

Hidden Markov Models – II

Tapas Kanungo

Language and Media Processing Lab

Center for Automation Research

University of Maryland

Web: www.cfar.umd.edu/~kanungo

Email: kanungo@cfar.umd.edu

April 6, 1998

Outline

1. Review: Markov Models, HMM, Forward
2. Backward algorithm
3. Viterbi algorithm
4. Baum-Welch estimation algorithm

Markov Models

- Observable states:

$$1, 2, \dots, N$$

- Observed sequence:

$$q_1, q_2, \dots, q_t, \dots, q_T$$

- First order Markov assumption:

$$P(q_t = j | q_{t-1} = i, q_{t-2} = k, \dots) = P(q_t = j | q_{t-1} = i)$$

- Stationarity:

$$P(q_t = j | q_{t-1} = i) = P(q_{t+l} = j | q_{t+l-1} = i)$$

Markov Models

- State transition matrix A :

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1j} & \cdots & a_{1N} \\ a_{21} & a_{22} & \cdots & a_{2j} & \cdots & a_{2N} \\ \vdots & \vdots & \cdots & \vdots & \cdots & \vdots \\ a_{i1} & a_{i2} & \cdots & a_{ij} & \cdots & a_{iN} \\ \vdots & \vdots & \cdots & \vdots & \cdots & \vdots \\ a_{N1} & a_{N2} & \cdots & a_{Nj} & \cdots & a_{NN} \end{bmatrix}$$

where

$$a_{ij} = P(q_t = j | q_{t-1} = i) \quad 1 \leq i, j, \leq N$$

- Constraints on a_{ij} :

