

# Hidden Markov Model Cheatsheet

Kanru Hua\*

*University of Illinois at Urbana-Champaign*

April 2016

## 1 Preliminaries

### Conditional (Posterior) Probability

$$P(A|B) = \frac{P(A, B)}{P(B)}$$

where  $P(A, B)$  is called **joint probability** and  $P(B)$  is called **prior probability**.

### Statistical Independence

If A and B are independent events,

$$P(A|B) = P(A) \text{ and } P(B|A) = P(B) \\ \text{or } P(A, B) = P(A)P(B)$$

### Marginal Probability

$$P(A) = \sum_b P(A, B = b)$$

### Expectation

$$E[X] = \sum_j P(X = x_j)x_j$$

### Jensen's Inequality

(You are **not** required to know this beforehand)

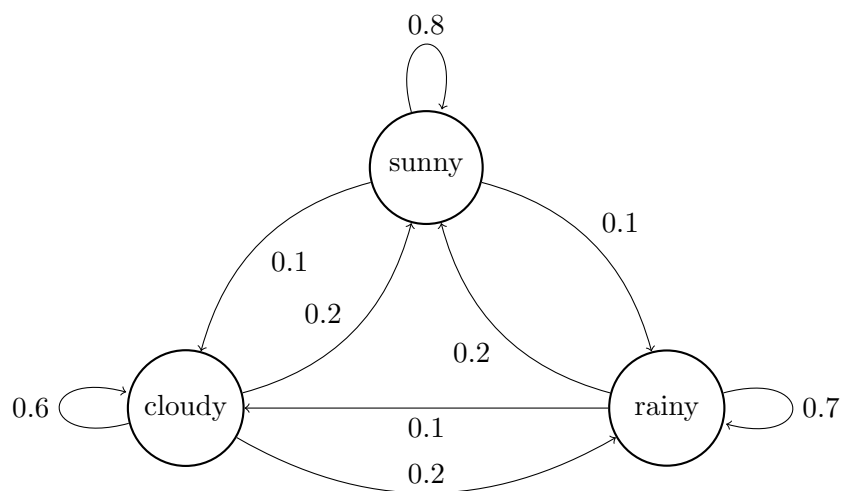
$$E[f(X)] \leq f(E[X]) \text{ for any concave function } f \\ E[f(X)] = f(E[X]) \text{ if } X \text{ is constant}$$

---

\*Email address: [khua5@illinois.edu](mailto:khua5@illinois.edu)

## 2 Markov Chain

### 2.1 Example



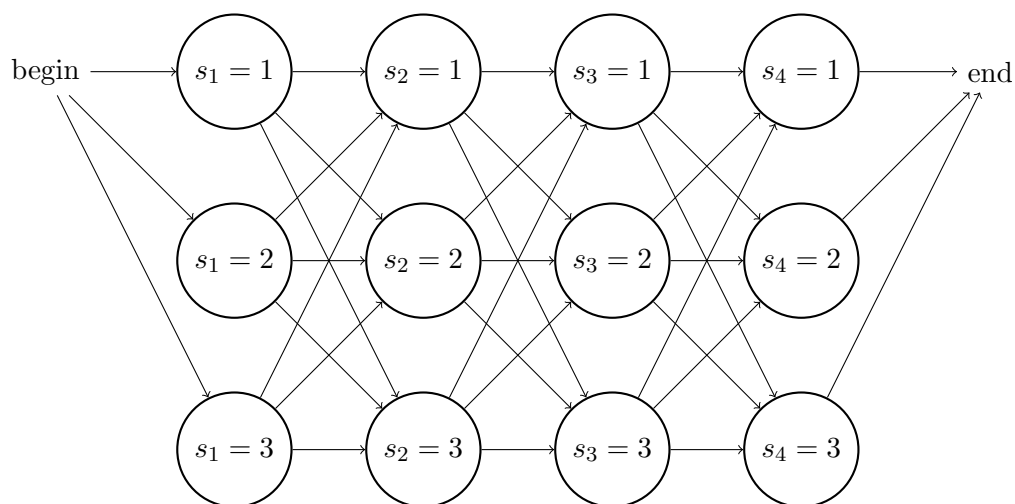
### 2.2 Definition

parameter set:  $\lambda = \{\pi, \mathbf{A}\}$   
 initial probability:  $\pi_j = P(s_1 = j | \lambda)$   
 transition probability:  $a_{ij} = P(s_t = j | s_{t-1} = i, \lambda)$

