A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition

LAWRENCE R. RABINER, FELLOW, IEEE

Although initially introduced and studied in the late 1960s and early 1970s, statistical methods of Markov source or hidden Markov modeling have become increasingly popular in the last several years. There are two strong reasons why this has occurred. First the models are very rich in mathematical structure and hence can form the theoretical basis for use in a wide range of applications. Second the models, when applied properly, work very well in practice for several important applications. In this paper we attempt to carefully and methodically review the theoretical aspects of this type of statistical modeling and show how they have been applied to selected problems in machine recognition of speech.

I. Introduction

Real-world processes generally produce observable outputs which can be characterized as signals. The signals can be discrete in nature (e.g., characters from a finite alphabet, quantized vectors from a codebook, etc.), or continuous in nature (e.g., speech samples, temperature measurements, music, etc.). The signal source can be stationary (i.e., its statistical properties do not vary with time), or nonstationary (i.e., the signal properties vary over time). The signals can be pure (i.e., coming strictly from a single source), or can be corrupted from other signal sources (e.g., noise) or by transmission distortions, reverberation, etc.

A problem of fundamental interest is characterizing such real-world signals in terms of signal models. There are several reasons why one is interested in applying signal models. First of all, a signal model can provide the basis for a theoretical description of a signal processing system which can be used to process the signal so as to provide a desired output. For example if we are interested in enhancing a speech signal corrupted by noise and transmission distortion, we can use the signal model to design a system which will optimally remove the noise and undo the transmission distortion. A second reason why signal models are important is that they are potentially capable of letting us learn a great deal about the signal source (i.e., the real-world process which produced the signal) without having to have the source available. This property is especially important when the cost of getting signals from the actual source is high. In this case, with a good signal model, we can simulate the source and learn as much as possible via simulations. Finally, the most important reason why signal models are important is that they often work extremely well in practice, and enable us to realize important practical systems—e.g., prediction systems, recognition systems, identification systems, etc., in a very efficient manner.

These are several possible choices for what type of signal model is used for characterizing the properties of a given signal. Broadly one can dichotomize the types of signal models into the class of deterministic models, and the class of statistical models. Deterministic models generally exploit some known specific properties of the signal, e.g., that the signal is a sine wave, or a sum of exponentials, etc. In these cases, specification of the signal model is generally straightforward; all that is required is to determine (estimate) values of the parameters of the signal model (e.g., amplitude, frequency, phase of a sine wave, amplitudes and rates of exponentials, etc.). The second broad class of signal models is the set of statistical models in which one tries to characterize only the statistical properties of the signal. Examples of such statistical models include Gaussian processes, Poisson processes, Markov processes, and hidden Markov processes, among others. The underlying assumption of the statistical model is that the signal can be well characterized as a parametric random process, and that the parameters of the stochastic process can be determined (estimated) in a precise, well-defined manner.

For the applications of interest, namely speech processing, both deterministic and stochastic signal models have had good success. In this paper we will concern ourselves strictly with one type of stochastic signal model, namely the hidden Markov model (HMM). (These models are referred to as Markov sources or probabilistic functions of Markov chains in the communications literature.) We will first review the theory of Markov chains and then extend the ideas to the class of hidden Markov models using several simple examples. We will then focus our attention on the three fundamental problems[1] for HMM design, namely: the

Manuscript received January 15, 1988; revised October 4, 1988.
The author is with AT&T Bell Laboratories, Murray Hill, NJ 07974-2070, USA.
IEEE Log Number 8825949.

[1]The idea of characterizing the theoretical aspects of hidden Markov modeling in terms of solving three fundamental problems is due to Jack Ferguson of IDA (Institute for Defense Analysis) who introduced it in lectures and writing.

0018-9219/89/0200-0257$01.00 © 1989 IEEE

evaluation of the probability (or likelihood) of a sequence of observations given a specific HMM; the determination of a best sequence of model states; and the adjustment of model parameters so as to best account for the observed signal. We will show that once these three fundamental problems are solved, we can apply HMMs to selected problems in speech recognition.

Neither the theory of hidden Markov models nor its applications to speech recognition is new. The basic theory was published in a series of classic papers by Baum and his colleagues [1]–[5] in the late 1960s and early 1970s and was implemented for speech processing applications by Baker [6] at CMU, and by Jelinek and his colleagues at IBM [7]–[13] in the 1970s. However, widespread understanding and application of the theory of HMMs to speech processing has occurred only within the past several years. There are several reasons why this has been the case. First, the basic theory of hidden Markov models was published in mathematical journals which were not generally read by engineers working on problems in speech processing. The second reason was that the original applications of the theory to speech processing did not provide sufficient tutorial material for most readers to understand the theory and to be able to apply it to their own research. As a result, several tutorial papers were written which provided a sufficient level of detail for a number of research labs to begin work using HMMs in individual speech processing applications [14]–[19]. This tutorial is intended to provide an overview of the basic theory of HMMs (as originated by Baum and his colleagues), provide practical details on methods of implementation of the theory, and describe a couple of selected applications of the theory to distinct problems in speech recognition. The paper combines results from a number of original sources and hopefully provides a single source for acquiring the background required to pursue further this fascinating area of research.

The organization of this paper is as follows. In Section II we review the theory of discrete Markov chains and show how the concept of hidden states, where the observation is a probabilistic function of the state, can be used effectively. We illustrate the theory with two simple examples, namely coin-tossing, and the classic balls-in-urns system. In Section III we discuss the three fundamental problems of HMMs, and give several practical techniques for solving these problems. In Section IV we discuss the various types of HMMs that have been studied including ergodic as well as left-right models. In this section we also discuss the various model features including the form of the observation density function, the state duration density, and the optimization criterion for choosing optimal HMM parameter values. In Section V we discuss the issues that arise in implementing HMMs including the topics of scaling, initial parameter estimates, model size, model form, missing data, and multiple observation sequences. In Section VI we describe an isolated word speech recognizer, implemented with HMM ideas, and show how it performs as compared to alternative implementations. In Section VII we extend the ideas presented in Section VI to the problem of recognizing a string of spoken words based on concatenating individual HMMs of each word in the vocabulary. In Section VIII we briefly outline how the ideas of HMM have been applied to a large vocabulary speech recognizer, and in Section IX we summarize the ideas discussed throughout the paper.

