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:

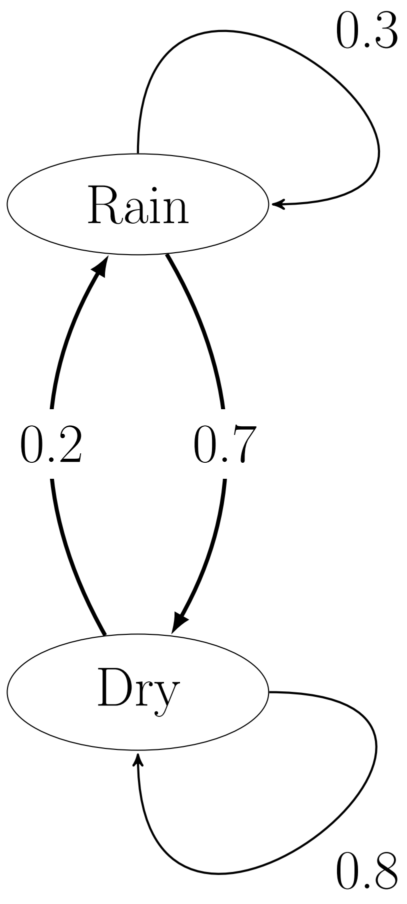
- probability of each subsequent state depends only on the previous state

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

To define Markov model, the following probabilities have to be specified:

- transition probabilities: $a_{ij} = P(s_i \mid s_j)$
- initial probabilities $\pi_i = P(s_i)$

Markov Models



Two states:

- $s_1 = Rain$ and $s_2 = Dry$

Transition probabilities:

- $P(Rain \mid Rain) = 0.3$
- $P(Dry \mid Rain) = 0.7$
- $P(Rain \mid Dry) = 0.2$
- $P(Dry \mid Dry) = 0.8$

Initial probabilities:

- $P(Rain) = 0.4$
- $P(Dry) = 0.6$

Calculation of sequence probability

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

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

Suppose we want to calculate a probability of a sequence of states in our example (Dry, Dry, Rain, Rain).

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

Hidden Markov Models

Set of states: $\{s_1, s_2, \ldots, s_N\}$

Process moves from one state to another generating a sequence of states:

$$(s_{i1}, s_{i2}, \ldots, s_{ik}, \ldots)$$

Markov chain property: probability of each subsequent state depends only on what was the previous state:

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

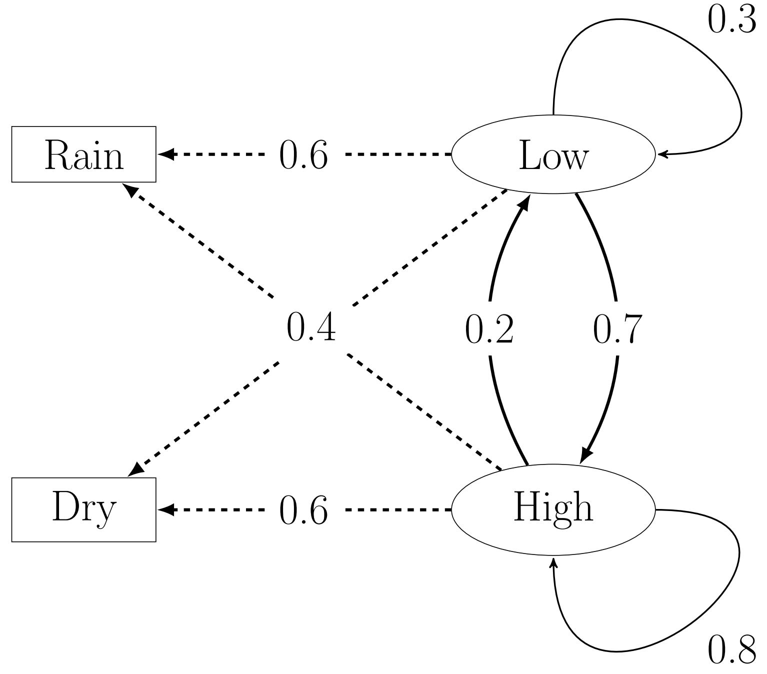
States are not visible, but each state randomly generates one of M observations (or visible states):

$$\{v_1, v_2, \ldots, v_M\}$$

The Hidden Markov Model $M = (A, B, \pi)$ is defined by the specification of the following probabilities:

