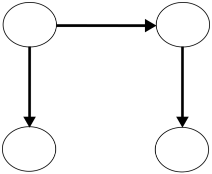
Probabilistic Graphical Models: Principles and Applications

Chapter 5: HIDDEN MARKOV MODELS

L. Enrique Sucar, INAOE

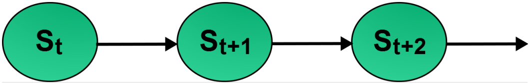


Outline

Introduction

- Markov Chains are another class of PGMs that represent dynamic processes
- For instance, consider that we are modeling how the weather in a particular place changes over time
- A simple weather model as a Markov chain in which there is a state variable per day, with 3 possible values: *sunny, cloudy, raining*; these variables are linked in a *chain*

Markov Chain



This implies what is known as the *Markov property*, the state of the weather for the next day, S_{t+1}, is independent of all previous days given the present weather, S_t, i.e., $P(S_{t+1} \mid S_t, S_{t-1}, ...) = P(S_{t+1} \mid S_t)$

Hidden Markov Models

- The previous model assumes that we can measure the weather with precision each day, that is, the state is *observable*
- In many applications we cannot observe the state of the process directly, so we have what is called a *Hidden Markov Model*, where the state is hidden
- In addition to the probability of the next state given the current state, there is another parameter which models the uncertainty about the state, represented as the probability of the *observation* given the state, $P(O_t \mid S_t)$

Definition

- A Markov chain (MC) is a *state machine* that has a discrete number of states, $q_1, q_2, ..., q_n$, and the transitions between states are non-deterministic
- Formally, a Markov chain is defined by:
  Set of states: $Q = \{q_1, q_2, ..., q_n\}$
  Vector of prior probabilities: $\Pi = \{\pi_1, \pi_2, ..., \pi_n\}$, where
  $$\pi_i = P(S_0 = q_i)$$
  Matrix of transition probabilities: $A = \{a_{ij}\}$,
  $i = [1..n], j = [1..n]$, where
  $$a_{ij} = P(S_t = q_j \mid S_{t-1} = q_i)$$
- In a compact way, a MC is represented as $\lambda = \{A, \Pi\}$

Properties

1. Probability axioms: $\sum_i \pi_i = 1$ and $\sum_j a_{ij} = 1$
2. Markov property: $P(S_t = q_j \mid S_{t-1} = q_i, S_{t-2} = q_k, \ldots) = P(S_t = q_j \mid S_{t-1} = q_i)$

Example - simple weather model

sunny (q1)	cloudy (q2)	raining (q3)
0.2	0.5	0.3

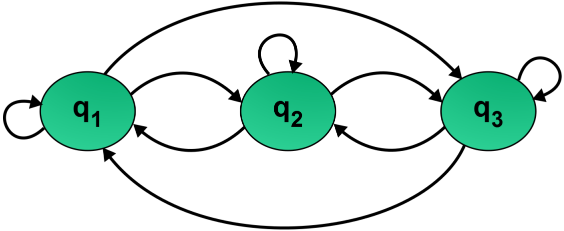
Table: Prior probabilities.

	sunny	cloudy	raining
sunny	0.8	0.1	0.1
cloudy	0.2	0.6	0.2
raining	0.3	0.3	0.4

Table: Transition probabilities.

State Transition Diagram

- This diagram is a directed graph, where each node is a state and the arcs represent possible transitions between states



Basic Questions

Given a Markov chain model, there are three basic questions that we can ask:

- What is the probability of a certain state sequence?
- What is the probability that the chain remains in a certain state for a period of time?
- What is the expected time that the chain will remain in a certain state?

Probability of a state sequence

- The probability of a sequence of states given the model is basically the product of the transition probabilities of the sequence of states:

$$P(q_i, q_j, q_k, ...) = a_{0i} a_{ij} a_{jk} .... \quad (1)$$

- For example, in the weather model, we might want to know the probability of the following sequence of states: $Q =$ *sunny, sunny, rainy, rainy, sunny, cloudy, sunny*.

