11 *Hidden Markov Models*

Hidden Markov Models are a popular *machine learning* approach in bioinformatics. Machine learning algorithms are presented with *training data*, which are used to derive important insights about the (often hidden) parameters. Once an algorithm has been suitably trained, it can apply these insights to the analysis of a *test sample*. As the amount of training data increases, the accuracy of the machine learning algorithm typically increases as well. The parameters that are learned during training represent knowledge; application of the algorithm with those parameters to new data (not used in the training phase) represents the algorithm's use of that knowledge. The Hidden Markov Model (HMM) approach, considered in this chapter, learns some unknown probabilistic parameters from training samples and uses these parameters in the framework of dynamic programming (and other algorithmic techniques) to find the best explanation for the experimental data.

11.1 *CG*-Islands and the "Fair Bet Casino"

The least frequent dinucleotide in many genomes is CG. The reason for this is that the C within CG is easily *methylated*, and the resulting methyl-C has a tendency to mutate into T.[1] However, the methylation is often suppressed around genes in areas called *CG-islands* in which CG appears relatively frequently. An important problem is to define and locate CG-islands in a long genomic text.

Finding CG-islands can be modeled after the following toy gambling problem. The "Fair Bet Casino" has a game in which a dealer flips a coin and

1. Cells often biochemically modify DNA and proteins. Methylation is the most common DNA modification and results in the addition of a methyl (CH_3) group to a nucleotide position in DNA.

the player bets on the outcome (heads or tails). The dealer in this (crooked) casino uses either a fair coin (heads or tails are equally likely) or a biased coin that will give heads with a probability of $\frac{3}{4}$. For security reasons, the dealer does not like to change coins, so this happens relatively rarely, with a probability of 0.1. Given a sequence of coin tosses, the problem is to find out when the dealer used the biased coin and when he used the fair coin, since this will help you, the player, learn the dealer's psychology and enable you to win money. Obviously, if you observe a long line of heads, it is likely that the dealer used the biased coin, whereas if you see an even distribution of heads and tails, he likely used the fair one. Though you can never be certain that a long string of heads is not just a fluke, you are primarily interested in the most probable explanation of the data. Based on this sensible intuition, we might formulate the problem as follows:

Fair Bet Casino Problem:
Given a sequence of coin tosses, determine when the dealer used a fair coin and when he used a biased coin.

Input: A sequence $x = x_1\, x_2\, x_3 \ldots x_n$ of coin tosses (either H or T) made by two possible coins (F or B).

Output: A sequence $\pi = \pi_1\, \pi_2\, \pi_3 \cdots \pi_n$, with each π_i being either F or B indicating that x_i is the result of tossing the fair or biased coin, respectively.

Unfortunately, this problem formulation simply makes no sense. The ambiguity is that *any* sequence of coins could *possibly* have generated the observed outcomes, so technically $\pi = FFF \ldots FF$ is a valid answer to this problem for every observed sequence of coin flips, as is $\pi = BBB \ldots BB$. We need to incorporate a way to grade different coin sequences as being better answers than others. Below we explain how to turn this ill-defined problem into the Decoding problem based on HMM paradigm.

First, we consider the problem under the assumption that the dealer never changes coins. In this case, letting 0 denote tails and 1 heads, the question is which of the two coins he used, fair ($p^+(0) = p^+(1) = \frac{1}{2}$) or biased ($p^-(0) = \frac{1}{4}$, $p^-(1) = \frac{3}{4}$). If the resulting sequence of tosses is $x = x_1 \ldots x_n$, then the

probability that x was generated by a fair coin is[2]

$$P(x|\text{fair coin}) = \prod_{i=1}^{n} p^+(x_i) = \frac{1}{2^n}.$$

On the other hand, the probability that x was generated by a biased coin is

$$P(x|\text{biased coin}) = \prod_{i=1}^{n} p^-(x_i) = \left(\frac{1}{4^{n-k}}\right)\left(\frac{3^k}{4^k}\right) = \frac{3^k}{4^n}.$$