◦ matrix of transition probabilities	$A = (a_{ij})$,	$a_{ij} = P(s_i \mid s_j)$
◦ matrix of observation probabilities	$B = (b_i(v_m))$,	$b_i(v_m) = P(v_m \mid s_i)$
◦ a vector of initial probabilities	$\pi = (\pi_i)$,	$\pi_i = P(s_i)$

Example of a Hidden Markov Model



Two states:

- *Low* and *High* atmospheric pressure.

Two observations : *Rain* and *Dry*.

Transition probabilities:

- $P(Low \mid Low) = 0.3$
- $P(High \mid Low) = 0.7$
- $P(Low \mid High) = 0.2$
- $P(High \mid High) = 0.8$

Observation probabilities:

- $P(Rain \mid Low) = 0.6$
- $P(Dry \mid Low) = 0.4$
- $P(Rain \mid High) = 0.4$
- $P(Dry \mid High) = 0.3$

Initial probabilities:

- $P(Low) = 0.4$, $P(High) = 0.6$

Calculation of Observation Sequence Probability

Suppose we want to calculate a probability of a sequence of observations in our example, $(Dry, Rain)$.

Consider all possible hidden state sequences:

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

where the first term is:

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

Main issues using HMMs

Let $O = (o_1, \ldots, o_K)$ denote a sequence of observations $o_k \in \{v_1, \ldots, v_M\}$ we define the following problems:

Evaluation Problem

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

Decoding Problem

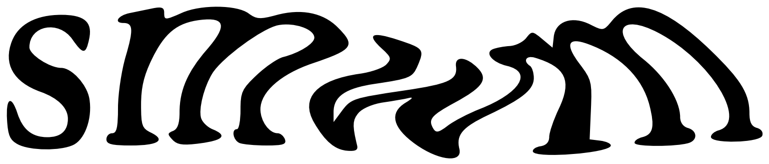
- Given the HMM $M = (A, B, \pi)$ and the observation sequence $O = (o_1, o_2, \ldots, o_K)$, calculate the most likely sequence of hidden states s_i that produced this observation sequence O.

Learning problem

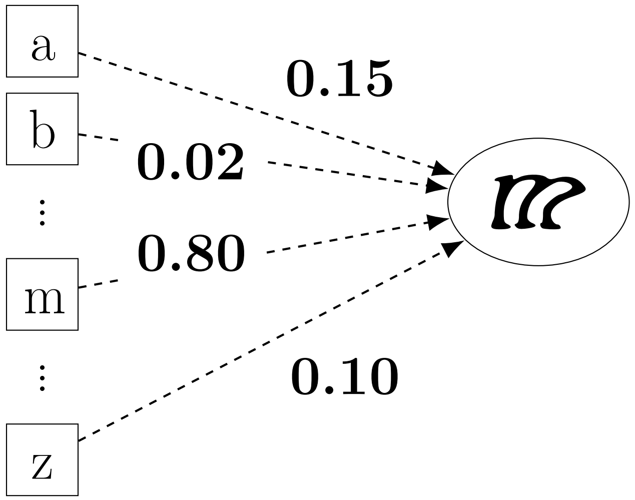
- Given some training observation sequences $O = (o_1, o_2, \ldots, o_K)$ and general structure of HMM (numbers of hidden and visible states), determine HMM parameters $M = (A, B, \pi)$ that fits the training data best.

Example Word Recognition

Typed word recognition, assume all characters are separated.



Character recognizer outputs probability of the image being particular character, $P(image|character)$.



Example Word Recognition

Hidden states of HMM: characters.

Observations: typed images of characters segmented from the images v_α. Note that there is an infinite number of observations.

