called a hidden Markov model) is a doubly embedded stochastic process with an underlying stochastic process that is *not* directly observable (it is hidden) but can be observed only through another set of stochastic processes that produce the sequence of observations.

To illustrate the basic concepts of the hidden Markov model, we will use several simple examples including simple coin-tossing experiments. We begin with a review of some basic ideas of probability in the following exercise.

Exercise 6.1

Given a single fair coin, i.e., $P(\text{Heads}) = P(\text{Tails}) = 0.5$, which you toss once and observe Tails,

1. What is the probability that the next 10 tosses will provide the sequence $(HHTHTTHTTH)$?
2. What is the probability that the next 10 tosses will produce the sequence $(HHHHHHHHHH)$?
3. What is the probability that 5 of the next 10 tosses will be tails? What is the expected number of tails over the next 10 tosses?

Solution 6.1

1. For a fair coin, with independent coin tosses, the probability of any specific observation sequence of length 10 (10 tosses) is $(1/2)^{10}$ since there are 2^{10} such sequences and all are equally probable. Thus:

$$P(H\,H\,T\,H\,T\,T\,H\,T\,T\,H) = \left(\frac{1}{2}\right)^{10}.$$

2.

$$P(H\,H\,H\,H\,H\,H\,H\,H\,H\,H) = \left(\frac{1}{2}\right)^{10}.$$

Thus a specified run of length 10 is as likely as a specified run of interlaced H and T.

3. The probability of 5 tails in the next 10 tosses is just the number of observation sequences with 5 tails and 5 heads (in any order) and this is

$$P(5H, 5T) = \binom{10}{5}\left(\frac{1}{2}\right)^{10} = \frac{252}{1024} \cong 0.25$$

since there are $\binom{10}{5}$ ways of getting $5H$ and $5T$ in 10 tosses, and each sequence has probability of $\left(\frac{1}{2}\right)^{10}$. The expected number of tails in 10 tosses is

$$E(T \text{ in } 10 \text{ tosses}) = \sum_{d=0}^{10} d\binom{10}{d}\left(\frac{1}{2}\right)^{10} = 5.$$

Thus, on average, there will be $5H$ and $5T$ in 10 tosses, but the probability of exactly $5H$ and $5T$ is only 0.25.

6.3.1 Coin-Toss Models

Assume the following scenario. You are in a room with a barrier (e.g., a curtain) through which you cannot see what is happening. On the other side of the barrier is another person who is performing a coin-tossing experiment (using one or more coins). The person will not tell you which coin he selects at any time; he will only tell you the result of each coin flip. Thus a sequence of *hidden* coin-tossing experiments is performed, with the observation sequence consisting of a series of heads and tails. A typical observation sequence would be

$$\begin{aligned} \mathbf{O} &= (\mathbf{o}_1\, \mathbf{o}_2\, \mathbf{o}_3 \ldots \mathbf{o}_T) \\ &= (HHTTTHTTH \ldots H) \end{aligned}$$

where H stands for heads and T stands for tails.

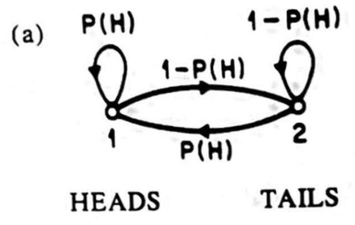
Given the above scenario, the question is, How do we build an HMM to explain (model) the observed sequence of heads and tails? The first problem we face is deciding what the states in the model correspond to, and then deciding how many states should be in the model. One possible choice would be to assume that only a single biased coin was being tossed. In this case, we could model the situation with a two-state model in which each state corresponds to the outcome of the previous toss (i.e., heads or tails). This model is depicted in Figure 6.3a. In this case, the Markov model is observable, and the only issue for complete specification of the model would be to decide on the best value for the single parameter of the model (i.e., the probability of, say, heads). Interestingly, an equivalent HMM to that of Figure 6.3a would be a degenerate one-state model in which the state corresponds to the single biased coin, and the unknown parameter is the bias of the coin.

A second HMM for explaining the observed sequence of coin toss outcomes is given in Figure 6.3b. In this case there are two states in the model, and each state corresponds to a different, biased coin being tossed. Each state is characterized by a probability distribution of heads and tails, and transitions between states are characterized by a state-transition matrix. The physical mechanism that accounts for how state transitions are selected could itself be a set of independent coin tosses or some other probabilistic event.

A third form of HMM for explaining the observed sequence of coin toss outcomes is given in Figure 6.3c. This model corresponds to using three biased coins, and choosing from among the three, based on some probabilistic event.

Given the choice among the three models shown in Figure 6.3 for explaining the observed sequence of heads and tails, a natural question would be which model best matches the actual observations. It should be clear that the simple one-coin model of Figure 6.3a has only one unknown parameter; the two-coin model of Figure 6.3b has four unknown parameters; and the three-coin model of Figure 6.3c has nine unknown parameters. Thus, with the greater degrees of freedom, the larger HMMs would seem to be inherently more capable of modeling a series of coin-tossing experiments than would equivalently smaller models. Although this is theoretically true, we will see later in this chapter that practical considerations impose some strong limitations on the size of models that we can consider. A fundamental question here is whether the observed head-tail sequence is long and rich enough to be able to specify a complex model. Also, it might just be the case that only a single coin is being tossed. Then using the three-coin model of Figure 6.3c would be inappropriate because we would be using an underspecified system.

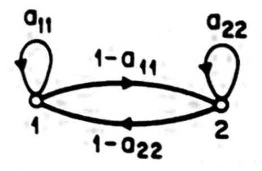
(a)



1-COIN MODEL
(OBSERVABLE MARKOV MODEL)

$O = H\ H\ T\ T\ H\ T\ H\ H\ T\ T\ H \ldots$
$S = 1\ 1\ 2\ 2\ 1\ 2\ 1\ 1\ 2\ 2\ 1 \ldots$

(b)

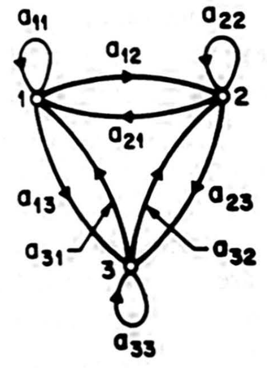


$P(H) = P_1 \quad P(H) = P_2$
$P(T) = 1-P_1 \quad P(T) = 1-P_2$

2-COINS MODEL
(HIDDEN MARKOV MODEL)

$O = H\ H\ T\ T\ H\ T\ H\ H\ T\ T\ H \ldots$
$S = 2\ 1\ 1\ 2\ 2\ 2\ 1\ 2\ 2\ 1\ 2 \ldots$

(c)



3-COINS MODEL
(HIDDEN MARKOV MODEL)

$O = H\ H\ T\ T\ H\ T\ H\ H\ T\ T\ H \ldots$
$S = 3\ 1\ 2\ 3\ 3\ 1\ 1\ 2\ 3\ 1\ 3 \ldots$

STATE

	1	2	3
$P(H)$	P_1	P_2	P_3
$P(T)$	$1-P_1$	$1-P_2$	$1-P_3$

Figure 6.3 Three possible Markov models that can account for the results of hidden coin-tossing experiments. (a) one-coin model, (b) two-coins model, (c) three-coins model.

6.3.2 The Urn-and-Ball Model

To extend the ideas of the HMM to a somewhat more complicated situation, consider the urn-and-ball system of Figure 6.4. We assume that there are N (large) glass urns in a room. Within each urn is a large quantity of colored balls. We assume there are M distinct colors of the balls. The physical process for obtaining observations is as follows. A genie is in the room, and, according to some random procedure, it chooses an initial urn. From this urn, a ball is chosen at random, and its color is recorded as the observation. The ball is

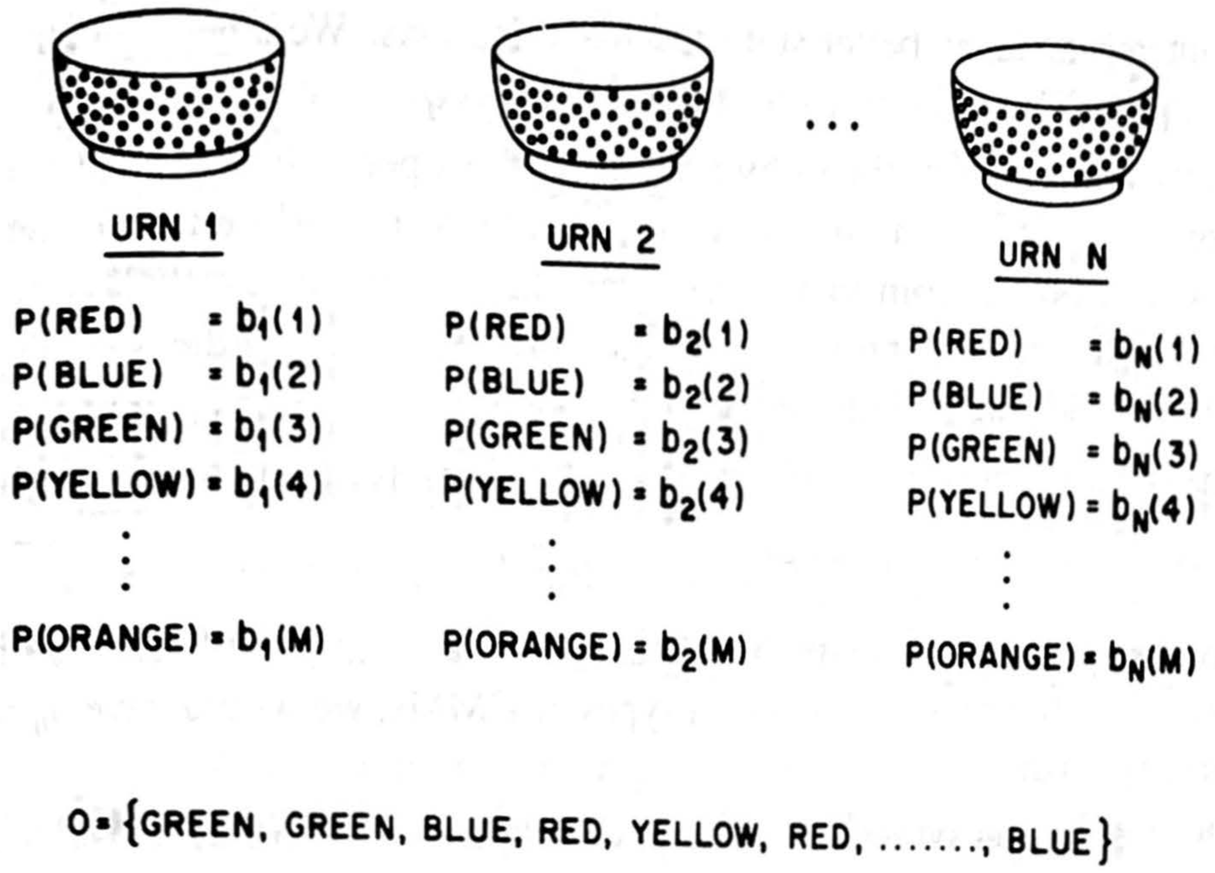


Figure 6.4 An N-state urn-and-ball model illustrating the general case of a discrete symbol HMM.

then replaced in the urn from which it was selected. A new urn is then selected according to the random selection procedure associated with the current urn, and the ball selection process is repeated. This entire process generates a finite observation sequence of colors, which we would like to model as the observable output of an HMM.

It should be obvious that the simplest HMM that corresponds to the urn-and-ball process is one in which each state corresponds to a specific urn, and for which a (ball) color probability is defined for each state. The choice of urns is dictated by the state-transition matrix of the HMM.

It should be noted that the ball colors in each urn may be the same, and the distinction among various urns is in the way the collection of colored balls is composed. Therefore, an isolated observation of a particular color ball does not immediately tell which urn it is drawn from.

6.3.3 Elements of an HMM

The above examples give us some idea of what an HMM is and how it can be applied to some simple scenarios. We now formally define the elements of an HMM.

An HMM for discrete symbol observations such as the above urn-and-ball model is characterized by the following:

1. N, the number of states in the model. Although the states are hidden, for many practical applications there is often some physical significance attached to the states or to sets of states of the model. Thus, in the coin-tossing experiments, each state corresponded to a distinct biased coin. In the urn-and-ball model, the states corresponded to the urns. Generally the states are interconnected in such a way that any state can be reached from any other state (i.e., an ergodic model); however, we will see later in this chapter that other possible interconnections of states are often

of interest and may better suit speech applications. We label the individual states as $\{1, 2, \ldots, N\}$, and denote the state at time t as q_t.

2. M, the number of distinct observation symbols per state—i.e., the discrete alphabet size. The observation symbols correspond to the physical output of the system being modeled. For the coin-toss experiments the observation symbols were simply heads or tails; for the ball-and-urn model they were the colors of the balls selected from the urns. We denote the individual symbols as $V = \{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_M\}$.
3. The state-transition probability distribution $A = \{a_{ij}\}$ where

$$a_{ij} = P[q_{t+1} = j | q_t = i], \qquad 1 \le i, j \le N. \tag{6.7}$$

For the special case in which any state can reach any other state in a single step, we have $a_{ij} > 0$ for all i, j. For other types of HMMs, we would have $a_{ij} = 0$ for one or more (i, j) pairs.

4. The observation symbol probability distribution, $B = \{b_j(k)\}$, in which

$$b_j(k) = P[\mathbf{o}_t = \mathbf{v}_k | q_t = j], \qquad 1 \le k \le M, \tag{6.8}$$

defines the symbol distribution in state $j, j = 1, 2, \ldots, N$.

5. The initial state distribution $\pi = \{\pi_i\}$ in which

$$\pi_i = P[q_1 = i], \qquad 1 \le i \le N. \tag{6.9}$$

It can be seen from the above discussion that a complete specification of an HMM requires specification of two model parameters, N and M, specification of observation symbols, and the specification of the three sets of probability measures A, B, and π. For convenience, we use the compact notation

$$\lambda = (A, B, \pi) \tag{6.10}$$

to indicate the complete parameter set of the model. This parameter set, of course, defines a probability measure for $\mathbf{O}$, i.e. $P(\mathbf{O}|\lambda)$, which we discuss in the next section. We use the terminology HMM to indicate the parameter set λ and the associated probability measure interchangeably without ambiguity.

6.3.4 HMM Generator of Observations

Given appropriate values of N, M, A, B, and π, the HMM can be used as a generator to give an observation sequence

$$\mathbf{O} = (\mathbf{o}_1 \mathbf{o}_2 \ldots \mathbf{o}_T) \tag{6.11}$$

(in which each observation $\mathbf{o}_t$ is one of the symbols from V, and T is the number of observations in the sequence) as follows:

1. Choose an initial state $q_1 = i$ according to the initial state distribution π.
2. Set $t = 1$.
3. Choose $\mathbf{o}_t = \mathbf{v}_k$ according to the symbol probability distribution in state i, i.e., $b_i(k)$.

4. Transit to a new state $q_{t+1} = j$ according to the state-transition probability distribution for state i, i.e., a_{ij}.
5. Set $t = t + 1$; return to step 3 if $t < T$; otherwise, terminate the procedure.

The following table shows the sequence of states and observations generated by the above procedure:

time, t	1	2	3	4	5	6	...	T
state	q_1	q_2	q_3	q_4	q_5	q_6	...	q_T
observation	$\mathbf{o}_1$	$\mathbf{o}_2$	$\mathbf{o}_3$	$\mathbf{o}_4$	$\mathbf{o}_5$	$\mathbf{o}_6$	...	$\mathbf{o}_T$

The above procedure can be used as both a generator of observations and as a model to simulate how a given observation sequence was generated by an appropriate HMM.

Exercise 6.2

Consider an HMM representation (parametrized by λ) of a coin-tossing experiment. Assume a three-state model (corresponding to three different coins) with probabilities

	State 1	State 2	State 3
$P(H)$	0.5	0.75	0.25
$P(T)$	0.5	0.25	0.75

and with all state-transition probabilities equal to 1/3. (Assume initial state probabilities of 1/3.)

1. You observe the sequence

$$\mathbf{O} = (HHHHTHTTTT).$$

What state sequence is most likely? What is the probability of the observation sequence and this most likely state sequence?

2. What is the probability that the observation sequence came entirely from state 1?
3. Consider the observation sequence

$$\tilde{\mathbf{O}} = (HTTHTHHTTH).$$

How would your answers to parts a and b change?

4. If the state-transition probabilities were

$$\begin{array}{lll} a_{11} = 0.9 \quad , & a_{21} = 0.45 \quad , & a_{31} = 0.45 \\ a_{12} = 0.05 \quad , & a_{22} = 0.1 \quad , & a_{32} = 0.45 \\ a_{13} = 0.05 \quad , & a_{23} = 0.45 \quad , & a_{33} = 0.1 \end{array}$$

that is, a new model λ', how would your answers to parts 1–3 change? What does this suggest about the type of sequences generated by the models?

Solution 6.2

1. Given $\mathbf{O} = (HHHHTHTTTT)$ and that all state transitions are equiprobable, the most likely state sequence is the one for which the probability of each individual observation

is maximum. Thus for each H, the most likely state is 2 and for each T the most likely state is 3. Thus the most likely state sequence is

$$\mathbf{q} = (2\,2\,2\,2\,3\,2\,3\,3\,3\,3).$$

The probability of $\mathbf{O}$ and $\mathbf{q}$ (given the model) is

$$P(\mathbf{O}, \mathbf{q}|\lambda) = (0.75)^{10} \left(\frac{1}{3}\right)^{10}.$$

2. The probability of $\mathbf{O}$ given that $\hat{\mathbf{q}}$ is

$$\hat{\mathbf{q}} = (1\,1\,1\,1\,1\,1\,1\,1\,1\,1)$$

is

$$P(\mathbf{O}, \hat{\mathbf{q}}|\lambda) = (0.50)^{10} \left(\frac{1}{3}\right)^{10}.$$

The ratio of $P(\mathbf{O}, \mathbf{q}|\lambda)$ to $P(\mathbf{O}, \hat{\mathbf{q}}|\lambda)$ is:

$$R = \frac{P(\mathbf{O}, \mathbf{q}|\lambda)}{P(\mathbf{O}, \hat{\mathbf{q}}|\lambda)} = \left(\frac{3}{2}\right)^{10} = 57.67$$

which shows, as expected, that $\mathbf{q}$ is more likely than $\hat{\mathbf{q}}$.

3. Given $\tilde{\mathbf{O}}$ which has the same number of Hs and Ts, the answers to parts 1 and 2 would remain the same, as the most likely states occur the same number of times in both cases.

