

Introduction to Hidden Markov Models

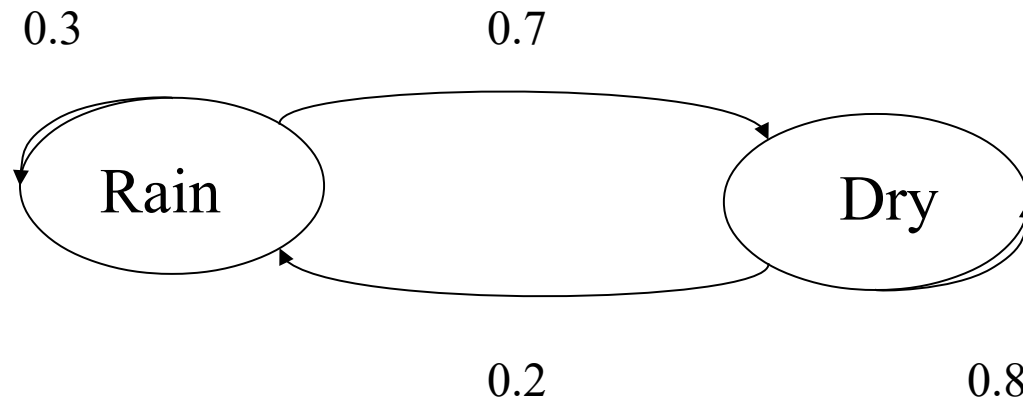
Markov Models

- Set of states: $\{s_1, s_2, \dots, s_N\}$
- Process moves from one state to another generating a sequence of states : $s_{i1}, s_{i2}, \dots, s_{ik}, \dots$
- Markov chain property: probability of each subsequent state depends only on what was the previous state:

$$P(s_{ik} \mid s_{i1}, s_{i2}, \dots, s_{ik-1}) = P(s_{ik} \mid s_{ik-1})$$

- To define Markov model, the following probabilities have to be specified: transition probabilities $a_{ij} = P(s_i \mid s_j)$ and initial probabilities $\pi_i = P(s_i)$

Example of Markov Model



- Two states : ‘Rain’ and ‘Dry’.
- Transition probabilities: $P(\text{‘Rain’}|\text{‘Rain’})=0.3$,
 $P(\text{‘Dry’}|\text{‘Rain’})=0.7$, $P(\text{‘Rain’}|\text{‘Dry’})=0.2$, $P(\text{‘Dry’}|\text{‘Dry’})=0.8$
- Initial probabilities: say $P(\text{‘Rain’})=0.4$, $P(\text{‘Dry’})=0.6$.

Calculation of sequence probability

- By Markov chain property, probability of state sequence can be found by the formula:

$$\begin{aligned} P(s_{i1}, s_{i2}, \dots, s_{ik}) &= P(s_{ik} \mid s_{i1}, s_{i2}, \dots, s_{ik-1}) P(s_{i1}, s_{i2}, \dots, s_{ik-1}) \\ &= P(s_{ik} \mid s_{ik-1}) P(s_{i1}, s_{i2}, \dots, s_{ik-1}) = \dots \\ &= P(s_{ik} \mid s_{ik-1}) P(s_{ik-1} \mid s_{ik-2}) \dots P(s_{i2} \mid s_{i1}) P(s_{i1}) \end{aligned}$$

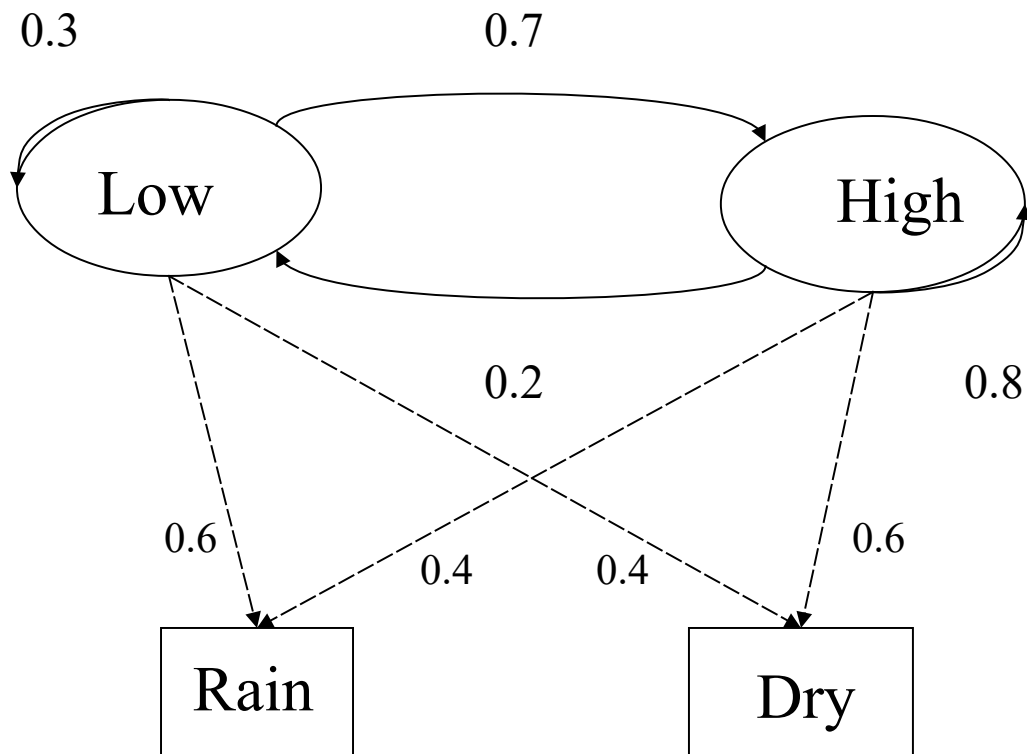
- Suppose we want to calculate a probability of a sequence of states in our example, {‘Dry’, ‘Dry’, ‘Rain’, ‘Rain’}.