### Markov Property

$$P(s_t = j | s_1, s_2, \dots, s_{t-2}, s_{t-1} = i, \lambda) = P(s_t = j | s_{t-1} = i, \lambda) = a_{ij}$$

A lateral view where each edge has a transition probability, or an initial probability for  $t = 1$ :



### 3 Hidden Markov Model

#### 3.1 Definition

parameter set:  $\lambda = \{\pi, \mathbf{A}, \mathbf{B}\}$

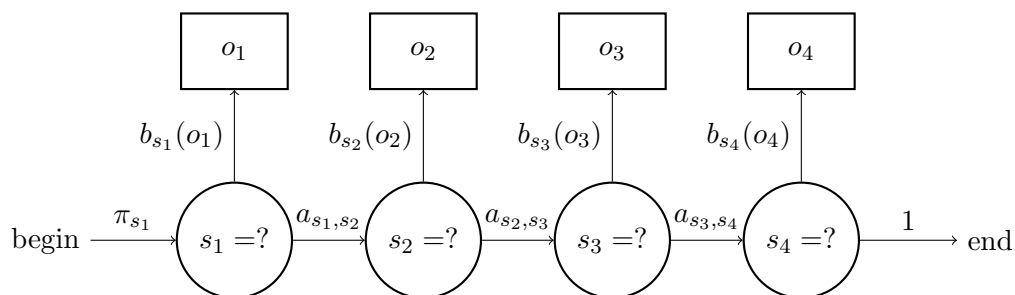
initial probability:  $\pi_j = P(s_1 = j | \lambda)$

transition probability:  $a_{ij} = P(s_t = j | s_{t-1} = i, \lambda)$

output probability (discrete case):  $b_{jk} = P(o_t = k | s_t = j, \lambda)$

output probability density (continuous case):  $b_j(o_t) = p(o_t | s_t = j, \lambda)$

#### 3.2 Graph



#### 3.3 List of Operations & Algorithms

What can we do with HMM?

- Generation
  - Random walk
- Inference
  - **Forward/Backward algorithm**
  - **Viterbi algorithm**
  - Total probability:  $P(O | \lambda)$
  - State occupancy probability:  $P(s_t = j | O, \lambda)$
  - Optimal probability & state sequence:  $\arg \max_s P(O, s | \lambda)$
- Parameter estimation (training)
  - Initialization
  - Viterbi training
  - **Baum-Welch algorithm** (expectation-maximization)

## 4 HMM Inference

### 4.1 Building Blocks

What we already know:

$$\begin{aligned}P(s_1 = i|\lambda) &= \pi_i \\P(s_t = j|s_{t-1} = i, \lambda) &= a_{ij} \\P(o_t|s_t = j, \lambda) &= b_j(o_t)\end{aligned}$$

Trivial extensions using Markov property:

$$\begin{aligned}P(s_1, s_2, \dots, s_t|\lambda) &= \pi_{s_1} \prod_{\tau=2}^t a_{s_{\tau-1}, s_{\tau}} \\P(o_1, o_2, \dots, o_t|s_1, s_2, \dots, s_t, \lambda) &= \prod_{\tau=1}^t b_{s_{\tau}}(o_{\tau})\end{aligned}$$

By combining above equations, we get

$$P(o_1, o_2, \dots, o_t, s_1, s_2, \dots, s_t|\lambda) = \pi_{s_1} b_{s_1}(o_1) \prod_{\tau=2}^t a_{s_{\tau-1}, s_{\tau}} b_{s_{\tau}}(o_{\tau})$$

