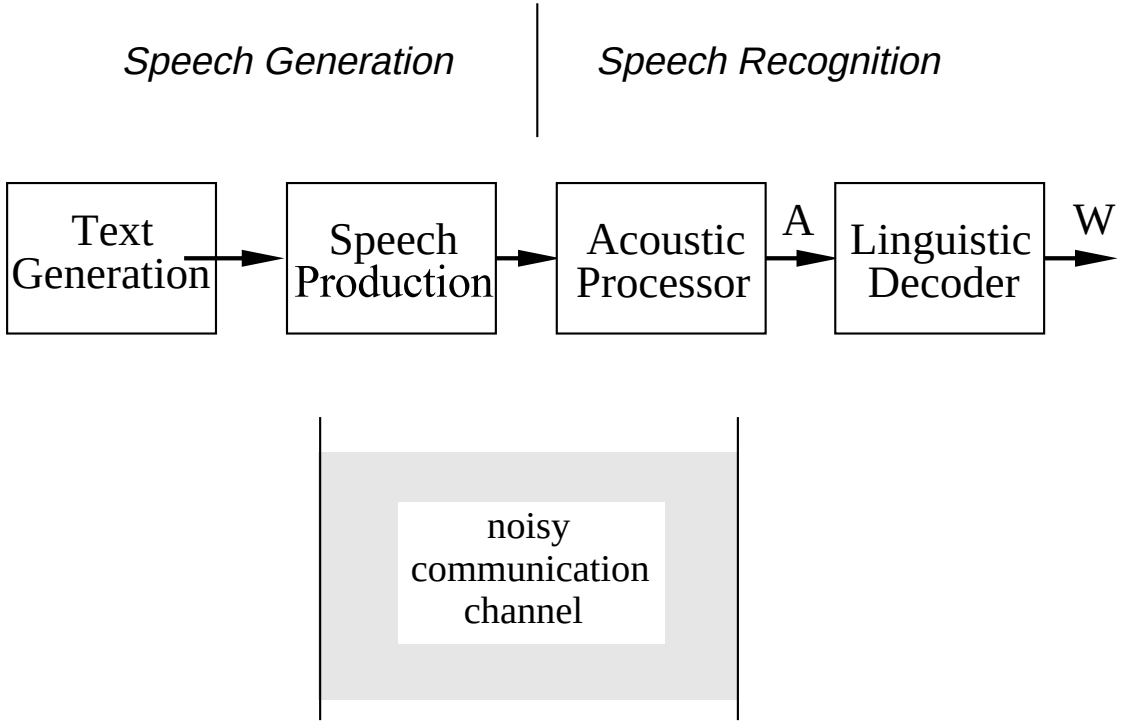
MIT

Hidden Markov Modelling

- Introduction
- Problem formulation
- Forward-Backward algorithm
- Viterbi search
- Baum-Welch parameter estimation
- Other considerations
  - Multiple observation sequences
  - Phone-based models for continuous speech recognition
  - Continuous density HMMs
  - Implementation issues

Information Theoretic Approach to ASR



Recognition is achieved by maximizing the probability of the linguistic string, $\boldsymbol{W}$, given the acoustic evidence, $\boldsymbol{A}$, i.e., choose the linguistic sequence $\hat{\boldsymbol{W}}$ such that

$$P(\hat{\boldsymbol{W}}|\boldsymbol{A}) = \max_{\boldsymbol{W}} P(\boldsymbol{W}|\boldsymbol{A})$$

Information Theoretic Approach to ASR

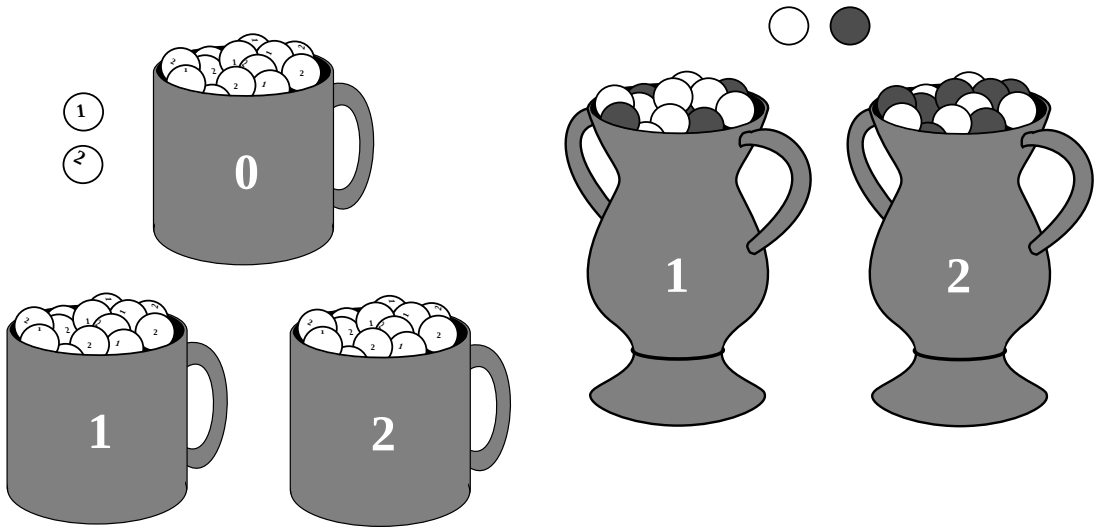
- From Bayes rule:

$$P(\boldsymbol{W}|\boldsymbol{A}) = \frac{P(\boldsymbol{A}|\boldsymbol{W})P(\boldsymbol{W})}{P(\boldsymbol{A})}$$

- Hidden Markov modelling (HMM) deals with the quantity $P(\boldsymbol{A}|\boldsymbol{W})$
- Change in notation:

$$\begin{aligned} \boldsymbol{A} &\rightarrow \boldsymbol{O} \\ \boldsymbol{W} &\rightarrow \lambda \\ P(\boldsymbol{A}|\boldsymbol{W}) &\rightarrow P(\boldsymbol{O}|\lambda) \end{aligned}$$