$$\begin{aligned} P(\{\text{‘Dry’}, \text{‘Dry’}, \text{‘Rain’}, \text{‘Rain’}\}) &= \\ P(\text{‘Rain’} \mid \text{‘Rain’}) P(\text{‘Rain’} \mid \text{‘Dry’}) P(\text{‘Dry’} \mid \text{‘Dry’}) P(\text{‘Dry’}) &= \\ = 0.3 * 0.2 * 0.8 * 0.6 \end{aligned}$$

Hidden Markov models.

- Set of states: $\{s_1, s_2, \dots, s_N\}$
- Process moves from one state to another generating a sequence of states : $s_{i1}, s_{i2}, \dots, s_{ik}, \dots$
- Markov chain property: probability of each subsequent state depends only on what was the previous state:
$$P(s_{ik} \mid s_{i1}, s_{i2}, \dots, s_{ik-1}) = P(s_{ik} \mid s_{ik-1})$$
- States are not visible, but each state randomly generates one of M observations (or visible states) $\{v_1, v_2, \dots, v_M\}$
- To define hidden Markov model, the following probabilities have to be specified: matrix of transition probabilities $A=(a_{ij})$, $a_{ij}= P(s_i \mid s_j)$, matrix of observation probabilities $B=(b_i(v_m))$, $b_i(v_m)= P(v_m \mid s_i)$ and a vector of initial probabilities $\pi=(\pi_i)$, $\pi_i = P(s_i)$. Model is represented by $M=(A, B, \pi)$.

Example of Hidden Markov Model



Example of Hidden Markov Model

- Two states : 'Low' and 'High' atmospheric pressure.
- Two observations : 'Rain' and 'Dry'.
- Transition probabilities: $P(\text{'Low'}|\text{'Low'})=0.3$,
 $P(\text{'High'}|\text{'Low'})=0.7$, $P(\text{'Low'}|\text{'High'})=0.2$,
 $P(\text{'High'}|\text{'High'})=0.8$
- Observation probabilities : $P(\text{'Rain'}|\text{'Low'})=0.6$,
 $P(\text{'Dry'}|\text{'Low'})=0.4$, $P(\text{'Rain'}|\text{'High'})=0.4$,
 $P(\text{'Dry'}|\text{'High'})=0.3$.
- Initial probabilities: say $P(\text{'Low'})=0.4$, $P(\text{'High'})=0.6$.

Calculation of observation sequence probability

- Suppose we want to calculate a probability of a sequence of observations in our example, $\{\text{'Dry'}, \text{'Rain'}\}$.
- Consider all possible hidden state sequences:

$$\begin{aligned} P(\{\text{'Dry'}, \text{'Rain'}\}) &= P(\{\text{'Dry'}, \text{'Rain'}\}, \{\text{'Low'}, \text{'Low'}\}) + \\ &P(\{\text{'Dry'}, \text{'Rain'}\}, \{\text{'Low'}, \text{'High'}\}) + P(\{\text{'Dry'}, \text{'Rain'}\}, \\ &\{\text{'High'}, \text{'Low'}\}) + P(\{\text{'Dry'}, \text{'Rain'}\}, \{\text{'High'}, \text{'High'}\}) \end{aligned}$$

where first term is :

$$\begin{aligned} &P(\{\text{'Dry'}, \text{'Rain'}\}, \{\text{'Low'}, \text{'Low'}\}) = \\ &P(\{\text{'Dry'}, \text{'Rain'}\} \mid \{\text{'Low'}, \text{'Low'}\}) P(\{\text{'Low'}, \text{'Low'}\}) = \\ &P(\text{'Dry'} \mid \text{'Low'}) P(\text{'Rain'} \mid \text{'Low'}) P(\text{'Low'}) P(\text{'Low'} \mid \text{'Low'}) \\ &= 0.4 * 0.4 * 0.6 * 0.4 * 0.3 \end{aligned}$$