II. Discrete Markov Processes[2]

Consider a system which may be described at any time as being in one of a set of N distinct states, $S_1, S_2, \cdots, S_N$, as illustrated in Fig. 1 (where $N = 5$ for simplicity). At regularly spaced discrete times, the system undergoes a change of state (possibly back to the same state) according to a set of probabilities associated with the state. We denote the time instants associated with state changes as $t = 1, 2, \cdots$, and we denote the actual state at time t as q_t. A full probabilistic description of the above system would, in general, require specification of the current state (at time t), as well as all the predecessor states. For the special case of a discrete, first order, Markov chain, this probabilistic description is truncated to just the current and the predecessor state, i.e.,

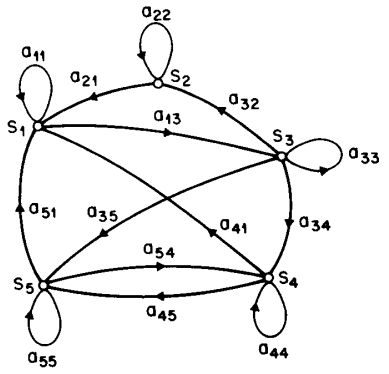

Fig. 1. A Markov chain with 5 states (labeled S_1 to S_5) with selected state transitions.

$$P[q_t = S_j | q_{t-1} = S_i, q_{t-2} = S_k, \cdots] = P[q_t = S_j | q_{t-1} = S_i]. \quad (1)$$

Furthermore we only consider those processes in which the right-hand side of (1) is independent of time, thereby leading to the set of state transition probabilities a_{ij} of the form

$$a_{ij} = P[q_t = S_j | q_{t-1} = S_i], \quad 1 \le i, j \le N \quad (2)$$

with the state transition coefficients having the properties

$$a_{ij} \ge 0 \quad (3a)$$

$$\sum_{j=1}^{N} a_{ij} = 1 \quad (3b)$$

since they obey standard stochastic constraints.

The above stochastic process could be called an observable Markov model since the output of the process is the set of states at each instant of time, where each state corresponds to a physical (observable) event. To set ideas, consider a simple 3-state Markov model of the weather. We assume that once a day (e.g., at noon), the weather is

[2]A good overview of discrete Markov processes is in [20, ch. 5].

observed as being one of the following:

State 1: rain or (snow)
State 2: cloudy
State 3: sunny.

We postulate that the weather on day t is characterized by a single one of the three states above, and that the matrix A of state transition probabilities is

$$A = \{a_{ij}\} = \begin{bmatrix} 0.4 & 0.3 & 0.3 \\ 0.2 & 0.6 & 0.2 \\ 0.1 & 0.1 & 0.8 \end{bmatrix}.$$

Given that the weather on day 1 ($t = 1$) is sunny (state 3), we can ask the question: What is the probability (according to the model) that the weather for the next 7 days will be "sun-sun-rain-rain-sun-cloudy-sun $\cdots$"? Stated more formally, we define the observation sequence O as $O = \{S_3, S_3, S_3, S_1, S_1, S_3, S_2, S_3\}$ corresponding to $t = 1, 2, \cdots, 8$, and we wish to determine the probability of O, given the model. This probability can be expressed (and evaluated) as

$$\begin{aligned} P(O|\text{Model}) &= P[S_3, S_3, S_3, S_1, S_1, S_3, S_2, S_3|\text{Model}] \\ &= P[S_3] \cdot P[S_3|S_3] \cdot P[S_3|S_3] \cdot P[S_1|S_3] \\ &\quad \cdot P[S_1|S_1] \cdot P[S_3|S_1] \cdot P[S_2|S_3] \cdot P[S_3|S_2] \\ &= \pi_3 \cdot a_{33} \cdot a_{33} \cdot a_{31} \cdot a_{11} \cdot a_{13} \cdot a_{32} \cdot a_{23} \\ &= 1 \cdot (0.8)(0.8)(0.1)(0.4)(0.3)(0.1)(0.2) \\ &= 1.536 \times 10^{-4} \end{aligned}$$

where we use the notation

$$\pi_i = P[q_1 = S_i], \quad 1 \le i \le N \tag{4}$$

to denote the initial state probabilities.

Another interesting question we can ask (and answer using the model) is: Given that the model is in a known state, what is the probability it stays in that state for exactly d days? This probability can be evaluated as the probability of the observation sequence

$$O = \{\underset{1}{S_i}, \underset{2}{S_i}, \underset{3}{S_i}, \cdots, \underset{d}{S_i}, \underset{d+1}{S_j} \neq S_i\},$$

given the model, which is

$$P(O|\text{Model}, q_1 = S_i) = (a_{ii})^{d-1}(1 - a_{ii}) = p_i(d). \tag{5}$$

The quantity $p_i(d)$ is the (discrete) probability density function of duration d in state i. This exponential duration density is characteristic of the state duration in a Markov chain. Based on $p_i(d)$, we can readily calculate the expected number of observations (duration) in a state, conditioned on starting in that state as

$$\bar{d}_i = \sum_{d=1}^{\infty} d p_i(d) \tag{6a}$$

$$= \sum_{d=1}^{\infty} d(a_{ii})^{d-1}(1 - a_{ii}) = \frac{1}{1 - a_{ii}}. \tag{6b}$$

Thus the expected number of consecutive days of sunny weather, according to the model, is $1/(0.2) = 5$; for cloudy it is 2.5; for rain it is 1.67.