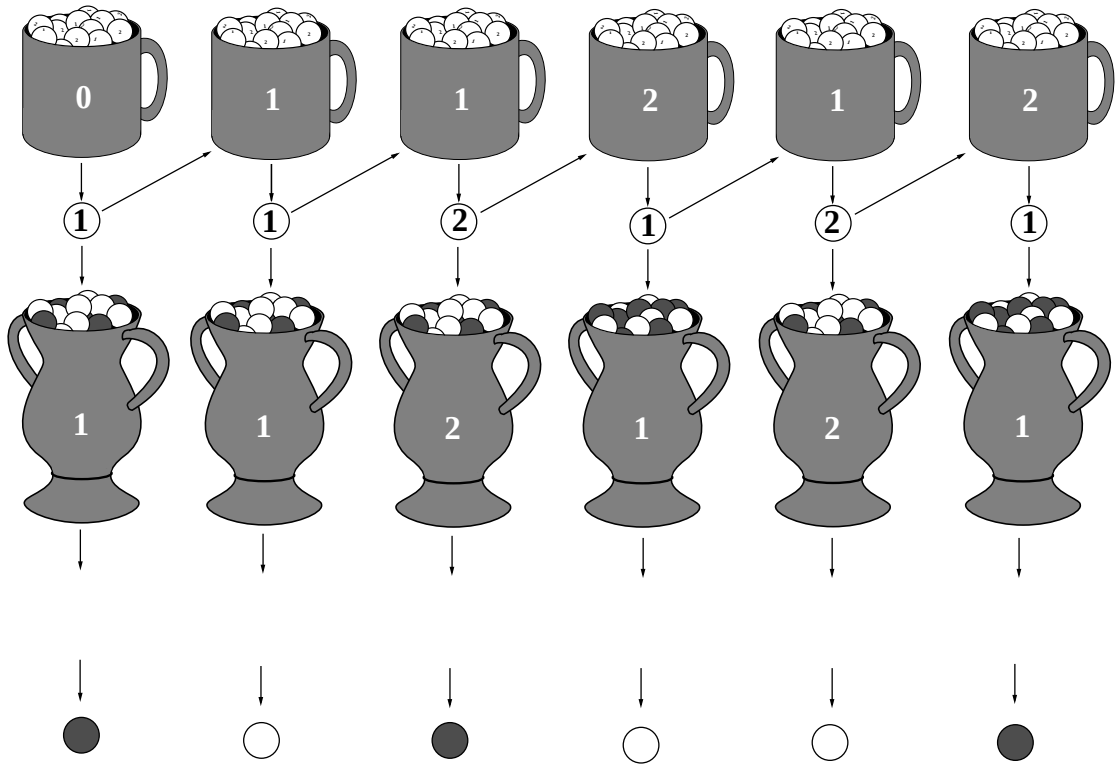
HMM: An Example



- Consider 3 mugs, each with mixtures of *state* stones, 1 and 2
- The fractions for the i^{th} mug are a_{i1} and a_{i2}, and $a_{i1} + a_{i2} = 1$
- Consider 2 urns, each with mixtures of black and white balls
- The fractions for the i^{th} urn are $b_i(B)$ and $b_i(W)$; $b_i(B) + b_i(W) = 1$
- The parameter vector for this model is:

$$\lambda = \{a_{01}, a_{02}, a_{11}, a_{12}, a_{21}, a_{22}, b_1(B), b_1(W), b_2(B), b_2(W)\}$$

HMM: An Example (cont'd)



Observation Sequence: $\boldsymbol{O} = \{B, W, B, W, W, B\}$

State Sequence: $\boldsymbol{Q} = \{1, 1, 2, 1, 2, 1\}$

Goal: Given the model λ and the observation sequence $\boldsymbol{O}$, how can the underlying state sequence $\boldsymbol{Q}$ be determined?

Elements of a Discrete Hidden Markov Model

- N: number of states in the model
  - states, $\boldsymbol{s} = \{s_1, s_2, \ldots, s_N\}$
  - state at time t, $q_t \in \boldsymbol{s}$
- M: number of observation symbols (i.e., discrete observations)
  - observation symbols, $\boldsymbol{v} = \{v_1, v_2, \ldots, v_M\}$
  - observation at time t, $o_t \in \boldsymbol{v}$
- $\boldsymbol{A} = \{a_{ij}\}$: state transition probability distribution
  - $a_{ij} = P(q_{t+1} = s_j | q_t = s_i), \ 1 \leq i, j \leq N$
- $\boldsymbol{B} = \{b_j(k)\}$: observation symbol probability distribution in state j
  - $b_j(k) = P(v_k \text{ at } t | q_t = s_j), \ 1 \leq j \leq N, 1 \leq k \leq M$
- $\pi = \{\pi_i\}$: initial state distribution
  - $\pi_i = P(q_1 = s_i), \ 1 \leq i \leq N$

Notationally, an HMM is typically written as: $\lambda = \{\boldsymbol{A}, \boldsymbol{B}, \pi\}$

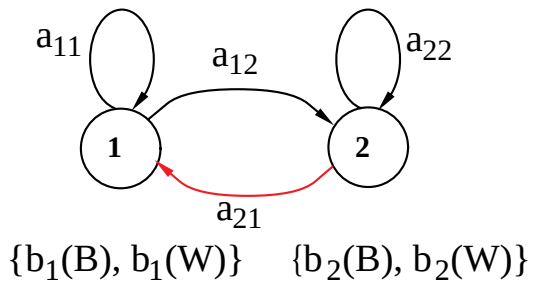
HMM: An Example (cont'd)

For our simple example:

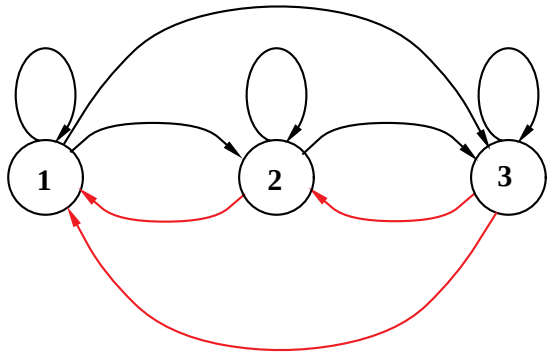
$$\pi = \{a_{01}, a_{02}\}, \quad \boldsymbol{A} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}, \text{ and } \quad \boldsymbol{B} = \begin{bmatrix} b_1(B) & b_1(W) \\ b_2(B) & b_2(W) \end{bmatrix}$$