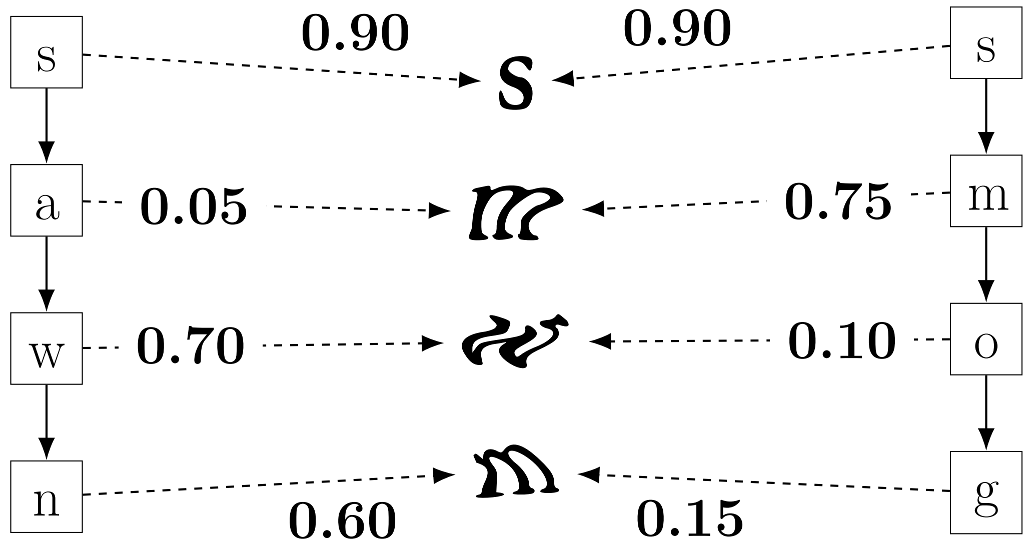
Observation probabilities: character recognizer scores

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

Transition probabilities will be defined differently in two subsequent models.

Example Word Recognition

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 the **Evaluation problem**.

Example Word Recognition

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.

$$\begin{tikzcd}
\boxed{s} \arrow[r] & \boxed{m} \arrow[r, bend left] & \boxed{w} \arrow[l, bend left] & \boxed{a} & \boxed{b} & \boxed{c}
\end{tikzcd}$$

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:

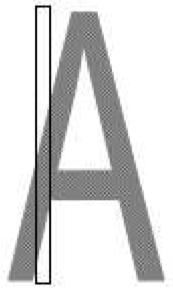
$$\begin{tikzcd} s_1 \arrow[loop above] \arrow[r] & s_2 \arrow[loop above] \arrow[r] & s_3 \arrow[loop above] \end{tikzcd}$$

Observation: number of islands in the vertical slice.

HMM for character A :

Transition probabilities: Observation probabilities:

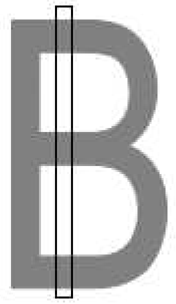
$$a_{ij} = \begin{pmatrix} 0.8 & 0.2 & 0.0 \\ 0.0 & 0.8 & 0.2 \\ 0.0 & 0.0 & 1.0 \end{pmatrix} \qquad b_{jk} = \begin{pmatrix} 0.9 & 0.1 & 0.0 \\ 0.1 & 0.8 & 0.1 \\ 0.9 & 0.1 & 0.0 \end{pmatrix}$$



HMM for character B :

Transition probabilities: Observation probabilities:

$$a_{ij} = \begin{pmatrix} 0.8 & 0.2 & 0.0 \\ 0.0 & 0.8 & 0.2 \\ 0.0 & 0.0 & 1.0 \end{pmatrix} \qquad b_{jk} = \begin{pmatrix} 0.9 & 0.1 & 0.0 \\ 0.0 & 0.2 & 0.8 \\ 0.6 & 0.4 & 0.0 \end{pmatrix}$$



Evaluation Problem

Evaluation Problem.

