PDEEC – Machine Learning 2018/19

Lecture
Context Dependent Classification
Sequential data
Hidden Markov Models

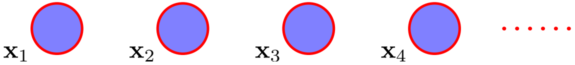
Jaime S. Cardoso
jaime.cardoso@inesctec.pt

INESC TEC and Faculdade Engenharia, Universidade do Porto

Jan, 03, 2019

Context Dependent Classification

Introduction



- Sets of data points assumed to be independent and identically distributed (i.i.d) so far
- i.i.d is a poor assumption for sequential data
  - measurements of time series (rainfall), daily values of a currency exchange rate, acoustic features in speech recognition
  - sequence of nucleotide base pairs along a strand of DNA, sequence of characters in an English sentence

- In general

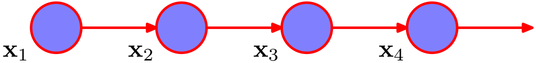
$$P(\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N) = P(\mathbf{x}_1) \prod_{n=2}^{N} P(\mathbf{x}_n | \mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_{n-1})$$

- Markov model: Each of the conditional distributions is independent of all previous observations except M most recent

Context Dependent Classification

The first-order Markov chain

- Homogeneous Markov chain



- Joint distribution for a sequence of N observations

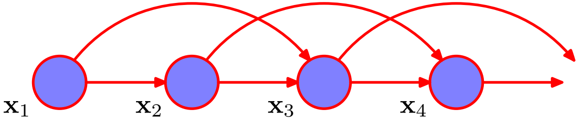
$$P(\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N) = P(\mathbf{x}_1) \prod_{n=2}^{N} P(\mathbf{x}_n | \mathbf{x}_{n-1})$$

- $$P(\mathbf{x}_n | \mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_{n-1}) = P(\mathbf{x}_n | \mathbf{x}_{n-1})$$

Context Dependent Classification

A higher-order Markov chain

The second-order Markov chain



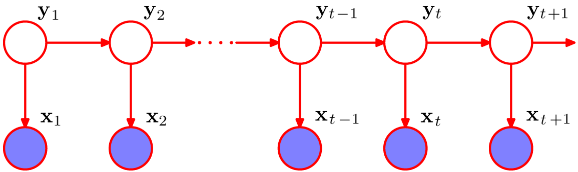
- The joint distribution

$$P(\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N) = P(\mathbf{x}_1)P(\mathbf{x}_2|\mathbf{x}_1)\prod_{n=3}^{N} P(\mathbf{x}_n|\mathbf{x}_{n-1}, \mathbf{x}_{n-2})$$

- A higher-order Markov chain
- Suppose the observations are discrete variables having K states
- first-order: $K-1$ parameters for each K states $\rightarrow K(K-1)$ parameters
- Mth-order: $K^M(K-1)$ parameters

Context Dependent Classification

Hidden Markov models (HMM)

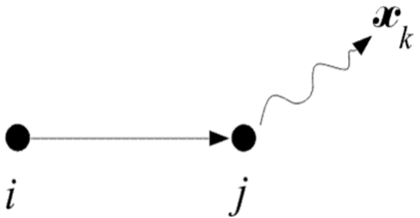


- $\mathbf{y}_t$ latent variables (discrete)
- $\mathbf{x}_t$ observed variables
- The joint distribution of the state space model

$$P(\mathbf{x}_1, \cdots, \mathbf{x}_T, \mathbf{y}_1, \cdots, \mathbf{y}_T) = P(\mathbf{y}_1) \prod_{t=2}^{T} P(\mathbf{y}_t | \mathbf{y}_{t-1}) \prod_{t=1}^{T} P(\mathbf{x}_t | \mathbf{y}_t)$$

