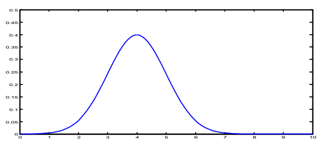
Hidden Markov models

So far: Density estimation (fitting a parametric model, e.g. Mixture of Gaussians) to data $X = \{x_1, x_2, \dots, x_N\}$.



Each data point $x \in R^D$ has **fixed** dimensionality D.

But what about modelling:

Speech? (e.g. model words)

Activities? (e.g. Video of football game)

Natural Language Sentences?

Characteristics: Not fixed duration, temporal dimension (sequence)

Hidden Markov Models

Example: Speech modelling

From several utterances of word “yes”

$X = \{x_1, x_2, \dots, x_N\}$, of different duration/dimensionality

$x_i = (x_i(1), x_i(2), \dots, x_i(T_i))$

Learn $p(x_i; \theta_{yes})$

A model of the sequence of phonemes (like vowels and consonants) that make up the observed speech sound.

Can be used for Automatic Speech Recognition (ASR). How?

Hidden Markov Models

Example: Speech modelling

From several utterances of word "yes"

$X = \{x_1, x_2, \dots, x_N\}$, of different duration/dimensionality

$x_i = (x_i(1), x_i(2), \dots, x_i(T_i))$

Learn $p(x_i; \theta_{yes})$

A model of the sequence of phonemes (like vowels and consonants) that make up the observed speech sound.

Can be used for Automatic Speech Recognition (ASR). How? Learn models for each word and compare the likelihoods:

$$p(x_*; \theta_{yes}) < p(x_*; \theta_{no})$$

First-order Markov Models (not hidden)

Markov Models describe the evolution of a finite state machine.

State at time t is denoted $\omega(t)$. A sequence of length T is denoted :

$$\boldsymbol{\omega}^{1:T} = \{\omega(1), \omega(2), \dots, \omega(T)\}$$

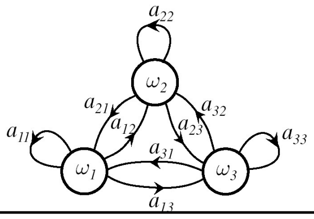
e.g. $\boldsymbol{\omega}^{1:6} = \{\omega_1, \omega_4, \omega_2, \omega_2, \omega_1, \omega_4\}$

Markov Model characterised by : *transition probabilities*

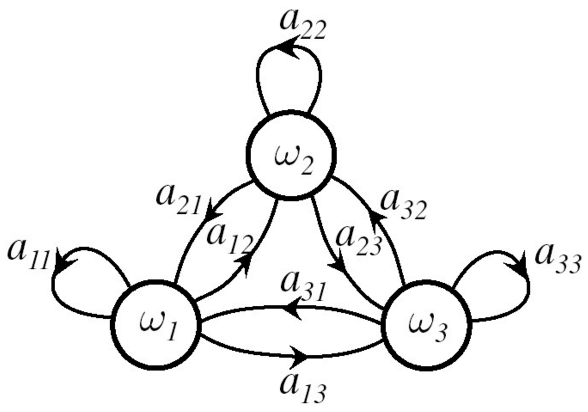
$$P(\omega_j(t+1) \mid \omega_i(t)) = a_{ij}$$

Prob. of moving to state ω_j at time $(t+1)$ given state ω_i at time t.

NOTE: First-order depends only on previous state



Markov Model State Transition Graph



In general, $a_{ij} \neq a_{ji}$ (not necessarily symmetric), a_{ij} can equal 0 and typically $a_{ii} \neq 0$ (non - zero prob. of staying in the same state)

13-5

Calculating the model probability

Given a_{ij} what is the probability the model will generate a particular sequence $\boldsymbol{\omega}^{1:T}$? Note in a Markov model we have

$$P(\omega_j(t) \mid \omega_i(t-1), \omega_k(t-2), \ldots, \omega_l(1)) = P(\omega_j(t) \mid \omega_i(t-1)) = a_{ij}$$

(i.e. it's only dependent on the previous state).

