Expectation Maximization (EM)

The Expectation Maximization (EM) algorithm is one approach to unsupervised, semi-supervised, or lightly supervised learning. In this kind of learning either no labels are given (unsupervised), labels are given for only a small fraction of the data (semi-supervised), or incomplete labels are given (lightly supervised). Completely unsupervised learning is believed to take place in human babies. Babies learn to divide continuous speech into words. There is significant experiemental evidence that babies can do this simply by listening to tapes of speech. Computer systems are not yet able to learn to do speech recognition simply by listenting to speech sound. In practice the EM algorithm is most effective for lightly supervised data. For example, the text in closed caption television is a light labeling of the television speech sound. Although the sequence of words is given, the alignment between the words and the sound is not given. The text-to-speech alignment can be infered by EM. Another example of partially labeled data is translation pairs, for example English text paired with a French translation. Translation pairs are used to train machine translation systems. Although the translation is given, the alignment between the words is not (and the word order is different in different languages). Word alignment can be inferred by EM. Although EM is most useful in practice for lightly supervised data, it is more easily formulated for the case of unsupervised learning.

1 Hard EM

Here we assume a model with parameters Θ defining probabilities of the form $P_\Theta(x_1, \ldots, x_N, z_1, \ldots, z_N)$.[1] In a mixture of Gaussian model we have that each x_t is a point in R^D and each z_t is a class label with $z_t \in \{1, \ldots, K\}$. Gaussian mixture models are defined in the next section. In an HMM we have that x_1, …, x_N is a sequence of observations and $z_1, \ldots, z_N$ is a sequence of hidden states. Alternatively, for an HMM we could take each x_t to be an observation sequence (such as a protein amino acid sequence) and each z_t a hidden state sequence (such a sequence of secodary structure classifications). It is more common with HMMs to consider one very long sequence where each x_t is a single observation of the underlying model. For a PCFG we typically have that $x_1, \ldots, x_N$ is a sequence of strings and $z_1, \ldots, z_n$ is a sequence of parse trees where z_t is a parse tree for the string x_t.

We now consider an arbitrary model defining $P_\Theta(x_1, \ldots, x_n, z_1, \ldots, z_n)$. In unsupervised training we are given only $x_1, \ldots, x_n$ and, given only this information, we set the parameter values Θ of the model. Hard EM approximatey

[1] We view $P_\Theta(x, z)$ as a special case with $N = 1$. Alterntatively, one can view $P_\Theta(x_1, \ldots, x_N, z_1, \ldots, z_N)$ as a special case of $P_\Theta(x, z)$ where x is the sequence $x_1, \ldots, x_N$ and z is the sequence $z_1, \ldots, z_N$. All the examples we consider have the form $P_\Theta(x_1, \ldots, x_N, z_1, \ldots, z_N)$ and so it seems natural to keep sequences throughout the formulation.

solves the following optimization problem.

$$\Theta^* = \operatorname*{argmax}_{\Theta} \max_{z_1,\ldots,z_n} P_\Theta(x_1,\ldots,x_n,z_1,\ldots,z_n) \tag{1}$$

A local optimum of (1) can be found by coordinate ascent. We wish to find values of the two "coordinates" Θ and $(z_1,\ldots,z_n)$ maximizing $P_\Theta(x_1,\ldots,x_n,z_1,\ldots z_n)$. In coordinate ascent we alternately optimize each coordinate holding the other coordinates fixed. Hard EM is the following coordinate ascent algorithmm.

1. Initialize Θ (the initiaization is important but application specific).
2. Repeat the following until $P_\Theta(x_1,\ldots x_n,z_1,\ldots z_n)$ converges.
   (a) $(z_1,\ldots z_N) := \operatorname{argmax}_{z_1,\ldots,z_N} P_\Theta(x_1,\ldots,x_N,z_1,\ldots,z_N)$
   (b) $\Theta := \operatorname{argmax}_\Theta P_\Theta(x_1,\ldots,x_N,z_1,\ldots,z_N)$

2 K-Means Clustering as an Example of Hard EM

K-means clustering is a special case of hard EM. In K-means clustering we consider sequences $x_1,\ldots,x_n$ and $z_1,\ldots,z_N$ with $x_t \in R^D$ and $z_t \in \{1,\ldots,K\}$. In other words, z_t is a class label, or cluster label, for the data point x_t. We can define a K-means probability model as follows where $\mathcal{N}(\mu, I)$ denotes the D-dimensional Gaussian distribution with mean $\mu \in R^D$ and with the identity covariance matrix.

