Hidden Markov Models, II. Notation, Problems, Algorithms

Steven R. Dunbar

February 24, 2017

Outline

Notation

$$
\begin{aligned}
T &= \text{ length of observation sequence.} \\
N &= \text{ number of states in the model.} \\
M &= \text{ number of observation symbols.} \\
Q &= \{q_0, q_1, \dots, q_{N-1}\} = \text{ states of the Markov chain.} \\
V &= \{0, 1, 2, \dots, M-1\} = \text{ set of possible observations.} \\
A &= (a_{ij}) = (\mathbb{P}\left[q_j \mid q_i\right]) = \text{ state transition probability matri} \\
B &= (b_i(j)) = (b_{ij}) = \mathbb{P}\left[v_j \mid q_i\right] \text{ observation probability.} \\
\pi &= \{\pi_j\} = \text{ initial state distribution at time } 0. \\
\mathcal{O} &= (\mathcal{O}_0, \mathcal{O}_1, \dots \mathcal{O}_{T-1}) = \text{ the observation sequence.}
\end{aligned}
$$

The HMM is denoted by $\lambda = (A, B, \pi)$.

Process

- Set $t = 0$.
- Choose an initial state i_0 according to the initial state distribution π.
- Choose $\mathcal{O}_j$ according to $b_i(\cdot)$ the symbol probability distribution in state i.
- Set $t = t + 1$.
- Choose the new state q_j according to the probability transition matrix A.
- Return to step 3 if $t < T$; otherwise terminate the process.

Diagram of Hidden Markov Model

$$
\begin{tikzcd}
\text{Markov process:} & X_0 \arrow[r, "A"] \arrow[d, "B"'] & X_1 \arrow[r, "A"] \arrow[d, "B"'] & X_2 \arrow[r, "A"] \arrow[d, "B"'] & \cdots \arrow[r, "A"] & X_{T-1} \arrow[d, "B"'] \\
\text{Observations:} & \mathcal{O}_0 & \mathcal{O}_1 & \mathcal{O}_2 & \cdots & \mathcal{O}_{T-1}
\end{tikzcd}
$$

Evaluation Problem 1

Given the model $\lambda = (A, B, \pi)$ and a sequence of observations $\mathcal{O}$, find $\mathbb{P}\left[\mathcal{O} \mid \lambda\right]$. That is, *determine the likelihood of the observed sequence* $\mathcal{O}$, given the model.

Problem 1 is the **evaluation problem**: given a model and observations, how can we compute the probability that the model produced the observed sequence. We can also view the problem as: how we "score" or evaluate the model. If we think of the case in which we have several competing models, the solutions of problem 1 allows us to choose the model that best matches the observations.

Problem 2: Uncover the Hidden States

Given the model $\lambda = (A, B, \pi)$ and a sequence of observations $\mathcal{O}$, find an optimal state sequence. In other words, we want to uncover the hidden part of the Hidden Markov Model.

Problem 2 is the one in which we attempt to uncover the hidden part of the model, i.e. the state sequence. This is the *estimation problem*. Use an optimality criterion to discriminate which sequence best matches the observations. Two optimality criteria are common, and so the choice of criterion is a strong influence on the revealed state sequence.

Problem 3: Training and Parameter Fitting

Given an observation sequence $\mathcal{O}$ and the dimensions N and M, find the model $\lambda = (A, B, \pi)$ that maximizes the probability of $\mathcal{O}$. This can interpreted as training a model to best fit the observed data. We can also view this as search in the parameter space represented by A, B and π.

The solution of Problem 3 attempts to optimize the model parameters so as best to describe how the observed sequence comes about. The observed sequence used to solve Problem 3 is called a *training sequence* since it is used to train the model. This training problem is the crucial one for most applications of hidden Markov models since it creates best models for real phenomena.

Forward Algorithm

The **forward algorithm** or **alpha pass**. For $t = 0, 1, 2, \dots T-1$ and $i = 0, 1, \dots, N-1$, define

$$\alpha_t(i) = \mathbb{P}\left[\mathcal{O}_0, \mathcal{O}_1, \mathcal{O}_2, \dots \mathcal{O}_t, x_t = q_i \mid \lambda\right]$$

Then $\alpha_t(i)$ is the probability of the partial observation of the sequence up to time t with the Markov process in state q_i at time t.