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

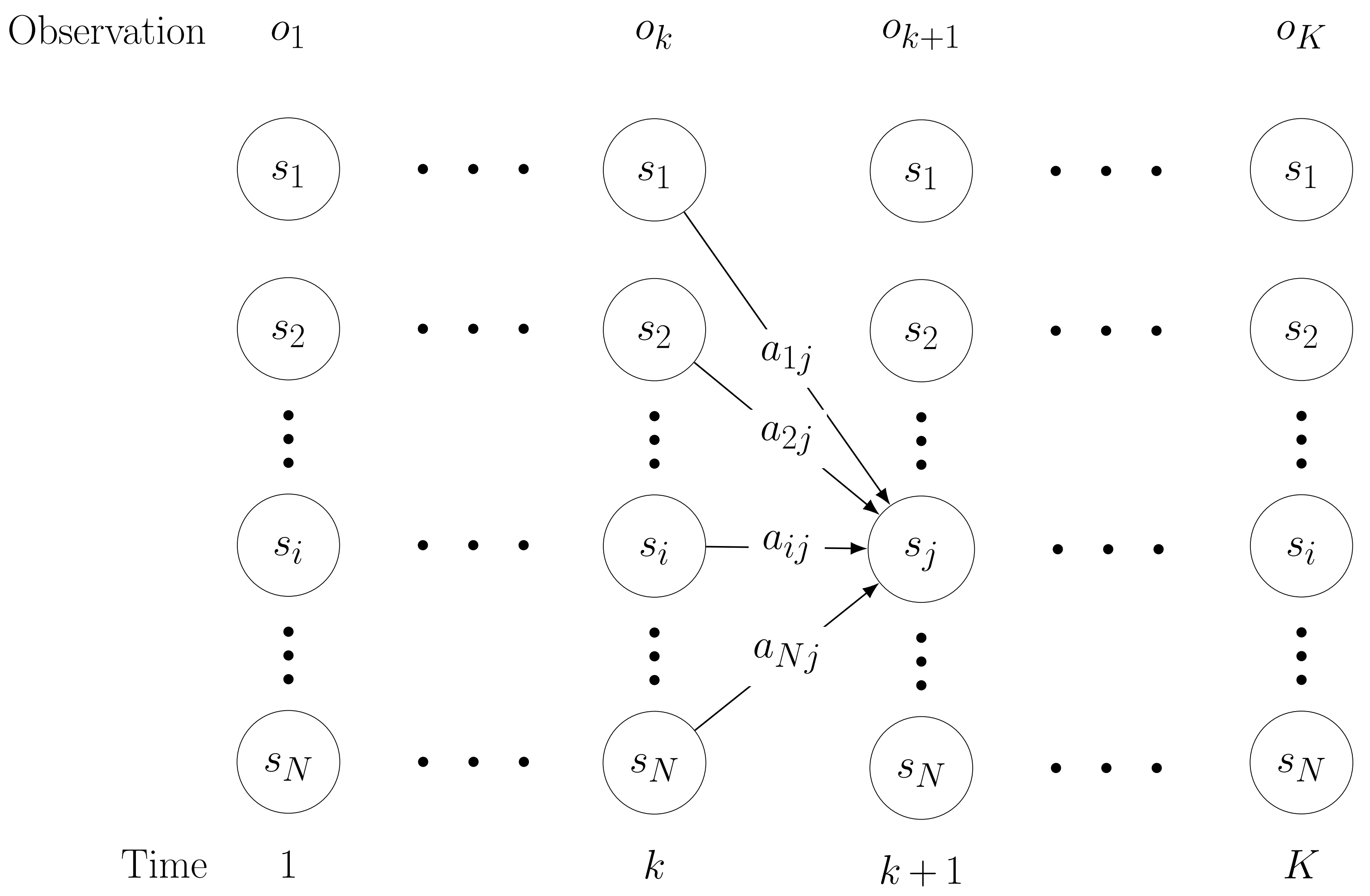
Trying to find probability of observations $O = (o_1, o_2, \ldots, o_K)$ by means of considering all hidden state sequences is impractical (N^K hidden state sequences)
$\Rightarrow$ exponential complexity