$$a_{ij} \geq 0, \quad \forall i, j$$

$$\sum_{j=1}^N a_{ij} = 1, \quad \forall i$$

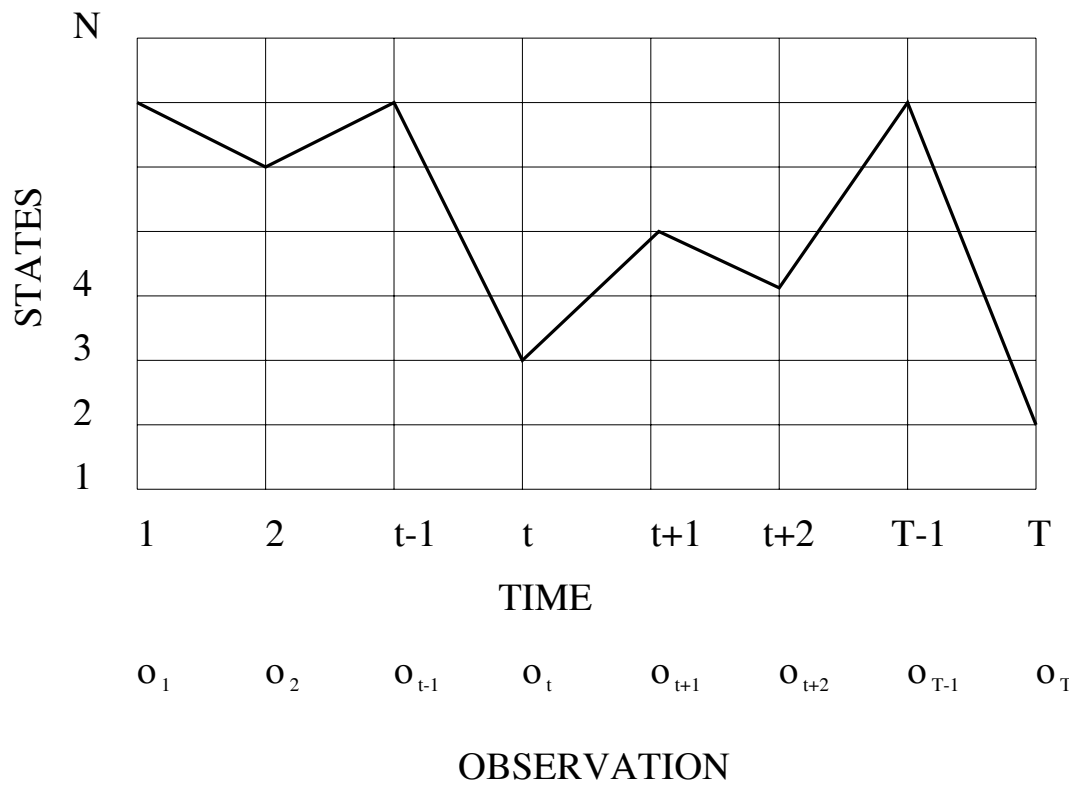
Hidden Markov Models

- States are not observable
- Observations are probabilistic functions of state
- State transitions are still probabilistic

Urn and Ball Model

- N urns containing colored balls
- M distinct colors of balls
- Each urn has a (possibly) different distribution of colors
- Sequence generation algorithm:
 1. Pick initial urn according to some random process.
 2. Randomly pick a ball from the urn and then replace it
 3. Select another urn according a random selection process associated with the urn
 4. Repeat steps 2 and 3

The Trellis



Elements of Hidden Markov Models

- N – the number of hidden states
- Q – set of states $Q = \{1, 2, \dots, N\}$
- M – the number of symbols
- V – set of symbols $V = \{1, 2, \dots, M\}$
- A – the state-transition probability matrix.

$$a_{ij} = P(q_{t+1} = j | q_t = i) \quad 1 \leq i, j, \leq N$$

- B – Observation probability distribution:

$$B_j(k) = P(o_t = k | q_t = j) \quad 1 \leq k \leq M$$

- π – the initial state distribution:

$$\pi_i = P(q_1 = i) \quad 1 \leq i \leq N$$

- λ – the entire model $\lambda = (A, B, \pi)$

Three Basic Problems

1. Given observation $O = (o_1, o_2, \dots, o_T)$ and model $\lambda = (A, B, \pi)$, efficiently compute $P(O|\lambda)$.
 - Hidden states complicate the evaluation
 - Given two models λ_1 and λ_2 , this can be used to choose the better one.
2. Given observation $O = (o_1, o_2, \dots, o_T)$ and model λ find the optimal state sequence $q = (q_1, q_2, \dots, q_T)$.
 - Optimality criterion has to be decided (e.g. maximum likelihood)
 - “Explanation” for the data.
3. Given $O = (o_1, o_2, \dots, o_T)$, estimate model parameters $\lambda = (A, B, \pi)$ that maximize $P(O|\lambda)$.

Solution to Problem 1

- **Problem:** Compute $P(o_1, o_2, \dots, o_T | \lambda)$

- **Algorithm:**

- Let $q = (q_1, q_2, \dots, q_T)$ be a state sequence.
- Assume the observations are independent:

$$\begin{aligned} P(O|q, \lambda) &= \prod_{i=1}^T P(o_t|q_t, \lambda) \\ &= b_{q_1}(o_1)b_{q_2}(o_2) \cdots b_{q_T}(o_T) \end{aligned}$$

- Probability of a particular state sequence is:

$$P(q|\lambda) = \pi_{q_1} a_{q_1 q_2} a_{q_2 q_3} \cdots a_{q_{T-1} q_T}$$

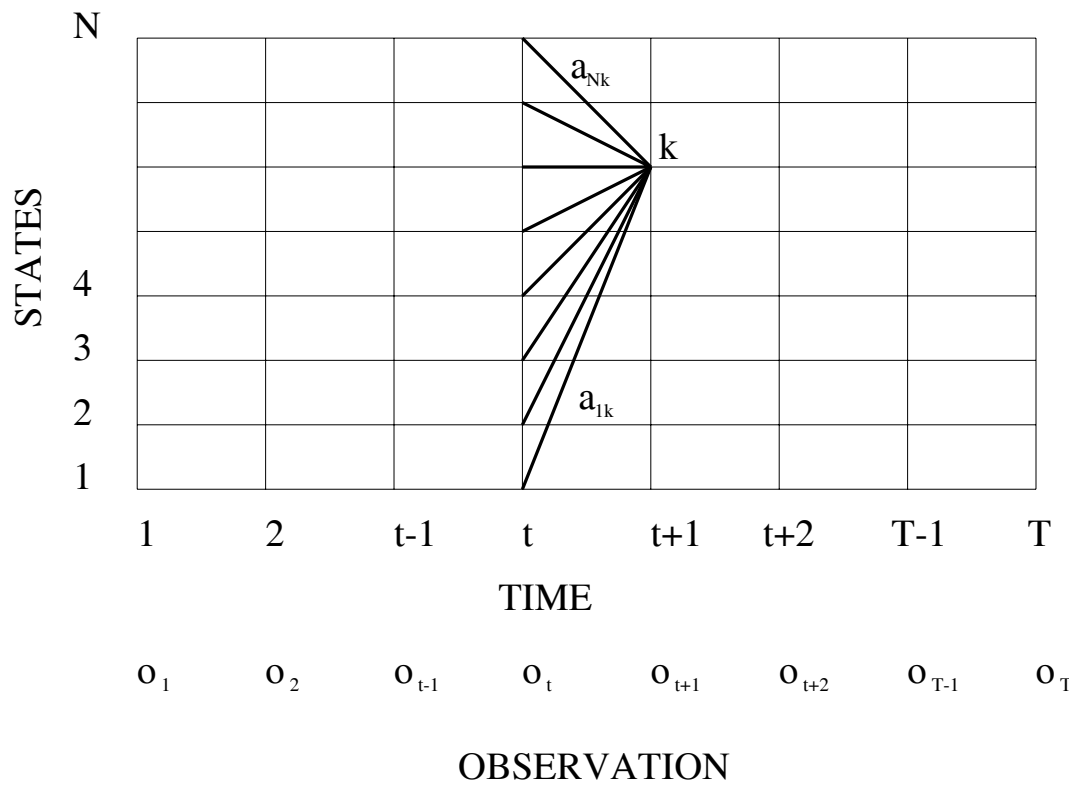
- Also, $P(O, q|\lambda) = P(O|q, \lambda)P(q|\lambda)$
- Enumerate paths and sum probabilities:

$$P(O|\lambda) = \sum_q P(O|q, \lambda)P(q|\lambda)$$

- N^T state sequences and $O(T)$ calculations.

Complexity: $O(TN^T)$ calculations.

Forward Algorithm: Intuition



Forward Algorithm

- Define forward variable $\alpha_t(i)$ as:

$$\alpha_t(i) = P(o_1, o_2, \dots, o_t, q_t = i | \lambda)$$

- $\alpha_t(i)$ is the probability of observing the partial sequence (o_1, o_2, \dots, o_t) such that the state q_t is i .

- Induction:

1. Initialization: $\alpha_1(i) = \pi_i b_i(o_1)$

2. Induction:

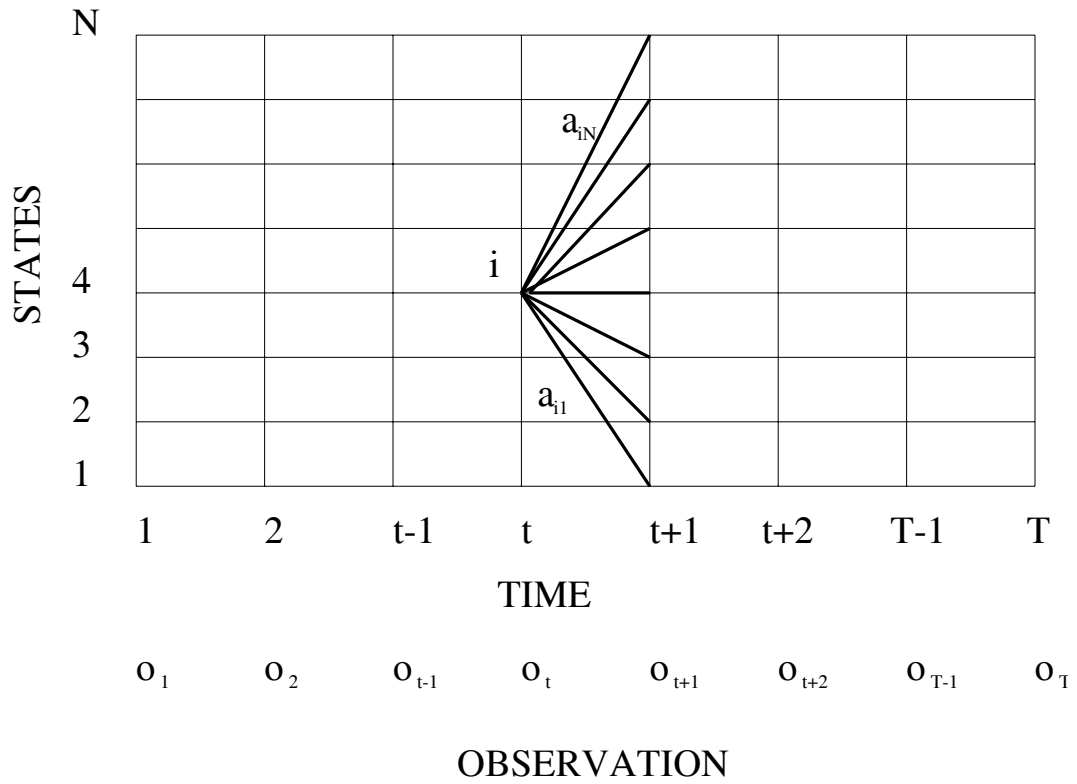
$$\alpha_{t+1}(j) = \left[\sum_{i=1}^N \alpha_t(i) a_{ij} \right] b_j(o_{t+1})$$

3. Termination:

$$P(O | \lambda) = \sum_{i=1}^N \alpha_T(i)$$

- Complexity: $O(N^2T)$.

Backward Algorithm



Backward Algorithm

- Define backward variable $\beta_t(i)$ as:

$$\beta_t(i) = P(o_{t+1}, o_{t+2}, \dots, o_T | q_t = i, \lambda)$$

- $\beta_t(i)$ is the probability of observing the partial sequence $(o_{t+1}, o_{t+2}, \dots, o_T)$ such that the state q_t is i .

- Induction:

1. Initialization: $\beta_T(i) = 1$

2. Induction:

$$\beta_t(i) = \sum_{j=1}^N a_{ij} b_j(o_{t+1}) \beta_{t+1}(j),$$

$$1 \leq i \leq N,$$

$$t = T - 1, \dots, 1$$

Solution to Problem 2

- Choose the most likely path
- Find the path (q_1, q_2, \dots, q_T) that maximizes the likelihood:

$$P(q_1, q_2, \dots, q_T | O, \lambda)$$

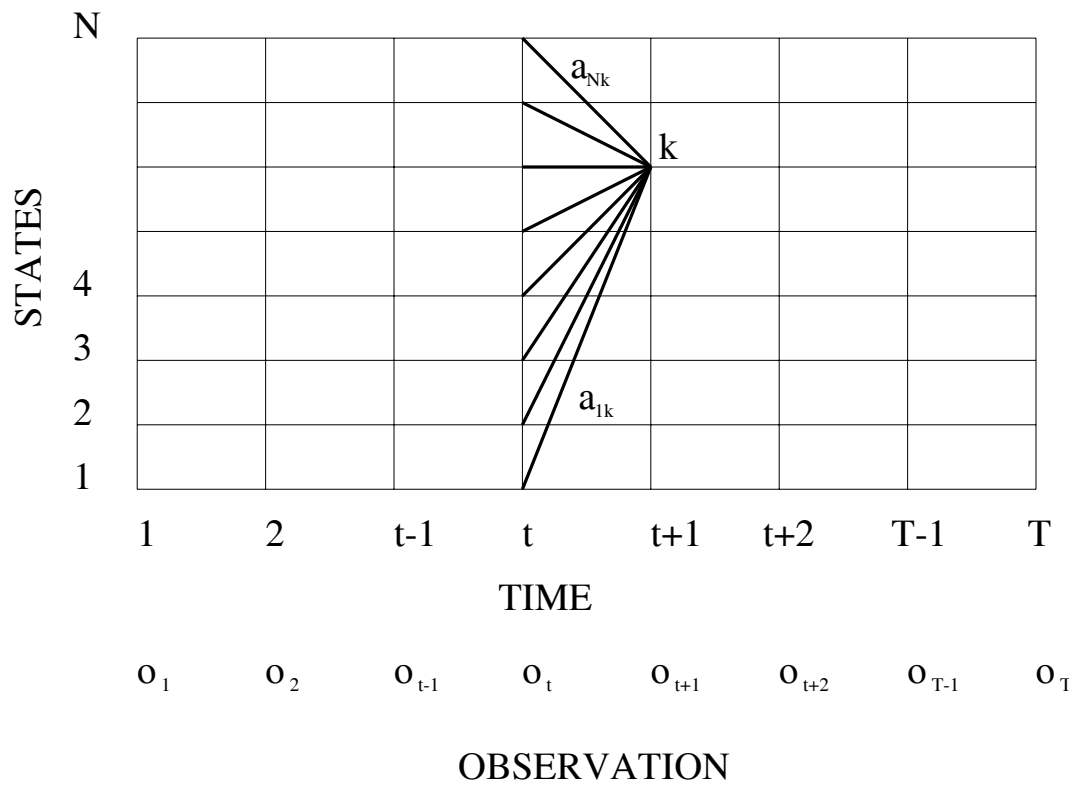
- Solution by Dynamic Programming
- Define:

$$\delta_t(i) = \max_{q_1, q_2, \dots, q_{t-1}} P(q_1, q_2, \dots, q_t = i, o_1, o_2, \dots, o_t | \lambda)$$

- $\delta_t(i)$ is the highest prob. path ending in state i
- By induction we have:

$$\delta_{t+1}(j) = \max_i [\delta_t(i) a_{ij}] \cdot b_j(o_{t+1})$$

Viterbi Algorithm



Viterbi Algorithm

- Initialization:

$$\delta_1(i) = \pi_i b_i(o_1), \quad 1 \leq i \leq N$$

$$\psi_1(i) = 0$$

- Recursion:

$$\delta_t(j) = \max_{1 \leq i \leq N} [\delta_{t-1}(i) a_{ij}] b_j(o_t)$$

$$\psi_t(j) = \arg \max_{1 \leq i \leq N} [\delta_{t-1}(i) a_{ij}]$$

$$2 \leq t \leq T, 1 \leq j \leq N$$

- Termination:

$$P^* = \max_{1 \leq i \leq N} [\delta_T(i)]$$

$$q_T^* = \arg \max_{1 \leq i \leq N} [\delta_T(i)]$$

- Path (state sequence) backtracking:

$$q_t^* = \psi_{t+1}(q_{t+1}^*), \quad t = T - 1, T - 2, \dots, 1$$

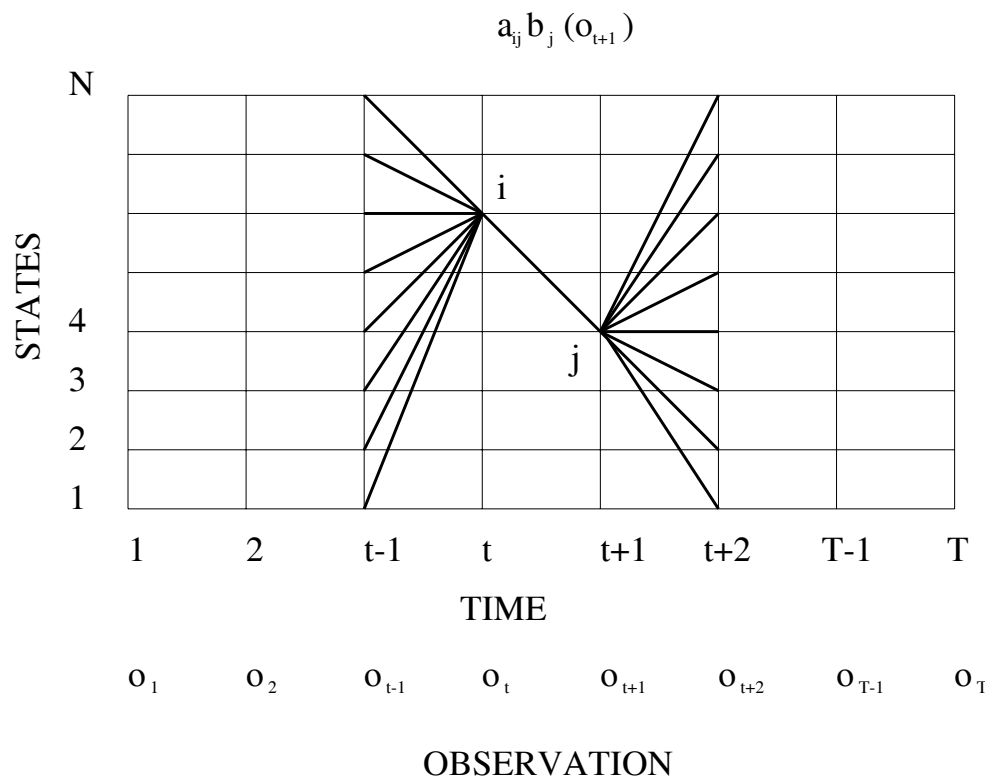
Solution to Problem 3

- Estimate $\lambda = (A, B, \pi)$ to maximize $P(O|\lambda)$
- No analytic method because of complexity – iterative solution.
- $\xi(i, j)$ is the probability of being in state i at time t and in state j at time $t + 1$.

-

$$\begin{aligned} \xi(i, j) &= \frac{\alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)}{P(O|\lambda)} \\ &= \frac{\alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)}{\sum_{i=1}^N \sum_{j=1}^N \alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)} \end{aligned}$$

Baum-Welch Algorithm



Baum-Welch Algorithm

- Define $\gamma_t(i)$ as prob. of being in state i at time t , given the observation sequence.

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

- $\sum_{t=1}^T \gamma_t(i)$ is the expected number of times state i is visited.
- $\sum_{t=1}^{T-1} \xi_t(i, j)$ is the expected number of transitions from state i to state j .
- $\pi_i =$ expected frequency in state i at time ($t = 1$)
 $= \gamma_1(i)$.
- $a_{ij} =$ (expected number of transition from state i to state j) / (expected number of transitions from state i):

$$a_{ij} = \frac{\sum \xi_t(i, j)}{\sum \gamma_t(i)}$$

- $b_j(k)$ = (expected number of times in state j and observing symbol k) / (expected number of times in state j):

$$b_j(k) = \frac{\sum_{t, o_t=k} \gamma_t(i)}{\sum_t \gamma_t(i)}$$

Properties

- Covariance of the estimated parameters
- Convergence rates

Types of HMM

- Continuous density
- Ergodic
- State duration

Implementation Issues

- **Scaling**
- **Initial parameters**
- **Multiple observation**

Comparison of HMMs

- What is a natural distance function?
- If $\rho(\lambda_1, \lambda_2)$ is large, does it mean that the models are really different?