1 Hidden Markov Models
for all pairs \((s, t) \in Q \times Q\), and \(\sum_{t=1}^{Q} a_{s,t} = 1\) for all states \(s \in Q\). For instance, in the fair bet casino we have that \(a_{\text{fair,biased}} = a_{\text{biased,fair}} = 0.1\) and that \(a_{\text{fair,fair}} = a_{\text{biased,biased}} = 0.9\).

The main use of a Markov model (or chain) is to evaluate the probability \(\Pr(\pi)\) of a state path \(\pi = \pi_1, \ldots, \pi_n\), where \(\pi_i \in Q\) for each \(i \in [n]\). Using the chain rule\(^1\), we have

\[
\Pr(\pi) = \Pr(\pi_1, \ldots, \pi_n) = \Pr(\pi_n | \pi_{n-1}, \ldots, \pi_1) \Pr(\pi_{n-1} | \pi_{n-2}, \ldots, \pi_1) \ldots \Pr(\pi_1). 
\]

The key property that we use in computing this probability is the Markov property, which states that the next state depends only on the current state and not any of the previous states. That is, \(\Pr(\pi_i | \pi_{i-1}, \ldots, \pi_1) = \Pr(\pi_i | \pi_{i-1}) = a_{\pi_{i-1}, \pi_i}\). Thus, we have

\[
\Pr(\pi) = \Pr(\pi_1, \ldots, \pi_n) = \Pr(\pi_n | \pi_{n-1}, \ldots, \pi_1) \Pr(\pi_{n-1} | \pi_{n-2}, \ldots, \pi_1) \ldots \Pr(\pi_1) \\
= \Pr(\pi_n | \pi_{n-1}) \Pr(\pi_{n-1} | \pi_{n-2}) \ldots \Pr(\pi_2 | \pi_1) \Pr(\pi_1) \\
= \Pr(\pi_1) \prod_{i=2}^{n} a_{\pi_{i-1}, \pi_i} \\
= a_{0, \pi_1} \prod_{i=2}^{n} a_{\pi_{i-1}, \pi_i}.
\]

Observe that \(a_{0,s}\) is the initial probability \(\Pr(\pi_1 = s)\) of starting in state \(s \in Q\). That is, \(a_{0,s} \geq 0\) for all \(s \in Q\) and \(\sum_{s=1}^{Q} a_{0,s} = 1\). Thus, \(A\) is an \((|Q| + 1) \times |Q|\) matrix. Setting \(\pi_0 = 0\), we rewrite the above equation as

\[
\Pr(\pi) = \prod_{i=1}^{n} a_{\pi_{i-1}, \pi_i}.
\]

**Hidden Markov model.** Given a state path \(\pi\) and Markov model \(M = (Q, A)\) it is trivial to compute \(\Pr(\pi)\). In practice, we are not given \(\pi = \pi_1, \ldots, \pi_n\) but rather a sequence \(x = x_1, \ldots, x_n\) of symbols that were emitted in each state \(\pi_i\). More formally, we are given a set \(\Sigma\) of symbols and emission probabilities \(E = [e_{s,k}]\) whose entries \(e_{s,k}\) indicate the probability of emitting symbol \(k\) in state \(s\). A hidden Markov model (HMM) is a four-tuple \(M = (Q, A, \Sigma, E)\). It is convenient to think of an HMM \(M = (Q, A, \Sigma, E)\) as generative model, given \(M\) and a state path \(\pi\), we generate a symbol sequence \(x = x_1, \ldots, x_n\) using emission probabilities \(E\). That is, the probability of generating symbol \(s\) in state \(\pi_i\) is precisely \(e_{\pi_i, s}\). Thus, the joint probability of a symbol sequence \(x\) and state path \(\pi\) is computed as

\[
\Pr(x, \pi) = \Pr(x_1, \pi_1, \ldots, x_n, \pi_n) \\
= \Pr(x_n, \pi_n | x_{n-1}, \pi_{n-1}, \ldots, x_1, \pi_1) \ldots \Pr(x_2, \pi_2 | x_1, \pi_1) \Pr(x_1, \pi_1) \\
= \Pr(x_n, \pi_n | \pi_{n-1}) \ldots \Pr(x_2, \pi_2 | \pi_1) \Pr(x_1 | \pi_1) \Pr(\pi_1) \\
= \prod_{i=1}^{n} e_{\pi_i, x_i} \cdot a_{\pi_{i-1}, \pi_i}.
\]

In practice, the state path \(\pi\) that generated the symbol sequence \(x\) is hidden to us (hence the name).