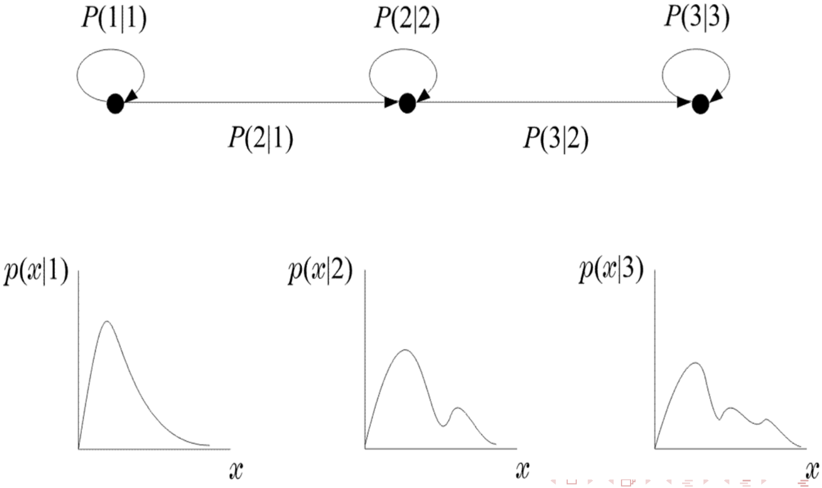
Hidden Markov Models

- An HMM is a stochastic finite state automaton, that generates the observation sequence, $\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N$
- We assume that: The observation sequence is produced as a result of successive transitions between states, upon arrival at a state:



Hidden Markov Models

- This type of modeling is used for nonstationary stochastic processes that undergo distinct transitions among a set of different stationary processes.

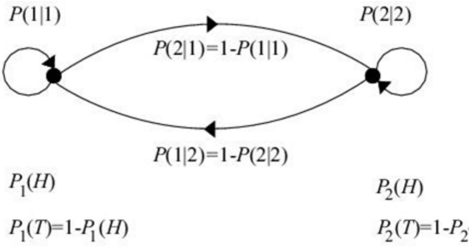


Hidden Markov Models

Example of HMM

- The two-coins case: Assume one of two coins is tossed behind a curtain. We observe a sequence of H or T. However, we have no access to know which coin was tossed. Identify one state for each coin. This is an example where states are not observable. H or T can be emitted from either state. The model depends on four parameters.

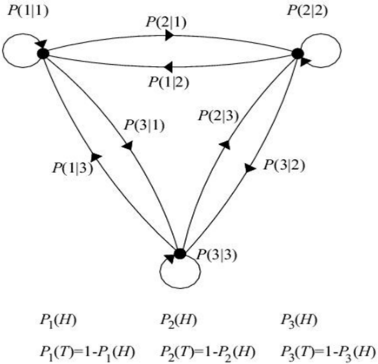
$$P_1(H), P_2(H), P(1|1), P(2|2)$$



Hidden Markov Models

Example of HMM

- The three-coins case example is shown below:



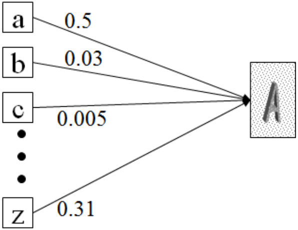
- Note that in all previous examples, specifying the model is equivalent to knowing:
  - The probability of each observation (H,T) to be emitted from each state.
  - The transition probabilities among states: $P(i|j)$.

Hidden Markov Models

Word recognition example(I)

- Typed word recognition, assume all characters are separated.

- Character recognizer outputs probability of the image being particular character, P(image—character).



Hidden Markov Models

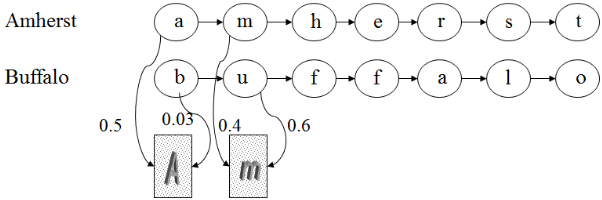
Word recognition example(II)

- Hidden states of HMM = characters.
- Observations = typed images of characters segmented from the image. Note that there is an infinite number of observations.
- Observation probabilities = character recognizer scores.
- Transition probabilities will be defined differently in two subsequent models.

Hidden Markov Models

Word recognition example(III)

- If lexicon is given, we can construct separate HMM models for each lexicon word.

