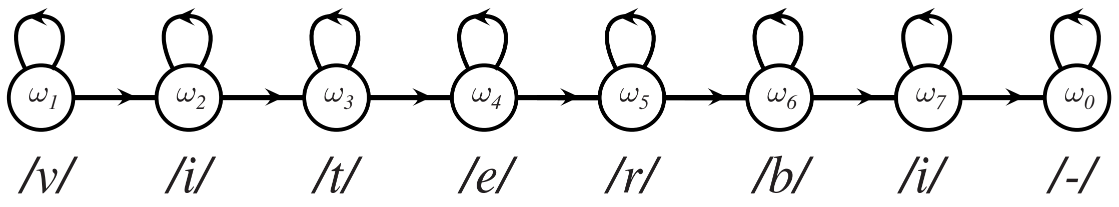
Hidden Markov Models

DHS 3.10

HMMs

Model likelihood of a *sequence* of observations as a series of state transitions.

- Set of states set in advance; likelihood of state transitions, observed features from each state learned
    - Each state has an associated feature space and density/likelihood function ($p(x \mid \omega_i)$)
- Often used to find most likely sequence of state transitions, according to the model
- Applications: speech recognition, handwriting recognition



First-Order Markov Models

Represent Probabilistic State Transitions

"First Order:" probability of a state for each time step depends only on the *previous* state:

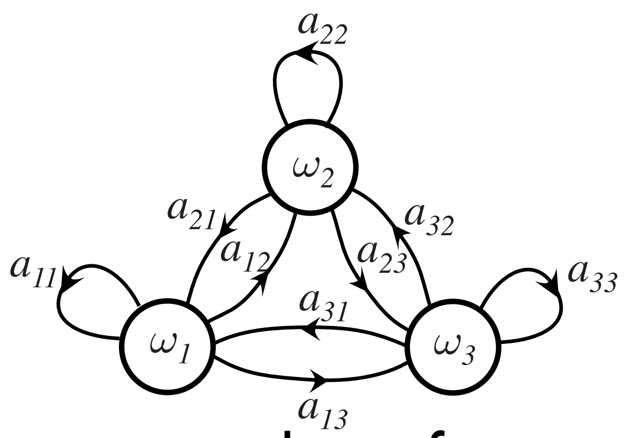
$$P(\omega_j(t+1)|\omega_i(t)) = a_{ij} \qquad \sum_{j=1}^{|\Omega|} a_{ij} = 1$$

Transition probabilities coming out of each state sum to one. States at times t and t+1 may be the same.

R·I·T

Example: First-Order M. Model

Consider State Sequence w⁶



$$\omega^6 = \{\omega_1, \omega_3, \omega_2, \omega_2, \omega_1, \omega_3\}$$

Probability of this Sequence

Given transition probability table θ, first state known: product of transition probabilities, a_{ij}

$$P(\omega^6|\theta) = a_{13}a_{32}a_{22}a_{21}a_{13}$$

Problem

In practice, we often can't directly observe the states of interest (e.g. speech recognition: wave features, not phonemes as input)

R·I·T

First-Order HMMs

Modification

As we can't directly observe states, let's model the likelihood of observed features for each state

- Add 'visible' states **V** for observed features
- States of interest are 'hidden' (must be inferred)

Restriction

We will discuss HMMs where the observations are discrete

Observations

Are now sequences of *visible* states

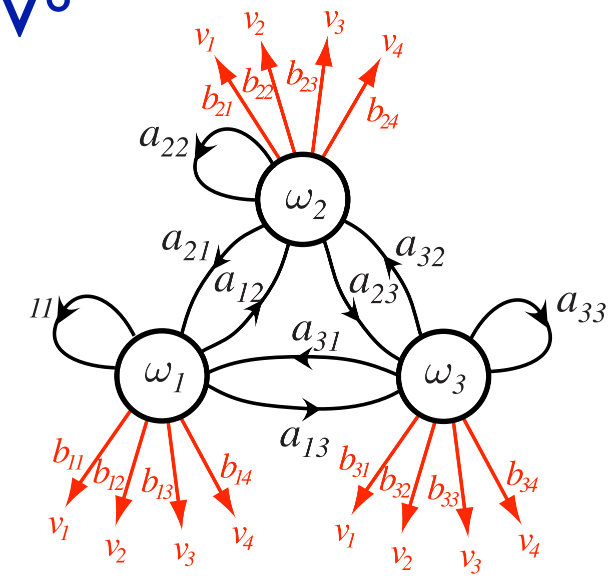
Probability of transition to visible state depends on current (hidden state); transitions to visible states from each hidden state must sum to one

$$b_{jk} = P(v_k(t) | \omega_j(t)) \qquad \sum_k b_{jk} = 1$$

Example: HMM

Consider Visible State Sequence V⁶

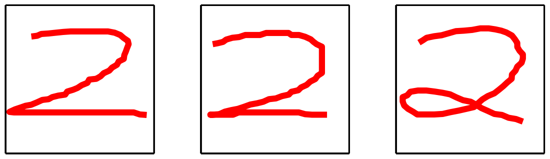
$$\mathbf{V}^6 = \{v_4, v_1, v_1, v_4, v_2, v_3\}$$