Probability of remaining in a state

- The probability of staying d time steps in a certain state, q_i, is equivalent to the probability of a sequence in this state for $d - 1$ time steps and then transiting to a different state.

$$P(d_i) = a_{ii}^{d-1}(1 - a_{ii}) \tag{2}$$

- Considering the weather model, what is the probability of 3 cloudy days?

Average duration

- The average duration of a state sequence in a certain state is the expected value of the number of stages in that state, that is: $E(D) = \sum_i d_i P(d_i)$

$$E(d_i) = \sum_i d_i a_{ii}^{d-1}(1 - a_{ii}) \tag{3}$$

- Which can be written in a compact form as:

$$E(d_i) = 1/(1 - a_{ii}) \tag{4}$$

- What is the expected number of days that the weather will remain cloudy?

Parameter Estimation

- The parameters can be estimated simply by counting the number of times that the sequence is in a certain state, i; and the number of times there is a transition from a state i to a state j:
  Initial probabilities:

$$\pi_i = \gamma_{0i}/N \tag{5}$$

  Transition probabilities:

$$a_{ij} = \gamma_{ij}/\gamma_i \tag{6}$$

- γ_{0i} is the number of times that the state i is the initial state in a sequence, γ_i is the number of times that we observe state i, and γ_{ij} is the number of times that we observe a transition from state i to state j

Weather Example - data

- Consider that for the weather example we have the following 4 observation sequences:

  $q_2, q_2, q_3, q_3, q_3, q_3, q_1$
  $q_1, q_3, q_2, q_3, q_3, q_3, q_3$
  q_3, q_3, q_2, q_2
  $q_2, q_1, q_2, q_2, q_1, q_3, q_1$

Weather Example - parameters

sunny	cloudy	raining
0.25	0.5	0.25

Table: Calculated prior probabilities for the weather example.

	sunny	cloudy	raining
sunny	0	0.33	0.67
cloudy	0.285	0.43	0.285
raining	0.18	0.18	0.64

Table: Calculated transition probabilities for the weather example.

Convergence

- If a sequence transits from one state to another a large number of times, M, what is the probability in the limit (as $M \to \infty$) of each state, q_i?
- Given an initial probability vector, Π, and transition matrix, A, the probability of each state, $P = \{p_1, p_2, ..., p_n\}$ after M *iterations* is:

$$P = \pi A^M \tag{7}$$

- The solution is given by the Perron-Frobenius theorem

Perron-Frobenius theorem

- Conditions:
  1. Irreducibility: from every state i there is a probability $a_{ij} > 0$ of transiting to any state j.
  2. Aperiodicity: the chain does not form *cycles* (a subset of states in which the chain remains once it transits to one of these state).
- Then as $M \rightarrow \infty$, the chain converges to an invariant distribution P, such that $P \times A = P$
- The rate of convergence is determined by the second *eigen-value* of matrix A

Example

- Consider a MC with three states and the following transition probability matrix:

$$A = \begin{matrix} 0.9 & 0.075 & 0.025 \\ 0.15 & 0.8 & 0.05 \\ 0.25 & 0.25 & 0.5 \end{matrix}$$

- It can be shown that in this case the steady state probabilities converge to $P = \{0.625, 0.3125, 0.0625\}$
- An interesting application of this convergence property of Markov chains is for ranking web pages

HMM

- A Hidden Markov model (HMM) is a Markov chain where the states are not directly observable.
- A HMM is that it is a double stochastic process: (i) a hidden stochastic process that we cannot directly observe, (ii) and a second stochastic process that produces the sequence of observations given the first process.
- For instance, consider that we have two unfair or "biased" coins, M_1 and M_2. M_1 has a higher probability of *heads*, while M_2 has a higher probability of *tails*. Someone sequentially flips these two coins, however we do not know which one. We can only observe the outcome, *heads* or *tails*

Example - two unfair coins

Aside from the prior and transition probabilities for the states (as with a MC), in a HMM we need to specify the *observation* probabilities

$\Pi =$

M_1	M_2
0.5	0.5

$A =$

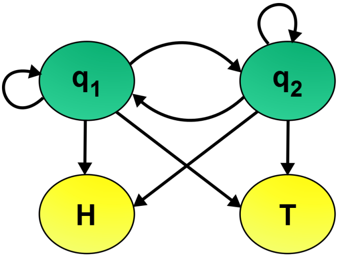
	M_1	M_2
M_1	0.5	0.5
M_2	0.5	0.5

$B =$

	M_1	M_2
H	0.8	0.2
T	0.2	0.8

Table: The prior probabilities (Π), transition probabilities (A) and observation probabilities (B) for the unfair coins example.

