Hidden Markov Model

Prediction of Traffic based on sequential speed data of EV

Markov Process

- A markov chain is a model that tells us something about the probabilities of sequences of random variables.

- A markov chain makes a very strong assumption that if we want to predict the future in the sequence, all that matters is the current state.

- The states before the current state have no impact on the future except via the current state

Markov Model

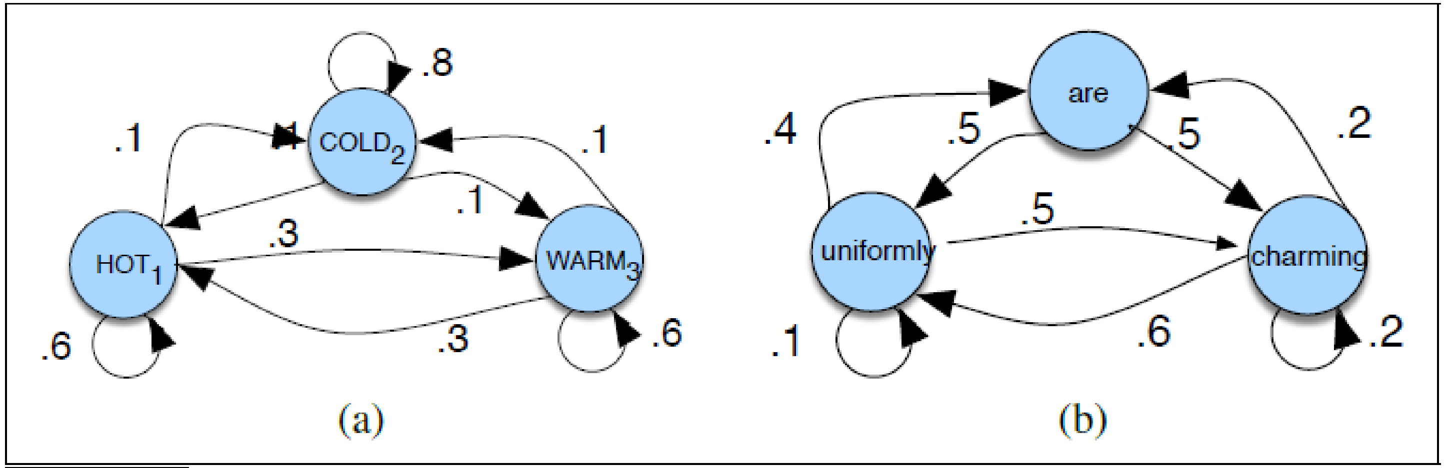


Figure A.1 A Markov chain for weather (a) and one for words (b), showing states and transitions. A start distribution π is required; setting $\pi = [0.1, 0.7, 0.2]$ for (a) would mean a probability 0.7 of starting in state 2 (cold), probability 0.1 of starting in state 1 (hot), etc.

Markov Assumption: $P(q_i = a | q_1 \ldots q_{i-1}) = P(q_i = a | q_{i-1})$

Terminologies

$Q = q_1 q_2 \ldots q_N$	a set of N **states**
$A = a_{11} a_{12} \ldots a_{n1} \ldots a_{nn}$	a **transition probability matrix** A, each a_{ij} representing the probability of moving from state i to state j, s.t. $\sum_{j=1}^{n} a_{ij} = 1 \quad \forall i$
$\pi = \pi_1, \pi_2, \ldots, \pi_N$	an **initial probability distribution** over states. π_i is the probability that the Markov chain will start in state i. Some states j may have $\pi_j = 0$, meaning that they cannot be initial states. Also, $\sum_{i=1}^{n} \pi_i = 1$

Hidden Markov Model

- A Markov chain is useful when we need to compute a probability for a sequence of observable events.
- In many cases, however, the events we are interested in are hidden (we don't observe them directly).
- HMM allows us to compute probabilities of both hidden and visible states
- Ex : Prediction of weather based upon number of ice-cream
- **In Our case, Prediction of Traffic based on sequential speed data.**

Terminologies

$Q = q_1 q_2 \ldots q_N$	a set of N **states**
$A = a_{11} \ldots a_{ij} \ldots a_{NN}$	a **transition probability matrix** A, each a_{ij} representing the probability of moving from state i to state j, s.t. $\sum_{j=1}^{N} a_{ij} = 1 \quad \forall i$
$O = o_1 o_2 \ldots o_T$	a sequence of T **observations**, each one drawn from a vocabulary $V = v_1, v_2, \ldots, v_V$
$B = b_i(o_t)$	a sequence of **observation likelihoods**, also called **emission probabilities**, each expressing the probability of an observation o_t being generated from a state i
$\pi = \pi_1, \pi_2, \ldots, \pi_N$	an **initial probability distribution** over states. π_i is the probability that the Markov chain will start in state i. Some states j may have $\pi_j = 0$, meaning that they cannot be initial states. Also, $\sum_{i=1}^{n} \pi_i = 1$

Assumptions in HMM

- The probability of a particular state depends only on the previous state(First Order Markov Process)

$$\textbf{Markov Assumption:} \quad P(q_i|q_1 \ldots q_{i-1}) = P(q_i|q_{i-1})$$

- The probability of an output observation Oi depends only on the state that produced the observation qi and not on any other states or any other observations.