4. The new probability of $\mathbf{O}$ and $\mathbf{q}$ becomes

$$P(\mathbf{O}, \mathbf{q}|\lambda') = (0.75)^{10} \left(\frac{1}{3}\right) (0.1)^6 (0.45)^3.$$

The new probability of $\mathbf{O}$ and $\hat{\mathbf{q}}$ becomes

$$P(\mathbf{O}, \hat{\mathbf{q}}|\lambda') = (0.50)^{10} \left(\frac{1}{3}\right) (0.9)^9.$$

The ratio is

$$R = \left(\frac{3}{2}\right)^{10} \left(\frac{1}{9}\right)^6 \left(\frac{1}{2}\right)^3 = 1.36 \times 10^{-5}.$$

In other words, because of the nonuniform transition probabilities, $\hat{\mathbf{q}}$ is more likely than $\mathbf{q}$. (The reader is encouraged to find the most likely state sequence in this case.) Now, the probability of $\tilde{\mathbf{O}}$ and $\mathbf{q}$ is not the same as the probability of $\mathbf{O}$ and $\mathbf{q}$. We have

$$P(\tilde{\mathbf{O}}, \mathbf{q}|\lambda') = \frac{1}{3} (0.1)^6 (0.45)^3 (0.25)^4 (0.75)^6$$
$$P(\tilde{\mathbf{O}}, \hat{\mathbf{q}}|\lambda') = (0.50)^{10} \left(\frac{1}{3}\right) (0.9)^9$$

with ratio

$$R = \left(\frac{1}{9}\right)^6 \left(\frac{1}{2}\right)^3 \left(\frac{1}{2}\right)^4 \left(\frac{3}{2}\right)^6 = 1.67 \times 10^{-7}.$$

Clearly, because $a_{11} = 0.9$, $\hat{\mathbf{q}}$ is more likely.

6.4 THE THREE BASIC PROBLEMS FOR HMMS

Given the form of HMM of the previous section, three basic problems of interest must be solved for the model to be useful in real-world applications. These problems are the following:

Problem 1

Given the observation sequence $\mathbf{O} = (\mathbf{o}_1 \mathbf{o} \ldots \mathbf{o}_T)$, and a mode¹ $\lambda = (A, B, \pi)$, how do we efficiently compute $P(\mathbf{O}|\lambda)$, the probability of the observation sequence, given the model?

Problem 2

Given the observation sequence $\mathbf{O} = (\mathbf{o}_1 \mathbf{o} \ldots \mathbf{o}_T)$, and the model λ, how do we choose a corresponding state sequence $\mathbf{q} = (q_1 q_2 \ldots q_T)$ that is optimal in some sense (i.e., best "explains" the observations)?

Problem 3

How do we adjust the model parameters $\lambda = (A, B, \pi)$ to maximize $P(\mathbf{O}|\lambda)$?

Problem 1 is the evaluation problem; namely, given a model and a sequence of observations, how do we compute the probability that the observed sequence was produced by the model? We can also view the problem as one of scoring how well a given model matches a given observation sequence. The latter viewpoint is extremely useful. For example, if we consider the case in which we are trying to choose among several competing models, the solution to Problem 1 allows us to choose the model that best matches the observations.

Problem 2 is the one in which we attempt to uncover the hidden part o᷉ the model—that is, to find the "correct" state sequence. It should be clear that for all but the case of degenerate models, there is no "correct" state sequence to be found. Hence for practical situations, we usually use an optimality criterion to solve this problem as best as possible. As we will see, several reasonable optimal ıy criteria can be imposed, and hence the choice of criterion is a strong function of the intended use for the uncovered state sequence. Typical uses might be to learn about the structure of the model, to find optimal state sequences for continuous speech recognition, or to get average statistics of individual states, etc.

Problem 3 is the one in which we attempt to optimize the model parameters to best describe how a given observation sequence comes about. The observation sequence used to adjust the model parameters is called a training sequence because it is used to "train" the HMM. The training problem is the crucial one for most applications of HMMs, because it allows us to optimally adapt model parameters to observed training data—i.e., to create best models for real phenomena.

To fix ideas, consider the following simple isolated-word speech recognizer. For each word of a W word vocabulary, we want to design a separate N-state HMM. We represent the speech signal of a given word as a time sequence of coded spectral vectors. We assume that the coding is done using a spectral codebook with M unique spectral vectors; hence each observation is the index of the spectral vector closest (in some spectral distortion sense) to the original speech signal. Thus, for each vocabulary word, we have a training sequence consisting of a number of repetitions of sequences of codebook indices of the word (by one or more talkers). The first task is to build individual word models. This task is done by using the solution to Problem 3 to optimally estimate model parameters for each word model. To develop an understanding of the physical meaning of the model states, we use the solution to Problem 2 to segment each of the word training sequences into states, and then study the properties of the spectral vectors that lead to the observations occurring in each state. The goal here is to make refinements of the model (e.g., more states, different codebook size) to improve its capability of modeling the spoken word sequences. Finally, once the set of W HMMs has been designed and optimized, recognition of an unknown word is performed using the solution to Problem 1 to score each word model based upon the given test observation sequence, and select the word whose model score is highest (i.e., the highest likelihood).

In the next sections we present formal mathematical solutions to each fundamental problem for HMMs. We shall see that the three problems are tightly linked together under the probabilistic framework.

6.4.1 Solution to Problem 1—Probability Evaluation

We wish to calculate the probability of the observation sequence, $\mathbf{O} = (\mathbf{o}_1\, \mathbf{o} \ldots \mathbf{o}_T)$, given the model λ, i.e., $P(\mathbf{O}|\lambda)$. The most straightforward way of doing this is through enumerating every possible state sequence of length T (the number of observations). There are N^T such state sequences. Consider one such fixed-state sequence

$$\mathbf{q} = (q_1 q_2 \ldots q_T) \tag{6.12}$$

where q_1 is the initial state. The probability of the observation sequence $\mathbf{O}$ given the state sequence of Eq. (6.12) is

$$P(\mathbf{O}|\mathbf{q}, \lambda) = \prod_{t=1}^{T} P(\mathbf{o}_t|q_t, \lambda) \tag{6.13a}$$

where we have assumed statistical independence of observations. Thus we get

$$P(\mathbf{O}|\mathbf{q}, \lambda) = b_{q_1}(\mathbf{o}_1) \cdot b_{q_2}(\mathbf{o}_2) \ldots b_{q_T}(\mathbf{o}_T). \tag{6.13b}$$

The probability of such a state sequence $\mathbf{q}$ can be written as

$$P(\mathbf{q}|\lambda) = \pi_{q_1} a_{q_1 q_2} a_{q_2 q_3} \ldots a_{q_{T-1} q_T}. \tag{6.14}$$

The joint probability of $\mathbf{O}$ and $\mathbf{q}$, i.e., the probability that $\mathbf{O}$ and $\mathbf{q}$ occur simultaneously, is simply the product of the above two terms, i.e.,

$$P(\mathbf{O}, \mathbf{q}|\lambda) = P(\mathbf{O}|\mathbf{q}, \lambda) P(\mathbf{q}|\lambda). \tag{6.15}$$

The probability of $\mathbf{O}$ (given the model) is obtained by summing this joint probability over all possible state sequences $\mathbf{q}$, giving

$$P(\mathbf{O}|\lambda) = \sum_{\text{all } \mathbf{q}} P(\mathbf{O}|\mathbf{q}, \lambda) P(\mathbf{q}|\lambda) \tag{6.16}$$

$$= \sum_{q_1, q_2, \ldots, q_T} \pi_{q_1} b_{q_1}(\mathbf{o}_1) a_{q_1 q_2} b_{q_2}(\mathbf{o}_2) \ldots a_{q_{T-1} q_T} b_{q_T}(\mathbf{o}_T). \tag{6.17}$$

The interpretation of the computation in the above equation is the following. Initially (at time $t = 1$) we are in state q_1 with probability π_{q_1}, and generate the symbol $\mathbf{o}_1$ (in this state) with probability $b_{q_1}(\mathbf{o}_1)$. The clock changes from time t to $t+1$ (time = 2) and we make a transition to state q_2 from state q_1 with probability $a_{q_1 q_2}$, and generate symbol $\mathbf{o}_2$ with probability $b_{q_2}(\mathbf{o}_2)$. This process continues in this manner until we make the last transition (at time T) from state q_{T-1} to state q_T with probability $a_{q_{T-1} q_T}$ and generate symbol $\mathbf{o}_T$ with probability $b_{q_T}(\mathbf{o}_T)$.

A little thought should convince the reader that the calculation of $P(\mathbf{O}|\lambda)$, according to its direct definition (Eq. (6.17)) involves on the order of $2T \cdot N^T$ calculations, since at every $t = 1, 2, \ldots, T$, there are N possible states that can be reached (i.e., there are N^T possible state sequences), and for each such state sequence about $2T$ calculations are required for each term in the sum of Eq. (6.17). (To be precise, we need $(2T-1)N^T$ multiplications, and $N^T - 1$ additions.) This calculation is computationally infeasible, even for small values of N and T; e.g., for $N = 5$ (states), $T = 100$ (observations), there are on the order of $2 \cdot 100 \cdot 5^{100} \approx 10^{72}$ computations! Clearly a more efficient procedure is required to solve problem 1. Fortunately such a procedure (called the forward procedure) exists.

6.4.1.1 The Forward Procedure

Consider the forward variable $\alpha_t(i)$ defined as

$$\alpha_t(i) = P(\mathbf{o}_1 \mathbf{o}_2 \ldots \mathbf{o}_t, q_t = i|\lambda) \tag{6.18}$$

that is, the probability of the partial observation sequence, $\mathbf{o}_1 \mathbf{o}_2 \ldots \mathbf{o}_t$, (until time t) and state i at time t, given the model λ. We can solve for $\alpha_t(i)$ inductively, as follows:

1. Initialization

$$\alpha_1(i) = \pi_i b_i(\mathbf{o}_1), \qquad 1 \le i \le N. \tag{6.19}$$

2. Induction

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij}\right] b_j(\mathbf{o}_{t+1}), \qquad \begin{array}{l} 1 \le t \le T-1 \\ 1 \le j \le N \end{array}. \tag{6.20}$$

3. Termination

$$P(\mathbf{O}|\lambda) = \sum_{i=1}^{N} \alpha_T(i). \tag{6.21}$$

Step 1 initializes the forward probabilities as the joint probability of state i and initial observation $\mathbf{o}_1$. The induction step, which is the heart of the forward calculation, is illustrated in Figure 6.5(a). This figure shows how state j can be reached at time $t+1$ from the N possible states, i, $1 \le i \le N$, at time t. Since $\alpha_t(i)$ is the probability of the joint event that $\mathbf{o}_1 \mathbf{o}_2 \ldots \mathbf{o}_t$ are observed, and the state at time t is i, the product $\alpha_t(i) a_{ij}$ is

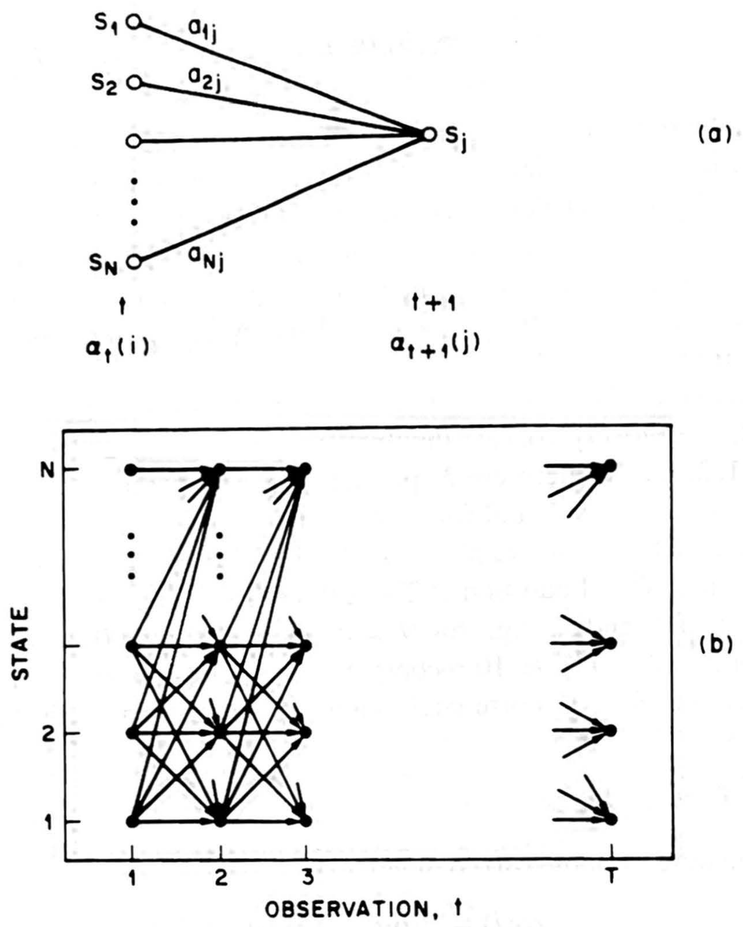


Figure 6.5 (a) Illustration of the sequence of operations required for the computation of the forward variable $\alpha_{t+1}(j)$. (b) Implementation of the computation of $\alpha_t(i)$ in terms of a lattice of observations t, and states i.

then the probability of the joint event that $\mathbf{o}_1\,\mathbf{o}_2\ldots\mathbf{o}_t$ are observed, and state j is reached at time $t+1$ via state i at time t. Summing this product over all the N possible states, i, $1 \leq i \leq N$ at time t results in the probability of j at time $t+1$ with all the accompanying previous partial observations. Once this is done and j is known, it is easy to see that $\alpha_{t+1}(j)$ is obtained by accounting for observation $\mathbf{o}_{t+1}$ in state j, i.e., by multiplying the summed quantity by the probability $b_j(\mathbf{o}_{t+1})$. The computation of Eq. (6.20) is performed for all states j, $1 \leq j \leq N$, for a given t; the computation is then iterated for $t = 1, 2, \ldots, T-1$. Finally, step 3 gives the desired calculation of $P(\mathbf{O}|\lambda)$ as the sum of the terminal forward variables $\alpha_T(i)$. This is the case since, by definition,

$$\alpha_T(i) = P(\mathbf{o}_1\mathbf{o}_2\ldots\mathbf{o}_T, q_T = i|\lambda) \tag{6.22}$$

and hence $P(\mathbf{O}|\lambda)$ is just the sum of the $\alpha_T(i)$'s.

If we examine the computation involved in the calculation of $\alpha_t(j)$, $1 \leq t \leq T$, $1 \leq j \leq N$, we see that it requires on the order of N^2T calculations, rather than $2TN^T$ as required by the direct calculation. (Again, to be precise, we need $N(N+1)(T-1)+N$

multiplications and $N(N-1)(T-1)$ additions.) For $N = 5$, $T = 100$, we need about 3000 computations for the forward method, versus 10^{72} computations for the direct calculation, a savings of about 69 orders of magnitude.

The forward probability calculation is, in effect, based upon the lattice (or trellis) structure shown in Figure 6.5(b). The key is that, because there are only N states (nodes at each time slot in the lattice), all the possible state sequences will remerge into these N nodes, no matter how long the observation sequence. At time $t = 1$ (the first time slot in the lattice), we need to calculate values of $\alpha_1(i)$, $1 \le i \le N$. At times $t = 2, 3, \ldots, T$, we need only calculate values of $\alpha_t(j)$, $1 \le j \le N$, where each calculation involves only the N previous values of $\alpha_{t-1}(i)$ because each of the N grid points can be reached from only the N grid points at the previous time slot.

6.4.1.2 The Backward Procedure

In a similar manner, we can consider a backward variable $\beta_t(i)$ defined as

$$\beta_t(i) = P(\mathbf{o}_{t+1}\mathbf{o}_{t+2}\ldots\mathbf{o}_T | q_t = i, \lambda) \tag{6.23}$$

that is, the probability of the partial observation sequence from $t+1$ to the end, given state i at time t and the model λ. Again we can solve for $\beta_t(i)$ inductively, as follows:

1. Initialization

$$\beta_T(i) = 1, \qquad 1 \le i \le N. \tag{6.24}$$

2. Induction

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(\mathbf{o}_{t+1}) \beta_{t+1}(j),$$
$$t = T-1, T-2, \ldots, 1, \qquad 1 \le i \le N. \tag{6.25}$$

The initialization step 1 *arbitrarily* defines $\beta_T(i)$ to be 1 for all i. Step 2, which is illustrated in Figure 6.6, shows that in order to have been in state i at time t, and to account for the observation sequence from time $t+1$ on, you have to consider all possible states j at time $t+1$, accounting for the transition from i to j (the a_{ij} term), as well as the observation $\mathbf{o}_{t+1}$ in state j (the $b_j(\mathbf{o}_{t+1})$ term), and then account for the remaining partial observation sequence from state j (the $\beta_{t+1}(j)$ term). We will see later how the backward as well as the forward calculations are used to help solve fundamental Problems 2 and 3 of HMMs.

Again, the computation of $\beta_t(i)$, $1 \le t \le T$, $1 \le i \le N$, requires on the order of N^2T calculations, and can be computed in a lattice structure similar to that of Figure 6.5(b).

6.4.2 Solution to Problem 2—"Optimal" State Sequence

Unlike Problem 1, for which an exact solution can be given, there are several possible ways of solving Problem 2—namely, finding the "optimal" state sequence associated with the given observation sequence. The difficulty lies with the definition of the optimal state sequence—that is, there are several possible optimality criteria. For example, one possible

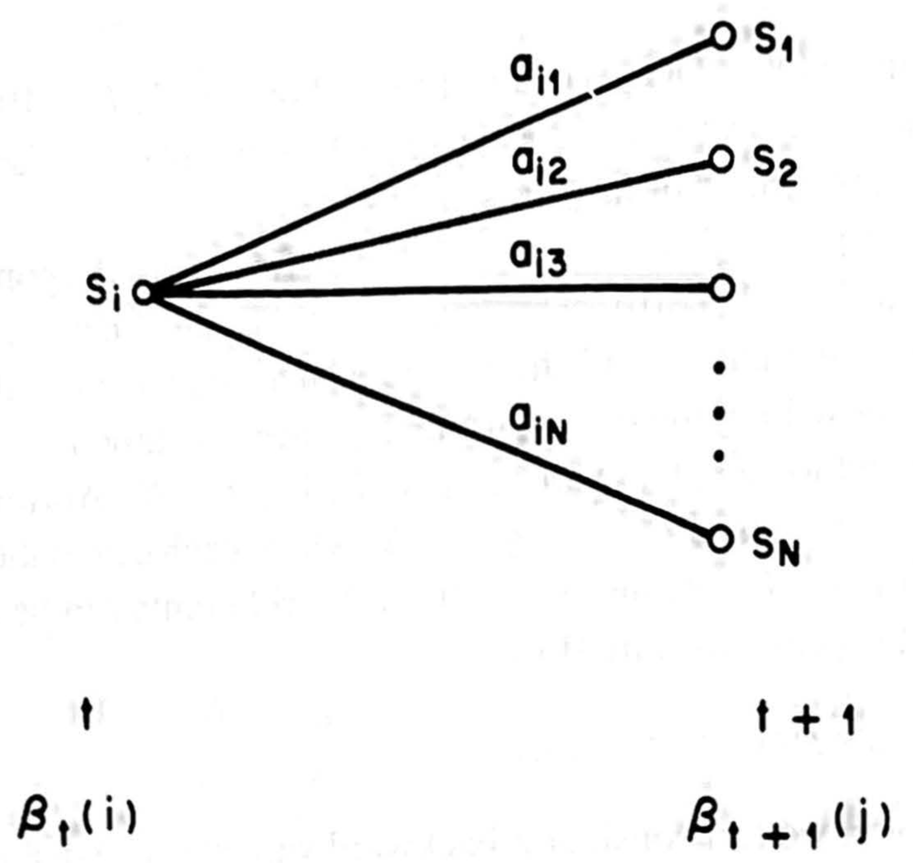


Figure 6.6 Sequence of operations required for the computation of the backward variable $\beta_t(i)$.

optimality criterion is to choose the states q_t that are *individually* most likely at each time t. This optimality criterion maximizes the expected number of correct individual states. To implement this solution to Problem 2, we can define the a posteriori probability variable

$$\gamma_t(i) = P(q_t = i|\mathbf{O}, \lambda) \tag{6.26}$$

that is, the probability of being in state i at time t, given the observation sequence $\mathbf{O}$, and the model λ. We can express $\gamma_t(i)$ in several forms, including

$$\begin{aligned}\gamma_t(i) &= P(q_t = i \mid \mathbf{O}, \lambda)\\ &= \frac{P(\mathbf{O}, q_t = i \mid \lambda)}{P(\mathbf{O} \mid \lambda)}\\ &= \frac{P(\mathbf{O}, q_t = i \mid \lambda)}{\sum_{i=1}^{N} P(\mathbf{O}, q_t = i \mid \lambda)}.\end{aligned} \tag{6.27}$$

Since $P(\mathbf{O}, q_t = i \mid \lambda)$ is equal to $\alpha_t(i)\beta_t(i)$, we can write $\gamma_t(i)$ as

$$\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{\sum_{i=1}^{N} \alpha_t(i)\beta_t(i)} \tag{6.28}$$

where we see that $\alpha_t(i)$ accounts for the partial observation sequence $\mathbf{o}_1\, \mathbf{o}_2 \ldots \mathbf{o}_t$ and state i at t, while $\beta_t(i)$ accounts for the remainder of the observation sequence $\mathbf{o}_{t+1}\mathbf{o}_{t+2} \cdots \mathbf{o}_T$, given state $q_t = i$ at t.

