12 Hidden Markov Model

LEARNING OBJECTIVES

- To introduce the category of recognition problems using temporal patterns.
- To introduce the basics of Hidden Markov Model (HMM).
- To understand the issues in HMM.

LEARNING OUTCOMES

- Students will understand and appreciate the use of HMM as a classifier.
- Students will be get an insight on speech recognition problems.
- Students will understand the issues in HMM.

12.1 Introduction to Hidden Markov Model

Speech recognition problems come under the category of recognition problems with temporal patterns. We can usually see some persons with some mannerisms. As an example, some may use lots of hand gestures while talking. Some frequently repeat some words like "Got it", "Catch my point". So in a temporal sequence, that in a speech signal if some patterns are frequently repeated its called as temporal pattern. Speech signals are time baring signals (i.e., at different instances of time we have different signal strength). For speech recognition purposes, the speech is divided into a number of phonemes. The word spoken can be identified based on the sequence in which the phonemes occur.

Activity recognition is another application of temporal pattern recognition. These activities can be recognized by sign language recognition using hand movements. Based on the sequence of hand gestures, we can interpret the message conveyed. An application area of activity recognition is security surveillance where abnormal movement or activity can be monitored. Movement in security surveillance is nothing but body postures at different instances of time. So if the movement is found to be abnormal, it implies that it would be from a suspicious person.

To recognize or identify such temporal sequence we need a machine which is similar to a sequential machine or finite state machine. In other words, a machine takes us through a finite set of output symbols from a finite set of input symbols. Thus, we have a finite set of states through which the machine makes a transition, i.e., at any time of time the machine moves from one state to another, it takes an input and emits an output.

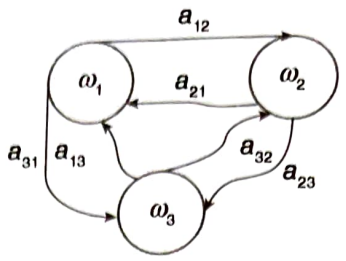

Figure 12.1 State transitions of HMM.

The machine makes a transition from one input state to an output state. If the set of outputs are limited to only two symbols, 0 and 1, the sequence machine becomes a finite state automaton and has huge application in sequence generation and sequence detection. Moreover, it is a useful concept in communication. So if a finite state automaton detects the beat sequence, the next portion of the sequence can also be detected.

Hidden Markov Model (HMM) is a statistical model that takes statistical property of signal into account. There are two types of states in HMM:

1. Visible state (V).
2. Hidden state (ω).

HMM can have a number of hidden states and visible states. As an example, let us consider an HMM model (θ) having three hidden states (ω) and three visible states (V). This is given by $\omega = \{\omega_1, \omega_2, \omega_3\}$ and $V = \{V_1, V_2, V_3\}$.

Figure 12.1 shows the state transitions of the HMM from state ω_{t-1} at time $t-1$ to state t with a probability a_{ij}.

In each state, the machine can emit one of the visible states. The probability of emission from a visible state is given by $P(v_k|\omega_j) = b_{jk}$, that is the probability of v_k given ω_j. So from ω_1 the machine emits v_2 with a probability v_{12} and v_3 with a probability v_{13}. These are the probabilities of emission of visible states from different hidden states of HMM. The above process is called *observable Markov model*, since the output of the process is the set of states at each instant of time, where each state corresponds to an observable event. Thus, we can see that given any state there will always be transition to some of the states including the original state itself.

Redrawing Fig. 12.1 with the visible states, hidden states, emission probabilities, and transition probabilities, we get Fig. 12.2.

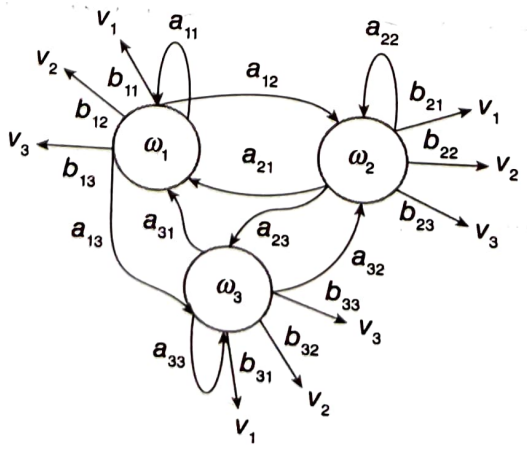

Figure 12.2 State transition diagrams showing emission and transition probabilities.

