NOTES FOR COS424, 25 MARCH 2008

NATHAN SAVIR AND MASON SIMON

Abstract. We review the basics of Markov models and hidden Markov models (HMMs) from last class. We discuss the two types of HMMs, which are analogous to naive Bayes and mixture models, respectively. We then begin computations regarding the efficient implementation of the EM algorithm as it applies to HMMs.

1. Markov Models

Recall from last time: a Markov model is a sequence of variables in which each one is probabilistically dependent on the previous one(s).

model 1.jpg

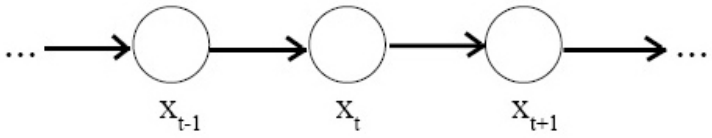

Figure 1. sequential dependence of random variables

We can see the following fundamental factorization property of the joint distribution from our model:

$$p(x_{t-1}, x_t, x_{t+1}) = p(x_{t-1})p(x_t|x_{t-1})p(x_{t+1}|x_t). \tag{1.1}$$

Let's consider the conditional joint distribution $p(x_{t-1}, x_{t+1}|x_t)$. Using the chain rule and (1.1), we have

$$\begin{aligned} p(x_{t-1}, x_{t+1}|x_t) &= \frac{p(x_{t-1})p(x_t|x_{t-1})p(x_{t+1}|x_t)}{p(x_t)} \\ &= \frac{p(x_{t-1}, x_t)p(x_{t+1}|x_t)}{p(x_t)} \\ &= p(x_{t-1}|x_t)p(x_{t+1}|x_t). \end{aligned} \tag{1.2}$$

The two conditionals are independent! In plain English, this can be read as "the past is independent of the future, given the present." This is a fundamental assumption of the Markov model.

Date: 26 March 2008.

2. Hidden Markov Models

Recall that the generic HMM is represented by a graphical model like this:

model 2.jpg

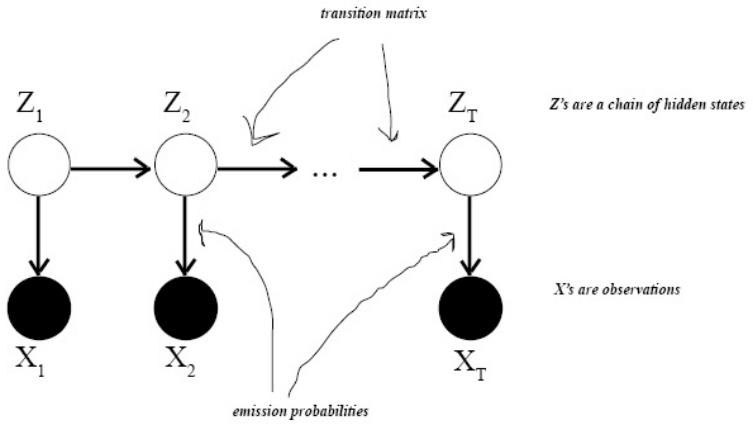

Figure 2. HMM

This is fundamentally similar to a mixture model, except that the generating distribution for the data changes at each iteration and is dependent on the previous distribution. We see that this model can be used in two ways, the first of which should be reminiscent of naive Bayes, and the second of which is more like the mixture model.

(1) In the first approach, we have a set of observed data in which both the z_t and the x_t values are known. Thus, we can approximate both $p(x|z)$ and $p(z_{t+1}|z_t)$ directly from the observed data. Then, we can predict the classes of each element in a new sequence of data using $p(z_1, \dots z_T|x_1^{new}, \dots, x_T^{new})$. Some examples of applicaitons of this model are data extraction from web pages (where you may be looking for data in a specific category, like whether it is a job listing) or speech recognition (such as automated telephone customer service).

