An Introduction to Hidden Markov Models

Hidden Markov Models (HMMs) are commonly used in many real world applications, including speech recognition, gesture recognition, score following, as well as many other temporal pattern recognitions problems.

So what exactly is a Hidden Markov Model (HMM)? In order to understand HMMs we must first examine a more basic premise---the Markov process. A Markov Process is any process in which the current state depends only on the previous N states of a sequence. Traffic lights provide a good example. The sequence of a simple set of traffic lights is: red - green - amber - repeat. If the current state is green, the next state will be amber. Thus, traffic lights represent a first order Markov process. The case of traffic lights is deterministic (the state transitions are known with absolute certainty), which isn't very interesting from a modeling perspective. We are more interested in modeling stochastic processes using Markov models. For example, the weather is a stochastic process (there is a 60% chance that it will be sunny tomorrow, based on the fact that it is sunny today).

In the examples discussed thus far, the states of the model are directly observable (ie: sunny, cloudy, rainy). However, what if the states of the model are not directly observable? This is where Hidden Markov Models find their application. A HMM is a doubly stochastic process in which the underlying states are hidden, and probabilistically related to a sequence of observed symbols[1]. An example scenario where a HMM could be applied is described below.

Imagine you are a hermit that lives in a cave by the ocean. You never go outside, yet you still want to know what the weather is like. The only clue you have about the weather is a piece of seaweed that you can see from the entrance of your cave. The seaweed can be dry, dryish, damp, or soggy, depending on the weather. This scenario could be adequately modeled using an HMM. The weather is a set of hidden states, and the condition of the seaweed is an observable symbol (that is probabilistically related to the hidden state).

Hidden Markov Models are used to solve three canonical problems:

1. Given a sequence of observations, determine the probability that the set of observations came form a given HMM,
2. Given a sequence of observations, determine the most likely sequence of hidden states that led to the observations, and
3. Determine the optimal set of model parameters given a set of observations. This last problem is the most difficult - having no simple analytical expression. Different training schemes exist, most of which are based on iterative optimization or gradient decent techniques.

Solving the first problem is relatively straight forward using an exhaustive search. To find the probability that a sequence of observations came from a particular HMM, we simply need to add up the probabilities from all of the possible hidden state sequences multiplied by the probability of observing the appropriate symbol in each hidden state. It turns out that this method is extremely computationally complex, requiring on the order of $2T * N^T$ operations, where T is the number of observations, and N is the number of hidden states in the model[2]. Consider that, if N=5 and T=100, the number of operations required would be on the order of 10^{72}! If we are to make use of HMM a more efficient algorithm is required. Fortunately such an algorithm exists (the forward algorithm).

The Forward algorithm makes use of recursion to reduce the computational load when solving problem 1 (described above). The forward algorithm works by traversing a trellis structure, and calculating the partial probabilities at each node. The sum of partial probabilities for the final observation is the solution to the problem. This algorithm requires on the order of N^2T calculations. Using the same example as before (N=5, T=100), 3000 computations are required, which is dramatically less that 10^{72} (69 orders of magnitude less)!

The solution to problem 2 is not as straight forward as the solution to problem 1. In fact, there are many possible solutions to problem 2, depending on what the optimality criterion is. One popular algorithm for solving problem 2 is the Viterbi algorithm. This is also a recursive algorithm (and thus is computationally inexpensive). The Viterbi algorithm finds the single best sequence of hidden state transitions in the HMM given a set of observations[3].

Hidden Markov Models have proven to be useful in many temporal

pattern recognition problems. In music technology HMMs have found a number of applications from gesture recognition, to score following, to timber recognition, and partial tracking.

Bibliography

[1] L.R. Rabiner, and B.H. Juang. 1986. An introduction to hidden markov models. IEEE ASSP Magazine. (3)1:4-15.

[2] L. R. Rabiner. 1989. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. Proceedings of the IEEE 77(2): 257-85.

[3] University of Leeds. 2007. Hidden Markov Model Introduction. [cited. Available from http://www.comp.leeds.ac.uk/roger/HiddenMarkovModels/html_dev/main.html]