universität innsbruck

Hidden Markov Models

703075. Machine Learning

https://iis.uibk.ac.at/courses/2020s/703075

assoz.Prof. Antonio Rodríguez-Sánchez

Hidden Markov models

Introduction

Discrete Markov process

Observable Markov model

Hidden Markov model (HMM)

Solving HMMs

Hidden Markov models

Introduction

Discrete Markov process

Observable Markov model

Hidden Markov model (HMM)

Solving HMMs

Introduction

- Modeling dependencies in input; no longer iid
- Sequences:
  - Temporal: In speech; phonemes in a word (dictionary), words in a sentence (syntax, semantics of the language).
    - In handwriting, pen movements
  - Spatial: In a DNA sequence; base pairs

Introduction

- Modeling dependencies in input; no longer iid
- Sequences:
  - A sequence can be characterized as being generated by a *parametric random process*

Hidden Markov models

Introduction

Discrete Markov process

Observable Markov model

Hidden Markov model (HMM)

Solving HMMs

Discrete Markov Process

- Consider a system as with

  N states: $S_1, S_2, \ldots, S_N$ State at "time" t, $q_t = S_i$

Discrete Markov Process

- Consider a system as with

  N states: $S_1, S_2, \ldots, S_N$ State at "time" t, $q_t = S_i$

- The system moves to a state with a probability

$$P(q_{t+1}=S_j \mid q_t=S_i, q_{t-1}=S_k, \ldots)$$

Discrete Markov Process

- Consider a system as with

  N states: $S_1, S_2, \ldots, S_N$ State at "time" t, $q_t = S_i$

- The system moves to a state with a probability

$$P(q_{t+1}=S_j \mid q_t=S_i, q_{t-1}=S_k, \ldots)$$

- First-order Markov

$$P(q_{t+1}=S_j \mid q_t=S_i, q_{t-1}=S_k, \ldots) = P(q_{t+1}=S_j \mid q_t=S_i)$$

Discrete Markov Process

- Consider a system as with

  N states: $S_1, S_2, \ldots, S_N$ State at "time" t, $q_t = S_i$

- First-order Markov

  $$P(q_{t+1}=S_j \mid q_t=S_i, q_{t-1}=S_k, \ldots) = P(q_{t+1}=S_j \mid q_t=S_i)$$

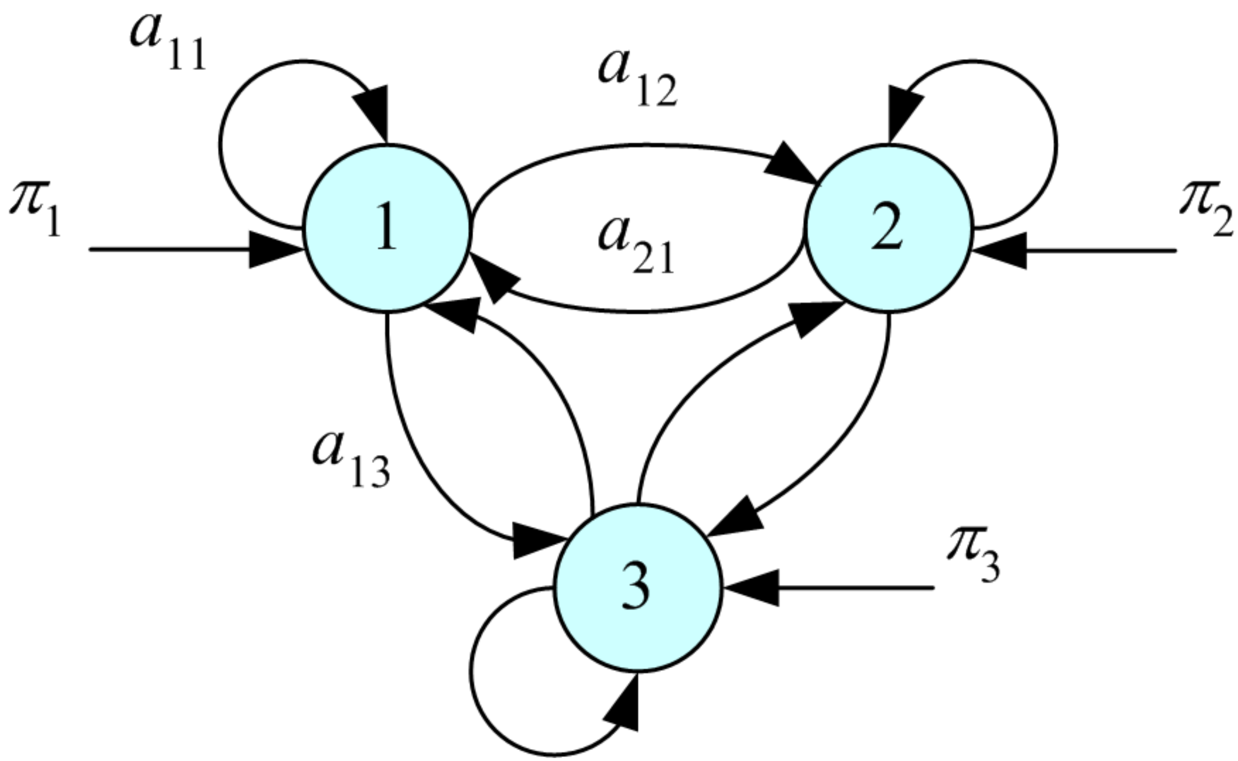
- Transition probabilities

  $$a_{ij} \equiv P(q_{t+1}=S_j \mid q_t=S_i) \qquad a_{ij} \geq 0 \text{ and } \Sigma_{j=1}^{N} a_{ij}=1$$

- Initial probabilities

  $$\pi_i \equiv P(q_1=S_i) \qquad \Sigma_{j=1}^{N} \pi_i=1$$

Stochastic Automation



- Transition probabilities

$$a_{ij} \equiv P(q_{t+1}=S_j \mid q_t=S_i) \qquad a_{ij} \geq 0 \text{ and } \Sigma_{j=1}{}^{N} a_{ij}=1$$

- Initial probabilities

$$\pi_i \equiv P(q_1=S_i) \qquad \Sigma_{j=1}{}^{N} \pi_i=1$$

Hidden Markov models

Introduction

Discrete Markov process

Observable Markov model

Hidden Markov model (HMM)

Solving HMMs

Observable Markov model

- The states are observable
  - At any time t we know q_t
  - Having an observation sequence

$$O = Q = \{q_1 q_2 \,\ldots.\, q_T\}$$

$$P(O = Q|\mathbf{A}, \mathbf{\Pi}) = P(q_1) \prod_{t=2}^{T} P(q_t|q_{t-1}) = \pi_{q_1} a_{q_1 q_2} \cdots a_{q_{T-1} q_T}$$

Example: Balls and Urns

- Rabiner and Juang (1986)
- Three urns each full of balls of one color

  S_1: red, S_2: blue, S_3: green

Example: Balls and Urns

- Rabiner and Juang (1986)
- Three urns each full of balls of one color

S_1: red, S_2: blue, S_3: green

$$\Pi = \left[0.5, 0.2, 0.3\right]^T \qquad \mathbf{A} = \begin{bmatrix} 0.4 & 0.3 & 0.3 \\ 0.2 & 0.6 & 0.2 \\ 0.1 & 0.1 & 0.8 \end{bmatrix}$$

Example: Balls and Urns

- Rabiner and Juang (1986)
- Three urns each full of balls of one color

S_1: red, S_2: blue, S_3: green

$$\Pi = [0.5, 0.2, 0.3]^T \qquad \mathbf{A} = \begin{bmatrix} 0.4 & 0.3 & 0.3 \\ 0.2 & 0.6 & 0.2 \\ 0.1 & 0.1 & 0.8 \end{bmatrix}$$

$$O = \{S_1, S_1, S_3, S_3\}$$

Example: Balls and Urns

- Rabiner and Juang (1986)
- Three urns each full of balls of one color

S_1: red, S_2: blue, S_3: green

$$\Pi = [0.5, 0.2, 0.3]^T \qquad \mathbf{A} = \begin{bmatrix} 0.4 & 0.3 & 0.3 \\ 0.2 & 0.6 & 0.2 \\ 0.1 & 0.1 & 0.8 \end{bmatrix}$$

$$O = \{S_1, S_1, S_3, S_3\}$$

$$P(O \mid \mathbf{A}, \Pi)$$

Example: Balls and Urns

- Rabiner and Juang (1986)
- Three urns each full of balls of one color

S_1: red, S_2: blue, S_3: green

$$\Pi = [0.5, 0.2, 0.3]^T \qquad \mathbf{A} = \begin{bmatrix} 0.4 & 0.3 & 0.3 \\ 0.2 & 0.6 & 0.2 \\ 0.1 & 0.1 & 0.8 \end{bmatrix}$$

$$O = \{S_1, S_1, S_3, S_3\}$$

$$P(O \mid \mathbf{A}, \Pi) = P(S_1) \cdot P(S_1 \mid S_1) \cdot P(S_3 \mid S_1) \cdot P(S_3 \mid S_3)$$

$$= \pi_1 \cdot a_{11} \cdot a_{13} \cdot a_{33}$$

$$= 0.5 \cdot 0.4 \cdot 0.3 \cdot 0.8 = 0.048$$

Example: Balls and Urns

- Learning
  - Given K example sequences of length T

$$\hat{\pi}_i = \frac{\#\{\text{sequences starting with } S_i\}}{\#\{\text{sequences}\}} = \frac{\sum_k 1\left(q_1^k = S_i\right)}{K}$$

Example: Balls and Urns

- Learning
  - Given K example sequences of length T

$$\hat{\pi}_i = \frac{\#\{\text{sequences starting with } S_i\}}{\#\{\text{sequences}\}} = \frac{\sum_k 1\left(q_1^k = S_i\right)}{K}$$

$$\hat{a}_{ij} = \frac{\#\{\text{transitions from } S_i \text{ to } S_j\}}{\#\{\text{transitions from } S_i\}}$$

$$= \frac{\sum_k \sum_{t=1}^{T\text{-}1} 1\left(q_t^k = S_i \text{ and } q_{t+1}^k = S_j\right)}{\sum_k \sum_{t=1}^{T\text{-}1} 1\left(q_t^k = S_i\right)}$$