\(^1\)Repeated application of \(\Pr(A, B) = \Pr(A | B) \Pr(B)\).
Three questions. Given $\mathcal{M} = (Q, A, \Sigma, E)$, we are interested in the following three questions.

1. What is the most probable state path $\pi^*$ that generated a given symbol sequence $x$? That is, find $\pi^* = \arg \max_{\pi} \Pr(x, \pi)$.

2. What is the probability $\Pr(x) = \sum_\pi \Pr(x, \pi)$?

3. What is the posterior probability that the $i$-th observation came from state $s$ given the observed symbol sequence $x$? That is, compute $\Pr(\pi_i = s \mid x)$.

Using the fair bet casino example, question 1 would be to determine for each coin flip whether the crooked dealer used the fair or biased coin. An example of question 2, would be to compute the probability of observing $n$ times heads in a row (not caring about whether the dealer was cheating or not), i.e. $\Pr(x_1 = H, \ldots, x_n = H)$. Finally, question 3 seeks to answer whether the dealer cheated at time $i$ given $n$ heads/tails observations $x$, i.e. $\Pr(\pi_i = B \mid x)$. We will solve these three questions using different algorithms. The key observation is that the Markov property leads to optimal substructure in each of these questions, thus enabling the use of dynamic programming.

2 Viterbi Algorithm: Most Probable State Path

We are interested in identifying $\pi^* \in Q^n$ with maximum likelihood $\Pr(x, \pi^*)$. From (7), we immediately observe optimal substructure. Thus, we define $x_i = x_1, \ldots, x_i$, $\pi^*_i = \pi^*_1, \ldots, \pi^*_i$. Let $v[s, i]$ denote the probability $\Pr(x_i, \pi_i = s, \pi^*_{i-1})$, i.e. the probability of the most probable state path of the first $i$ observations with final state $\pi_i = s$. Clearly,

$$\Pr(x, \pi^*) = \max_{s \in Q} \Pr(x_n, \pi_n = s, \pi^*_n) = \max_{s \in Q} v[s, n].$$

(11)

Initially, $v[s, 1] = a_{0,s}e_{s,x_1}$ for all states $s \in Q$. The following recurrence follows from the Markov property and (7).

$$v[s, i] = \begin{cases} a_{0,s}e_{s,x_1}, & \text{if } i = 1, \\ e_{s,x_i} \max_{t \in Q} v[t, i - 1] \cdot a_{t,s}, & \text{if } i > 1. \end{cases}$$

(12)

We view $v$ as an $|Q| \times (n + 1)$ table, where the entries of the first column are 1. From the recurrence, we see that the rest of this table can be filled out column-by-column. Each entry in column $i > 0$ requires lookups of all $|Q|$ entries in the preceding column $i - 1$. Thus, the running time is $O(n|Q|^2)$. We obtain the final maximum likelihood $\Pr(x, \pi^*)$ and hidden state $\pi_n$ from the completed table by scanning through the last column $n$ and identifying the state $s \in Q$ with largest likelihood $v[s, n]$ and setting $\pi_n = s$. We obtain the corresponding maximum likelihood state path $\pi^*$ by back tracking from $(\pi_n, n)$. This algorithm is known as the Viterbi algorithm.
3 Forward Algorithm

We are interested in the probability \( \Pr(\mathbf{x}) \), the probability of observing a given symbol sequence \( \mathbf{x} = x_1, \ldots, x_n \) marginalizing over all combinations of hidden states \( \pi = \pi_1, \ldots, \pi_n \). That is,

\[
\Pr(\mathbf{x}) = \sum_{\pi} \Pr(\mathbf{x}, \pi) = \sum_{(\pi_1, \ldots, \pi_n)} \Pr(x_1, \pi_1, \ldots, x_n, \pi_n) = \sum_{(\pi_1, \ldots, \pi_n)} \prod_{i=1}^{n} e_{\pi_i, x_i} \cdot a_{\pi_{i-1}, \pi_i}.
\]  

(13)

There are \( |Q|^n \) possible state paths \( \pi = (\pi_1, \ldots, \pi_n) \). Thus, computing \( \Pr(\mathbf{x}) \) by brute force takes \( O(n|Q|^n) \) time, where the linear factor \( n \) is the time required to compute \( \Pr(\mathbf{x}, \pi) \). Can we do better than exponential time?