A. Extension to Hidden Markov Models

So far we have considered Markov models in which each state corresponded to an observable (physical) event. This model is too restrictive to be applicable to many problems of interest. In this section we extend the concept of Markov models to include the case where the observation is a probabilistic function of the state—i.e., the resulting model (which is called a hidden Markov model) is a doubly embedded stochastic process with an underlying stochastic process that is *not* observable (it is hidden), but can only be observed through another set of stochastic processes that produce the sequence of observations. To fix ideas, consider the following model of some simple coin tossing experiments.

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 (or multiple coin) tossing experiment. The other person will not tell you anything about what he is doing exactly; 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; e.g., a typical observation sequence would be

$$\begin{aligned} \boldsymbol{O} &= O_1\, O_2\, O_3 \cdots O_T \\ &= \mathcal{H}\, \mathcal{H}\, \mathcal{T}\, \mathcal{T}\, \mathcal{T}\, \mathcal{H}\, \mathcal{T}\, \mathcal{T}\, \mathcal{H} \cdots \mathcal{H} \end{aligned}$$

where $\mathcal{H}$ stands for heads and $\mathcal{T}$ stands for tails.

Given the above scenario, the problem of interest is how do we build an HMM to explain (model) the observed sequence of heads and tails. The first problem one faces 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 2-state model where each state corresponds to a side of the coin (i.e., heads or tails). This model is depicted in Fig. 2(a).[3] 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 bias (i.e., the probability of, say, heads). Interestingly, an equivalent HMM to that of Fig. 2(a) would be a degenerate 1-state model, where the state corresponds to the single biased coin, and the unknown parameter is the bias of the coin.

A second form of HMM for explaining the observed sequence of coin toss outcome is given in Fig. 2(b). In this case there are 2 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 which 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 Fig. 2(c). This model corresponds to using 3 biased coins, and choosing from among the three, based on some probabilistic event.

[3]The model of Fig. 2(a) is a memoryless process and thus is a degenerate case of a Markov model.

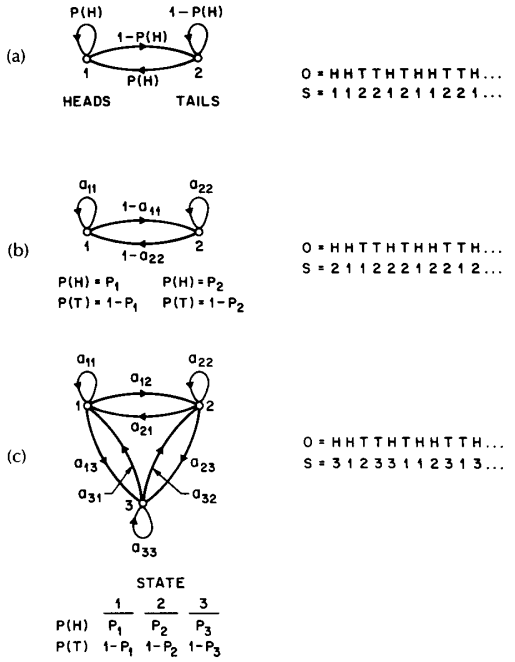

Fig. 2. Three possible Markov models which can account for the results of hidden coin tossing experiments. (a) 1-coin model. (b) 2-coins model. (c) 3-coins model.

Given the choice among the three models shown in Fig. 2 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 1-coin model of Fig. 2(a) has only 1 unknown parameter; the 2-coin model of Fig. 2(b) has 4 unknown parameters; and the 3-coin model of Fig. 2(c) has 9 unknown parameters. Thus, with the greater degrees of freedom, the larger HMMs would seem to inherently be 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 paper that practical considerations impose some strong limitations on the size of models that we can consider. Furthermore, it might just be the case that only a single coin is being tossed. Then using the 3-coin model of Fig. 2(c) would be inappropriate, since the actual physical event would not correspond to the model being used—i.e., we would be using an underspecified system.

The Urn and Ball Model[4]: To extend the ideas of the HMM to a somewhat more complicated situation, consider the urn and ball system of Fig. 3. We assume that there are N (large) glass urns in a room. Within each urn there are a large number 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 process, he (or she) chooses an initial urn. From this urn, a ball is chosen at random, and its color is recorded as the observation. The ball is then replaced in the urn from which it was selected. A new urn is then selected

[4]The urn and ball model was introduced by Jack Ferguson, and his colleagues, in lectures on HMM theory.

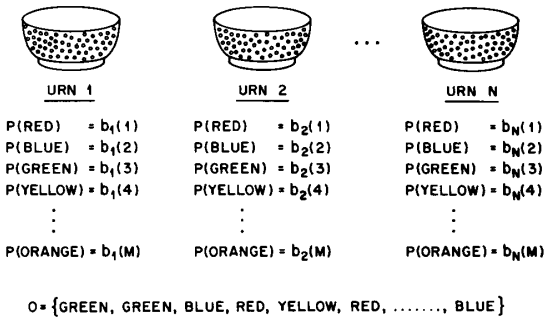

Fig. 3. An N-state urn and ball model which illustrates the general case of a discrete symbol HMM.

according to the random selection process 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.

B. Elements of an HMM

The above examples give us a pretty good 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, and explain how the model generates observation sequences.

An HMM 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. Hence, 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 (e.g., an ergodic model); however, we will see later in this paper that other possible interconnections of states are often of interest. We denote the individual states as $S = \{S_1, S_2, \cdots, S_N\}$, and 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 $V = \{v_1, v_2, \cdots, v_M\}$.

