)) \)

\[
\log(P_\lambda(D)) = \log(P_\lambda(s_1, s_2, \ldots, s_n)) = \log \prod_i P_\lambda(s_i) = \sum_i \log P_\lambda(s_i) = \sum_i \log(\lambda P_\lambda(s_i) + (1 - \lambda) P_B(s_i))
\]

\[\lambda\] Maximizing the incomplete log likelihood is hard!

Maximizing Log Likelihood

Goal: \( \arg \max_{\lambda} \log(P_\lambda(D, M)) \)

\[
\log(P_\lambda(D, M)) = \log(P_\lambda(s_1, M_1, s_2, M_2, \ldots, s_n, M_n)) = \log \prod_i P_\lambda(s_i, M_i)
\]

\[z_{1i} = 1 \text{ iff } M_A \text{ generated } s_i, \quad z_{2i} = 1 \text{ iff } M_B \text{ generated } s_i\]

\[= \sum_i z_{1i} \log(\lambda P_\lambda(s_i)) + z_{2i} \log((1 - \lambda) P_B(s_i)) \quad z_{ij} \in \{0, 1\}\]

But \(M\) is hidden data! \(EM\) can help us …

Is Maximizing Complete Log Likelihood Easier?

\[
\arg \max_{\lambda} \log(P_\lambda(D, M))
\]

\[
\log(P_\lambda(D, M)) = \log(P_\lambda(s_1, M_1, s_2, M_2, \ldots, s_n, M_n)) = \log \prod_i P_\lambda(s_i, M_i)
\]

\[z_{1i} = 1 \text{ iff } M_A \text{ generated } s_i, \quad z_{2i} = 1 \text{ iff } M_B \text{ generated } s_i\]

\[= \sum_i z_{1i} \log(\lambda P_\lambda(s_i)) + z_{2i} \log((1 - \lambda) P_B(s_i)) \quad z_{ij} \in \{0, 1\}\]

But \(M\) is hidden data! \(EM\) can help us …

The story so far …

Goal: \( \arg \max_{\lambda} \log(P_\lambda(D)) \)

But this is hard!

\(EM\) helps us compute \( \arg \max_{\lambda} \log(P_\lambda(D, M)) \)

To achieve our goal, we will

\(\checkmark\) show how \(EM\) maximizes the complete log likelihood

\(\checkmark\) show that maximizing the complete log likelihood also maximizes the incomplete log likelihood
Maximizing Complete Log Likelihood

**Observation**
- If we knew which of the two models was used to generate each $s_i$, we could estimate $\lambda$ as follows:
  \[ \lambda = \frac{\text{number of times } M_A \text{ was chosen}}{n} \]
- But the choice of the two models is a *hidden event*!
- But we do *not* need to know which model was used
- We only need to know # times $M_A$ was chosen
- Still we do not know this quantity
- But we can calculate its expected value!

**Idea:** guess the value of $\lambda$ and iteratively improve the estimate (w.r.t. the complete log likelihood)

Maximizing Complete Log Likelihood

**Initialize $\lambda$ to some arbitrarily non-zero value**

**At step $k$**
- Let $\lambda^k$ be the current estimate for $\lambda$
- Compute the expected # times $M_A$ was used from data

\[
E_{\lambda^k}(M = A \mid s_1, s_2, \ldots, s_n) = \lambda^k * P_A(s_i) + (1 - \lambda^k) * P_B(s_i)
\]

\[
= \sum_i \frac{\lambda^k * P_A(s_i)}{P_A(s_i) + (1 - \lambda^k) * P_B(s_i)}
\]

\[
= \sum_i \frac{\lambda^k * P_A(s_i)}{\lambda^k * P_A(s_i) + (1 - \lambda^k) * P_B(s_i)}
\]

\[
= \sum_i \frac{\lambda^k * P_A(s_i)}{\lambda^k * P_A(s_i) + (1 - \lambda^k) * P_B(s_i)}
\]

\[
= \frac{\lambda^k}{\lambda^k + (1 - \lambda^k) * \frac{P_B(s_i)}{P_A(s_i)}}
\]

\[
= \frac{\lambda^k}{\lambda^k + (1 - \lambda^k) * \frac{P_B(s_i)}{P_A(s_i)}}
\]

**E-step**

**At step $k$**
- Let $\lambda^k$ be the current estimate for $\lambda$
- Compute the expected # times $M_A$ was used from data
- Improve the estimate of $\lambda$ using the statistics obtained from the E-step

\[
\lambda^{k+1} = \frac{E_{\lambda^k}(M = A \mid s_1, s_2, \ldots, s_n)}{n}
\]

Find $\lambda^{k+1}$ so that the expected incomplete log likelihood is maximized given $\lambda^k$
Maximizing Complete Log Likelihood