Recursive computation

The crucial insight is that the $\alpha(i)$ can be computed recursively as follows

1. Let $\alpha_0(i) = \pi_i b_i(\mathcal{O}_0)$, for $i = 0, 1, \dots N-1$.
2. For $t = 1, 2, \dots, T-1$ and $i = 0, 1, \dots N-1$, compute
$$\alpha_t(i) = \left[\sum_{j=0}^{N-1} \alpha_{t-1}(j) a_{ji} \right] b_i(\mathcal{O}_t)$$
3. Then it follows that
$$\mathbb{P}\left[\mathcal{O} \mid \lambda\right] = \sum_{i=0}^{N-1} \alpha_{T-1}(i).$$

The forward algorithm only requires about $N^2 T$ multiplications, a large improvement over the $2TN^T$

Backward Algorithm

The **backward algorithm**, or **beta-pass**. This is analogous to the alpha-pass described in the solution to the first problem of HMMs, except that it starts at the end and works back toward the beginning.

It is independent of the forward algorithm, so it can be done in parallel.

Recursive computation

For $t = 0, 1, 2, \dots T-1$ and $i = 0, 1, \dots, N-1$, define

$$\beta_t(i) = \mathbb{P}\left[\mathcal{O}_{t+1}, \mathcal{O}_{t+2}, \dots \mathcal{O}_{T-1} \,\middle|\, x_t = q_i, \lambda\right].$$

The crucial insight again is that the $\beta_t(i)$ can be computed recursively as follows

1. Let $\beta_{T-1}(i) = 1$, for $i = 0, 1, \dots N-1$.
2. For $t = T-2, T-3, \dots, 0$ and $i = 0, 1, \dots N-1$, compute

$$\beta_t(i) = \sum_{j=0}^{N-1} a_{ij} b_j(\mathcal{O}_{t+1}) \beta_{t+1}(j)$$

The Viterbi (Forward-Backward) Algorithm

For $t = 0, 1, 2, \dots T-1$ and $i = 0, 1, \dots N-1$ define the *posteriors*

$$\gamma_t(i) = \mathbb{P}\left[x_t = q_i \,|\, \mathcal{O}, \lambda\right].$$

Since $\alpha_t(i)$ measures the probability up to time t and $\beta_t(i)$ measures the probability after time t

$$\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{\mathbb{P}\left[\mathcal{O} \,|\, \lambda\right]}.$$

Recall $\mathbb{P}\left[\mathcal{O} \,|\, \lambda\right] = \sum_{i=0}^{N-1} \alpha_{T-1}(i)$.

The most likely state at time t is the state q_i for which $\gamma_i(t)$ is a maximum.

Baum-Welch Algorithm

For $t = 0, 1, \dots, T-2$ and $i, j \in \{0, 1, \dots, N-1\}$, define the **di-gamma passes** as

$$\gamma_t(i,j) = \mathbb{P}\left[x_t = q_i, x_{t+1} = j \mid \mathcal{O}, \lambda\right]$$

so $\gamma_t(i,j)$ is the probability of being in state q_i at time t and transitioning to state q_j at time $t+1$. The gammas can be written in terms of α, β, A and B as

$$\gamma_t(i,j) = \frac{\alpha_t(i) a_{ij} b_j(\mathcal{O}_{t+1}) \beta_{t+1}(j)}{\mathbb{P}\left[\mathcal{O} \mid \lambda\right]}$$

For $t = 0, 1, \dots, T-2$, the $\gamma_t(i)$ and $\gamma_t(i,j)$ are related by

$$\gamma_t(i) = \sum_{j=0}^{N-1} \gamma_t(i,j)$$

Training and Estimating Parameters

Re-estimation is an iterative process. First we initialize $\lambda = (A, B, \pi)$ with a best guess, or if no reasonable guess is available, we choose random values such that $\pi_i \approx 1/N$ and $a_{ij} \approx 1/N$ and $b_j(k) \approx 1/M$. It is critical that A, B and π be randomized since exactly uniform values will results in a local maximum from which the model cannot climb. As always, A, B, π must be row stochastic.

Iterative Process

1. Initialize $\lambda = (A, B, \pi)$ with a best guess.
2. Compute $\alpha_t(i)$, $\beta_t(i)$, $\gamma_t(i, j)$, and $\gamma_t(i)$.
3. Re-estimate the model$\lambda = (A, B, \pi)$.
4. If $\mathbb{P}[\mathcal{O} \mid \lambda]$ increases by at least some predetermined threshold or the predetermined maximum number of iterations has not been exceeded, go to step 2.
5. Else stop and output $\lambda = (A, B, \pi)$.