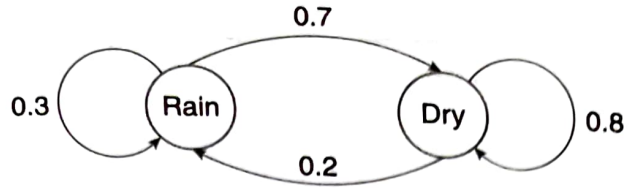

Figure 12.3 Markov chain for Rain and Dry.

It can also be seen that at any point of time

$$\sum a_{ij} = 1, \quad \forall i \quad \text{and} \quad \sum b_{jk} = 1, \quad \forall j$$

Let us consider the example of two hidden states Rain and Dry with the initial probabilities as shown in Fig. 12.3.

Figure 12.3 shows a simple example of Markov chain with two hidden states Rain and Dry. The different state transition probabilities when the state changes from state Rain to Rain, Rain to Dry, Dry to Dry, Dry to Rain are given below, along with the initial probability of Rain and Dry.

$$P(\text{Rain}|\text{Rain}) = 0.3$$
$$P(\text{Dry}|\text{Dry}) = 0.8$$
$$P(\text{Dry}|\text{Rain}) = 0.7$$
$$P(\text{Rain}|\text{Dry}) = 0.2$$
$$P(\text{Rain}) = 0.4$$
$$P(\text{Dry}) = 0.6$$

It is seen that the sum of all transition probabilities is equal to 1 and the sum of emission probabilities is also equal to 1, as given in the calculations below.

$$P(\text{Rain}|\text{Rain}) + P(\text{Dry}|\text{Rain}) = 0.3 + 0.7 = 1$$
$$P(\text{Dry}|\text{Dry}) + P(\text{Rain}|\text{Dry}) = 0.2 + 0.8 = 1$$

Finally, HMM has a specific hidden state called *receiving state* or *accepting state*. Once the machine reaches that state it cannot come out. Only one visible state can be emitted from this hidden state. Incorporating this concept in Fig. 12.2 and redrawing it we get Fig. 12.4. It is seen that v_0 is the final visible state that is emitted from the hidden state. There are no transitions to other hidden states from this state.

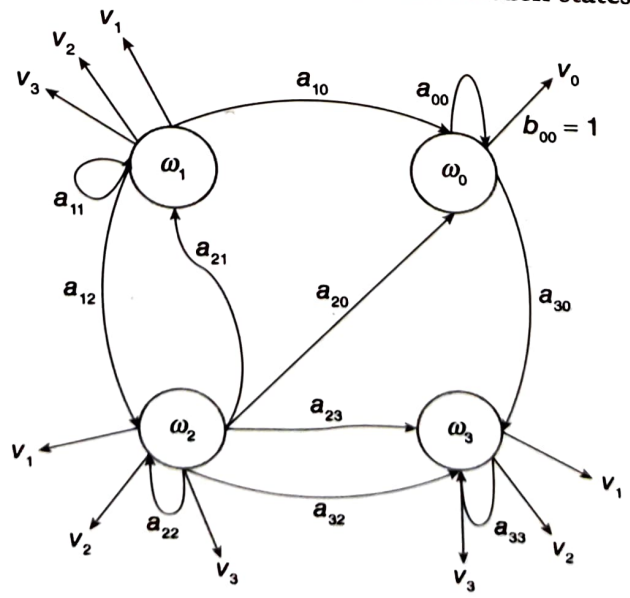

Figure 12.4 Markov model with visible and hidden states.

This HMM model is used for temporal pattern recognition. As discussed in Section 12.1, a temporal pattern is a frequently recurring pattern in a sequence of signals. Thus, HMM is used in the identification of such patterns in a sequence.