3) The state transition probability distribution $A = \{a_{ij}\}$ where

$$a_{ij} = P[q_{t+1} = S_j | q_t = S_i], \quad 1 \le i, j \le N. \tag{7}$$

For the special case where 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 in state j, $B = \{b_j(k)\}$, where

$$b_j(k) = P[v_k \text{ at } t | q_t = S_j], \quad 1 \le j \le N$$
$$1 \le k \le M. \tag{8}$$

5) The initial state distribution $\pi = \{\pi_i\}$ where

$$\pi_i = P[q_1 = S_i], \quad 1 \le i \le N. \tag{9}$$

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

$$O = O_1 O_2 \cdots O_T \tag{10}$$

(where each observation 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 $q_1 = S_i$ according to the initial state distribution π.
2) Set $t = 1$.
3) Choose $O_t = v_k$ according to the symbol probability distribution in state S_i, i.e., $b_i(k)$.
4) Transit to a new state $q_{t+1} = S_j$ according to the state transition probability distribution for state S_i, i.e., a_{ij}.
5) Set $t = t + 1$; return to step 3)if $t < T$; otherwise terminate the procedure.

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

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 probability measures A, B, and π. For convenience, we use the compact notation

$$\lambda = (A, B, \pi) \tag{11}$$

to indicate the complete parameter set of the model.

C. The Three Basic Problems for HMMs[5]

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

Problem 1: Given the observation sequence $O = O_1 O_2 \cdots O_T$, and a model $\lambda = (A, B, \pi)$, how do we efficiently compute $P(O|\lambda)$, the probability of the observation sequence, given the model?

Problem 2: Given the observation sequence $O = O_1 O_2 \cdots O_T$, and the model λ, how do we choose a corresponding state sequence $Q = q_1 q_2 \cdots q_T$ which is optimal in some meaningful sense (i.e., best "explains" the observations)?

Problem 3: How do we adjust the model parameters $\lambda = (A, B, \pi)$ to maximize $P(O|\lambda)$?

[5]The material in this section and in Section III is based on the ideas presented by Jack Ferguson of IDA in lectures at Bell Laboratories.

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 which best matches the observations.

Problem 2 is the one in which we attempt to uncover the hidden part of the model, i.e., 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. Unfortunately, as we will see, there are several reasonable optimality criteria that 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 so as to best describe how a given observation sequence comes about. The observation sequence used to adjust the model parameters is called a training sequence since it is used to "train" the HMM. The training problem is the crucial one for most applications of HMMs, since 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 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 would be to make refinements on the model (e.g., more states, different codebook size, etc.) so as to improve its capability of modeling the spoken word sequences. Finally, once the set of W HMMs has been designed and optimized and thoroughly studied, 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 section we present formal mathematical solutions to each of the three fundamental problems for HMMs.

We shall see that the three problems are linked together tightly under our probabilistic framework.

III. Solutions to the Three Basic Problems of HMMs

A. Solution to Problem 1

We wish to calculate the probability of the observation sequence, $O = O_1 O_2 \cdots O_T$, given the model λ, i.e., $P(O|\lambda)$. The most straightforward way of doing this is through enumerating every possible state sequence of length T (the number of observations). Consider one such fixed state sequence

$$Q = q_1 q_2 \cdots q_T \tag{12}$$

where q_1 is the initial state. The probability of the observation sequence O for the state sequence of (12) is

$$P(O|Q, \lambda) = \prod_{t=1}^{T} P(O_t|q_t, \lambda) \tag{13a}$$

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

$$P(O|Q, \lambda) = b_{q_1}(O_1) \cdot b_{q_2}(O_2) \cdots b_{q_T}(O_T). \tag{13b}$$

The probability of such a state sequence Q can be written as

$$P(Q|\lambda) = \pi_{q_1} a_{q_1 q_2} a_{q_2 q_3} \cdots a_{q_{T-1} q_T}. \tag{14}$$

The joint probability of O and Q, i.e., the probability that O and Q occur simultaneously, is simply the product of the above two terms, i.e.,

$$P(O, Q|\lambda) = P(O|Q, \lambda) P(Q, \lambda). \tag{15}$$

The probability of O (given the model)is obtained by summing this joint probability over all possible state sequences q giving

$$P(O|\lambda) = \sum_{\text{all } Q} P(O|Q, \lambda) P(Q|\lambda) \tag{16}$$

$$= \sum_{q_1, q_2, \cdots, q_T} \pi_{q_1} b_{q_1}(O_1) a_{q_1 q_2} b_{q_2}(O_2) \cdots a_{q_{T-1} q_T} b_{q_T}(O_T). \tag{17}$$

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 O_1 (in this state) with probability $b_{q_1}(O_1)$. The clock changes from time t to $t + 1$ ($t = 2$) and we make a transition to state q_2 from state q_1 with probability $a_{q_1 q_2}$, and generate symbol O_2 with probability $b_{q_2}(O_2)$. This process continues in this manner until we make the list transition (at time T) from state q_{T-1} to state q_T with probability $a_{q_{T-1} q_T}$ and generate symbol O_T with probability $b_{q_T}(O_T)$.

A little thought should convince the reader that the calculation of $P(O|\lambda)$, according to its direct definition (17) involves on the order of $2T \cdot N^T$ calculations, since at every $t = 1, 2, \cdots, T$, there are N possible states which 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 (17). (To be precise, we need $(2T - 1)N^T$ multiplications, and $N^T - 1$ additions.) This calculation is computationally unfeasible, 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 exists and is called the forward-backward procedure.