### 4.2 Inference Algorithms

#### Forward Algorithm

$$\begin{aligned}\alpha_t(i) &= P(o_1, o_2, \dots, o_t, s_t = i|\lambda) \\&= P(o_t|s_t = i, \lambda) \sum_j P(o_1, \dots, o_{t-1}, s_{t-1} = j|\lambda) P(s_t = i|s_{t-1} = j, \lambda) \\&= b_i(o_t) \sum_j a_{ji} \alpha_{t-1}(j) \\\alpha_1(i) &= P(o_1|s_1 = i, \lambda) P(s_1 = i|\lambda) = b_i(o_1) \pi_i\end{aligned}$$

#### Backward Algorithm

$$\begin{aligned}\beta_t(i) &= P(o_{t+1}, o_{t+2}, \dots, o_T|s_t = i, \lambda) \\&= \sum_j P(o_{t+1}|s_{t+1} = j, \lambda) P(o_{t+2}, \dots, o_T|s_{t+1} = j, \lambda) P(s_{t+1} = j|s_t = i, \lambda) \\&= \sum_j b_j(o_{t+1}) \beta_{t+1}(j) a_{ij} \\\beta_T(i) &= 1\end{aligned}$$

## Viterbi Algorithm

$$\begin{aligned}
\alpha_t^*(i) &= \max_{s_1, \dots, s_{t-1}} P(o_1, \dots, o_t, s_1, \dots, s_{t-1}, s_t = i | \lambda) \\
&= P(o_t | s_t = i, \lambda) \max_j \left( P(s_t = i | s_{t-1} = j, \lambda) \max_{s_1, \dots, s_{t-2}} P(o_1, \dots, o_{t-1}, s_1, \dots, s_{t-2}, s_{t-1} = i | \lambda) \right) \\
&= b_i(o_t) \max_j a_{ji} \alpha_{t-1}^*(j) \\
p_t^*(i) &= \arg \max_j a_{ji} \alpha_{t-1}^*(j) \\
\alpha_1^*(i) &= b_i(o_1) \pi_i \\
p_1^*(i) &= 0
\end{aligned}$$

## Total Probability

$$\begin{aligned}
P(O | \lambda) &= \sum_j P(o_1, \dots, o_T, s_t = j | \lambda) = \sum_j \alpha_T(j) \quad (\text{from forward probability}) \\
P(O | \lambda) &= \sum_j P(o_1 | s_1 = j, \lambda) P(o_2, \dots, o_T | s_1 = j, \lambda) P(s_1 = j | \lambda) \\
&= \sum_j b_j(o_1) \beta_1(j) \pi_j \quad (\text{from backward probability}) \\
P(O | \lambda) &= \sum_j P(o_1, \dots, o_t, o_{t+1}, \dots, o_T, s_t = j | \lambda) \\
&= \sum_j P(o_1, \dots, o_t, s_t = j | \lambda) P(o_{t+1}, \dots, o_T | s_t = j, \lambda) \\
&= \sum_j \alpha_t(j) \beta_t(j) \quad (\text{from both forward and backward probability, for arbitrary } t)
\end{aligned}$$

## State Occupancy Probability

$$\begin{aligned}
\gamma_t(j) &= P(s_t = j | O, \lambda) \\
&= \frac{P(o_1, \dots, o_t, s_t = j | \lambda) P(o_{t+1}, \dots, o_T | s_t = j, \lambda)}{P(O | \lambda)} \\
&= \frac{\alpha_t(j) \beta_t(j)}{P(O | \lambda)}
\end{aligned}$$

note that  $\sum_j \gamma_t(j) = 1$

## State Transition Probability (not to be confused with $a_{ij}$ )

$$\begin{aligned}
\gamma_t(i, j) &= P(s_{t-1} = i, s_t = j | O, \lambda) \\
&= \frac{\alpha_{t-1}(i) \beta_t(j) b_j(o_t) a_{ij}}{P(O | \lambda)} \quad \text{whose derivation is similar to } \gamma_t(j)
\end{aligned}$$

## 5 HMM Parameter Estimation

### 5.1 Expectation-Maximization Algorithm

Goal: obtain maximum likelihood estimation of  $\lambda$ :

$$\begin{aligned}\lambda^* &= \arg \max_{\lambda} l(\lambda) = \arg \max_{\lambda} \log P(O|\lambda) \\ l(\lambda) &= \log P(O|\lambda) = \log \sum_{\mathbf{s}} P(O, \mathbf{s}|\lambda)\end{aligned}$$

Motivation: find an alternative likelihood function whose derivatives are easier to calculate.