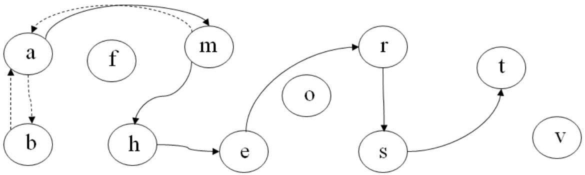


- Here recognition of word image is equivalent to the problem of evaluating few HMM models
- This is an application of Evaluation problem.

Hidden Markov Models

Word recognition example(IV)

- We can construct a single HMM for all words.
- Hidden states = all characters in the alphabet.
- Transition probabilities and initial probabilities are calculated from language model.
- Observations and observation probabilities are as before

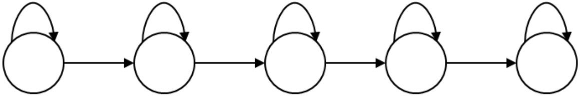


- Here we have to determine the best sequence of hidden states, the one that most likely produced word image.
- This is an application of Decoding problem

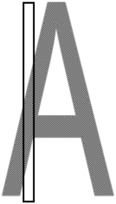
Hidden Markov Models

Character recognition with HMM example

- The structure of hidden states is chosen



- Observations are feature vectors extracted from vertical slices.

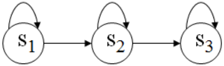


- Probabilistic mapping from hidden state to feature vectors:
  - use mixture of Gaussian models.
  - Quantize feature vector space

Hidden Markov Models

Exercise: character recognition with HMM(I)

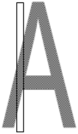
- The structure of hidden states:



- Observation = number of islands in the vertical slice.
- HMM for character 'A' :

Transition probabilities: $\{a_{ij}\}=\begin{pmatrix} .8 & .2 & 0 \\ 0 & .8 & .2 \\ 0 & 0 & 1 \end{pmatrix}$

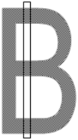
Observation probabilities: $\{b_{jk}\}=\begin{pmatrix} .9 & .1 & 0 \\ .1 & .8 & .1 \\ .9 & .1 & 0 \end{pmatrix}$



- HMM for character 'B' :

Transition probabilities: $\{a_{ij}\}=\begin{pmatrix} .8 & .2 & 0 \\ 0 & .8 & .2 \\ 0 & 0 & 1 \end{pmatrix}$

Observation probabilities: $\{b_{jk}\}=\begin{pmatrix} .9 & .1 & 0 \\ 0 & .2 & .8 \\ .6 & .4 & 0 \end{pmatrix}$



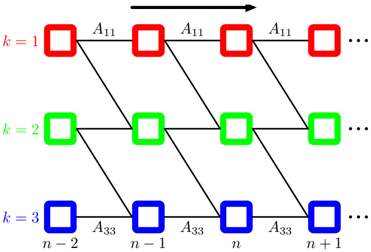
Exercise: character recognition with HMM(II)

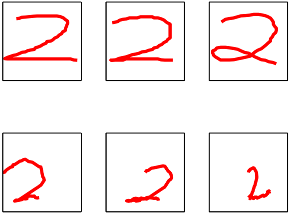
- Suppose that after character image segmentation the following sequence of island numbers in 4 slices was observed: $\{1, 3, 2, 1\}$
- What HMM is more likely to generate this observation sequence , HMM for ‘A’ or HMM for ‘B’ ?

Context Dependent Classification

HMM applications

- Speech recognition
- Natural language modelling
- Analysis of biological sequences (e.g. proteins and DNA)
- On-line handwriting recognition; Example: Handwritten digits
  - Left-to-right architecture
  - On-line data: each digit represented by the trajectory of the pen as a function of time

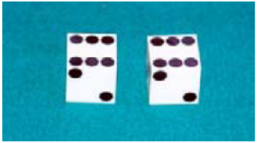




A casino has two dice:

- Fair die
  $P(1) = P(2) = P(3) = P(5) = P(6) = 1/6$
- Loaded die
  $P(1) = P(2) = P(3) = P(5) = 1/10$
  $P(6) = 1/2$

In average, casino player switches back-&-forth between fair and loaded die once every 20 turns