Main issues using HMMs :

Evaluation problem. Given the HMM $M=(A, B, \pi)$ and the observation sequence $O=o_1 o_2 \dots o_K$, calculate the probability that model M has generated sequence O .

• **Decoding problem.** Given the HMM $M=(A, B, \pi)$ and the observation sequence $O=o_1 o_2 \dots o_K$, calculate the most likely sequence of hidden states S_i that produced this observation sequence O .

• **Learning problem.** Given some training observation sequences $O=o_1 o_2 \dots o_K$ and general structure of HMM (numbers of hidden and visible states), determine HMM parameters $M=(A, B, \pi)$ that best fit training data.

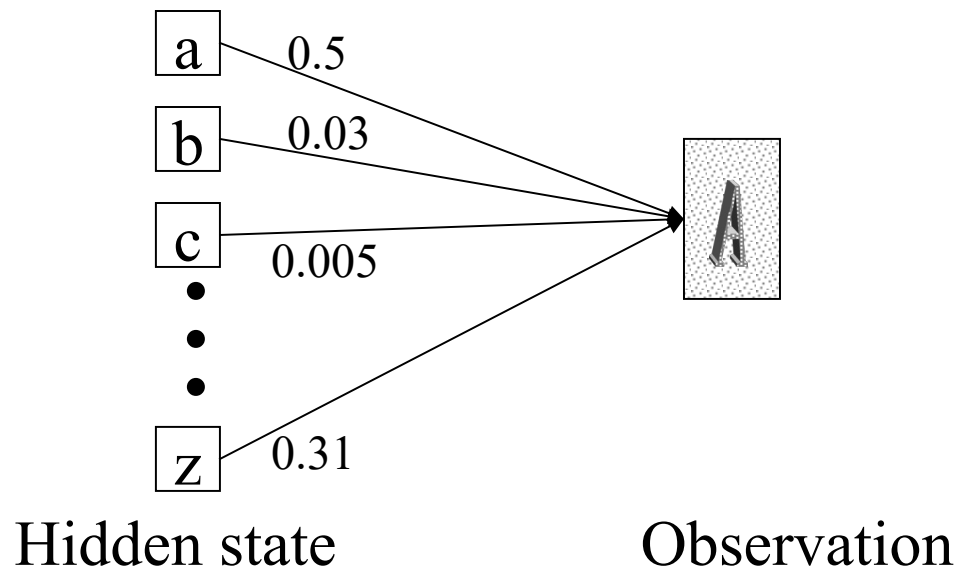
$O=o_1 \dots o_K$ denotes a sequence of observations $o_k \in \{v_1, \dots, v_M\}$.

Word recognition example(1).

- Typed word recognition, assume all characters are separated.



- Character recognizer outputs probability of the image being particular character, $P(\text{image}|\text{character})$.



Word recognition example(2).

