Machine Learning Course - CS-433

Expectation-Maximization Algorithm

Nov 22, 2018

minor changes by Martin Jaggi 2016

minor changes by Martin Jaggi 2017

changes by Ruediger Urbanke 2018

Last updated: November 22, 2018

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Motivation

In our last lecture we considered the Gaussian mixture model. To recall, in this model we assume that the data vectors $\{\mathbf{x}_n\}$ are iid samples from a density that is the mixture (weighted sum) of K D-dimensional Gaussians. This density is hence characterized by the following parameters: $\{\boldsymbol{\mu}_k\}_{k=1}^K$, the means, $\{\boldsymbol{\Sigma}_k\}_{k=1}^K$, the covariance matrices, and $\{\pi_k\}_{k=1}^K$, the weights of the individual Gaussians. Let

$$\boldsymbol{\theta} = \{\{\boldsymbol{\mu}_k\}_{k=1}^K, \{\boldsymbol{\Sigma}_k\}_{k=1}^K, \{\pi_k\}_{k=1}^K\}.$$

Assume that we have given the training data $S_{\text{train}} = \{\mathbf{x}_n\}$ and that we want to find the $\boldsymbol{\theta}$ that maximize the likelihood. This gave rise to the optimization problem

$$\max_{\boldsymbol{\theta}} \ \mathcal{L}(\boldsymbol{\theta}) := \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k).$$

Note that this cost function is not easy to minimize due to the logarithm of the sum that it contains.
The expectation-maximization (EM) algorithm provides an elegant and general method to tackle such problems. It uses an iterative two-step procedure where each step is typically "easy." Similar to the iterative algorithm we saw for the K-means problem, this algorithm is guaranteed to improve the cost function at every step and will converge but is not guaranteed to converge to the optimum solution.

Summary

The EM algorithm is a very general algorithm. We start by explaining it by means of our Gaussian mixture problem. In this case the algorithm is somewhat reminiscent of how we dualized the cost function involving the hinge loss and the procedure is easy to explain. At the end we then briefly explain the general idea of the EM algorithm.

Derivation

Recall that we want to maximize

$$\sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n \,|\, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

over all choices of $\boldsymbol{\theta} = \{\{\boldsymbol{\mu}_k\}_{k=1}^K, \{\boldsymbol{\Sigma}_k\}_{k=1}^K, \{\pi_k\}_{k=1}^K\}$. Since we will consider an iterative algorithm, where we will update $\boldsymbol{\theta}$ at each step, let us denote the set of parameters at step t by

$$\boldsymbol{\theta}^{(t)} = \{\{\boldsymbol{\mu}_k^{(t)}\}_{k=1}^K, \{\boldsymbol{\Sigma}_k^{(t)}\}_{k=1}^K, \{\pi_k^{(t)}\}_{k=1}^K\}.$$

Assume that we made some initial choice for $\boldsymbol{\theta}^{(0)}$ and assume that we already did t steps of the algorithm, i.e., the current set of parameters is $\boldsymbol{\theta}^{(t)}$. We are trying to find an even better set of parameters, called $\boldsymbol{\theta}^{(t+1)}$.
Consider any probability distribution $q_n^{(t)}$

$$q_{nk}^{(t)} \geq 0, \quad \sum_{k=1}^{K} q_{nk}^{(t)} = 1.$$

Then, due to the concavity of the function $\ln(\cdot)$ we have

$$\log \sum_{k=1}^{K} \pi_k^{(t)} \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_k^{(t)}, \boldsymbol{\Sigma}_k^{(t)}) \geq \sum_{k=1}^{K} q_{nk}^{(t)} \log \frac{\pi_k^{(t)} \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_k^{(t)}, \boldsymbol{\Sigma}_k^{(t)})}{q_{nk}^{(t)}}.$$