Alternatively, we can approximate \( \Pr(\mathbf{x}) \) by the probability \( \Pr(\mathbf{x}, \pi^*) \) where \( \pi^* \) is the state path identified by the Viterbi algorithm in \( O(n|Q|^2) \) time. The underlying assumption is that most of the probability mass is contributed by the most probable state path \( \pi^* \). However, this is an approximation. We will describe an exact algorithm for computing \( \Pr(\mathbf{x}) \) with the same running time of \( O(n|Q|^2) \) as the Viterbi algorithm.

Recall that \( \mathbf{x}_i = x_1, \ldots, x_i, \pi_i = \pi_1, \ldots, \pi_i \). Let \( f[s, i] \) denote the probability of observing \( \mathbf{x}_i = (x_1, \ldots, x_i) \) given that \( \pi_i = s \). That is,

\[
\Pr(\mathbf{x}_i, \pi_i = s) = \Pr(x_1, \ldots, x_i, \pi_i = s) = f[s, i].
\]  

(14)

Clearly,

\[
\Pr(\mathbf{x}) = \Pr(x_1, \ldots, x_n) = \sum_{s \in Q} \Pr(x_1, \ldots, x_n, \pi_n = s) = \sum_{s \in Q} f[s, n].
\]  

(15)

Recalling that \( \mathbf{x}_1 = x_1 \) is the singleton sequence, we have that \( f[s, 1] = \Pr(x_1, \pi_1 = s) = a_{0,s} e_{s,x_1} \) initially. We derive the step \( i > 1 \) as follows.

\[
f[s, i] = \Pr(\mathbf{x}_i, \pi_i = s) = \Pr(x_1, \ldots, x_i, \pi_i = s)
\]

(16)

\[
= \sum_{(\pi_1, \ldots, \pi_{i-1})} \Pr(x_1, \ldots, x_{i-1}, x_i, \pi_1, \ldots, \pi_{i-1}, \pi_i = s)
\]

(17)

\[
= \sum_{(\pi_1, \ldots, \pi_{i-1})} \Pr(x_1, \ldots, x_{i-1}, \pi_1, \ldots, \pi_{i-1}, \pi_i = s) \cdot e_{s,x_i}
\]

(18)

\[
= \sum_{t \in Q} \sum_{(\pi_1, \ldots, \pi_{i-1})} \Pr(x_1, \ldots, x_{i-1}, \pi_1, \ldots, \pi_{i-2}, \pi_{i-1} = t) \cdot a_{t,s} \cdot e_{s,x_i}
\]

(19)

\[
= \sum_{t \in Q} \Pr(x_1, \ldots, x_{i-1}, \pi_{i-1} = t) \cdot a_{t,s} \cdot e_{s,x_i}
\]

(20)

\[
= \sum_{t \in Q} f[t, i-1] \cdot a_{t,s} \cdot e_{s,x_i}
\]

(21)

\[
= e_{s,x_i} \sum_{t \in Q} f[t, i-1] \cdot a_{t,s}
\]

(22)
Hence, we have the following recurrence.

\[
    f[s, i] = \begin{cases} 
    a_{0,s,e_{s,x_1}}, & \text{if } i = 1, \\
    e_{s,x_i} \sum_{t \in Q} \{f[t, i - 1] \cdot a_{t,s}\}, & \text{if } i > 1.
    \end{cases}
\] (23)

We compute \( f \) in exactly the same way as \( v \), thus requiring \( O(n|Q|^2) \) time. The probability \( \Pr(x) \) that we seek is obtained by summing the entries in the last column, i.e. \( \Pr(x) = \sum_{s \in Q} f[s, n] \).