State Diagram

2-state

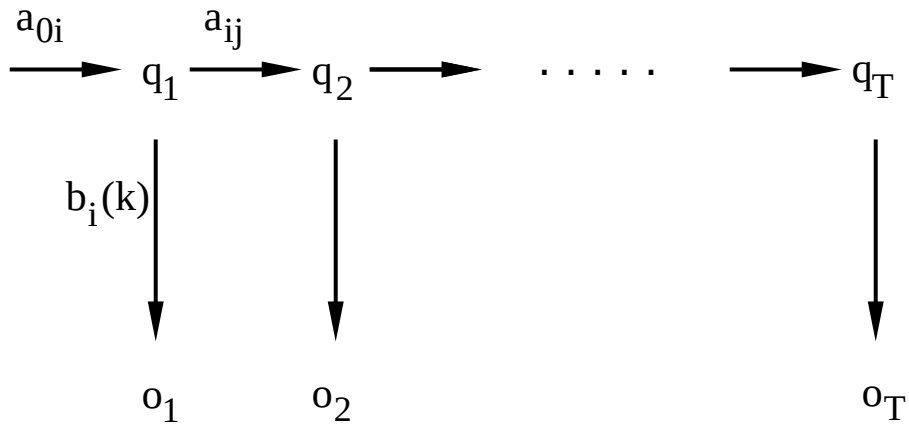


3-state

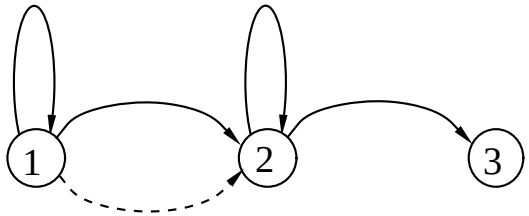


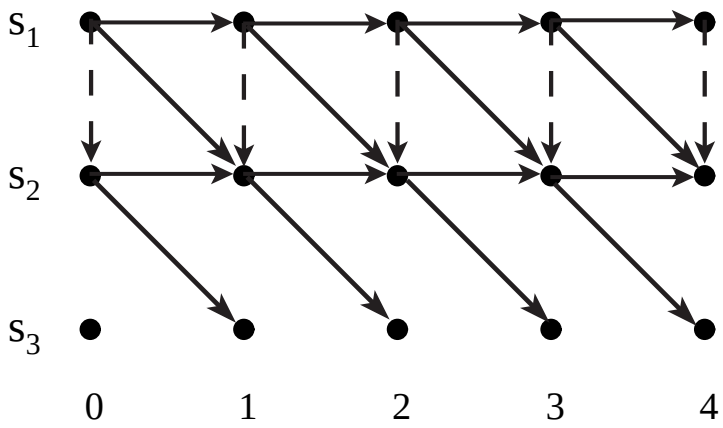
Generation of HMM Observations

1. Choose an initial state, $q_1 = s_i$, based on the initial state distribution, π
2. For $t = 1$ to T:
   - Choose $o_t = v_k$ according to the symbol probability distribution in state s_i, $b_i(k)$
   - Transition to a new state $q_{t+1} = s_j$ according to the state transition probability distribution for state s_i, a_{ij}
3. Increment t by 1, return to step 2 if $t \leq T$; else, terminate



Representing State Diagram by Trellis





The dashed line represents a *null* transition, where no observation symbol is generated

Three Basic HMM Problems

1. **Scoring:** Given an observation sequence $\boldsymbol{O} = \{o_1, o_2, ..., o_T\}$ and a model $\lambda = \{\boldsymbol{A}, \boldsymbol{B}, \pi\}$, how do we compute $P(\boldsymbol{O} \mid \lambda)$, the probability of the observation sequence?
   ==> The Forward-Backward Algorithm

2. **Matching:** Given an observation sequence $\boldsymbol{O} = \{o_1, o_2, ..., o_T\}$, how do we choose a state sequence $\boldsymbol{Q} = \{q_1, q_2, ..., q_T\}$ which is optimum in some sense?
   ==> The Viterbi Algorithm

3. **Training:** How do we adjust the model parameters $\lambda = \{A, B, \pi\}$ to maximize $P(\boldsymbol{O} \mid \lambda)$?
   ==> The Baum-Welch Re-estimation Procedures

Computation of $P(\boldsymbol{O}|\lambda)$

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

$$P(\boldsymbol{O}, \boldsymbol{Q}|\lambda) = P(\boldsymbol{O}|\boldsymbol{Q}, \lambda)P(\boldsymbol{Q}|\lambda)$$

- Consider the *fixed* state sequence: $\boldsymbol{Q} = q_1 q_2 \ldots q_T$

$$P(\boldsymbol{O}|\boldsymbol{Q}, \lambda) = b_{q_1}(o_1)b_{q_2}(o_2)\ldots b_{q_T}(o_T)$$

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

  Therefore:

$$P(\boldsymbol{O}|\lambda) = \sum_{q_1, q_2, \ldots, q_T} \pi_{q_1} b_{q_1}(o_1) a_{q_1 q_2} b_{q_2}(o_2) \ldots a_{q_{T-1} q_T} b_{q_T}(o_T)$$

- Calculation required $\approx 2T \cdot N^T$ (there are N^T such sequences)
  For $N = 5, T = 100 \Rightarrow 2 \cdot 100 \cdot 5^{100} \approx 10^{72}$ computations!

The Forward Algorithm

- Let us define the forward variable, $\alpha_t(i)$, as the probability of the partial observation sequence up to time t *and* state s_i at time t, given the model, i.e.

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

- It can easily be shown that:

$$\alpha_1(i) = \pi_i b_i(o_1), \qquad 1 \le i \le N$$

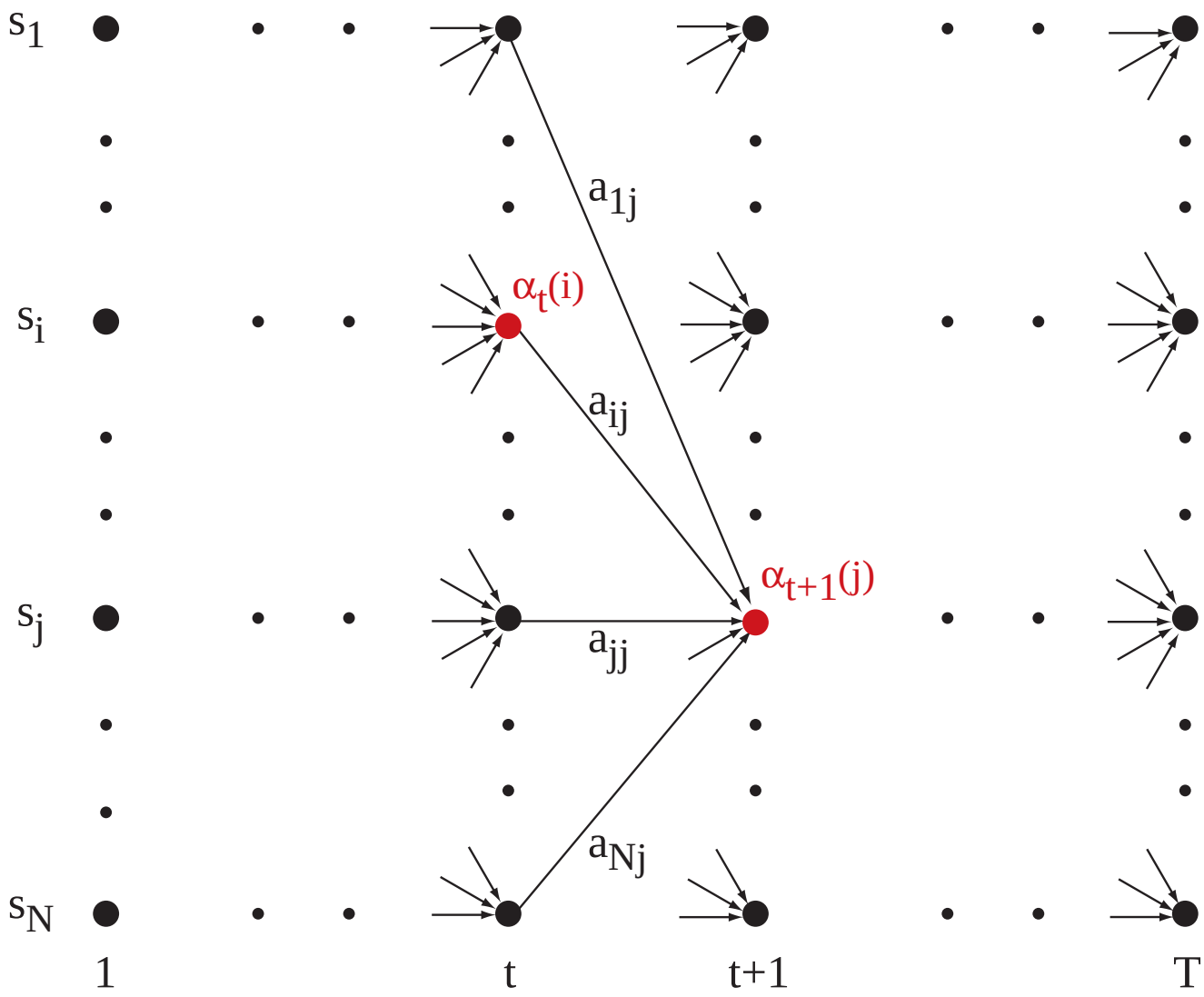
$$P(\boldsymbol{O}|\lambda) = \sum_{i=1}^{N} \alpha_T(i)$$

- By induction:

$$\alpha_{t+1}(j) = [\sum_{i=1}^{N} \alpha_t(i)\, a_{ij}] b_j(o_{t+1}), \qquad \begin{matrix} 1 \le t \le T-1 \\ 1 \le j \le N \end{matrix}$$

- Calculation is on the order of $N^2 T$.
  For $N = 5, T = 100 \Rightarrow 100 \cdot 5^2$ computations, instead of 10^{72}

Forward Algorithm Illustration



The Backward Algorithm

- Similarly, let us define the backward variable, $\beta_t(i)$, as the probability of the partial observation sequence from time $t+1$ to the end, given state s_i at time t and the model, i.e.

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

- It can easily be shown that:

$$\beta_T(i) = 1, \qquad 1 \le i \le N$$

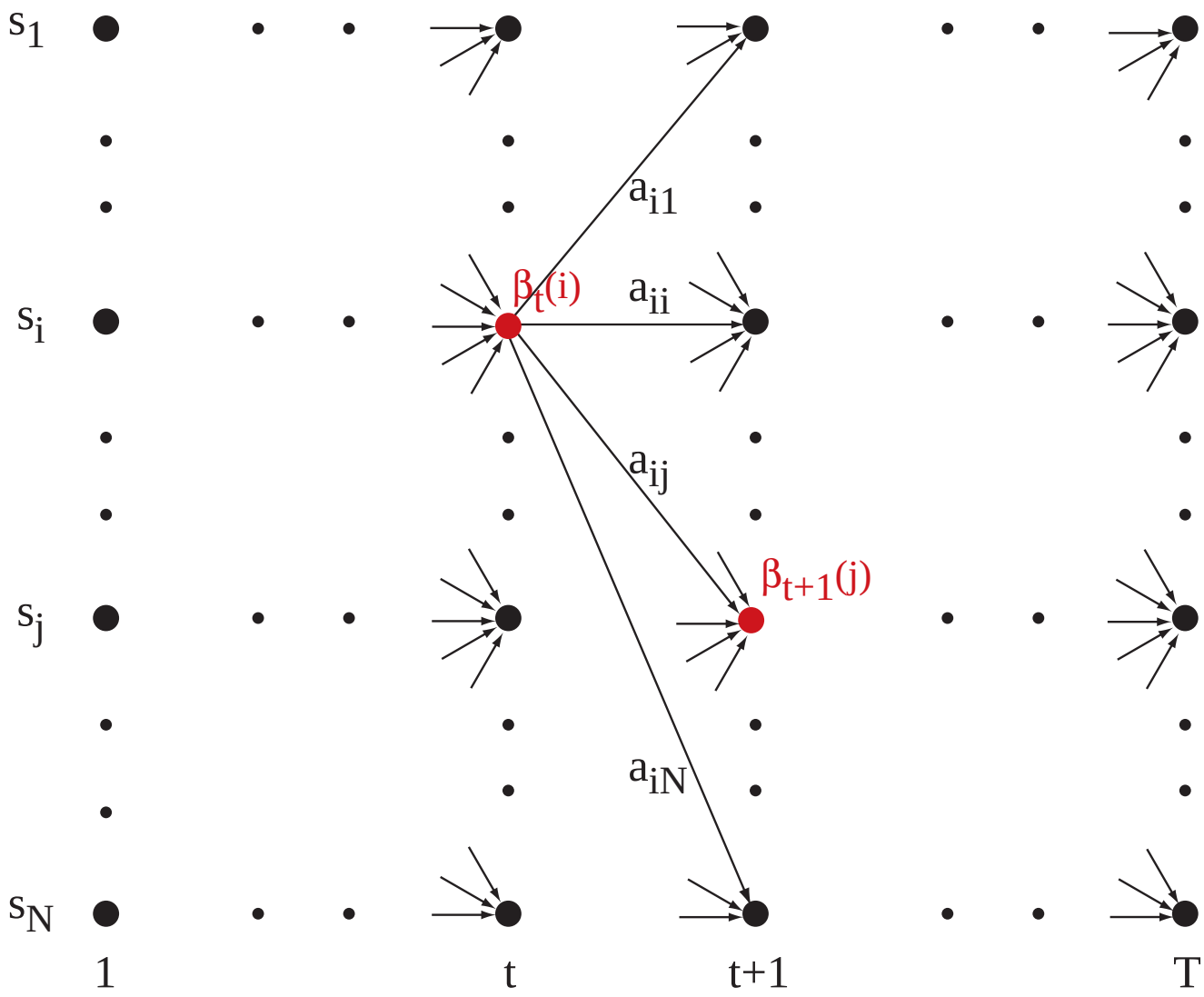
and:

$$P(\boldsymbol{O}|\lambda) = \sum_{i=1}^{N} \pi_i b_i(o_1) \beta_1(i)$$

- By induction:

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(o_{t+1}) \beta_{t+1}(j), \qquad \begin{array}{c} t = T-1, T-2, \ldots, 1 \\ 1 \le i \le N \end{array}$$

Backward Procedure Illustration



Finding Optimal State Sequences

- One criterion chooses states, q_t, which are *individually* most likely
  - This maximizes the expected number of correct states
- Let us define $\gamma_t(i)$ as the probability of being in state s_i at time t, given the observation sequence and the model, i.e.

$$\gamma_t(i) = P(q_t = s_i | \boldsymbol{O}, \lambda) \qquad \sum_{i=1}^{N} \gamma_t(i) = 1, \qquad \forall t$$

- Then the individually most likely state, q_t, at time t is:

$$q_t = \underset{1 \le i \le N}{\operatorname{argmax}} \ \gamma_t(i) \qquad 1 \le t \le T$$

- Note that it can be shown that:

$$\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{P(\boldsymbol{O}|\lambda)}$$

Finding Optimal State Sequences

- The individual optimality criterion has the problem that the optimum state sequence may not obey state transition constraints
- Another optimality criterion is to choose the state sequence which maximizes $P(\boldsymbol{Q}, \boldsymbol{O}|\lambda)$; This can be found by the *Viterbi* algorithm
- Let us define $\delta_t(i)$ as the highest probability along a single path, at time t, which accounts for the first t observations, i.e.

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

- By induction: $\delta_{t+1}(j) = [\max_i \delta_t(i) a_{ij}] b_j(o_{t+1})$
- To retrieve the state sequence, we must keep track of the state sequence which gave the best path, at time t, to state s_i
  - We do this in a separate array $\psi_t(i)$

The Viterbi Algorithm

1. Initialization:

$$\begin{aligned}\delta_1(i) &= \pi_i b_i(o_1), \qquad 1 \le i \le N \\ \psi_1(i) &= 0\end{aligned}$$

2. Recursion:

$$\begin{aligned}\delta_t(j) &= \max_{1\le i\le N}[\delta_{t-1}(i)a_{ij}]b_j(o_t), \qquad 2 \le t \le T \qquad 1 \le j \le N \\ \psi_t(j) &= \operatorname*{argmax}_{1\le i\le N}[\delta_{t-1}(i)a_{ij}], \qquad 2 \le t \le T \qquad 1 \le j \le N\end{aligned}$$

3. Termination:

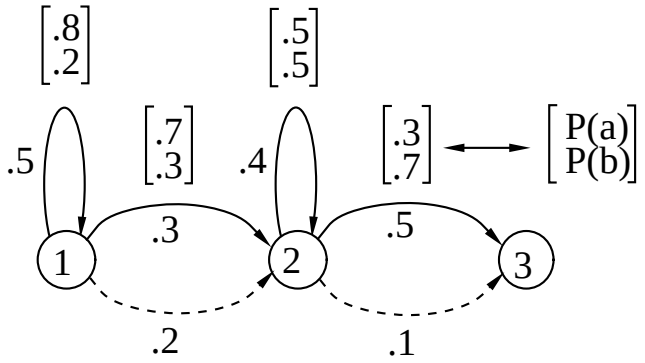
$$\begin{aligned}P^* &= \max_{1\le i\le N}[\delta_T(i)] \\ q_T^* &= \operatorname*{argmax}_{1\le i\le N}[\delta_T(i)]\end{aligned}$$

4. Path (state-sequence) backtracking:

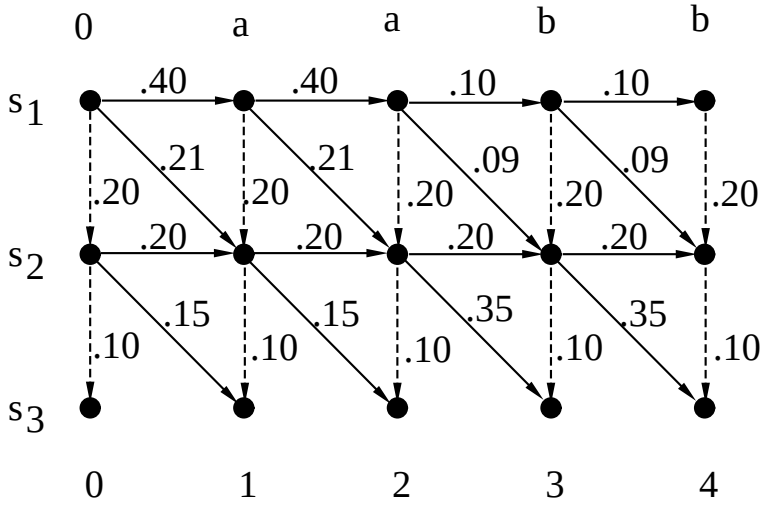
$$q_t^* = \psi_{t+1}(q_{t+1}^*), \qquad t = T-1, T-2, \ldots, 1$$