Coins example - state diagram



Definition

Set of states: $Q = \{q_1, q_2, ..., q_n\}$

Set of observations: $O = \{o_1, o_2, ..., o_m\}$

Vector of prior probabilities: $\Pi = \{\pi_1, \pi_2, ..., \pi_n\}$, where

$$\pi_i = P(S_0 = q_i)$$

Matrix of transition probabilities: $A = \{a_{ij}\}$,

$$i = [1..n], j = [1..n], \text{ where}$$
$$a_{ij} = P(S_t = q_j \mid S_{t-1} = q_i)$$

Matrix of observation probabilities: $B = \{b_{ij}\}$,

$$i = [1..n], j = [1..m], \text{ where}$$
$$b_{ik} = P(O_t = o_k \mid S_t = q_i)$$

Compactly, a HMM is represented as $\lambda = \{A, B, \Pi\}$

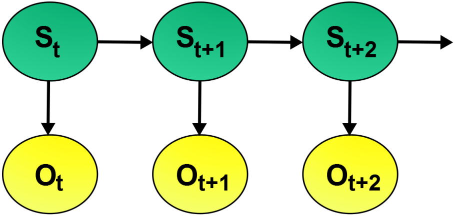
Properties

Markov property: $P(S_t = q_j \mid S_{t-1} = q_i, S_{t-2} = q_k, ...) = P(S_t = q_j \mid S_{t-1} = q_i)$

Stationary process: $P(S_{t-1} = q_j \mid S_{t-2} = q_i) = P(S_t = q_j \mid S_{t-1} = q_i)$
and
$P(O_{t-1} = o_k \mid S_{t-1} = q_j) = P(O_t = o_k \mid S_t = q_i), \forall(t)$

Independence of observations: $P(O_t = o_k \mid S_t = q_i, S_{t-1} = q_j, ...) = P(S_O = o_k \mid S_t = q_i)$

HMM: Graphical Model



Basic Questions

1. *Evaluation*: given a model, estimate the probability of a sequence of observations.
2. *Optimal Sequence*: given a model and a particular observation sequence, estimate the most probable state sequence that produced the observations.
3. *Parameter learning*: given a number of sequence of observations, adjust the parameters of the model.

Evaluation - Direct Method

- Evaluation consists in determining the probability of an observation sequence, $O = \{o_1, o_2, o_3, ...\}$, given a model, λ, that is, estimating $P(O \mid \lambda)$
- A sequence of observations, $O = \{o_1, o_2, o_3, ...o_T\}$, can be generated by different state sequences
- To calculate the probability of an observation sequence, we can estimate it for a certain state sequence, and then add the probabilities for all the possible state sequences:

$$P(O \mid \lambda) = \sum_i P(O, Q_i \mid \lambda) \quad (8)$$

- Where:

$$P(O, Q_i \mid \lambda) = \pi_1 b_1(o_1) a_{12} b_2(o_2) ... a_{(T-1)T} b_T(o_T) \quad (9)$$

Direct Method

- Thus, the probability of O is given by a summation over all the possible state sequences, Q:

$$P(O \mid \lambda) = \sum_{Q} \pi_1 b_1(o_1) a_{12} b_2(o_2) \ldots a_{(T-1)T} b_T(o_T) \quad (10)$$

- For a model with N states and an observation length of T, there are N^T possible state sequences. Each term in the summation requires $2T$ operations. As a result, the evaluation requires a number of operations in the order of $2T \times N^T$
- A more efficient method is required!