Here k is the number of heads in x. If $P(x|\text{fair coin}) > P(x|\text{biased coin})$, then the dealer most likely used a fair coin; on the other hand, we can see that if $P(x|\text{fair coin}) < P(x|\text{biased coin})$, then the dealer most likely used a biased coin. The probabilities $P(x|\text{fair coin}) = \frac{1}{2^n}$ and $P(x|\text{biased coin}) = \frac{3^k}{4^n}$ become equal at $k = \frac{n}{\log_2 3}$. As a result, when $k < \frac{n}{\log_2 3}$, the dealer most likely used a fair coin, and when $k > \frac{n}{\log_2 3}$, he most likely used a biased coin. We can define the *log-odds ratio* as follows:

$$\log_2 \frac{P(x|\text{fair coin})}{P(x|\text{biased coin})} = \sum_{i=1}^{k} \log_2 \frac{p^+(x_i)}{p^-(x_i)} = n - k \log_2 3$$

However, we know that the dealer *does* change coins, albeit rarely. One approach to making an educated guess as to which coin the dealer used at each point would be to slide a window of some width along the sequence of coin flips and calculate the log-odds ratio of the sequence under each window. In effect, this is considering the log-odds ratio of short regions of the sequence. If the log-odds ratio of the short sequence falls below 0, then the dealer most likely used a biased coin while generating this window of sequence; otherwise the dealer most likely used a fair coin.

Similarly, a naive approach to finding CG-islands in long DNA sequences is to calculate log-odds ratios for a sliding window of some particular length, and to declare windows that receive positive scores to be potential CG-islands. Of course, the disadvantage of this approach is that we do not know the length of CG-islands in advance and that some overlapping windows may classify the same nucleotide differently. HMMs represent a different probabilistic approach to this problem.

2. The notation $P(x|y)$ is shorthand for the "probability of x occurring under the assumption that (some condition) y is true." The notation $\prod_{i=1}^{n} a_i$ means $a_1 \cdot a_2 \cdot a_3 \cdots a_n$.

11.2 The Fair Bet Casino and Hidden Markov Models

An HMM can be viewed as an abstract machine that has an ability to produce some output using coin tossing. The operation of the machine proceeds in discrete steps: at the beginning of each step, the machine is in a *hidden state* of which there are k. During the step, the HMM makes two decisions: (1) "What state will I move to next?" and (2) "What symbol—from an alphabet Σ—will I emit?" The HMM decides on the former by choosing randomly among the k states; it decides on the latter by choosing randomly among the $|\Sigma|$ symbols. The choices that the HMM makes are typically biased, and may follow arbitrary probabilities. Moreover, the *probability distributions*[3] that govern which state to move to and which symbols to emit change from state to state. In essence, if there are k states, then there are k different "next state" distributions and k different "symbol emission" distributions. An important feature of HMMs is that an observer can see the emitted symbols but has no ability to see what state HMM is in at any step, hence the name *Hidden* Markov Models. The goal of the observer is to infer the most likely states of the HMM by analyzing the sequences of emitted symbols. Since an HMM effectively uses dice to emit symbols, the sequence of symbols it produces does not form any readily recognizable pattern.

Formally, an HMM $\mathcal{M}$ is defined by an alphabet of emitted symbols Σ, a set of (hidden) states Q, a matrix of state transition probabilities A, and a matrix of emission probabilities E, where

- Σ is an alphabet of symbols;
- Q is a set of states, each of which will emit symbols from the alphabet Σ;
- $A = (a_{kl})$ is a $|Q| \times |Q|$ matrix describing the probability of changing to state l after the HMM is in state k; and
- $E = (e_k(b))$ is a $|Q| \times |\Sigma|$ matrix describing the probability of emitting the symbol b during a step in which the HMM is in state k.

Each row of the matrix A describes a "state die"[4] with $|Q|$ sides, while each row of the matrix E describes a "symbol die" with $|\Sigma|$ sides. The Fair

3. A probability distribution is simply an assignment of probabilities to outcomes; in this case, the outcomes are either symbols to emit or states to move to. We have seen probability distributions, in a disguised form, in the context of motif finding. Every column of a profile, when each element is divided by the number of sequences in the sample, forms probability distributions.
4. Singular of "dice."