Hidden Markov models

Introduction

Discrete Markov process

Observable Markov model

Hidden Markov model (HMM)

Solving HMMs

Hidden Markov Models

- States are not observable
- Discrete observations $\{v_1, v_2, \dots, v_M\}$ are recorded
  - A probabilistic function of the state

Hidden Markov Models

- States are not observable
- Discrete observations $\{v_1, v_2, \ldots, v_M\}$ are recorded
  - A probabilistic function of the state
- Emission probabilities
  - *Observation* that we observe
    v_m , $m = 1, \ldots, M$ in state S_j

$$b_j(m) \equiv P(O_t = v_m \mid q_t = S_j)$$

Hidden Markov Models

- States are not observable
- Discrete observations $\{v_1, v_2, \ldots, v_M\}$ are recorded
- Emission probabilities
  - *Observation* that we observe

    v_m , $m = 1, \ldots, M$ in state S_j

$$b_j(m) \equiv P(O_t = v_m \mid q_t = S_j)$$

- The state sequence Q is not observed
  - but it should be inferred from the observation sequence O

Example: Balls and Urns

- In each urn, there are balls of different colors, but with different probabilities.
  - For each observation sequence, there are multiple state sequences
  - $b_j(m) \equiv P(O_t=v_m \mid q_t=S_j)$ denotes the probability of drawing a ball of color m from urn j

Example: Balls and Urns

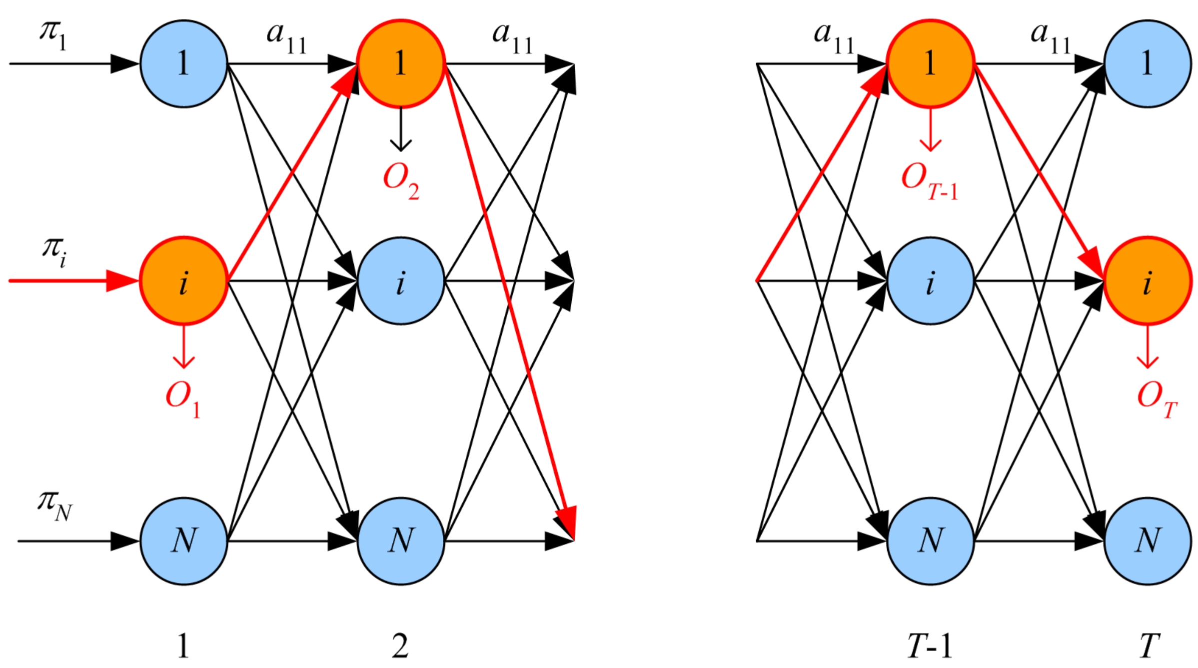
- In each urn, there are balls of different colors, but with different probabilities.
  - For each observation sequence, there are multiple state sequences
  - $b_j(m) \equiv P(O_t = v_m \mid q_t = S_j)$ denotes the probability of drawing a ball of color m from urn j
  - We again observe a sequence of ball colors but without knowing the sequence of urns from which the balls were drawn

Example: Balls and Urns

- In each urn, there are balls of different colors, but with different probabilities.
  - The observable model is a special case of the hidden model
  - where $M = N$
  - and $b_j(m)$ is 1 if $j = m$ and 0 otherwise

Example: Balls and Urns

- In each urn, there are balls of different colors, but with different probabilities.
  - For the same observation sequence O, there may be many possible state sequences Q that could have generated O



Elements of an HMM

- N: Number of states $S = \{S_1, S_2, \ldots, S_N\}$
- M: Number of observation symbols

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

Elements of an HMM

- N: Number of states $S = \{S_1, S_2, \ldots, S_N\}$
- M: Number of observation symbols

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

- $\mathbf{A} = [a_{ij}]$: N by N state transition probability matrix

$$\mathbf{A} = [a_{ij}] \text{ where } a_{ij} \equiv P(q_{t+1} = S_j | q_t = S_i)$$

- $\mathbf{B} = b_j(m)$: N by M observation probability matrix

$$\mathbf{B} = [b_j(m)] \text{ where } b_j(m) \equiv P(O_t = v_m | q_t = S_j)$$

- $\mathbf{\Pi} = [\pi_i]$: N by 1 initial state probability vector

$$\mathbf{\Pi} = [\pi_i] \text{ where } \pi_i \equiv P(q_1 = S_i)$$

Elements of an HMM

- $\mathbf{A} = [a_{ij}]$: N by N state transition probability matrix

$$\mathbf{A} = [a_{ij}] \text{ where } a_{ij} \equiv P(q_{t+1} = S_j | q_t = S_i)$$

- $\mathbf{B} = b_j(m)$: N by M observation probability matrix

$$\mathbf{B} = [b_j(m)] \text{ where } b_j(m) \equiv P(O_t = v_m | q_t = S_j)$$

- $\mathbf{\Pi} = [\pi_i]$: N by 1 initial state probability vector

$$\mathbf{\Pi} = [\pi_i] \text{ where } \pi_i \equiv P(q_1 = S_i)$$

$\lambda = (\mathbf{A}, \mathbf{B}, \mathbf{\Pi})$, parameter set of HMM

Hidden Markov models

Introduction

Discrete Markov process

Observable Markov model

Hidden Markov model (HMM)

Solving HMMs

Three basic problems of HMMs

- Given a number of sequences of observations, we are interested in:
  1. Evaluation

     Given λ, and O, calculate $P(O \mid \lambda)$

Three basic problems of HMMs

- Given a number of sequences of observations, we are interested in:

1. Evaluation

   Given λ, and O, calculate $P(O \mid \lambda)$

2. State sequence

   Given λ, and O, find Q^* such that

$$P(Q^* \mid O, \lambda) = \max_Q P(Q \mid O, \lambda)$$

Three basic problems of HMMs

- Given a number of sequences of observations, we are interested in:
  1. Evaluation

     Given λ, and O, calculate $P(O \mid \lambda)$
  2. State sequence

     Given λ, and O, find Q^* such that

     $$P(Q^* \mid O, \lambda) = \max_Q P(Q \mid O, \lambda)$$
  3. Learning

     Given $\mathrm{X} = \{O^k\}_k$, find λ^* such that

     $$P(\mathrm{X} \mid \lambda^*) = \max_\lambda P(\mathrm{X} \mid \lambda)$$

1. Evaluation

- Given an observation sequence $O = \{O_1\, O_2 \cdots O_T\}$ and a state sequence $Q = \{q_1 q_2 \cdots q_T\}$, the probability of observing O given the state sequence Q is

$$P(O|Q,\lambda) = \prod_{t=1}^{T} P(O_t|q_t,\lambda) = b_{q_1}(O_1) \cdot b_{q_2}(O_2) \cdots b_{q_T}(O_T)$$

1. Evaluation

- Given an observation sequence $O = \{O_1 \, O_2 \cdots O_T\}$ and a state sequence $Q = \{q_1 q_2 \cdots q_T\}$, the probability of observing O given the state sequence Q is

$$P(O|Q,\lambda) = \prod_{t=1}^{T} P(O_t|q_t,\lambda) = b_{q_1}(O_1) \cdot b_{q_2}(O_2) \cdots b_{q_T}(O_T)$$

- which we cannot calculate because we do not know the state sequence.