Evaluation - iterative method

- The basic idea of the iterative method, also known as *Forward*, is to estimate the probabilities of the states/observations per time step
- Calculate the probability of a partial sequence of observations until time t, and based on this partial result, calculate it for time $t + 1$, and so on ...
- Until the last stage is reached and the probability of the complete sequence is obtained.

Iterative method

- Define an auxiliary variable called *forward*:

$$\alpha_t(i) = P(o_1, o_2, ..., o_t, S_t = q_i \mid \lambda) \tag{11}$$

- The iterative algorithm consists of three main parts:
  - Initialization – the α variables for all states at the initial time are obtained: $\alpha_1(i) = P(O_1, S_1 = q_i) = \pi_i b_i(O_1)$
  - Induction – calculate $\alpha_{t+1}(i)$ in terms of $\alpha_t(i)$: $\alpha_t(j) = [\sum_i \alpha_{t-1}(i) a_{ij}] b_j(O_t)$
  - Termination – $P(O \mid \lambda)$ is obtained by adding all the α_T: $P(O) = \sum_i \alpha_T(i)$

Complexity

- Each iteration requires N multiplications and N additions (approx.), so for the T iterations, the number of operations is in the order of $N^2 \times T$
- The time complexity is reduced from exponential in T for the direct method to linear in T and quadratic in N for the iterative method

State Estimation

- Finding the most probable sequence of states for an observation sequence, $O = \{o_1, o_2, o_3, ...\}$, can be interpreted in two ways: (i) obtaining the most probable state, S_t at each time step t, (ii) obtaining the most probable sequence of states, $s_0, s_1, ...s_T$
- First we solve the problem of finding the most probable or *optimum* state for a certain time t, and then the problem of finding the *optimum* sequence

Auxiliary variables

- The *backward* variable is analogous to the forward one, but in this case we start from the end of the sequence, that is:

$$\beta_t(i) = P(o_{t+1}, o_{t+2}, ..., o_T, S_t = q_i \mid \lambda) \qquad (12)$$

- In a similar way to α, $\beta_t(i)$ can be calculated iteratively but now backwards:

$$\beta_t(i) = \sum_j \beta_{t+1}(j) a_{ij} b_j(o_t) \qquad (13)$$

The β variables for T are defined as $\beta_T(j) = 1$

- So $P(O \mid \lambda)$ can be obtained in terms of β or a combination of α and β

Most probable state

- γ, that is the conditional probability of being in a certain state q_i given the observation sequence:

$$\gamma_t(i) = P(s_t = q_i \mid O, \lambda) = P(s_t = q_i, O \mid \lambda)/P(O) \quad (14)$$

- Which can be written in terms of α and β as:

$$\gamma_t(i) = \alpha_t(i)\beta_t(i) / \sum_i \alpha_t(i)\beta_t(i) \quad (15)$$

- This variable, γ, provides the answer to the first subproblem, the most probable state (MPS) at a time t; we just need to find for which state it has the maximum value:

$$MPS(t) = ArgMax_i \gamma_t(i) \quad (16)$$

Most probable sequence

- The most probable state sequence Q given the observation sequence O, such that we want to maximize $P(Q \mid O, \lambda)$
- By Bayes rule: $P(Q \mid O, \lambda) = P(Q, O \mid \lambda)/P(O)$. Given that $P(O)$ does not depend on Q, this is equivalent to maximizing $P(Q, O \mid \lambda)$
- The method for obtaining the optimum state sequence is known as the *Viterbi* algorithm

Viterbi Algorithm

- δ gives the maximum value of the probability of a subsequence of states and observations until time t, being at state q_i at time t; that is:

$$\delta_t(i) = MAX[P(s_1, s_2, ...s_t = q_i, o_1, o_2, ..., o_t \mid \lambda)] \quad (17)$$

- Which can also be obtained in an iterative way:

$$\delta_{t+1}(i) = [MAX \delta_t(i) a_{ij}] b_j(o_{t+1}) \quad (18)$$

- The Viterbi algorithm requires four phases: initialization, recursion, termination and backtracking. It requires an additional variable, $\psi_t(i)$, that stores for each state i at each time step t the previous state that gave the maximum probability - used to reconstruct the sequence by backtracking

Algorithm

FOR $i = 1$ to N (Initialization)