$$\begin{aligned}
\Theta &= \langle \mu^1,\ldots,\mu^K\rangle,\ \mu^k \in R^D \\
P_\Theta(x_1,\ldots,x_n,z_1,\ldots z_n) &= \prod_{t=1}^{N} P_\Theta(z_t)P_\Theta(x_t|z_t) \\
&= \prod_{t=1}^{N} \frac{1}{K}\mathcal{N}(\mu^{z_t}, I)(x_t)
\end{aligned}$$

We now consider the optimization problem defined by (1) for this model. For this model one can show that (1) is equivalent to the following.

$$\left(\mu^1,\ldots,\mu^K\right)^* = \operatorname*{argmin}_{\mu^1,\ldots,\mu^k} \min_{z_1,\ldots,z_n} \sum_{t=1}^{N} ||\mu^{z_t} - x_t||^2 \tag{2}$$

The optimization problem (2) defines K-means clustering (under quadratic distortion). This problem is nonconvex and in fact is NP-hard (worse than nonconvex). The K means algorithm is coordinate descent applied to this objective and is equivalent to hard EM under tha above probability model. The K-means clustering algorithm can be written as follows where we specify a typical initialization step.

1. Initialize μ^z to be equal to a randomly selected point x_t.
2. Repeat the following until $(z_1, \ldots z_n)$ stops changing.
   (a) $z_t := \text{argmin}_z \ ||\mu^z - x_t||^2$
   (b) $N_z := |\{t : \ z_t = z\}|$
   (c) $\mu^z := \frac{1}{N_z} \sum_{t: \ z_t = z} x_t$

In words, the K-means algorithm first assigns a class center μ^z for each class z. It then repeatedly classifies each point x_t as belonging to the class whose center is nearest x_t and then recomputes the class centers to be the mean of the point placed in that class. Because it is a coordinate descent algorithm for (2), the sum of squares of the difference between each point and its class center is reduced by each update. This implies that the classification must eventually stabilize. The procedure terminates when the class labels stop changing.

3 Hard EM for Mixtures of Gaussians

Again we consider sequences $x_1, \ldots, x_n$ and $z_1, \ldots, z_N$ with $x_t \in R^D$ and $z_t \in \{1, \ldots, K\}$. Now however we consider a probability model that is a mixture of Gaussians including mixture weights and covariance matrices.

$$\Theta = \langle \pi^1, \ldots, \pi^K, \mu^1, \ldots, \mu^K, \Sigma^1, \ldots, \Sigma^K \rangle$$

$$\begin{aligned} P_\Theta(x_1, \ldots, x_n, z_1, \ldots z_n) &= \prod_{t=1}^{N} P_\Theta(z_t) P_\Theta(x_t | z_t) \\ &= \prod_{t=1}^{N} \pi^{z_t} \mathcal{N}(\mu^{z_t}, \Sigma^{z_t})(x_t) \end{aligned}$$

Again we can consider the optimization problem defined by (1) for this model. It can now be shown that step 2b of the hard EM is equivalent to the following.

$$\begin{aligned} N^k &= |\{t:\ z_t = k\}| \\ \pi^k &= N^k/N \\ \mu^k &= \frac{1}{N^k}\sum_{t:\ z_t^*=k} x_t \\ \Sigma^k &= \frac{1}{N^k}\sum_{t:\ z_t^*=k} (x_t-\mu^k)(x_t-\mu^k)^T \qquad (3) \end{aligned}$$

4 An Informal Statement of General Soft EM

Again we assume a model with parameters Θ defining probabilities of the form $P_\Theta(x_1, \ldots, x_N, z_1, \ldots, z_N)$. Again we are given only $x_1, \ldots, x_n$ and, given only this information, we set the parameter values Θ of the model. Soft EM approximatey solves the following optimization problem.

$$\begin{aligned} \Theta^* &= \underset{\Theta}{\operatorname{argmax}} \sum_{z_1,\ldots,z_n} P_\Theta(x_1,\ldots,x_n,z_1,\ldots,z_n) \qquad (4) \\ &= P_\Theta(x_1,\ldots,x_N) \qquad (5) \end{aligned}$$

One should compare (4) with (1) and note that the objective in hard EM is different from the objective in soft EM. Which objective is more appropriate can depend on how the model is to be used. If the model is used to infer the most likely value of $z_1, \ldots, z_N$, as in speech recognition, then (1) seems more appropriate than (4). But if the model is to be used for computing $P_\Theta(x_1, \ldots, x_N)$ then (4) seems more appropriate. Soft EM can be described informally as the following iterative improvement algorithmm for the objective given in (4).