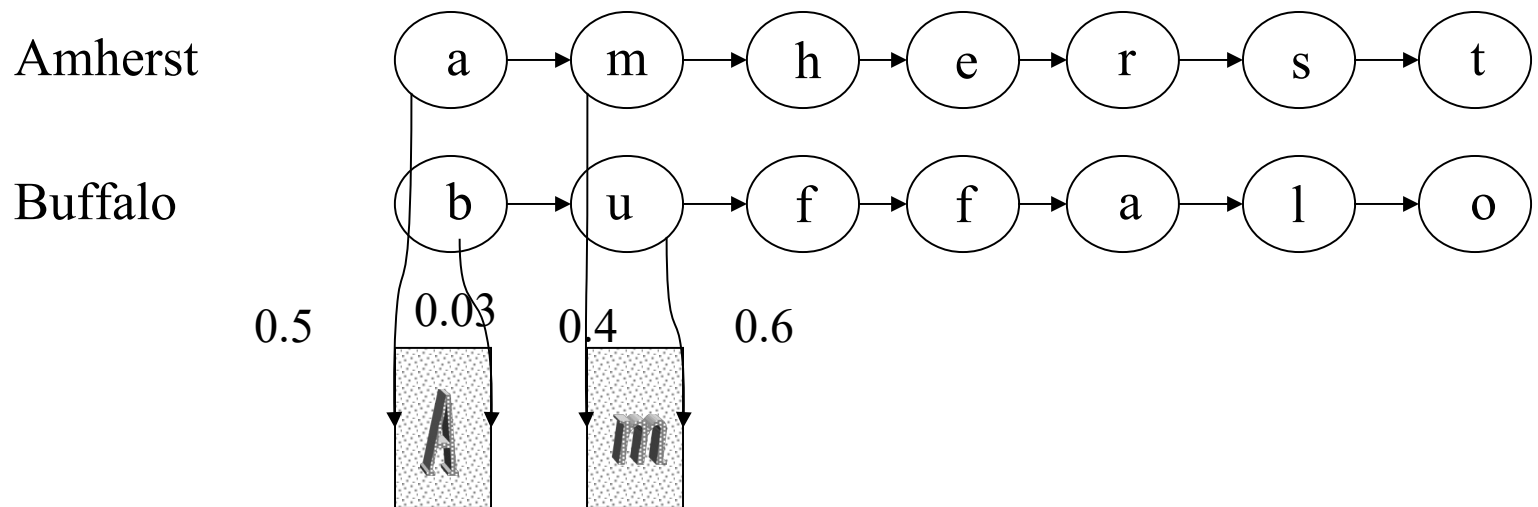
- Hidden states of HMM = characters.
- Observations = typed images of characters segmented from the image v_α . Note that there is an infinite number of observations
- Observation probabilities = character recognizer scores.

$$B = (b_i(v_\alpha)) = (P(v_\alpha | s_i))$$

- Transition probabilities will be defined differently in two subsequent models.

Word recognition example(3).

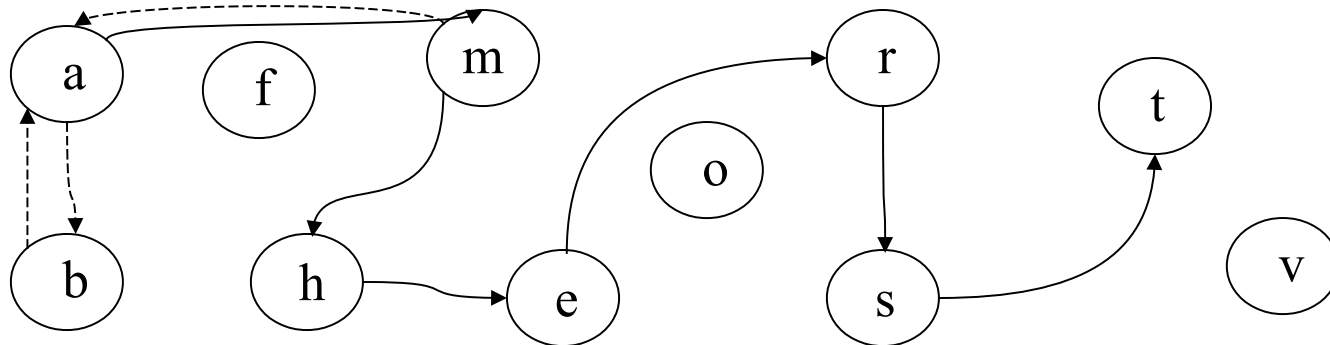
- If lexicon is given, we can construct separate HMM models for each lexicon word.



- Here recognition of word image is equivalent to the problem of evaluating few HMM models.
- This is an application of **Evaluation problem**.

Word recognition example(4).

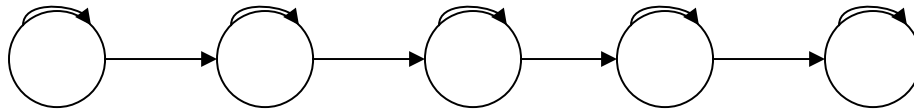
- We can construct a single HMM for all words.
- Hidden states = all characters in the alphabet.
- Transition probabilities and initial probabilities are calculated from language model.
- Observations and observation probabilities are as before.



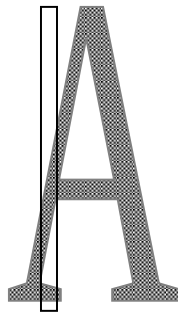
- Here we have to determine the best sequence of hidden states, the one that most likely produced word image.
- This is an application of **Decoding problem**.

Character recognition with HMM example.

- The structure of hidden states is chosen.



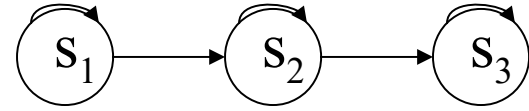
- Observations are feature vectors extracted from vertical slices.



- Probabilistic mapping from hidden state to feature vectors:
 1. use mixture of Gaussian models
 2. Quantize feature vector space.