Andrew Viterbi

- Born 1935 in Bergamo Italy
- Italian Jewish family emigrated to US, August 1939
- Graduated BS/MS/EE from MIT in 1957
- JPL/NASA from 1958-1962
- Ph.D./EE USC in 1962
- Professor EE at UCLA, 1963-1973
- Viterbi ALgorithm, March 1966
- 1969 co-founds startup Linkabit
- Acquired by Microwave Assoc. Communications, 1980
- Co-founds Qualcomm, 1985-2000
- 2004: Gives $52 Million to USC School of Engineering

The Variable Factory 1

Production process in a factory

- If in good state 0 then, independent of the past, with probability 0.9 it will be in state 0 during the next period,
- with probability 0.1 it will be in bad state 1,
- once in state 1 it remains in that state forever.

Classic Markov chain:

$$A = \begin{array}{c c} & \begin{array}{cc} 0 & 1 \end{array} \\ \begin{array}{c} 0 \\ 1 \end{array} & \begin{pmatrix} 0.9 & 0.1 \\ 0 & 1 \end{pmatrix} \end{array}.$$

The Variable Factory 2

- Each item produced in state 0 is acceptable quality with probability 0.99.
- Each item produced in state 1 is acceptable quality with probability 0.96.

$$\mathbb{P}\left[u \mid 0\right] = 0.01 \qquad \mathbb{P}\left[a \mid 0\right] = 0.99$$
$$\mathbb{P}\left[u \mid 1\right] = 0.04 \qquad \mathbb{P}\left[a \mid 1\right] = 0.96$$

Alpha Pass (Forward)

$\alpha_t(i)$	0 (a)	1 (u)	2 (a)
0	$(0.8)(0.99)$ $= 0.792$	$[(0.792)(0.9)$ $+(0.192)(0)]$ $\times(0.01)$ $= 0.007128$	$[0.007128)(0.9)$ $+(0.010848)(0)]$ $\times(0.99)$ $= 0.006351$
1	$(0.2)(0.96)$ $= 0.192$	$[(0.792)(0.1)$ $+(0.192)(1)]$ $\times(0.04)$ $= 0.010848$	$[0.007128)(0.1)$ $+(0.010848)(1)]$ $\times(0.96)$ $= 0.0011098$
			$0.006351 + 0.011098$ $= 0.01745$

Beta-Pass (Backward)

$\beta_t(i)$	0	1	2
0	$(0.9)(0.01)(0.987)$ $+(0.1)(0.04)(0.96)$ $= 0.012723$	$(0.9)(0.99)(1)$ $+(0.1)(0.96)(1)$ $= 0.987$	1
1	$(0)(0.01)(0.987)$ $+(1)(0.04)(0.96)$ $= 0.0384$	$(0)(0.99)(1)$ $+(1)(0.96)(1)$ $= 0.96$	1

Gamma Pass, (Posterior)

$\alpha\beta$	0	1	2
	$(0.792)(0.012723)$ $= 0.01007662$	$(0.0077128)(0.987)$ $= 0.007035$	(0.006351) $= 0.00635$
	$(0.192)(0.0384)$ $= 0.007373$	$(0.010848)(0.96)$ $= 0.010414$	(0.011098) $= 0.01109$
	0.01745	0.01745	0.01745

$\gamma_t(i)$	0	1	2
0	0.5775	0.4031	0.3640
1	0.4225	0.5968	0.6360

Most Probable States

$\gamma_t(i)$	0	1	2
0	0.5775	0.4031	0.3640
1	0.4225	0.5968	0.6360

At each time the state which is individually most likely is $(0, 1, 1)$ (i.e. factory good, not good, not good) Note that here all state transitions are valid, but need not always be the case.

This is different from the most likely overall path which was $(1, 1, 1)$ (exhaustively computed last time.)

In a hypothetical dishonest casino,

- the casino uses a fair die most of the time,
- occasionally the casino secretly switches to a loaded die,
- later the casino switches back to the fair die.
- the switch from fair-to-loaded occurs with probability 0.05
- from loaded-to-fair with probability 0.1.
- assume that the loaded die will come up “six” with probability 0.5
- the remaining five numbers with probability 0.1 each.