Hidden Markov Models


Markov Chain

Set of states, e.g. (S1,S2,S3,S4,S5)
Table of transition probabilities (a_{ij} = Prob of going from state S_i to S_j)
\[ a_{ij} = \begin{cases} \pi_{i,j} \quad & \text{Prob of starting in state } S_i \text{ and going to state } S_j \\ \prod_{k=1}^{S_j} a_{ik} \quad & \text{Prob of starting in state } S_i \\ \prod_{k=1}^{S_j} a_{ik} \quad & \text{Prob of starting in state } S_i \end{cases} \]

Hidden Markov Model

Set of states, e.g. (S1,S2,S3,S4,S5)
Table of transition probabilities (a_{ij} = Prob of going from state S_i to S_j)
\[ a_{ij} = \begin{cases} \pi_{i,j} \quad & \text{Prob of starting in state } S_i \\ \prod_{k=1}^{S_j} a_{ik} \quad & \text{Prob of starting in state } S_i \end{cases} \]

HMM as Graphical Model

Given HMM with states S1, S2, ..., Sn and symbols v1, v2, ..., vk, transition 
probs A={a_{ij}}, initial probs \( \pi_i \), and observation probs B={b_i}
Let state random variables be \( X_1, X_2, X_3, X_4, ... \) with \( X_t \) being state at time \( t \).
Allowable values of \( X_t \) (a discrete random variable) are \{S1, S2, ..., Sn\}
Let observed random variables be \( O_1, O_2, O_3, ... \) with \( O_t \) observed at time \( t \).
Allowable values of \( O_t \) (a discrete random variable) are \{v1, v2, ..., vk\}
HMM as Graphical Model

Given HMM with states $S_1, S_2, ..., S_n$, symbols $v_1, v_2, ..., v_k$, transition probs $A = \{a_{ij}\}$, initial probs $\Pi = \{\pi_i\}$, and observation probs $B = \{b_{ik}\}$

Let state random variables be $X_1, X_2, X_3, X_4, ...$ with $X_t$ being state at time $t$. Allowable values of $X_t$ (a discrete random variable) are $\{S_1, S_2, ..., S_n\}$

Let observed random variables be $O_1, O_2, O_3, ...$ with $O_t$ observed at time $t$. Allowable values of $O_t$ (a discrete random variable) are $\{v_1, v_2, ..., v_k\}$

$P(X_1, X_2, X_3, X_4, O_1, O_2, O_3, O_4) = P(X_1) P(O_1|X_1) P(X_2|X_1) P(O_2|X_2) P(X_3|X_2) P(O_3|X_3) P(X_4|X_3) P(O_4|X_4)$ ...

Oi are observed variables. Xj are hidden (latent) variables.

Three Computations for HMMs

Problem 1: Given the observation sequence $O = O_1, O_2, O_3, ...$ and a model $\lambda = \{A, B, \pi\}$, how do we efficiently compute $P(O|\lambda)$, the probability of the observation sequence, given the model?

Problem 2: Given the observation sequence $O = O_1, O_2, O_3, ...$ and the model $\lambda$, how do we choose a corresponding state sequence $Q = q_1, q_2, ...$ which is optimal in some meaningful sense (i.e., best “explains” the observations)?

Problem 3: How do we adjust the model parameters $\lambda = \{A, B, \pi\}$ to maximize $P(O|\lambda)$?

HMM Problem 1

What is the likelihood of observing a sequence, e.g., $O_1, O_2, O_3$?

Note: there are multiple ways that a given sequence could be emitted, involving different sequences of hidden states $X_1, X_2, X_3$.

One (inefficient) way to compute the answer: generate all sequences of three states $S_x, S_y, S_z$ and compute $P(S_x, S_y, S_z, O_1, O_2, O_3)$ [which we know how to do]. Summing up over all sequences of three states gives us our answer.

Drawback, there are $3^N$ subsequences that can be formed from $N$ states.

HMM Problem 2

What is the most likely sequence of hidden states $S_1, S_2, S_3$ given that we have observed $O_1, O_2, O_3$?