Exercise: character recognition with HMM(1)

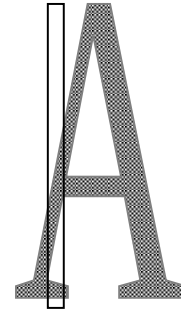
- The structure of hidden states:



- Observation = number of islands in the vertical slice.
- HMM for character 'A' :

$$\text{Transition probabilities: } \{a_{ij}\} = \begin{pmatrix} .8 & .2 & 0 \\ 0 & .8 & .2 \\ 0 & 0 & 1 \end{pmatrix}$$

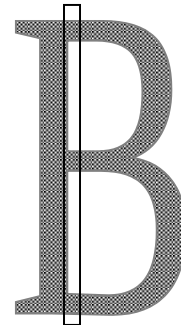
$$\text{Observation probabilities: } \{b_{jk}\} = \begin{pmatrix} .9 & .1 & 0 \\ .1 & .8 & .1 \\ .9 & .1 & 0 \end{pmatrix}$$



- HMM for character 'B' :

$$\text{Transition probabilities: } \{a_{ij}\} = \begin{pmatrix} .8 & .2 & 0 \\ 0 & .8 & .2 \\ 0 & 0 & 1 \end{pmatrix}$$

$$\text{Observation probabilities: } \{b_{jk}\} = \begin{pmatrix} .9 & .1 & 0 \\ 0 & .2 & .8 \\ .6 & .4 & 0 \end{pmatrix}$$



Exercise: character recognition with HMM(2)

- Suppose that after character image segmentation the following sequence of island numbers in 4 slices was observed:

$\{ 1, 3, 2, 1 \}$

- What HMM is more likely to generate this observation sequence , HMM for 'A' or HMM for 'B' ?

Exercise: character recognition with HMM(3)

Consider likelihood of generating given observation for each possible sequence of hidden states:

- HMM for character ‘A’:

Hidden state sequence	Transition probabilities	Observation probabilities
$S_1 \rightarrow S_1 \rightarrow S_2 \rightarrow S_3$	$.8 * .2 * .2$	$* .9 * 0 * .8 * .9 = 0$
$S_1 \rightarrow S_2 \rightarrow S_2 \rightarrow S_3$	$.2 * .8 * .2$	$* .9 * .1 * .8 * .9 = 0.0020736$
$S_1 \rightarrow S_2 \rightarrow S_3 \rightarrow S_3$	$.2 * .2 * 1$	$* .9 * .1 * .1 * .9 = 0.000324$
Total = 0.0023976		

- HMM for character ‘B’:

Hidden state sequence	Transition probabilities	Observation probabilities
$S_1 \rightarrow S_1 \rightarrow S_2 \rightarrow S_3$	$.8 * .2 * .2$	$* .9 * 0 * .2 * .6 = 0$
$S_1 \rightarrow S_2 \rightarrow S_2 \rightarrow S_3$	$.2 * .8 * .2$	$* .9 * .8 * .2 * .6 = 0.0027648$
$S_1 \rightarrow S_2 \rightarrow S_3 \rightarrow S_3$	$.2 * .2 * 1$	$* .9 * .8 * .4 * .6 = 0.006912$
Total = 0.0096768		

Evaluation Problem.

- **Evaluation problem.** Given the HMM $M=(A, B, \pi)$ and the observation sequence $O=o_1 o_2 \dots o_K$, calculate the probability that model M has generated sequence O .