Computation $\approx N^2 T$

The Viterbi Algorithm: An Example

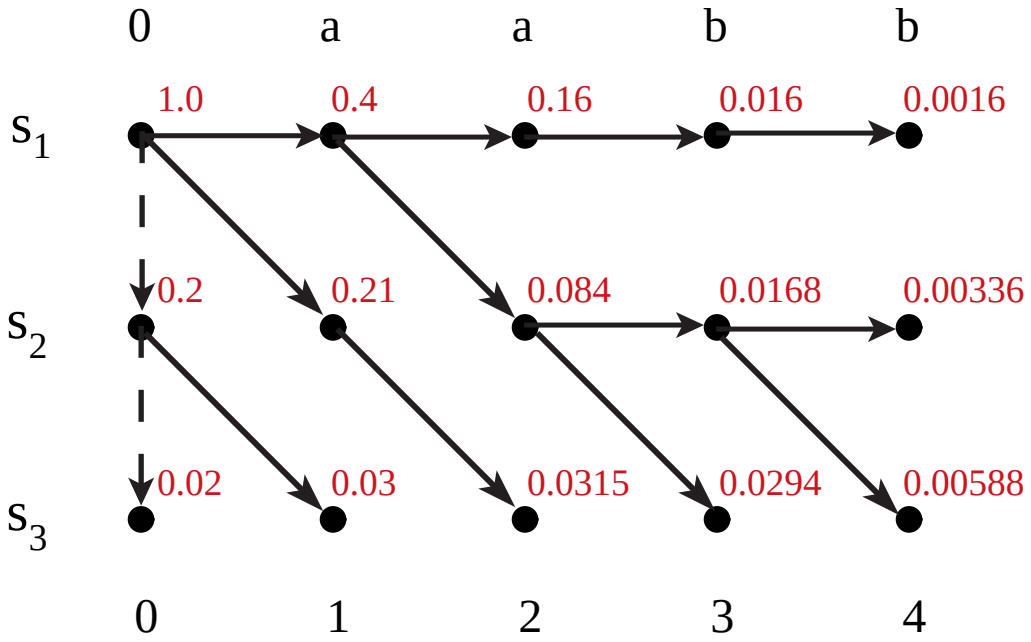


O={a a b b}



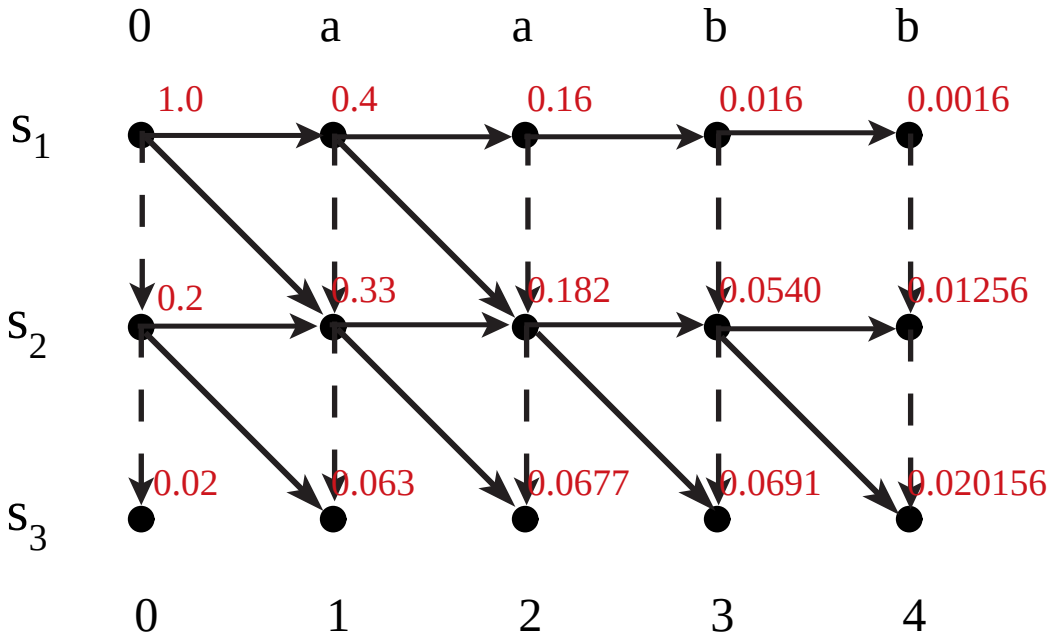
The Viterbi Algorithm: An Example (cont'd)

	0	*a*	*aa*	*aab*	*aabb*
s_1	1.0	s_1, a .4	s_1, a .16	s_1, b .016	s_1, b .0016
s_2	$s_1, 0$ .2	$s_1, 0$ .08 s_1, a .21 s_2, a .04	$s_1, 0$ .032 s_1, a .084 s_2, a .042	$s_1, 0$ .0032 s_1, b .0144 s_2, b .0168	$s_1, 0$ .00032 s_1, b .00144 s_2, b .00336
s_3	$s_2, 0$ .02	$s_2, 0$ .021 s_2, a .03	$s_2, 0$ .0084 s_2, a .0315	$s_2, 0$ .00168 s_2, b .0294	$s_2, 0$ .000336 s_2, b .00588



Matching Using Forward-Backward Algorithm

	0	*a*	*aa*	*aab*	*aabb*
s_1	1.0	s_1, a .4	s_1, a .16	s_1, b .016	s_1, b .0016
s_2	$s_1, 0$ .2	$s_1, 0$ .08 s_1, a .21 s_2, a .04	$s_1, 0$ .032 s_1, a .084 s_2, a .066	$s_1, 0$ .0032 s_1, b .0144 s_2, b .0364	$s_1, 0$ .00032 s_1, b .00144 s_2, b .0108
s_3	$s_2, 0$ .02	$s_2, 0$ .033 s_2, a .03	$s_2, 0$ .0182 s_2, a .0495	$s_2, 0$ .0054 s_2, b .0637	$s_2, 0$ .001256 s_2, b .0189



Baum-Welch Re-estimation

- Baum-Welch re-estimation uses EM to determine ML parameters
- Define $\xi_t(i,j)$ as the probability of being in state s_i at time t and state s_j at time $t+1$, given the model and observation sequence

$$\xi_t(i,j) = P(q_t = s_i, q_{t+1} = s_j | \boldsymbol{O}, \lambda)$$

- Then:

$$\xi_t(i,j) = \frac{\alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)}{P(\boldsymbol{O}|\lambda)}$$

