Reading for Lecture 8

Release v10

Christopher Lee

October 19, 2011

Contents

Read Jones & Pevzner sections 11.1 - 11.3, and the following material.

1 Markov Processes

We first define Markov processes in terms of a sequence of observable variables $X_1, X_2, X_3, ...X_n$, as a *Markov chain*. We then introduce the problem of infering a sequence of hidden variables from an associated sequence of observable variables; this is called a *hidden Markov model*, commonly abbreviated as "HMM".

1.1 The Markov Property

Recall that the general conditional probability chain rule always permits us to factor a joint probability $p(X_1, X_2, X_3, ...X_n)$ into n factors of the form $p(X_t|X_1, X_2, ..., X_{t-1})$ where term t of the factorization depends on *all* of the variables that preceded it in the factorization. You could say that each variable "remembers" *every* previous variable, in the sense that it depends directly on all previous variables.

The *Markov property* is when the variables in a process each depend only a *fixed* number of other variables (by default, only one). In other words, the joint probability can be written as the product of a set of conditional probabilities

$p(X_u|X_t)$. This is sometimes called a "memoryless" process to reflect the fact that this conditioning has no direct memory of the rest of the preceding variables.

1.2 Markov Chains

The simplest version of such a process is simply a linear chain of variables that obey the Markov property:

$$p(X_1, X_2, X_3, ...X_n) = p(X_1)p(X_2|X_1)p(X_3|X_2)...p(X_n|X_{n-1})$$

This is called a *Markov chain*. It follows that any pair of variables X_t, X_v separated by at least one intervening variable (i.e. $v > t+1$) are conditionally independent given any variable X_u between them (i.e. $t < u < v$). Thus we can represent a Markov chain by an information graph whose nodes are the X_t variables, with edges drawn between pairs of nodes that have a conditional dependency relationship (and no edge between nodes that are conditionally independent), e.g.

$$X_1 \rightarrow X_2 \rightarrow X_3 \rightarrow ... \rightarrow X_{n-1} \rightarrow X_n$$

We can consider a Markov chain to represent a stochastic process that traverses a series of states one at a time, each with a probability $p(X_t|X_{t-1})$. Note that by default we will index the variables in a Markov chain with the letter *t*, implying a *time* index.

The joint probability $p(X_1, X_2, X_3, ...X_n)$ represents a complete traversal of this chain, in which each variable X_t adopts a particular value according to the Markov chain probabilities. Thus we can refer to a particular sequence of values of $X_1, X_2, X_3, ...$ as a *path* through the Markov model, which we will symbolize as a vector $\vec{X}$. We can also show that the Markov chain property allows the joint probability to be re-expressed in terms of the reverse conditional probabilities:

$$p(X_1, X_2, X_3, ...X_n) = p(X_n)p(X_{n-1}|X_n)p(X_{n-2}|X_{n-1})...p(X_1|X_2)$$

Discrete Markov chain models: State Graphs

One major class of Markov models is the case where the variables X_t are restricted to a set of discrete values (often referred to as the allowed *states* of the model) $s_1, s_2, ...s_m$. Then we can represent a Markov chain as a *state graph*, a directed graph whose nodes are the states $s_1, s_2, ...s_m$ and whose edges represent the non-zero conditional probabilities $p(s_j|s_i)$, which are called the "transition probabilities" of the model.

Note that a state graph is a completely different concept from an information graph: whereas each node in an information graph is a *random variable*, a node in a state graph is a state; a set of nodes in a state graph thus represents all the possible states of a *single variable*. Be careful not to mix up state graphs vs. information graphs.

START and END States

Computationally, it is convenient to introduce explicit START and END states into the state graph representing a Markov chain.

- The transition probabilities shown by a state graph without a START node do not include the *prior* probabilities $p(X_1 = s_i)$. Rather than creating a separate matrix and computations for handling these prior probabilities, it is convenient to simply add an explicit START state to our state graph as follows:
  - $p(s_j|\text{START}) = p(X_1 = s_i)$, in other words the transition probabilities from START store our priors.

- $p(START|s_j) = 0$, in other words no state can ever go back to START.
- Setting $p(X_0 = \text{START}) = 1$ enables us to calculate $p(\vec{X}^1)$ exactly like we calculate any $p(\vec{X}^t)$, i.e. $p(\vec{X}^t) = p(\vec{X}^{t-1})p(X_t|X_{t-1})$

- Adding an explicit END state serves the analogous purpose as START, on the other end of the Markov chain. This is particularly useful for variable length Markov chains (to be discussed later). The basic rules are:
  - $p(s_j|\text{END}) = 0$, i.e. once the Markov chain goes to END, it nevers goes to any further state. The Markov chain is terminated.
  - $p(\text{END}|s_j)$ represents the probability of terminating the Markov chain at any given state s_j.