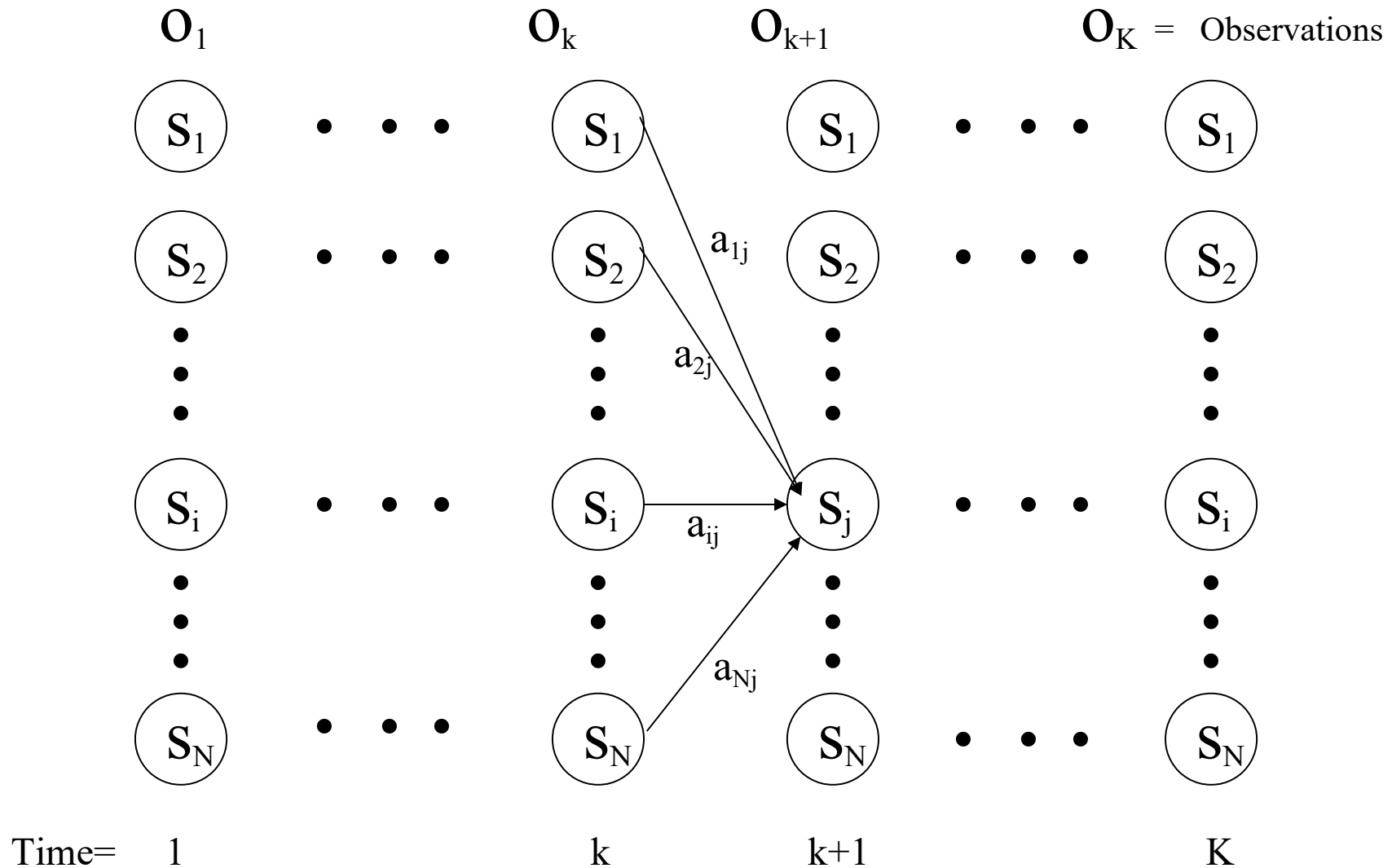
- Trying to find probability of observations $O=o_1 o_2 \dots o_K$ by means of considering all hidden state sequences (as was done in example) is impractical:

N^K hidden state sequences - exponential complexity.

- Use **Forward-Backward HMM algorithms** for efficient calculations.

- Define the forward variable $\alpha_k(i)$ as the joint probability of the partial observation sequence $o_1 o_2 \dots o_k$ and that the hidden state at time k is S_i : $\alpha_k(i) = P(o_1 o_2 \dots o_k, q_k = S_i)$

Trellis representation of an HMM



Forward recursion for HMM

- Initialization:

$$\alpha_1(i) = P(o_1, q_1 = s_i) = \pi_i b_i(o_1), \quad 1 \leq i \leq N.$$

- Forward recursion:

$$\begin{aligned} \alpha_{k+1}(j) &= P(o_1 o_2 \dots o_{k+1}, q_{k+1} = s_j) = \\ &= \sum_i P(o_1 o_2 \dots o_{k+1}, q_k = s_i, q_{k+1} = s_j) = \\ &= \sum_i P(o_1 o_2 \dots o_k, q_k = s_i) a_{ij} b_j(o_{k+1}) = \\ &= \left[\sum_i \alpha_k(i) a_{ij} \right] b_j(o_{k+1}), \quad 1 \leq j \leq N, 1 \leq k \leq K-1. \end{aligned}$$

- Termination:

$$P(o_1 o_2 \dots o_K) = \sum_i P(o_1 o_2 \dots o_K, q_K = s_i) = \sum_i \alpha_K(i)$$

- Complexity :

N^2K operations.