Using $\gamma_t(i)$, we can solve for the individually most likely state q_t^* at time t, as

$$q_t^* = \arg\min_{1 \le i \le N} [\gamma_t(i)], \qquad 1 \le t \le T. \tag{6.29}$$

Although Eq. (6.29) maximizes the expected number of correct states (by choosing the most likely state for each t), there could be some problems with the resulting state sequence. For example, when the HMM has state transitions which have zero probability ($a_{ij} = 0$ for some i and j), the "optimal" state sequence may, in fact, not even be a valid state sequence. This is because the solution of Eq. (6.29) simply determines the most likely state at every instant, without regard to the probability of occurrence of sequences of states.

One possible solution to the above problem is to modify the optimality criterion. For example, one could solve for the state sequence that maximizes the expected number of correct pairs of states (q_t, q_{t+1}), or triples of states (q_t, q_{t+1}, q_{t+2}), etc. Although these criteria might be reasonable for some applications, the most widely used criterion is to find the *single* best state sequence (path)—that is, to maximize $P(\mathbf{q}|\mathbf{O}, \lambda)$, which is equivalent to maximizing $P(\mathbf{q}, \mathbf{O}|\lambda)$. A formal technique for finding this single best state sequence exists, based on dynamic programming methods, and is called the Viterbi algorithm [15, 16].

6.4.2.1 The Viterbi Algorithm

To find the single best state sequence, $\mathbf{q} = (q_1 \, q_2 \ldots q_T)$, for the given observation sequence $\mathbf{O} = (\mathbf{o}_1 \, \mathbf{o}_2 \ldots \mathbf{o}_T)$, we need to define the quantity

$$\delta_t(i) = \max_{q_1, q_2, \ldots, q_{t-1}} P[q_1 q_2 \ldots q_{t-1}, \; q_t = i, \; \mathbf{o}_1 \mathbf{o}_2 \ldots \mathbf{o}_t|\lambda] \tag{6.30}$$

that is, $\delta_t(i)$ is the best score (highest probability) along a single path, at time t, which accounts for the first t observations and ends in state i. By induction we have

$$\delta_{t+1}(j) = [\max_i \delta_t(i) \, a_{ij}] \cdot b_j(\mathbf{o}_{t+1}). \tag{6.31}$$

To actually retrieve the state sequence, we need to keep track of the argument that maximized Eq. (6.31), for each t and j. We do this via the array $\psi_t(j)$. The complete procedure for finding the best state sequence can now be stated as follows:

1. Initialization

$$\delta_1(i) = \pi_i b_i(\mathbf{o}_1), \qquad 1 \le i \le N \tag{6.32a}$$

$$\psi_1(i) = 0. \tag{6.32b}$$

2. Recursion

$$\delta_t(j) = \max_{1 \le i \le N} [\delta_{t-1}(i) \, a_{ij}] b_j(\mathbf{o}_t), \qquad \begin{matrix} 2 \le t \le T \\ 1 \le j \le N \end{matrix} \tag{6.33a}$$

$$\psi_t(j) = \arg\max_{1 \le i \le N} [\delta_{t-1}(i) \, a_{ij}], \qquad \begin{matrix} 2 \le t \le T \\ 1 \le j \le N. \end{matrix} \tag{6.33b}$$

3. Termination

$$P^* = \max_{1 \le i \le N} [\delta_T(i)] \tag{6.34a}$$

$$q_T^* = \arg \max_{1 \le i \le N} [\delta_T(i)]. \tag{6.34b}$$

4. Path (state sequence) backtracking

$$q_t^* = \psi_{t+1}(q_{t+1}^*), \qquad t = T-1, T-2, \ldots, 1. \tag{6.35}$$

It should be noted that the Viterbi algorithm is similar (except for the backtracking step) in implementation to the forward calculation of Eqs. (6.19)–(6.21). The major difference is the maximization in Eq. (6.33a) over previous states, which is used in place of the summing procedure in Eq. (6.20). It also should be clear that a lattice (or trellis) structure efficiently implements the computation of the Viterbi procedure.

6.4.2.2 Alternative Viterbi Implementation

By taking logarithms of the model parameters, the Viterbi algorithm of the preceding section can be implemented without the need for any multiplications. Thus:

0. Preprocessing

$$\begin{aligned} \tilde{\pi}_i &= \log(\pi_i), & 1 \le i \le N \\ \tilde{b}_i(\mathbf{o}_t) &= \log[b_i(\mathbf{o}_t)], & 1 \le i \le N,\ 1 \le t \le T \\ \tilde{a}_{ij} &= \log(a_{ij}), & 1 \le i, j \le N \end{aligned}$$

1. Initialization

$$\begin{aligned} \tilde{\delta}_1(i) &= \log(\delta_1(i)) = \tilde{\pi}_i + \tilde{b}_i(\mathbf{o}_1), & 1 \le i \le N \\ \psi_1(i) &= 0, & 1 \le i \le N \end{aligned}$$

2. Recursion

$$\begin{aligned} \tilde{\delta}_t(j) &= \log(\delta_t(j)) = \max_{1 \le i \le N} \left[\tilde{\delta}_{t-1}(i) + \tilde{a}_{ij}\right] + \tilde{b}_j(\mathbf{o}_t) \\ \psi_t(j) &= \arg \max_{1 \le i \le N} [\tilde{\delta}_{t-1}(i) + \tilde{a}_{ij}], \qquad 2 \le t \le T,\ 1 \le j \le N \end{aligned}$$

3. Termination

$$\tilde{P}^* = \max_{1 \le i \le N} [\tilde{\delta}_T(i)]$$

$$q_T^* = \arg \max_{1 \le i \le N} [\tilde{\delta}_T(i)]$$

4. Backtracking

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

The calculation required for this alternative implementation is on the order of N^2T additions (plus the calculation for preprocessing). Because the preprocessing needs to be performed once and saved, its cost is negligible for most systems.

Exercise 6.3

Given the model of the coin-toss experiment used in Exercise 6.2 (i.e., three different coins) with probabilities

	State 1	State 2	State 3
$P(H)$	0.5	0.75	0.25
$P(T)$	0.5	0.25	0.75

and with all state transition probabilities equal to 1/3, and with initial probabilities equal to 1/3, for the observation sequence

$$\mathbf{O} = (HHHHTHTTTT)$$

find the most likely path with the Viterbi algorithm.

Solution 6.3

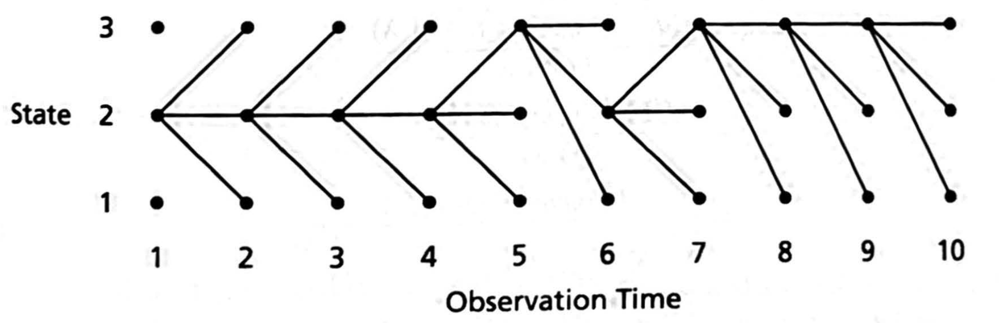
Since all a_{ij} terms are equal to 1/3, we can omit these terms (as well as the initial state probability term), giving

$$\delta_1(1) = 0.5, \quad \delta_1(2) = 0.75, \quad \delta_1(3) = 0.25.$$

The recursion for $\delta_t(j)$ gives $(2 \leq t \leq 10)$

$$\begin{aligned}
&\delta_2(1) = (0.75)(0.5), && \delta_2(2) = (0.75)^2, && \delta_2(3) = (0.75)(0.25)\\
&\delta_3(1) = (0.75)^2(0.5), && \delta_3(2) = (0.75)^3, && \delta_3(3) = (0.75)^2(0.25)\\
&\delta_4(1) = (0.75)^3(0.5), && \delta_4(2) = (0.75)^4, && \delta_4(3) = (0.75)^3(0.25)\\
&\delta_5(1) = (0.75)^4(0.5), && \delta_5(2) = (0.75)^4(0.25), && \delta_5(3) = (0.75)^5\\
&\delta_6(1) = (0.75)^5(0.5), && \delta_6(2) = (0.75)^6, && \delta_6(3) = (0.75)^5(0.25)\\
&\delta_7(1) = (0.75)^6(0.5), && \delta_7(2) = (0.75)^6(0.25), && \delta_7(3) = (0.75)^7\\
&\delta_8(1) = (0.75)^7(0.5), && \delta_8(2) = (0.75)^7(0.25), && \delta_8(3) = (0.75)^8\\
&\delta_9(1) = (0.75)^8(0.5), && \delta_9(2) = (0.75)^8(0.25), && \delta_9(3) = (0.75)^9\\
&\delta_{10}(1) = (0.75)^9(0.5), && \delta_{10}(2) = (0.75)^9(0.25), && \delta_{10}(3) = (0.75)^{10}
\end{aligned}$$

This leads to a diagram (trellis) of the form:



Hence, the most likely state sequence is $\{2, 2, 2, 2, 3, 2, 3, 3, 3, 3\}$.

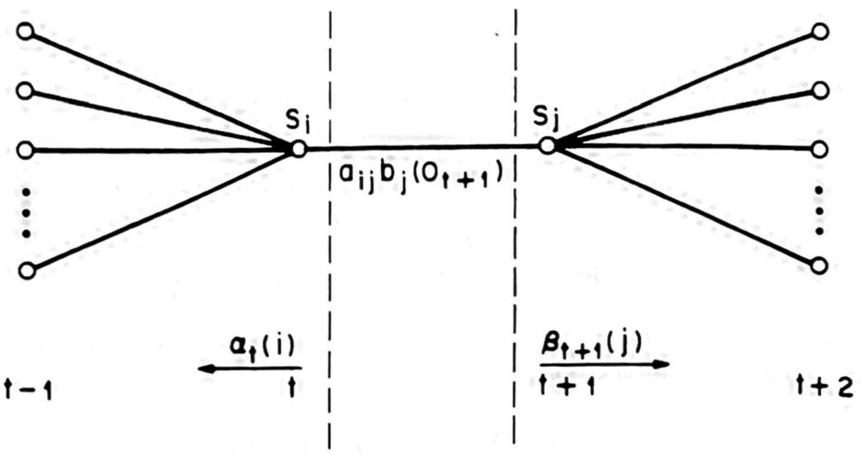


Figure 6.7 Illustration of the sequence of operations required for the computation of the joint event that the system is in state i at time t and state j at time $t+1$.

6.4.3 Solution to Problem 3—Parameter Estimation

The third, and by far the most difficult, problem of HMMs is to determine a method to adjust the model parameters (A, B, π) to satisfy a certain optimization criterion. There is no known way to analytically solve for the model parameter set that maximizes the probability of the observation sequence in a closed form. We can, however, choose $\lambda = (A, B, \pi)$ such that its likelihood, $P(\mathbf{O}|\lambda)$, is locally maximized using an iterative procedure such as the Baum-Welch method (also known as the EM (expectation-maximization) method [17]), or using gradient techniques [18]. In this section we discuss one iterative procedure, based primarily on the classic work of Baum and his colleagues, for choosing the maximum likelihood (ML) model parameters.

To describe the procedure for reestimation (iterative update and improvement) of HMM parameters, we first define $\xi_t(i,j)$, the probability of being in state i at time t, and state j at time $t+1$, given the model and the observation sequence, i.e.

$$\xi_t(i,j) = P(q_t = i,\ q_{t+1} = j|\mathbf{O}, \lambda). \tag{6.36}$$

The paths that satisfy the conditions required by Eq. (6.36) are illustrated in Figure 6.7. From the definitions of the forward and backward variables, we can write $\xi_t(i,j)$ in the form

$$\begin{aligned}\xi_t(i,j) &= \frac{P(q_t = i,\ q_{t+1} = j,\ \mathbf{O} \mid \lambda)}{P(\mathbf{O} \mid \lambda)} \\ &= \frac{\alpha_t(i)\, a_{ij} b_j(\mathbf{o}_{t+1})\beta_{t+1}(j)}{P(\mathbf{O}|\lambda)} \\ &= \frac{\alpha_t(i)\, a_{ij} b_j(\mathbf{o}_{t+1})\beta_{t+1}(j)}{\displaystyle\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_t(i)\, a_{ij} b_j(\mathbf{o}_{t+1})\beta_{t+1}(j)}.\end{aligned} \tag{6.37}$$

We have previously defined $\gamma_t(i)$ as the probability of being in state i at time t, given the entire observation sequence and the model; hence, we can relate $\gamma_t(i)$ to $\xi_t(i,j)$ by summing over j, giving

$$\gamma_t(i) = \sum_{j=1}^{N} \xi_t(i,j). \tag{6.38}$$

If we sum $\gamma_t(i)$ over the time index t, we get a quantity that can be interpreted as the expected (over time) number of times that state i is visited, or equivalently, the expected number of transitions made from state i (if we exclude the time slot $t = T$ from the summation). Similarly, summation of $\xi_t(i,j)$ over t (from $t = 1$ to $t = T - 1$) can be interpreted as the expected number of transitions from state i to state j. That is,

$$\sum_{t=1}^{T-1} \gamma_t(i) = \text{expected number of transitions from state } i \text{ in } \mathbf{O} \tag{6.39a}$$

$$\sum_{t=1}^{T-1} \xi_t(i,j) = \text{expected number of transitions from state } i \text{ to state } j \text{ in } \mathbf{O}. \tag{6.39b}$$

Using the above formulas (and the concept of counting event occurrences), we can give a method for reestimation of the parameters of an HMM. A set of reasonable reestimation formulas for π, A, and B is

$$\begin{aligned}\bar{\pi}_j = {} & \text{expected frequency (number of times) in state } i \\ & \text{at time } (t = 1) = \gamma_1(i)\end{aligned} \tag{6.40a}$$

$$\bar{a}_{ij} = \frac{\text{expected number of transitions from state } i \text{ to state } j}{\text{expected number of transitions from state } i}$$

$$= \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)} \tag{6.40b}$$

$$\bar{b}_j(k) = \frac{\text{expected number of times in state } j \text{ and observing symbol } \mathbf{v}_k}{\text{expected number of times in state } j}$$

$$= \frac{\sum_{\substack{t=1 \\ \text{s.t. } \mathbf{o}_t = \mathbf{v}_k}}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)}. \tag{6.40c}$$

If we define the current model as $\lambda = (A, B, \pi)$ and use that to compute the right-hand sides of Eqs. (6.40a)–(6.40c), and we define the reestimated model as $\bar{\lambda} = (\bar{A}, \bar{B}, \bar{\pi})$, as determined from the left-hand sides of Eqs. (6.40a)–(6.40c), then it has been proven by Baum and his colleagues 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 model λ in the sense that $P(\mathbf{O}|\bar{\lambda}) > P(\mathbf{O}|\lambda)$; that is, we have found a new model $\bar{\lambda}$ from which the observation sequence is more likely to have been produced.

Based on the above procedure, if we iteratively use $\bar{\lambda}$ in place of λ and repeat the reestimation calculation, we then can improve the probability of $\mathbf{O}$ being observed from the model until some limiting point is reached. The final result of this reestimation procedure is an ML estimate of the HMM. It should be pointed out that the forward-backward algorithm leads to local maxima only, and that in most problems of interest, the likelihood function is very complex and has many local maxima.

The reestimation formulas of Eqs. (6.40a)–(6.40c) can be derived directly by maximizing (using standard constrained optimization techniques) Baum's auxiliary function

$$Q(\lambda', \lambda) = \sum_{\mathbf{q}} P(\mathbf{O}, \mathbf{q}|\lambda') \log P(\mathbf{O}, \mathbf{q}|\lambda) \tag{6.41}$$

over λ. Because

$$Q(\lambda', \lambda) \geq Q(\lambda', \lambda') \Rightarrow P(\mathbf{O}|\lambda) \geq P(\mathbf{O}|\lambda') \tag{6.42}$$

we can maximize the function $Q(\lambda', \lambda)$ over λ to improve λ' in the sense of increasing the likelihood $P(\mathbf{O}|\lambda)$. Eventually the likelihood function converges to a critical point if we iterate the procedure.