For a sequence of length 2, assuming known initial state $\omega(0) = \omega_1$,

$$\begin{aligned} P(\{\omega_j, \omega_i\}) &= P(\omega_j(2) \mid \omega_i(1)) P(\omega_i(1) \mid \omega_1(0)) \\ &= P(\omega_j(t) \mid \omega_i(t-1)) P(\omega_i(1) \mid \omega_1(0)) = a_{ij} a_{1i} \end{aligned}$$

13-6

Calculating (cont)

In general, if $\omega^{1:T} = \{\omega(1), \omega(2), \ldots, \omega(T)\}$, we have

$$\begin{aligned} P(\omega^{1:T}) &= P(\omega(T) \mid \omega(T-1))P(\omega^{1:T-1}) \\ &= P(\omega(T) \mid \omega(T-1))P(\omega(T-1) \mid \omega(T-2))P(\omega^{1:T-2}) \\ &= \prod_{t=1}^{T} P(\omega(t) \mid \omega(t-1)) \\ &= \prod_{t=1}^{T} a_{[\omega(t-1)][\omega(t)]} \end{aligned}$$

(where again we are assuming we have a fixed $\omega(0) = \omega_1$)

So, we simply multiply the successive transition probabilities.

e.g. for sequence $\boldsymbol{\omega}^{1:T} = \boldsymbol{\omega}^{1:6} = \{\omega_1, \omega_4, \omega_2, \omega_2, \omega_1, \omega_4\}$

we get $P(\boldsymbol{\omega}^{1:T} \mid \boldsymbol{\theta}) = a_{11}a_{14}a_{42}a_{22}a_{21}a_{14}$.

13-7

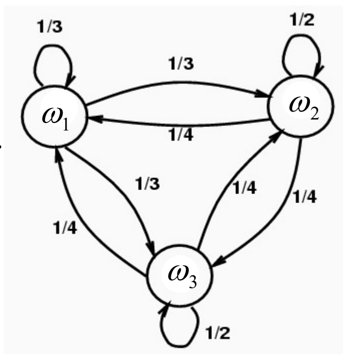
Markov Model (not hidden): Example

Weather example:

$\omega_1 =$ sunny, $\omega_2 =$ cloudy, $\omega_3 =$ rainy.

Suppose we have transition matrix

$$A = [a_{ij}] = \begin{bmatrix} 1/3 & 1/3 & 1/3 \\ 1/4 & 1/2 & 1/4 \\ 1/4 & 1/4 & 1/2 \end{bmatrix}$$



e.g. P("sunny tomorrow given it is cloudy today") $= a_{21} = 1/4$.

Pr. of sequence "$\omega_1\omega_2\omega_3$" (given we start in state ω_1) is:

$P(\{\omega_1, \omega_2, \omega_3\}) = a_{12}a_{23} = (1/3) \times (1/4) = 1/12$.

13-8

Markov: Example 2

Using same model, this time we want

Prob of sequence "$\{\omega_1, \mathrm{X}, \omega_3\}$" where "X" can be anything,

given that we start in state ω_1.

This is :

$$p_{aXb} = P(\{\omega_1,\omega_1,\omega_3\} \text{ or } \{\omega_1,\omega_2,\omega_3\} \text{ or } \{\omega_1,\omega_3,\omega_3\})$$
$$= P(\{\omega_1,\omega_1,\omega_3\}) + P(\{\omega_1,\omega_2,\omega_3\}) + P(\{\omega_1,\omega_3,\omega_3\})$$

(since the sequences are mutually exclusive).

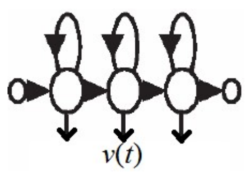
Hence this is

$$p_{aXb} = \sum_{\mathrm{i}=1}^{3} \left(a_{1i}a_{i3}\right) = a_{11}a_{13} + a_{12}a_{23} + a_{13}a_{33}$$
$$= (1/3)(1/3) + (1/3)(1/4) + (1/3)(1/2)$$
$$= 13/36 \approx 0.36$$

13-9

Hidden Markov Model

Generally (e.g. speech recognition) we do not have access to the states $\omega(t)$. Instead, we observe some visible symbols (features), $v(t)$, that are in some way dependent on the current state $\omega(t)$.



In HMMs this relationship is also modelled by probabilities :

$$b_{jk} = P(v_k(t) \mid \omega_j(t))$$

is the probability of emitting symbol v_k at time t given the state ω_j at time t. Instead of observing a state sequence $\omega^{1:\mathrm{T}}$ we observe a visible symbol sequence $\mathbf{V}^{1:T} = \{v(1), v(2), \ldots, v(T)\}$

Hence we have a *hidden* Markov model (hidden state variable).

