An Introduction to Hidden Markov Models

Corey Kereliuk

February 1, 2007

Outline

Outline

Introduction

- In the physical sciences there are many real-world phenomena that we would like to model in order to explain/characterize our observations
- Often, there will be a sequence of observable symbols

For example:

Traffic lights

Red→Green→Amber (Deterministic)

Weather

Sunny→Cloudy→Rainy (Stochastic)

Outline

First, What is A Markov Model?

- A Markov model is one in which the current state of the model is conditionally dependent on the previous N states of the model.
- So a 1^{st} order Markov model is one in which the next state is conditionally dependent on the current state.
- For example, the probability that it will be sunny tomorrow is 0.7 given that it is sunny today.
- The Markov assumption may not be 100% true, however, it simplifies modeling

A Hidden Markov Model

- In the above examples, the states that we wish to model are directly observable. What if, however, we wish to model states that are not *directly* observable?
- A Hidden Markov model is a model in which the underlying states are hidden. We only have access to an observable set of symbols that are probabilistically related to the hidden states.

An Example HMM

- Imagine you are a hermit living in a cave next to the ocean
- You would like to know the weather outside, but since you are a hermit, you never actually go outside (the weather is a hidden state)
- You can, however, observe some seaweed lying at the entrance of your cave. The seaweed will be alternately dry, dryish, damp, or soggy, depending on the weather (the state of the seaweed is an observable symbol)
- You, the hermit, might want to use a Hidden Markov Model to predict the weather outside, or in order to determine the season, for instance

Continuing With The Weather Example

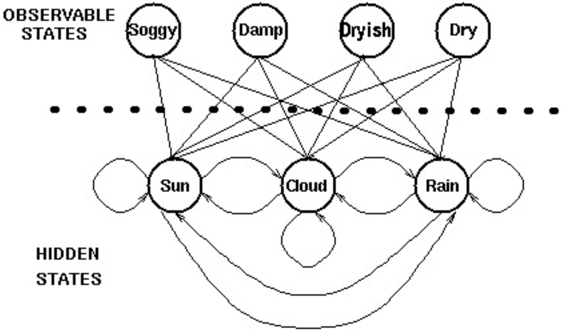

Figure: Weather Example [3]

Probability Matrices

weather today

weather yesterday	Sun	Cloud	Rain
Sun	0.5	0.25	0.25
Cloud	0.375	0.125	0.375
Rain	0.125	0.625	0.375

Figure: Transition Matrix

Seaweed

weather	Dry	Dryish	Damp	Soggy
Sun	0.60	0.20	0.15	0.05
Cloud	0.25	0.25	0.25	0.25
Rain	0.05	0.10	0.35	0.50

Figure: Confusion Matrix

The Parameters of a Hidden Markov Model

T = length of observation sequence
N = number of states in the model
M = number of observable symbols
$Q = \{q_1, q_2, ..., q_{N-1}, q_N\}$ sequence of hidden states
$V = \{v_1, v_2, ..., v_{N-1}, v_N\}$ set of observable symbols
$A = \{a_{ij}\}$ state transition probabilities
$B = \{b_j(k)\}$ observation probability distribution
$\pi = \{\pi_i\}$ initial state distribution
$\lambda = (A, B, \pi)$ set of probability distributions that describe the HMM

Outline

The Three Canonical Problems We Solve Using HMMs

- Problem 1: Given the observation sequence $O = O_1, O_2, ...O_T$ and the Hidden Markov Model, λ, how can we compute $Pr(O|\lambda)$. In other words, what is the probability that the observed sequence came from the model.
- Problem 2: Given the observation sequence $O = O_1, O_2, ...O_T$ determine the most likely sequence of hidden model states (there are several possible solutions—depending on the optimality criterion selected)
- Problem 3: How can we adjust the HMM, λ, in order to maximize $Pr(O|\lambda)$. In other words, how do we train the model given a set of observations

The Exhaustive Search

Consider problem 1. It is possible to solve this problem using brute force. First we need to calculate the probability of observing the sequence of symbols, O, given a sequence of hidden states, $Q = \{q_1, q_2, ..., q_T\}$:

$$Pr(O|Q,\lambda) = b_{q_1}(O_1) * b_{q_2}(O_2) * ... * b_{q_T}(O_T)$$

We also note the probability that Q will occur in the model is:

$$Pr(Q|\lambda) = \pi_1 * a_{12} * a_{23} * ... * a_{T-1T}$$

So, the joint probability of O and Q occurring simultaneously is:

$$Pr(O,Q|\lambda) = Pr(O|Q,\lambda) * Pr(Q|\lambda)$$

Since there are N hidden states, and T observations, the probability of O occurring in λ is:

$$Pr(O,\lambda) = \sum_{allQ} Pr(O,Q|\lambda)$$

The brute force method requires on the order of $2T * N^T$ operations. That's 10^{72} computations for $N = 5$, $T = 100$

The Forward Algorithm

Obviously, the exhaustive search is too computationally intensive to be practical. Fortunately, there are other, more efficient algorithms for solving the canonical HMM problems. The first algorithm is the *forward* algorithm, which makes use of recursion to lower the computational complexity. The forward variable is defined as:

$$\alpha_t(i) = Pr(O_1, O_2, ..., O_t, i_t = q_t|\lambda)$$

We solve for $\alpha_t(i)$ as follows:

1. $\alpha_1(i) = \pi_i * b_i(O_1), 1 \leq i \leq N$
2. For $t = 1, 2, ..., T-1$,

$$\alpha_{t+1}(j) = b_j(O_{t+1}) \sum_{i=1}^{N} \alpha_t(i) * a_{ij}, 1 \leq j \leq N$$

3. Then, $Pr(O|\lambda) = \sum_{i=1}^{N} \alpha_T(i)$

The Viterbi Algorithm

The *Viterbi* algorithm is used to solve canonical problem 2 (it finds the single best sequence of hidden states leading to the observations O).
Step 1 — Initialization

$$\delta_1(i) = \pi_i * b_i(O_1), 1 \leq i \leq N$$
$$\Psi_1(i) = 0$$

Step 2 — Recursion

$$\text{For } 2 \leq t \leq T, 1 \leq j \leq N$$
$$\delta_t(j) = max_{1 \leq i \leq N}[\delta_{t-1}(i) * a_{ij}] * b_j(O_t)$$
$$\Psi_t(j) = argmax_{1 \leq i \leq N}[\delta_{t-1}(i) * a_{ij}]$$

Step 3 — Termination

$$P^* = max_{1 \leq i \leq N}[\delta_T(i)]$$
$$i_T^* = argmax_{1 \leq i \leq N}[\delta_T(i)]$$

Step 4 — Path Backtracking

$$i_t^* = \Psi_{t+1}(i_{t+1}^*), t = T-1, T-2, ..., 1$$

Model Training

- Given a set of observations, we would like to tune the parameters of the HMM to maximize the probability that the HMM will yield the observations set.
- This is the most difficult problem, with no analytical solution
- Iterative procedures, or gradient decent methods are often used to optimize an HMM

Outline

Real World Applications of HMMs

- Speech recognition systems
- Modeling of DNA protein and genome sequences
- Score following
- Gesture recognition
- Partial Tracking

Outline

Conclusion

- HMMs allow us to model processes with a hidden state, based on observable parameters
- HMMs are a valuable tool in temporal pattern recognition
- The main problems solved with HMMs are i) determining how likely it is that a set of observations came from a particular model, and ii) determining the most likely sequence of hidden states
- The most difficult problem with HMMs is to determine appropriate model parameters