12.2 Issues in Hidden Markov Model

There are three central issues that need to be addressed in HMM:

1. Evaluation Problem.
2. Decoding Problem.
3. Learning Problem.

If an HMM θ has number of hidden states w, number of visible states v, transmission probability from one hidden state to another a_{ij}, emission probability from visible state b_{jk}, then the probability that a particular visible state is emitted is given by $P(V^T/\theta)$ where technically

- $\theta = \theta_1, \theta_2, \ldots, \theta_c$
- $\theta \Rightarrow w, v, a_{ij}, b_{jk}$
- $V^T \Rightarrow$ sequence of visible symbols

where θ is the Markov model having hidden states, visible states, transition probabilities, and emission probabilities.

In other words, the evaluation problem is given the observation sequence V^T. Now, the question is: What is the probability that this sequence is generated by any of the models $\theta = \theta_1, \theta_2, \ldots, \theta_c$. We can think of this as a scoring problem. If you have to choose between many competing models, then one with maximum probability will give better results. In other words, the model with maximum probability will be the best model for the sequence of visible states. Thus, finding this probability $P(V^T/\theta)$ is the evaluation problem. So when there are 'c' number of model sequences for every sequence, there exists an HMM θ where θ_1 is the first HMM, θ_2 is the second HMM, θ_n is the n^{th} sequence HMM. The probability has to be computed given the unknown sequence $P(V^T/\theta^n)$, where it can be produced by $\theta_1, \theta_2, \ldots, \theta_n$. Once this probability is known, the visible sequence V^T can be classified provided we know the HMM. So if there are 'c' number of sequences, the unknown sequence has to be classified into one of these sequences. Every sequence has an HMM θ. So if there are 'c' sequences, then HMM ranges from θ_1 to θ_c.

Every θ in the sequence of $\theta_1, \theta_2, \ldots, \theta_c$ will have its own a_{ij}, b_{jk}, w, v. The evaluation problem is: Given a V^T, what is the probability that this state is produced by any of $\theta_1, \theta_2, \ldots, \theta_c$?

The decoding problem is what is the most likely $\omega(t)$ of the hidden state which has led to the generation of V^T, where ω^T is the generated V^T. Problem 2 is one which attempts to uncover the hidden part of the problem. There is no correct solution to this problem, in practice we usually use optimal criterion to find best possible solution.

The learning problem is estimating a_{ij} and b_{jk} knowing w and V. Here, we try to optimize model parameters to describe how a given observation sequence comes out. The observation sequence used here is called *training sequence* since it is used for training HMM. Training is one of the crucial elements of HMM. It allows us to adjust model parameters as to create the best model for a given training sequence. This learning problem is important not only in HMM, but in any classifier problem. Now let us see each of these issues in depth.

Let us take a simple example of a coin toss problem to explain HMM. Assume the following scenario. You are isolated in a room, and you cannot see what is going on outside. Your friend outside tosses a coin and tells you the result of each coin flip. Thus, a sequence of heads and tails would be performed by your

friend and you know the sequence. A typical sequence would have (H, T, T, H, T, H, …, H), where H stands for heads and T stands for tails. To model this coin tossing event with HMM, two questions are raised.

1. What state is the status of the model?
2. How many states should there be in the model?

This depends on the number of coins being tossed. If it is only a single coin then we have either heads or tails. But if the number of coins is more than one, then we get a sequence from the different coins that are tossed. In other words, we get a matrix with rows corresponding to the states and columns corresponding to the coins that are tossed. The person inside the isolated room is not aware about which coin is tossed. He can only hear the sequence of heads and tails. This problem corresponding to these coins would be similar to the one depicted in Fig. 12.2.

12.2.1 Evaluation Problem

The evaluation problem is to find the probability of V^T given θ when θ and V^T are known. For this all the possible sequences of hidden state of length T have to be found out. This is given by

$$P\left(V^T \middle| \theta\right) = \sum_{r=1}^{r_{max}} P\left(\frac{V^T}{\omega_r^T}\right) P\left(\omega_r^T\right) \tag{12.1}$$

where

- r is one of the sequences
- $\omega^T \rightarrow$ sequence of T number of those possible sequences of visible states
- r_{max} is the number of such sequences.
- $W_r^T = \{\omega(1), \omega(2) \ldots \omega(T)\}$

If this is on N number of hidden states then

$r_{max} = N^T$

This means that there are N^T number of possible sequences of hidden states of length T.