6.4.3.1 Derivation of Reestimation Formulas from the Q Function

The auxiliary function $Q(\lambda', \lambda)$ was defined in Eq. (6.41) as

$$Q(\lambda', \lambda) = \sum_{\mathbf{q}} P(\mathbf{O}, \mathbf{q}|\lambda') \log P(\mathbf{O}, \mathbf{q}|\lambda)$$

in which we can express P and $\log P$ (in terms of the HMM parameters) as

$$P(\mathbf{O}, \mathbf{q}|\lambda) = \pi_{q_0} \prod_{t=1}^{T} a_{q_{t-1}q_t} b_{q_t}(\mathbf{o}_t)$$

$$\log P(\mathbf{O}, \mathbf{q}|\lambda) = \log \pi_{q_0} + \sum_{t=1}^{T} \log a_{q_{t-1}q_t} + \sum_{t=1}^{T} \log b_{q_t}(\mathbf{o}_t)$$

(There is a slight difference between the above equations and the expression of Eq. (6.17) in which the first observation is associated with the initial state before any state transition is made. This difference is inconsequential and should not impede our understanding of the method.) Thus we can write $Q(\lambda', \lambda)$ as

$$Q(\lambda', \lambda) = Q_\pi(\lambda', \boldsymbol{\pi}) + \sum_{i=1}^{N} Q_{a_i}(\lambda', \mathbf{a}_i) + \sum_{i=1}^{N} Q_{b_i}(\lambda', \mathbf{b}_i)$$

where

$$\boldsymbol{\pi} = [\pi_1, \pi_2, \ldots, \pi_N],$$

$\mathbf{a}_i = [a_{i1}, a_{i2}, \ldots, a_{iN}]$, $\mathbf{b}_i$ is the parameter vector that defines $b_i(\cdot)$

and

$$Q_\pi(\lambda', \boldsymbol{\pi}) = \sum_{i=1}^{N} P(\mathbf{O}, q_0 = i|\lambda') \log \pi_i$$

$$Q_{a_i}(\lambda', \mathbf{a}_i) = \sum_{j=1}^{N} \sum_{t=1}^{T} P(\mathbf{O}, q_{t-1} = i, q_t = j|\lambda') \log a_{ij}$$

$$Q_{b_i}(\lambda', \mathbf{b}_i) = \sum_{t=1}^{T} P(\mathbf{O}, q_t = i|\lambda') \log b_i(\mathbf{o}_t)$$

Because of the separability of $Q(\lambda', \lambda)$ into three independent terms, we can maximize $Q(\lambda', \lambda)$ over λ by maximizing the individual terms separately, subject to the stochastic constraints

$$\sum_{j=1}^{N} \pi_j = 1$$

$$\sum_{j=1}^{N} a_{ij} = 1, \qquad \forall j$$

and (for discrete densities where $b_i(\mathbf{o}_t = \mathbf{v}_k) = b_i(k)$)

$$\sum_{k=1}^{K} b_i(k) = 1, \qquad \forall i.$$

Because the individual auxiliary functions all have the form

$$\sum_{j=1}^{N} w_j \log y_j$$

which, as a function of $\{y_j\}_{j=1}^{N}$, subject to the constraints $\sum_{j=1}^{N} y_j = 1$, $y_j \geq 0$, attains a global maximum at the single point

$$y_j = \frac{w_j}{\sum_{i=1}^{N} w_i}, \qquad j = 1, 2, \ldots, N$$

then the maximization leads to the model reestimate $\overline{\lambda} = [\overline{\pi}, \overline{A}, \overline{B}]$ where

$$\overline{\pi}_i = \frac{P(\mathbf{O}, q_0 = i|\lambda)}{P(\mathbf{O}|\lambda)}$$

$$\bar{a}_{ij} = \frac{\sum_{t=1}^{T} P(\mathbf{O}, q_{t-1} = i, q_t = j|\lambda)}{\sum_{t=1}^{T} P(\mathbf{O}, q_{t-1} = i|\lambda)}$$

$$\bar{b}_i(k) = \frac{\sum_{t=1}^{T} P(\mathbf{O}, q_t = i|\lambda)\ \delta(\mathbf{o}_t, \mathbf{v}_k)}{\sum_{t=1}^{T} P(\mathbf{O},\ q_t = i|\lambda)}$$

where we have defined

$$\begin{aligned} \delta(\mathbf{o}_t, \mathbf{v}_k) &= 1 \quad \text{if } \mathbf{o}_t = \mathbf{v}_k \\ &= 0 \quad \text{otherwise.} \end{aligned}$$

Using the definitions of the forward variable, $\alpha_t(i) = P(\mathbf{o}_1, \mathbf{o}_2, \ldots, \mathbf{o}_t, q_t = i|\lambda)$ and the backward variable, $\beta_t(i) = P(\mathbf{o}_{t+1}, \ldots, \mathbf{o}_T|q_t = i, \lambda)$, the reestimation transformations can be easily calculated as

$$\begin{aligned} &P(\mathbf{O}, q_t = i|\lambda) = \alpha_t(i)\beta_t(i) \\ &P(\mathbf{O}|\lambda) = \sum_{i=1}^{N} \alpha_t(i)\beta_t(i) = \sum_{i=1}^{N} \alpha_T(i) \\ &P(\mathbf{O}, q_{t-1} = i,\ q_t = j|\lambda) = \alpha_{t-1}(i)\, a_{ij} b_j(\mathbf{o}_t)\beta_t(j) \end{aligned}$$

giving

$$\bar{\pi}_i = \frac{\alpha_0(i)\beta_0(i)}{\sum_{j=1}^{N} \alpha_T(j)} = \gamma_0(i)$$

$$\bar{a}_{ij} = \frac{\sum_{t=1}^{T} \alpha_{t-1}(i)\, a_{ij} b_j(\mathbf{o}_t)\beta_t(j)}{\sum_{t=1}^{T} \alpha_{t-1}(i)\beta_{t-1}(i)} = \frac{\sum_{t=1}^{T} \xi_{t-1}(i,j)}{\sum_{t=1}^{T} \gamma_{t-1}(i)}$$

$$\bar{b}_i(k) = \frac{\sum_{t=1}^{T} \alpha_t(i)\beta_t(i)\delta(\mathbf{o}_t, \mathbf{v}_k)}{\sum_{t=1}^{T} \alpha_t(i)\beta_t(i)} = \frac{\sum_{\substack{t=1 \\ \text{s.t. } \mathbf{o}_t = \mathbf{v}_k}}^{T} \gamma_t(i)}{\sum_{t=1}^{T} \gamma_t(i)}$$

which are the formulas given in Eqs. (6.40a)–(6.40c).

6.4.4 Notes on the Reestimation Procedure

The reestimation formulas can be readily interpreted as an implementation of the EM algorithm of statistics [17] in which the E (expectation) step is the calculation of the auxiliary function $Q(\lambda', \lambda)$, (which is the expectation of log $P(\mathbf{O}, \mathbf{q}|\lambda)$), and the M (maximization) step is the maximization of $Q(\lambda', \lambda)$ over λ to obtain $\overline{\lambda}$. Thus the Baum-Welch reestimation equations are essentially identical to the EM steps for this particular problem.

An important property of the reestimation procedure is that the stochastic constraints of the HMM parameters, namely

$$\sum_{i-1}^{N} \overline{\pi}_i = 1 \tag{6.43a}$$

$$\sum_{j=1}^{N} \overline{a}_{ij} = 1, \qquad 1 \le i \le N \tag{6.43b}$$

$$\sum_{k=1}^{M} \overline{b}_j(k) = 1, \qquad 1 \le j \le N \tag{6.43c}$$

are automatically incorporated at each iteration. By looking at the parameter estimation problem as a constrained optimization of $P(\mathbf{O}|\lambda)$ (subject to the constraints of Eq. (6.43)), we can formulate the solution procedure by use of the techniques of variational calculus to maximize P (we use the notation $P = P(\mathbf{O}|\lambda)$ as shorthand in this section). Based on a standard Lagrange optimization setup using Lagrange multipliers, it can readily be shown that P is maximized when the following conditions are met:

$$\pi_i = \frac{\pi_i \dfrac{\partial P}{\partial \pi_i}}{\displaystyle\sum_{k=1}^{N} \pi_k \frac{\partial P}{\partial \pi_k}} \tag{6.44a}$$

$$a_{ij} = \frac{a_{ij} \dfrac{\partial P}{\partial a_{ij}}}{\displaystyle\sum_{k=1}^{N} a_{ik} \frac{\partial P}{\partial a_{ik}}} \tag{6.44b}$$

$$b_j(k) = \frac{b_j(k) \dfrac{\partial P}{\partial b_j(k)}}{\displaystyle\sum_{\ell=1}^{M} b_j(\ell) \frac{\partial P}{\partial b_j(\ell)}}. \tag{6.44c}$$

By appropriate manipulation of Eq. (6.44), the right-hand sides of each equation can be readily shown to be *identical* to the right-hand sides of each part of Eqs. (6.40a)–(6.40c), thereby showing that the reestimation formulas are indeed exactly correct at critical points of P. In fact, the form of Eq. (6.44) is essentially that of a reestimation formula in which the left-hand side is the reestimate and the right-hand side is computed using the current values of the variables.

Finally, we note that since the entire problem can be set up as an optimization problem, standard gradient techniques can be used to solve for "optimal" values of the model parameters. Such procedures have been tried and have been shown to yield solutions comparable to those of the standard reestimation procedures [18]. One critical shortcoming of standard gradient technique, as applied to the maximization of $P(\mathbf{O}|\lambda)$, is that the descent algorithms, which are critically dependent on taking a small step in the direction of the gradient, often do not produce *monotonic* improvement in the likelihood as the Baum-Welch reestimation is guaranteed by Eq. (6.42) to do.

6.5 TYPES OF HMMS

One way to classify types of HMMs is by the structure of the transition matrix, A, of the Markov chain. Until now, we have only considered the special case of ergodic or fully connected HMMs in which every state of the model could be reached (in a single step) from every other state of the model. (Strictly speaking, an ergodic model has the property that every state can be reached from every other state in a finite but aperiodic number of steps.) As shown in Figure 6.8(a), for an $N = 4$ state model, this type of model has the property that every a_{ij} coefficient is positive. Hence for the example of Figure 6.8(a) we have

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix}.$$

For some applications, particularly those to be discussed later in this chapter, other types of HMMs have been found to account for observed properties of the signal being modeled better than the standard ergodic model. One such model is shown in Figure 6.8(b). This model is called a left-right model or a Bakis model ([11], [10]) because the underlying state sequence associated with the model has the property that, as time increases, the state index increases (or stays the same)—that is, the system states proceed from left to right. Clearly the left-right type of HMM has the desirable property that it can readily model signals whose properties change over time in a successive manner—e.g., speech. The fundamental property of all left-right HMMs is that the state-transition coefficients have the property

$$a_{ij} = 0, \qquad j < i \tag{6.45}$$

that is, no transitions are allowed to states whose indices are lower than that of the current state. Furthermore, the initial state probabilities have the property

$$\pi_i = \begin{cases} 0, & i \neq 1 \\ 1, & i = 1 \end{cases} \tag{6.46}$$

because the state sequence must begin in state 1 (and end in state N). Often, with left-right

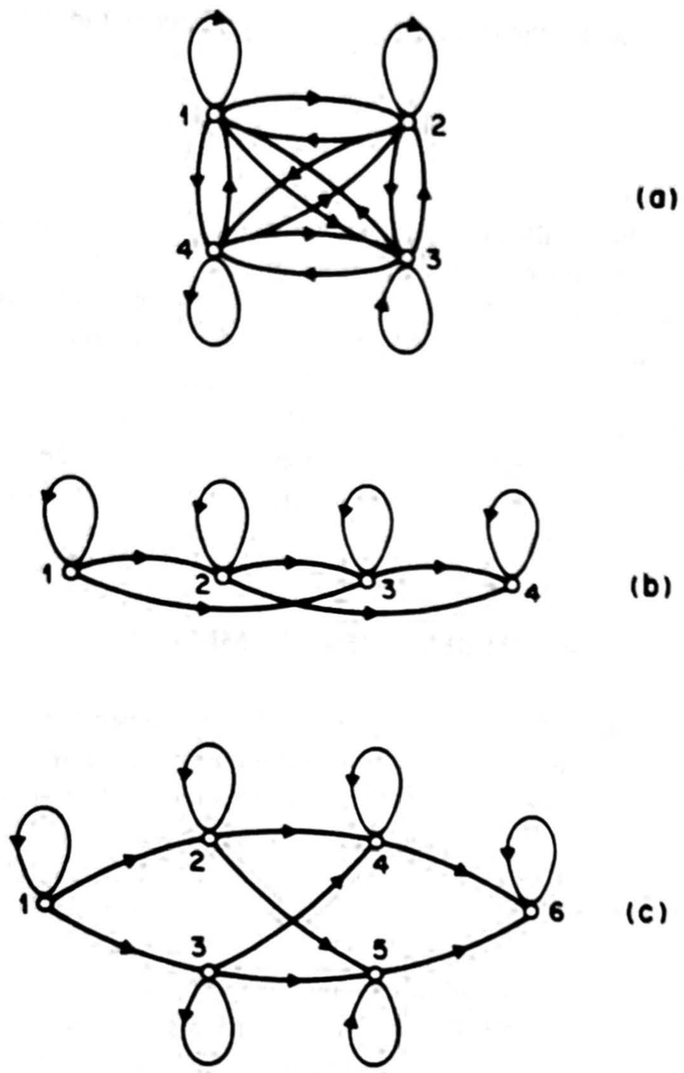


Figure 6.8 Illustration of three distinct types of HMMs. (a) A 4-state ergodic model. (b) A 4-state left-right model. (c) A 6-state parallel path left-right model.

models, additional constraints are placed on the state-transition coefficients to make sure that large changes in state indices do not occur; hence a constraint of the form

$$a_{ij} = 0, \qquad j > i + \Delta i \tag{6.47}$$

is often used. In particular, for the example of Figure 6.8(b), the value of Δi is 2; that is, no jumps of more than two states are allowed. The form of the state-transition matrix for the example of Figure 6.8(b) is thus

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & 0 \\ 0 & a_{22} & a_{23} & a_{24} \\ 0 & 0 & a_{33} & a_{34} \\ 0 & 0 & 0 & a_{44} \end{bmatrix}.$$

It should be clear that, for the last state in a left-right model, the state-transition coefficients are specified as

$$a_{NN} = 1 \tag{6.48a}$$

$$a_{Ni} = 0, \qquad i < N. \tag{6.48b}$$

Besides the above fully connected and left-right models, there are many other possible variations and combinations. By way of example, Figure 6.8(c) shows a cross-coupled connection of two parallel left-right HMMs. Strictly speaking, this model is a left-right model (it obeys all the a_{ij} constraints); however, it has certain flexibility not present in a strict left-right model (i.e., one without parallel paths).

It should be clear that the imposition of the constraints of the left-right model, or those of the constrained jump model, essentially have no effect on the reestimation procedure. This is the case because any HMM parameter set to zero initially will remain at zero throughout the reestimation procedure (see Eq. (6.44)).

6.6 CONTINUOUS OBSERVATION DENSITIES IN HMMS

All of our discussion to this point has considered only when the observations were characterized as discrete symbols chosen from a finite alphabet, and therefore we could use a discrete probability density within each state of this model ([19]–[21]). The problem with this approach, at least for some applications, is that the observations are often continuous signals (or vectors). Although it is possible to convert such continuous signal representations into a sequence of discrete symbols via vector quantization codebooks and other methods, there might be serious degradation associated with such discretization of the continuous signal. Hence it would be advantageous to be able to use HMMs with continuous observation densities to model continuous signal representations directly.

To use a continuous observation density, some restrictions must be placed on the form of the model probability density function (pdf) to ensure that the parameters of the pdf can be reestimated in a consistent way. The most general representation of the pdf, for which a reestimation procedure has been formulated, is a finite mixture of the form

$$b_j(\mathbf{o}) = \sum_{k=1}^{M} c_{jk}\mathcal{N}(\mathbf{o}, \boldsymbol{\mu}_{jk}, \mathbf{U}_{jk}), \qquad 1 \le j \le N \tag{6.49}$$

where $\mathbf{o}$ is the observation vector being modeled, c_{jk} is the mixture coefficient for the kth mixture in state j and $\mathcal{N}$ is any log-concave or elliptically symmetric density [18] (e.g., Gaussian). Without loss of generality, we assume that $\mathcal{N}$ is Gaussian in Eq. (6.49) with mean vector $\boldsymbol{\mu}_{jk}$ and covariance matrix $\mathbf{U}_{jk}$ for the kth mixture component in state j. The mixture gains c_{jk} satisfy the stochastic constraint

$$\sum_{k=1}^{M} c_{jk} = 1, \qquad 1 \le j \le N \tag{6.50a}$$

$$c_{jk} \ge 0, \; 1 \le j \le N, \; 1 \le k \le M \tag{6.50b}$$

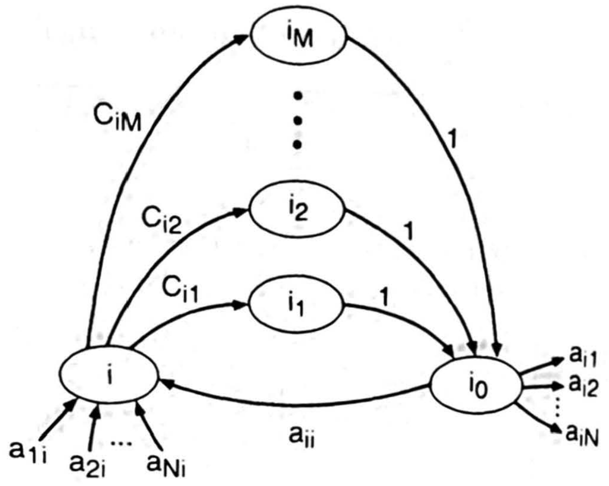


Figure 6.9 Equivalence of a state with a mixture density to a multistate single-density distribution (after Juang et al. [21]).

so that the pdf is properly normalized, i.e.,

$$\int_{-\infty}^{\infty} b_j(\mathbf{o})d\mathbf{o} = 1, \qquad 1 \le j \le N. \tag{6.51}$$

The pdf of Eq. (6.49) can be used to approximate, arbitrarily closely, any finite, continuous-density function. Hence it can be applied to a wide range of problems.

It has been shown that an HMM state with a mixture density is equivalent to a multistate single-mixture density model in the following way [21]. Consider a state i with an M-mixture Gaussian density. Because the mixture gain coefficients sum up to 1, they define a set of transition coefficients to substates i_1 (with transition probability c_{i1}), i_2 (with transition probability c_{i2}) through i_M (with transition probability c_{iM}). Within each substate i_k, there is a single mixture with mean $\boldsymbol{\mu}_{ik}$ and variance $\mathbf{U}_{ik}$ (see Figure 6.9 for a graphical interpretation). Each substate makes a transition to a wait state i_0 with probability 1. The distribution of the composite set of substates (each with a single density) is mathematically equivalent to the composite mixture density within a single state.

It can be shown that the reestimation formulas for the coefficients of the mixture density, i.e., c_{jk}, $\boldsymbol{\mu}_{jk}$, and $\mathbf{U}_{jk}$, are of the form

$$\bar{c}_{jk} = \frac{\sum_{t=1}^{T} \gamma_t(j,k)}{\sum_{t=1}^{T}\sum_{k=1}^{M} \gamma_t(j,k)} \tag{6.52}$$

$$\bar{\boldsymbol{\mu}}_{jk} = \frac{\sum_{t=1}^{T} \gamma_t(j,k) \cdot \mathbf{o}_t}{\sum_{t=1}^{T} \gamma_t(j,k)} \tag{6.53}$$

$$\bar{\mathbf{U}}_{jk} = \frac{\sum_{t=1}^{T} \gamma_t(j,k) \cdot (\mathbf{o}_t - \boldsymbol{\mu}_{jk})(\mathbf{o}_t - \boldsymbol{\mu}_{jk})'}{\sum_{t=1}^{T} \gamma_t(j,k)} \tag{6.54}$$

where prime denotes vector transpose and where $\gamma_t(j,k)$ is the probability of being in state j at time t with the kth mixture component accounting for $\mathbf{o}_t$, i.e.,

$$\gamma_t(j,k) = \left[\frac{\alpha_t(j)\beta_t(j)}{\sum_{j=1}^{N} \alpha_t(j)\beta_t(j)} \right] \left[\frac{c_{jk}\mathcal{N}(\mathbf{o}_t, \boldsymbol{\mu}_{jk}, \mathbf{U}_{jk})}{\sum_{m=1}^{M} c_{jm}\, \mathcal{N}(\mathbf{o}_t, \boldsymbol{\mu}_{jm}, \mathbf{U}_{jm})} \right].$$

(The term $\gamma_t(j,k)$ generalizes to $\gamma_t(j)$ of Eq. (6.26) in the case of a simple mixture, or a discrete density.) The reestimation formula for a_{ij} is identical to the one used for discrete observation densities (i.e., Eq. (6.40(b))). The interpretation of Eqs. (6.52)–(6.54) is fairly straightforward. The reestimation formula for c_{jk} is the ratio between the expected number of times the system is in state j using the kth mixture component, and the expected number of times the system is in state j. Similarly, the reestimation formula for the mean vector $\boldsymbol{\mu}_{jk}$ weights each numerator term of Eq. (6.52) by the observation, thereby giving the expected value of the portion of the observation vector accounted for by the kth mixture component. A similar interpretation can be given for the reestimation term for the covariance matrix $\mathbf{U}_{jk}$.