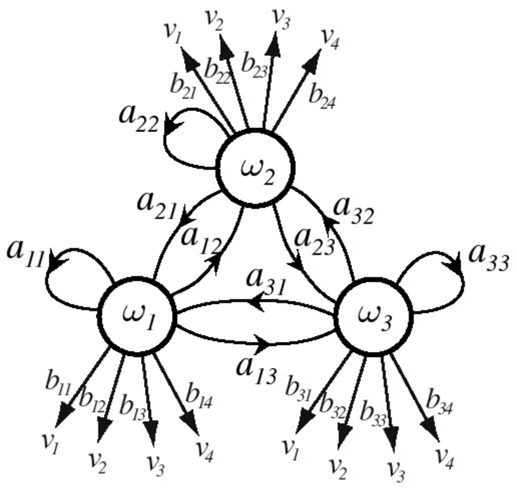
13-10

Hidden Markov model

States denoted by ω_i.

Visible symbols denoted by v_j.

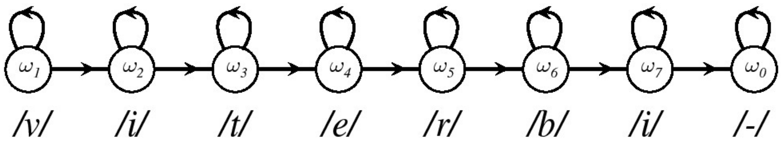
This model shows all state transitions as being possible: not always the case.



13-11

Left-to-Right Models

In speech recognition, most HMMs are *left-to-right* models, i.e. we have $a_{ij} = 0$ if $j < i$, so the state sequence always goes "left to right".



In a left-to-right model we can also specify "no state skips". This means that

$a_{ij} = 0$ unless $j = i$ or $j = i + 1$ or $i = i_{\max}$ and $j = 0$.

The diagram shows a left-to-right model with no state skips.

13-12

Probability Parameters

To summarize, HMMs are defined by the following :

$a_{ij} = P(\omega_j(t+1) \mid \omega_i(t))$ matrix of transition probabilities

$b_{jk} = P(v_k(t) \mid \omega_j(t))$ matrix of emission probabilities

Further, we require that

(1) some transition must occur from step t to $t+1$ (even if to the same state)

(2) some visible symbol must be emitted at each step.

Hence we have

$$\sum_j a_{ij} = 1 \text{ for all } i \quad \text{and} \quad \sum_k b_{jk} = 1 \text{ for all } j$$

We will also assume a given starting state e.g. $\omega(0) = \omega_1$

13-13

3 central issues addressed by HMM

(1) The Evaluation Problem

Given an HMM (i.e. a_{ij} and b_{jk} and $\omega(0)$), what is the probability of generating the sequence $\mathbf{V}^{1:T}$?

(2) The Decoding Problem

Given an HMM and and observation sequence $\mathbf{V}^{1:T}$, determine the most likely hidden state sequence $\boldsymbol{\omega}^{1:T}$.

(3) The Learning Problem

Given the structure of a HMM (i.e. number of states, connections (non - zero transitions) and number of visible symbols), determine the best model (a_{ij} and b_{jk}) for a set of training observations.

13-14

Evaluation

Probability that the model produces visible sequence $\mathbf{V}^{1:T}$:

$$P(\mathbf{V}^{1:T}) = \sum_{r=1}^{r_{\max}} P(\mathbf{V}^{1:T} \mid \boldsymbol{\omega}_r^{1:T}) P(\boldsymbol{\omega}_r^{1:T})$$

where each r indexes a particular sequence of T states

$$\boldsymbol{\omega}_r^{1:T} = \{\omega(1), \omega(2), \ldots, \omega(T)\}$$

For c hidden states we have $r_{\max} = c^T$ sequences.

Since we have a Markov model, 2nd probability term is

$$P(\boldsymbol{\omega}_r^{1:T}) = \prod_{t=1}^{T} P(\omega(t) \mid \omega(t-1))$$

(we've assumed $\omega(0)$ is given).

13-15

Evaluation (cont)

Since emission probs depend only on hidden state,

$$P(\mathbf{V}^{1:T} \mid \boldsymbol{\omega}_r^{1:T}) = \prod_{t=1}^{T} P(v(t) \mid \omega(t))$$

giving

$$P(\mathbf{V}^{1:T}) = \sum_{r=1}^{r_{\max}} \prod_{t=1}^{T} P(v(t) \mid \omega(t)) P(\omega(t) \mid \omega(t-1))$$

This can be evaluated since we know

$$P(\omega_j(t) \mid \omega_i(t-1)) = a_{ij} \quad \text{and} \quad P(v_k(t) \mid \omega_j(t)) = b_{jk}.$$

However, this is an $O(c^T T)$ calculation, e.g. if $c = 10$ and $T = 20$, we need $O(10^{21})$ operations! Can we do better?