Hidden Markov Models

Main Questions Regarding the Dishonest Casino

GIVEN: A sequence of rolls by the casino player

1245526462146146136136661664661636616366163616515615115146123562344

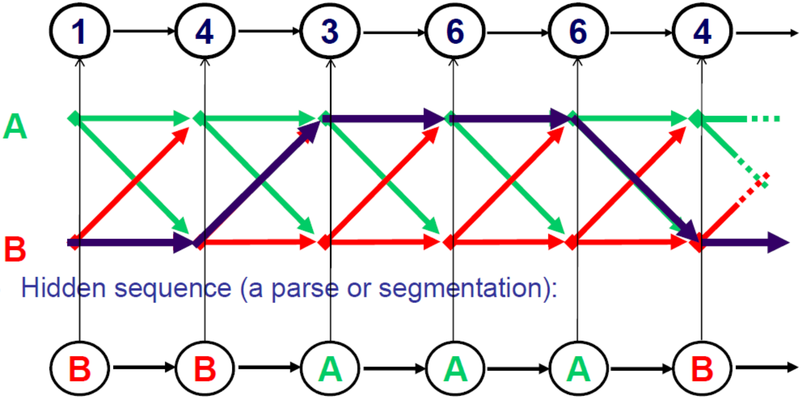
QUESTION

- How likely is this sequence, given our model of how the casino works? (This is the EVALUATION problem)
- What portion of the sequence was generated with the fair die, and what portion with the loaded die? This is the DECODING question
- How "loaded" is the loaded die? How "fair" is the fair die? How often does the casino player change from fair to loaded, and back? This is the LEARNING question

Hidden Markov Models

Definition (of HMM)

- Observed sequence:



- Hidden sequence (a parse or segmentation):

Hidden Markov Models

An HMM is a Stochastic Generative Model

- **Observation space**

  Alphabetic set: $\mathrm{C} = \{c_1, c_2, \cdots, c_K\}$

  Euclidean space: $\mathbb{R}^d$

- **Index set of hidden states**

  $\mathrm{I} = \{1, 2, \cdots, M\}$

- **Transition probabilities between any two states**

  $p(y_t^j = 1 \mid y_{t-1}^i = 1) = a_{i,j}$,

  or $p(y_t \mid y_{t-1}^i = 1) \sim \mathrm{Multinomial}(a_{i,1}, a_{i,2}, \ldots, a_{i,M}), \forall i \in \mathrm{I}.$

- **Start probabilities**

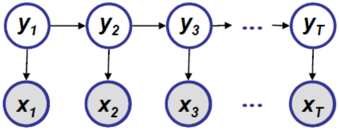
  $p(y_1) \sim \mathrm{Multinomial}(\pi_1, \pi_2, \ldots, \pi_M).$

- **Emission probabilities associated with each state**

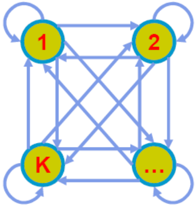
  $p(x_t \mid y_t^i = 1) \sim \mathrm{Multinomial}(b_{i,1}, b_{i,2}, \ldots, b_{i,K}), \forall i \in \mathrm{I}.$

  or in general:

  $p(x_t \mid y_t^i = 1) \sim \mathrm{f}(\cdot \mid \theta_i), \forall i \in \mathrm{I}.$