1. Initialize Θ (the initiaization is important but application specific).
2. Repeat the following until $P_\Theta(x_1, \ldots x_n, z_1, \ldots z_n)$ converges.
   (a) $\rho(z_1, \ldots, z_N) := P_\Theta(z_1, \ldots, z_N \mid x_1, \ldots, x_N)$ (The E step.)
   (b) $\Theta :=$ The best fit of Θ to $x_1, \ldots, x_N$ and the distribution ρ on $z_1, \ldots, z_N$. (The M step.)

5 Soft EM for Mixtures of Gaussians

Again consider the mixture of Gaussian model defined in section 3. We can implement step 2a (the E step) by computing probabilities (or weights) p_t^k as follows.

$$\begin{aligned} p^k &= P_{\Theta_w}(z_t = k | x_t) \\ \\ &= \frac{\pi_w^k \mathcal{N}\left(\mu_w^k, \Sigma_w^k\right)(x_t)}{\sum_{k=1}^K \pi_w^k \mathcal{N}\left(\mu_w^k, \Sigma_w^k\right)(x_t)} \end{aligned}$$

Given the numbers (the matrix) p_t^k, step 2b is then implemented as follows.

$$\begin{aligned} N^k &= \sum_{t=1}^N p_t^k \\ \pi^k &= \frac{N^k}{N} \\ \mu_{w+1}^k &= \frac{1}{N^k} \sum_{t=1}^N p_t^k x_t \\ \Sigma_{w+1}^k &= \frac{1}{N^k} \sum_{t=1}^N p_t^k (x_t - \mu^k)(x_t - \mu^k)^T \end{aligned}$$

It is instructive to compare these update equations with the update equations of section 3.

6 Soft EM for HMMs

EM for HMMs is called the Baum-Welch algorithm. The Baum-Welch algorithm predates the general formlation of EM — most special cases of EM predate the general formulation. Recall that in an HMM we have $x_t \in \{1, \ldots, O\}$ and $z_t \in \{1, \ldots, S\}$. An HMM is defined as follows where π is a S-dimensional vector, A is a $S \times S$ hidden state transition probability matrix, and C is an $O \times S$ emission probability matrix.

$$\begin{aligned} \Theta &= \langle \pi, A, C \rangle \\ \\ P_\Theta(z_1, \ldots, z_N, x_1, \ldots, x_N) &= \pi_{z_1} \left(\prod_{t=1}^{N-1} A_{z_{t+1}, z_t} \right) \left(\prod_{t=1}^N C_{x_t, z_t} \right) \end{aligned}$$

Here we will use superscript indeces for model generation so that we are computng Θ^{w+1} from Θ^w. Given a model Θ^w, and a sequence $x_1, \ldots, x_n$, we can use the forward-backward procedure to compute the following two matrices.

$$\begin{aligned} F_{i,t} &= P_{\Theta^w}(x_1, \ldots, x_{t-1}, z_t = i) \\ B_{i,t} &= P_{\Theta^w}(x_t, \ldots, x_N \mid z_t = i) \end{aligned}$$

We will also use the following matrix computable from F and B.

$$\begin{aligned} P_{i,t} &= P_{\Theta^w}(z_t = i \mid x_t, \ldots, x_N) \\ &= \frac{F_{i,t} B_{i,t}}{P(x_1, \ldots, x_N)} \\ P(x_1, \ldots x_n) &= \sum_{i=1}^{S} F_{i,t} B_{i,t} \end{aligned}$$

The model Θ^{w+1} is then computed as follows.

$$\begin{aligned} \pi_i^{w+1} &= P_{i,1} \\ C_{k,i}^{w+1} &= \frac{\sum_{t:\, x_t = k} P_{i,t}}{\sum_{t=1}^{N} P_{i,t}} \\ A_{i,j}^{w+1} &= \frac{\sum_{t=1}^{N-1} P_{\Theta^w}(z_t = j \wedge z_{t+1} = i \mid x_1, \ldots, x_N)}{\sum_{t=1}^{N-1} P_{j,t}} \\ &= \frac{\sum_{t=1}^{N-1} F_{j,t} C_{x_t,j}^{w} A_{i,j}^{w} B_{j,t}}{P(x_1, \ldots, x_N) \sum_{t=1}^{N-1} P_{j,t}} \end{aligned}$$

Intuition into these equations can be gained by thinking about the distribution over $z_1, \ldots, z_N$ as training data. Each entry in the model Θ^{w+1} can be viewed as a conditional probability $P(\Phi|\Psi)$ which is being set to a "count" of $\Phi \wedge \Psi$ divided by a "count" of Ψ. In each case the count is an *expected* number of occurances. Hence the term "expectation" for the E-step of EM.