13-16

Recursive calculation of $P(V^{1:T})$

Let us write $P(V^{1:T})$ as:

$$P(V^T) = \sum_{\omega(T)} \sum_{\omega(T-1)} \cdots \sum_{\omega(1)} \prod_{t=1}^{T} P(v(t)\,|\,\omega(t)) \times P(\omega(t)\,|\,\omega(t-1))$$

However we don't have to do the calculation in this order!

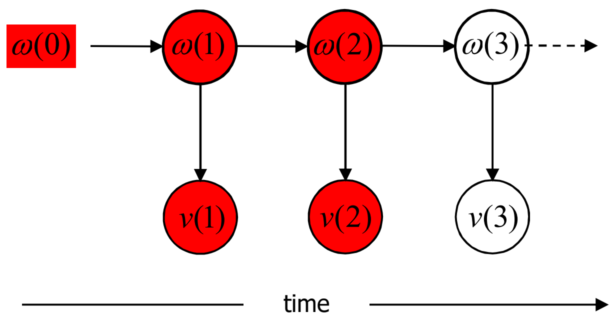
Re-ordering we get the recursion:

$$\underbrace{\sum_{\omega(T)} P(v(T)\,|\,\omega(T)) \sum_{\omega(T-1)} P(\omega(T)\,|\,\omega(T-1)) \times \underbrace{P(v(T-1)\,|\,\omega(T-1)) \sum_{\omega(T-2)} \cdots}_{\boxed{P(V^{T-1},\omega(T\text{-}1))}}}_{\boxed{P(V^T,\omega(T))}}$$

13-17

Recursive calculation of $P(V^{1:T})$

Graphically we can illustrate this as follows (**Note this is not a state transition diagram**). We observe {v(1),v(2),v(3),…}.



We then calculate:

$$P(V^{1:2},\omega(2)) = P(v(2)\,|\,\omega(2)) \sum_{\omega(1)} P(\omega(2)\,|\,\omega(1)) P(v(1),\omega(1))$$

13-18

HMM Forward Algorithm

This recursion is called the forward algorithm.

For all $j = 1,...,c$, we calculate

$$\alpha_j(t) = \begin{cases} 0 & t = 0 \text{ and } j \neq \text{initial state} \\ 1 & t = 0 \text{ and } j = \text{initial state} \\ [\sum_i \alpha_i(t-1)a_{ij}]b_{j[v(t)]} & \text{otherwise} \end{cases}$$

where $b_{j[v(t)]}$ is the emission prob from state j for the visible symbol emitted at time t.

$\alpha_j(t)$ represents the prob that HMM is in hidden state ω_j at time t having generated first t elements of $\mathbf{V}^{1:T}$.

13-19

HMM Forward Algorithm (cont)

The Forward Algorithm can be written :

1. Initialize $t \leftarrow 0, a_{ij}, b_{jk}$, visible sequence $\mathbf{V}^{1:T}$, $\alpha_j(0)$
2. Increment $t \leftarrow t+1$
3. $\alpha_j(t) \leftarrow b_{j[v(t)]}\sum_{i=1}^{c}\alpha_i(t-1)a_{ij}$ for all j
4. If $t < T$, repeat from 2.
5. Return $P(\mathbf{V}^{1:T}) \leftarrow \sum_j \alpha_j(T)$

If final state α_0 is known, we can return $P(\mathbf{V}^{1:T}) \leftarrow \alpha_0(T)$ at Step 5.

The Forward Algorithm has complexity $O(c^2T)$.

E.g. for $c = 10, T = 20$, needs $O(2000)$ calculations.

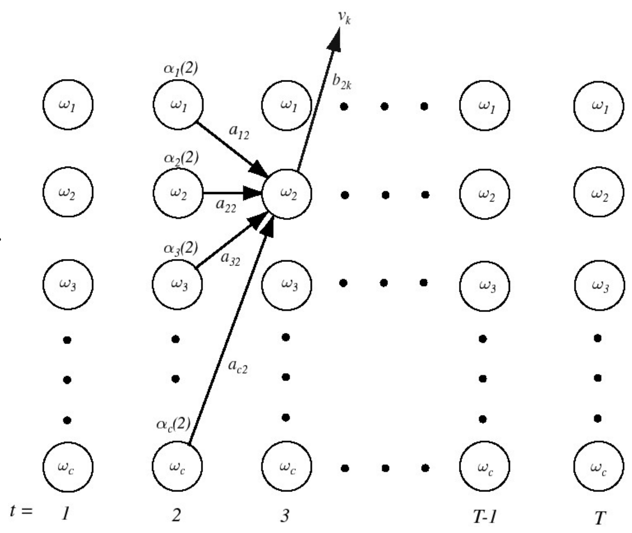
13-20

Forward Algorithm Step

Prob that HMM was in state $\omega_i(t=2)$ and generated observed sequence up to $t=2$ is $\alpha_i(t)$ for $i=1,2,\ldots,c$.