Backward recursion for HMM

- Define the forward variable $\beta_k(i)$ as the joint probability of the partial observation sequence $O_{k+1} O_{k+2} \dots O_K$ given that the hidden state at time k is s_i : $\beta_k(i) = P(o_{k+1} o_{k+2} \dots o_K | q_k = s_i)$

- Initialization:

$$\beta_K(i) = 1, \quad 1 \leq i \leq N.$$

- Backward recursion:

$$\begin{aligned} \beta_k(j) &= P(o_{k+1} o_{k+2} \dots o_K | q_k = s_j) = \\ &\sum_i P(o_{k+1} o_{k+2} \dots o_K, q_{k+1} = s_i | q_k = s_j) = \\ &\sum_i P(o_{k+2} o_{k+3} \dots o_K | q_{k+1} = s_i) a_{ji} b_i(o_{k+1}) = \\ &\sum_i \beta_{k+1}(i) a_{ji} b_i(o_{k+1}), \quad 1 \leq j \leq N, 1 \leq k \leq K-1. \end{aligned}$$

- Termination:

$$\begin{aligned} P(o_1 o_2 \dots o_K) &= \sum_i P(o_1 o_2 \dots o_K, q_1 = s_i) = \\ &\sum_i P(o_1 o_2 \dots o_K | q_1 = s_i) P(q_1 = s_i) = \sum_i \beta_1(i) b_i(o_1) \pi_i \end{aligned}$$

Decoding problem

- **Decoding problem.** Given the HMM $M=(A, B, \pi)$ and the observation sequence $O=o_1 o_2 \dots o_K$, calculate the most likely sequence of hidden states S_i that produced this observation sequence.
- We want to find the state sequence $Q=q_1 \dots q_K$ which maximizes $P(Q \mid o_1 o_2 \dots o_K)$, or equivalently $P(Q, o_1 o_2 \dots o_K)$.
- Brute force consideration of all paths takes exponential time. Use efficient **Viterbi algorithm** instead.
- Define variable $\delta_k(i)$ as the maximum probability of producing observation sequence $o_1 o_2 \dots o_k$ when moving along any hidden state sequence $q_1 \dots q_{k-1}$ and getting into $q_k = S_i$.

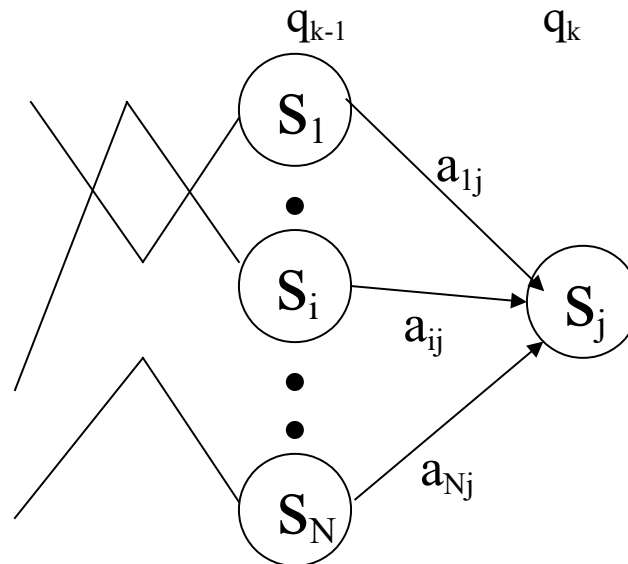
$$\delta_k(i) = \max P(q_1 \dots q_{k-1}, q_k = S_i, o_1 o_2 \dots o_k)$$

where max is taken over all possible paths $q_1 \dots q_{k-1}$.

Viterbi algorithm (1)

- General idea:

if best path ending in $q_k = S_j$ goes through $q_{k-1} = S_i$ then it should coincide with best path ending in $q_{k-1} = S_i$.



- $$\delta_k(i) = \max P(q_1 \dots q_{k-1}, q_k = S_j, o_1 o_2 \dots o_k) =$$

$$\max_i [a_{ij} b_j(o_k) \max P(q_1 \dots q_{k-1} = S_i, o_1 o_2 \dots o_{k-1})]$$

- To backtrack best path keep info that predecessor of S_j was S_i .

Viterbi algorithm (2)

- Initialization:

$$\delta_1(i) = \max P(q_1 = s_i, o_1) = \pi_i b_i(o_1), \quad 1 \leq i \leq N.$$

- Forward recursion:

$$\begin{aligned} \delta_k(j) &= \max P(q_1 \dots q_{k-1}, q_k = s_j, o_1 o_2 \dots o_k) = \\ &= \max_i [a_{ij} b_j(o_k) \max P(q_1 \dots q_{k-1} = s_i, o_1 o_2 \dots o_{k-1})] = \\ &= \max_i [a_{ij} b_j(o_k) \delta_{k-1}(i)], \quad 1 \leq j \leq N, 2 \leq k \leq K. \end{aligned}$$

- Termination: choose best path ending at time K

$$\max_i [\delta_K(i)]$$

- Backtrack best path.