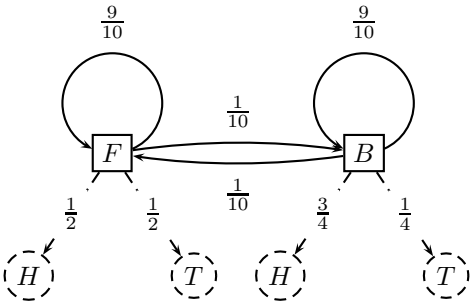

Figure 11.1 The HMM designed for the Fair Bet Casino problem. There are two states: F (fair) and B (biased). From each state, the HMM can emit either heads (H) or tails (T), with the probabilities shown. The HMM will switch between F and B with probability $1/10$.

Bet Casino process corresponds to the following HMM $\mathcal{M}(\Sigma, Q, A, E)$ shown in figure 11.1:

- $\Sigma = \{0, 1\}$, corresponding to tails (0) or heads (1)
- $Q = \{F, B\}$, corresponding to a fair (F) or biased (B) coin
- $a_{FF} = a_{BB} = 0.9$, $a_{FB} = a_{BF} = 0.1$
- $e_F(0) = \frac{1}{2}$, $e_F(1) = \frac{1}{2}$, $e_B(0) = \frac{1}{4}$, $e_B(1) = \frac{3}{4}$

A *path* $\pi = \pi_1 \dots \pi_n$ in the HMM $\mathcal{M}$ is a sequence of states. For example, if a dealer used the fair coin for the first three and the last three tosses and the biased coin for five tosses in between, the corresponding path π would be $\pi =$ FFFBBBBBFFF. If the resulting sequence of tosses is 01011101001, then the following shows the matching of x to π and the probability of x_i being generated by π_i at each flip:

$$\begin{matrix} x \\ \pi \\ P(x_i|\pi_i) \end{matrix} = \begin{pmatrix} 0 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 \\ F & F & F & B & B & B & B & B & F & F & F \\ \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & \frac{3}{4} & \frac{3}{4} & \frac{3}{4} & \frac{1}{4} & \frac{3}{4} & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \end{pmatrix}$$

We write $P(x_i|\pi_i)$ to denote the probability that symbol x_i was emitted from state π_i—these values are given by the matrix E. We write $P(\pi_i \to \pi_{i+1})$

to denote the probability of the transition from state π_i to π_{i+1}—these values are given by the matrix A.

The path $\pi =$ FFFBBBBBFFF includes only two switches of coins, first from F to B (after the third step), and second from B to F (after the eighth step). The probability of these two switches, $\pi_3 \to \pi_4$ and $\pi_8 \to \pi_9$, is $\frac{1}{10}$, while the probability of all other transitions, $\pi_{i-1} \to \pi_i$, is $\frac{9}{10}$ as shown below:[5]

$$\begin{matrix} x \\ \pi \\ P(x_i|\pi_i) \\ P(\pi_{i-1} \to \pi_i) \end{matrix} = \begin{pmatrix} 0 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 \\ F & F & F & B & B & B & B & B & F & F & F \\ \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & \frac{3}{4} & \frac{3}{4} & \frac{3}{4} & \frac{1}{4} & \frac{3}{4} & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{9}{10} & \frac{9}{10} & \frac{1}{10} & \frac{9}{10} & \frac{9}{10} & \frac{9}{10} & \frac{9}{10} & \frac{1}{10} & \frac{9}{10} & \frac{9}{10} \end{pmatrix}$$

The probability of generating x through the path π (assuming for simplicity that in the first moment the dealer is equally likely to have a fair or a biased coin) is roughly 2.66×10^{-6} and is computed as:

$$\left(\tfrac{1}{2} \cdot \tfrac{1}{2}\right)\left(\tfrac{1}{2} \cdot \tfrac{9}{10}\right)\left(\tfrac{1}{2} \cdot \tfrac{9}{10}\right)\left(\tfrac{3}{4} \cdot \tfrac{1}{10}\right)\left(\tfrac{3}{4} \cdot \tfrac{9}{10}\right)\left(\tfrac{3}{4} \cdot \tfrac{9}{10}\right)\left(\tfrac{1}{4} \cdot \tfrac{9}{10}\right)\left(\tfrac{3}{4} \cdot \tfrac{9}{10}\right)\left(\tfrac{1}{2} \cdot \tfrac{1}{10}\right)\left(\tfrac{1}{2} \cdot \tfrac{9}{10}\right)\left(\tfrac{1}{2} \cdot \tfrac{9}{10}\right)$$