We get equality if each term inside the log is equal, i.e., if

$$q_{nk}^{(t)} \sim \pi_k^{(t)} \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_k^{(t)}, \boldsymbol{\Sigma}_k^{(t)})$$

so that

$$q_{nk}^{(t)} = \frac{\pi_k^{(t)} \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k^{(t)}, \boldsymbol{\Sigma}_k^{(t)})}{\sum_{k=1}^{K} \pi_k^{(t)} \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k^{(t)}, \boldsymbol{\Sigma}_k^{(t)})}.$$

Assume that we have made this choice for the probability distribution $q_n^{(t)}$ for each of the n terms. Then our overall objective function at the parameter $\boldsymbol{\theta}^{(t)}$ is equal to

$$\prod_{n=1}^{N} \sum_{k=1}^{K} q_{nk}^{(t)} \log \frac{\pi_k^{(t)} \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_k^{(t)}, \boldsymbol{\Sigma}_k^{(t)})}{q_{nk}^{(t)}}.$$

Now freeze the $q_n^{(t)}$ but think of $\boldsymbol{\theta}$ as a variable, i.e., consider

$$\prod_{n=1}^{N} \sum_{k=1}^{K} q_{nk}^{(t)} \log \frac{\pi_k \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{q_{nk}^{(t)}}.$$

Note that by our derivation this function is in general not equal to the original cost function but it is always a lower bound and it is equal to the original cost function for $\boldsymbol{\theta} = \boldsymbol{\theta}^{(t)}$. Since we want to maximize the original cost function it makes sense to maximize this lower bound. Let us do this and call the maximizing parameter $\boldsymbol{\theta}^{(t+1)}$.

This leads us to maximizing the expression

$$\sum_{n=1}^{N}\sum_{k=1}^{K} q_{nk}^{(t)}\left[\log \pi_k - \log q_{nk}^{(t)} + \log \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)\right] \quad (1)$$

over the set of parameters $\boldsymbol{\theta}$.
In the above optimization problem the $\{\pi_k\}$ are constrained to be non-negative and sum up to 1. Let us hence add the term $\lambda \sum_{k=1}^{K} \pi_k$ to the optimization problem to turn this into an unconstrained problem. Note that we do not add any terms to enfore the positivity of the $\{\pi_k\}$. As we will see, the unconstrained problem will automatically return non-negative quantities and hence such a constraint is not needed. In summary, we want to maximize

$$\sum_{n=1}^{N}\sum_{k=1}^{K} q_{nk}^{(t)}\left[\log \pi_k - \log q_{nk}^{(t)} + \log \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)\right] + \lambda \sum_{k=1}^{K} \pi_k,$$

over all choices of $\boldsymbol{\theta}$ and λ.
Differentiating wrt π_k and setting the result to 0 yields

$$\sum_{n=1}^{N} q_{nk}^{(t)} \frac{1}{\pi_k} + \lambda = 0,$$

which has the solution

$$\pi_k = -\frac{1}{\lambda}\sum_{n=1}^{N} q_{nk}^{(t)}.$$

Now we can choose λ so as to ensure a proper normalization.

This leads to $\lambda = -N$. Hece we get

$$\pi_k^{(t+1)} := \frac{1}{N} \sum_{n=1}^{N} q_{nk}^{(t)}.$$

Note that one term $\log \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ has the form

$$-\frac{D}{2} \log(2\pi) + \frac{1}{2} \log |\boldsymbol{\Sigma}^{-1}| - \frac{1}{2} (\mathbf{x} - \boldsymbol{\mu}_k)^\top \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}_k).$$

where we used the fact that $|\boldsymbol{\Sigma}| = 1/|\boldsymbol{\Sigma}^{-1}|$.
Differentiating the cost function wrt $\boldsymbol{\mu}_k$ and setting the result to 0 yields

$$\sum_{n=1}^{N} q_{nk}^{(t)} \boldsymbol{\Sigma}^{-1} (\mathbf{x}_n - \boldsymbol{\mu}_k) = 0.$$