Assume we have a function  $Q(\mathbf{s})$  such that  $\sum_{\mathbf{s}} Q(\mathbf{s}) = 1$  and  $Q(\mathbf{s}) > 0 \forall \mathbf{s}$ ,

$$\begin{aligned}l(\lambda) &= \log \sum_{\mathbf{s}} Q(\mathbf{s}) \frac{P(O, \mathbf{s}|\lambda)}{Q(\mathbf{s})} \\ &= \log E_{\mathbf{s}} \left[ \frac{P(O, \mathbf{s}|\lambda)}{Q(\mathbf{s})} \right] \\ &\geq E_{\mathbf{s}} \left[ \log \frac{P(O, \mathbf{s}|\lambda)}{Q(\mathbf{s})} \right] \quad (\text{whose partial derivatives have closed form})\end{aligned}$$

To make the lower bound more “effective”, i.e., we want  $\log E_{\mathbf{s}}[\dots] = E_{\mathbf{s}}[\log(\dots)]$ ,

$$\begin{cases} \frac{P(O, \mathbf{s}|\lambda)}{Q(\mathbf{s})} &= c \\ \sum_{\mathbf{s}} Q(\mathbf{s}) &= 1 \end{cases} \Rightarrow Q(\mathbf{s}) = P(\mathbf{s}|O, \lambda)$$

#### EM Algorithm

Repeat until convergence {  
  Expectation:  $Q(\mathbf{s}) = P(\mathbf{s}|O, \lambda)$   
  Maximization:  $\lambda^* = \arg \max_{\lambda} \sum_{\mathbf{s}} Q(\mathbf{s}) \log \frac{P(O, \mathbf{s}|\lambda)}{Q(\mathbf{s})}$   
}

### 5.2 Baum-Welch Algorithm

Error function (in maximization step):

$$\begin{aligned}J(\lambda) &= \sum_{\mathbf{s}} Q(\mathbf{s}) \log \frac{P(O, \mathbf{s}|\lambda)}{Q(\mathbf{s})} \\ &= \sum_{\mathbf{s}} Q(\mathbf{s}) \left( \log \pi_{s_1} + \sum_{t=2}^T \log a_{s_{t-1}, s_t} + \sum_{t=1}^T \log b_{s_t}(o_t) - \log(Q(\mathbf{s})) \right)\end{aligned}$$

Take partial derivative with respect to, for example,  $a_{ij}$ ,

$$\begin{aligned}
\frac{\partial J}{\partial a_{ij}} &= \frac{\partial}{\partial a_{ij}} \sum_{\mathbf{s}} Q(\mathbf{s}) \sum_{t=2}^T \log a_{s_{t-1}, s_t} \\
&= \frac{\partial}{\partial a_{ij}} \sum_{t=2}^T \sum_m \sum_n \log a_{mn} \sum_{\substack{s_1, \dots, s_{t-2} \\ s_{t+1}, \dots, s_T}} Q(\mathbf{s}) \\
&= \frac{\partial}{\partial a_{ij}} \sum_{t=2}^T \sum_m \sum_n \log a_{mn} \underbrace{P(s_{t-1} = m, s_t = n | O, \lambda')}_{\gamma_t(m, n)} \\
&= \frac{1}{a_{ij}} \sum_{t=2}^T \gamma_t(i, j)
\end{aligned}$$

To make sure  $\sum_j a_{ij} = 1 \forall i$ , introduce Lagrange multiplier  $l$ ,

$$\begin{cases} \sum_{t=2}^T \gamma_t(i, j) &= l a_{ij} \\ \sum_j a_{ij} &= 1 \end{cases}$$