1. Evaluation

$$P(O|Q,\lambda) = \prod_{t=1}^{T} P(O_t|q_t,\lambda) = b_{q_1}(O_1)\cdot b_{q_2}(O_2)\cdots b_{q_T}(O_T)$$

- Which we cannot calculate because we do not know the state sequence.
- But, we can compute the probability of a state sequence $P(Q|\lambda) = P(q_1)\prod_{t=2}^{T} P(q_t|q_{t-1}) = \pi_{q_1} a_{q_1 q_2} \cdots a_{q_{T-1} q_T}$

1. Evaluation

$$P(O|Q,\lambda) = \prod_{t=1}^{T} P(O_t|q_t,\lambda) = b_{q_1}(O_1)\cdot b_{q_2}(O_2)\cdots b_{q_T}(O_T)$$

- Which we cannot calculate because we do not know the state sequence.
- But, we can compute

$$P(Q|\lambda) = P(q_1)\prod_{t=2}^{T} P(q_t|q_{t-1}) = \pi_{q_1} a_{q_1 q_2}\cdots a_{q_{T-1} q_T}$$

$$\begin{aligned} P(O,Q|\lambda) &= P(q_1)\prod_{t=2}^{T} P(q_t|q_{t-1})\prod_{t=1}^{T} P(O_t|q_t) \\ &= \pi_{q_1} b_{q_1}(O_1) a_{q_1 q_2} b_{q_2}(O_2)\cdots a_{q_{T-1} q_T} b_{q_T}(O_T) \end{aligned}$$

1. Evaluation

$$P(O|Q,\lambda) = \prod_{t=1}^{T} P(O_t|q_t,\lambda) = b_{q_1}(O_1) \cdot b_{q_2}(O_2) \cdots b_{q_T}(O_T)$$

- Which we cannot calculate because we do not know the state sequence.
- But, we can compute

$$P(Q|\lambda) = P(q_1)\prod_{t=2}^{T} P(q_t|q_{t-1}) = \pi_{q_1} a_{q_1 q_2} \cdots a_{q_{T-1} q_T}$$

$$\begin{aligned} P(O,Q|\lambda) &= P(q_1)\prod_{t=2}^{T} P(q_t|q_{t-1}) \prod_{t=1}^{T} P(O_t|q_t) \\ &= \pi_{q_1} b_{q_1}(O_1) a_{q_1 q_2} b_{q_2}(O_2) \cdots a_{q_{T-1} q_T} b_{q_T}(O_T) \end{aligned}$$

$$P(O|\lambda) = \sum_{\text{all possible } Q} P(O,Q|\lambda)$$

1. Evaluation

$$P(O|Q,\lambda) = \prod_{t=1}^{T} P(O_t|q_t,\lambda) = b_{q_1}(O_1) \cdot b_{q_2}(O_2) \cdots b_{q_T}(O_T)$$

- Which we cannot calculate because we do not know the state sequence.
- But, we can compute

$$P(O|\lambda) = \sum_{\text{all possible } Q} P(O,Q|\lambda)$$

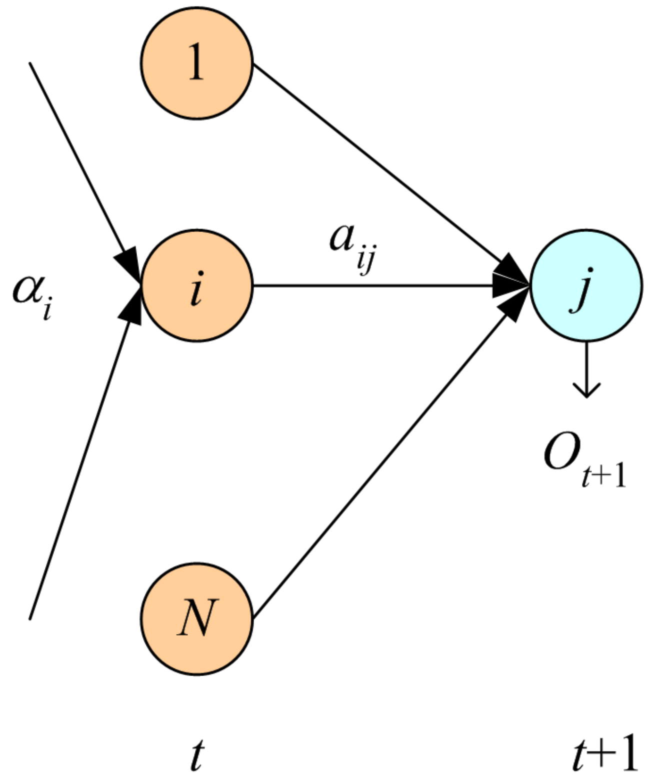
- There are N^T possible Q

1. Evaluation

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

- Forward-Backward procedure
  - Forward variable

$$\alpha_t(i) \equiv P(O_1 \cdots O_t, q_t = S_i \mid \lambda)$$



1. Evaluation

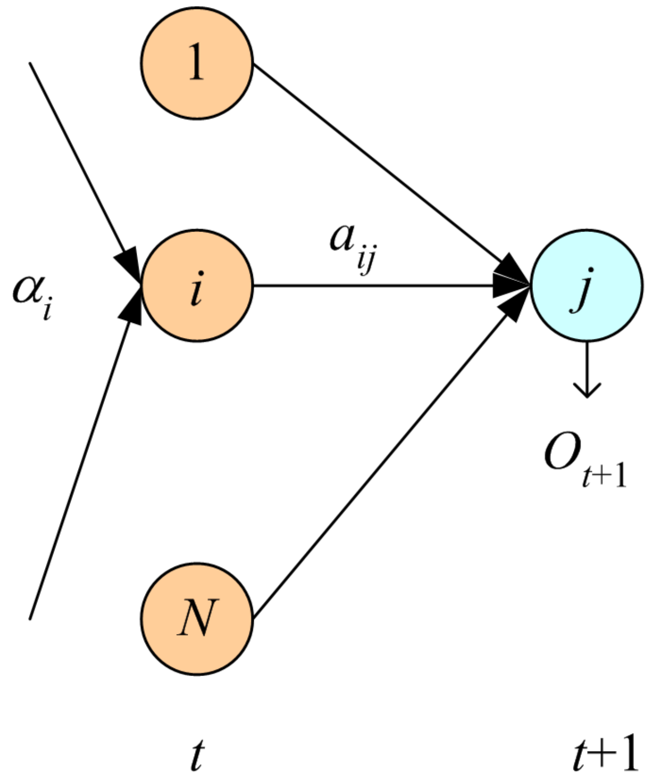
$$P(O|\lambda) = \sum_{i=1}^{N} P(O, q_T = S_i|\lambda)$$

- Forward-Backward procedure
  - Forward variable

$$\alpha_t(i) \equiv P(O_1 \cdots O_t, q_t = S_i \mid \lambda)$$

Initialization:

$$\alpha_1(i) = \pi_i b_i(O_1)$$



1. Evaluation

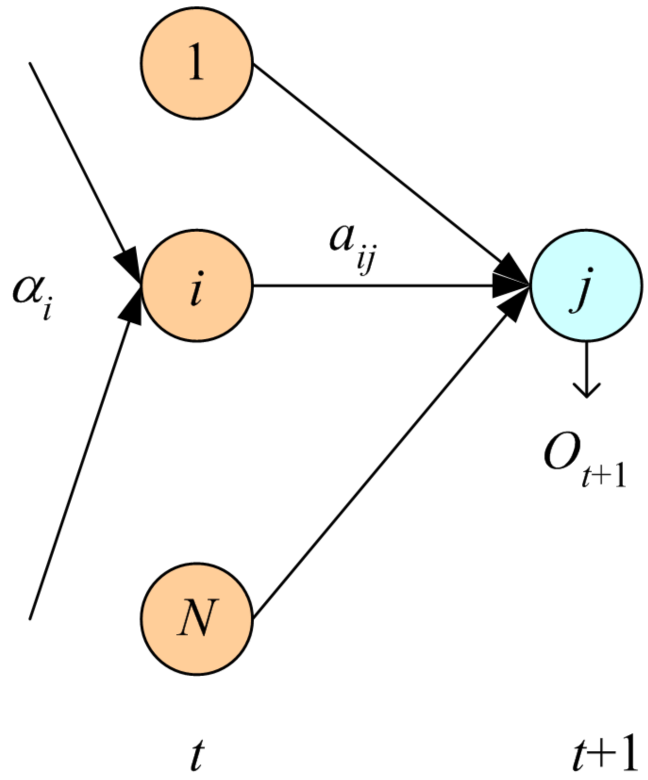
$$P(O|\lambda) \quad = \quad \sum_{i=1}^{N} P(O, q_T = S_i|\lambda)$$

- Forward-Backward procedure
  - Forward variable

$$\alpha_t(i) \equiv P(O_1 \cdots O_t, q_t = S_i \mid \lambda)$$

Initialization:

$$\alpha_1(i) = \pi_i b_i(O_1)$$



$$\begin{aligned}
\alpha_1(i) &\equiv P(O_1, q_1 = S_i|\lambda) \\
&= P(O_1|q_1 = S_i, \lambda)P(q_1 = S_i|\lambda) \\
&= \pi_i b_i(O_1)
\end{aligned}$$

1. Evaluation

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

- Forward-Backward procedure
  - Forward variable

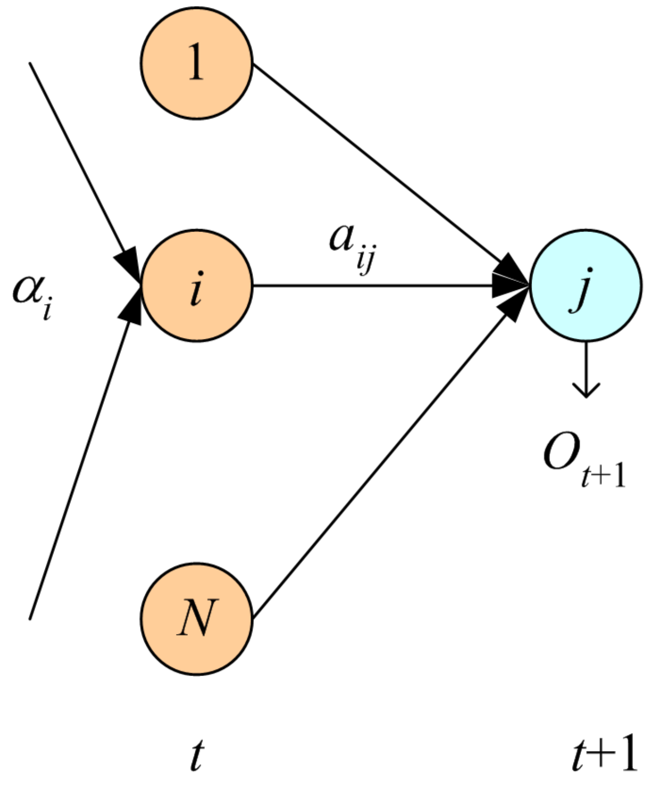
$$\alpha_t(i) \equiv P(O_1 \cdots O_t, q_t = S_i \mid \lambda)$$

Initialization:

$$\alpha_1(i) = \pi_i b_i(O_1)$$

Recursion:

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

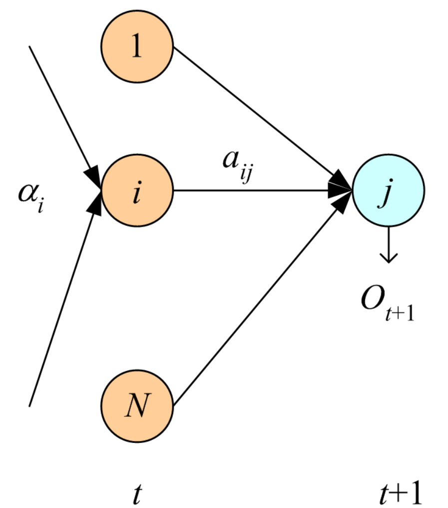


1. Evaluation

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

Recursion

$$\alpha_{t+1}(j) \equiv P(O_1 \cdots O_{t+1}, q_{t+1} = S_j|\lambda)$$



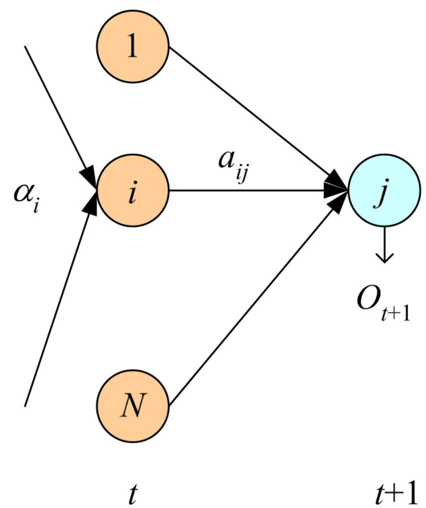
1. Evaluation

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

Recursion

$$\alpha_{t+1}(j) \equiv P(O_1 \cdots O_{t+1}, q_{t+1} = S_j|\lambda)$$

$$= P(O_1 \cdots O_{t+1}|q_{t+1} = S_j, \lambda)P(q_{t+1} = S_j|\lambda)$$



1. Evaluation

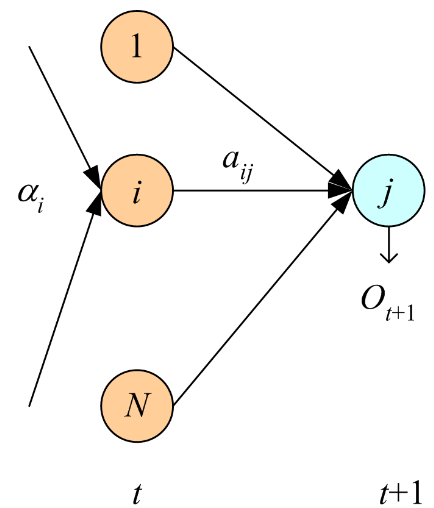
$$P(O|\lambda) = \sum_{i=1}^{N} P(O, q_T = S_i|\lambda)$$

Recursion

$$\alpha_{t+1}(j) \equiv P(O_1 \cdots O_{t+1}, q_{t+1} = S_j|\lambda)$$

$$= P(O_1 \cdots O_{t+1}|q_{t+1} = S_j, \lambda)P(q_{t+1} = S_j|\lambda)$$

$$= P(O_1 \cdots O_t|q_{t+1} = S_j, \lambda)P(O_{t+1}|q_{t+1} = S_j, \lambda)P(q_{t+1} = S_j|\lambda)$$



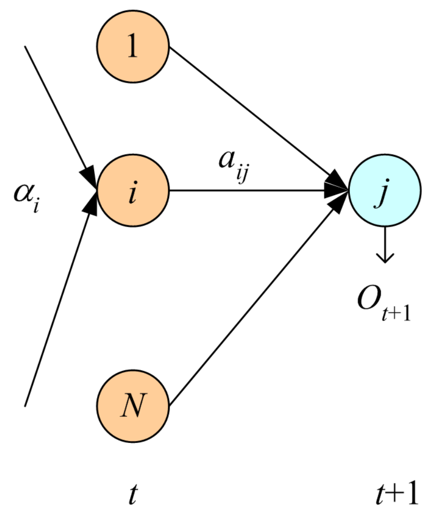
1. Evaluation

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

Recursion

$$\alpha_{t+1}(j) \equiv P(O_1 \cdots O_{t+1}, q_{t+1} = S_j|\lambda)$$

$$\begin{aligned}
&= P(O_1 \cdots O_{t+1}|q_{t+1} = S_j, \lambda)P(q_{t+1} = S_j|\lambda) \\
&= P(O_1 \cdots O_t|q_{t+1} = S_j, \lambda)P(O_{t+1}|q_{t+1} = S_j, \lambda)P(q_{t+1} = S_j|\lambda) \\
&= P(O_1 \cdots O_t, q_{t+1} = S_j|\lambda)P(O_{t+1}|q_{t+1} = S_j, \lambda)
\end{aligned}$$



1. Evaluation

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

Recursion

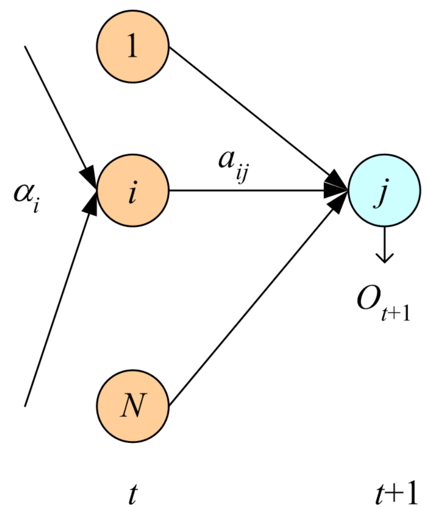
$$\alpha_{t+1}(j) \equiv P(O_1 \cdots O_{t+1}, q_{t+1} = S_j|\lambda)$$

$$= P(O_1 \cdots O_{t+1}|q_{t+1} = S_j, \lambda)P(q_{t+1} = S_j|\lambda)$$

$$= P(O_1 \cdots O_t|q_{t+1} = S_j, \lambda)P(O_{t+1}|q_{t+1} = S_j, \lambda)P(q_{t+1} = S_j|\lambda)$$

$$= P(O_1 \cdots O_t, q_{t+1} = S_j|\lambda)P(O_{t+1}|q_{t+1} = S_j, \lambda)$$

$$= P(O_{t+1}|q_{t+1} = S_j, \lambda) \sum_i P(O_1 \cdots O_t, q_t = S_i, q_{t+1} = S_j|\lambda)$$



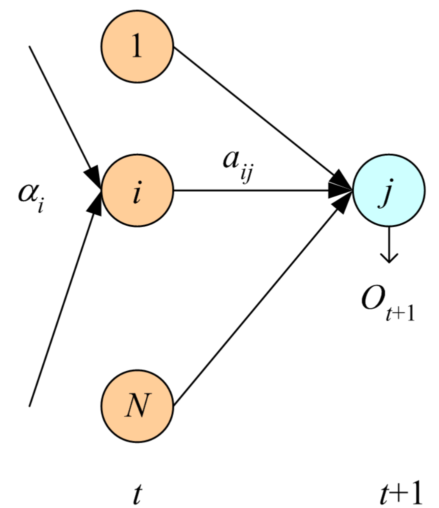
1. Evaluation

$$P(O|\lambda) \quad = \quad \sum_{i=1}^{N} P(O, q_T = S_i|\lambda)$$

Recursion

$$\alpha_{t+1}(j) \equiv P(O_1 \cdots O_{t+1}, q_{t+1} = S_j|\lambda)$$

$$\begin{aligned}
&= \quad P(O_1 \cdots O_{t+1}|q_{t+1} = S_j, \lambda)P(q_{t+1} = S_j|\lambda) \\
&= \quad P(O_1 \cdots O_t|q_{t+1} = S_j, \lambda)P(O_{t+1}|q_{t+1} = S_j, \lambda)P(q_{t+1} = S_j|\lambda) \\
&= \quad P(O_1 \cdots O_t, q_{t+1} = S_j|\lambda)P(O_{t+1}|q_{t+1} = S_j, \lambda) \\
&= \quad P(O_{t+1}|q_{t+1} = S_j, \lambda) \sum_i P(O_1 \cdots O_t, q_t = S_i, q_{t+1} = S_j|\lambda) \\
&= \quad P(O_{t+1}|q_{t+1} = S_j, \lambda) \\
&\qquad \sum_i P(O_1 \cdots O_t, q_{t+1} = S_j|q_t = S_i, \lambda)P(q_t = S_i|\lambda)
\end{aligned}$$



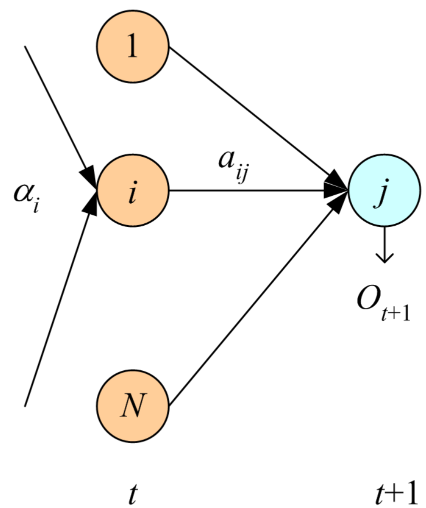
1. Evaluation

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

Recursion

$$\alpha_{t+1}(j) \equiv P(O_1 \cdots O_{t+1}, q_{t+1} = S_j|\lambda)$$

$$\begin{aligned}
&= P(O_1 \cdots O_{t+1}|q_{t+1} = S_j, \lambda)P(q_{t+1} = S_j|\lambda) \\
&= P(O_1 \cdots O_t|q_{t+1} = S_j, \lambda)P(O_{t+1}|q_{t+1} = S_j, \lambda)P(q_{t+1} = S_j|\lambda) \\
&= P(O_1 \cdots O_t, q_{t+1} = S_j|\lambda)P(O_{t+1}|q_{t+1} = S_j, \lambda) \\
&= P(O_{t+1}|q_{t+1} = S_j, \lambda) \sum_i P(O_1 \cdots O_t, q_t = S_i, q_{t+1} = S_j|\lambda) \\
&= P(O_{t+1}|q_{t+1} = S_j, \lambda) \\
&\quad \sum_i P(O_1 \cdots O_t, q_{t+1} = S_j|q_t = S_i, \lambda)P(q_t = S_i|\lambda) \\
&= P(O_{t+1}|q_{t+1} = S_j, \lambda) \\
&\quad \sum_i P(O_1 \cdots O_t|q_t = S_i, \lambda)P(q_{t+1} = S_j|q_t = S_i, \lambda)P(q_t = S_i|\lambda)
\end{aligned}$$