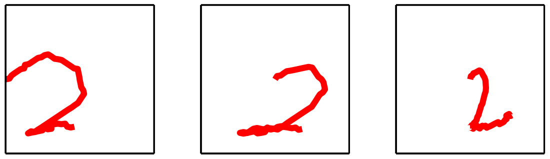
Sequence of Hidden States

Is non-unique, even if we know that we begin in state one. However, some may be more likely than others.

HMMs: On-line Handwriting Recognition



(example from Bishop, "Pattern Recognition and Machine Learning")



**Examples generated from the HMM

Symbols

Represented as a sequence of (x,y) locations for each pen stroke

A Simple HMM

16 states representing a line segment of fixed length in one of 16 angles: another 'left-to-right' model with a fixed sequence of states

Key Problems

Evaluation

Computing the probability that a sequence of visible states was generated by an HMM

Decoding

Determine the most likely sequence of *hidden* states that produced a sequence of visible states

Learning

Given the HMM structure (number of visible and hidden states) and a training set of visible state sequences, determine the transition probabilities for hidden and visible states.

Evaluation

Probability of Visible Sequence V^T:

Is the sum of probabilities for the sequence over all possible length T paths through the hidden states:

$$P(\mathbf{V}^T) = \sum_{r=1}^{r_{max}} P(\mathbf{V}^T|\omega_{\mathbf{r}}{}^T)P(\omega_{\mathbf{r}}{}^T)$$

where $r_{max} = c^T$ for c hidden states. State T is a final or *absorbing* state (e.g. silence in speech applications), ω_0

Evaluation, Cont'd

Probability of Visible Sequence (alt.)

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

Interpretation

Sum over all possible hidden state sequences of: conditional probability of each transition multiplied by probability of visible symbol being emitted from current hidden state

Problem

Computation is $O(c^T T)$ e.g. c =10, T = 20: ~10^{21}

Forward Algorithm

(Much) Faster Evaluation

In $O(c^2 T)$ time

Idea

Compute the probability recursively for each state at each time step from 1...T:

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

(prob. of visible symbol v_k)

Return $\alpha_0(T)$ (probability of final state ω_0 at last time step)