To find $\alpha_2(3)$ (which emits v_k) we use
$\alpha_2(3) =$

$$b_{2k}\sum_{i=1}^{c}\alpha_i(2)a_{i2}$$



13-21

Evaluation Example

Consider HMM with absorber state ω_0, null symbol v_0, and

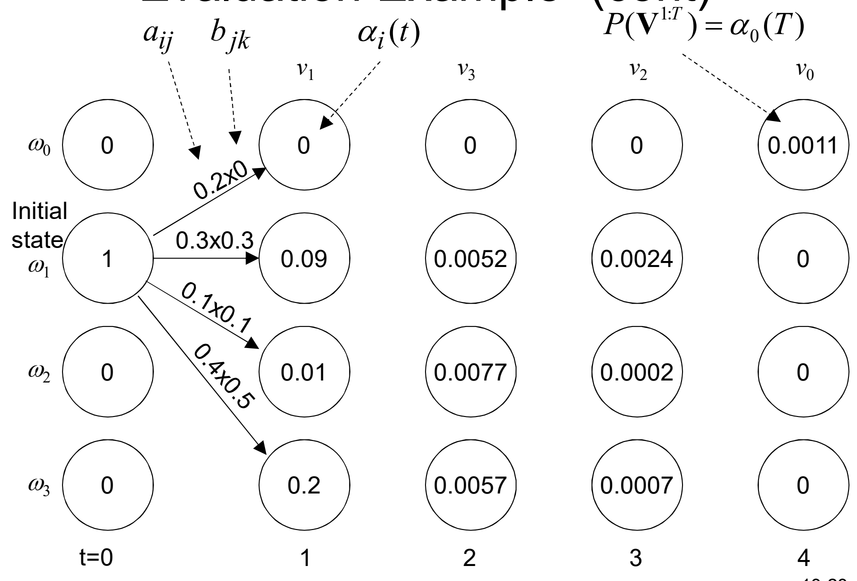
$$a_{ij} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0.2 & 0.3 & 0.1 & 0.4 \\ 0.2 & 0.5 & 0.2 & 0.1 \\ 0.8 & 0.1 & 0.0 & 0.1 \end{pmatrix} \quad b_{jk} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0.3 & 0.4 & 0.1 & 0.2 \\ 0 & 0.1 & 0.1 & 0.7 & 0.1 \\ 0 & 0.5 & 0.2 & 0.1 & 0.2 \end{pmatrix}$$

(matrix indexes start at 0, so top row is for ω_0)

What is the probability this generates the sequence
$\mathbf{V}^{1:4} = \{v_1, v_3, v_2, v_0\}$, given initial state at $t=0$ is ω_1 and end state ω_0?

13-22

Evaluation Example (cont)



13-23

Making Decisions

Given the ability to calculate the probability of an observed sequence. We can now compare different HMMs. This is just Bayesian Decision theory revisited!

Recall:

$$P(\boldsymbol{\theta} \mid \mathbf{V}^{1:T}) = \frac{P(\mathbf{V}^{1:T} \mid \boldsymbol{\theta})P(\boldsymbol{\theta})}{P(\mathbf{V}^{1:T})}$$

Hence given model θ$_1$ and θ$_2$ (e.g. HMMs for different words) we select θ$_1$ if:

$$P(\boldsymbol{\theta_1} \mid \mathbf{V}^{1:T}) > P(\boldsymbol{\theta_2} \mid \mathbf{V}^{1:T})$$

or equivalently

$$P(\mathbf{V}^{1:T} \mid \boldsymbol{\theta_1})P(\boldsymbol{\theta_1}) > P(\mathbf{V}^{1:T} \mid \boldsymbol{\theta_2})P(\boldsymbol{\theta_2})$$

We can calculate this since Forward Alg. gives us $P(\mathbf{V}^{1:T} \mid \boldsymbol{\theta})$ and $P(\boldsymbol{\theta})$ is our prior belief in the model (e.g. from a language model).

Example: suppose θ$_1$ = 'y'-'e'-'s' and θ$_2$ = 'n'-'o'. If we expect that the answer is more likely to be 'yes' we weight the priors accordingly.