6.7 AUTOREGRESSIVE HMMS

Another very interesting class of HMMs that is particularly applicable to speech processing is the class of autoregressive HMMs ([22, 23]). For this class, the observation vectors are drawn from an autoregression process. (The autoregressive density is, of course, just another continuous-probability density. However, we elaborate on the subject here separately from Section 6.6 because of its importance in speech analysis as will be shown later.)

To be more specific, consider the observation vector $\mathbf{o} = (x_0, x_1, x_2, \ldots, x_{K-1})$. The elements, x_i, could be simply the speech waveform samples. The components of $\mathbf{o}$ are assumed to be from an autoregressive Gaussian source, satisfying the relationship

$$x_k = -\sum_{i=1}^{p} a_i x_{k-i} + e_k \tag{6.55}$$

where e_k, $k = 0, 1, 2, \ldots, K-1$ are Gaussian, independent, identically distributed random variables with zero mean and variance σ_e^2, and a_i, $i = 1, 2, \ldots, p$, are the autoregression or predictor coefficients. It can be shown that for large K [22, 23], the density function for $\mathbf{o}$

is approximately

$$f(\mathbf{o}) = (2\pi\sigma_e^2)^{-K/2} \exp\left\{-\frac{1}{2\sigma_e^2}\delta(\mathbf{o}, \mathbf{a})\right\} \tag{6.56}$$

where

$$\delta(\mathbf{o}, \mathbf{a}) = r_a(0)r(0) + 2\sum_{i=1}^{p} r_a(i)r(i) \tag{6.57a}$$

$$\mathbf{a} = [1, a_1, a_2, \ldots, a_p]', \qquad (a_0 = 1) \tag{6.57b}$$

$$r_a(i) = \sum_{n=0}^{p-i} a_n a_{n+i}, \qquad 1 \le i \le p \tag{6.57c}$$

$$r(i) = \sum_{n=0}^{K-i-1} x_n x_{n+i} \qquad 0 \le i \le p. \tag{6.57d}$$

In the above equations $r(i)$ is the autocorrelation of the observation samples, and $r_a(i)$ is the autocorrelation of the autoregressive coefficients. Furthermore, $\delta(\mathbf{o}, \mathbf{a})$ is a form of residual energy resulting from inverse filtering the data x_i with an all-zero filter defined by $\mathbf{a}$. (See Eqs. 4.40–44.)

As discussed in Chapter 4, the signal level in the observation $\mathbf{o}$ is often treated in a different fashion from the general spectral shape when it comes to speech-pattern comparison. One way to separate the signal level from the spectral shape is to use gain normalization; that is, we use $\hat{\mathbf{o}}$ instead of $\mathbf{o}$ as the observation, where

$$\hat{\mathbf{o}} = \mathbf{o}/\sigma_{eo} \tag{6.58}$$

and where σ_{eo}^2 is the minimum linear prediction residual energy per sample. (It is shown in Exercise 6.5 that $\sigma_{eo}^2 = (\sigma_e^2)_{\mathrm{ML}}$ for *the given observation* $\mathbf{o}$.) The elements of $\hat{\mathbf{o}}$, $\hat{x}_k = x_k/\sigma_{eo}$, still satisfy the autoregressive relationship

$$\hat{x}_k = -\sum_{i=1}^{p} a_i \hat{x}_{k-i} + \hat{e}_k. \tag{6.59}$$

However, the variance of $\hat{e}_k$ is now unity. Therefore, we can write the probability density function for the output of an all-pole system defined by $\mathbf{a}$, driven by a zero mean, unit variance Gaussian i.i.d. sequence, as

$$f(\hat{\mathbf{o}}) = (2\pi)^{-K/2} \exp\left\{-\frac{1}{2}\delta(\hat{\mathbf{o}}, \mathbf{a})\right\} \tag{6.60}$$

if the data dimension K is sufficiently large. (Note that the normalization factor σ_{eo} depends on the original data observation $\mathbf{o}$.) This type of pdf is often referred to as a "gain-independent" pdf.

The way in which we use a Gaussian autoregressive density in HMMs is straightforward. We assume a mixture density of the form

$$b_j(\mathbf{o}) = \sum_{k=1}^{M} c_{jk} b_{jk}(\mathbf{o}) \tag{6.61}$$

where each $b_{jk}(\mathbf{o})$ is the density defined by Eq. (6.60) with autoregression vector $\mathbf{a}_{jk}$ (or equivalently by autocorrelation vector $\mathbf{r}_{a_{jk}}$); that is,

$$b_{jk}(\mathbf{o}) = (2\pi)^{-K/2} \exp\left\{-\frac{1}{2}\delta(\mathbf{o}, \mathbf{a}_{jk})\right\}. \tag{6.62}$$

A reestimation formula for the sequence autocorrelation for the jth state, kth mixture component has been derived [22, 23], and is of the form

$$\bar{\mathbf{r}}_{jk} = \frac{\sum_{t=1}^{T} \gamma_t(j,k) \cdot \mathbf{r}_t}{\sum_{t=1}^{T} \gamma_t(j,k)} \tag{6.63a}$$

where $\mathbf{r}_t = [r_t(0), r_t(1), \ldots, r_t(p)]'$ is the autocorrelation vector as defined by Eq. (6.57d) for the t^{th} frame, and $\gamma_t(j,k)$ is defined as the probability of being in state j at time t and using mixture component k, i.e.,

$$\gamma_t(j,k) = \left[\frac{\alpha_t(j)\beta_t(j)}{\sum_{j=1}^{N}\alpha_t(j)\beta_t(j)}\right]\left[\frac{c_{jk}b_{jk}(\mathbf{o}_t)}{\sum_{k=1}^{M} c_{jk}b_{jk}(\mathbf{o}_t)}\right]. \tag{6.63b}$$

It can be seen that $\bar{\mathbf{r}}_{jk}$ is a weighted sum (by probability of occurrence) of the normalized autocorrelations of the frames in the observation sequence. From $\bar{\mathbf{r}}_{jk}$, one can solve a set of normal equations to obtain the corresponding autoregressive coefficient vector $\bar{\mathbf{a}}_{jk}$, for the kth mixture of state j. The new autocorrection vectors of the autoregression coefficients as needed in the density function can then be calculated using Eq. (6.57c), thereby closing the reestimation loop.

Exercise 6.4

The probability density function (pdf) [22, 23] of Eq. (6.56) is defined by parameters σ_e^2 and $\mathbf{a}$. Given a data observation vector $\mathbf{o} = (x_0, x_1, \ldots, x_{K-1})$, determine the maximum likelihood estimate of σ_e^2 and $\mathbf{a}$ that best characterizes the observed $\mathbf{o}$.

Solution 6.4

We write the likelihood function of $\mathbf{o}$ as a function of σ_e^2 and $\mathbf{a}$ as

$$f(\mathbf{o}|\sigma_e^2, \mathbf{a}) = (2\pi\sigma_e^2)^{-K/2} \exp\left\{-\frac{1}{2\sigma_e^2}\delta(\mathbf{o}, \mathbf{a})\right\}$$

and the log likelihood function as

$$\log f(\mathbf{o}|\sigma_e^2, \mathbf{a}) = -\frac{K}{2}\log(2\pi\sigma_e^2) - \frac{\delta(\mathbf{o}, \mathbf{a})}{2\sigma_e^2}.$$

Therefore, the maximum likelihood (ML) estimate is

$$(\mathbf{a})_{\mathrm{ML}} = \arg\max_{\mathbf{a}} \log f(\mathbf{o}|\sigma_e^2, \mathbf{a}) = \arg\min_{\mathbf{a}} \delta(\mathbf{o}, \mathbf{a})$$
$$= \arg\min_{\mathbf{a}}(\mathbf{a}'\mathbf{R}\mathbf{a})$$

where $\mathbf{R} = [r_{ij}]$ with $r_{ij} = r(|i-j|)$ as defined by Eq. (6.57d). This establishes the relationship between maximum likelihood estimation and the classical method of autocorrelation matching (also called the method of minimization of prediction residuals) for LPC analysis (see Eq. (4.8)–(4.10)). Furthermore, it is easy to show that

$$(\sigma_e^2)_{\mathrm{ML}} = \min_{\mathbf{a}} \delta(\mathbf{o}, \mathbf{a})/K$$

which leads to

$$\max_{\sigma_e^2, \mathbf{a}} \log f(\mathbf{o}|\sigma_e^2, \mathbf{a}) = -\frac{K}{2} \log \left[2\pi(\sigma_e^2)_{\mathrm{ML}}\right] - \frac{K}{2}.$$

Exercise 6.5

The pdf of Eq. (6.56) is related to the Itakura-Saito distortion measure of Eq. (4.45). Establish this relationship.

Solution 6.5

Consider the following likelihood difference

$$\begin{aligned} L_d &= \left[\max_{\sigma_e^2, \mathbf{a}} \log f(\mathbf{o}|\sigma_e^2, \mathbf{a})\right] - \log f(\mathbf{o}|\sigma_e^2, \mathbf{a}) \\ &= -\frac{K}{2} \log [2\pi(\sigma_e^2)_{\mathrm{ML}}] - \frac{K}{2} + \frac{K}{2} \log (2\pi\sigma_e^2) + \frac{\delta(\mathbf{o}, \mathbf{a})}{2\sigma_e^2} \\ &= \frac{K}{2}\left[\frac{1}{K\sigma_e^2}\delta(\mathbf{o}, \mathbf{a}) + \log \sigma_e^2 - \log (\sigma_e^2)_{\mathrm{ML}} - 1\right]. \end{aligned}$$

For distortion measures, we denote the all-pole (power) spectra by $\sigma_{eo}^2/\left|A_o(e^{j\omega})\right|^2$ and $\sigma_e^2/\left|A(e^{j\omega})\right|^2$, corresponding to parameter sets $\{(\sigma_e^2)_{\mathrm{ML}}, (\mathbf{a})_{\mathrm{ML}}\}$ and $\{\sigma_e^2, \mathbf{a}\}$, respectively. ($\sigma_{fo}^2 = (\sigma_e^2)_{\mathrm{ML}}$.) Then, the bracketed term in the log likelihood difference, L_d, is simply $d_{\mathrm{IS}}(\sigma_{fo}^2/|A_o|^2, \sigma_e^2/|A|^2)$ according to Eq. (4.45) because

$$\sigma_{fo}^2 \int_{-\pi}^{\pi} \frac{\left|A(e^{j\omega})\right|^2}{|A_o(e^{j\omega})|^2} \frac{d\omega}{2\pi} = \frac{1}{K}\delta(\mathbf{o}, \mathbf{a})$$

due to autocorrelation matching and Eq. (6.57a). (Note that we have defined σ_e^2 (and σ_{eo}^2) as the sample variance. The factor K would disappear from the above equation if we use the total frame prediction residual variance instead of the sample variance.) Therefore,

$$\begin{aligned} f(\mathbf{o}|\sigma_e^2, \mathbf{a}) &= (2\pi\sigma_e^2)^{-K/2} \exp\left\{-\frac{K}{2}\left[d_{\mathrm{IS}}\left(\frac{\sigma_{eo}^2}{|A_o|^2}, \frac{\sigma_e^2}{|A|^2}\right) + \log \sigma_{fo}^2 - \log \sigma_e^2 + 1\right]\right\} \\ &= G_1(\sigma_{fo}^2, \sigma_e^2) \exp\left\{-\frac{K}{2} d_{\mathrm{IS}}\left(\frac{\sigma_{fo}^2}{|A_o|^2}, \frac{\sigma_e^2}{|A|^2}\right)\right\} \end{aligned}$$

where $G_1(\sigma_{f_0}^2, \sigma_e^2)$ encompasses only the gain terms.

Exercise 6.6

The pdf of Eq. (6.60) is related to the likelihood ratio distortion measure of Eq. (4.53). Establish this relationship.

Solution 6.6

Similar to the case of Eq. (6.56),

$$\begin{aligned}\frac{1}{K}\delta(\hat{\mathbf{o}}, \mathbf{a}) &= \int_{-\pi}^{\pi} \frac{\left|A(e^{j\omega})\right|^2}{|A_o(e^{j\omega})|^2} \frac{d\omega}{2\pi} \\ &= d_{\mathrm{LR}}\left(\frac{1}{|A_o|^2}, \frac{1}{|A|^2}\right) + 1.\end{aligned}$$

Therefore,

$$\begin{aligned}f(\hat{\mathbf{o}}|\mathbf{a}) &= (2\pi)^{-K/2} \exp\left\{-\frac{K}{2}\left[d_{\mathrm{LR}}\left(\frac{1}{|A_o|^2}, \frac{1}{|A|^2}\right) + 1\right]\right\} \\ &= G_2 \exp\left\{-\frac{K}{2} d_{\mathrm{LR}}\left(\frac{1}{|A_o|^2}, \frac{1}{|A|^2}\right)\right\}.\end{aligned}$$

Note that when the pdf is expressed in terms of the distortion measures (d_{IS} or d_{LR}), the exponential term includes a factor K that represents the data dimension. In practice, this factor K is replaced by an effective frame length $\hat{K}$, which is the net shift of consecutive data frames. Thus, if consecutive data frames (vectors) have $2/3$ overlap, then an effective frame length $\hat{K} = K/3$ is appropriate, so that the rate of characteristic change in terms of the spectral parameters $\mathbf{a}$ is kept at the original waveform sampling rate.

6.8 VARIANTS ON HMM STRUCTURES—NULL TRANSITIONS AND TIED STATES

Throughout this chapter we have considered HMMs in which the observations were associated with states of the model. It is also possible to consider models in which the observations are associated with the arcs of the model. This type of HMM has been used extensively in the IBM continuous speech recognizer [13]. It has been found useful, for this type of model, to allow transitions that produce no output; that is, jumps from one state to another that produce no observation [13]. Such transitions are called null transitions and are designated by a dashed line, with the symbol ϕ used to denote the null output.

Figure 6.10 illustrates three examples (from speech-processing tasks) where null arcs have been successfully utilized. The example of part (a) corresponds to an HMM (a left-right model) with a large number of states in which it is possible to omit transitions between any pair of states. Hence it is possible to generate observation sequences with as few as

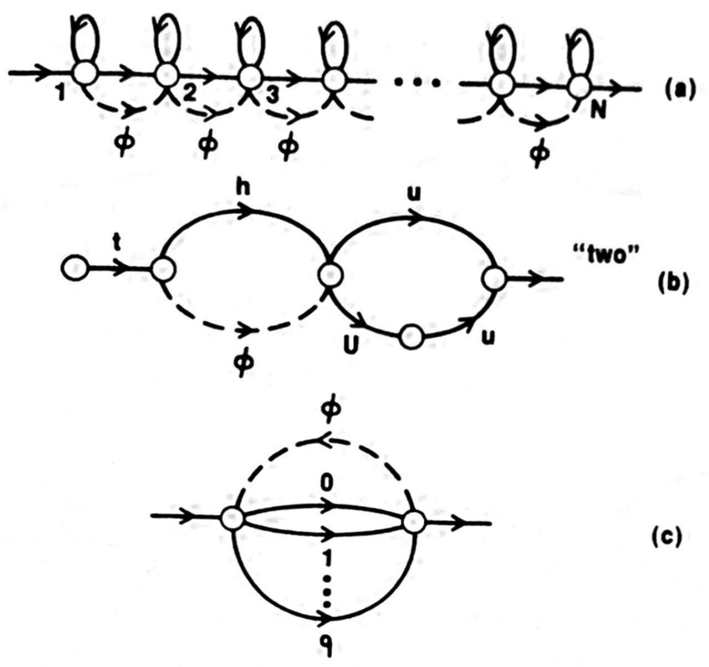


Figure 6.10 Examples of networks incorporating null transitions. (a) Left-right model. (b) Finite state network. (c) Grammar network.

one observation and still account for a path that begins in state one and ends in state N.

The example of Figure 6.10(b) is a finite-state network (FSN) representation of a word in terms of linguistic unit models (i.e., the sound on each arc is itself an HMM). For this model the null transition gives a compact and efficient way of describing alternative word pronunciations (i.e., symbol deletions).

Finally the FSN of Figure 6.10(c) shows how the ability to insert a null transition into a grammar network allows a relatively simple network to generate arbitrarily long word (digit) sequences. In the example shown in Figure 6.10(c), the null transition allows the network to generate arbitrary sequences of digits of arbitrary length by returning to the initial state after each individual digit is produced.

Another interesting variation in the HMM structure is the concept of parameter tying [13]. Basically, the idea is to set up an equivalence relation between HMM parameters in different states. In this manner the number of independent parameters in the model is reduced and the parameter estimation becomes somewhat simpler and in some cases more reliable. Parameter tieing is used when the observation density, for example, is known to be the same in two or more states. Such cases occur often in characterizing speech sounds. The technique is especially appropriate when there is insufficient training data to estimate, reliably, a large number of model parameters. For cases such as these, it is appropriate to tie model parameters so as to reduce the number of parameters (i.e., size of the model), thereby making the parameter estimation problem somewhat simpler. We will discuss this method later in this chapter.

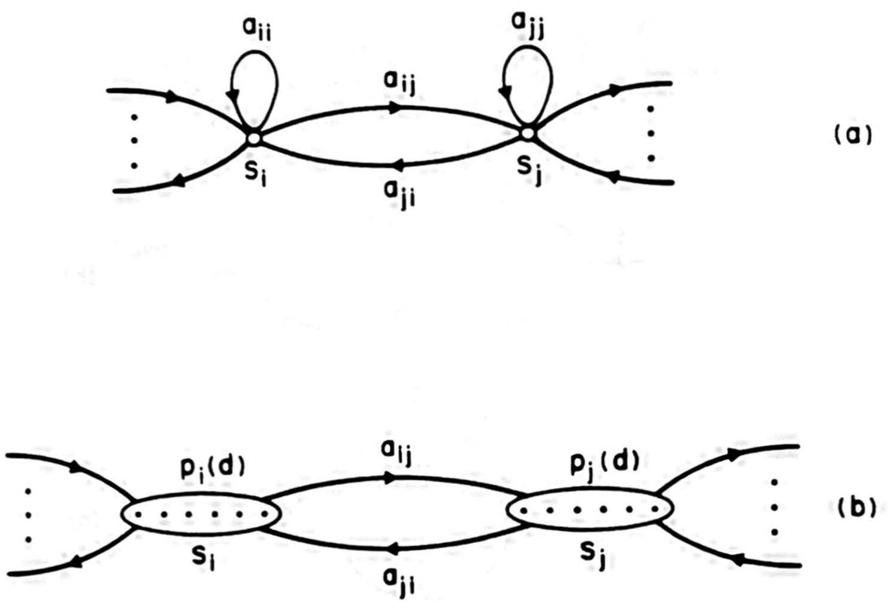


Figure 6.11 Illustration of general interstate connections of (a) a normal HMM with exponential state duration density, and (b) a variable duration HMM with specified state densities and no self-transitions from a state back to itself.