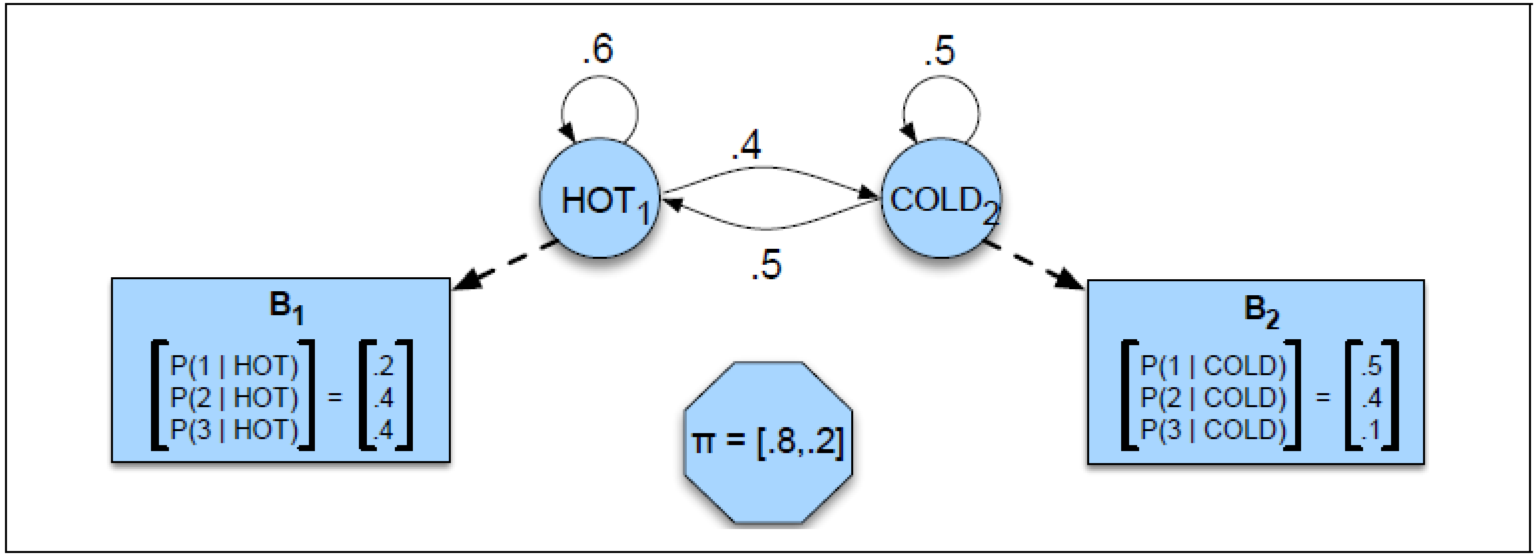
$$\textbf{Output Independence:} \quad P(o_i|q_1 \ldots q_i, \ldots, q_T, o_1, \ldots, o_i, \ldots, o_T) = P(o_i|q_i)$$

How we can apply HMM to our data?

- Given a sequence of observations O (each an integer representing the number of ice creams eaten on a given day) find the '**hidden**' sequence Q of weather states (H or C) which caused Jason to eat the ice cream.

- **Note: Both hidden and visible state must discrete**.
  - Discretizing Speed into **certain number of buckets** and **predicting traffic** i.e High, Low, Medium

- Given a sequence of observation O (each an integer representing the speed bucket) find the '**hidden**' sequence Q of **Traffic** states (H or L or M) which caused change in speed.



Learned Parameters of HMM

Problem 1 (Likelihood):	Given an HMM $\lambda = (A,B)$ and an observation sequence O, determine the likelihood $P(O\|\lambda)$.
Problem 2 (Decoding):	Given an observation sequence O and an HMM $\lambda = (A,B)$, discover the best hidden state sequence Q.
Problem 3 (Learning):	Given an observation sequence O and the set of states in the HMM, learn the HMM parameters A and B.

So, Problem 2 is what we need !