13-24

An Alternative Recursion

Alternatively given:

$$P(V^{1:T}) = \sum_{\omega(T)} \sum_{\omega(T-1)} \cdots \sum_{\omega(1)} \prod_{t=1}^{T} P(v(t) \mid \omega(t)) \times P(\omega(t) \mid \omega(t-1))$$

We can re-order as:

$$\sum_{\omega(1)} P(v(1) \mid \omega(1)) \underbrace{\sum_{\omega(2)} P(\omega(2) \mid \omega(1)) \times P(v(2) \mid \omega(2)) \underbrace{\sum_{\omega(3)} P(\omega(3) \mid \omega(2)) \times \ldots}_{\boxed{P(v(3)\ldots v(T) \mid \omega(2))}}}_{\boxed{P(v(2), v(3), \ldots, v(T) \mid \omega(1))}}$$

13-25

The Backward Algorithm

This is called the Backward algorithm. Just as $\alpha_i(t)$ is prob that model is in state $\omega_i(t)$ and given $\{v(1)\ldots v(t)\}$, we now define $\beta_i(t)$ as prob that model is in state $\omega_i(t)$ and *will generate* remaining $\{v(t+1),\ldots,v(T)\}$. We have:

$$\beta_i(t) = \begin{cases} 1 & t = T \\ \sum_j \beta_j(t+1) a_{ij} b_{j[v(t+1)]} & \text{otherwise} \end{cases}$$

which we can calculate recursively, starting from $t = T$ and working backwards.

13-26

Backward Algorithm

We know $\beta_i(0) = P(\mathbf{V}^{1:T} \mid \boldsymbol{\theta})$ for the known initial state, since this is the probability of being in the initial state and generating the entire sequence.

We therefore have the following *Backward Algorithm*:

1. Initialize $t \leftarrow T, a_{ij}, b_{jk}$, visible sequence $\mathbf{V}^{1:T}$, $\beta_j(T)$
2. Decrement $t \leftarrow t-1$
3. $\beta_i(t) = \sum_{j=1}^{c} \beta_j(t+1) a_{ij} b_{j[v(t+1)]}$
4. If $t > 1$, repeat from 2.
5. Return $P(\mathbf{V}^T \mid \boldsymbol{\theta}) \leftarrow \beta_i(0)$ for the known initial state

13-27

Decoding Problem

The problem is to choose the most likely state sequence, $\omega^{1:T}$, for a given observation sequence $V^{1:T}$.

13-28

Viterbi Algorithm

Finds the single most probable state sequence, $\omega^{1:T}$.

Suppose that we have the probability of the best sequences up to time t that end in state ω_i. Define

$$\delta_i(t) = \max_{\omega(1)\cdots\omega(t-1)} P(\omega(1)\cdots\omega(t-1), \omega(t)=\omega_i, v(1)\cdots v(t))$$

recall $P(\omega^{1:T}, V^{1:T}) = P(v(t)\mid\omega(t))P(\omega(t)\mid\omega(t-1))P(\omega^{t-1}, V^{t-1})$, hence

$$\delta_j(t+1) = \max_i[\delta_i(t)a_{ij}]b_{j,v(t+1)}$$

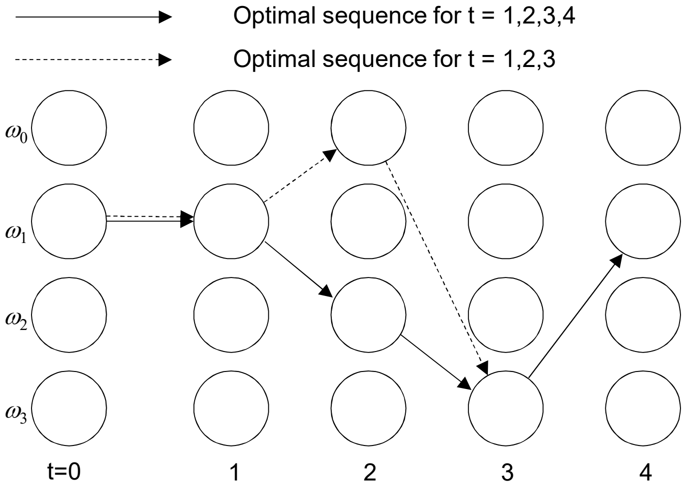
To find the path, we keep track of the argument i of max using

$$\psi_j(t+1) = \arg\max_i[\delta_i(t)a_{ij}].$$

Note the similarity to the forward algorithm. Here we have replaced the $\sum_i$ operation by a $\max_i$ operation.

13-29

Viterbi: is this possible?


Optimal sequence for t = 1,2,3,4
Optimal sequence for t = 1,2,3
ω_0
ω_1
ω_2
ω_3
t=0
1
2
3
4

If not – why not?