The Forward-Backward Procedure [2], [3][6]: Consider the forward variable $\alpha_t(i)$ defined as

$$\alpha_t(i) = P(O_1 O_2 \cdots O_t, q_t = S_i|\lambda) \tag{18}$$

i.e., the probability of the partial observation sequence, $O_1 O_2 \cdots O_t$, (until time t) *and* state S_i at time t, given the model λ. We can solve for $\alpha_t(i)$ inductively, as follows:

1) Initialization:

$$\alpha_1(i) = \pi_i b_i(O_1), \qquad 1 \le i \le N. \tag{19}$$

2) Induction:

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij}\right] b_j(O_{t+1}), \qquad 1 \le t \le T - 1, \quad 1 \le j \le N. \tag{20}$$

3) Termination:

$$P(O|\lambda) = \sum_{i=1}^{N} \alpha_T(i). \tag{21}$$

Step 1) initializes the forward probabilities as the joint probability of state S_i and initial observation O_1. The induction step, which is the heart of the forward calculation, is illustrated in Fig. 4(a). This figure shows how state S_j can be

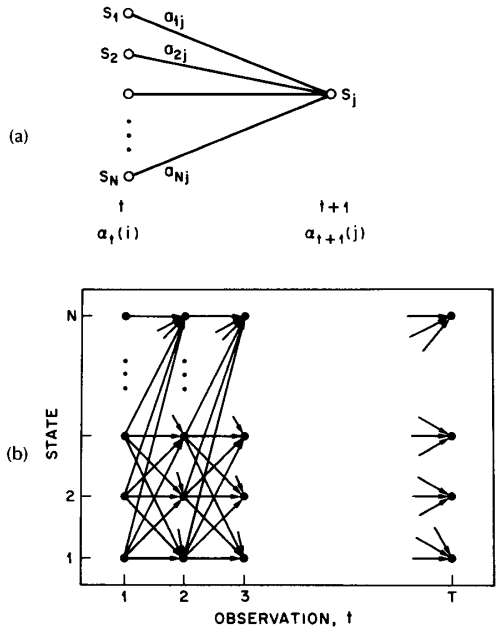

Fig. 4. (a) Illustration of the sequence of operations required for the computation of the forward variable $\alpha_{t+1}(j)$. (b) Implementation of the computation of $\alpha_t(i)$ in terms of a lattice of observations t, and states i.

[6]Strictly speaking, we only need the forward part of the forward-backward procedure to solve Problem 1. We will introduce the backward part of the procedure in this section since it will be used to help solve Problem 3.

reached at time $t + 1$ from the N possible states, S_i, $1 \le i \le N$, at time t. Since $\alpha_t(i)$ is the probability of the joint event that $O_1 O_2 \cdots O_t$ are observed, and the state at time t is S_i, the product $\alpha_t(i) a_{ij}$ is then the probability of the joint event that $O_1 O_2 \cdots O_t$ are observed, and state S_j is reached at time $t + 1$ via state S_i at time t. Summing this product over all the N possible states S_i, $1 \le i \le N$ at time t results in the probability of S_j at time $t + 1$ with all the accompanying previous partial observations. Once this is done and S_j is known, it is easy to see that $\alpha_{t+1}(j)$ is obtained by accounting for observation O_{t+1} in state j, i.e., by multiplying the summed quantity by the probability $b_j(O_{t+1})$. The computation of (20) is performed for all states j, $1 \le j \le N$, for a given t; the computation is then iterated for $t = 1, 2, \cdots, T - 1$. Finally, step 3) gives the desired calculation of $P(O|\lambda)$ as the sum of the terminal forward variables $\alpha_T(i)$. This is the case since, by definition,

$$\alpha_T(i) = P(O_1 O_2 \cdots O_T, q_T = S_i|\lambda) \tag{22}$$

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

If we examine the computation involved in the calculation of $\alpha_t(j)$, $1 \le t \le T$, $1 \le j \le N$, 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 Fig. 4(b). The key is that since 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 $t = 2, 3, \cdots, T$, we only need to calculate values of $\alpha_t(j)$, $1 \le j \le N$, where each calculation involves only N previous values of $\alpha_{t-1}(i)$ because each of the N grid points is reached from the same N grid points at the previous time slot.

In a similar manner,[7] we can consider a backward variable $\beta_t(i)$ defined as

$$\beta_t(i) = P(O_{t+1} O_{t+2} \cdots O_T|q_t = S_i, \lambda) \tag{23}$$

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

1) Initialization:

$$\beta_T(i) = 1, \quad 1 \le i \le N. \tag{24}$$

2) Induction:

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

$$t = T - 1, T - 2, \cdots, 1, \quad 1 \le i \le N. \tag{25}$$

The initialization step 1) *arbitrarily* defines $\beta_T(i)$ to be 1 for all i. Step 2), which is illustrated in Fig. 5, shows that in order to have been in state S_i at time t, and to account for the observation sequence from time $t + 1$ on, you have to consider all possible states S_j at time $t + 1$, accounting for the transition from S_i to S_j (the a_{ij} term), as well as the observation O_{t+1} in state j (the $b_j(O_{t+1})$ 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 extensively to help solve fundamental Problems 2 and 3 of HMMs.

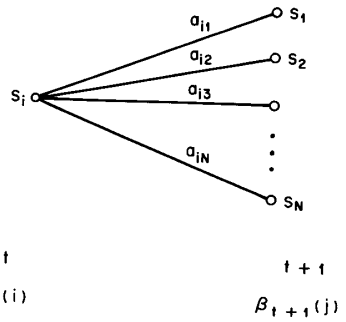

Fig. 5. Illustration of the sequence of operations required for the computation of the backward variable $\beta_t(i)$.

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 Fig. 4(b).

