CS 4650/7650, Lecture 5
Expectation Maximization

Jacob Eisenstein

September 3, 2013

1 Recap

Expectation maximization (EM)

- EM clustering is like K-means, but probabilistic. Unlike K-means, generalizes to tasks beyond clustering.
- EM clustering is like Naive Bayes but without labels. Since we don't know y, we treat it as **missing data**, and maximize the likelihood

$$\begin{aligned}
\log P(\boldsymbol{x}_i; \boldsymbol{\theta}, \phi) =& \log \sum_y P(\boldsymbol{x}_i, y; \boldsymbol{\theta}, \phi)\\
=& \log \sum_y P(\boldsymbol{x}_i | y; \phi) P(y; \boldsymbol{\theta})\\
=& \log \sum_y \frac{q_i(y)}{q_i(y)} P(\boldsymbol{x}_i | y; \phi) P(y; \boldsymbol{\theta})\\
=& \log \sum_y q_i(y) \frac{P(\boldsymbol{x}_i | y_i; \phi) P(y_i; \boldsymbol{\theta})}{q_i(y)}\\
=& \log E_q[\frac{P(\boldsymbol{x}_i | y; \phi) P(y; \boldsymbol{\theta})}{q_i(y)}]\\
\geq& E_q[\log \frac{P(\boldsymbol{x}_i | y; \phi) P(y; \boldsymbol{\theta})}{q_i(y)}]\\
=& E_q[\log P(\boldsymbol{x}_i | y; \phi)] + E_q[\log P(y; \boldsymbol{\theta})] - E_q[\log q_i(y)]
\end{aligned}$$

The last step is an application of Jensen's inequality. We are maximizing a **lower bound** on the joint likelihood. The bound has the following parameters: $q_i(y)$, a distribution over the latent variable y for instance $\langle \boldsymbol{x}_i \rangle$; and $\langle \phi, \theta \rangle$, the parameters of the model, specifically: emission and likelihood parameters.

The EM algorithm:

1. **E-step**: update the "expectations", via the distribution $q(y)$,

$$q_i(y) = P(y|\boldsymbol{x}_i; \boldsymbol{\theta}, \phi) \propto P(\boldsymbol{x}_i|y; \phi)P(y|\boldsymbol{\theta}) \tag{1}$$

   This is also called "imputing" y_i, especially in the stats literature.

2. **M-step**: maximize with respect to the parameters $\langle \phi, \theta \rangle$,

$$\theta_y \propto \sum_i q_i(y) \tag{2}$$

$$\phi_{y,j} \propto \sum_i q_i(y) x_{ij} = E_q[\text{count}(y, j)] \tag{3}$$

3. iterate until convergence.

Guarantees: EM is not guaranteed to find the global optimum of $\log P(\boldsymbol{x})$. This function is non-convex, so it is hard to optimize. EM is guaranteed to converge to a **local optima** – a point which is as good or better than any of its immediate neighbors. But there may be many such points. Note that this is not true in fully supervised learning settings, such as logistic regression or naive bayes. These learning algorithms can find the globally optimal parameters, because $P(\boldsymbol{x}, \boldsymbol{y})$ and $P(\boldsymbol{y}|\boldsymbol{x})$ are convex.

2 Applications of EM

As an example, here's a demo on word sense clustering. We are assuming we know that there are two senses, and that the senses can be distinguished by the contextual information in the document.

2.1 Semi-suprevised learning

The Nigam et al reading shows another application of EM: **semi-supervised learning**.

- In this case, we have labels for some of the instances, $\langle \boldsymbol{x}^{(\ell)}, \boldsymbol{y}^{(\ell)} \rangle$, but not for others, $\langle \boldsymbol{x}^{(u)} \rangle$.
- Can unlabeled data improve learning?

$$\log P(\boldsymbol{x}^{(\ell)}, \boldsymbol{x}^{(u)}, \boldsymbol{y}^{(\ell)}) = \log P(\boldsymbol{x}^{(\ell)}, \boldsymbol{y}^{(\ell)}) + \log P(\boldsymbol{x}^{(u)}) \tag{4}$$

Nigam et al apply this idea to document classification in the classic "20 Newsgroup" dataset.

- We treat the labels of the unlabeled documents as missing data. In the E-step we impute $q(y)$ for the unlabeled documents only.
- The M-step computes estimates of $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ from the sum of the observed counts from $\langle \boldsymbol{x}^{(\ell)}, \boldsymbol{y}^{(\ell)} \rangle$ and the expected counts from $\langle \boldsymbol{x}^{(u)} \rangle$ and $q(\boldsymbol{y})$.
- We can further parametrize this approach by weighting the unlabeled documents by a scalar λ, which is a tuning parameter.

2.2 Multi-component modeling

- One of the classes in 20 newsgroups is `comp.sys.mac.hardware`.
- Suppose that there are two kinds of posts: reviews of new hardware, and question-answer posts about hardware problems.
- The language in these **components** of the mac.hardware class might have little in common.
- So we might do better if we model these components separately.

We can envision a new generative process here:

- For each document i,
  - draw the label $y_i \sim \text{Categorical}(\theta)$
  - draw the component $z_i | y_i \sim \text{Categorical}(\psi_{y_i})$
  - draw the vector of counts $\boldsymbol{x}_i | z_i \sim \text{Multinomial}(\phi_{z_i})$

Our labeled data includes $\langle \boldsymbol{x}_i, y_i \rangle$, but not z_i, so this is another case of missing data.

$$\begin{aligned} P(\boldsymbol{x}_i, y_i) =& \sum_z P(\boldsymbol{x}_i, y_i, z) \\ =& P(\boldsymbol{x}_i | z; \boldsymbol{\phi}) P(z | y_i; \psi) P(y_i; \boldsymbol{\theta}) \end{aligned}$$

Again, we can apply EM

- We need a distribution over the missing data, $q_i(z)$. This is updated during the E-step.
- During the m-step, we compute:

$$\begin{aligned} \psi_{y,z} =& \frac{E_q[\text{count}(y, z)]}{\sum_{z'} E_q[\text{count}(y, z')]} \\ \boldsymbol{\phi}_{j,y,z} =& \frac{E_q[\text{count}(z, j)]}{\sum_{j'} E_q[\text{count}(z, j')]} \end{aligned}$$

- Suppose we assume each class y is associated with K components, $\mathcal{Z}_y$. We can add a constraint to the E-step so that $q_i(z) = 0$ if $z \notin \mathcal{Z}_y \wedge Y_i = y$.
- Notice how EM is a little different in this case from in semi-supervised learning and in clustering. EM is not really an "algorithm" like, say, Quicksort; rather, it's a framework for learning with missing data.

References

[Jae07] T. Florian Jaeger. Optimal language production: Uniform information density, 2007.

[MS99] Christopher D. Manning and Hinrich Schütze. *Foundations of statistical natural language processing*. MIT Press, Cambridge, MA, USA, 1999.