In these sorts of situations, the vocabulary of zs is known. We find a transition matrix and emission probabilities given this data. Then, given a new sequence of data, we can predict the corresponding sequence of zs using the standard highest probability method (as in Bayes). Note that in some applications, you may learn the transition probabilities and the emission probabilities from separate sources. A good example of this is the speech recognition problem – huge sets of language data are available for generating an accurate language model, whereas the quantity of data available for voice recognition is much smaller. So it makes sense to generate the transition matrix using the language data, and only use the voice data for the emission probabilities.

(2) In the second approach, the only observed data we have is the sequence of xs. We don't have any observations of the z variables at all; they remain hidden. Obviously, this becomes a clustering problem, and we approach it in the same way as we approach mixture modeling.

Some terminology is in order now. The HMM of type (1) is a *classification* model; it's the *sequential* version of naive Bayes. The HMM of type (2) is a *clustering* model; it's the sequential version of mixture modeling. Naive Bayes and mixture modeling are known as *exchangeable* models, because you can exchange the data (change the order) and the model is unchanged; HMMs are sequential models.

3. The EM algorithm

As in our previous models, the important computation is that of $p(z_{1:T}|x_{1:T})$: the probability of each possible sequence of classes, given the data. The method of estimation will be essentially the same as in the mixture model. We'll use the EM algorithm. In literature on HMMs, the EM algorithm is sometimes referred to as the Baum-Welch algorithm.

Let's begin with the E step (the prediction step). Consider the probability of a particular z : $p(z_t|x_{1:T})$. Using Bayes' rule and the past-future independence, we have

$$\begin{aligned} p(z_t|x_{1:T}) &= \frac{p(x_{1:T}|z_t)p(z_t)}{p(x_{1:T})} \\ &= \frac{p(x_1,\ldots,x_t,z_t)p(x_{t+1},\ldots,x_T|z_t)}{p(x_{1:T})} \\ &= \frac{\alpha(z_t)\beta(z_t)}{p(x_{1:T})} \end{aligned} \tag{3.1}$$

The first step is simply an application of Bayes' rule, but the second step may seem a bit mysterious. Consider again the graphical representation of the model (see Figure 3 below).

Observe that the fact that z_t is observed here means that not only are all of the things after z_t independent of the things before z_t, but x_t is independent of *everything*. We could put x_t in either of the two terms up there; once we know z_t, knowing x_t gives us no new information for determining the other zs. Also note that we've defined two important functions:

$$\begin{aligned} \alpha(z_t) &:= p(x_1,\ldots,x_t,z_t) \\ \beta(z_t) &:= p(x_{t+1},\ldots,x_T|z_t). \end{aligned}$$

From (3.1) it's easy to see that

$$p(x_{1:T}) = \sum_{z_t=1}^{k} \alpha(z_t)\beta(z_t),$$

where k is the number of classes. Also, we see that this equation holds for all times $t = 1,\ldots,T$; the sum on the right is independent of the value of t.

model 3.jpg

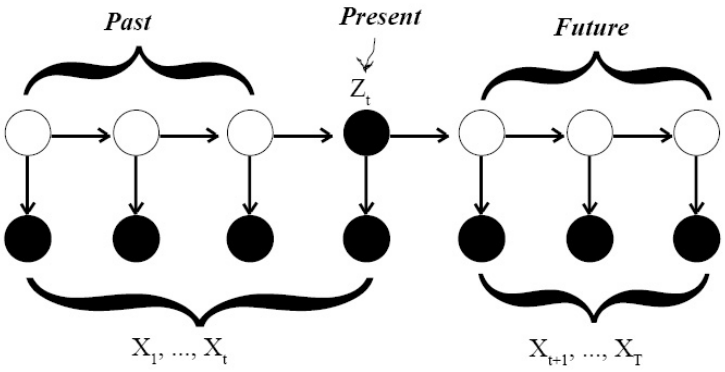

FIGURE 3. "past is independent of future, given present" illustrated

We see that we have reduced our problem to the computation of the α and β. Keep in mind that α and β are *vectors (of length k) of distributions.* For each possible value of z_t, each of those functions is a probability distribution.