Multiplying this equation by $\boldsymbol{\Sigma}$ from the left and solving for $\boldsymbol{\mu}_k$ we get

$$\boldsymbol{\mu}_k^{(t+1)} := \frac{\sum_n q_{nk}^{(t)} \mathbf{x}_n}{\sum_n q_{nk}^{(t)}}.$$

Finally, taking the derivative wrt $\boldsymbol{\Sigma}_k^{-1}$ and setting the result to 0 we get

$$\sum_{n=1}^{N} q_{nk}^{(t)} \frac{1}{2} \boldsymbol{\Sigma}_k^\top - \frac{1}{2} \sum_{n=1}^{N} q_{nk}^{(t)} (\mathbf{x}_n - \boldsymbol{\mu}_k)(\mathbf{x}_n - \boldsymbol{\mu}_k)^\top = 0.$$

This has the solution

$$\boldsymbol{\Sigma}_k^{(t+1)} := \frac{\sum_n q_{nk}^{(t)} (\mathbf{x}_n - \boldsymbol{\mu}_k^{(t+1)})(\mathbf{x}_n - \boldsymbol{\mu}_k^{(t+1)})^\top}{\sum_n q_{nk}^{(t)}}$$

In the last step we have used the fact that for an invertible matrix $\mathbf{A}$ we have

$$\frac{\partial \log |\mathbf{A}|}{\partial \mathbf{A}_{i,j}} = (\mathbf{A}^{-T})_{i,j}.$$

Since for us the matrix $\mathbf{\Sigma}_k$ is symmetric, the transpose can be dropped. The above formula can be seen as follows. Recall that

$$|\mathbf{A}| = \sum_{i,k} \mathbf{A}_{i,k} C_{i,k},$$
$$\mathbf{A}^{-1}_{i,j} = \frac{C_{j,i}}{|\mathbf{A}|},$$

where $C_{i,j}$ is the co-factor of the matrix $\mathbf{A}$ and is given by $(-1)^{i+j}$ times the determinant of the matrix that you get if you remove row i and column j from $\mathbf{A}$.
Therefore,

$$\frac{\partial \log |\mathbf{A}|}{\partial \mathbf{A}_{i,j}} = \frac{\partial \log \sum_{i,k} \mathbf{A}_{i,k} C_{i,k}}{\partial \mathbf{A}_{i,j}} = \frac{C_{i,j}}{\sum_{i,k} \mathbf{A}_{i,k} C_{i,k}}$$
$$= \frac{C_{i,j}}{|\mathbf{A}|} = (\mathbf{A}^{-1})_{j,i} = (\mathbf{A}^{-T})_{i,j}$$

In the parlance of this algorithm, the first step where we compute the probability distributions q_n is called the *expectation* step, whereas the second step is called the *maximization* step. Let us summarize this algorithm.

Summary of EM for GMM

Initialize $\boldsymbol{\mu}^{(0)}, \boldsymbol{\Sigma}^{(0)}, \boldsymbol{\pi}^{(0)}$ and iterate between the E and M step, until $\mathcal{L}(\boldsymbol{\theta})$ stabilizes.

1. *E-step:*

   a) Compute the marginal likelihood (cost).

$$\mathcal{L}(\boldsymbol{\theta}^{(t)}) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k^{(t)} \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_k^{(t)}, \boldsymbol{\Sigma}_k^{(t)})$$

   b) Compute assignments $q_{nk}^{(t)}$:

$$q_{nk}^{(t)} := \frac{\pi_k^{(t)} \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_k^{(t)}, \boldsymbol{\Sigma}_k^{(t)})}{\sum_{k=1}^{K} \pi_k^{(t)} \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_k^{(t)}, \boldsymbol{\Sigma}_k^{(t)})}$$

2. *M-step*:

   a) Update $\boldsymbol{\mu}_k^{(t+1)}, \boldsymbol{\Sigma}_k^{(t+1)}, \pi_k^{(t+1)}$.

