CS674 Natural Language Processing

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 θ, find the θ 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 λ of choosing M_A.
- Now, given a sample $D = \{s_1, s_2, \ldots, s_n\}$ generated by M_H, find an ML estimate for λ.

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 λ of choosing M_A.
- Now, given a sample $D = \{s_1, s_2, \ldots, s_n\}$ generated by M_H, find an ML estimate for λ.
- Observable data: D
- Hidden data: M (which model generated s_i)

Maximizing Log Likelihood

Goal: $\arg\max_\lambda \log(P_\lambda(D))$

$$\log(P_\lambda(D)) \quad \text{Incomplete log-likelihood}$$

$$= \log(P_\lambda(s_1, s_2, \ldots, s_n))$$

$$= \log \prod_i P_\lambda(s_i)$$

$$= \sum_i \log P_\lambda(s_i)$$

$$= \sum_i \log(\lambda P_A(s_i) + (1-\lambda) P_B(s_i))$$

- Maximizing the incomplete log likelihood is hard!

Maximizing Log Likelihood

- Goal: $\arg\max_\lambda \log(P_\lambda(D))$ Incomplete log-likelihood
- Assume we have access to the hidden data.
- We can try to $\arg\max_\lambda \log(P_\lambda(D, M))$ where M_i is the model used to generate s_i.
- Claim: $\arg\max_\lambda \log(P_\lambda(D, M))$ is easier

Complete log-likelihood

Is Maximizing Complete Log Likelihood Easier?

$\arg\max_\lambda \log(P_\lambda(D, M))$

$$\log(P_\lambda(D, M))$$

$$= \log(P_\lambda(s_1, M_1, s_2, M_2, \ldots, s_n, M_n))$$

$$= \log \prod_i P_\lambda(s_i, M_i)$$

$$= \sum_i \log P_\lambda(s_i, M_i)$$

$$= \sum_i z_{i1} \log(\lambda P_A(s_i)) + z_{i2} \log((1-\lambda) P_B(s_i))$$

z_{i1} = 1 iff M_A generated s_i
z_{i2} = 1 iff M_B generated s_i

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
  - show how EM maximizes the complete log likelihood
  - 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 λ as follows:

$$\lambda = \frac{number\ of\ times\ M_A\ was\ chosen}{n}$$

  - 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!

Maximizing Complete Log Likelihood

- Idea: guess the value of λ and iteratively improve the estimate (w.r.t. the complete log likelihood)

Maximizing Complete Log Likelihood

- Initialize λ to some arbitrarily non-zero value
- At step k
  - Let λ^k be the current estimate for λ
  - Compute the expected # times M_A was used from data

$$\left.\begin{aligned}
&E_{\lambda^k}(M = A \mid s_1, s_2, \ldots, s_n) \\
&= \sum_i P_{\lambda^k}(M = A \mid s_i) \\
&= \sum_i \frac{P_{\lambda^k}(M_A, s_i)}{P_{\lambda^k}(s_i)} \\
&= \sum_i \frac{\lambda^k * P_A(s_i)}{\lambda^k * P_A(s_i) + (1-\lambda^k) * P_B(s_i)}
\end{aligned}\right\} \text{E-step}$$

Maximizing Complete Log Likelihood

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

$$\left.\lambda^{k+1} = \frac{E_{\lambda^k}(M = A \mid s_1, s_2, \ldots, s_n)}{n}\right\} \text{M-step}$$

Find λ^{k+1} so that the expected incomplete log likelihood is maximized given λ^k

Maximizing Complete Log Likelihood

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

$$\left.\lambda^{k+1} = \frac{E_{\lambda^k}(M = A \mid s_1, s_2, \ldots, s_n)}{n}\quad\right\}\ \text{M-step}$$

- Terminate if change in log likelihood < δ

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 EM Algorithm

- Initialize the θ's to some arbitrary (non-zero) values $\theta_i^{\,0}$
- 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 θ_i^k (E-step)
  - derive θ_i^{k+1} as an ML estimate using the values of the hidden data computed in the E-step (M-step)

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

Is Incomplete Log Likelihood Maximized?

Theorem:

Let *x* be our incomplete data, *h* our hidden data, and *θ* a parametric model that generates *x* and *h*.

If we choose *θ*' such that

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

then

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

Is Incomplete Log Likelihood Maximized?

Theorem:

Let *x* be our incomplete data, *h* our hidden data, and *θ* a parametric model that generates *x* and *h*.

If we choose *θ*' such that

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

then

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

Lemma:

$$-\sum_x p(x)\log p(x) \le -\sum_x p(x)\log q(x)$$

Proof of EM

$$P_{\theta_i}(x,h) = P_{\theta_i}(x) P_{\theta_i}(h \mid x)$$

$$\log P_{\theta_i}(x,h) = \log(P_{\theta_i}(x) P_{\theta_i}(h \mid x))$$

$$\log P_{\theta_i}(x) = \log P_{\theta_i}(x,h) - \log P_{\theta_i}(h \mid x)$$

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

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

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

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

Proof of EM (Cont')

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

Substitute *θ'* for θ_i in (*), we have:

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

Now, by assumption and the lemma, we have:

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

By the lemma, we have:

$$(-E_{P_{\theta_i P(h|x)}} \log P_{\theta'}(h \mid x)) \ge (-E_{P_{\theta_i P(h|x)}} \log P_{\theta_i}(h \mid x))$$

Adding the two gives $\log P_{\theta'}(x) > \log P_{\theta_i}(x)$

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 θ'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