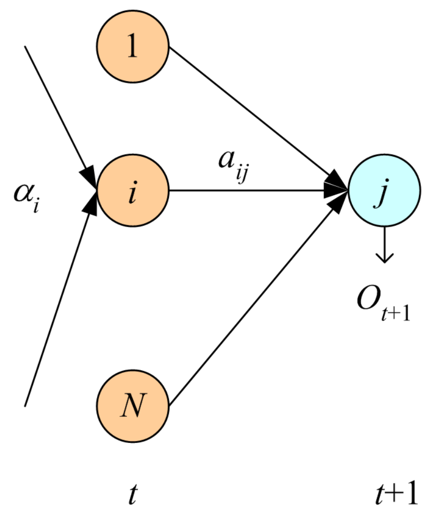
1. Evaluation

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

Recursion

$$\alpha_{t+1}(j) \equiv P(O_1 \cdots O_{t+1}, q_{t+1} = S_j|\lambda)$$

$$\begin{aligned} &= P(O_{t+1}|q_{t+1} = S_j, \lambda) \\ &\quad \sum_i P(O_1 \cdots O_t, q_{t+1} = S_j|q_t = S_i, \lambda)P(q_t = S_i|\lambda) \\ &= P(O_{t+1}|q_{t+1} = S_j, \lambda) \\ &\quad \sum_i P(O_1 \cdots O_t|q_t = S_i, \lambda)P(q_{t+1} = S_j|q_t = S_i, \lambda)P(q_t = S_i|\lambda) \end{aligned}$$



1. Evaluation

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

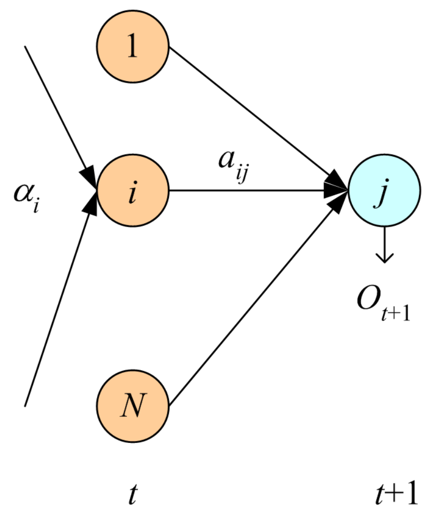
Recursion

$$\alpha_{t+1}(j) \equiv P(O_1 \cdots O_{t+1}, q_{t+1} = S_j|\lambda)$$

$$= P(O_{t+1}|q_{t+1} = S_j, \lambda) \sum_i P(O_1 \cdots O_t, q_{t+1} = S_j|q_t = S_i, \lambda) P(q_t = S_i|\lambda)$$

$$= P(O_{t+1}|q_{t+1} = S_j, \lambda) \sum_i P(O_1 \cdots O_t|q_t = S_i, \lambda) P(q_{t+1} = S_j|q_t = S_i, \lambda) P(q_t = S_i|\lambda)$$

$$= P(O_{t+1}|q_{t+1} = S_j, \lambda) \sum_i P(O_1 \cdots O_t, q_t = S_i|\lambda) P(q_{t+1} = S_j|q_t = S_i, \lambda)$$



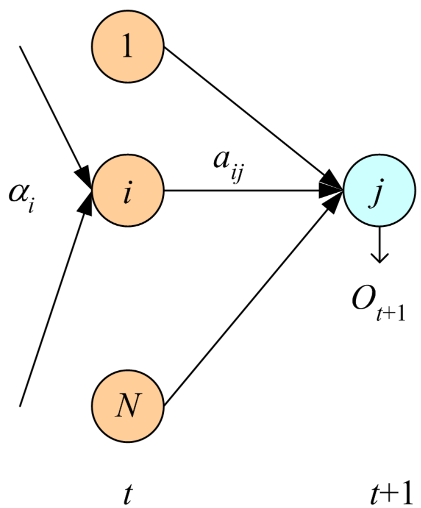
1. Evaluation

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

Recursion

$$\alpha_{t+1}(j) \equiv P(O_1 \cdots O_{t+1}, q_{t+1} = S_j|\lambda)$$

$$\begin{aligned}
&= P(O_{t+1}|q_{t+1} = S_j, \lambda) \\
&\quad \sum_i P(O_1 \cdots O_t, q_{t+1} = S_j|q_t = S_i, \lambda) P(q_t = S_i|\lambda) \\
&= P(O_{t+1}|q_{t+1} = S_j, \lambda) \\
&\quad \sum_i P(O_1 \cdots O_t|q_t = S_i, \lambda) P(q_{t+1} = S_j|q_t = S_i, \lambda) P(q_t = S_i|\lambda) \\
&= P(O_{t+1}|q_{t+1} = S_j, \lambda) \\
&\quad \sum_i P(O_1 \cdots O_t, q_t = S_i|\lambda) P(q_{t+1} = S_j|q_t = S_i, \lambda) \\
&= \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij} \right] b_j(O_{t+1})
\end{aligned}$$



1. Evaluation

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

- Forward-Backward procedure
  - Forward variable

$$\alpha_t(i) \equiv P(O_1 \cdots O_t, q_t = S_i \mid \lambda)$$

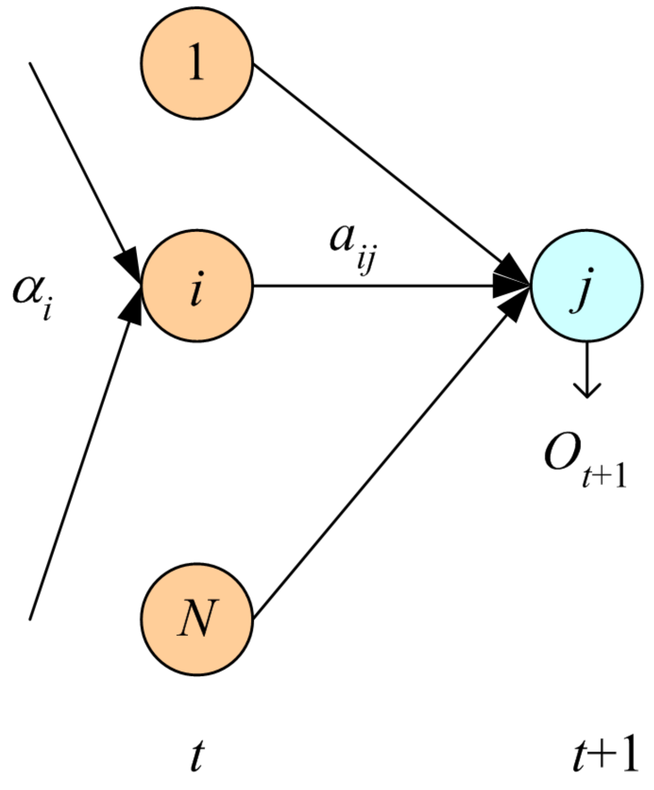
Initialization:

$$\alpha_1(i) = \pi_i b_i(O_1)$$

Recursion:

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

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

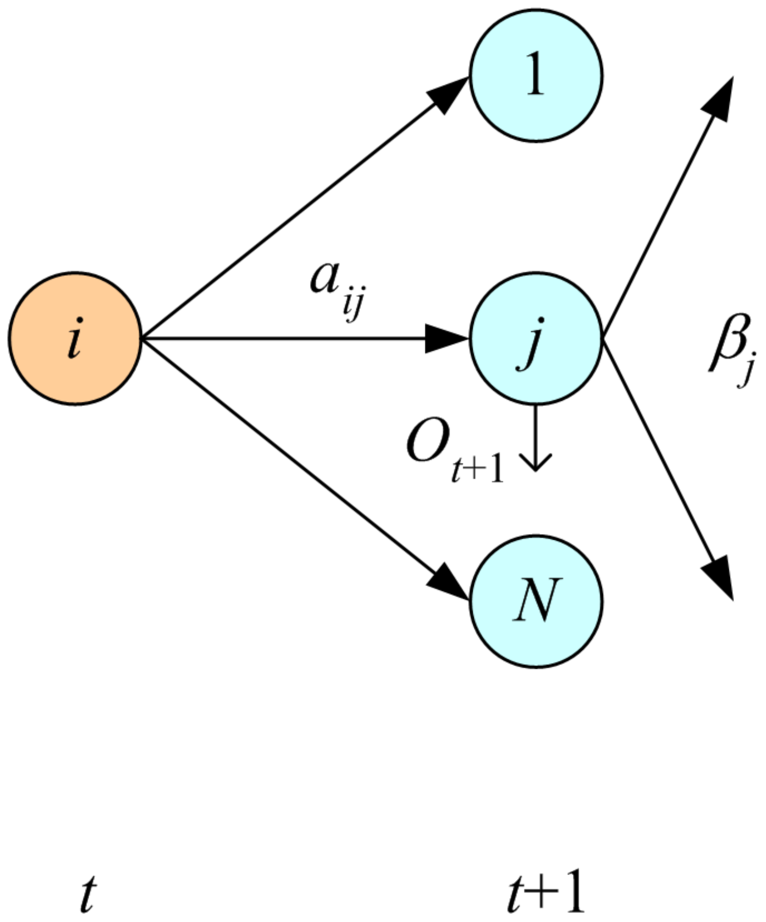


1. Evaluation

$$P(O|Q,\lambda) = \prod_{t=1}^{T} P(O_t|q_t,\lambda) = b_{q_1}(O_1)\cdot b_{q_2}(O_2)\cdots b_{q_T}(O_T)$$