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

Define the forward variable $\alpha_k(i)$ as the joint probability of the partial observation sequence $(o_1, o_2, \ldots, o_k)$ and that the hidden state at time k is s_i:

$$\alpha(i) = P(\ (o_1, o_2, \ldots, o_k), q_k = s_i)$$

Evaluation Problem



Forward Recursion for HMM

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

2. **Forward Recursion**

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

3. **Termination**

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

4. **Complexity:** N^2K operations

Backward Recursion for HMM

Define the forward variable $\beta_k(i)$ as the joint probability of the partial observation sequence $o_{k+1}, o_{k+2}, \ldots, o_K$ given that the hidden state at time k is s_i:

$$\beta_k(i) = P(\ (o_{k+1}, o_{k+2}, \ldots, o_K) \mid q_k = s_i)$$

1. **Initialization** $\quad \beta_K(i) = 1, \quad 1 \le i \le N$
2. **Backward Recursion**

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

3. **Termination**

$$\begin{aligned}
P(\ (o_1, o_2, \ldots, o_K)) &= \sum_i P(\ (o_1, o_2, \ldots, o_K), q_1 = s_i) \\
&= \sum_i P(\ (o_1, o_2, \ldots, o_K) \mid q_1 = s_i) P(q_1 = s_1) = \sum_i \beta_1(i)\ b_i(o_1)\ \pi_i
\end{aligned}$$

Decoding Problem

Decoding Problem

- Given the HMM $M = (A, B, \pi)$ and the observation sequence $O = o_1, o_2, \ldots, o_K$, calculate the most likely sequence of hidden states s_i that produced this observation sequence.

We want to find the state sequence $Q = (q_1, \ldots, q_K)$ which maximizes:

$$P(Q \mid (o_1, o_2, \ldots, o_K)), \text{ or equivalently } P(Q, (o_1, o_2, \ldots, o_K))$$

Brute force consideration of all paths takes exponential time. Use efficient **Viterbi algorithm** instead.

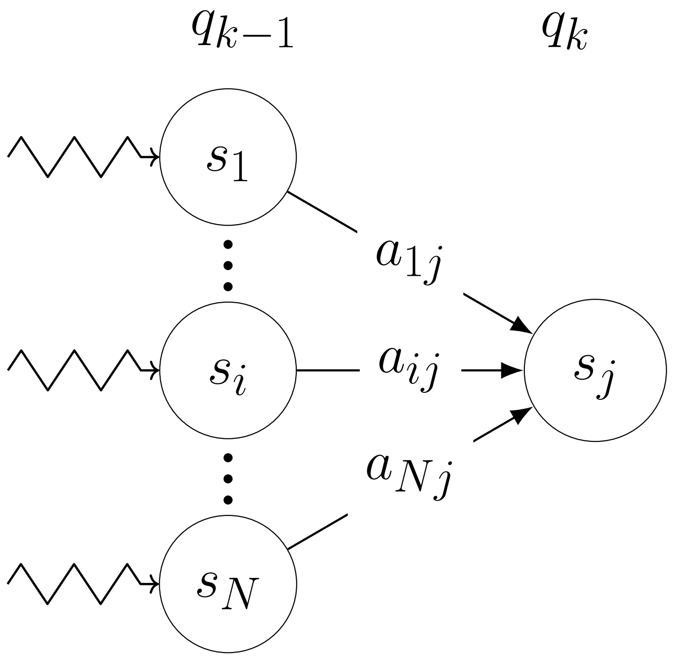
Define variable $\delta_k(i)$ as the maximum probability of producing observation sequence $(o_1, o_2, \ldots, o_k)$ when moving along any hidden state sequence $q_1, \ldots, q_{k-1}$ and getting into $q_k = s_i$.

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

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

Viterbi Algorithm

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 $q_{k-1} = s_i$.



$$\begin{aligned}\delta_k(i) &= \max P(q_1, \ldots, q_{k-1}, q_k = s_j, (o_1, o_2, \ldots, o_k)) \\ &= \max_i [a_{ij}\ b_j(o_k)\ \max P(q_1, \ldots, q_{k-1} = s_i, (o_1, o_2, \ldots, o_{k-1}))]\end{aligned}$$

To backtrack best path keep info that predecessor of s_j was s_i.