In the above example, we assumed that we knew π and observed x. However, in reality we do not have access to π. If you only observe that $x = 01011101001$, then you might ask yourself whether or not $\pi =$FFFBBBBBFFF is the "best" explanation for x. Furthermore, if it is not the best explanation, is it possible to reconstruct the best one? It turns out that FFFBBBBBFFF is not the most probable path for $x = 01011101001$: FFFBBBFFFFF is slightly better, with probability 3.54×10^{-6}.

$$\begin{matrix} x \\ \pi \\ P(x_i|\pi_i) \\ P(\pi_{i-1} \to \pi_i) \end{matrix} = \begin{pmatrix} 0 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 \\ F & F & F & B & B & B & F & F & F & F & F \\ \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & \frac{3}{4} & \frac{3}{4} & \frac{3}{4} & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{9}{10} & \frac{9}{10} & \frac{1}{10} & \frac{9}{10} & \frac{9}{10} & \frac{1}{10} & \frac{9}{10} & \frac{9}{10} & \frac{9}{10} & \frac{9}{10} \end{pmatrix}$$

The probability that sequence x was generated by the path π, given the model $\mathcal{M}$, is

$$P(x|\pi) = P(\pi_0 \to \pi_1) \cdot \prod_{i=1}^{n} P(x_i|\pi_i) P(\pi_i \to \pi_{i+1}) = a_{\pi_0,\pi_1} \cdot \prod_{i=1}^{n} e_{\pi_i}(x_i) \cdot a_{\pi_i,\pi_{i+1}}.$$

5. We have added a fictitious term, $P(\pi_0 \to \pi_1) = \frac{1}{2}$ to model the initial condition: the dealer is equally likely to have either a fair or a biased coin before the first flip.

For convenience, we have introduced π_0 and π_{n+1} as the fictitious initial and terminal states $begin$ and end.

This model defines the probability $P(x|\pi)$ for a given sequence x and a given path π. Since only the dealer knows the real sequence of states π that emitted x, we say that π is *hidden* and attempt to solve the following Decoding problem:

Decoding Problem:
Find an optimal hidden path of states given observations.

Input: Sequence of observations $x = x_1 \ldots x_n$ generated by an HMM $\mathcal{M}(\sum, Q, A, E)$.

Output: A path that maximizes $P(x|\pi)$ over all possible paths π.

The Decoding problem is an improved formulation of the ill-defined Fair Bet Casino problem.

11.3 Decoding Algorithm

In 1967 Andrew Viterbi used an HMM-inspired analog of the Manhattan grid for the Decoding problem, and described an efficient dynamic programming algorithm for its solution. Viterbi's Manhattan is shown in figure 11.2 with every choice of $\pi_1, \ldots, \pi_n$ corresponding to a path in this graph. One can set the edge weights in this graph so that the product of the edge weights for path $\pi{=}\pi_1 \ldots \pi_n$ equals $P(x|\pi)$. There are $|Q|^2(n-1)$ edges in this graph with the weight of an edge from (k, i) to $(l, i+1)$ given by $e_l(x_{i+1}) \cdot a_{kl}$. Unlike the alignment approaches covered in chapter 6 where the set of valid directions was restricted to south, east, and southeast edges, the Manhattan built to solve the decoding problem only forces the tourists to move in any eastward direction (e.g., northeast, east, southeast, etc.), and places no additional restrictions (fig. 11.3). To see why the length of the edge between the vertices (k, i) and $(l, i+1)$ in the corresponding graph is given by $e_l(x_{i+1}) \cdot a_{kl}$, one should compare $p_{k,i}$ [the probability of a path ending in vertex (k, i)] with