$N \rightarrow$ # of hidden states

$P\left(\omega_r^T\right)$ is the probability of the sequence from state at time t to a state at $t-1$. It is a sequence of T number of such states. It is given by

$$P\left(\omega_r^T\right) = P\left(\omega(t) \middle| \omega(t-1)\right) \tag{12.2}$$

The product of all such transition probabilities for consecutive time sequences is given by

$$P\left(\omega_r^T\right) = \prod_{t=1}^{T} P\left(\omega(t) \middle| \omega(t-1)\right) \tag{12.3}$$

Let us consider the example of the problem given in Fig. 12.3. Suppose we want to calculate the probability of a sequence of states {Dry, Dry, Rain, Rain}.

$$P\{\text{Dry, Dry, Rain, Rain}\} = P(\text{Rain}|\text{Rain}) \cdot P(\text{Rain}|\text{Dry}) \cdot P(\text{Dry}|\text{Dry}) \cdot P(\text{Dry})$$

$$= 0.3 \times 0.2 \times 0.8 \times 0.6 = 0.0288$$

The probability of finding a visible state given a hidden state is

$$P\left(V^T \middle| \omega_r^T\right) = \prod_{t=1}^{T} P\left(v(t) \middle| \omega(t)\right) \tag{12.4}$$

Substituting Eqs. (12.3) and (12.4) in Eq. (12.5), we get

$$P\left(V^T \middle| \theta\right) = \sum_{r=1}^{r_{max}} \prod_{t=1}^{T} P\left(v(t) \middle| \omega(t)\right) \cdot P\left(\frac{\omega(t)}{\omega(t-1)}\right) \tag{12.5}$$

Complexity of this expression is $O(N^T T)$. This is because there are N^T state paths, each costing $O(T)$ calculations leading to $O(N^T T)$ time complexity. Thus, instead of using a complex expression for computing a visible state, a recursive algorithm can be used. The recursive algorithm is: Given a set of V^T, what would be the probability that HMM would be at a particular state at a particular time?

At $t = 0$ the HMM is at ω_1. Then at $t = 1$ the probability of whether the machine would be in $\omega = 0$, $\omega = 1, 2, \ldots$ can be determined. The reason is that from ω_1 it is possible to make a transition to $\omega_1, \omega_0, \omega_2, \ldots, \omega_i$.

If $V(1) = V(0)$ then the hidden state has been emitted by the previous state ω_1. However, if $V(1) \notin V(0)$ then it has not been emitted from the previous state. This can be written in the form of probability of emission of visible symbol at time step t.

At the time instant t, if the machine is in a particular state, then the probability of transition from $(t-1)$ state to t state can be computed. The corresponding probabilities are given by $\alpha_1(t-1)$, $\alpha_2(t-2) - \alpha_i(t-1)$; the transition probability a_{ij} is also given.

The probability that the machine is in state α at $(t-1)$ instant is given by

$$\alpha_j(t) = \begin{cases} 0, t = 0 \text{ and } j \neq \text{ initial state} \\ 1, t = 0 \text{ and } j = \text{ initial state} \\ \left[\sum \alpha_i(t-1) a_{ij}\right] \cdot b_{jk} V(t) \ldots \text{ otherwise} \end{cases}$$

The recursive algorithm can be either forward algorithm or backward algorithm. The forward algorithm is given by

Initialize: $t \leftarrow 0$ a_{ij}, b_{jk}, V^T, $\alpha_j(0)$ for $t \leftarrow t+1$

$$\alpha_j(t) = b_{jkv}(i) \cdot \sum_{i=1}^{N} \alpha_i(t-1) a_{ij}$$

Until $t = T$

Return $P\left(\frac{V^T}{\theta}\right) \leftarrow \alpha_0(T)$ for final state end.

12.2.2 Decoding Problem

The decoding problem finds out the most likely sequence or most probable sequence of hidden states through which the machine has transmitted while generating V^T. In other words, the decoding problem finds the most probable state. A trellis diagram is used for this purpose. A trellis diagram is made up of a matrix of nodes. The column of the nodes represents all the possible status at a certain time. The first

column represents all the possible status at the time instant 0. Similarly, the second and third columns represent the status at different time intervals. Thus, it would be easy to interpret and view the state changes at different time intervals using a trellis diagram. This diagram is used in convolution networks along with state diagram. This seems an easy way of understanding the flow to signals during every state change. Since HMM has a different number of hidden states, the trellis diagram is used in this case especially to find out the value of each most probable sequence of hidden states.