◆ At step $k$
  ‣ Let $\lambda^k$ be the current estimate for $\lambda$.
  ‣ Compute the expected # times $M_A$ was used from data
  ‣ Improve the estimate of $\lambda$ using the statistics obtained from the E-step

$$
\lambda^{k+1} = \frac{E_{\theta_k}(M = A | s_1, s_2, ..., s_n)}{n}
$$

◆ Terminate if change in log likelihood $< \delta$

The EM Algorithm

◆ Initialize the $\theta$'s to some arbitrary (non-zero) values $\theta^0_i$
◆ Iterate the E-step and the M-step until convergence.
  During step $k$,
  ‣ compute the expected values of the hidden data based on the current parameter estimates $\theta^k_i$ (E-step)
  ‣ derive $\theta^{k+1}_i$ as an ML estimate using the values of the hidden data computed in the E-step (M-step)

Properties of EM

◆ The complete log likelihood function is bounded and increases after each iteration
  ‣ EM always converges (in terms of log likelihood)
  ‣ But convergence may be to a local maximum

The story so far …

◆ Showed how EM can be used to maximize the complete log likelihood
◆ But our goal is to maximize the incomplete log likelihood
◆ Need to show that maximizing the complete log likelihood also maximizes the incomplete log likelihood
**Theorem:**
Let $x$ be our incomplete data, $h$ our hidden data, and $\theta$ a parametric model that generates $x$ and $h$.
If we choose $\theta'$ such that

$$ E_{P_{\theta'(h|x)}} \log P_{\theta'}(x, h) > E_{P_{\theta'(h|x)}} \log P_{\theta}(x, h) $$

then

$$ P_{\theta'}(x) > P_{\theta}(x) $$

**Proof of EM**

$$ P_{\theta'}(x, h) = P_{\theta}(x) P_{\theta}(h | x) $$

$$ \log P_{\theta'}(x, h) = \log P_{\theta}(x) P_{\theta}(h | x) $$

$$ \log P_{\theta'}(x) = \log P_{\theta}(x, h) - \log P_{\theta}(h | x) $$

Taking expectation on both sides w.r.t. $P_{\theta}(h | x)$, we have

$$ E_{P_{\theta}(h|x)} \log P_{\theta}(x) = E_{P_{\theta}(h|x)} \log P_{\theta}(x, h) - E_{P_{\theta}(h|x)} \log P_{\theta}(h | x) $$

$$(*) \quad \log P_{\theta}(x) = E_{P_{\theta}(h|x)} \log P_{\theta}(x, h) - E_{P_{\theta}(h|x)} \log P_{\theta}(h | x) $$

Is $\log P_{\theta'}(x) > \log P_{\theta}(x)$?

$$(*) \quad \log P_{\theta}(x) = E_{P_{\theta}(h|x)} \log P_{\theta}(x, h) - E_{P_{\theta}(h|x)} \log P_{\theta}(h | x) $$

Now, by assumption and the lemma, we have:

$$ E_{P_{\theta}(h|x)} \log P_{\theta}(x, h) \geq E_{P_{\theta}(h|x)} \log P_{\theta}(x, h) $$

By the lemma, we have:

$$ (-E_{P_{\theta}(h|x)} \log P_{\theta}(h | x)) \geq (-E_{P_{\theta}(h|x)} \log P_{\theta}(h | x)) $$

Adding the two gives $\log P_{\theta'}(x) > \log P_{\theta}(x)$
NLP Applications using EM

- Estimating the values of hidden variables
  - HMM training: forward-backward/Baum-Welch (1972)
  - PCFG training: inside-outside (Baker, 1979)
  - Word alignment in a parallel corpus (Brown et al., 1993)

- Unsupervised learning of clusters
  - Distributional clustering of nouns (Periera et al., 1993)
  - Learning subcategorization frames (Rooth et al., 1999)

- Improving parameter estimates of finite mixtures
  - Semi-supervised text classification (Nigam et al., 2000)

Using EM in Practice

- EM may not work well in practice …

  - Potential problems
    - get stuck at a local maximum
    - toggle between two local maxima with different $\theta$'s

  - Solutions
    - select a number of different starting points
    - searching by simulated annealing
    - bootstrap EM by labeled data (Nigam et al., 2000)
    - combine EM with active learning (McCallum and Nigam, 1998)

Using EM in Practice (Cont’)

- EM may not work well in practice …

  - Potential problem
    - overfitting to the training data

  - Solution
    - use held-out data to monitor overfitting

Using EM in Practice (Cont’)

- EM may not work well in practice …

  - Potential problem
    - the underlying generative model is incorrect

  - Solution
    - fix your model!
    - designing a “good” model is not a trivial problem