- Forward-Backward procedure
  - Backward variable

$$\beta_t(i) \equiv P\left(O_{t+1}\cdots O_T \,|\, q_t = S_i, \lambda\right)$$



1. Evaluation

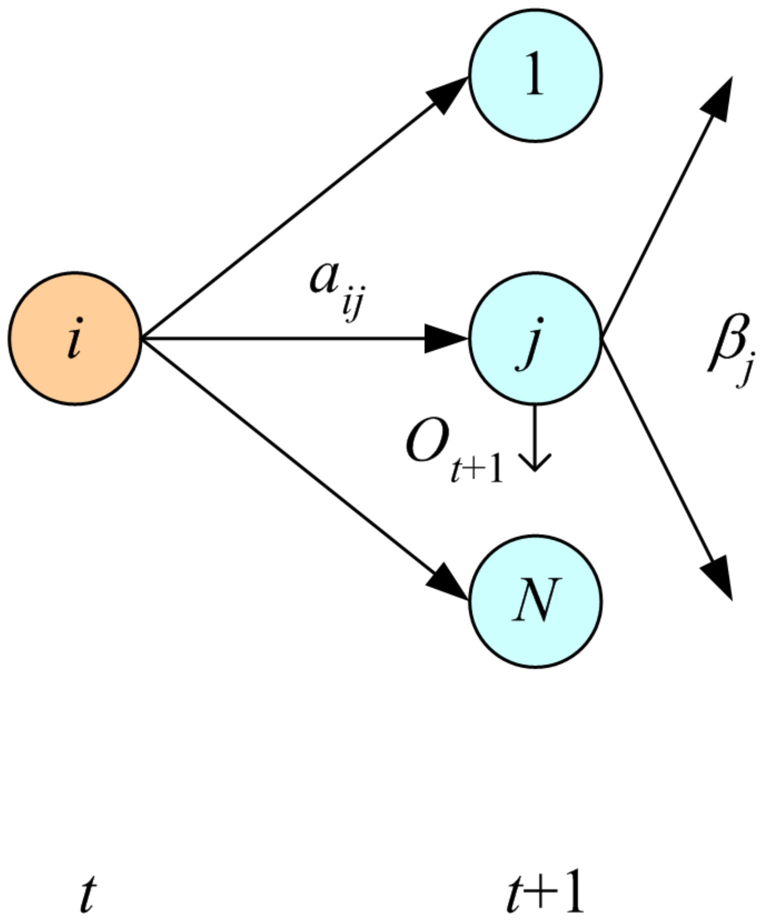
$$P(O|Q,\lambda) = \prod_{t=1}^{T} P(O_t|q_t,\lambda) = b_{q_1}(O_1) \cdot b_{q_2}(O_2) \cdots b_{q_T}(O_T)$$

- Forward-Backward procedure
  - Backward variable

$$\beta_t(i) \equiv P(O_{t+1} \cdots O_T \mid q_t = S_i, \lambda)$$

Initialization:

$$\beta_T(i) = 1$$



1. Evaluation

$$P(O|Q,\lambda) = \prod_{t=1}^{T} P(O_t|q_t,\lambda) = b_{q_1}(O_1) \cdot b_{q_2}(O_2) \cdots b_{q_T}(O_T)$$

- Forward-Backward procedure
  - Backward variable

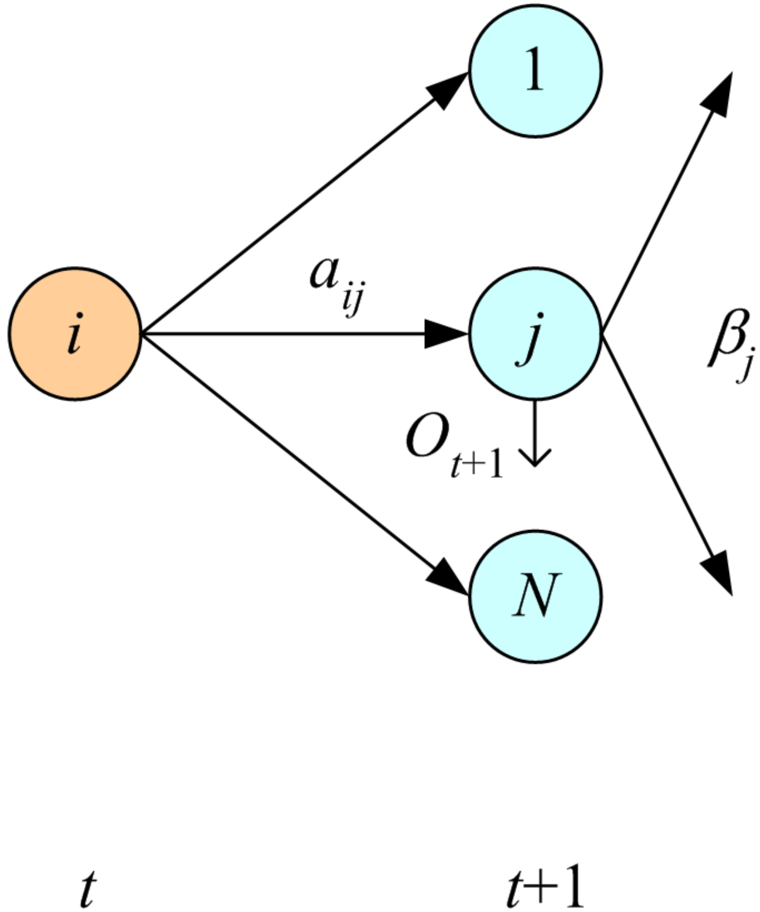
$$\beta_t(i) \equiv P(O_{t+1} \cdots O_T \mid q_t = S_i, \lambda)$$

Initialization:

$$\beta_T(i) = 1$$

Recursion:

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



1. Evaluation

$$P(O|Q,\lambda) = \prod_{t=1}^{T} P(O_t|q_t,\lambda) = b_{q_1}(O_1)\cdot b_{q_2}(O_2)\cdots b_{q_T}(O_T)$$

- Forward-Backward procedure
  - Forward variable

$$\alpha_t(i) \equiv P(O_1 \cdots O_t, q_t = S_i \mid \lambda)$$

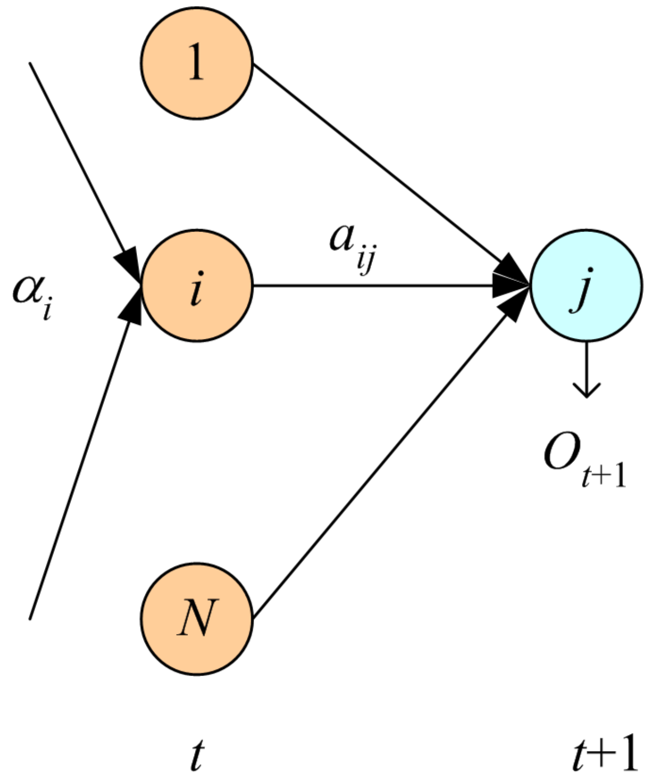
Initialization:

$$\alpha_1(i) = \pi_i b_i(O_1)$$

Recursion:

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

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



2. State sequence

- Find the state sequence $Q = \{q_1 q_2 \cdots q_T\}$ having the highest probability of generating the observation sequence $O = \{O_1\ O_2 \cdots O_T\}$, given the model λ

$$P(Q^* \mid O, \lambda) = \max_Q P(Q \mid O, \lambda)$$

2. State sequence

$$P(Q^* \mid O, \lambda) = \max_Q P(Q \mid O, \lambda)$$

- Let us define $\gamma_t(i)$ as the probability of being in state S_i at time t, given O and λ

$$\begin{aligned} \gamma_t(i) &\equiv P(q_t = S_i | O, \lambda) \\ &= \frac{P(O | q_t = S_i, \lambda) P(q_t = S_i | \lambda)}{P(O | \lambda)} \end{aligned}$$

