Hidden Markov Models

1 Introduction

[TBW]

2 Basic Hidden Markov Models

Hidden Markov Models (HMM) are created to model a sequence of events, based on hidden observations. In the demonstrated model in Fig. 1, let's assume that $\mathbf{X} = (X_1, \ldots, X_n)$ is the set of observations, and $\mathbf{Z} = (Z_1, \ldots, Z_n)$ is the set of latent states. Each state X_i could take on m_x possible value, and each Z_i could take on m_z possible values. The matrix $A_{m_z \times m_z}$ characterizes the transition probabilities for Z variables. Similarly the matrix $B_{m_z \times m_x}$ denote the probability of observing X_t from Z_t. So we are assuming the Markovian transitions from a hidden-state to a next hidden-states.

In this sequential model, we are in interested in the following probabilities:

- Inference: Given the sequence $(\mathbf{x}, \mathbf{Z})$, what is the probability of $p(\mathbf{x}|\mathbf{Z})$.
- Learning: Given the sequence $\mathbf{x}$ what is the sequence $\mathbf{Z}$ that maximizes $p(\mathbf{Z}|\mathbf{x})$.

Now the question is given a sequence of training data, how can we find the most likely model which has generated the given sequence? We should assume an initial distribution for Z_1 by $w_{m_z \times 1}$. We can write the log-likelihood by

$$\mathcal{L} = \log w(Z_1) + \sum_{t=1}^{n-1} \log A\left(Z_t, Z_{t+1}\right) + \sum_{t=1}^{n} \log B\left(Z_t, X_t\right)$$

Figure 1: Structure of a Hidden Markov Model.

We shall train the model using EM algorithm, which also known as Baum-Welch. We define

$$\gamma_t(i,j) = p_{\theta_0}\left(Z_t = i, Z_{t+1} = j|\mathbf{x}\right),\ \gamma_t\left(i\right) = p_{\theta_0}\left(Z_t = i|\mathbf{x}\right) = \sum_j \gamma_t(i,j)$$

E-Step: In terms of the above definitions,

$$\mathbb{E}_{\mathbf{Z}|\mathbf{x},\theta} \log p(\mathbf{Z},\mathbf{x}|\theta) = \sum_{i=1}^{m_z} \gamma_1(i) \log w(i) + \sum_{i,j=1}^{m_z} \sum_{t=1}^{n-1} \gamma_t(i,j) \log A(i,j) + \sum_{i=1}^{m_z} \sum_{t=1}^{n} \gamma_t(i) \log B(i,x_t)$$

M-Step: and using derivatives of the the above term we have the following estimated values:

$$w^*(i) = \gamma_1(i),\ i = 1,\ldots,m_z;$$

$$A^*(i,j) = \frac{\sum_{t=1}^{n-1} \gamma_t(i,j)}{\sum_{j'} \sum_{t=1}^{n-1} \gamma_t(i,j')},\ i,j = 1,\ldots,m_z;$$

$$B^*(i,l) = \frac{\sum_{t=1;x_t=l} \gamma_t(i)}{\sum_t \gamma_t(i)},\ i = 1,\ldots,m_z,\ l = 1,\ldots,m_x;$$

Forward-Backward probabilities: Define forward $\alpha_{t+1}(i)$ and backward $\beta_{t-1}(i)$ probabilities as following:

$$\alpha_{t+1}(i) = p_\theta(x_1,\ldots,x_{t+1}, Z_{t+1} = i),\ \ \beta_{t-1}(i) = p_\theta(x_t,\ldots,x_n|Z_{t-1} = i)$$

We can show that the forward probability $\alpha_{t+1}(i)$ could be expressed based of the previous ones:

$$\begin{aligned}
\alpha_{t+1}(i) &= p_\theta(x_1,\ldots,x_{t+1}, Z_{t+1} = i) && \text{(1a)}\\
&= \sum_j p_\theta(x_1,\ldots,x_{t+1}, Z_t = j, Z_{t+1} = i) && \text{(1b)}\\
&= \sum_j \alpha_t(j) A(j,i) B(i,x_{t+1}) && \text{(1c)}
\end{aligned}$$

Similarly the backward probability could be written based on the future ones:

$$\begin{aligned}
\beta_{t-1}(i) &= p_\theta(x_t,\ldots,x_n|Z_{t-1} = i) && \text{(2a)}\\
&= \sum_j p_\theta(x_t,\ldots,x_n, Z_t = j, Z_{t-1} = i) && \text{(2b)}\\
&= \sum_j A(i,j) B(j,x_t) \beta_t(j) && \text{(2c)}
\end{aligned}$$

Now we can define the joint probabilities in terms of $\alpha(.)$ and $\beta(.)$.

$$p(Z_t = i, x_1, \ldots, x_n) = p(x_1, \ldots, x_n | Z_t = i).p(Z_t = i) \tag{3a}$$
$$= p(x_1, \ldots, x_t, Z_t = i).p(x_{t+1}, \ldots, x_n | Z_t = i) \tag{3b}$$
$$= \alpha_t(i).\beta_t(i) \tag{3c}$$

And also we can express the whole likelihood in terms of forward probabilities:

$$p(\mathbf{x}|\theta) = \sum_i p_\theta(x_1, \ldots, x_n, Z_n = i) = \sum_i \alpha_n(i)$$

The computation of all probabilities in terms of forward/backward probabilities makes everything so simple. We just need to calculate them in two swipes. It turns out that forward-backward algorithms is an special case of *belief-propagation* algorithm in Graphical Models.
Now the question is given a model, how can we make inference about the hidden states. The inference about the hidden variables depend on the definition of the optimality for the inference. One criterion is to choose the hidden values that are individually most likely, that is:

$$\hat{Z}_t^* = \arg\max_i p(Z_t = i|\mathbf{x}, \theta) = \arg\max_i \gamma_t(i)$$

But the problem is that considering each observation individually is not reasonable, and one needs to choose the most-likely sequence:

$$\hat{\mathbf{Z}}^* = \arg\max_{i_1,\ldots,i_n} p(Z_1 = i_1, \ldots, Z_n = i_n|\mathbf{x}, \theta)$$

This solution could be found using Viterbi algorithm. Assume that we have a trained HMM and we want to find the most probable latent path, given some observations. The highest probability of the sequence of hidden states given first t observations is:

$$\delta_t(i) = \max_{j1,\ldots,j_{t-1}} p(Z_1 = j_1, \ldots, Z_{t-1} = j_{t-1}, Z_t = i, \mathbf{x}_{1:t}|\theta)$$

Rewriting the optimality criterion it a recursive form :

$$\delta_{t+1}(i) = \left[\max_j \delta_t(j)A(j,i)\right].B(i, x_{t+1})$$

Similar implementation to forward/backward algorithm, will give us the optimal latent sequence. This turns out that Viterbi is an special case of *max-product* algorithm in graphical models.