  We will discuss this more in a later section on variable length Markov models.

Transition Matrices

A *homogeneous Markov chain* uses the same set of transition probabilities at all time steps u:

$$p(X_{u+1} = s_j|X_u = s_i) = p(s_j|s_i) = \tau_{ij}$$

Therefore

$$p(X_{u+1} = s_j) = \sum_i p(X_{u+1} = s_j|X_u = s_i)p(X_u = s_i) = \sum_i p(X_u = s_i)\tau_{ij}$$

Since this exactly matches the definition of multiplying a row vector by a square matrix, it is convenient to represent the transition probabilities as a matrix whose elements are just the τ_{ij}

$$T = \begin{bmatrix} \tau_{11} & \tau_{12} & \cdots & \tau_{1m} \\ \tau_{21} & \tau_{22} & \cdots & \tau_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ \tau_{m1} & \tau_{m2} & \cdots & \tau_{mm} \end{bmatrix}$$

and the state probabilities at time u as a row vector

$$\vec{p}_u = \begin{bmatrix} p(X_u = s_1) & p(X_u = s_2) & \cdots & p(X_u = s_m) \end{bmatrix}$$

enabling us to rewrite the transition operation as

$$\vec{p}_{u+1} = \vec{p}_u T$$

Applying this to two successive time steps, we obtain:

$$\vec{p}_{u+2} = (\vec{p}_u T)T = \vec{p}_u T^2$$

or in general

$$\vec{p}_{u+t} = \vec{p}_u T^t$$

In other words, the transition matrix for t time steps is simply T^t.

The Stationary Distribution

What happens after longer and longer times, i.e. as $t \to \infty$? Even if the Markov chain starts in a single fixed state, e.g. $p(X_1 = s_4) = 1$, after just one step there will be uncertainty about the state, i.e. $p(X_1 = s_4) < 1$ (assuming the $\tau_{ij} \neq 1$). Each successive application of the transition matrix will randomize the state more and more. Over time we expect it to converge to a "completely random" distribution. The question is, what exactly does "completely random" mean for our specific transition matrix T? We can immediately write a condition for a *stationary distribution* $\vec{\pi}$ that will be unchanged by T:

$$\vec{\pi} = \vec{\pi}T$$

$$\pi_j = \sum_i \pi_i \tau_{ij}$$

$$\sum_i \pi_j \tau_{ji} = \sum_i \pi_i \tau_{ij}$$

$$\sum_{i \neq j} \pi_j \tau_{ji} = \sum_{i \neq j} \pi_i \tau_{ij}$$

This is refered to as the *balance equation*, since it requires that the total probability exiting any state j must match the total probability entering state j. The stationary distribution $\vec{\pi}$ is unique for a given transition matrix T if under that matrix it is possible to get from any state i to any other state j and the chain is aperiodic (i.e. it is possible for a step to stay in any state, rather than being forced to go on a cycle of some minimum number of steps before returning to that state). Thus, *all* starting distributions will converge to the same stationary distribution. This implies that the t-step transition matrix T^t must converge to the stationary distribution in the following way:

$$\lim_{t \to \infty} T^t = \begin{bmatrix} \pi_1 & \pi_2 & \cdots & \pi_m \\ \pi_1 & \pi_2 & \cdots & \pi_m \\ \vdots & \vdots & \ddots & \vdots \\ \pi_1 & \pi_2 & \cdots & \pi_m \end{bmatrix}$$

In other words, every row of T^t converges to $\vec{\pi}$.

Solving for the Stationary Distribution

For an exact solution, we can directly solve the balance equations. For example, for the two-state problem this is simply

$$\pi_1 = \pi_1 \tau_{11} + \pi_2 \tau_{21} = \pi_1(1 - \tau_{12}) + (1 - \pi_1)\tau_{21}$$

So

$$\pi_1 = \frac{\tau_{21}}{\tau_{12} + \tau_{21}}$$

Alternatively, for an approximate solution, we can simply compute T^t for a sufficiently high value of t to give convergence. We can check convergence not only by comparing different values of t, but also by comparing the different rows of T^t, which should converge to the same row vector.