The algorithm for decoding problem is

Given V^T find the most probably sequence of hidden states $\Rightarrow$ Decoding problem.

Also

Initialize: Path $\leftarrow \{\}\ t \leftarrow 0, j \leftarrow 0$

for $t \leftarrow t+1$

$j \leftarrow j+1$

for $j \leftarrow j+1$

$$\alpha_j(t) \leftarrow b_{jk} v(t) \cdot \sum_{i=1}^{n} \alpha_i(t-1) a_{ij}$$

until $j = N$ (where N = number of in-between states)

$j' \leftarrow \arg\max \alpha_j(t)$

Appending ω_j' to path

until $t = T$ (length of sequence)

Return Path

12.2.3 Learning Problem

In the learning process of HMM, we have to estimate the transmission probability (a_{ij}) and emission probability (b_{jk}). For learning, we use a number of known sequences. For example, if there exists a string of visible states $\langle V_1, V_3, V_1, V_5, V_7, V_2, V_0 \rangle$, then

$\alpha_3(4)$ means the probability that the machine is in state ω_3 and generates sequence V_1, V_3, V_1, V_5.

$\beta_3(4)$ means the probability that the machine is in state ω_3 and generates the next three visible states.

In general, $\beta_i(t)$ is the probability that the model will be in $\omega_i(t)$ and will generate the remainder of the given target sequence V^T. All symbols from $V(t+1)$ to $V(t)$ will be generated. The backward algorithm helps in computing $\beta_i(t)$.

$$\beta_i(t) = \begin{cases} 0, & \omega_i(t) \neq \omega_0 \text{ and } t = T \\ 1, & \omega_i(t) = \omega_0 \text{ and } t = T \\ \sum_j \beta_j(t+1) a_{ij} \cdot b_{jkv}(t+1) \ldots & \text{otherwise} \end{cases}$$

The HMM backward is

Initialize $\beta_j(T)$; $t \leftarrow T\ a_{ij}, b_{jk}, V^T$

for $t \leftarrow t-1$

$$\beta_i(t) = \sum_j \beta_j(t+1) a_{ij} b_{jkv}(t)$$

until $t = 1$

Return $P(V^T) \leftarrow \beta_i(0)$ for the known initial state

end.

For estimation of model parameters a_{ij} and b_{jk}, both forward and backward algorithms have to be used simultaneously.

12.2.4 Learning Problem (Classifier)

The final goal is to classify the sequence of symbols. Bayes rule is used for this, which has already been discussed in detail in Chapter 9.

$P(x|\omega_i)$ is the class or conditional probability of x given the hidden state. The task is to classify x in one of the classes. For this classification we need to compute $P(\omega_i|x)$ and i for which $P(\omega_i|x)$ is maximum. The unknown sample would be classified in that particular class.

$P(\omega_i|x) \rightarrow$ posterior probability

$P(x|\omega_i) \rightarrow$ prior probability

This is given by

$$P(\omega_i|x) = \frac{P(x|\omega_i) \cdot P(\omega_i)}{P(x)} \tag{12.6}$$

If there are two classes, we compute

$$P(\omega_j|x) > P(\omega_i|x)$$

Then the unknown sample is classified under that class.

In HMM, we find $P(V^T|\theta)$ in the evaluation step. In the classification step, we need to compute $P(\theta|V^T)$. Applying Bayes rule as shown in Eq. (12.7), we find the probability of whether the given sequence belongs to a particular class given the visible state.

$$P(\theta/V^T) = \frac{P(V^T/\theta) \cdot P(\theta)}{P(V^T)} \tag{12.7}$$

If there are two models then we have θ_1 and θ_2. We then compute $P(\theta_1/V^T)$ and $P(\theta_2/V^T)$. If $P(\theta_i/V^T) > P(\theta_j/V^T)$ then V^T can be classified in θ_i.

The learning process of HMM is supervised learning because we try to learn with sequences of visible states.