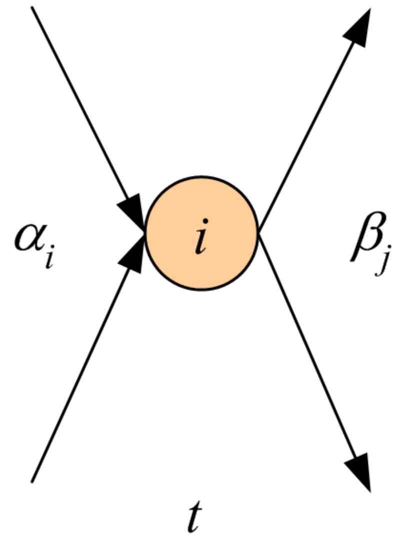
2. State sequence

$$P(Q^* \mid O, \lambda) = \max_Q P(Q \mid O, \lambda)$$

- Let us define $\gamma_t(i)$ as the probability of being in state S_i at time t, given O and λ

$$\gamma_t(i) \equiv P(q_t = S_i \mid O, \lambda)$$

$$= \frac{\alpha_t(i)\beta_t(i)}{\sum_{j=1}^{N} \alpha_t(j)\beta_t(j)}$$

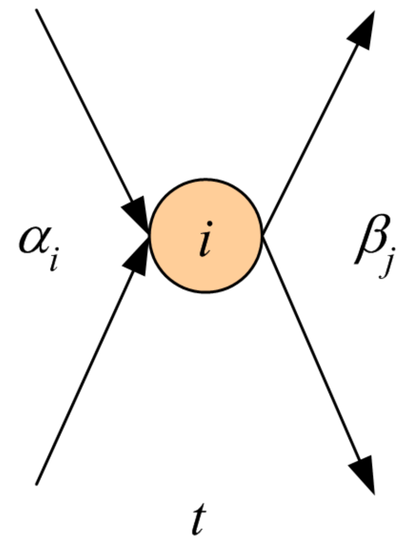


2. State sequence

$$P(Q^* \mid O, \lambda) = \max_Q P(Q \mid O, \lambda)$$

- Let us define $\gamma_t(i)$ as the probability of being in state S_i at time t, given O and λ

$$\gamma_t(i) \equiv P(q_t = S_i \mid O, \lambda) = \frac{\alpha_t(i)\beta_t(i)}{\sum_{j=1}^{N} \alpha_t(j)\beta_t(j)}$$



- Choose the state that has the highest probability
- for each time step: $q_t^* = \arg\max_i \gamma_t(i)$

2. State sequence

$$\gamma_t(i) \equiv P(q_t = S_i | O, \lambda)$$
$$= \frac{\alpha_t(i)\beta_t(i)}{\sum_{j=1}^{N} \alpha_t(j)\beta_t(j)}$$

- Choose the state that has the highest probability
- for each time step: $q_t^* = \arg\max_i \gamma_t(i)$. **NO**
- **Viterbi Algorithm**

2. State sequence

$$\gamma_t(i) \equiv P(q_t = S_i | O, \lambda)$$
$$= \frac{\alpha_t(i)\beta_t(i)}{\sum_{j=1}^{N} \alpha_t(j)\beta_t(j)}$$

- Viterbi Algorithm
  - Given state sequence $Q = q_1 q_2 \cdots q_T$ and observation sequence $O = O_1 \cdots O_T$, we define $\delta_t(i)$ as the probability of the highest probability path at time t that accounts for the first t observations and ends in S_i

$$\delta_t(i) \equiv \max_{q_1 q_2 \cdots q_{t-1}} p(q_1 q_2 \cdots q_{t-1}, q_t = S_i, O_1 \cdots O_t | \lambda)$$

2. State sequence

$$\delta_t(i) \equiv \max_{q_1 q_2 \cdots q_{t-1}} p(q_1 q_2 \cdots q_{t-1}, q_t = S_i, O_1 \cdots O_t | \lambda)$$

- Viterbi Algorithm
  - Initialization:

$$\delta_1(i) = \pi_i b_i(O_1)$$

$$\psi_1(i) = 0$$

2. State sequence

$$\delta_t(i) \equiv \max_{q_1 q_2 \cdots q_{t-1}} p(q_1 q_2 \cdots q_{t-1}, q_t = S_i, O_1 \cdots O_t | \lambda)$$

- Viterbi Algorithm
  - Initialization:

$$\delta_1(i) = \pi_i b_i(O_1)$$

$$\psi_1(i) = 0$$

  - Recursion

$$\delta_t(j) = \max_i \delta_{t\text{-}1}(i) a_{ij} b_j(O_t)$$

$$\psi_t(j) = \mathrm{argmax}_i\, \delta_{t\text{-}1}(i) a_{ij}$$

2. State sequence

$$\delta_t(i) \equiv \max_{q_1 q_2 \cdots q_{t-1}} p(q_1 q_2 \cdots q_{t-1}, q_t = S_i, O_1 \cdots O_t | \lambda)$$

- Viterbi Algorithm
  - Initialization:

$$\delta_1(i) = \pi_i b_i(O_1)$$

$$\psi_1(i) = 0$$

  - Recursion

$$\delta_t(j) = \max_i \delta_{t\text{-}1}(i) a_{ij} b_j(O_t)$$

$$\psi_t(j) = \operatorname{argmax}_i \delta_{t\text{-}1}(i) a_{ij}$$

  - Termination:

$$p^* = \max_i \delta_T(i)$$

$$q_T^* = \operatorname{argmax}_i \delta_T(i)$$

2. State sequence

$$\delta_t(i) \equiv \max_{q_1 q_2 \cdots q_{t-1}} p(q_1 q_2 \cdots q_{t-1}, q_t = S_i, O_1 \cdots O_t | \lambda)$$

- Viterbi Algorithm
  - Initialization:

$$\delta_1(i) = \pi_i b_i(O_1)$$

$$\psi_1(i) = 0$$

  - Recursion

$$\delta_t(j) = \max_i \delta_{t\text{-}1}(i) a_{ij} b_j(O_t)$$

$$\psi_t(j) = \text{argmax}_i \delta_{t\text{-}1}(i) a_{ij}$$

  - Path bactracking

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

  - Termination:

$$p^* = \max_i \delta_T(i)$$

$$q_T^* = \text{argmax}_i \delta_T(i)$$

2. State sequence

$$\delta_t(i) \equiv \max_{q_1 q_2 \cdots q_{t-1}} p(q_1 q_2 \cdots q_{t-1}, q_t = S_i, O_1 \cdots O_t | \lambda)$$

- Viterbi Algorithm
  - Recursion

    $$\delta_t(j) = \max_i \delta_{t-1}(i) a_{ij} b_j(O_t)$$

    $$\psi_t(j) = \text{argmax}_i \delta_{t-1}(i) a_{ij}$$

  - Termination:

    $$p^* = \max_i \delta_T(i)$$

    $$q_T^* = \text{argmax}_i \delta_T(i)$$

  - Path bactracking

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

  - $\psi_t(j)$ keeps track of the state that maximizes $\delta_t(j)$ at time $t-1$, that is, the best previous state

3. Learning

- Maximum Likelihood
  - Calculate $\lambda*$ that maximizes the likelihood of the sample of training sequences, $\mathrm{X} = \{O^k\}^K_{k=1}$, namely, $P(\mathrm{X}|\lambda)$

3. Learning

- Maximum Likelihood
  - Calculate $\lambda*$ that maximizes the likelihood of the sample of training sequences, $\mathrm{X} = \{O^k\}^K{}_{k=1}$, namely, $P(\mathrm{X}|\lambda)$

$$\xi_t(i,j) \equiv P(q_t = S_i, q_{t+1} = S_j \mid O, \lambda)$$

3. Learning

- Maximum Likelihood

$$\xi_t(i,j) \equiv P(q_t = S_i, q_{t+1} = S_j \mid O, \lambda)$$

$$\begin{aligned}\xi_t(i,j) &\equiv P(q_t = S_i, q_{t+1} = S_j | O, \lambda) \\ &= \frac{P(O|q_t = S_i, q_{t+1} = S_j, \lambda) P(q_t = S_i, q_{t+1} = S_j | \lambda)}{P(O|\lambda)}\end{aligned}$$

3. Learning

- Maximum Likelihood

$$\xi_t(i,j) \equiv P(q_t = S_i, q_{t+1} = S_j | O, \lambda)$$

$$\begin{aligned}
\xi_t(i,j) &\equiv P(q_t = S_i, q_{t+1} = S_j | O, \lambda) \\
&= \frac{P(O|q_t = S_i, q_{t+1} = S_j, \lambda) P(q_t = S_i, q_{t+1} = S_j | \lambda)}{P(O|\lambda)} \\
&= \frac{P(O|q_t = S_i, q_{t+1} = S_j, \lambda) P(q_{t+1} = S_j | q_t = S_i, \lambda) P(q_t = S_i | \lambda)}{P(O|\lambda)} \\
&= \left(\frac{1}{P(O|\lambda)}\right) P(O_1 \cdots O_t | q_t = S_i, \lambda) P(O_{t+1} | q_{t+1} = S_j, \lambda) \\
&\quad P(O_{t+2} \cdots O_T | q_{t+1} = S_j, \lambda) a_{ij} P(q_t = S_i | \lambda)
\end{aligned}$$

3. Learning

- Maximum Likelihood

$$\xi_t(i,j) \equiv P(q_t = S_i, q_{t+1} = S_j \mid O, \lambda)$$

$$\begin{aligned}
\xi_t(i,j) &\equiv P(q_t = S_i, q_{t+1} = S_j|O,\lambda) \\
&= \frac{P(O|q_t = S_i, q_{t+1} = S_j, \lambda)P(q_t = S_i, q_{t+1} = S_j|\lambda)}{P(O|\lambda)} \\
&= \frac{P(O|q_t = S_i, q_{t+1} = S_j, \lambda)P(q_{t+1} = S_j|q_t = S_i, \lambda)P(q_t = S_i|\lambda)}{P(O|\lambda)} \\
&= \left(\frac{1}{P(O|\lambda)}\right) P(O_1 \cdots O_t|q_t = S_i, \lambda)P(O_{t+1}|q_{t+1} = S_j, \lambda) \\
&\quad P(O_{t+2} \cdots O_T|q_{t+1} = S_j, \lambda)a_{ij}P(q_t = S_i|\lambda) \\
&= \left(\frac{1}{P(O|\lambda)}\right) P(O_1 \cdots O_t, q_t = S_i|\lambda)P(O_{t+1}|q_{t+1} = S_j, \lambda) \\
&\quad P(O_{t+2} \cdots O_T|q_{t+1} = S_j, \lambda)a_{ij} \\
&= \frac{\alpha_t(i)b_j(O_{t+1})\beta_{t+1}(j)a_{ij}}{\sum_k \sum_l P(q_t = S_k, q_{t+1} = S_l, O|\lambda)}
\end{aligned}$$