Reversibility

We consider a Markov chain to obey *reversibility*, if when it has reached its equilibrium (stationary) distribution, there is no way to distinguish a forward sequence of events (i.e. $X_t, X_{t+1}, X_{t+2}, X_{t+3}, ...$) from a reverse sequence of events (i.e. $...X_{t+3}, X_{t+2}, X_{t+1}, X_t$). This is true if the joint probability of any consecutive pair of states i, j is equal to the joint probability of the reverse sequence:

$$p(X_t = i, X_{t+1} = j) = p(X_t = j, X_{t+1} = i)$$

$$p(X_t = i)p(X_{t+1} = j|X_t = i) = p(X_t = j)p(X_{t+1} = i|X_t = j)$$

$$\pi_i \tau_{ij} = \pi_j \tau_{ji}$$

This is called the *detailed balance equation*, because it immediately implies the general balance equation.

- reversible Markov processes are very important for evolution and phylogeny algorithms.
- The detailed balance equation is the foundation for a wide range of computational sampling methods such as Gibbs sampling and Markov chain Monte Carlo (MCMC) algorithms.

1.3 The Viterbi Algorithm

The key feature of a Markov chain is that it dramatically reduces the computational complexity of the fundamental inference operation of searching over all possible states. For example, say we want to find the path $\vec{X}^*$ that has maximum probability, i.e. $p(\vec{X}^*) \geq p(\vec{X})$ for all other paths $\vec{X}$. For the fully general form of $p(x_1, x_2, x_3, ...x_n)$, the only way to answer this question is exhaustive search of the entire volume of the n-dimensional vector space defined by $x_1, x_2, x_3, ...x_n$. For example, if the probability density was simply random, even after comparing every point in the space except one, we still don't know whether our "best point so far" actually has maximum likelihood (the last point might have a higher probability).

By contrast, for a linear Markov chain, we can employ a simple algorithm to find the maximum. For any path $\vec{X}$ that contains an edge $X_{t-1} \rightarrow X_t$ such that $p(X_t|X_{t-1})p(X_1, X_2, ...X_{t-1}) < p(X_t|X'_{t-1})p(X_1, X_2, ...X'_{t-1})$ (i.e. the path $X_1 \rightarrow X_2 \rightarrow ... \rightarrow X_{t-1} \rightarrow X_t$ is not the maximum probability path to X_t), it follows by definition that $\vec{X}$ cannot be $\vec{X}^*$. Thus we can safely exclude any suboptimal $X_{t-1} \rightarrow X_t$; this enables us to convert the *global optimality condition* $p(\vec{X}^*) \geq p(\vec{X})$ into a purely *local optimality condition* which we can apply iteratively. Specifically, for each node X_t we define an associated maximum probability path to each of its possible states:

$$V_{ti} = \text{Max}\left[p(X_t = s_i|X_{t-1} = s_j)V_{t-1,j}\right]$$

where the maximimization is performed over all possible values of X_{t-1} (indexed here by j). This maximization simply applies our local optimality condition. V_{ti} stores the probability of the maximum likelihood path up to $X_t = s_i$; our optimality condition guarantees that there is no path with higher probability than this path. We also define a reverse-mapping $R_{ti} = j^*$, where j^* is the value of X_{t-1} that yielded the maximum for the associated function V_{ti}. We define $V_{0,START} = 1$. Since each each V_{ti} depends on its predecessor $V_{t-1,j}$ we compute them in forward order $V_1, V_2, ...V_n$. Finally we find the state i^* with maximum $V_{n,i^*} \geq V_{n,i}$ for all other values $i \neq i^*$. Then we obtain the rest of $\vec{X}^*$ via backtracking $X^*_{t-1} = R_{t,i^*}$. *Proof*: by definition, $\vec{X}^*$ always passes the local optimality test, and therefore is included in each V_t.

Because V_{ti} is defined in terms of itself, this algorithm is *recursive*. In general, this strategy of defining a solution of a problem in terms of the solution to each possible sub-problem is known as "dynamic programming". We will examine it in more detail later. This specific application of dynamic programming to find the maximum likelihood path is called the *Viterbi algorithm*.