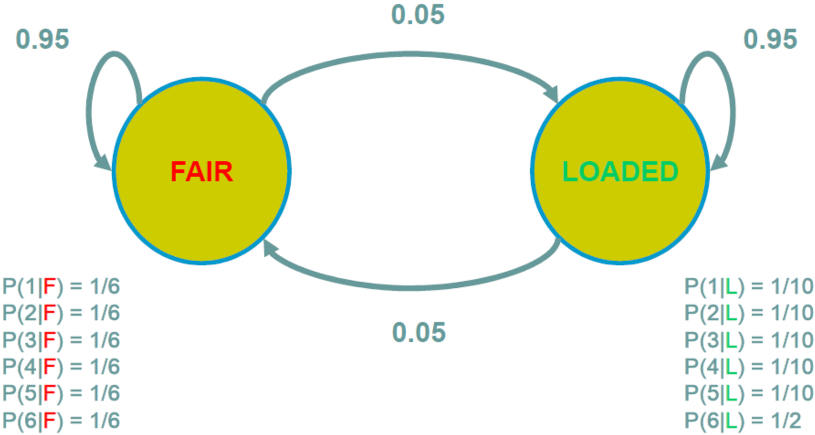
Graphical model



State automata

Hidden Markov Models

The Dishonest Casino Model



1. Evaluation

GIVEN an HMM $\boldsymbol{M}$, and a sequence $\boldsymbol{x}$,

FIND Prob ($\boldsymbol{x}$ | $\boldsymbol{M}$)

ALGO. Forward

2. Decoding

GIVEN an HMM $\boldsymbol{M}$, and a sequence $\boldsymbol{x}$,

FIND the sequence $\mathbf{y}$ of states that maximizes, e.g., P($\boldsymbol{y}$ | $\boldsymbol{x}$, $\boldsymbol{M}$), or the most probable subsequence of states

ALGO. Viterbi, Forward-backward

3. Learning

GIVEN an HMM $\boldsymbol{M}$, with unspecified transition/emission probs., and a sequence $\boldsymbol{x}$,

FIND parameters $\theta = (\pi_i, a_{ij}, \eta_{ik})$ that maximize P($\boldsymbol{x}$ | θ)

ALGO. Baum-Welch (EM)

Hidden Markov Models

Joint Probability

1245526462146146136136661664661636616366163616515615115146123562344

FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF

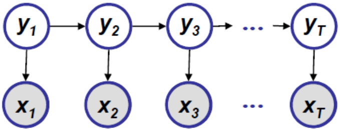
- When the state-labeling is known, this is easy...

$$P(X, Y) = P(\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_T, \mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_T)$$

Hidden Markov Models

Probability of a Parse

- Given a sequence $\mathbf{x} = x_1 \ldots\ldots x_T$ and a parse $\mathbf{y} = y_1, \ldots\ldots, y_T,$
- To find how likely is the parse: (given our HMM and the sequence)

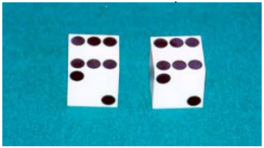


$$\begin{aligned} p(\mathbf{x}, \mathbf{y}) &= p(x_1 \ldots\ldots x_T, y_1, \ldots\ldots, y_T) \qquad \text{(Joint probability)} \\ &= p(y_1)\, p(x_1 \mid y_1)\, p(y_2 \mid y_1)\, p(x_2 \mid y_2) \ldots p(y_T \mid y_{T-1})\, p(x_T \mid y_T) \\ &= p(y_1)\, P(y_2 \mid y_1) \ldots p(y_T \mid y_{T-1}) \times p(x_1 \mid y_1)\, p(x_2 \mid y_2) \ldots p(x_T \mid y_T) \end{aligned}$$

- Marginal probability: $p(\mathbf{x}) = \sum_{\mathbf{y}} p(\mathbf{x}, \mathbf{y}) = \sum_{y_1} \sum_{y_2} \cdots \sum_{y_N} \pi_{y_1} \prod_{t=2}^{T} p(y_t \mid y_{t-1}) \prod_{t=1}^{T} p(x_t \mid y_t)$
- Posterior probability: $p(\mathbf{y} \mid \mathbf{x}) = p(\mathbf{x}, \mathbf{y}) / p(\mathbf{x})$

Hidden Markov Models

Example: the Dishonest Casino



- Let the sequence of rolls be:
  - x = 1, 2, 1, 5, 6, 2, 1, 6, 2, 4
- Then, what is the likelihood of
  - y = Fair, Fair, Fair, Fair, Fair, Fair, Fair, Fair, Fair, Fair?