- $\delta_1(i) = \pi_i b_i(O_1)$
- $\psi_1(i) = 0$

FOR $t = 2$ to T (recursion) FOR $j = 1$ to N

- $\delta_t(j) = MAX_i[\delta_{t-1}(i) a_{ij}] b_j(O_t)$
- $\psi_t(j) = ARGMAX_i[\delta_{t-1}(i) a_{ij}]$

$P^* = MAX_i[\delta_T(i)]$ (Termination)
$q^*_T = ARGMAX_i[\delta_T(i)]$
FOR $t = T$ to 2 (Backtracking)

- $q^*_{t-1} = \psi_t(q^*_t)$

Parameter Learning

- This method assumes that the *structure* of the model is known: the number of states and observations is previously defined; therefore it only estimates the parameters
- The Baum-Welch algorithm determines the parameters of a HMM, $\lambda = A, B, \Pi$, given a number of observation sequences, $\mathbf{O} = O_1, O_2, ...O_K$
- It maximizes the probability of the model given the observations: $P(\mathbf{O} \mid \lambda)$

Auxiliary Variables

- ξ, the probability of a transition from a state i at time t to a state j at time $t+1$ given an observation sequence O:

$$\xi_t(i,j) = P(s_t = q_i, s_{t+1} = q_j \mid O, \lambda) \tag{19}$$

$$\xi_t(i,j) = P(s_t = q_i, s_{t+1} = q_j, O \mid \lambda)/P(O) \tag{20}$$

- In terms of α and β:

$$\xi_t(i,j) = \alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)/P(O) \tag{21}$$

- γ can also be written in terms of ξ: $\gamma_t(i) = \sum_j \xi_t(i,j)$
- By adding $\gamma_t(i)$ for all time steps, $\sum_t \gamma_t(i)$, we obtain an estimate of the number of times that the chain is in state i; and by accumulating $\xi_t(i,j)$ over t, $\sum_t \xi_t(i,j)$, we estimate the number of transitions from state i to state j

Baum-Welch Algorithm

1. Estimate the prior probabilities – the number of times being in state i at time t.
   $\pi_i = \gamma_1(i)$
2. Estimate the transition probabilities – the number of transitions from state i to j between the number of times in state i.
   $a_{ij} = \sum_t \xi_t(i,j) / \sum_t \gamma_t(i)$
3. Estimate the observation probabilities – the number of times being in state j and observing k between the number of times in state j.
   $b_{jk} = \sum_{t,O=k} \gamma_t(i) / \sum_t \gamma_t(i)$

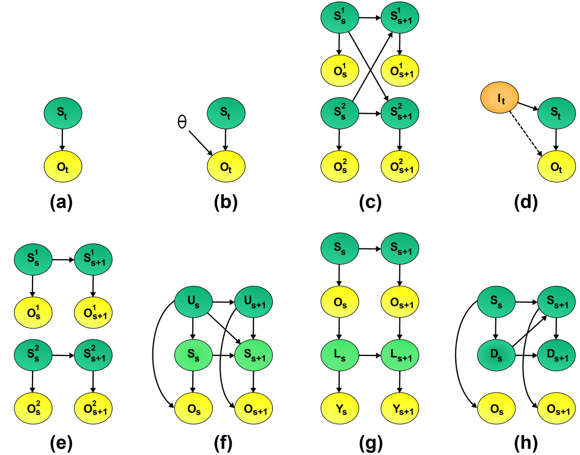
Expectation-Maximization

- Notice that the calculation of γ and ξ variables is done in terms of α and β, which require the parameters of the HMM, Π, A, B. So we have encountered a “chicken and egg” problem!
- The solution to this problem is based on the EM (for expectation-maximization) principle
- The idea is to start with some initial parameters for the model (E-step), $\lambda = \{A, B, \Pi\}$, which can be initialized randomly or based on some domain knowledge
- Then, via the Baum-Welch algorithm, these parameters are re-estimated (M-step)
- This cycle is repeated until convergence

Extensions to the basic HMM model

- Several extensions for the basic HMM have been proposed
- Some of these are illustrated in the next slide