7 A Formal Treatment of General Soft EM

Here we let x abbreviate $(x_1, \ldots, x_n)$ and let z abbreviate $(z_1, \ldots, z_n)$. More generally we can consider any model $P_\Theta(x, z)$ on a pair of variables x and z. Soft EM is the following iterative improvement algorithmm for the objective given in (4).

1. Initialize Θ (the initiaization is important but application specific).
2. Repeat the following until $P_\Theta(x, z)$ converges.
   (a) $\rho(z) \; := \; P_\Theta(z \mid x)$
   (b) $\Theta \; := \; \mathrm{argmax}_\Theta \, \mathrm{E}_{z\sim\rho} \left[\ln P_\Theta(x, z)\right]$

First we given an intuition as to why step 2b in the formal soft EM algorithm corresponds to step 2b in the informal version. Step 2b of the informal version can now be stated as "fit Θ to x and ρ". Intuitively, we can think of fitting Θ to x and ρ as fitting Θ to a very large sample $(x, z_1), \ldots, (x, z_M)$ where each z_i is drawn at random from ρ. For such a sample we have the following.

$$\begin{aligned}
P_\Theta((x, z_1), \ldots, (x, z_M)) &= \prod_{i=1}^{m} P_\Theta(x, z_i) \\
\ln P_\Theta((x, z_1), \ldots, (x, z_M)) &= \sum_{i=1}^{m} \ln P_\Theta(x, z_i) \\
&= \sum_z \mathrm{count}(z) \ln P(x, z) \\
\frac{1}{M} \ln P_\Theta((x, z_1), \ldots, (x, z_M)) &= \frac{\mathrm{count}(z)}{M} \ln P(x, z) \\
&\approx \sum_z \rho(z) \ln P(x, z) \\
&= \mathrm{E}_{z\sim\rho} \left[\ln P(x, z)\right] \\
\operatorname*{argmax}_\Theta P_\Theta((x, z_1), \ldots, (x, z_M)) &\approx \operatorname*{argmax}_\Theta \mathrm{E}\left[z \sim \rho\right] \ln P_\Theta(x, z)
\end{aligned}$$

So we should think of the distribution ρ as providing a "viritual sample" to which we fit Θ in step 2b.

We now show that the update 2b improves the objective function (4) unless we are already at a local optimum. Here we consider an update starting at the parameter vector Θ and ending in the parameter vector $\Theta + \epsilon$. Step 2b can be defined as follows.

$$\epsilon = \operatorname*{argmax}_\epsilon Q(\Theta + \epsilon) \tag{6}$$

$$Q(\Psi) = \mathrm{E}_{z\sim\rho} \left[\ln P_\Psi(x, z)\right] \tag{7}$$

$$\rho(z) = P_\Theta(z|x) \tag{8}$$

We now write $P_\Theta(\cdot \mid x)$ for the distribution on z defined by $P_\Theta(z|x)$ and show the following.

$$\begin{aligned} \ln P_{\Theta+\epsilon}(x) &= \ln P_\Theta(x) + Q(\Theta+\epsilon) - Q(\Theta) + KL(P_\Theta(\cdot|x), P_{\Theta+\epsilon}(\cdot|x)) \quad (9) \\ &\geq \ln P_\Theta(x) + Q(\Theta+\epsilon) - Q(\Theta) \quad (10) \end{aligned}$$

Proof:

$$\begin{aligned} Q(\Theta+\epsilon) - Q(\Theta) &= \mathrm{E}_{z\sim\rho}\left[\ln \frac{P_{\Theta+\epsilon}(x,z)}{P_\Theta(x,z)}\right] \\ &= \mathrm{E}_{z\sim\rho}\left[\ln \frac{P_{\Theta+\epsilon}(x)P_{\Theta+\epsilon}(z|x)}{P_\Theta(x)P_\Theta(z|x)}\right] \\ &= \mathrm{E}_{z\sim\rho}\left[\ln \frac{P_{\Theta+\epsilon}(X)}{P_\Theta(x)}\right] + \mathrm{E}_{z\sim\rho}\left[\ln \frac{P_{\Theta+\epsilon}(z|x)}{P_\Theta(z|x)}\right] \\ &= \ln \frac{P_{\Theta+\epsilon}(x)}{P_\Theta(x)} - KL(P_\Theta(\cdot|x), P_{\Theta+\epsilon}(\cdot|x)) \end{aligned}$$

Formula (10) gives a differentiable lower bound on an differentiable objective function to be maximized. Furthermore, the lower bound equals the objective function at the "current point" Θ. We can now make a very general observation. For any differentiable lower bound on a differentiable objective function where the bound equals the objective at the current point, and where the gradient of the objective is nonzero at the current point, maximizing the lower bound will strictly improve the objective. To see this note that at the current point the gradient of the bound must equal the gradient of the objective. Hence the lower bound itself can be strictly improved. At any point where the lower bound is larger, the objective must also be larger. The EM update corresponds to maximizing the lower bound in (10).