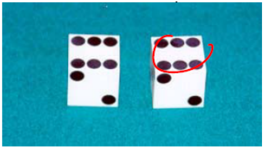
    (say initial probs $a_{0Fair} = ½$, $a_{0Loaded} = ½$)

½ × P(1 | Fair) P(Fair | Fair) P(2 | Fair) P(Fair | Fair) … P(4 | Fair) =

$½ \times (1/6)^{10} \times (0.95)^{9}$ = .00000000521158647211 = 5.21×10^{-9}

Hidden Markov Models

Example: the Dishonest Casino



- So, the likelihood the die is fair in all this run is just 5.21×10^{-9}

- OK, but what is the likelihood of
  - π = Loaded, Loaded, Loaded, Loaded, Loaded, Loaded, Loaded, Loaded, Loaded, Loaded?

$$\tfrac{1}{2} \times P(1 \mid \text{Loaded})\, P(\text{Loaded} \mid \text{Loaded}) \ldots P(4 \mid \text{Loaded}) =$$

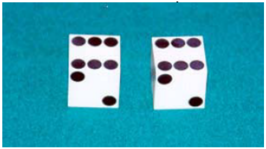
$$\tfrac{1}{2} \times (1/10)^8 \times (1/2)^2\, (0.95)^9 = .00000000078781176215 = 0.79 \times 10^{-9}$$

- Therefore, it is after all 6.59 times more likely that the die is fair all the way, than that it is loaded all the way

Hidden Markov Models

Example: the Dishonest Casino

- Let the sequence of rolls be:
  - $x = 1, 6, 6, 5, 6, 2, 6, 6, 3, 6$



- Now, what is the likelihood π = F, F, …, F?
  - $\frac{1}{2} \times (1/6)^{10} \times (0.95)^9 = 0.5 \times 10^{-9}$, same as before

- What is the likelihood y = L, L, …, L?

$$\frac{1}{2} \times (1/10)^4 \times (1/2)^6 (0.95)^9 = .00000049238235134735 = 5 \times 10^{-7}$$

- So, it is 100 times more likely the die is loaded

Hidden Markov Models

Marginal Probability

1245526462146146136136661664661636616366163616515615115146123562344

- What if state-labeling Y is not observed

$$P(X) = \sum_{Y} P(X, Y)$$

- the wrong way would be to...

We want to calculate $P(\mathbf{x})$, the likelihood of $\mathbf{x}$, given the HMM

- Sum over all possible ways of generating $\mathbf{x}$:

$$p(\mathbf{x}) = \sum_{\mathbf{y}} p(\mathbf{x}, \mathbf{y}) = \sum_{y_1} \sum_{y_2} \cdots \sum_{y_M} \pi_{y_1} \prod_{t=2}^{T} a_{y_{t-1}, y_t} \prod_{t=1}^{T} p(x_t \mid y_t)$$

- To avoid summing over an exponential number of paths $\mathbf{y}$, define

$$\alpha(y_t^k = 1) = \alpha_t^k \stackrel{\text{def}}{=} P(x_1, \ldots, x_t, y_t^k = 1)$$

(the forward probability)

- The recursion:

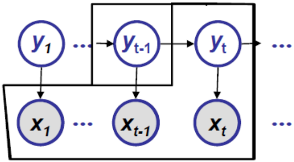
$$\alpha_t^k = p(x_t \mid y_t^k = 1) \sum_i \alpha_{t-1}^i a_{i,k}$$

$$P(\mathbf{x}) = \sum_k \alpha_T^k$$

Hidden Markov Models

The Forward Algorithm

- Compute the forward probability:



$$\alpha_t^k = P(x_1,\ldots,x_{t-1},x_t,y_t^k=1)$$

$$= \sum_{y_{t-1}} P(x_1,\ldots,x_{t-1},y_{t-1})P(y_t^k=1 \mid y_{t-1},x_1,\ldots,x_{t-1})P(x_t \mid y_t^k=1,x_1,\ldots,x_{t-1},y_{t-1})$$

$$= \sum_{y_{t-1}} P(x_1,\ldots,x_{t-1},y_{t-1})P(y_t^k=1 \mid y_{t-1})P(x_t \mid y_t^k=1)$$

$$= P(x_t \mid y_t^k=1)\sum_i P(x_1,\ldots,x_{t-1},y_{t-1}^i=1)P(y_t^k=1 \mid y_{t-1}^i=1)$$

$$= P(x_t \mid y_t^k=1)\sum_i \alpha_{t-1}^i a_{i,k}$$

Chain rule: $P(A,B,C)=P(A)P(B \mid A)P(C \mid A,B)$

Hidden Markov Models

The Forward Algorithm

- We can compute α_t^k for all k, t, using dynamic programming!