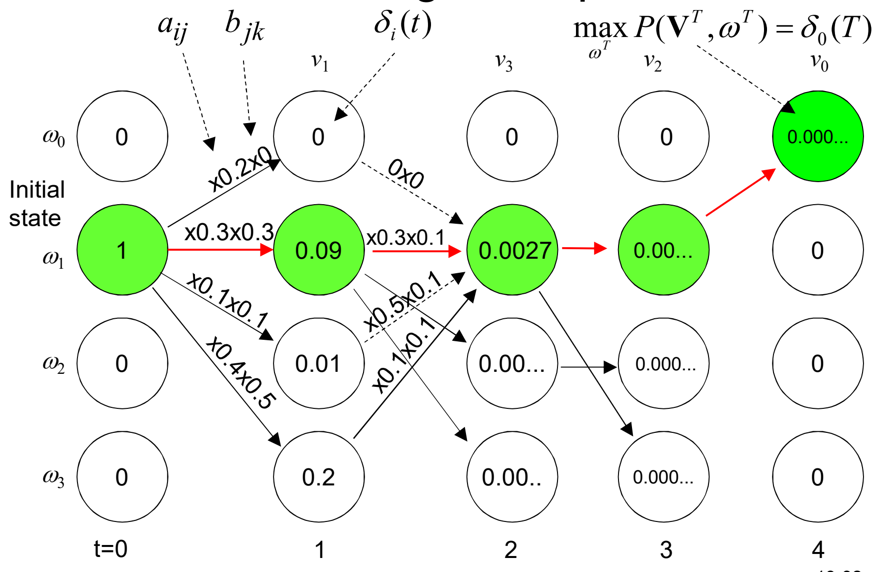
13-30

Viterbi Algorithm

1. Initialize: $\delta_i(0) = 1$ for init state $\tilde{\omega}(0)$, 0 otherwise; $t \leftarrow 0$
2. Recursion: $t \leftarrow t+1$
3. $\delta_j(t) \leftarrow \max_i [\delta_i(t-1) a_{ij}] b_{j,v(t)}$
4. $\psi_j(t) \leftarrow \arg\max_i [\delta_i(t-1) a_{ij}]$
5. If $t < T$ repeat from 2.
6. Termination: $\tilde{\omega}(T) \leftarrow \arg\max_i \delta_i(T)$
7. Backtracking:
8. $t \leftarrow t-1$; $\tilde{\omega}(t) \leftarrow \psi_{\tilde{\omega}(t+1)}(t+1)$
9. If $t > 1$ repeat from 8.
10. Output state sequence $\{\tilde{\omega}(0), \ldots, \tilde{\omega}(T)\}$.

13-31

Decoding Example



13-32

The Learning Problem

The 3rd problem is the most difficult. Aim: to learn the parameters, a_{ij} and b_{jk} from a set of training data.

Obvious approach: *Maximum Likelihood Learning*

However we have a familiar problem:

$$\begin{aligned}\{\hat{a}_{ij}, \hat{b}_{jk}\} &= \arg\max_{a_{ij}, b_{jk}} P(V^{1:T} \mid a_{ij}, b_{jk}) \\ &= \arg\max_{a_{ij}, b_{jk}} \sum_{\omega^{1:T}} P(\omega^{1:T}, V^{1:T} \mid a_{ij}, b_{jk})\end{aligned}$$

That is: we must marginalise out the state sequences, $\omega^{1:T}$.

13-33

The Learning Problem

This problem is very similar to learning in MoGs. Hence we again use an EM algorithm. In MoG model we updated the prior probability weights, w_k, as:

$$w_k^{(i)} = \frac{1}{N} \sum_n P(x_n \sim \text{model } k \mid \mu, \Sigma, w)$$

The equivalent here is to determine:

The prob. of moving from state *i* to state *j* and emitting visible symbol v_k when in state *j*

This can be calculated using the Forward-Backward algorithm.

13-34

Improving the estimates

To begin we need

$$\gamma_{ij}(t) = P(\omega_j(t+1), \omega_i(t) \mid V^{1:T}, a_{ij}, b_{jk})$$

since:

$\alpha_i(t-1)$ is prob. of generating $\mathbf{V}^{1:T}$ up to $t-1$ and $\omega_i(t-1)$

$a_{ij}b_{j[v(t)]}$ is prob. of transition and generating $v(t)$

and $\beta_j(t)$ is prob of generating $\mathbf{V}^{1:T}$ after t and $\omega_j(t)$

we have

$$\gamma_{ij}(t) = \frac{\alpha_i(t-1)a_{ij}b_{j,v(t)}\beta_j(t)}{P(\mathbf{V}^T \mid \boldsymbol{\theta})}$$

where $P(\mathbf{V}^{1:T} \mid \boldsymbol{\theta})$ is prob of generating $\mathbf{V}^{1:T}$ by any path.