Viterbi Algorithm

1. **Initialization**

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

2. **Forward Recursion**

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

3. **Termination** choose best path ending at time K

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

4. Backtrack best path

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

Learning Problem

Learning Problem

- Given some training observation sequences $O = (o_1, o_2, \ldots, o_K)$ 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 $P(O \mid M)$.

There is no algorithm producing optimal parameter values.

Use iterative expectation-maximization algorithm to find local maximum of $P(O|M)$

$\Rightarrow$ **Baum-Welch algorithm**

Learning Problem

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

$$\begin{aligned} a_{ij} &= P(s_i \mid s_j) \\ &= \frac{\text{Number of transitions from state } s_j \text{ to state } s_i}{\text{Number of transitions out of state } s_j} \end{aligned}$$

$$\begin{aligned} b_i(v_m) &= P(v_m \mid s_i) \\ &= \frac{\text{Number of times observation } v_m \text{ occurs in state } s_i}{\text{Number of times in state } s_i} \end{aligned}$$

Baum-Welch algorithm

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

$$a_{ij} = P(s_i \mid s_j)$$

$$= \frac{\text{Expected number of transitions from state } s_j \text{ to state } s_i}{\text{Expected number of transitions out of state } s_j}$$

$$b_i(v_m) = P(v_m \mid s_i)$$

$$= \frac{\text{Expected number of times observation } v_m \text{ occurs in state } s_i}{\text{Expected number of times in state } s_i}$$

$$\pi_i = P(s_i)$$

$$= \text{Expected frequency in state } s_i \text{ at time } k = 1$$

Baum-Welch algorithm: expectation step

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 $o_1, o_2, \ldots, o_K$.

$$\begin{aligned}
\xi_{ij} &= P(q_k = s_i, q_{k+1} = s_j \mid (o_1, o_2, \ldots, o_k)) \\
&= \frac{P(q_k = s_i, q_{k+1} = s_j, (o_1, o_2, \ldots, o_k))}{P(\ (o_1, o_2, \ldots, o_k))} \\
&= \frac{P(q_k = s_i, (o_1, o_2, \ldots, o_k))\ a_{ij}\ b_j(o_{k+1})\ P(o_{k+2}, \ldots, o_K \mid q_{k+1} = s_j)}{P(\ (o_1, o_2, \ldots, o_k))} \\
&= \frac{\alpha_k(i)\ a_{ij}\ b_j(o_{k+1})\ \beta_{k+1}(j)}{\sum_i \sum_j \alpha_k(i)\ a_{ij}\ b_j(o_{k+1})\ \beta_{k+1}(j)}
\end{aligned}$$

Baum-Welch algorithm: expectation step

Define variable $\gamma_k(i)$ as the probability of being in state s_i at time k, given the observation sequence $o_1, o_2, \ldots, o_K$.

$$\begin{aligned}
\gamma_k(i) &= P(q_k = s_i \mid (o_1, o_2, \ldots, o_K)) \\
&= \frac{P(q_k = s_i, (o_1, o_2, \ldots, o_K))}{P(\ (o_1, o_2, \ldots, o_k))} \\
&= \frac{a_k(i)\beta_k(i)}{\sum_i \alpha_k(i)\beta_k(i)}
\end{aligned}$$

Baum-Welch algorithm: expectation step

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 is equal is represented by $\sum_k \xi_k(i,j)$

Expected number of transitions out of state s_i is represented by $\sum_k \gamma_k(i)$

Expected number of times observation v_m occurs in state s_i is equal to $\sum_k \gamma_k(i)$, 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

$$a_{ij} = \frac{\text{Expected number of transitions from state } s_j \text{ to state } s_i}{\text{Expected number of transitions out of state } s_j}$$

$$= \frac{\sum_k \xi_k(i,j)}{\sum_k \gamma_k(i)}$$

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

$$= \frac{\sum_k \xi_k(i,j)}{\sum_{k,o_k=v_m} \gamma_k(i)}$$

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