6.9 INCLUSION OF EXPLICIT STATE DURATION DENSITY IN HMMS

Earlier we showed via Eq. (6.5) that the inherent duration probability density $p_i(d)$ associated with state i, with self-transition coefficient a_{ii}, was of the form

$$\begin{aligned} p_i(d) &= (a_{ii})^{d-1}(1 - a_{ii}) \\ &= \text{probability of } d \text{ consecutive observations in state } i. \end{aligned} \tag{6.64}$$

For most physical signals, this exponential state duration density is inappropriate. Instead we would prefer to explicitly model duration density in some analytic form. (An extensive treatment of state duration modeling can be found in the work of Ferguson of IDA [14], which is the basis of the material presented here. Other valuable references include [24] and [25].) Figure 6.11 illustrates, for a pair of model states i and j, the differences between HMMs without and with explicit duration density. In part (a) the states have exponential duration densities based on self-transition coefficients a_{ii} and a_{jj}, respectively. In part (b), the self-transition coefficients are set to zero, and an explicit duration density is specified. For this case, a transition is made only after the appropriate number of observations have occurred in the state (as specified by the duration density). Such a model is called a semi-Markov model.

Based on the simple model of Figure 6.11(b), the sequence of events of the variable duration HMM is as follows:

1. An initial state, $q_1 = i$, is chosen according to the initial state distribution π_i.

2. A duration d_1 is chosen according to the state duration density $p_{q_1}(d_1)$. (For expedience and ease of implementation, the duration density $p_q(d)$ is truncated at a maximum duration value D.)
3. Observations $\mathbf{o}_1\,\mathbf{o}_2\ldots\mathbf{o}_{d_1}$ are chosen according to the joint observation density, $b_{q_1}(\mathbf{o}_1\,\mathbf{o}_2\ldots\mathbf{o}_{d_1})$. Generally we assume that in each state observations are independent so that $b_{q_1}(\mathbf{o}_1\,\mathbf{o}_2\ldots\mathbf{o}_{d_1}) = \prod_{t=1}^{d_1} b_{q_1}(\mathbf{o}_t)$.
4. The next state, $q_2 = j$, is chosen according to the state transition probabilities, $a_{q_1q_2}$, with the constraint that $a_{q_1q_1} = 0$, i.e., no transition back to the same state can occur. (Clearly this is a requirement, because we assume that, in state q_1, exactly d_1 observations occur.)

A little thought should convince the reader that the variable duration HMM can be made equivalent to the standard HMM by setting $p_i(d)$ to be the exponential density of (6.64).

Using the above formulation, we must make several changes to the formulas of Section 6.4.3 to allow calculation of $P(\mathbf{O}|\lambda)$ and for reestimation of all model parameters. In particular we assume that the first state begins at $t = 1$ and the last state ends at $t = T$. We then define the forward variable $\alpha_t(i)$ as

$$\alpha_t(i) = P(\mathbf{o}_1\,\mathbf{o}_2\ldots\mathbf{o}_t, \text{ the stay in state } i \text{ ends at } t|\lambda). \tag{6.65}$$

We assume that a total of r states have been visited during the first t observations, and we denote the states as $q_1, q_2, \ldots, q_r$ with durations associated with each state of $d_1, d_2, \ldots, d_r$. Thus the constraints of Eq. (6.65) are

$$q_r = i \tag{6.66a}$$

$$\sum_{s=1}^{r} d_s = t. \tag{6.66b}$$

Eq. (6.65) can then be written as

$$\begin{aligned}\alpha_t(i) = \sum_q \sum_d \pi_{q_1} \cdot p_{q_1}(d_1) \cdot P(\mathbf{o}_1\,\mathbf{o}_2\ldots\mathbf{o}_{d_1}|q_1)\\ \cdot a_{q_1q_2}p_{q_2}(d_2)P(\mathbf{o}_{d_1+1}\ldots\mathbf{o}_{d_1+d_2}|q_2)\ldots\\ \cdot a_{q_{r-1}q_r}p_{q_r}(d_r)P(\mathbf{o}_{d_1+d_2+\cdots+d_{r-1}+1}\ldots\mathbf{o}_t|q_r)\end{aligned} \tag{6.67}$$

where the sum is over all states q and all possible state durations d. By induction we can write $\alpha_t(j)$ as

$$\alpha_t(j) = \sum_{i=1}^{N}\sum_{d=1}^{D} \alpha_{t-d}(i)a_{ij}p_j(d) \prod_{s=t-d+1}^{t} b_j(\mathbf{o}_s) \tag{6.68}$$

where D is the maximum duration within any state. To initialize the computation of $\alpha_t(j)$ we use

$$\alpha_1(i) = \pi_i p_i(1) \cdot b_i(\mathbf{o}_1) \tag{6.69a}$$

$$\alpha_2(i) = \pi_i p_i(2) \prod_{s=1}^{2} b_i(\mathbf{o}_s) + \sum_{\substack{j=1 \\ j \neq i}}^{N} \alpha_1(j) a_{ji} p_i(1) b_i(\mathbf{o}_2) \tag{6.69b}$$

$$\alpha_3(i) = \pi_i p_i(3) \prod_{s=1}^{3} b_i(\mathbf{o}_s) + \sum_{d=1}^{2} \sum_{\substack{j=1 \\ j \neq i}}^{N} \alpha_{3-d}(j) a_{ji} p_i(d) \tag{6.69c}$$

$$\cdot \prod_{s=4-d}^{3} b_i(\mathbf{o}_s) \tag{6.69d}$$

and so on, until $\alpha_D(i)$ is computed; then Eq. (6.68) can be used for all $t > D$. It should be clear that the desired probability of $\mathbf{O}$ given the model λ can be written in terms of the αs as

$$P(\mathbf{O}|\lambda) = \sum_{i=1}^{N} \alpha_T(i) \tag{6.70}$$

as was previously used for ordinary HMMs.

To give reestimation formulas for all the variables of the variable duration HMM, we must define three more forward-backward variables, namely

$$\alpha_t^*(i) = P(\mathbf{o}_1 \mathbf{o}_2 \ldots \mathbf{o}_t, \text{ stay in state } i \text{ starts at } t+1|\lambda) \tag{6.71}$$

$$\beta_t(i) = P(\mathbf{o}_{t+1} \ldots \mathbf{o}_T | \text{ stay in state } i \text{ ends at } t, \lambda) \tag{6.72}$$

$$\beta_t^*(i) = P(\mathbf{o}_{t+1} \ldots \mathbf{o}_T | \text{ stay in state } i \text{ starts at } t+1, \lambda). \tag{6.73}$$

The relationships between α, α^*, β, and β^* are as follows:

$$\alpha_t^*(j) = \sum_{i=1}^{N} \alpha_t(i) a_{ij} \tag{6.74}$$

$$\alpha_t(i) = \sum_{d=1}^{D} \alpha_{t-d}^*(i) p_i(d) \prod_{s=t-d+1}^{t} b_i(\mathbf{o}_s) \tag{6.75}$$

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} \beta_t^*(j) \tag{6.76}$$

$$\beta_t^*(i) = \sum_{d=1}^{D} \beta_{t+d}(i) p_i(d) \prod_{s=t+1}^{t+d} b_i(\mathbf{o}_s). \tag{6.77}$$

Based on the above relationships and definitions, the reestimation formulas for the variable duration HMM, with discrete observations, are

$$\bar{\pi}_i = \frac{\pi_i \beta_0^*(i)}{P(\mathbf{O}|\lambda)} \tag{6.78}$$

$$\bar{a}_{ij} = \frac{\sum_{t=1}^{T} \alpha_t(i) a_{ij} \beta_t^*(j)}{\sum_{j=1}^{N} \sum_{t=1}^{T} \alpha_t(i) a_{ij} \beta_t^*(j)} \tag{6.79}$$

$$\bar{b}_i(k) = \frac{\sum_{\substack{t=1 \\ \text{s.t. } \mathbf{o}_t = \mathbf{v}_k}}^{T} \left[\sum_{\tau<t} \alpha_\tau^*(i) \cdot \beta_\tau^*(i) - \sum_{\tau<t} \alpha_\tau(i) \beta_\tau(i) \right]}{\sum_{k=1}^{M} \sum_{\substack{t=1 \\ \text{s.t. } \mathbf{o}_t = \mathbf{v}_k}}^{T} \left[\sum_{\tau<t} \alpha_\tau^*(i) \cdot \beta_\tau^*(i) - \sum_{\tau<t} \alpha_\tau(i) \beta_\tau(i) \right]} \tag{6.80}$$

$$\bar{p}_i(d) = \frac{\sum_{t=1}^{T} \alpha_t^*(i) p_i(d) \beta_{t+d}(i) \prod_{s=t+1}^{t+d} b_i(\mathbf{o}_s)}{\sum_{d=1}^{D} \sum_{t=1}^{T} \alpha_t^*(i) p_i(d) \beta_{t+d}(i) \prod_{s=t+1}^{t+d} b_i(\mathbf{o}_s)}. \tag{6.81}$$

The interpretation of the reestimation formulas is the following: The formula for $\bar{\pi}_i$ is the probability that state i was the first state, given $\mathbf{O}$. The formula for $\bar{a}_{ij}$ is almost the same as for the usual HMM, except it uses the condition that the alpha terms in which a state ends at t, join with the beta terms in which a new state begins at $t + 1$. The formula for $\bar{b}_i(k)$ is the expected number of times that observation $\mathbf{o}_t = \mathbf{v}_k$ occurred in state i, normalized by the expected number of times that any observation occurred in state i. Finally, the reestimation formula for $\bar{p}_i(d)$ is the ratio of the expected number of times state i occurred with any duration.

The importance of incorporating state duration densities is reflected in the observation that, for some problems, the quality of the modeling is significantly improved when explicit state duration densities are used. However, there are drawbacks to the use of the variable duration model discussed in this section. One is the greatly increased computational load associated with using variable durations. It can be seen from the definition and initialization conditions on the forward variable $\alpha_t(i)$, from Eqs. (6.68)–(6.69), that about D times the storage and $D^2/2$ times the computation is required. For D on the order of 25 (as is reasonable for many speech-processing problems), computation is increased by a factor of 300. Another problem with the variable duration models is the large number of parameters (D), associated with each state, that must be estimated, in addition to the usual HMM parameters. Furthermore, for a fixed number of observations T, in the training set, there are, on average, fewer state transitions and much less data to estimate $p_i(d)$ than would be used in a standard HMM. Thus the reestimation problem is more difficult for variable duration HMMs than for the standard HMM.

One proposal to alleviate some of these problems is to use a parametric state duration density instead of the nonparametric $p_i(d)$ used above [23–24]. In particular, proposals include the Gaussian family with

$$p_i(d) = \mathcal{N}(d, \mu_i, \sigma_i^2) \tag{6.82}$$

with parameters μ_i and σ_i^2, or the Gamma family with

$$p_i(d) = \frac{\eta_i^{\nu_i} d^{\nu_i - 1} e^{-\eta_i d}}{\Gamma(\nu_i)} \tag{6.83}$$

with parameters ν_i and η_i and with mean $\nu_i\eta_i^{-1}$ and variance $\nu_i\eta_i^{-2}$. Reestimation formulas for η_i and ν_i have been derived and used with good results. Another possibility, which has been used with good success, is to assume a uniform duration distribution over an appropriate range of durations and use a path-constrained Viterbi decoding procedure.

6.10 OPTIMIZATION CRITERION—ML, MMI, AND MDI

The standard ML design criterion is to use a training sequence of observations $\mathbf{O}$ to derive the set of model parameters λ, yielding

$$\lambda_{\text{ML}} = \arg\max_{\lambda} P(\mathbf{O}|\lambda). \tag{6.84}$$

Any of the reestimation algorithms discussed previously provides a solution to this optimization problem.

The need to consider alternative design criteria, however, comes from several concerns ([26–28]). The basic philosophy in statistical modeling methods, such as HMM, is that the signal or observation sequence can be well modeled if the parameters of the model are carefully and correctly chosen. The problem with this philosophy is that the assumed model—HMM in the present case—is sometimes inadequate to model the observed signal so that no matter how carefully the parameters are chosen, the modeling accuracy is limited. Often, this situation is described as a "model mismatch." The first alternative optimization criterion we discuss here is one that tries to overcome the problem of model mismatch in order to achieve a more accurate modeling of the observation signal.

The observed signal $\mathbf{O} = (\mathbf{o}_1, \mathbf{o}_2, \ldots, \mathbf{o}_T)$ is associated with a sequence of constraints $\mathcal{R} = (\mathbf{R}_1, \mathbf{R}_2, \ldots, \mathbf{R}_T)$. For example, $\mathbf{R}_t$ may be the autocorrelation matrix that characterizes the observation $\mathbf{o}_t$. Then, obviously, $\mathbf{O}$ is only one of possibly uncountably many observation sequences that satisfy the constraint sequence $\mathcal{R}$. Furthermore, in terms of the probability distributions of the observation sequences, there exists a set of such distributions that would also satisfy $\mathcal{R}$. We denote this set $\Omega(\mathcal{R})$. The minimum discrimination information (MDI) is a measure of closeness between two probability measures (one of which bears the HMM form here) under the given constraint $\mathcal{R}$ and is defined by

$$\nu(\mathcal{R}, P_\lambda) \triangleq \inf_{Q \in \Omega(\mathcal{R})} I(Q : P_\lambda) \tag{6.85}$$

where

$$I(Q : P_\lambda) = \int q(\mathbf{O}) \log \frac{q(\mathbf{O})}{p(\mathbf{O}|\lambda)} d\mathbf{O} \tag{6.86}$$

is the discrimination information between distributions Q and P_λ [27,28]. (The functions $q(\cdot)$ and $p(\cdot|\lambda)$ are the probability density functions corresponding to Q and P_λ respectively.) The discrimination information is calculated based on the given training set of observations.

The MDI criterion tries to choose a model parameter set λ such that $\nu(\mathcal{R}, P_\lambda)$ is minimized. An interpretation of MDI is that the model parameter set λ is chosen so that the model $p(\mathbf{O}|\lambda)$ is as close as it can be to a member of the set $\Omega(\mathcal{R})$. Since the closeness is always measured in terms of the discrimination information evaluated on the given observation, the intrinsic characteristics of the training sequences would then have substantial influence on the parameter selection. By emphasizing the measure discrimination, the model estimation is no longer solely dictated by the assumed model form. The MDI optimization problem is, however, not as straightforward as the ML optimization problem and no simple robust implementation of the procedure is known.

Another concern about the HMM optimization criterion arises when we attempt to use it to solve a class of speech-recognition problems. Consider recognition of a vocabulary of V words, each of which is represented by an HMM, with parameter set λ_v, $v = 1, 2, \ldots, V$. We assume $P(v)$ to be the a priori probability for word v, $v = 1, 2, \ldots, V$. The set of HMMs $\Lambda = \{\lambda_v\}$ together with the a priori probabilities thus defines a probability measure for an arbitrary observation sequence $\mathbf{O}$

$$P_\Lambda(\mathbf{O}) = \sum_{v=1}^{V} P(\mathbf{O}|\lambda_v, v)P(v). \tag{6.87}$$

(The notation $P(\mathbf{O}|\lambda_v, v)$ indicates that it is a probability conditioned on the word v. We include the model parameter λ_v, sometimes, because of the necessity of treating λ_v as random variables for estimation purposes. Obviously, when λ_v is fixed, $P(\mathbf{O}|\lambda_v, v)$ is the conditional probability, parameterized by λ_v.) To train these models (i.e., to estimate the optimum parameters of the associated models), utterances of known (labeled) words are used. We denote the labeled training sequence by $\mathbf{O}^v$ where superscript v reflects the fact that $\mathbf{O}^v$ is a rendition of word v. The standard ML criterion of Eq. (6.84) is to use $\mathbf{O}^v$ to estimate model parameters λ_v, yielding

$$(\lambda_v)_{\mathrm{ML}} = \arg\min_{\lambda} P(\mathbf{O}^v|\lambda).$$

Each model is estimated *separately* using the correspondingly labeled training observation sequence(s). The resultant models, however, need not be the optimal solution for minimizing the probability of recognition error.

An alternative design criterion that aims at maximizing the "discrimination" of each model (i.e., the ability to distinguish between observation sequences generated by the correct word model and those generated by alternative models) is the maximum mutual information (MMI) criterion [26]. The mutual information between an observation sequence $\mathbf{O}^v$ and the word v, parameterized by $\Lambda = \{\lambda_v\}$, $v = 1, 2, \ldots, V$, is

$$I_\Lambda(\mathbf{O}^v, v) = \log \frac{P(\mathbf{O}^v, v|\Lambda)}{P_\Lambda(\mathbf{O}^v)P(v)}. \tag{6.88}$$

Since

$$P(\mathbf{O}^v, v|\Lambda)/P(v) = P(\mathbf{O}^v|\lambda_v),$$

$$I_\Lambda(\mathbf{O}^v, v) = \log\, P(\mathbf{O}^v|\lambda_v) - \log \sum_{w=1}^{V} P(\mathbf{O}^v|\lambda_w, w)P(w). \tag{6.89}$$

The MMI criterion is to find the entire model set Λ such that the mutual information is maximized,

$$(\Lambda)_{\text{MMI}} = \max_{\Lambda}\left\{\sum_{v=1}^{V} I_\Lambda(\mathbf{O}^v, v)\right\}. \tag{6.90}$$

The MMI criterion is obviously different from the ML criterion. Both are minimum cross-entropy approaches. In the ML approach, an HMM for the distribution of the *data given the word* is matched to the empirical distribution. In the MMI approach, a model for the distribution of the *word given the data* is matched to the empirical distribution. This explains the merit of the MMI approach. The optimization procedure for the MMI approach involves the entire model parameter set Λ even if only one labeled training sequence $\mathbf{O}^v$ is used. The ML criterion addresses the likelihood $P(\mathbf{O}^v|\lambda_v)$ alone, while the MMI criterion compares the likelihood $P(\mathbf{O}^v|\lambda_v)$ against the "probability background" $P_\Lambda(\mathbf{O}^v)$ and attempts to maximize the difference. However, $(\Lambda)_{\text{MMI}}$ is not as straightforward to obtain as $(\Lambda)_{\text{ML}}$. One often has to use general optimization procedures like the descent algorithms to solve Eq. (6.90). Such optimization procedures often lead to numerical problems in implementation.

6.11 COMPARISONS OF HMMS

An interesting question associated with HMMs is the following: Given two HMMs, λ_1 and λ_2, what is a reasonable measure of the similarity of the two models ([29])? Consider the case of two models

$$\lambda_1 = (A_1, B_1, \pi_1) \qquad \lambda_2 = (A_2, B_2, \pi_2)$$

with

$$A_1 = \begin{bmatrix} p & 1-p \\ 1-p & p \end{bmatrix} \quad B_1 = \begin{bmatrix} q & 1-q \\ 1-q & q \end{bmatrix} \quad \pi_1 = [1/2 \;\; 1/2]$$

and

$$A_2 = \begin{bmatrix} r & 1-r \\ 1-r & r \end{bmatrix} \quad B_2 = \begin{bmatrix} s & 1-s \\ 1-s & s \end{bmatrix} \quad \pi_2 = [1/2 \;\; 1/2].$$

For λ_1 to be equivalent to λ_2, in the sense of having the same statistical properties for the observation symbols, i.e., $E[\mathbf{o}_t = \mathbf{v}_k|\lambda_1] = E[\mathbf{o}_t = \mathbf{v}_k|\lambda_2]$, for all $\mathbf{v}_k$, we require

$$pq + (1-p)(1-q) = rs + (1-r)(1-s)$$

or, by solving for s, we get

$$s = \frac{p + q - 2pq}{1 - 2r}.$$

By choosing (arbitrarily) $p = 0.6, q = 0.7, r = 0.2$, we get $s = 13/30 \approx 0.433$. Thus, even when the two models, λ_1 and λ_2, look ostensibly very different (i.e., A_1 is very different from A_2 and B_1 is very different from B_2), statistical equivalence of the models can occur.