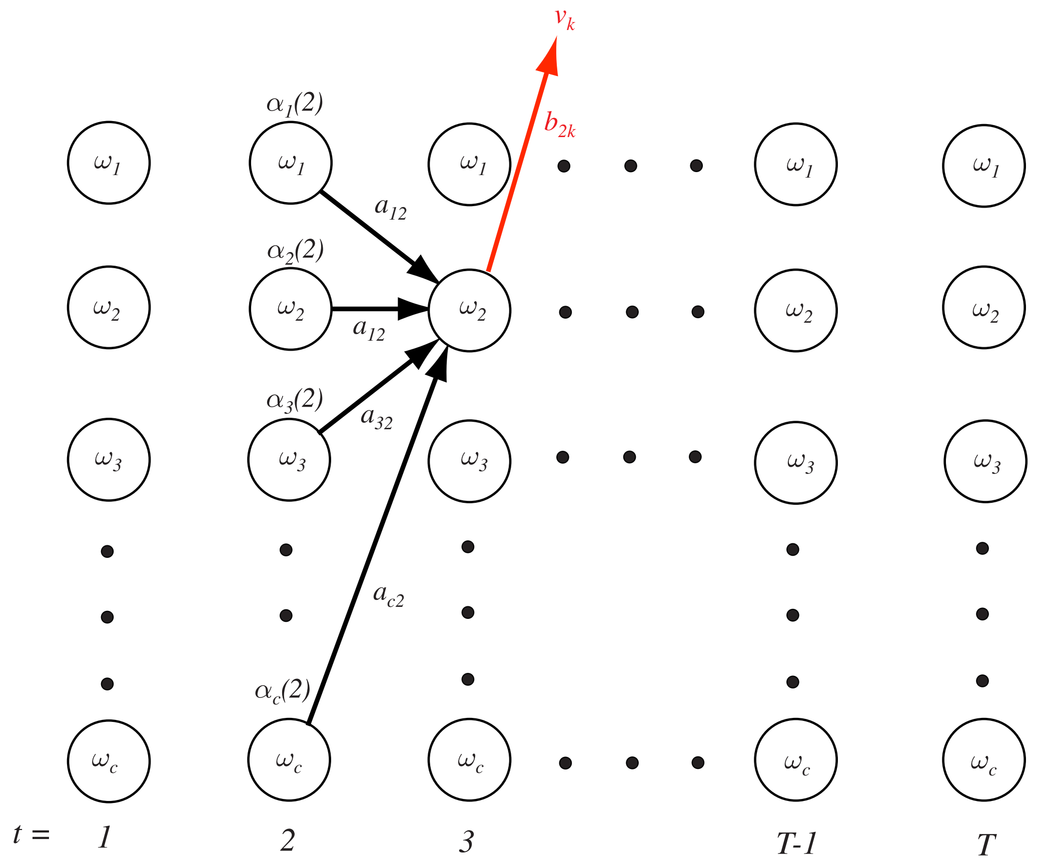


FIGURE 3.10. The computation of probabilities by the Forward algorithm can be visualized by means of a trellis—a sort of "unfolding" of the HMM through time. Suppose we seek the probability that the HMM was in state ω_2 at $t = 3$ and generated the observed visible symbol up through that step (including the observed visible symbol v_k). The probability the HMM was in state $\omega_j(t = 2)$ and generated the observed sequence through $t = 2$ is $\alpha_j(2)$ for $j = 1, 2, \ldots, c$. To find $\alpha_2(3)$ we must sum these and multiply the probability that state ω_2 emitted the observed symbol v_k. Formally, for this particular illustration we have $\alpha_2(3) = b_{2k} \sum_{j=1}^{c} \alpha_j(2) a_{j2}$. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

Prob. of HMM for Observation

Prob. of model, given visible sequence: $P(\theta|\mathbf{V}^T) = \dfrac{P(\mathbf{V^T}|\theta)P(\theta)}{P(\mathbf{V}^T)}$

Use Bayes Formula

Forward algorithm provides $P(V^T \mid \theta)$

Other two parameters are provided by domain knowledge

- likelihood of the sequence itself, and prior probability of the model; e.g. using a language model for speech recognition
- Probability of the HMM (model, $P(\theta)$) often assumed uniform, and ignored for classification

R·I·T

Decoding: Finding Hidden States

Decoding

Find most likely sequence of hidden states in HMM

Brute Force (Enumerate Sequences):

Again $O(c^T T)$; need a more efficient approach

Greedy Algorithm (with modification, $O(c^2 T)$)

Modify forward algorithm so that we add the most likely hidden state to a list after updating the state probabilities at time *t*. Return the list.

- Not guaranteed to produce a valid path.