13-35

Improving the estimates for a_{ij}

Now, expected number of transitions from state $\omega_i(t-1)$ to state $\omega_j(t)$ at *any* time t is $\sum_{t=1}^{T} \gamma_{ij}(t)$.

Expected number of transitions from state ω_i (at any time) to *any* state is $\sum_{t=1}^{T} \sum_k \gamma_{ik}(t)$.

Therefore estimate of prob. of transition to state ω_j given we are currently in state ω_i is ratio of these values

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T} \gamma_{ij}(t)}{\sum_{t=1}^{T} \sum_k \gamma_{ik}(t)}$$

13-36

Improving the estimates b_{jk}

Similarly, for new estimate of b_{jk} calculate ratio between frequency that symbol v_k emitted in state ω_j and frequency that any symbol emitted in state ω_j.
We have

$$\hat{b}_{jk} = \frac{\sum_{t:v(t)=v_k} \sum_{l} \gamma_{lj}(t)}{\sum_{t=1}^{T} \sum_{l} \gamma_{lj}(t)}$$

i.e. a weighted average as in the EM algorithm for MoGs

13-37

Baum-Welch (forward-backward) Alg

```
1. Initialize $a_{ij}, b_{jk}, \mathbf{V}^T$, stopping criterion $\theta$, counter $n \leftarrow 0$
2. $n \leftarrow n+1$
3. Compute $\gamma_{ij}(t)$ using $a_{ij}^{(n-1)}$ and $b_{jk}^{(n-1)}$
4. Compute $\hat{a}_{ij}^{(n)}$ using $\gamma_{ij}(t)$ as defined above
5. Compute $\hat{b}_{jk}^{(n)}$ using $\gamma_{ij}(t)$ as defined above
6. $a_{ij}^{(n)} \leftarrow \hat{a}_{ij}^{(n)}$ ; $b_{jk}^{(n)} \leftarrow \hat{b}_{jk}^{(n)}$
7. If $\max_{i,j,k}\{\|a_{ij}^{(n)} - a_{ij}^{(n-1)}\|, \|b_{jk}^{(n)} - b_{jk}^{(n-1)}\|\} \geq$ threshold
   repeat from 2.
8. Return $a_{ij} \leftarrow a_{ij}^{(n)}, b_{jk} \leftarrow b_{jk}^{(n)}$
```

13-38

HMMs for speech recognition

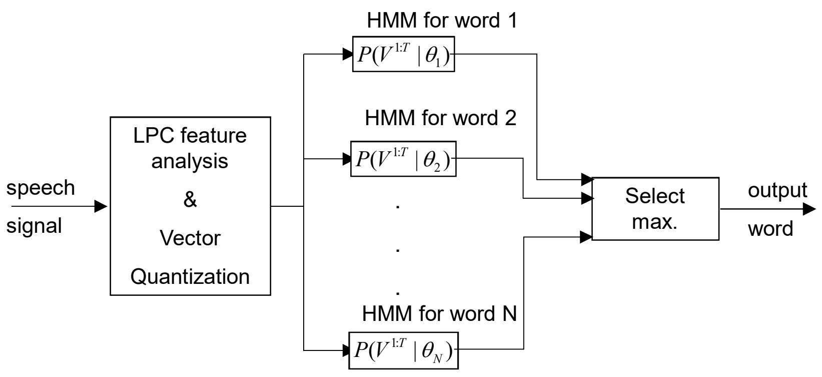
In practice, in ASR, the observed data is usually a measure of the short term spectral properties of the speech. It is therefore necessary to interface the HMM with this continuous feature space. There are two popular approaches:

1. Continuous Density observations – The finite states *ω(t)* are mapped into a continuous feature space using a MoG density model.
2. VQ observations – the continuous feature space is discretised into a finite symbol set using vector quantization.

13-39

HMMs for speech recognition

An example of an isolated word HMM recognition system:



13-40

Limitations of HMMs

We have seen how to do inference and learning using HMMs. These have proved very valuable in Automatic Speech Recognition. However it is important to note some limitations:

1. HMMs assume that the sequence of individual observations are independent of each other.

2.The Markov transition implies that the probability of remaining in a given state decays exponentially with time – this may be too crude an approximation

For further information on HMMs, particularly for speech recognition see the excellent tutorial paper:

L. R. Rabiner, A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition, *Proceedings of the IEEE*, **77**(2), 257-286, 1999).

13-41