$X_{MAP} = \arg\max_X P(x_1, x_2, x_3 | O_1, O_2, O_3)$

$= \arg\max_X P(x_1, x_2, x_3, O_1, O_2, O_3)$

Again, there is an inefficient way based on explicitly generating the exponential number of possible state sequences... AND, there is a more efficient way using message passing.
HMM Problem 2

\[ X_{MAP} = \arg \max_X P(x_1, x_2, x_3; O_1, O_2, O_3) \]

\[ = \arg \max_X P(x_1) P(O_1 | x_1) P(x_2 | x_1) P(O_2 | x_2) P(O_3 | x_3) \]

Define:

\[ \Phi(X) = \max_{x_1} P(x_1) P(O_1 | x_1) \max_{x_2} P(x_2 | x_1) P(O_2 | x_2) \max_{x_3} P(x_3 | x_2) P(O_3 | x_3) \]

Then

\[ X_{MAP} = \arg \max_X \Phi(X) \]

This is the max-product procedure of message passing. It is called the Viterbi algorithm in HMM terminology.

Viterbi, under the hood

1) Build a state-space trellis.
2) Add a source and sink node.
3) Given observations.
4) Assign weights to edges based on observations.
5) Do multistage dynamic programming to find the max product path.
   (note: in some implementations, each edge is weighted by \(-\log(w_{ij})\) of the weights shown here, so that standard min length path DP can be performed.)

HMM Problem 3

Given a training set of observation sequences

\{(O_1, O_2, ...)\}

how do we determine the transition probs \(a_{ij}\), initial state probs \(p_i\), and observation probs \(b_{ik}\)?

This is a learning problem. It is the hardest problem of the three. We assume topology of the HMM is known (number and connectivity of the states) otherwise it is even harder.

Two popular approaches:

- Segmental K-means algorithm
- Baum-Welch algorithm (EM-based)

Problem 3

- Given some training data, build a HMM
- The training data is a set of observation sequences
- We assume that these sequences are representative and independent
- We want the HMM to be the one most likely to give rise to the training data
- We'll look at a solution based on k-means, which uses the solution to Problem 2 to define the best HMM

Segmental K-Means

Algorithm Overview

1. Initialise the HMM states and assign observation symbols to these states
2. Compute the initial state and transition functions given the current HMM
3. Compute some statistics about each state
4. Find the observation function for each state
5. Find the optimal path through the HMM for each observation sequence, and reassign its observation symbols as appropriate
6. If any changes have been made goto 2
Step 1
- We need to set up some initial states
  - We know there are $n$ of them
  - Choose $n$ (different) observation symbols (vectors) and assign these to the $n$ states at random
  - Assign all the other observed vectors to the state which are closest to (using Euclidean distance)

Note: this is essentially a clustering problem!

Example
- Suppose our observation symbols are velocity measurements, $[u, v]$
  - We're given three observation sequences:
    - $[0.1, -0.1, 0.0, 0.0, 2.1, 4.5, 5.4, \{-3, 1\}]$
    - $[0.1, -0.1, 0.0, 0.0, 2.1, 4.5, 5.4, \{-3, 1\}]$
    - $[0.1, -0.1, 0.0, 0.0, 2.1, 4.5, 5.4, \{-3, 1\}]$
  - We pick $k=3$ at random:
    - $[0.1, -0.1, 0.0, 0.0, 2.1, 4.5, 5.4, \{-3, 1\}]$
    - $[0.1, -0.1, 0.0, 0.0, 2.1, 4.5, 5.4, \{-3, 1\}]$
    - $[0.1, -0.1, 0.0, 0.0, 2.1, 4.5, 5.4, \{-3, 1\}]$

Example
- Each of these three vectors becomes a state, and we assign each symbol to the state nearest it:

Example
- Doing this for all 3 sequences gives:
  - In the state, $s_1$ centred around $[-3, 1]$:
    - $[0.1, -0.1, 0.0, 0.0, 2.1, 4.5, 5.4, \{-3, 1\}]$
    - $[0.1, -0.1, 0.0, 0.0, 2.1, 4.5, 5.4, \{-3, 1\}]$
    - $[0.1, -0.1, 0.0, 0.0, 2.1, 4.5, 5.4, \{-3, 1\}]$
  - In the state, $s_2$ centred around $[2, 1]$:
    - $[0.1, -0.1, 0.0, 0.0, 2.1, 4.5, 5.4, \{-3, 1\}]$
    - $[0.1, -0.1, 0.0, 0.0, 2.1, 4.5, 5.4, \{-3, 1\}]$
    - $[0.1, -0.1, 0.0, 0.0, 2.1, 4.5, 5.4, \{-3, 1\}]$
  - In the state, $s_3$ centred around $[4, 4]$:
    - $[0.1, -0.1, 0.0, 0.0, 2.1, 4.5, 5.4, \{-3, 1\}]$
    - $[0.1, -0.1, 0.0, 0.0, 2.1, 4.5, 5.4, \{-3, 1\}]$
    - $[0.1, -0.1, 0.0, 0.0, 2.1, 4.5, 5.4, \{-3, 1\}]$

Step 2
- We now compute the initial probabilities and transition functions:
  - $I(s)$ is just the proportion of times a sequence starts in state $s$
  - $T(s, s')$ is the proportion of times a sequence takes us from state $s$ to state $s'$

Example
- Initial state function:
  - The first sequence starts with $[0, 1]$, which is in state $s_1$
  - The second sequence starts with $[-3, 0]$, which is in state $s_1$
  - The third sequence starts with $[-3, -1]$, which is in state $s_1$
  - So $I(s_1)=2/3, I(s_2)=1/3, I(s_3)=0$

Example
- We're given three observation sequences:
  - $[0.1, -0.1, 0.0, 0.0, 2.1, 4.5, 5.4, \{-3, 1\}]$
  - $[0.1, -0.1, 0.0, 0.0, 2.1, 4.5, 5.4, \{-3, 1\}]$
  - $[0.1, -0.1, 0.0, 0.0, 2.1, 4.5, 5.4, \{-3, 1\}]$
Note: This part assumes continuous-valued observations are output. For a discrete set of observation symbols, we could just do histograms here.

Step 3

- We're given three observation sequences:
  - \( [0, 1], [-1, 0], [0, 0], [2, 1], [4, 5], [5, 4] \)
  - \( [-3, 0], [-4, 1], [-3, -1], [-2, -1], [0, 0], [-1, 1] \)
  - \( [-3, -1], [-1, 0], [0, 1], [3, 3], [4, 4], [4, 3] \)

Step 4

- The next step is to compute the observation probabilities:
  - Since the observations are vectors, this is usually a probability function.
  - Commonly a Gaussian model is used, but other models could be used if we want.
  - Since the vectors are \( d \)-dimensional, this Gaussian is \( d \)-dimensional also.

\[ O(x, s) = \frac{1}{2\pi \sigma^d e^{\frac{(x - \mu_s)^T (x - \mu_s)}}} \]
Step 5

- We now have estimates for the parameters of our HMM
- As in normal k-means the next step is to reassign the observations to the states
- We do this by running the Viterbi algorithm for each observation, giving an optimal assignment of observation symbols to states
- If any observation has changed state then we go back to step 2 to revise our HMM

Example

- Suppose the Viterbi algorithm gives the sequence $s_3 \rightarrow s_2 \rightarrow s_5 \rightarrow s_3 \rightarrow s_2$ as the best path for the first observation
- The second symbol (1.0) is now attributed to state $s_5$ rather than $s_3$
- This affects the statistics related with these two states, and also the transition function
- As a result, the paths explaining the observations may also change

Training HMMs

- This algorithm, the segmental k-means algorithm, allows us to build a HMM from training data
- It can be shown that this converges to an optimal result for a variety of observation functions (including Gaussian)
- In some cases the starting point doesn’t matter, although a poor choice might mean lots of iterations are needed

Baum-Welch Algorithm

Same thing, but with dancing bears—fractional assignments of observations to states

Basic idea is to use EM instead of K-means as a way of assessing ownership weights before computing observation statistics at each state.