Extensions



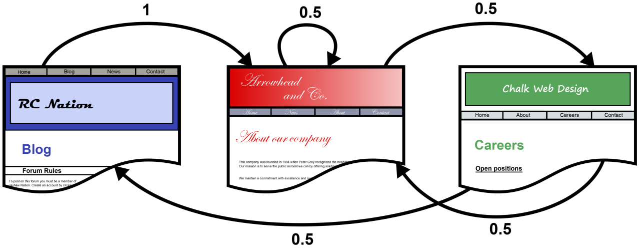
(a) Basic model. (b) Parametric HMMs. (c) Coupled HMMs. (d) Input-Output HMMs. (e) Parallel HMMs. (f) Hierarchical HMMs. (g) Mixed-state dynamic Bayesian networks. (h) Hidden semi-Markov models

Applications

- Markov chains for ordering web pages with the PageRank algorithm
- Application of HMMs in gesture recognition

WWW as a HMM

- We can think of the World Wide Web (WWW) as a very large Markov chain, such that each web page is a state and the hyperlinks between web pages correspond to state transitions
- Each outgoing link can be selected with equal probability; the transition probability from w_i to any of the web pages with which it has hyperlinks, w_j, is $A_{ij} = 1/m$



PageRank

- Given the transition probability matrix of the WWW, we can obtain the convergence probabilities for each state (web page) according to the Perron-Frobenius theorem
- The convergence probability of a certain web page can be thought to be equivalent to the probability of a person, who is navigating the WWW, visiting this web page.
- Based on the previous ideas, L. Page et al. developed the *PageRank* algorithm which is the basis of how web pages are ordered when we make a search in *Google*

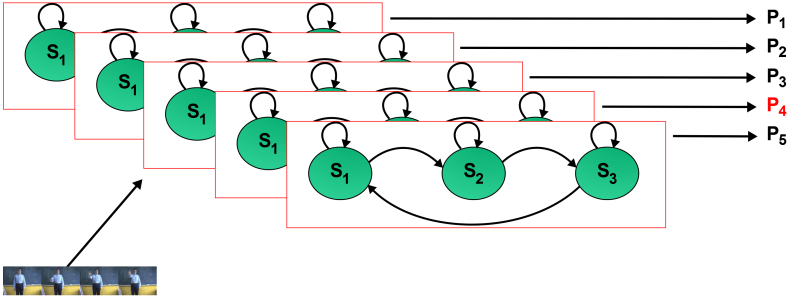
Gestures

Gestures are essential for human-human communication, so they are also important for human-computer interaction. For example, we can use gestures to command a service robot

Gesture Recognition

- For recognizing gestures, a powerful option is a hidden Markov model
- Before we can apply HMMs to model and recognize gestures, the images in the video sequence need to be processed and a set of features extracted from them; these will constitute the observations for the HMM
- To recognize N different gestures, we need to train N HMMs, one for each gesture using the Baum-Welch algorithm
- For recognition, the features are extracted from the video sequence. The probability of each model given the observation sequence, $P(O \mid \lambda_i)$, are obtained using the Forward algorithm. The model with the highest probability, λ_k^*, is selected as the recognized gesture

Gesture Recognition



Book

Sucar, L. E, *Probabilistic Graphical Models*, Springer 2015 – Chapter 5

Additional Reading

- Aviles, H., Sucar, L.E., Mendoza, C.E., Pineda, L.A.: A Comparison of Dynamic Naive Bayesian Classifiers and HMMs for Gesture Recognition. JART 9(1) (2011)
- Kanungo, T.: Hidden Markov Models Software. http://www.kanungo.com/.
- Page, L., Brin, S., Motwani, R., Winograd, T.: The PageRank Citation Ranking: Bringing Order to the Web, Stanford Digital Libraries Working Paper, 1998.
- Rabiner, L.E.: A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. In: Waibel A., Lee, K. (eds.) Readings in speech recognition, Morgan Kaufmann, 267-296 (1990)
- Rabiner, L., Juang, B.H.: Fundamentals on Speech Recognition. Prentice-Hall, New Jersey (1993)