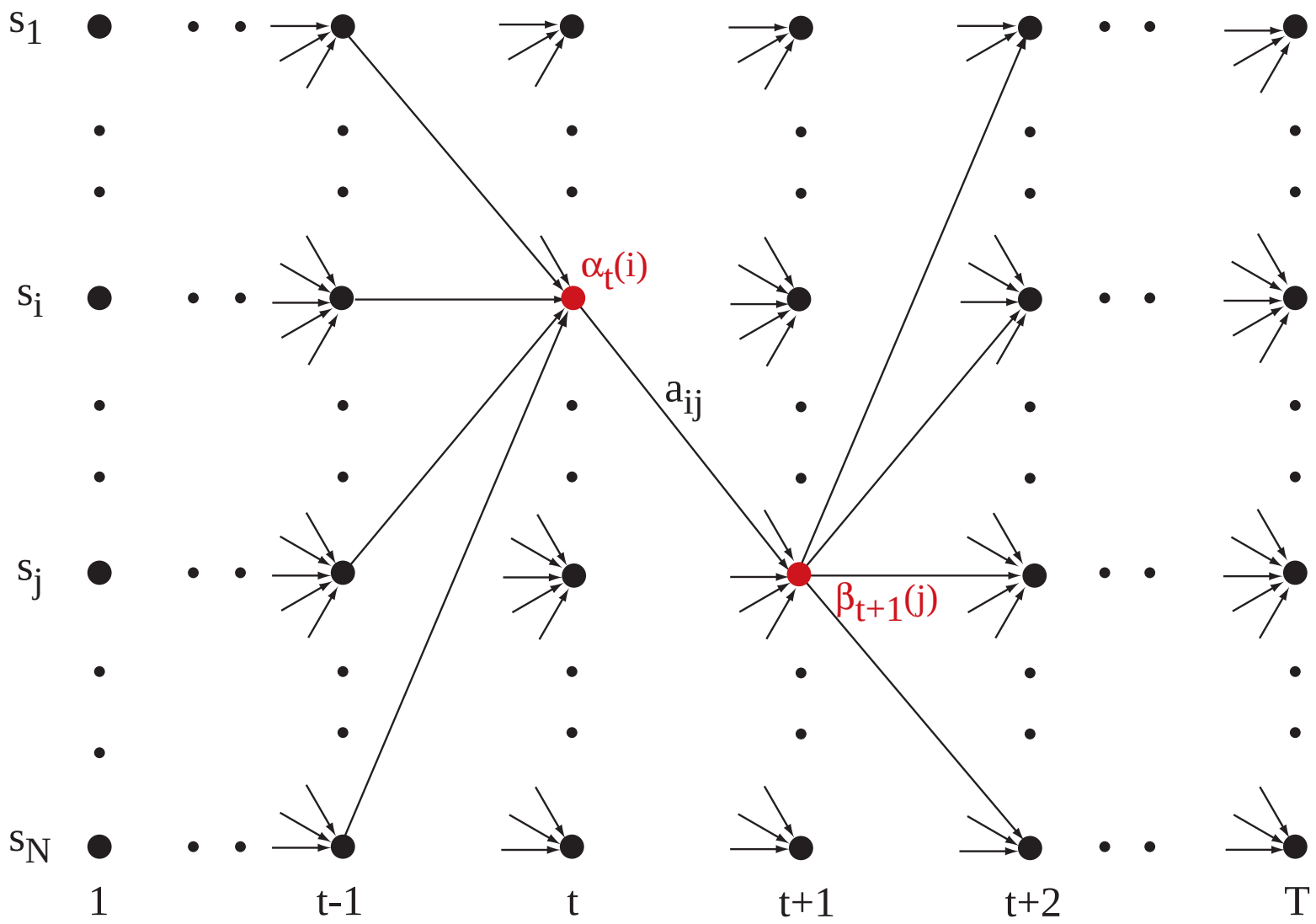
$$\gamma_t(i) = \sum_{j=1}^{N} \xi_t(i,j)$$

- Summing $\gamma_t(i)$ and $\xi_t(i,j)$, we get:

$$\sum_{t=1}^{T-1} \gamma_t(i) = \text{expected number of transitions from } s_i$$

$$\sum_{t=1}^{T-1} \xi_t(i,j) = \text{expected number of transitions from } s_i \text{ to } s_j$$

Baum-Welch Re-estimation Procedures



Baum-Welch Re-estimation Formulas

$$\begin{aligned}\bar{\pi} &= \text{expected number of times in state } s_i \text{ at } t = 1 \\ &= \gamma_1(i)\end{aligned}$$

$$\begin{aligned}\bar{a}_{ij} &= \frac{\text{expected number of transitions from state } s_i \text{ to } s_j}{\text{expected number of transitions from state } s_i} \\ &= \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)}\end{aligned}$$

$$\begin{aligned}\bar{b}_j(k) &= \frac{\text{expected number of times in state } s_j \text{ with symbol } v_k}{\text{expected number of times in state } s_j} \\ &= \frac{\sum_{t=1}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)}\end{aligned}$$

Baum-Welch Re-estimation Formulas

- If $\lambda = (\boldsymbol{A}, \boldsymbol{B}, \pi)$ is the initial model, and $\bar{\lambda} = (\bar{\boldsymbol{A}}, \bar{\boldsymbol{B}}, \bar{\pi})$ is the re-estimated model. Then it can be proved that either:

  1. The initial model, λ, defines a critical point of the likelihood function, in which case $\bar{\lambda} = \lambda$, or

  2. Model $\bar{\lambda}$ is more likely than λ in the sense that $P(\boldsymbol{O}|\bar{\lambda}) > P(\boldsymbol{O}|\lambda)$, i.e., we have found a new model $\bar{\lambda}$ from which the observation sequence is more likely to have been produced.

- Thus we can improve the probability of $\boldsymbol{O}$ being observed from the model if we iteratively use $\bar{\lambda}$ in place of λ and repeat the re-estimation until some limiting point is reached. The resulting model is called the maximum likelihood HMM.

Multiple Observation Sequences

- Speech recognition typically uses left-to-right HMMs. These HMMs can not be trained using a single observation sequence, because only a small number of observations are available to train each state. To obtain reliable estimates of model parameters, one must use multiple observation sequences. In this case, the re-estimation procedure needs to be modified.
- Let us denote the set of K observation sequences as

$$\boldsymbol{O} = \{\boldsymbol{O}^{(1)}, \boldsymbol{O}^{(2)}, \ldots, \boldsymbol{O}^{(K)}\}$$

  where $\boldsymbol{O}^{(k)} = \{o_1^{(k)}, o_2^{(k)}, \ldots, o_{T_k}^{(k)}\}$ is the k-th observation sequence.

- Assume that the observations sequences are mutually independent, we want to estimate the parameters so as to maximize

$$P(\boldsymbol{O} \mid \lambda) = \prod_{k=1}^{K} P(\boldsymbol{O}^{(k)} \mid \lambda) = \prod_{k=1}^{K} P_k$$

Multiple Observation Sequences (cont'd)

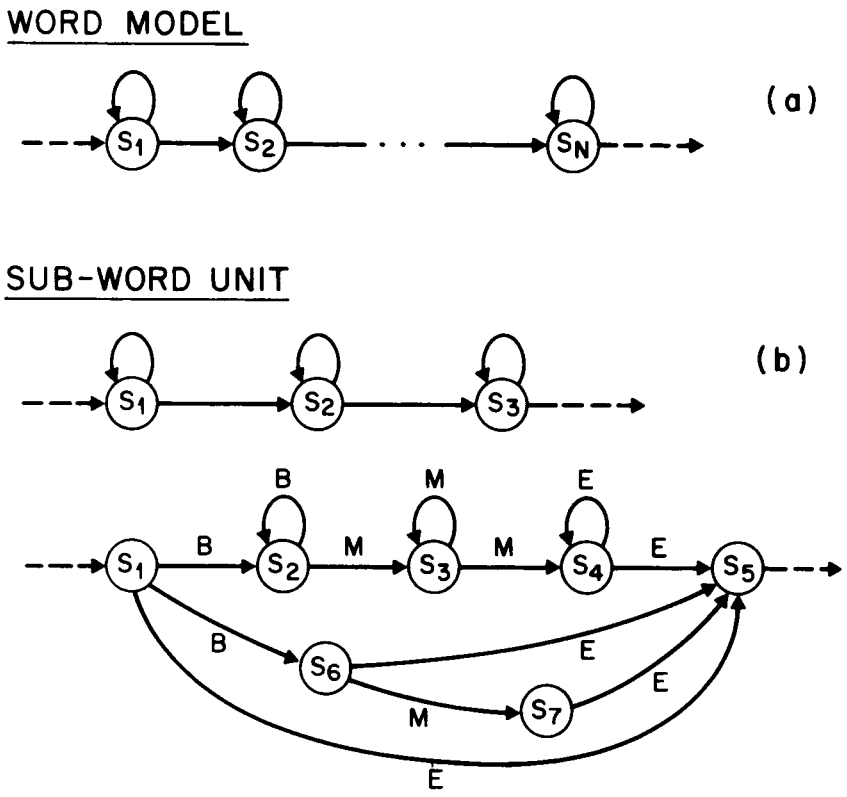
- Since the re-estimation formulas are based on frequency of occurrence of various events, we can modify them by adding up the individual frequencies of occurrence for each sequence