## 4 Backward Algorithm

How do we compute the posterior probability \( \Pr(\pi_i = s \mid x) \)? From probability theory, we have

\[
    \Pr(\pi_i = s \mid x) = \frac{\Pr(x, \pi_i = s)}{\Pr(x)}
\] (24)

We know how to compute \( \Pr(x) \) using the forward algorithm, but how do we compute \( \Pr(x, \pi_i = s) \)? We have that

\[
    \Pr(x, \pi_i = s) = \Pr(x_1, \ldots, x_n, \pi_i = s)
\] (25)

\[
    = \Pr(x_1, \ldots, x_i, \pi_i = s) \Pr(x_{i+1}, \ldots, x_n \mid x_1, \ldots, x_i, \pi_i = s)
\] (26)

\[
    = f[s, i] \cdot \Pr(x_{i+1}, \ldots, x_n \mid \pi_i = s).
\] (27)

Let \( b[s, i] \) denote the probability \( \Pr(x_{i+1}, \ldots, x_n \mid \pi_i = s) \). Thus,

\[
    \Pr(x, \pi_i = s) = f[s, i] \cdot b[s, i].
\] (28)

Initially, \( b[s, n] = \Pr(\emptyset \mid \pi_n = s) = \Pr(\emptyset, \pi_n = s) / \Pr(\pi_n = s) = 1 \). As for the step, we have

\[
    b[s, i] = \Pr(x_{i+1} \mid \pi_i = s) = \Pr(x_{i+1}, \ldots, x_n \mid \pi_i = s)
\] (29)

\[
    = \sum_{(\pi_{i+1}, \ldots, \pi_n)} \Pr(x_{i+1}, \ldots, x_n, \pi_{i+1} = t, \ldots, \pi_n = s)
\] (30)

\[
    = \sum_{t \in Q} \sum_{(\pi_{i+2}, \ldots, \pi_n)} \Pr(x_{i+1}, \pi_{i+1} = t, x_{i+2}, \ldots, x_n, \pi_{i+2}, \ldots, \pi_n \mid \pi_i = s)
\] (31)

\[
    = \sum_{t \in Q} \sum_{(\pi_{i+2}, \ldots, \pi_n)} \Pr(x_{i+1}, \pi_{i+1} = t \mid \pi_i = s) \Pr(x_{i+2}, \ldots, x_n, \pi_{i+2}, \ldots, \pi_n \mid \pi_i = s)
\] (32)

\[
    = \sum_{t \in Q} \sum_{(\pi_{i+2}, \ldots, \pi_n)} \Pr(x_{i+1}, \pi_{i+1} = t \mid \pi_i = s) \Pr(x_{i+2}, \ldots, x_n, \pi_{i+2}, \ldots, \pi_n \mid \pi_i = s, x_{i+1}, \pi_i = t)
\] (33)

\[
    = \sum_{t \in Q} \sum_{(\pi_{i+2}, \ldots, \pi_n)} \Pr(x_{i+1}, \pi_{i+1} = t \mid \pi_i = s) \Pr(x_{i+2}, \ldots, x_n, \pi_{i+2}, \ldots, \pi_n \mid \pi_{i+1} = t)
\] (34)

\[
    = \sum_{t \in Q} a_{s,t} \cdot e_{t, x_{i+1}} \Pr(x_{i+2}, \ldots, x_n \mid \pi_{i+1} = t)
\] (35)

\[
    = \sum_{t \in Q} a_{s,t} \cdot e_{t, x_{i+1}} \cdot b[t, i + 1]
\] (36)
Thus, we have the following recurrence.

\[
b[s, i] = \begin{cases} 
1, & \text{if } i = n, \\
\sum_{t \in Q} a_{s, t} \cdot e_{t, x_{i+1}} \cdot b[t, i + 1], & \text{if } 1 \leq i < n,
\end{cases}
\]  

(37)

Hence, the posterior probability is given by

\[
\Pr(\pi_i = s \mid \mathbf{x}) = \frac{\Pr(\mathbf{x}, \pi_i = s)}{\Pr(\mathbf{x})} = \frac{f[s, i] \cdot b[s, i]}{\sum_{s' \in Q} f[s', n]}.
\]  

(38)

This is the backward algorithm. We note that we can solve the data likelihood problem using the backward algorithm as follows.

\[
\Pr(\mathbf{x}) = \sum_{s \in Q} a_{0, s} \cdot e[s, x_1] \cdot b[s, 1].
\]  

(39)