B. Solution to Problem 2

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; i.e., there are several possible optimality criteria. For example, one possible optimality criterion is to choose the states q_t which are *individually* most likely. This optimality criterion maximizes the expected number of correct individual states. To implement this solution to Problem 2, we define the variable

$$\gamma_t(i) = P(q_t = S_i|O, \lambda) \tag{26}$$

i.e., the probability of being in state S_i at time t, given the observation sequence O, and the model λ. Equation (26) can be expressed simply in terms of the forward-backward variables, i.e.,

$$\gamma_t(i) = \frac{\alpha_t(i) \beta_t(i)}{P(O|\lambda)} = \frac{\alpha_t(i) \beta_t(i)}{\sum_{i=1}^{N} \alpha_t(i) \beta_t(i)} \tag{27}$$

since $\alpha_t(i)$ accounts for the partial observation sequence $O_1 O_2 \cdots O_t$ and state S_i at t, while $\beta_t(i)$ accounts for the remainder of the observation sequence $O_{t+1} O_{t+2} \cdots O_T$, given state S_i at t. The normalization factor $P(O|\lambda) = \sum_{i=1}^{N} \alpha_t(i) \beta_t(i)$ makes $\gamma_t(i)$ a probability measure so that

$$\sum_{i=1}^{N} \gamma_t(i) = 1. \tag{28}$$

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

$$q_t = \underset{1 \le i \le N}{\text{argmax}} [\gamma_t(i)], \quad 1 \le t \le T. \tag{29}$$

[7] Again we remind the reader that the backward procedure will be used in the solution to Problem 3, and is not required for the solution of Problem 1.

Although (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 probabilty ($a_{ij} = 0$ for some i and j), the "optimal" state sequence may, in fact, not even be a valid state sequence. This is due to the fact that the solution of (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), i.e., to maximize $P(Q|O, \lambda)$ which is equivalent to maximizing $P(Q, O|\lambda)$. A formal technique for finding this single best state sequence exists, based on dynamic programming methods, and is called the Viterbi algorithm.

Viterbi Algorithm [21], [22]: To find the single best state sequence, $Q = \{q_1\, q_2 \cdots q_T\}$, for the given observation sequence $O = \{O_1\, O_2 \cdots O_T\}$, we need to define the quantity

$$\delta_t(i) = \max_{q_1, q_2, \cdots, q_{t-1}} P[q_1\, q_2 \cdots q_t = i,\, O_1\, O_2 \cdots O_t|\lambda] \tag{30}$$

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

$$\delta_{t+1}(j) = [\max_i \delta_t(i) a_{ij}] \cdot b_j(O_{t+1}). \tag{31}$$

To actually retrieve the state sequence, we need to keep track of the argument which maximized (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:

$$\delta_1(i) = \pi_i b_i(O_1), \qquad 1 \le i \le N \tag{32a}$$

$$\psi_1(i) = 0. \tag{32b}$$

2) Recursion:

$$\delta_t(j) = \max_{1 \le i \le N} [\delta_{t-1}(i) a_{ij}] b_j(O_t), \qquad 2 \le t \le T, \quad 1 \le j \le N \tag{33a}$$

$$\psi_t(j) = \operatorname*{argmax}_{1 \le i \le N} [\delta_{t-1}(i) a_{ij}], \qquad 2 \le t \le T, \quad 1 \le j \le N. \tag{33b}$$

3) Termination:

$$P^* = \max_{1 \le i \le N} [\delta_T(i)] \tag{34a}$$

$$q_T^* = \operatorname*{argmax}_{1 \le i \le N} [\delta_T(i)]. \tag{34b}$$

4) Path (state sequence) backtracking:

$$q_t^* = \psi_{t+1}(q_{t+1}^*), \qquad t = T-1, T-2, \cdots, 1. \tag{35}$$

It should be noted that the Viterbi algorithm is similar (except for the backtracking step) in implementation to the forward calculation of (19)–(21). The major difference is the maximization in (33a) over previous states which is used in place of the summing procedure in (20). It also should be clear that a lattice (or trellis) structure efficiently implements the computation of the Viterbi procedure.

C. Solution to Problem 3 [1]–[5]

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 maximize the probability of the observation sequence given the model. There is no known way to analytically solve for the model which maximizes the probability of the observation sequence. In fact, given any finite observation sequence as training data, there is no optimal way of estimating the model parameters. We can, however, choose $\lambda = (A, B, \pi)$ such that $P(O|\lambda)$ is locally maximized using an iterative procedure such as the Baum–Welch method (or equivalently the EM (expectation-modification) method [23]), or using gradient techniques [14]. In this section we discuss one iterative procedure, based primarily on the classic work of Baum and his colleagues, for choosing model parameters.

In order 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 S_i at time t, and state S_j at time $t + 1$, given the model and the observation sequence, i.e.

$$\xi_t(i, j) = P(q_t = S_i,\, q_{t+1} = S_j|O, \lambda). \tag{36}$$

The sequence of events leading to the conditions required by (36) is illustrated in Fig. 6. It should be clear, from the

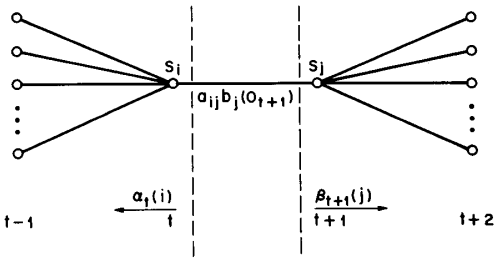

Fig. 6. Illustration of the sequence of operations required for the computation of the joint event that the system is in state S_i at time t and state S_j at time $t + 1$.

definitions of the forward and backward variables, that we can write $\xi_t(i, j)$ in the form

$$\xi_t(i, j) = \frac{\alpha_t(i)\, a_{ij} b_j(O_{t+1})\, \beta_{t+1}(j)}{P(O|\lambda)} = \frac{\alpha_t(i)\, a_{ij} b_j(O_{t+1})\, \beta_{t+1}(j)}{\sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_t(i)\, a_{ij} b_j(O_{t+1})\, \beta_{t+1}(j)} \tag{37}$$

where the numerator term is just $P(q_t = S_i, q_{t+1} = S_j, O|\lambda)$ and the division by $P(O|\lambda)$ gives the desired probability measure.