8 Problems

1. Suppose that we want to classify $x \in \mathcal{X}$ into one of K classes using a single feature vector $\Phi(x)$ where each feature is either 0 or 1, i.e., for each feature i and $x \in \mathcal{X}$ we have $\Phi_i(x) \in \{0, 1\}$. The naive Bayes model is a generative model for generating pairs $\langle x, y\rangle$ with $x \in \mathcal{X}$ and z a class label, i.e., $z \in \{1, \ldots, K\}$. The naive Bayes model can be defined by the following equations.

$$\begin{aligned} \beta_j &= P(z = j) \\ \beta_{i,j} &= P(\Phi_i(x) = 1 \mid z = j) \end{aligned}$$

$$P_\beta(x, z) = \beta_z \prod_i \begin{cases} \beta_{i,z} & \text{if } \Phi_i(x) = 1 \\ (1 - \beta_{i,z}) & \text{if } \Phi_i(x) = 0 \end{cases}$$

Suppose that we have a sample S of unlabeled values $x_1, \ldots, x_T$ from $\mathcal{X}$.

a. Give equations for $P_\beta(z_t = z|x_t)$.

b. Specify the hard EM algorithm for this model. Give an initialization that you think is reasable.

c. Specify the soft EM algorithm for this model. Again given an intialization that you think is reasonable.

2. Consider a Bayesian network with structure $X \leftarrow Y \rightarrow Z$ where each of X, Y, and Z take values from the finite sets $\mathcal{X}$, $\mathcal{Y}$ and $\mathcal{Z}$ respectively. This network has the following parameters where $x \in \mathcal{X}$, $y \in \mathcal{Y}$ and $z \in \mathcal{Z}$.

$$\begin{aligned}\Pi_y &= P(Y = y) \\ R_{x,y} &= P(X = x \mid Y = y) \\ L_{z,y} &= P(Z = z \mid Y = y)\end{aligned}$$

Give both the hard and soft EM algorithms for this model.

3. Give the hard EM algorithm for PCFGs (using the Viterbi parsing algorithm) and the soft EM algorithm fr PCFS (using the inside-outside algorithm).

4. Suppose we initialize the PCFG to be deterministic in the sense that for any word string there is at most one parse of that string with nonzero probability and suppose that each x_h does have a parse under the initial grammar. How rapidly does EM converge in this case? What does it converge to? Justify your answer.

5. Let γ be a distribution on $\{1, \ldots, S\}$. In other words we have the following.

$$\begin{aligned}\gamma_i &\geq 0 \\ \sum_{i=1}^{S} \gamma_i &= 1 \qquad (11)\end{aligned}$$

Suppose that we want to fit a distribution (or model) π to the distribution γ. Intuitively, the best fit is $\pi = \gamma$. But the general EM update fits using a formula similar to the following.

$$\pi^* = \operatorname*{argmax}_{\pi} \mathrm{E}_{j \sim \gamma} [\ln \pi_j]$$

Show that $\pi^* = \gamma$. Hint: express the problem as a minimization of cross entropy and use the fact that cross entropy is minimized when the distributions are equal, or perhaps better known, that $KL(P, Q) \geq 0$ and $KL(P, Q) = 0$ only if $P = Q$. Alternatively, you can use Lagrange multipliers and KKT conditions.

6. Let ρ be a distribution (density) on the real numbers R. Suppose that we want to fit a Gaussian with mean μ and standard deviation σ to the distribution ρ using the following "distribution fit equation".

$$\begin{aligned} \mu^*, \Sigma^* &= \underset{\mu,\Sigma}{\operatorname{argmax}} \, \mathrm{E}_{x\sim\rho} \left[\ln \mathcal{N}(\mu, \sigma)(x)\right] \\ \mathcal{N}(\mu, \sigma)(x) &= \frac{1}{\sqrt{2\pi}\sigma} \exp\left(\frac{-(x-\mu)^2}{2\sigma^2}\right) \end{aligned}$$

Show that $\mu^* = \mu_\rho = \mathrm{E}_{x\sim\rho}\left[x\right]$ and $\sigma^{*2} = \sigma_\rho^2 = \mathrm{E}_{x\sim\rho}\left[(x - \mu_\rho)^2\right]$.