Solve the equations,

$$l = \sum_{t=2}^T \sum_n \gamma_t(i, n), \quad a_{ij} = \frac{\sum_{t=2}^T \gamma_t(i, j)}{\sum_{t=2}^T \sum_n \gamma_t(i, n)} = \frac{\sum_{t=2}^T \gamma_t(i, j)}{\sum_{t=2}^T \gamma_{t-1}(i)}$$

Similarly for  $\pi_i$  and  $b_{ik}$  (discrete case) we get,

$$\begin{aligned}
\pi_i &= \gamma_1(i) \\
b_{ik} &= \frac{\sum_{t: o_t = k} \gamma_t(i)}{\sum_{t=1}^T \gamma_t(i)}
\end{aligned}$$

### Multiple Observation Sequences

$$\begin{aligned}
\pi_i &= \frac{1}{L} \sum_{l=0}^L \gamma_1^l(i) \\
a_{ij} &= \frac{\sum_{l=0}^L \sum_{t=2}^T \gamma_t^l(i, j)}{\sum_{l=0}^L \sum_{t=2}^T \gamma_{t-1}^l(i)} \quad b_{ik} = \frac{\sum_{l=0}^L \sum_{t: o_t = k} \gamma_t^l(i)}{\sum_{l=0}^L \sum_{t=1}^T \gamma_t^l(i)}
\end{aligned}$$

## 5.3 Baum-Welch Algorithm for Continuous Output Distributions

### Multivariate Normal Distribution

$$\mu_j = \frac{\sum_{t=1}^T \gamma_t(j) o_t}{\sum_{t=1}^T \gamma_t(j)} \quad \Sigma_j = \frac{\sum_{t=1}^T \gamma_t(j) o_t o_t^T}{\sum_{t=1}^T \gamma_t(j)} - \mu_j \mu_j^T$$

## Multivariate Gaussian Mixture Model

Gaussian mixture model:

$$p(x \in \mathbf{R}^k | \mathbf{c}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = (2\pi)^{-\frac{k}{2}} \sum_j c_j |\boldsymbol{\Sigma}_j|^{-\frac{k}{2}} e^{-\frac{k}{2}(x-\boldsymbol{\mu}_j)^T \boldsymbol{\Sigma}_j^{-1} (x-\boldsymbol{\mu}_j)}$$

HMM-GMM:

$$\begin{aligned} g_{jk}(o_t) &= p(o_t | s_t = j, m_t = k, \lambda) = p(o_t | \boldsymbol{\mu}_{jk}, \boldsymbol{\Sigma}_{jk}) \\ b_j(o_t) &= p(o_t | s_t = j, \lambda) = \sum_k c_{jk} g_{jk}(o_t) \end{aligned}$$

HMM-GMM inference:

$$\xi_t(j, k) = p(s_t = j, m_t = k | O, \lambda) = \frac{\sum_i \alpha_{t-1}(i) \beta_t(j) a_{ij} c_{jk} g_{jk}(o_t)}{P(O | \lambda)}$$

HMM-GMM parameter estimation:

$$\begin{aligned} c_{jk} &= \frac{\sum_{t=1}^T \xi_t(j, k)}{\sum_{t=1}^T \gamma_t(j)} \\ \mu_{jk} &= \frac{\sum_{t=1}^T \xi_t(j, k) o_t}{\sum_{t=1}^T \xi_t(j, k)} \quad \Sigma_{jk} = \frac{\sum_{t=1}^T \xi_t(j, k) o_t o_t^T}{\sum_{t=1}^T \xi_t(j, k)} - \mu_{jk} \mu_{jk}^T \end{aligned}$$

## References

- [1] Fink, Gernot A. “Markov models for pattern recognition: from theory to applications”. Springer Science & Business Media, 2014.
- [2] Ng, Andrew. “Mixtures of Gaussians and the EM algorithm.” Stanford University. CS229 Lecture Notes (2014).
- [3] Bilmes, Jeff A. “A gentle tutorial of the EM algorithm and its application to parameter estimation for Gaussian mixture and hidden Markov models.” International Computer Science Institute 4.510 (1998): 126.