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

$$\gamma_t(i) = \sum_{j=1}^{N} \xi_t(i, j). \tag{38}$$

If we sum $\gamma_t(i)$ over the time index t, we get a quantity which can be interpreted as the expected (over time) number of times that state S_i is visited, or equivalently, the expected number of transitions made from state S_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 S_i to state S_j. That is

$$\sum_{t=1}^{T-1} \gamma_t(i) = \text{expected number of transitions from } S_i \tag{39a}$$

$$\sum_{t=1}^{T-1} \xi_t(i, j) = \text{expected number of transitions from } S_i \text{ to } S_j. \tag{39b}$$

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 are

$$\bar{\pi}_i = \text{expected frequency (number of times) in state } S_i \text{ at time } (t = 1) = \gamma_1(i) \tag{40a}$$

$$\bar{a}_{ij} = \frac{\text{expected number of transitions from state } S_i \text{ to state } S_j}{\text{expected number of transitions from state } S_i} = \frac{\sum_{t=1}^{T-1} \xi_t(i, j)}{\sum_{t=1}^{T-1} \gamma_t(i)} \tag{40b}$$

$$\bar{b}_j(k) = \frac{\text{expected number of times in state } j \text{ and observing symbol } v_k}{\text{expected number of times in state } j} = \frac{\sum_{t=1,\ \text{s.t. } O_t = v_k}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)}. \tag{40c}$$

If we define the current model as $\lambda = (A, B, \pi)$, and use that to compute the right-hand sides of (40a)–(40c), and we define the reestimated model as $\bar{\lambda} = \bar{A}, \bar{B}, \bar{\pi})$, as determined from the left-hand sides of (40a)–(40c), then it has been proven by Baum and his colleagues [6], [3] 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 $P(O|\bar{\lambda}) > P(O|\lambda)$, i.e., 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 O being observed from the model until some limiting point is reached. The final result of this reestimation procedure is called a maximum likelihood 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 optimization surface is very complex and has many local maxima.

The reestimation formulas of (40a)–(40c) can be derived directly by maximizing (using standard constrained optimization techniques) Baum's auxiliary function

$$Q(\lambda, \bar{\lambda}) = \sum_{Q} P(Q|O, \lambda) \log [P(O, Q|\bar{\lambda})] \tag{41}$$

over $\bar{\lambda}$. It has been proven by Baum and his colleagues [6], [3] that maximization of $Q(\lambda, \bar{\lambda})$ leads to increased likelihood, i.e.

$$\max_{\bar{\lambda}} [Q(\lambda, \bar{\lambda})] \Rightarrow P(O|\bar{\lambda}) \geq P(O|\lambda). \tag{42}$$

Eventually the likelihood function converges to a critical point.

Notes on the Reestimation Procedure: The reestimation formulas can readily be interpreted as an implementation of the EM algorithm of statistics [23] in which the E (expectation) step is the calculation of the auxiliary function $Q(\lambda, \bar{\lambda})$, and the M (modification) step is the maximization over $\bar{\lambda}$. Thus the Baum–Welch reestimation equations are essentially identical to the EM steps for this particular problem.

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

$$\sum_{i=1}^{N} \bar{\pi}_i = 1 \tag{43a}$$

$$\sum_{j=1}^{N} \bar{a}_{ij} = 1, \quad 1 \leq i \leq N \tag{43b}$$

$$\sum_{k=1}^{M} \bar{b}_j(k) = 1, \quad 1 \leq j \leq N \tag{43c}$$

are automatically satisfied at each iteration. By looking at the parameter estimation problem as a constrained optimization of $P(O|\lambda)$ (subject to the constraints of (43)), the techniques of Lagrange multipliers can be used to find the values of π_i, a_{ij}, and $b_j(k)$ which maximize P (we use the notation $P = P(O|\lambda)$ as short-hand in this section). Based on setting up a standard Lagrange optimization using Lagrange multipliers, it can readily be shown that P is maximized when

the following conditions are met:

$$\pi_i = \frac{\pi_i \dfrac{\partial P}{\partial \pi_i}}{\sum_{k=1}^{N} \pi_k \dfrac{\partial P}{\partial \pi_k}} \tag{44a}$$

$$a_{ij} = \frac{a_{ij} \dfrac{\partial P}{\partial a_{ij}}}{\sum_{k=1}^{N} a_{ik} \dfrac{\partial P}{\partial a_{ik}}} \tag{44b}$$

$$b_j(k) = \frac{b_j(k) \dfrac{\partial P}{\partial b_j(k)}}{\sum_{\ell=1}^{M} b_j(\ell) \dfrac{\partial P}{\partial b_j(\ell)}}. \tag{44c}$$

By appropriate manipulation of (44), the right-hand sides of each equation can be readily converted to be *identical* to the right-hand sides of each part of (40a)–(40c), thereby showing that the reestimation formulas are indeed exactly correct at critical points of P. In fact the form of (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 [14]. Such procedures have been tried and have been shown to yield solutions comparable to those of the standard reestimation procedures.

IV. Types of HMMs

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 number of steps.) As shown in Fig. 7(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 Fig. 7a 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, in particular those to be discussed later in this paper, 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 Fig. 7(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), i.e., the 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—e.g., speech. The fundamental property of all left-right

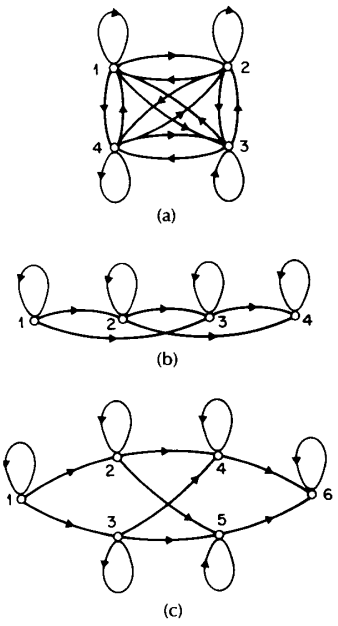

Fig. 7. Illustration of 3 distinct types of HMMs. (a) A 4-state ergodic model. (b) A 4-state left-right model. (c) A 6-state parallel path left-right model.