$$\boldsymbol{\mu}_k^{(t+1)} := \frac{\sum_n q_{nk}^{(t)} \mathbf{x}_n}{\sum_n q_{nk}^{(t)}}$$

$$\boldsymbol{\Sigma}_k^{(t+1)} := \frac{\sum_n q_{nk}^{(t)} (\mathbf{x}_n - \boldsymbol{\mu}_k^{(t+1)})(\mathbf{x}_n - \boldsymbol{\mu}_k^{(t+1)})^\top}{\sum_n q_{nk}^{(t)}}$$

$$\pi_k^{(t+1)} := \frac{1}{N} \sum_n q_{nk}^{(t)}$$

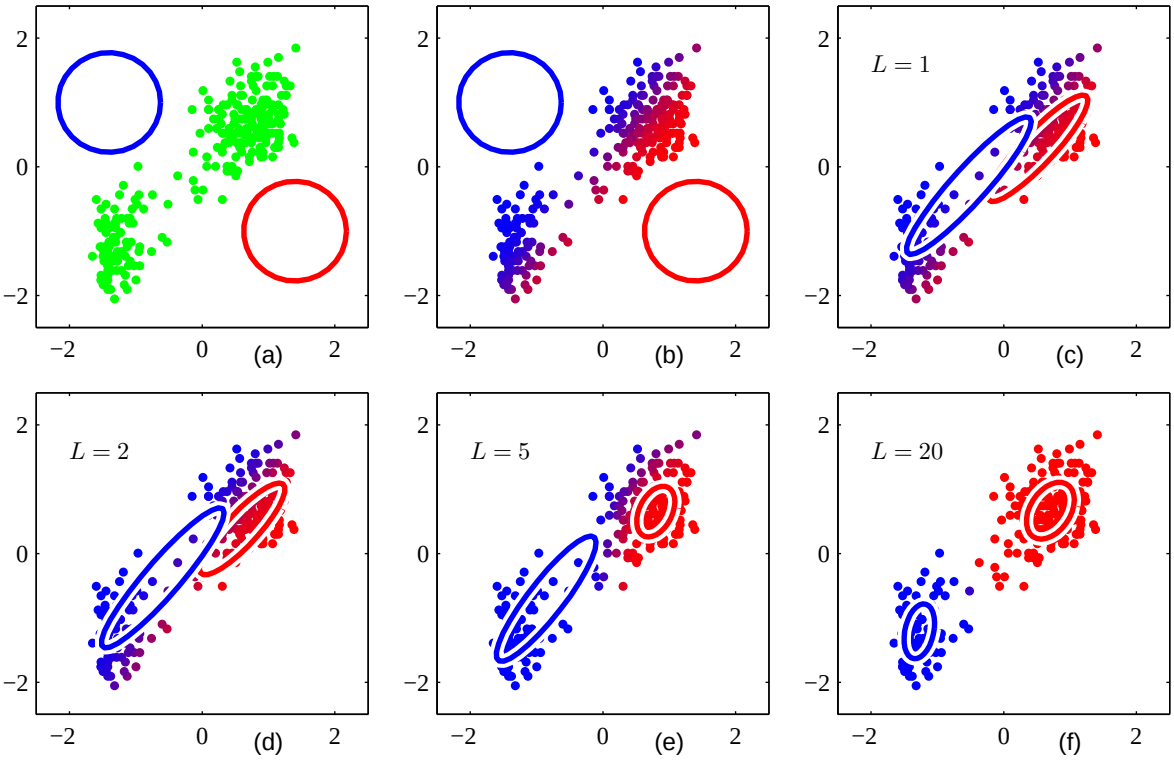

Figure 1: EM algorithm for GMM

Posterior distribution

Recall our original model. We assumed that our data points are iid from a mixture model with k Gaussian components,

$$p(\mathbf{x}_n|\boldsymbol{\theta}) = \sum_{k=1}^{K} p(z_n = k \mid \boldsymbol{\theta}) p(\mathbf{x}_n \mid z_n = k, \boldsymbol{\theta})$$
$$= \sum_{k=1}^{K} \pi_k \, \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k).$$

Recall that the z_n indicates from what Gaussian the sample $\mathbf{x}_n$ stems.
Given the sample $\mathbf{x}_n$ we can compute the posterior of z_n.