Computational complexity principles

Whereas the computational complexity of the fully general global optimality condition is $O(|\vec{X^n}|)$ (where $|\vec{X^n}|$ is the volume of the n-dimensional vector space), the computational complexity of this dynamic programming solution is just $O\left((n-1)|X|^2\right)$, because to compute the:math:*V_{ti}* we must generate all possible combinations of (X_t, X_{t-1}), and we must compute $n-1$ of the V_t functions. Thus the Markov chain property reduces the n-dimensional general problem to just a 2-dimensional problem. We should emphasize several points:

- Even with this drastic reduction in computational complexity, all of the $O(|\vec{X^n}|)$ possible paths $\vec{X}$ were actually considered by this computation. The optimality condition guarantees that none of these paths has higher probability than the $\vec{X}^*$ solution obtained by this algorithm.
- This reveals a general principle: *conditional independence is dimensional reduction*. That is, by making all variables X_t, X_u conditionally independent except adjacent pairs of variables X_t, X_{t+1}, the effective dimensionality of the problem was reduced from n to just 2.
- Without the Markov chain property, we would not have been able to prove global optimality $p(X^*_{t+1}|X^*_t)p(X^*_1, X^*_2, ...X^*_t) \geq p(X_1, X_2, ...X_t, X_{t+1})$ for all possible values $X_1, X_2, ...X_t, X_{t+1}$. Why not? Under the general chain rule we can still get a separate factor for X_{t+1}, but it depends on *all* the previous variables:

$$p(X_1, X_2, ...X_t, X_{t+1}) = p(X_{t+1}|X_1, X_2, ...X_t)p(X_1, X_2, ...X_t)$$

  So under the general chain rule, searching for the optimal path to a given state of X_{t+1} would take $O(m^t)$ time instead of just $O(m^2)$ time.
- Note that the one-to-one mapping function f as defined above only records a single "maximum value". In the case of multiple, equal maximum values, only one will be recorded. If we wish to record multiple maxima, we can amend our definition of f to do so.

2 Hidden Markov Models

We can apply Markov chains to inference problems, in the form of a Hidden Markov Model (often abbreviated "HMM"). We assume that the actual model states are hidden (i.e. not directly observable), but themselves are coupled to observable variables via an "emission probability". That is, given a particular hidden state θ_t, we define the probability of observing some observable state O, $p(O|\theta_t)$, as the "emission probability" for that hidden state. Then, assuming that each hidden state θ "emits" a single observable variable O, the probability of some sequence of observations $\vec{O}^n = O_1, O_2, ...O_n$ depends on two things: $p(\theta_1, \theta_2, ...\theta_n)$ representing the probability of different paths through the hidden states; and $p(O_1, O_2, ...O_n|\theta_1, \theta_2, ...\theta_n)$. Treating $\vec{\theta}^n = \theta_1, \theta_2, ...\theta_n$ as a Markov chain,

$$p(\vec{O}^n) = \sum_{\vec{\theta}^n} p(\vec{O}^n, \vec{\theta}^n)$$

$$= \sum_{\theta_1} \sum_{\theta_2} ... \sum_{\theta_n} p(\theta_1)p(O_1|\theta_1)p(\theta_2|\theta_1)p(O_2|\theta_2)...p(\theta_n|\theta_{n-1})p(O_n|\theta_n)$$

2.1 Example: the Occasionally Dishonest Casino

Problem: A casino sometimes cheats, by using a dice that gives much higher probability for rolling a six than for any other outcome. However, to evade detection of their cheating, the casino randomly switches back and forth between

the biased dice and a completely fair dice with a certain probability (for further description of this problem, see Durbin). This problem behaves like a Hidden Markov Model: there are two possible hidden states (Fair vs. Biased dice), which emit observations (rolls of the dice). Given observation likelihoods, $p(o|F) = \frac{1}{6}$ for all o, $p(6|B) = \frac{1}{2}$ and $p(o|B) = \frac{1}{10}$ for all $o \neq 6$, and transition probabilities $p(B|F) = p(F|B) = 0.1$, we can use the Viterbi solution above to find the maximum likelihood sequence of hidden states $\theta_1, \theta_2, ...\theta_n$ for any given series of rolls $O_1, O_2, ...O_n$ observed at the casino.

2.2 The Viterbi Algorithm on a HMM

To find the *maximum likelihood path* by Viterbi we define

$$V_{ti} = \text{Max}\left[p(\theta_t = s_i|\theta_{t-1} = s_j)p(O_t|\theta_t = s_i)V_{t-1,j}\right]$$

where the maximization is performed over all possible values θ_{t-1} (indexed here by j), and proceed as before.