3. Learning

- Maximum Likelihood

$$\xi_t(i,j) \equiv P(q_t = S_i, q_{t+1} = S_j \mid O, \lambda)$$

$$\begin{aligned}
\xi_t(i,j) &\equiv P(q_t = S_i,\ q_{t+1} = S_j \mid O, \lambda) \\
&= \frac{P(O \mid q_t = S_i,\ q_{t+1} = S_j, \lambda) P(q_t = S_i,\ q_{t+1} = S_j \mid \lambda)}{P(O \mid \lambda)} \\
&= \left(\frac{1}{P(O|\lambda)}\right) P(O_1 \cdots O_t, q_t = S_i|\lambda) P(O_{t+1}|q_{t+1} = S_j, \lambda) \\
&\quad P(O_{t+2} \cdots O_T|q_{t+1} = S_j, \lambda) a_{ij} \\
&= \frac{\alpha_t(i) b_j(O_{t+1}) \beta_{t+1}(j) a_{ij}}{\sum_k \sum_l P(q_t = S_k, q_{t+1} = S_l, O|\lambda)} \\
&= \frac{\alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)}{\sum_k \sum_l \alpha_t(k) a_{kl} b_l(O_{t+1}) \beta_{t+1}(l)}
\end{aligned}$$

3. Learning

- Maximum Likelihood

$$\xi_t(i,j) \equiv P(q_t = S_i, q_{t+1} = S_j \mid O, \lambda)$$

$$\xi_t(i,j) = \frac{\alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)}{\sum_k \sum_l \alpha_t(k) a_{kl} b_l(O_{t+1}) \beta_{t+1}(l)}$$

- We can calculate the probability of being in state S_i at time t by marginalizing over the arc probabilities for all possible next states

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

3. Learning

- Maximum Likelihood
  - We can calculate the probability of being in state S_i at time t by marginalizing over the arc probabilities for all possible next states

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

  - If the Markov model were not hidden but observable, both $\gamma_t(i)$ and $\xi_t(i,j)$ would be 0/1.
  - In this case when they are not, we estimate them with posterior probabilities that give us *soft counts*

3. Learning

- Maximum Likelihood
  - We can calculate the probability of being in state S_i at time t by marginalizing over the arc probabilities for all possible next states

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

  - In this case when they are not, we estimate them with posterior probabilities that give us *soft counts*
  - We estimate posterior probabilities first (in the E-step) and calculate the parameters with these estimates (in the M-step).

3. Learning

- Baum-Welch algorithm
  - E-step computes $\xi_t(i,j)$ and $\gamma_t(i)$
  - M-step re-calculate λ given $\xi_t(i,j)$ and $\gamma_t(i)$
  - Until convergence
    - $P(O|\lambda)$ never decreases

3. Learning

- Baum-Welch algorithm
  - E-step computes $\xi_t(i,j)$ and $\gamma_t(i)$

$$E[z_i^t] = \gamma_t(i)$$

$$E[z_{ij}^t] = \xi_t(i,j)$$

$$z_i^t = \begin{cases} 1 & \text{if } q_t = S_i \\ 0 & \text{otherwise} \end{cases} \qquad z_{ij}^t = \begin{cases} 1 & \text{if } q_t = S_i \text{ and } q_{t+1} = S_j \\ 0 & \text{otherwise} \end{cases}$$

$$\xi_t(i,j) = \frac{\alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)}{\sum_k \sum_l \alpha_t(k) a_{kl} b_l(O_{t+1}) \beta_{t+1}(l)}$$

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

3. Learning

- Baum-Welch algorithm
  - E-step computes $\xi_t(i,j)$ and $\gamma_t(i)$

$$E[z_i^t] = \gamma_t(i)$$
$$E[z_{ij}^t] = \xi_t(i,j)$$

$$z_i^t = \begin{cases} 1 & \text{if } q_t = S_i \\ 0 & \text{otherwise} \end{cases} \qquad z_{ij}^t = \begin{cases} 1 & \text{if } q_t = S_i \text{ and } q_{t+1} = S_j \\ 0 & \text{otherwise} \end{cases}$$

$$\xi_t(i,j) = \frac{\alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)}{\sum_k \sum_l \alpha_t(k) a_{kl} b_l(O_{t+1}) \beta_{t+1}(l)} \qquad \gamma_t(i) = \sum_{j=1}^{N} \xi_t(i,j)$$

  - M-step re-calculate λ given $\xi_t(i,j)$ and $\gamma_t(i)$
    - Ratio of number of transitions from S_i to S_j

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)}$$

    - Probability of observing v_m in S_j $\quad \hat{b}_j(m) = \dfrac{\sum_{t=1}^{T} \gamma_t(j) 1(O_t = v_m)}{\sum_{t=1}^{T} \gamma_t(j)}$

3. Learning

- Baum-Welch algorithm
  - E-step computes $\xi_t(i,j)$ and $\gamma_t(i)$

$$E[z_i^t] = \gamma_t(i)$$
$$E[z_{ij}^t] = \xi_t(i,j)$$

$$z_i^t = \begin{cases} 1 & \text{if } q_t = S_i \\ 0 & \text{otherwise} \end{cases} \qquad z_{ij}^t = \begin{cases} 1 & \text{if } q_t = S_i \text{ and } q_{t+1} = S_j \\ 0 & \text{otherwise} \end{cases}$$

$$\xi_t(i,j) = \frac{\alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)}{\sum_k \sum_l \alpha_t(k) a_{kl} b_l(O_{t+1}) \beta_{t+1}(l)} \qquad \gamma_t(i) = \sum_{j=1}^{N} \xi_t(i,j)$$

  - M-step re-calculate λ given $\xi_t(i,j)$ and $\gamma_t(i)$
    - For multiple observation sequences

$$\hat{a}_{ij} = \frac{\sum_{k=1}^{K} \sum_{t=1}^{T_k-1} \xi_t^k(i,j)}{\sum_{k=1}^{K} \sum_{t=1}^{T_k-1} \gamma_t^k(i)} \qquad \hat{\pi}_i = \frac{\sum_{k=1}^{K} \gamma_1^k(i)}{K}$$

$$\hat{b}_j(m) = \frac{\sum_{k=1}^{K} \sum_{t=1}^{T_k} \gamma_t^k(j) 1(O_t^k = v_m)}{\sum_{k=1}^{K} \sum_{t=1}^{T_k} \gamma_t^k(j)}$$

Classification with HMMs

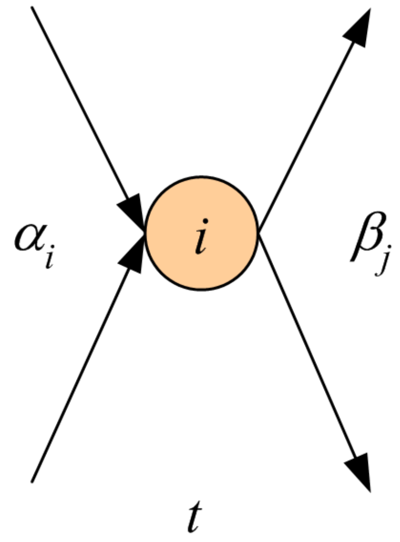
- We have a set of HMMs, each one modeling the sequences belonging to one class
  - For example, in spoken word recognition, examples of each word train a separate model, λ_i.
  - Given a new word utterance O to classify, all of the separate word models are evaluated to calculate $P(O|\lambda_i)$.
  - We then use Bayes' rule to get the posterior probabilities $$P(\lambda_i|O) = \frac{P(O|\lambda_i)P(\lambda_i)}{\sum_j P(O|\lambda_j)P(\lambda_j)}$$

Exercise 1

- Prove the recursion expression for the forward-backward algorithm:

Recursion:

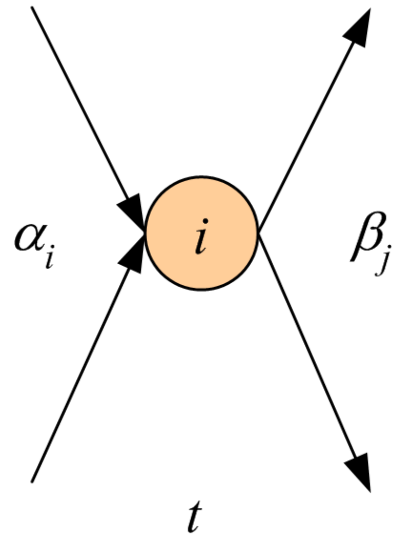
$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)$$



Exercise 2

- Prove that for finding the state sequence

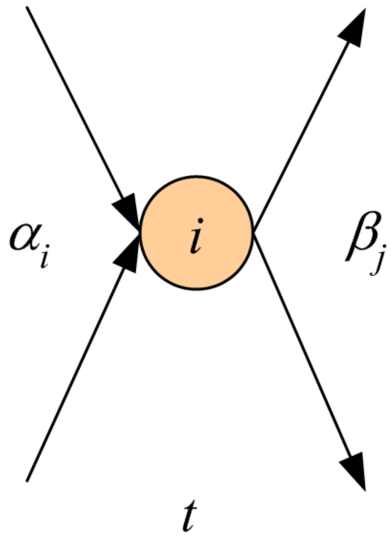
$$\gamma_t(i) \equiv P(q_t = S_i \mid O, \lambda) = \frac{\alpha_t(i)\beta_t(i)}{\sum_{j=1}^{N} \alpha_t(j)\beta_t(j)}$$



Exercise 2

- Prove that for finding the state sequence

$$\gamma_t(i) \equiv P(q_t = S_i | O, \lambda)$$
$$= \frac{\alpha_t(i)\beta_t(i)}{\sum_{j=1}^{N} \alpha_t(j)\beta_t(j)}$$



$$\gamma_t(i) \equiv P(q_t = S_i | O, \lambda)$$
$$= \frac{P(O | q_t = S_i, \lambda) P(q_t = S_i | \lambda)}{P(O | \lambda)}$$

Summary

Introduction

Discrete Markov process

Observable Markov model

Hidden Markov model (HMM)

Solving HMMs