We can generalize [29] the concept of mode' distance (dissimilarity) by defining a distance measure $D(\lambda_1, \lambda_2)$, between two Markov models, λ_1 and λ_2, as

$$D(\lambda_1, \lambda_2) = \frac{1}{T}\left[\log P(\mathbf{O}^{(2)}|\lambda_1) - \log P(\mathbf{O}^{(2)}|\lambda_2)\right] \tag{6.91}$$

where $\mathbf{O}^{(2)} = (\mathbf{o}_1\,\mathbf{o}_2\,\mathbf{o}_3 \ldots \mathbf{o}_T)$ is a sequence of observations generated by model λ_2. Basically, Eq. (6.91) is a measure of how well model λ_1 matches observations generated by model λ_2, relative to how well model λ_2 matches observations generated by itself. Several interpretations of Eq. (6.91) exist in terms of cross-entropy, or divergence, or discrimination information [29].

One of the problems with the distance measure of Eq. (6.91) is that it is nonsymmetric. Hence a natural expression of this measure is the symmetrized version, namely

$$D_s(\lambda_1, \lambda_2) = \frac{D(\lambda_1 \lambda_2) + D(\lambda_2, \lambda_1)}{2}. \tag{6.92}$$

6.12 IMPLEMENTATION ISSUES FOR HMMS

The discussion in the previous sections has dealt primarily with the theory of HMMs and several variations on the form of the model. In this section we deal with several practical implementation issues, including scaling, multiple observation sequences, initial parameter estimates, missing data, and choice of model size and type. For some of these implementation issues we can prescribe exact analytical solutions; for other issues we can provide only some seat-of-the-pants experience gained from working with HMMs.

6.12.1 Scaling

To understand why scaling ([18,23]) is required for implementing the reestimation procedure of HMMs, consider the definition of $\alpha_t(i)$ of Eq. (6.18). It can be seen that $\alpha_t(i)$ consists of the sum of a large number of terms, each of the form

$$\left(\prod_{s=1}^{t-1} a_{q_s q_{s+1}} \prod_{s=1}^{t} b_{q_s}(\mathbf{o}_s)\right)$$

with $q_t = i$ and b is a discrete probability as defined by Eq. (6.8). Since each a and b term is less than 1 (generally significantly less than 1), it can be seen that as t starts to get big (e.g., 10 or more), each term of $\alpha_t(i)$ starts to head exponentially to zero. For sufficiently large t (e.g., 100 or more) the dynamic range of the $\alpha_t(i)$ computation will exceed the precision range of essentially any machine (even in double precision). Hence the only reasonable way to perform the computation is to incorporate a scaling procedure.

The basic scaling procedure multiplies $\alpha_t(i)$ by a scaling coefficient that is independent of i (i.e., it depends only on t), with the goal of keeping the scaled $\alpha_t(i)$ within the dynamic range of the computer for $1 \leq t \leq T$. A similar scaling is done to the $\beta_t(i)$ coefficients (since these also tend to zero exponentially fast) and then, at the end of the computation, the scaling coefficients are canceled out exactly.

To understand this scaling procedure better, consider the reestimation formula for the state-transition coefficients a_{ij}. If we write the reestimation formula (Eq. (6.40b)) directly in terms of the forward and backward variables, we get

$$\bar{a}_{ij} = \frac{\sum_{t=1}^{T-1} \alpha_t(i) a_{ij} b_j(\mathbf{o}_{t+1}) \beta_{t+1}(j)}{\sum_{t=1}^{T} \sum_{j=1}^{N} \alpha_t(i) a_{ij} b_j(\mathbf{o}_{t+1}) \beta_{t+1}(j)}. \tag{6.93}$$

Consider the computation of $\alpha_t(i)$. We use the notation $\alpha_t(i)$ to denote the unscaled αs, $\hat{\alpha}_t(i)$ to denote the scaled (and iterated) αs, and $\widehat{\hat{\alpha}}_t(i)$ to denote the local version of α before scaling. Initially, for $t = 1$, we compute $\alpha_1(i)$ according to Eq. (6.19) and set $\widehat{\hat{\alpha}}_1(i) = \alpha_1(i)$, with $c_1 = \frac{1}{\sum_{i=1}^{N} \alpha_1(i)}$ and $\hat{\alpha}_1(i) = c_1 \alpha_1(i)$. For each t, $2 \leq t \leq T$, we first compute $\widehat{\hat{\alpha}}_t(i)$ according to the induction formula (Eq. (6.20)), in terms of the previously scaled $\hat{\alpha}_t(i)$; that is,

$$\widehat{\hat{\alpha}}_t(i) = \sum_{j=1}^{N} \hat{\alpha}_{t-1}(j) a_{ji} b_i(\mathbf{o}_t). \tag{6.94a}$$

We determine the scaling coefficient c_t as

$$c_t = \frac{1}{\sum_{i=1}^{N} \widehat{\hat{\alpha}}_t(i)} \tag{6.94b}$$

giving

$$\hat{\alpha}_t(i) = c_t \widehat{\hat{\alpha}}_t(i) \tag{6.94c}$$

From Eq. (6.94a–c) we can write the scaled $\hat{\alpha}_t(i)$ as $c_t \widehat{\hat{\alpha}}_t(i)$ or

$$\hat{\alpha}_t(i) = \frac{\sum_{j=1}^{N} \hat{\alpha}_{t-1}(j) a_{ji} b_i(\mathbf{o}_t)}{\sum_{i=1}^{N} \sum_{j=1}^{N} \hat{\alpha}_{t-1}(j) a_{ji} b_i(\mathbf{o}_t)}. \tag{6.95}$$

By induction we can write $\hat{\alpha}_{t-1}(j)$ as

$$\hat{\alpha}_{t-1}(j) = \left(\prod_{\tau=1}^{t-1} c_\tau\right)\alpha_{t-1}(j). \tag{6.96a}$$

Thus we can write $\hat{\alpha}_t(i)$ as

$$\hat{\alpha}_t(i) = \frac{\sum_{j=1}^{N} \alpha_{t-1}(j)\left(\prod_{\tau=1}^{t-1} c_\tau\right) a_{ji} b_i(\mathbf{o}_t)}{\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_{t-1}(j)\left(\prod_{\tau=1}^{t-1} c_\tau\right) a_{ji} b_i(\mathbf{o}_t)} = \frac{\alpha_t(i)}{\sum_{i=1}^{N} \alpha_t(i)} \tag{6.96b}$$

that is, each $\alpha_t(i)$ is effectively scaled by the sum over all states of $\alpha_t(i)$.

Next we compute the $\beta_t(i)$ terms from the backward recursion. The only difference here is that we use the *same* scale factors for each time t for the betas as was used for the alphas. Hence the scaled βs are of the form

$$\hat{\beta}_t(i) = c_t \beta_t(i). \tag{6.97}$$

Because each scale factor effectively restores the magnitude of the α terms to 1, and because the magnitudes of the α and β terms are comparable, using the same scaling factors on the βs as was used on the αs is an effective way to keep the computation within reasonable bounds. Furthermore, in terms of the scaled variables, we see that the reestimation Eq. (6.93) becomes

$$\bar{a}_{ij} = \frac{\sum_{t=1}^{T-1} \hat{\alpha}_t(i) a_{ij} b_j(\mathbf{o}_{t+1}) \hat{\beta}_{t+1}(j)}{\sum_{t=1}^{T-1}\sum_{j=1}^{N} \hat{\alpha}_t(i) a_{ij} b_j(\mathbf{o}_{t+1}) \hat{\beta}_{t+1}(j)} \tag{6.98}$$

but each $\hat{\alpha}_t(i)$ can be written as

$$\hat{\alpha}_t(i) = \left[\prod_{s=1}^{t} c_s\right]\alpha_t(i) = C_t \alpha_t(i) \tag{6.99}$$

and each $\hat{\beta}_{t+1}(j)$ can be written as

$$\hat{\beta}_{t+1}(j) = \left[\prod_{s=t+1}^{T} c_s\right]\beta_{t+1}(j) = D_{t+1}\beta_{t+1}(j). \tag{6.100}$$

Thus Eq. (6.98) can be written as

$$\overline{a_{ij}} = \frac{\displaystyle\sum_{t=1}^{T-1} C_t \alpha_t(i) a_{ij} b_j(\mathbf{o}_{t+1}) D_{t+1} \beta_{t+1}(j)}{\displaystyle\sum_{t=1}^{T-1} \sum_{j=1}^{N} C_t \alpha_t(i) a_{ij} b_j(\mathbf{o}_{t+1}) D_{t+1}\, \beta_{t+1}(j)}. \tag{6.101}$$

Finally the term $C_t D_{t+1}$ can be seen to be of the form

$$C_t D_{t+1} = \prod_{s=1}^{t} c_s \prod_{s=t+1}^{T} c_s = \prod_{s=1}^{T} c_s = C_T \tag{6.102}$$

independent of t. Hence the terms $C_t D_{t+1}$ cancel out of both the numerator and denominator of Eq. (6.101) and the exact reestimation equation is therefore realized.

It should be obvious that the above scaling procedure applies equally well to reestimation of the π or B coefficients. It should also be obvious that the scaling procedure of Eq. (6.95) need not be applied at every time instant t, but can be performed whenever desired, or whenever necessary (e.g., to prevent underflow). If scaling is not performed at some instant t, the scaling coefficients c_t are set to 1 at that time, and all the conditions discussed above are then met.

The only real change to the HMM procedure because of scaling is the procedure for computing $P(\mathbf{O}|\lambda)$. We cannot merely sum up the $\hat{\alpha}_T(i)$ terms, because these are scaled already. However, we can use the property that

$$\prod_{t=1}^{T} c_t \sum_{i=1}^{N} \alpha_T(i) = C_T \sum_{i=1}^{N} \alpha_T(i) = 1. \tag{6.103}$$

Thus we have

$$\prod_{t=1}^{T} c_t \cdot P(\mathbf{O}|\lambda) = 1 \tag{6.104}$$

or

$$P(\mathbf{O}|\lambda) = \frac{1}{\displaystyle\prod_{t=1}^{T} c_t} \tag{6.105}$$

or

$$\log\,[P(\mathbf{O}|\lambda)] = \sum_{t=1}^{T} \log\, c_t. \tag{6.106}$$

Thus the log of P can be computed, but not P, since it would be out of the dynamic range of the machine anyway.

Finally we note that when using the Viterbi algorithm to give the maximum likelihood state sequence, no scaling is required if we use logarithms as discussed in the alternate Viterbi implementation.

6.12.2 Multiple Observation Sequences

In Section 6.5 we discussed a form of HMM called the left-right or Bakis model, in which the state proceeds from state 1 at $t = 1$ to state N at $t = T$ in a sequential manner (recall the model of Figure 6.8(b)). We have already discussed how a left-right model imposes constraints on the state-transition matrix, and the initial state probabilities Eqs. (6.45)–(6.48). However, the major problem with left-right models is that one cannot use a single observation sequence to train the model (i.e., for reestimation of model parameters). This is because the transient nature of the states within the model allows only a small number of observations for any state (until a transition is made to a successor state). Hence, to have sufficient data to make reliable estimates of all model parameters, one has to use multiple observation sequences ([18]).

The modification of the reestimation procedure is straightforward and is as follows. We denote the set of K observation sequences as

$$\mathbf{O} = [\mathbf{O}^{(1)}, \mathbf{O}^{(2)}, \ldots, \mathbf{O}^{(K)}] \tag{6.107}$$

where $\mathbf{O}^{(k)} = (\mathbf{o}_1^{(k)} \mathbf{o}_2^{(k)} \ldots \mathbf{o}_{T_k}^{(k)})$ is the kth observation sequence. We assume each observation sequence is independent of every other observation sequence, and our goal is to adjust the parameters of the model λ to maximize

$$P(\mathbf{O}|\lambda) = \prod_{k=1}^{K} P(\mathbf{O}^{(k)}|\lambda) \tag{6.108}$$

$$= \prod_{k=1}^{K} P_k. \tag{6.109}$$

Since the reestimation formulas are based on frequencies of occurrence of various events, the reestimation formulas for multiple observation sequences are modified by adding together the individual frequencies of occurrence for each sequence. Thus the modified reestimation formulas for $\bar{a}_{ij}$ and $\overline{b_j}(\ell)$ are

$$\bar{a}_{ij} = \frac{\displaystyle\sum_{k=1}^{K} \frac{1}{P_k} \sum_{t=1}^{T_k-1} \alpha_t^k(i) a_{ij} b_j(\mathbf{o}_{t+1}^{(k)}) \beta_{t+1}^k(j)}{\displaystyle\sum_{k=1}^{K} \frac{1}{P_k} \sum_{t=1}^{T_k-1} \alpha_t^k(i) \beta_t^k(i)} \tag{6.110}$$

and

$$\overline{b_j}(\ell) = \frac{\displaystyle\sum_{k=1}^{K} \frac{1}{P_k} \sum_{\substack{t=1 \\ \text{s.t. } O_t = v_\ell}}^{T_k-1} \alpha_t^k(i) \beta_t^k(i)}{\displaystyle\sum_{k=1}^{K} \frac{1}{P_k} \sum_{t=1}^{T_k-1} \alpha_t^k(i) \beta_t^k(i)} \tag{6.111}$$

and π_i is not reestimated since $\pi_1 = 1, \pi_i = 0, i \neq 1$.

The proper scaling of Eqs. (6.110)–(6.111) is now straightforward since each observation sequence has its own scaling factor. The key idea is to remove the scaling factor from each term before summing. This can be accomplished by writing the reestimation equations in terms of the scaled variables, i.e.,

$$\bar{a}_{ij} = \frac{\sum_{k=1}^{K} \frac{1}{P_k} \sum_{t=1}^{T_k-1} \hat{\alpha}_t^k(i) a_{ij} b_j(\mathbf{o}_{t+1}^k) \hat{\beta}_{t+1}^k(j)}{\sum_{k=1}^{K} \frac{1}{P_k} \sum_{t=1}^{T_k-1} \hat{\alpha}_t^k(i) \hat{\beta}_t^k(i)}. \tag{6.112}$$

In this manner, for each sequence $\mathbf{O}^{(k)}$, the same scale factors will appear in each term of the sum over t as appears in the P_k term, and hence will cancel exactly. Thus using the scaled values of the αs and βs results in an unscaled $\bar{a}_{ij}$. A similar result is obtained for the $\bar{b}_j(\ell)$ term.

6.12.3 Initial Estimates of HMM Parameters

In theory, the reestimation equations should give values of the HMM parameters that correspond to a local maximum of the likelihood function. A key question is, therefore, How do we choose initial estimates of the HMM parameters so that the local maximum is equal to or as close as possible to the global maximum of the likelihood function?

Basically there is no simple or straightforward answer. Instead, experience has shown that either random (subject to the stochastic and the nonzero value constraints) or uniform initial estimates of the π and A parameters are adequate for giving useful reestimates of these parameters in almost all cases. However, for the B parameters, experience has shown that good initial estimates are helpful in the discrete symbol case and are essential (when dealing with multiple mixtures) in the continuous-distribution case. Such initial estimates can be obtained in a number of ways; these include (a) manual segmentation of the observation sequence(s) into states and averaging of observations within states, (b) maximum likelihood segmentation of observations and averaging, and (c) segmental k-means segmentation with clustering, etc. We discuss such segmentation techniques later in this chapter.

6.12.4 Effects of Insufficient Training Data

Another problem associated with training HMM parameters via reestimation methods is that the observation sequence used for training is, of necessity, finite ([30]). Thus there is always an inadequate number of occurrences of low-probability events (e.g., symbol occurrences within states) to give good estimates of the model parameters. By way of example, consider the case of a discrete observation HMM. Recall that the reestimation transformation of $\bar{b}_j(k)$, Eq. (6.40c), requires a count of the expected number of times in state j and observing symbol $\mathbf{v}_k$ simultaneously. If the training sequence is so small that it does not have any occurrences of this event (i.e., $q_t = j$ and $\mathbf{o}_t = \mathbf{v}_k$), $b_j(k) = 0$ and will stay 0 after reestimation. The resultant model would produce a zero probability result for any observation sequence that actually includes ($\mathbf{o}_t = \mathbf{v}_k$ and $q_t = j$). Such a singular outcome is obviously a consequence of the unreliable estimate that $b_j(k) = 0$ due to the insufficiency of the training set.

One solution to this problem is to increase the size of the training observation set. Often this is impractical. A second possible solution is to reduce the size of the model (e.g., number of states, number of symbols per state). Although this is always possible, often there are physical reasons why a given model is used, and therefore the model size cannot be changed. A third possible solution is to seek unconventional statistical estimation algorithms that can somehow enhance the reliability of the parameter estimates even based on limited training data. Deleted interpolation and parameter thresholding are two such alternatives. Since deleted interpolation is considered more an enhanced parameter estimation method, we discuss that subject in the next section.

The simplest way to handle the effects of insufficient training data is to add extra threshold constraints to the model parameters to ensure that no model parameter estimate falls below a specified level [18]. Thus, for example, we might specify the numeric floor, for a discrete symbol model, that

$$b_j(k) = \begin{cases} b_j(k), & \text{if } b_j(k) \geq \delta_b \\ \delta_b, & \text{otherwise} \end{cases} \tag{6.113a}$$

or, for a continuous distribution model, that

$$U_{jk}(r,\ r) = \begin{cases} U_{jk}(r,r), & \text{if } U_{jk}(r,r) \geq \delta_u \\ \delta_u, & \text{otherwise} \end{cases} . \tag{6.113b}$$

When the numeric floor is invoked in the reestimation equations, all remaining parameters need to be rescaled so that the densities obey the required stochastic constraints. Such postprocessor techniques are thus considered implementational measures to combat the insufficient data problem and have been applied with good success to several problems in speech processing. The method of parameter thresholding has a justification from a Bayes statistics point of view. It can be shown that Eq. (6.113b) is, in fact, a maximum a posteriori (MAP) estimate of the variance under the assumption that the parameter prior $P(U_{jk}(r,r))$ is an informative one with uniform distribution and $(U_{jk}(r,r))_{\min} = \delta_u$ [31]. (See Section 6.13.)

6.12.5 Choice of Model

The remaining issues in implementing HMMs are the choice of type of model (ergodic or left-right or some other form), choice of model size (number of states), and choice of observation symbols (discrete or continuous, single or multimixture, choice of observation parameters). Unfortunately, there is no simple, theoretically correct way of making such choices. These choices must be made depending on the signal being modeled. With these comments, we end our discussion of the theoretical aspects of hidden Markov models and proceed to a discussion of how such models have been applied to the isolated word-recognition problem.

6.13 IMPROVING THE EFFECTIVENESS OF MODEL ESTIMATES

We discuss three methods that have been shown to be able to enhance the effectiveness of HMM model estimates for speech recognition. These are (1) deleted interpolation, (2) Bayesian adaptation, and (3) corrective training. The first two methods are motivated by the problem of insufficient data, while the last method has the unique objective of trying to reduce recognition errors directly.