$$\bar{a}_{ij} = \frac{\sum_{k=1}^{K}\sum_{t=1}^{T_k-1}\xi_t^k(i,j)}{\sum_{k=1}^{K}\sum_{t=1}^{T_k-1}\gamma_t^k(i)} = \frac{\sum_{k=1}^{K}\frac{1}{P_k}\sum_{t=1}^{T_k-1}\alpha_t^k(i)a_{ij}b_j(o_{t+1}^{(k)})\beta_{t+1}^k(j)}{\sum_{k=1}^{K}\frac{1}{P_k}\sum_{t=1}^{T_k-1}\alpha_t^k(i)\beta_t^k(i)}$$

$$\bar{b}_j(\ell) = \frac{\sum_{k=1}^{K}\sum_{\substack{t=1\\ o_t^{(k)}=v_\ell}}^{T_k}\gamma_t^k(j)}{\sum_{k=1}^{K}\sum_{t=1}^{T_k}\gamma_t^k(j)} = \frac{\sum_{k=1}^{K}\frac{1}{P_k}\sum_{\substack{t=1\\ o_t^{(k)}=v_\ell}}^{T_k}\alpha_t^k(i)\beta_t^k(i)}{\sum_{k=1}^{K}\frac{1}{P_k}\sum_{t=1}^{T_k}\alpha_t^k(i)\beta_t^k(i)}$$

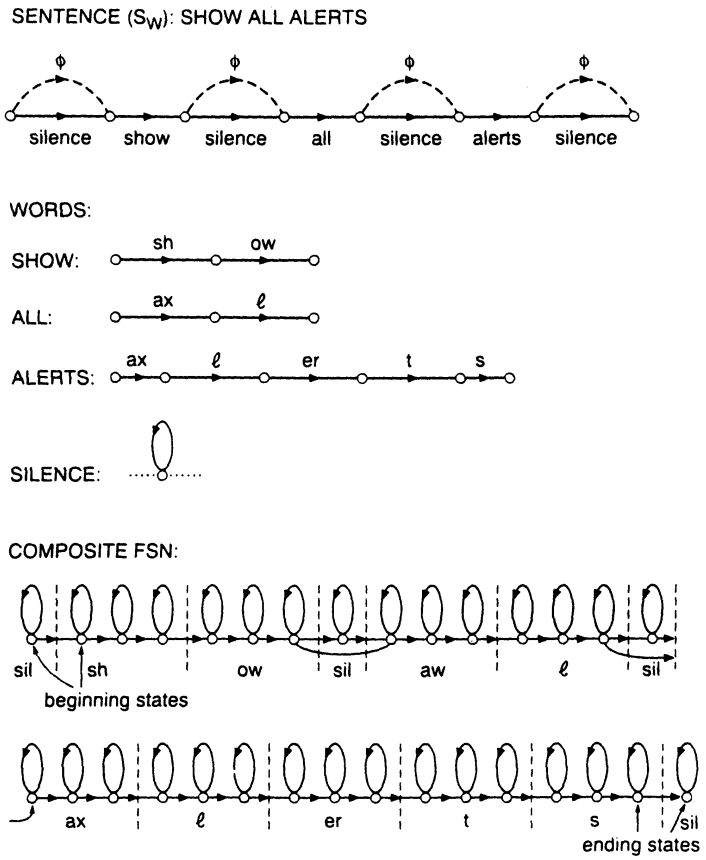
Phone-based HMMs

- Word-based HMMs are appropriate for small vocabulary speech recognition. For large vocabulary ASR, sub-word-based (e.g., phone-based) models are more appropriate.



Phone-based HMMs (cont'd)

- The phone models can have many states, and words are made up from a concatenation of phone models.



Continuous Density Hidden Markov Models

- A *continuous density* HMM replaces the discrete observation probabilities, $b_j(k)$, by a continuous PDF $b_j(\boldsymbol{x})$
- A common practice is to represent $b_j(\boldsymbol{x})$ as a mixture of Gaussians:

$$b_j(\boldsymbol{x}) = \sum_{k=1}^{M} c_{jk} N[\boldsymbol{x}, \mu_{jk}, \boldsymbol{\Sigma}_{jk}] \qquad 1 \le j \le N$$

where c_{jk} is the mixture weight

$c_{jk} \ge 0 \quad (1 \le j \le N, 1 \le k \le M$, and $\sum_{k=1}^{M} c_{jk} = 1, 1 \le j \le N)$,

N is the normal density, and
μ_{jk} and $\boldsymbol{\Sigma}_{jk}$ are the mean vector and covariance matrix associated with state j and mixture k.

Acoustic Modelling Variations

- *Semi-continuous* HMMs first compute a VQ codebook of size M
  - The VQ codebook is then modelled as a family of Gaussian PDFs
  - Each codeword is represented by a Gaussian PDF, and may be used together with others to model the acoustic vectors
  - From the CD-HMM viewpoint, this is equivalent to using the same set of M mixtures to model all the states
  - It is therefore often referred to as a *Tied Mixture* HMM
- All three methods have been used in many speech recognition tasks, with varying outcomes
- For large-vocabulary, continuous speech recognition with sufficient amount (i.e., tens of hours) of training data, CD-HMM systems currently yield the best performance, but with considerable increase in computation

Implementation Issues

- Scaling: to prevent underflow
- *Segmental K-means Training*: to train observation probabilities by first performing Viterbi alignment
- Initial estimates of λ: to provide robust models
- Pruning: to reduce search computation

References

- X. Huang, A. Acero, and H. Hon, *Spoken Language Processing*, Prentice-Hall, 2001.
- F. Jelinek, *Statistical Methods for Speech Recognition.* MIT Press, 1997.
- L. Rabiner and B. Juang, *Fundamentals of Speech Recognition*, Prentice-Hall, 1993.