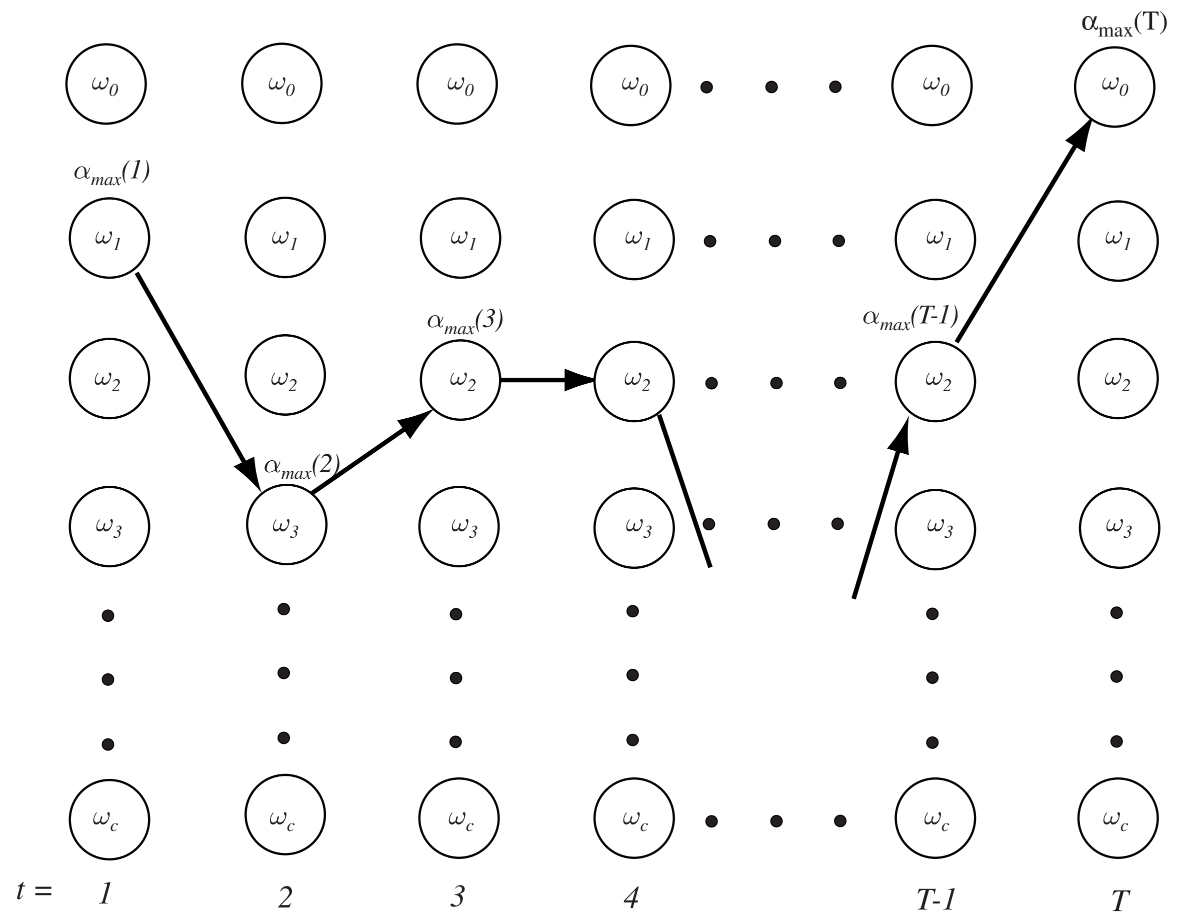


FIGURE 3.12. The decoding algorithm finds at each time step t the state that has the highest probability of having come from the previous step and generated the observed visible state v_k. The full path is the sequence of such states. Because this is a local optimization (dependent only upon the single previous time step, not the full sequence), the algorithm does not guarantee that the path is indeed allowable. For instance, it might be possible that the maximum at $t = 5$ is ω_1 and at $t = 6$ is ω_2, and thus these would appear in the path. This can even occur if $a_{12} = P(\omega_2(t+1)|\omega_1(t)) = 0$, precluding that transition. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

Viterbi Algorithm

Purpose

Decoding; computes the most likely sequence of states for an observation sequence

- State sequence will be *consistent* with HMM transition probabilities

Brief Summary

A modified 'shortest path' dynamic programming algorithm: incrementally computes the most likely sequence of states from start to finish state over trellis of possible states, maintaining the best sequence for each state at time *t*

Example:

http://www.cim.mcgill.ca/~latorres/Viterbi/va_main.html

Backward Algorithm

Computes $P(V^T \mid \theta)$

Like the forward algorithm, but moving from the final state back to the initial state. The algorithm computes the recurrence:

$$\beta_j(t) = \begin{cases} 0, & if\ t = T\ and\ j \neq\ final\ state \\ 1, & if\ t = T\ and\ j\ =\ final\ state \\ \sum_j \beta_j(t+1) a_{ij}\ \underbrace{b_{jk} v(t+1)}_{\text{(prob. of visible symbol } v_k)} & otherwise \end{cases}$$

and returns $\beta_k(0)$, for initial state *k*

Learning

Optimal HMM Parameter Setting

For transition probabilities; no known method.

Forward-Backward Algorithm

A form of *expectation-maximization (EM);* iteratively updates transitions to better fit the observed sequences

- Also known as the *Baum-Welch* algorithm

Individual Transition Probabilities

$$\gamma_{ij}(t) = \frac{\overset{\text{forward}}{\alpha_i(t-1)}\,\overset{\text{trans+emit}}{a_{ij}b_{jk}}\,\overset{\text{backward}}{\beta_j(t)}}{P(\mathbf{V}^T|\theta)}$$

(evaluation: e.g. by forward alg.)

Estimate Updates:

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

expected state i-j transitions (at any *t*)

expected state i-*any* transitions (at any *t*)

$$\hat{b}_{jk} = \frac{\sum_{\substack{t=1 \\ v(t)=v_k}}^{T} \sum_l \gamma_{jl}(t)}{\sum_{t=1}^{T} \sum_l \gamma_{jl}(t)}$$

expected state j transmitting v_k

expected state j transmitting any symbol

(transmitted after transition to state *l*)

Forward-Backward Algorithm

Initialization:

- Given training sequence V^T, convergence threshold λ
- Set transition probabilities randomly

Do:

- Compute updated hidden and visible state transition estimates, per last slide
- Compute largest difference between previous and current transition (a_{ij}) and emission (b_{ij}) transition estimates

Until: Largest estimated difference is $< \lambda$ (convergence)

- Return transition estimates (a_{ij}) and emission estimates (b_{ij})