HMMs is that the state transition coefficients have the property

$$a_{ij} = 0, \quad j < i \tag{45}$$

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

$$\pi_i = \begin{cases} 0, & i \neq 1 \\ 1, & i = 1 \end{cases} \tag{46}$$

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

$$a_{ij} = 0, \quad j > i + \Delta \tag{47}$$

is often used. In particular, for the example of Fig. 7(b), the value of Δ is 2, i.e., no jumps of more than 2 states are allowed. The form of the state transition matrix for the example of Fig. 7(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, that the state transition coefficients are specified as

$$a_{NN} = 1 \tag{48a}$$

$$a_{Ni} = 0, \quad i < N. \tag{48b}$$

Although we have dichotomized HMMs into ergodic and left-right models, there are many possible variations and combinations possible. By way of example, Fig. 7(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 can be seen that 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 (44)).

A. Continuous Observation Densities in HMMs [24]–[26]

All of our discussion, to this point, has considered only the case 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. The problem with this approach, at least for some applications, is that the observations are continuous signals (or vectors). Although it is possible to quantize such continuous signals via codebooks, etc., there might be serious degradation associated with such quantization. Hence it would be advantageous to be able to use HMMs with continuous observation densities.

In order to use a continuous observation density, some restrictions have to be placed on the form of the model probability density function (pdf) to insure 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 [24]–[26], is a finite mixture of the form

$$b_j(\boldsymbol{O}) = \sum_{m=1}^{M} c_{jm}\mathfrak{N}[\boldsymbol{O}, \boldsymbol{\mu}_{jm}, \boldsymbol{U}_{jm}], \quad 1 \le j \le N \tag{49}$$

where $\boldsymbol{O}$ is the vector being modeled, c_{jm} is the mixture coefficient for the mth mixture in state j and $\mathfrak{N}$ is any log-concave or elliptically symmetric density [24] (e.g., Gaussian), with mean vector $\boldsymbol{\mu}_{jm}$ and covariance matrix $\boldsymbol{U}_{jm}$ for the mth mixture component in state j. Usually a Gaussian density is used for $\mathfrak{N}$. The mixture gains c_{jm} satisfy the stochastic constraint

$$\sum_{m=1}^{M} c_{jm} = 1, \quad 1 \le j \le N \tag{50a}$$

$$c_{jm} \ge 0, \quad 1 \le j \le N, 1 \le m \le M \tag{50b}$$

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

$$\int_{-\infty}^{\infty} b_j(\boldsymbol{x})\, d\boldsymbol{x} = 1, \quad 1 \le j \le N. \tag{51}$$

The pdf of (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 can be shown [24]–[26] that the reestimation formulas for the coefficients of the mixture density, i.e., c_{jm}, $\boldsymbol{\mu}_{jk}$, and $\boldsymbol{U}_{jk}$, are of the form

$$\bar{c}_{jk} = \frac{\sum_{t=1}^{T} \gamma_t(j, k)}{\sum_{t=1}^{T}\sum_{k=1}^{M} \gamma_t(j, k)} \tag{52}$$

$$\bar{\boldsymbol{\mu}}_{jk} = \frac{\sum_{t=1}^{T} \gamma_t(j, k) \cdot \boldsymbol{O}_t}{\sum_{t=1}^{T} \gamma_t(j, k)} \tag{53}$$

$$\bar{\boldsymbol{U}}_{jk} = \frac{\sum_{t=1}^{T} \gamma_t(j, k) \cdot (\boldsymbol{O}_t - \boldsymbol{\mu}_{jk})(\boldsymbol{O}_t - \boldsymbol{\mu}_{jk})'}{\sum_{t=1}^{T} \gamma_t(j, k)} \tag{54}$$

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 $\boldsymbol{O}_t$, i.e.,

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

(The term $\gamma_t(j, k)$ generalizes to $\gamma_t(j)$ of (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., (40b)). The interpretation of (52)–(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 (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 $\boldsymbol{U}_{jk}$.

B. Autoregressive HMMS [27], [28]

Although the general formulation of continuous density HMMs is applicable to a wide range of problems, there is one other very interesting class of HMMs that is particularly applicable to speech processing. This is the class of autoregressive HMMs [27], [28]. For this class, the observation vectors are drawn from an autoregression process.

To be more specific, consider the observation vector $\boldsymbol{O}$ with components $(x_0, x_1, x_2, \cdots, x_{K-1})$. Since the basis probability density function for the observation vector is Gaussian autoregressive (or order p), then the components of $\boldsymbol{O}$ are related by

$$\boldsymbol{O}_k = -\sum_{i=1}^{p} a_i \boldsymbol{O}_{k-i} + e_k \tag{55}$$

where e_k, $k = 0, 1, 2, \cdots, K-1$ are Gaussian, independent, identically distributed random variables with zero mean and variance σ^2, and a_i, $i = 1, 2, \cdots, p$, are the autoregression or predictor coefficients. It can be shown that for large K, the density function for $\boldsymbol{O}$ is approximately

$$f(\boldsymbol{O}) = (2\pi\sigma^2)^{-K/2} \exp\left\{-\frac{1}{2\sigma^2}\,\delta(\boldsymbol{O}, \boldsymbol{a})\right\} \tag{56}$$

where

$$\delta(\boldsymbol{O}, \boldsymbol{a}) = r_a(0)\, r(0) + 2\sum_{i=1}^{p} r_a(i)\, r(i) \tag{57a}$$

$$\boldsymbol{a}' = [1, a_1, a_2, \cdots, a_p] \tag{57b}$$