Problem 1： forward algorithm

- Computing Likelihood: Given an HMM Model = (A,B) and an observation sequence O, determine the likelihood P(O)

$$P(O|Q) = \prod_{i=1}^{T} P(o_i|q_i)$$

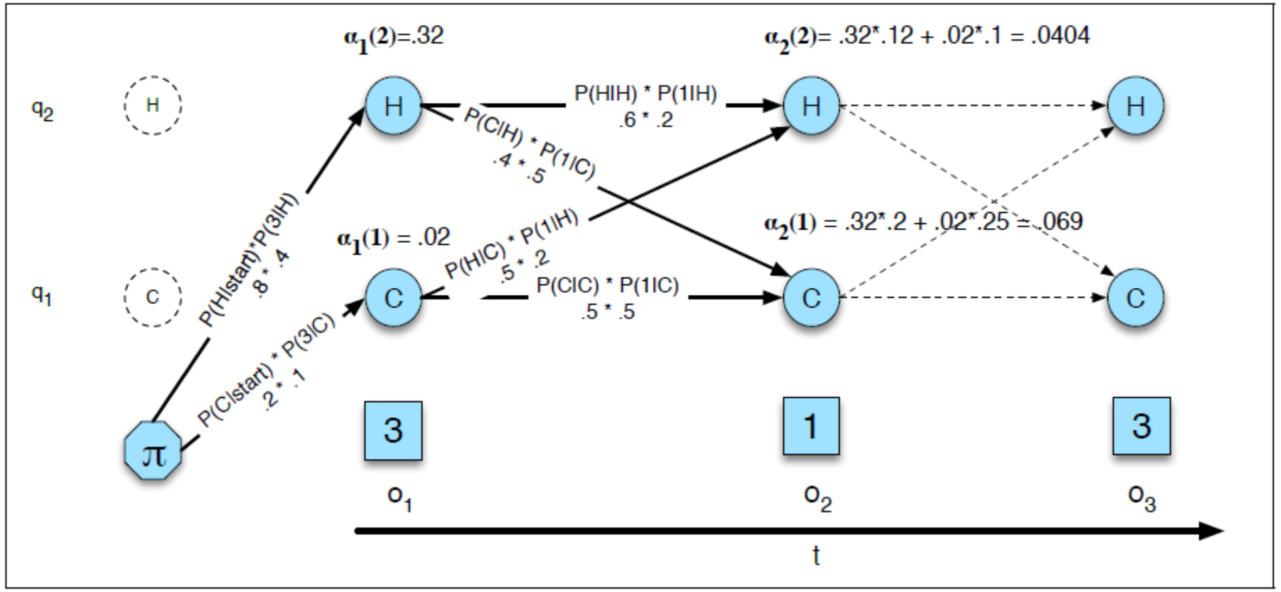
$$P(O,Q) = P(O|Q) \times P(Q) = \prod_{i=1}^{T} P(o_i|q_i) \times \prod_{i=1}^{T} P(q_i|q_{i-1})$$

$$\begin{aligned} P(3\ 1\ 3, \text{hot hot cold}) &= P(\text{hot}|\text{start}) \times P(\text{hot}|\text{hot}) \times P(\text{cold}|\text{hot}) \\ &\quad \times P(3|\text{hot}) \times P(1|\text{hot}) \times P(3|\text{cold}) \end{aligned}$$

$$P(O) = \sum_{Q} P(O,Q) = \sum_{Q} P(O|Q)P(Q)$$

$$P(3\ 1\ 3) = P(3\ 1\ 3, \text{cold cold cold}) + P(3\ 1\ 3, \text{cold cold hot}) + P(3\ 1\ 3, \text{hot hot cold}) + \ldots$$