We get

$$\begin{aligned} p(z_n = k \mid \mathbf{x}_n, \boldsymbol{\theta}) &= \frac{p(z_n = k, \mathbf{x}_n, \boldsymbol{\theta})}{p(\mathbf{x}_n, \boldsymbol{\theta})} = \frac{p(z_n = k, \mathbf{x}_n \mid \boldsymbol{\theta})}{p(\mathbf{x}_n \mid \boldsymbol{\theta})} \\ &= \frac{p(z_n = k \mid \boldsymbol{\theta}) p(\mathbf{x}_n \mid z_n = k, \boldsymbol{\theta})}{p(\mathbf{x}_n \mid \boldsymbol{\theta})} \\ &= \frac{p(z_n = k \mid \boldsymbol{\theta}) p(\mathbf{x}_n \mid z_n = k, \boldsymbol{\theta})}{\sum_{j=1}^{K} p(z_n = j \mid \boldsymbol{\theta}) p(\mathbf{x}_n \mid z_n = j, \boldsymbol{\theta})} \\ &= \frac{\pi_k \mathcal{N}(\mathbf{x}_n \mid \mu_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}_n \mid \mu_j, \boldsymbol{\Sigma}_j)}. \end{aligned}$$

From this we see that the variables q_{nk} are just the posteriors $p(z_n = k \mid \mathbf{x}_n, \boldsymbol{\theta})$. So in the step where we computed the distributions q_n, what we really did was to compute the posterior distributions of the assignment variables z_n.
We can now explain the reason why the first step is called the *expectation step*. Assume that a genie had told us for each sample x_n the probability that it came from a particular component k (these are exactly the q_{nk} quantities). Then, given the parameter $\boldsymbol{\theta}$, let us compute the *expected* value of the log-likelihood

$$\begin{aligned} \log p(x_n, z_n = k \mid \boldsymbol{\theta}) &= \log(p(z_n = k \mid \boldsymbol{\theta}) p(x_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)) \\ &= \log(\pi_k \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)), \end{aligned}$$

where the expectation is over the distribution of z_n, which, by assumption, we have given. We get

$$\sum_{k=1}^{K} q_{nk} \log \pi_k \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k).$$

If we now sum this over all samples $\mathbf{x}_n$ and compare this to (1) we see that these two expression are the same except for the terms $-q_{nk}\log(q_{nk}$ in (1). But these terms are just constants for the maximization that follows and so do not play a role. Hence, we have now an alternative probabilistic explanation of the EM algorithm, and in particular this explanation makes it clear why the first step is called the expectation step.

With this interpretation it is then clear that the EM algorithm can be compactly written as follows:

$$\boldsymbol{\theta}^{(t+1)} := \arg\max_{\boldsymbol{\theta}} \sum_{n=1}^{N} \mathbb{E}_{p(z_n|\mathbf{x}_n,\boldsymbol{\theta}^{(t)})}\big[\log p(\mathbf{x}_n, z_n|\boldsymbol{\theta})\big]$$

EM in general

In the above derivation we have assumed a specific form for $p(\mathbf{x}_n, z_n|\boldsymbol{\theta})$. In particular we assumed that the components are Gaussian. But the underlying idea can be applied more generally.

In our original derivation we motivated the introduction of the (posterior distributions) z_n in terms of a simple lower bound. In the previous paragraph we have seen that there is an alternative probabilistic interpretation. Namely, we can interpret the assignments z_n as "missing" data. If we were given this missing data our optimization would be simple. The idea then of the EM algorithm is to explicitly add this missing data to faciliate the optimization. This is the maximization step. It then averages out this "unobserved" data. This is the expectation step.