This algorithm is similar to the forward recursion of evaluation problem, with Σ replaced by max and additional backtracking.

Learning problem (1)

- **Learning problem.** Given some training observation sequences $O = o_1 o_2 \dots o_K$ and general structure of HMM (numbers of hidden and visible states), determine HMM parameters $M = (A, B, \pi)$ that best fit training data, that is maximizes $P(O | M)$.
- There is no algorithm producing optimal parameter values.
- Use iterative expectation-maximization algorithm to find local maximum of $P(O | M)$ - **Baum-Welch algorithm**.

Learning problem (2)

- If training data has information about sequence of hidden states (as in word recognition example), then use maximum likelihood estimation of parameters:

$$a_{ij} = P(s_i | s_j) = \frac{\text{Number of transitions from state } S_j \text{ to state } S_i}{\text{Number of transitions out of state } S_j}$$

$$b_i(v_m) = P(v_m | s_i) = \frac{\text{Number of times observation } V_m \text{ occurs in state } S_i}{\text{Number of times in state } S_i}$$

Baum-Welch algorithm

General idea:

$$a_{ij} = P(s_i | s_j) = \frac{\text{Expected number of transitions from state } S_j \text{ to state } S_i}{\text{Expected number of transitions out of state } S_j}$$

$$b_i(v_m) = P(v_m | s_i) = \frac{\text{Expected number of times observation } V_m \text{ occurs in state } S_i}{\text{Expected number of times in state } S_i}$$

$$\pi_i = P(s_i) = \text{Expected frequency in state } S_i \text{ at time } k=1.$$

Baum-Welch algorithm: expectation step(1)

- Define variable $\xi_k(i,j)$ as the probability of being in state S_i at time k and in state S_j at time $k+1$, given the observation sequence $O_1 O_2 \dots O_K$.

$$\xi_k(i,j) = P(q_k = s_i, q_{k+1} = s_j \mid o_1 o_2 \dots o_K)$$

$$\xi_k(i,j) = \frac{P(q_k = s_i, q_{k+1} = s_j, o_1 o_2 \dots o_K)}{P(o_1 o_2 \dots o_K)} =$$

$$\frac{P(q_k = s_i, o_1 o_2 \dots o_k) a_{ij} b_j(o_{k+1}) P(o_{k+2} \dots o_K \mid q_{k+1} = s_j)}{P(o_1 o_2 \dots o_K)} =$$

$$\frac{\alpha_k(i) a_{ij} b_j(o_{k+1}) \beta_{k+1}(j)}{\sum_i \sum_j \alpha_k(i) a_{ij} b_j(o_{k+1}) \beta_{k+1}(j)}$$

Baum-Welch algorithm: expectation step(2)

- Define variable $\gamma_k(i)$ as the probability of being in state S_i at time k , given the observation sequence $O_1 O_2 \dots O_K$.

$$\gamma_k(i) = P(q_k = s_i \mid o_1 o_2 \dots o_K)$$

$$\gamma_k(i) = \frac{P(q_k = s_i, o_1 o_2 \dots o_k)}{P(o_1 o_2 \dots o_k)} = \frac{\alpha_k(i) \beta_k(i)}{\sum_i \alpha_k(i) \beta_k(i)}$$

Baum-Welch algorithm: expectation step(3)

- We calculated $\xi_k(i,j) = P(q_k = s_i, q_{k+1} = s_j \mid o_1 o_2 \dots o_K)$
and $\gamma_k(i) = P(q_k = s_i \mid o_1 o_2 \dots o_K)$
- Expected number of transitions from state S_i to state $S_j =$
$$= \sum_k \xi_k(i,j)$$
- Expected number of transitions out of state $S_i = \sum_k \gamma_k(i)$
- Expected number of times observation V_m occurs in state $S_i =$
$$= \sum_k \gamma_k(i), \text{ k is such that } o_k = V_m$$
- Expected frequency in state S_i at time $k=1$: $\gamma_1(i)$.

Baum-Welch algorithm: maximization step

$$a_{ij} = \frac{\text{Expected number of transitions from state } S_j \text{ to state } S_i}{\text{Expected number of transitions out of state } S_j} = \frac{\sum_k \xi_k(i,j)}{\sum_k \gamma_k(i)}$$

$$b_i(v_m) = \frac{\text{Expected number of times observation } v_m \text{ occurs in state } S_i}{\text{Expected number of times in state } S_i} = \frac{\sum_k \xi_k(i,j)}{\sum_{k, o_k = v_m} \gamma_k(i)}$$

$$\pi_i = (\text{Expected frequency in state } S_i \text{ at time } k=1) = \gamma_1(i).$$