A note of caution: z_t takes the values $1, 2, \ldots, k$. These are the *classes*, NOT the indices. z_t is the class value at a particular *time index* $t = 1, 2, \ldots, T$. Don't mix these up!

Now recall the parameters of the HMM:
A, the transition matrix. $a_{ij} = p(z_{t+1} = j | z_t = i)$ for every t.
$p(x|z)$, the emission probabilities.
π, the initial state probabilities.

We'll continue our computation by considering α. Of course, we can "easily" (perhaps "straightforwardly") compute α by summing directly for each value of t and each value of z_t. Seems like this would be a lot of summation, though. It would be nice to find a more efficient way to compute all of the function values. As it happens, we will be able to do that. (What follows is not directly transcribed from the lecture. It's Nathan's addition.) It's a huge and mysterious computation, so in order to clarify things a bit, let's discuss some motivation first. Recall that $\alpha(z_t)$ is the joint distribution of $x_1, \ldots, x_t, z_t$. Consider $\alpha(z_{t+1})$. It would be nice if we could compute this from the values of α at an earlier time. It seems reasonable that we should be able to do this from the values of $\alpha(z_t)$, because z_{t+1} only depends

on the things before it, and $\alpha(z_t)$ is the joint distribution of a lot of what we know from before time $t+1$. We know that the past is independent of the future given the present, but all we can do with that to start is split the x_{t+1} away from the others. But here's an idea – if we pretend we know what z_t is, then we can split the $x_1, \ldots, x_t$ away from the z_{t+1}. Just think of z_t as the present, z_{t+1} as the future, and the xs as the past. This would give us some expression with $\alpha(z_t)$ and a few other things that might not be too bad.

Now, how could we do this rigorously? The rigorous version of "pretending that we know what z_t is" would be introducing z_t into the conditional expectation and summing over it. So this is exactly what we'll do! This is the only conceptually difficult part of the following computation. Everything else is either basic algebraic manipulation (the chain rule for joint distributions) or the "past independent of future given present" rule that we're now familiar with. (We return to notes transcribed from the lecture.) Here's the computation:

$$\begin{aligned}
\alpha(z_{t+1}) &= p(x_{1:t+1}, z_{t+1}) \\
&= p(x_1, \ldots, x_{t+1}|z_{t+1})p(z_{t+1}) \\
&= p(x_1, \ldots, x_t|z_{t+1})p(z_{t+1})p(x_{t+1}|z_{t+1}) \\
&= \sum_{z_t=1}^{k} p(x_1, \ldots, x_t, z_t, z_{t+1})p(x_{t+1}|z_{t+1}) \\
&= \sum_{z_t} p(x_1, \ldots, x_t, z_{t+1}|z_t)p(z_t)p(x_{t+1}|z_{t+1}) \\
&= \sum_{z_t} p(x_1, \ldots, x_t|z_t)p(z_t)p(z_{t+1}|z_t)p(x_{t+1}|z_{t+1}) \\
&= \sum_{z_t} \underbrace{p(x_1, \ldots, x_t, z_t)}_{=\alpha(z_t)} \underbrace{p(z_{t+1}|z_t)}_{\text{transition}} \underbrace{p(x_{t+1}|z_{t+1})}_{\text{emission}} \\
&= \sum_{z_t} \alpha(z_t) a_{z_{t+1}z_t} p(x_{t+1}|z_{t+1}). \qquad (3.2)
\end{aligned}$$

Now, we can compute $\alpha(z_0)$ from known data: $\alpha(z_0) = \pi p(x_0|z_0)$. From this, we inductively get all of the other αs. What's the complexity of this computation? In (3.2), we have a sum over k possible values of z_t. There are k possible values of z_{t+1}. We have to do this from $t = 1$ to $T - 1$. So the complexity is $O(k^2T)$. You can check for yourself that this is much better than if we compute each of the α distributions directly! (It's already kT different values, and the sums are going to be much bigger.) That's all for this class. Next time we will do a similar computation for the β function.