Note 25. Hidden Markov models*

Alex Fu

Fall 2022

1 Introduction

The hidden Markov model is a combination of Markov chains and Kalman filters, two models of stochastic dynamical systems we have already seen, both specific examples of *dynamic Bayesian networks*, which encode evolution over time, (conditional) dependencies, and partial information.

Definition 1 (Hidden Markov model).

A **hidden Markov model** (HMM) consists of a discrete-time Markov chain $(X_n)_{n\in\mathbb{N}}$ of *hidden* or *latent states*, together with a sequence of *observations* $(Y_n)_{n\in\mathbb{N}}$ whose values are given by the observation probabilities $q(x_n, y_n) := \mathbb{P}(Y_n = y_n \mid X_n = x_n)$.

The following familiar graphical model implicitly describes that Y_n only depends on X_n, in that Y_n is conditionally independent of all other random variables in the model when given X_n.

$$\begin{tikzcd}
X_0 \arrow[r] \arrow[d] & X_1 \arrow[r] \arrow[d] & X_2 \arrow[r] \arrow[d] & X_3 \arrow[r] \arrow[d] & \cdots \arrow[d] \\
Y_0 & Y_1 & Y_2 & Y_3 & \cdots
\end{tikzcd}$$

Proposition 1 (Probability of a sequence of states and observations).

Let π_n be the state distribution at time n, let $p(x_{n-1}, x_n)$ be the transition probabilities, and let $q(x_n, y_n)$ be the observation probabilities. Then

$$\mathbb{P}(X^{(0:n)} = x^{(0:n)}, Y^{(0:n)} = y^{(0:n)}) = \pi_0(x_0)q(x_0, y_0)\prod_{i=1}^{n} p(x_{i-1}, x_i)q(x_i, y_i).$$

Hidden Markov models allow for a few common forms of inference:

a. **Filtering**: finding $\hat{X}_n$ given $Y_0, \dots, Y_n$, as with tracking position given measurements, or monitoring health given a history of symptoms.

b. **Prediction**: finding $\hat{X}_{n+1}$ given $Y_0, \dots, Y_n$, as with radar tracking, stock price predictions, or predictive coding.

c. **Smoothing**: finding $\hat{X}_k$, $k \le n$, given $Y_0, \dots, Y_n$, as with inferring the cause of an accident or post-mortem analysis.

In particular, we will focus on the following inference problem, which finds applications in speech recognition, autocorrection, and convolutional coding.

Definition 2 (Maximum likelihood sequence estimation).

The **maximum likelihood sequence estimator** (MLSE) of given $Y_0, \dots, Y_n$ is

$$\hat{X}^{(0:n)} := \mathsf{MAP}(X^{(0:n)} \mid Y^{(0:n)} = y^{(0:n)}),$$

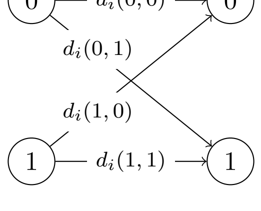
the *sequence* of states $\hat{X}_0, \dots, \hat{X}_n$ that best explains the observations, which is in contrast to the estimation of individual states done by smoothing.

Expanding the definition, we can reduce finding the MLSE to the following optimization problem:

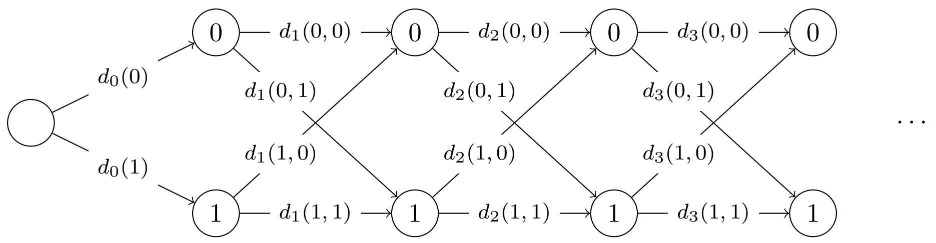
$$\begin{aligned}
\hat{X}^{(0:n)} &= \underset{x^{(0:n)}}{\operatorname{argmax}} \left[\pi_0(x_0) q(x_0, y_0) \prod_{i=1}^{n} p(x_{i-1}, x_i) q(x_i, y_i) \right] \\
&= \underset{x^{(0:n)}}{\operatorname{argmin}} \left[-\log(\pi_0(x_0) q(x_0, y_0)) - \sum_{i=1}^{n} \log(p(x_{i-1}, x_i) q(x_i, y_i)) \right] \\
&:= \underset{x^{(0:n)}}{\operatorname{argmin}} \left[d_0(x_0) + \sum_{i=1}^{n} d_i(x_{i-1}, x_i) \right]
\end{aligned}$$

where $d_0(x_0)$ and $d_i(x_{i-1}, x_i)$ are defined to be positive. The naïve algorithm that iterates over all possible sequences in $O(|S|^n)$ time is computationally infeasible, but the **Viterbi algorithm**, which uses dynamic programming on a *trellis diagram*, is quite efficient.

For simplicity, let $S = \{0, 1\}$ consist of just two states. A single stage of the **trellis** diagram is the following directed acyclic graph with positive edge weights:



The full trellis diagram can be thus populated in $O(|S|^2 n)$ time:



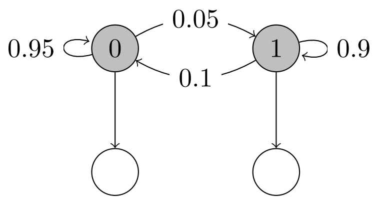
The dummy starting node allows us to account for the weight of the initial state in the sequence. We remind ourselves that the edge weights are computed explicitly as

$$d_0(x_0) = -\log(\pi_0(x_0)q(x_0, y_0))$$
$$d_i(x_{i-1}, x_i) = -\log(p(x_{i-1}, x_i)\log q(x_i, y_i)).$$

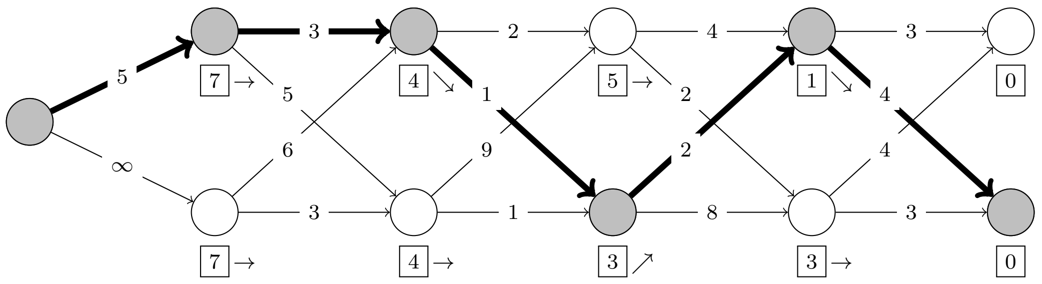
With a populated trellis, a dynamic programming algorithm such as **Bellman–Ford** can then be used to find the shortest path in $O(|S|\,n)$ time, which determines the MLSE.

Example 1 (Nearly honest casino).

Suppose that a casino switches between a fair die and a loaded die, represented by the states 0 and 1 respectively, according to the following Markov chain.



For simplicity, we assign integer edge weights below. We are told the casino will start with a fair die, so $d_0(1) = \infty$. The boxed weight below a node is the length of the shortest path from the node to the final stage, and the arrow gives the optimal next transition.



Other optional topics we invite you to explore include expectation maximization (EM) and linear-quadratic-Gaussian (LQG) control. However, for us, hidden Markov models will mark the end of EECS 126 course notes. We hope that they have been a helpful reference to you this semester.

■