Dynamic Approach



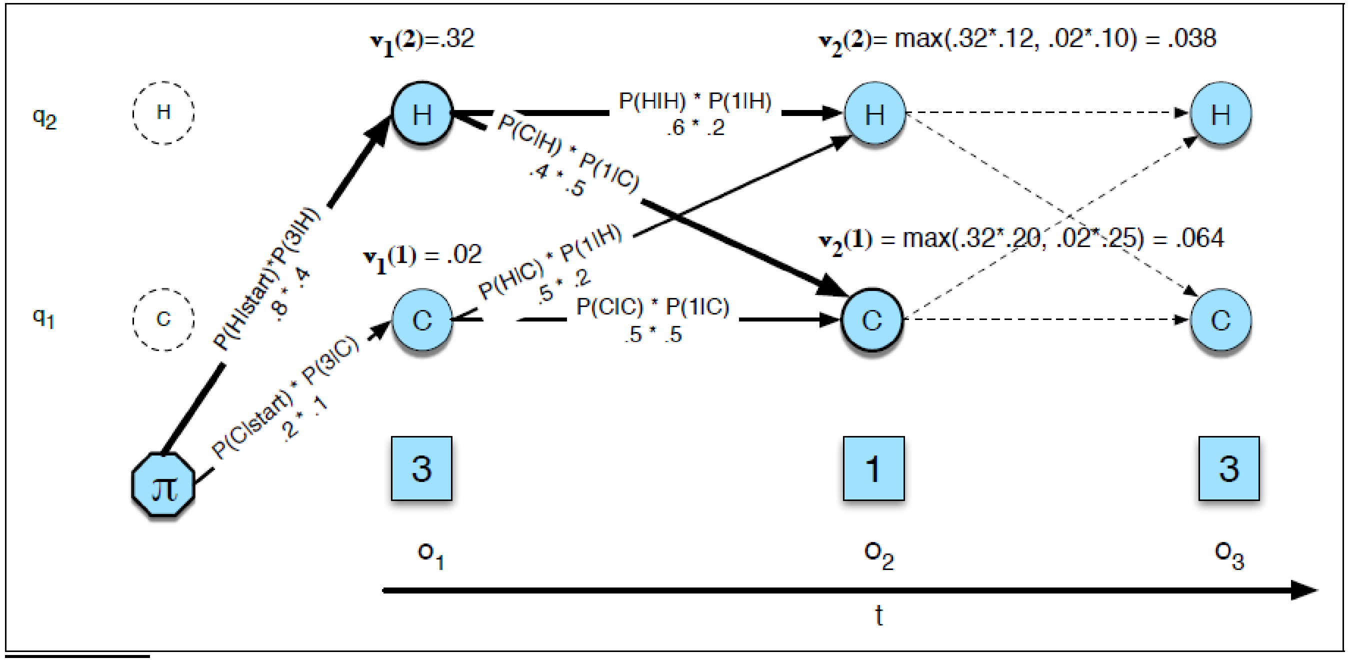
$$P(O|\lambda) = \sum_{i=1}^{N} \alpha_T(i)$$

Problem 2 : Decoding

- Given the visible sequence (speed buckets), the task is to determine hidden stated corresponding to each visible state

- (INPUT) SPEED : HIGH LOW MID HIGH LOW MID
- (OUTPUT) TRAFFIC : LOW HIGH HIGH LOW HIGH HIGH

Viterbi Algorithm



Problem 3 : Forward Backward Algorithm

- Goal : To learn matrices A (Transition Matrix) and B (Emission matrix)
- Input : The input to such a learning algorithm would be an unlabeled sequence of observations O and a vocabulary of potential hidden states Q.
- The standard algorithm for HMM training is forward backward algorithm, it’s an iterative algorithm.
- The algorithm will let us train both the transition probabilities A and the emission probabilities B of the HMM.
- We will start with an estimate for the transition and observation probabilities and then use these estimated probabilities to derive better and better probabilities.

Backward Algorithm

$$\beta_t(i) = P(o_{t+1}, o_{t+2} \ldots o_T | q_t = i, \lambda) \qquad \text{(A.15)}$$

It is computed inductively in a similar manner to the forward algorithm.

1. **Initialization:**

$$\beta_T(i) \;=\; 1, \quad 1 \leq i \leq N$$

2. **Recursion**

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij}\, b_j(o_{t+1})\, \beta_{t+1}(j), \quad 1 \leq i \leq N, 1 \leq t < T$$

3. **Termination:**

$$P(O|\lambda) = \sum_{j=1}^{N} \pi_j\, b_j(o_1)\, \beta_1(j)$$

$$\hat{a}_{ij} = \frac{\text{expected number of transitions from state } i \text{ to state } j}{\text{expected number of transitions from state } i}$$

$$\xi_t(i,j) = P(q_t = i, q_{t+1} = j | O, \lambda)$$

$$P(O|\lambda) = \sum_{j=1}^{N} \alpha_t(j)\beta_t(j)$$

$$P(X|Y,Z) = \frac{P(X,Y|Z)}{P(Y|Z)}$$

$$\xi_t(i,j) = \frac{\alpha_t(i)\, a_{ij} b_j(o_{t+1})\beta_{t+1}(j)}{\sum_{j=1}^{N} \alpha_t(j)\beta_t(j)}$$

$$\text{not-quite-}\xi_t(i,j) = P(q_t = i, q_{t+1} = j, O|\lambda)$$

$$\text{not-quite-}\xi_t(i,j) = \alpha_t(i)\, a_{ij} b_j(o_{t+1})\beta_{t+1}(j)$$

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \sum_{k=1}^{N} \xi_t(i,k)}$$

$$\hat{b}_j(v_k) = \frac{\text{expected number of times in state } j \text{ and observing symbol } v}{\text{expected number of times in state } j}$$

$$\gamma_t(j) = P(q_t = j|O,\lambda)$$

$$\gamma_t(j) = \frac{P(q_t = j, O|\lambda)}{P(O|\lambda)}$$

$$\hat{b}_j(v_k) = \frac{\sum_{t=1\, s.t. O_t = v_k}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)}$$

Initialize the transition and emission matrix and go on correcting them till they get converged

Thank you !