6.13.1 Deleted Interpolation

When training data are insufficient, reliable and robust determination of HMM parameters cannot be accomplished. The HMM obtained by the Baum-Welch reestimation method, based on the maximum likelihood criterion, may be adequate in characterizing the training data, but for new data the match may be quite poor. One parameter estimation method that aims to improve the model reliability is the method of "deleted interpolation."

The concept involves combining two (or more) separately trained models, one of which is more reliably trained than the other. A scenario in which this can happen is the case when we use tied states which forces "different" states to share an identical statistical characterization, effectively reducing the number of parameters in the model. A model with tied states is often more robust than a model without tied states when trained on the same amount of data. But a model with tied states is also less precise than a model without tied states if the training data are sufficient. Therefore, the idea of combining the two models is to allow us to fall back to the more reliable model when the supposedly more precise model is, in fact, unreliable. A similar scenario occurs when context-independent (more robust) and context-dependent (more precise) phone models are used in large vocabulary recognition (see Chapter 8).

Let the two models be defined by the parameter sets $\lambda = (A, B, \pi)$ and $\lambda' = (A', B', \pi')$, respectively. The interpolated model, $\tilde{\lambda} = (\tilde{A}, \tilde{B}, \tilde{\pi})$, is obtained as

$$\tilde{\lambda} = \epsilon\lambda + (1 - \epsilon)\lambda' \tag{6.114}$$

where ϵ represents the weighting of the parameters of the "full" model (with more detailed characterization of the observations) and $(1 - \epsilon)$ represents the weighting of the parameters of the reduced, but more reliable, model. A key issue is the determination of the optimal value of ϵ, which is a strong function of the amount of training data. This is easy to see because as the amount of training data gets large, λ becomes more reliable and we expect ϵ to tend to 1.0. Similarly, for small amounts of training data, λ is unreliable and we expect ϵ to tend to 0.0 so as to fall back to the more reliable model λ'.

The solution to the determination of an optimal value for ϵ was provided by Jelinek and Mercer [30], who showed how the optimal ϵ could be estimated using the forward-backward algorithm by interpreting Eq. (6.114) as an expanded HMM of the type shown in Figure 6.12. Figure 6.12a shows the part of the state-transition structure related to the state S. Using Figure 6.12b, we can interpret the interpolated model of Eq. (6.114) as an

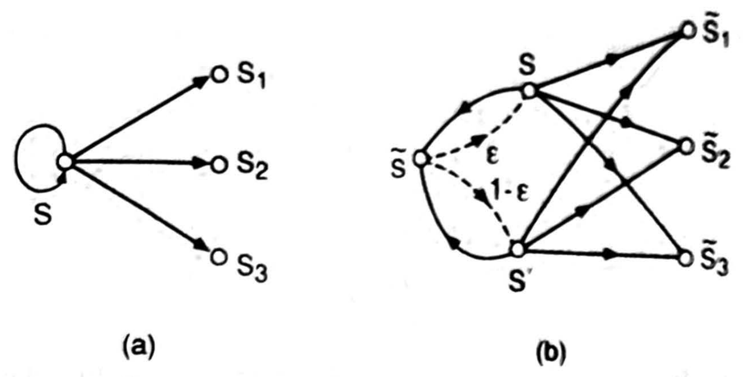


Figure 6.12 Example of how the process of deleted interpolation can be represented using a state diagram.

expanded HMM in which each state is replaced by three states. The null transitions from the expanded state $\tilde{S}$ to S and S' have transition probabilities ϵ and $1-\epsilon$, respectively. The transitions out of S are characterized by those of λ while those out of S' are associated with those of λ'.

The expanded HMM interpretation suggests that the parameter ϵ can be optimally determined by the usual forward-backward algorithm. However, since the interpolation is designed to better predict unseen (future) data, rather than to account for the training data, determination of ϵ must be based on data that was not used in obtaining either of the two models, λ and λ'. A key idea of deleted interpolation is thus to partition the training data $\mathcal{T}$ into two disjoint sets; that is, $\mathcal{T} = \mathcal{T}_1 \cup \mathcal{T}_2$. For example, one might consider a partition of the training set such that $\mathcal{T}_1$ is 90 percent of $\mathcal{T}$ and $\mathcal{T}_2$ is the remaining 10 percent of $\mathcal{T}$. Training set $\mathcal{T}_1$ is first used to train λ and λ'. Training set $\mathcal{T}_2$ is then used to give an estimate of ϵ, assuming λ and λ' are fixed. There are obviously a large number of ways in which such a partitioning can be accomplished, but one particularly simple one is to cycle $\mathcal{T}_2$ through the data. That is, the first partition uses the last 10 percent of the data as $\mathcal{T}_2$, the second partition uses the next-to-last 10 percent of the data as $\mathcal{T}_2$, etc. An interpretation of deleted interpolation is straightforward. If the unseen data fits the more elaborate model λ well (thus validating the reliability of λ), the forward-backward algorithm would give a value of ϵ which is close to 1. Otherwise, the forward-backward algorithm gives a small value of ϵ, indicating that the more reliable model λ' is a better characterization of the new data than λ.

The technique of deleted interpolation has been successfully applied to a number of problems in speech recognition, including the estimation of trigram word probabilities for language models [13], and the estimation of HMM output probabilities for trigram phone models (to be discussed in Chapter 8 of this book).

6.13.2 Bayesian Adaptation

Another insufficient data situation occurs when we attempt to estimate a speaker-dependent model based on a limited amount of speaker-specific training data. An approach to this problem is through speaker adaptation, in which a speaker-independent model, obtained by reliable training, is adapted to the particular talker using speaker-specific training data [31].

Speaker adaptation can be accomplished based on a Bayesian framework. Consider the HMM probability measure $P(\mathbf{O}|\lambda)$. If the HMM parameter λ is assumed to be fixed but unknown, the maximum likelihood (ML) estimate for λ, given the training sequence $\mathbf{O}$, is obtained by solving the likelihood equation, i.e.

$$\frac{\partial}{\partial\lambda}P(\mathbf{O}|\lambda) = 0. \tag{6.115}$$

(Normally, the Baum-Welch reestimation algorithm is used to obtain certain stationary point solutions instead of directly solving for Eq. (6.115).) If λ is assumed random with a priori distribution $P_0(\lambda)$, then the maximum a posteriori (MAP) estimate for λ is obtained by solving

$$\frac{\partial}{\partial\lambda}P(\lambda|\mathbf{O}) = 0 \tag{6.116}$$

for the given training sequence $\mathbf{O}$. Using the Bayes theorem, we rewrite $P(\lambda|\mathbf{O})$ as

$$P(\lambda|\mathbf{O}) = \frac{P(\mathbf{O}|\lambda)P_0(\lambda)}{P(\mathbf{O})}. \tag{6.117}$$

The influence of the parameter prior $P_0(\lambda)$ in the solution process thus becomes explicit. Note that if the distribution is correctly chosen, the MAP solution attains minimum Bayes risk.

The parameter prior distribution characterizes the statistics of the parameters of interest before any measurement is made. If the prior distribution indicates no preference as to what the parameter values are likely to be, then the prior is called a noninformative prior (which is essentially constant for the entire parameter space). In this case, the MAP estimate obtained by solving Eq. (6.116) is identical to the ML estimate of Eq. (6.115). If we do have prior knowledge about the parameter values of the model, the incorporation of such prior knowledge in the form of a prior distribution would become important in the MAP estimate for minimum Bayes risk. This type of prior is often called an informative prior. Intuitively, if we know what the parameter values are likely to be before observations are made, we may be able to make good use of the data, which may be limited, to obtain a good model estimate. If this is true, the questions remaining are how to derive the informative prior and how to use it in obtaining the MAP estimate.

For mathematical tractability, conjugate priors are often used in Bayesian adaptation. A conjugate prior for a random vector is defined as the prior distribution for the parameters of the probability density function of the random vector, such that the posterior distribution $P(\lambda|\mathbf{O})$ and the prior distribution $P(\lambda)$ belong to the same distribution family for any sample observations $\mathbf{O}$. For example, it is well known that the conjugate prior for the mean of a Gaussian density is also a Gaussian density. In the following, we therefore discuss only the use of conjugate priors. We also discuss only the case of Bayesian adaptation of the Gaussian mean as it is sufficient to demonstrate the idea of Bayesian adaptation in dealing with small training set problems.

Let us focus on a Gaussian mixture component $\mathcal{N}(\mu, \sigma^2)$ in a mixture density HMM. We use one-dimensional observations for simplicity. Assume the mean μ is random with prior distribution $P_0(\mu)$ and the variance σ^2 is known and fixed. It can be shown that the conjugate prior for μ is also Gaussian; that is, if we assume $P_0(\mu)$ to be the conjugate prior of μ, then $P_0(\mu)$ is Gaussian. Thus, let us denote the mean and variance of the prior for μ by ρ and τ^2, respectively. The MAP estimate for the mean parameter μ in Bayesian adaptation, from a set of n training observations, is given by

$$\hat{\mu}_{\text{MAP}} = \frac{n\tau^2}{\sigma^2 + n\tau^2}\bar{o} + \frac{\sigma^2}{\sigma^2 + n\tau^2}\rho \tag{6.118}$$

where $\bar{o}$ is the sample mean of the n training data. The interpretation of Eq. (6.118) is as follows. If there are no training data presented, $n = 0$ and the best estimate of μ is simply the mean ρ of the prior distribution of the μ parameter. When training data are collected and used, the MAP estimate becomes a weighted average of the prior mean ρ and the sample mean of the presented observations, $\bar{o}$. Ultimately, $n \to \infty$ and the best estimate of μ is, as expected, the sample mean $\bar{o}$. It should also be noted that if the prior variance τ^2 is much larger than σ^2/n, the MAP estimate in Eq. (6.118) is essentially the ML estimate, $\bar{o}$, which corresponds to the case of using noninformative priors.

A key question is, How do we determine ρ and τ^2? In practice, these prior parameters have to be estimated from a collection of speaker-dependent (or multispeaker) models, or from a speaker-independent model with mixture distributions in each state. For example, ρ and τ^2 can be estimated by

$$\rho = \sum_{m=1}^{M} c_m \rho_m \tag{6.119a}$$

and

$$\tau^2 = \sum_{m=1}^{M} c_m(\rho_m - \rho)^2 \tag{6.119b}$$

where c_m is the weight assigned to the mth model (or the mth mixture component in the corresponding state of a mixture density speaker-independent HMM) and ρ_m is the mean of the corresponding mth model (or mixture component). When using a speaker-independent Gaussian mixture HMM, the weight c_m is basically the mixture gain for the mth mixture component, and the estimates of Eq. (6.119) are the ML estimates of the mean and variance parameters of μ before any speaker-specific training data are observed.

The concept of Bayesian adaptation based on conjugate priors can be applied to other parameters as well. The adaptation method can be shown to provide good parameter estimates even when the number of speaker-specific training tokens is extremely limited. Experiments have shown that large improvements in recognition accuracy are obtained with the Bayesian adaptation method, compared to direct training, particularly when only a small number of training tokens are available [30].

6.13.3 Corrective Training

In statistical pattern recognition, the minimum Bayes' risk is the theoretical recognizer performance bound, conditioned on the exact knowledge of both the class prior $P(v)$ and the conditional distributions $P(\mathbf{O}|v)$. When both distributions are not known exactly, and the classifier needs to be designed based on a finite training set, there are several ways to try to reduce the error rate. One method is based on the theoretical link between discriminant analysis and distribution estimation [32]. The idea is to design a classifier (discriminant function) such that the minimum classification error rate is attained on the training set. In particular, we wish to design a classifier that uses estimates of $P(v)$ and $P(\mathbf{O}|v)$, and that achieves a minimum error rate for the training set in the same way a discriminant function is designed. The reason for using the HMM, $P(\mathbf{O}|\lambda_v)$, for modeling $P(\mathbf{O}|v)$, as opposed to other discriminant functions, is to exploit the strengths of the HMMs—consistency, flexibility and computational ease.

Bahl et al. [33] were the first to propose an error-correction strategy, which they named corrective training, to specifically deal with the misclassification problem. Their training algorithm was motivated by analogy with an error-correction training procedure for linear classifiers. In their proposed method, the observation distribution is of a discrete type, $B = [b_i(k)]$, where $b_i(k)$ is the probability of observing a vector quantization code index (acoustic label) k when the HMM source is in state i. Each $b_i(k)$ is obtained via the forward-backward algorithm as the weighted frequency of occurrence of the code index. The corrective training algorithm of Bahl et al. works as follows. First, use a labeled training set to estimate the parameters of the HMMs $\Lambda = \{\lambda_v\}$ with the forward-backward algorithm. For each utterance $\mathbf{O}$, labeled as v, for example, evaluate $P(\mathbf{O}|\lambda_v)$ for the correct class v and $P(\mathbf{O}|\lambda_w)$ for each incorrect class w. (The evaluation of likelihood for the incorrect classes need not be exhaustive.) For every utterance where $\log\ P(\mathbf{O}|\lambda_w) > \log\ P(\mathbf{O}|\lambda_v) - \Delta$, where Δ is a prescribed threshold, modify λ_v and λ_w according to the following mechanism: (1) Apply the forward-backward algorithm to obtain estimates $b_i'(k)$ and $b_i''(k)$, using the labeled utterance $\mathbf{O}$ only, for the correct class v and incorrect class w, respectively; (2) Modify the original $b_i(k)$ in λ_v to $b_i(k) + \gamma b_i'(k)$ and the $b_i(k)$ in λ_w to $b_i(k) - \gamma b_i''(k)$. When the state labels are tied for certain models, the above procedure is equivalent to replacing the original $b_i(k)$ by $b_i(k) + \gamma(b_i'(k) - b_i''(k))$. The prescribed adaptation parameter, γ, controls the "rate of convergence" and the threshold, Δ, defines the "near-miss" cases. This corrective training algorithm therefore focuses on those parts of the model that are most important for word discrimination, a clear difference from the maximum likelihood principle.

Bahl et al. reported that the corrective training procedure worked better (in isolated word-recognition tasks) than models obtained using the maximum mutual information or the conditional maximum likelihood criterion. The method, however, is primarily experimental.

Several other forms of discriminative training were also proposed by Katagiri et al. [34] together with a framework for the analysis of related training/learning ideas for minimizing recognition errors. The discriminative training method described in Sec. 5.6.3

can be applied to HMM training without difficulty. The corrective training algorithm of Bahl et al. can be shown to be just one possible choice for the minimization of a prescribed risk function.

6.14 MODEL CLUSTERING AND SPLITTING

One of the basic assumptions in statistical modeling is that the variability in the observations from an information source can be modeled by the assumed statistical distributions. For speech recognition, the source could be a single word, a subword unit like a phoneme, or a word sequence. Because of variability in the production (e.g., accents, speed of talking), or the processing (e.g., transmission distortion, noise), it is often expedient to consider using more than a single HMM to characterize the source. There are two motivations behind this multiple HMM approach. First, lumping together all the variability from inhomogeneous data sources leads to unnecessarily complex models, often yielding lower modeling accuracy. Second, some of the variability, or rather the inhomogeneity in the source data, may be known *a priori*, thus warranting separate modeling of the source data sets.

Several generalized clustering algorithms exist, such as the k-means clustering algorithm, the generalized Lloyd algorithm as is widely used in vector quantizer designs [35], and the greedy growing algorithm found in set partition or decision tree designs [36], all of which are suitable for the purpose of separating inconsistent training data so that each divided subgroup becomes more homogeneous and therefore is better modeled by a single HMM. The nearest-neighbor rule required in these clustering algorithms is simply to assign an observation sequence $\mathbf{O}$ to cluster i if

$$P(\mathbf{O}|\lambda_i) = \max_j P(\mathbf{O}|\lambda_j) \tag{6.120}$$

where λ_js denote the models of the clusters. Successful application of the model clustering algorithms to the speech-recognition problem, using the straightforward maximum likelihood criterion, has been reported.

An alternative to model clustering is to arbitrarily subdivide a given speech source into a large number of subclasses with specialized characteristics and then consider a generalized procedure for model merging based on source likelihood considerations. By way of example, for large vocabulary speech recognition we often try to build specialized units (context sensitive) for recognition. For example, we could consider building units that are a function of the sound immediately preceding the unit (left-context) and the sound immediately following the unit (right-context). There are on the order of 10,000 such units in English. Many of the units are functionally almost identical. The problem is how to determine which pairs of units should be merged (so that the number of model units is made more manageable and the variance of the parameter estimate is reduced). To get ideas, consider two distinct models, λ_a and λ_b, corresponding to training observation sets $\mathbf{O}_a$ and $\mathbf{O}_b$, and the merged model λ_{a+b}, corresponding to the merged observation sets $\{\mathbf{O}_a, \mathbf{O}_b\}$. We can then compute the change in entropy (i.e., loss of information) resulting from the merged model as

$$\begin{aligned}\Delta H_{ab} &= H_a + H_b - H_{a+b} \\ &= -P(\mathbf{O}_a|\lambda_a) \log P(\mathbf{O}_a|\lambda_a) - P(\mathbf{O}_b|\lambda_b) \log P(\mathbf{O}_b|\lambda_b) \\ &\quad + P(\{\mathbf{O}_a, \mathbf{O}_b\}|\lambda_{a+b}) \log P(\{\mathbf{O}_a, \mathbf{O}_b\}|\lambda_{a+b}).\end{aligned} \tag{6.121}$$

Whenever ΔH_{ab} is small enough, it means that the change in entropy resulting from merging the models will not affect system performance (at least on the training set) and the models can be merged. The question of how small is acceptable is dependent on specific applications. This model merging technique has been used successfully by Lee [37] to create a generalized set of triphone models for large vocabulary speech recognition.

6.15 HMM SYSTEM FOR ISOLATED WORD RECOGNITION

To illustrate the techniques discussed in this chapter, consider using HMMs to build an isolated word recognizer ([38]). Assume we have a vocabulary of V words to be recognized and that each word is to be modeled by a distinct HMM. Further assume, for simplicity of notation, that for each word in the vocabulary we have a training set of K utterances of the word (spoken by one or more talkers) where each utterance constitutes an observation sequence, of some appropriate representation of the (spectral and/or temporal) characteristics of the word. To do isolated word speech recognition, we must perform the following:

1. For each word v in the vocabulary, we must build an HMM λ_v—that is, we must estimate the model parameters (A, B, π) that optimize the likelihood of the training set observation vectors for the vth word.
2. For each unknown word to be recognized, the processing shown in Figure 6.13 must be carried out, namely, measurement of the observation sequence $\mathbf{O} = \{\mathbf{o}_1 \mathbf{o}_2 \dots \mathbf{o}_T\}$, via a feature analysis of the speech corresponding to the word; followed by calculation of model likelihoods for all possible models, $P(\mathbf{O}|\lambda_v)$, $1 \leq v \leq V$; followed by selection of the word whose model likelihood is highest—that is,

$$v^* = \arg \max_{1 \leq v \leq V} [P(\mathbf{O}|\lambda_v)]. \tag{6.122}$$

The probability computation step is generally performed using the Viterbi algorithm (i.e., the maximum likelihood path is used) and requires on the order of $V \cdot N^2 \cdot T$ computations. For modest vocabulary sizes, e.g., $V = 100$ words, with an $N = 5$ state model, and $T = 40$ observations for the unknown word, a total of 10^5 computations is required for recognition (where each computation is a multiply, and add, and a calculation of observation density, $b(\mathbf{o})$). Clearly this amount of computation is modest compared to the capabilities of most modern signal processor chips.