Initialization:

$$\alpha_1^k = P(x_1 \mid y_1^k = 1)\pi_k$$

$$\begin{aligned}\alpha_1^k &= P(x_1, y_1^k = 1) \\ &= P(x_1 \mid y_1^k = 1)P(y_1^k = 1) \\ &= P(x_1 \mid y_1^k = 1)\pi_k\end{aligned}$$

Iteration:

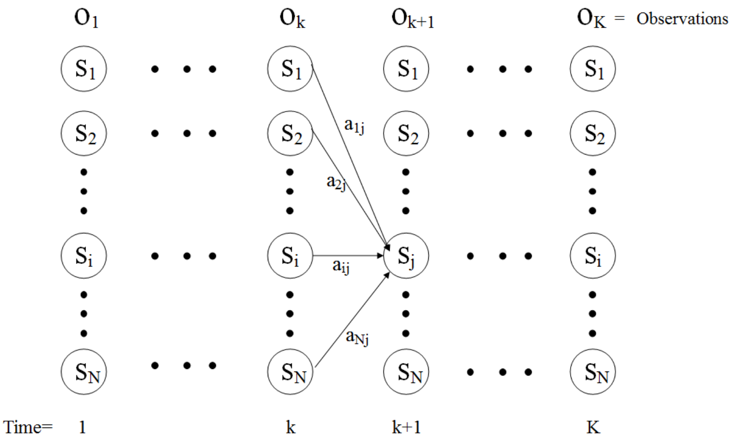
$$\alpha_t^k = P(x_t \mid y_t^k = 1)\sum_i \alpha_{t-1}^i a_{i,k}$$

Termination:

$$P(\mathbf{x}) = \sum_k \alpha_T^k$$

Hidden Markov Models

Recognition: Any path method

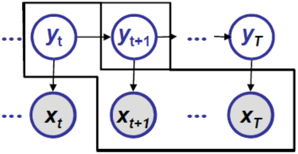


- we are exploiting the graph structure (the joint distribution) to do an efficient inference
- the algorithm can be expressed in terms of the propagation of local messages around the graph
- each summation effectively removes a variable from the distribution, which can be viewed as the removal of a node from the graph

Hidden Markov Models

The Backward Algorithm

- Define the backward probability:



$$
\begin{aligned}
\beta_t^k &= P(x_{t+1},\ldots,x_T \mid y_t^k = 1) \\
&= \sum_{y_{t+1}} P(x_{t+1},\ldots,x_T, y_{t+1} \mid y_t^k = 1) \\
&= \sum_i P(y_{t+1}^i = 1 \mid y_t^k = 1)\, p(x_{t+1} \mid y_{t+1}^i = 1, y_t^k = 1)\, P(x_{t+2},\ldots,x_T \mid x_{t+1}, y_{t+1}^i = 1, y_t^k = 1) \\
&= \sum_i P(y_{t+1}^i = 1 \mid y_t^k = 1)\, p(x_{t+1} \mid y_{t+1}^i = 1)\, P(x_{t+2},\ldots,x_T \mid y_{t+1}^i = 1) \\
&= \sum_i a_{k,i}\, p(x_{t+1} \mid y_{t+1}^i = 1)\, \beta_{t+1}^i
\end{aligned}
$$

Chain rule: $P(A,B,C \mid \alpha) = P(A \mid \alpha) P(B \mid A,\alpha) P(C \mid A,B,\alpha)$

- We can compute β_t^k for all k, t, using dynamic programming!

Initialization:

$$\beta_T^k = 1, \ \forall k$$

Iteration:

$$\beta_t^k = \sum_i a_{k,i} P(x_{t+1} \mid y_{t+1}^i = 1) \beta_{t+1}^i$$

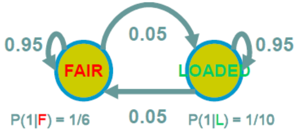
Termination:

$$P(\mathbf{x}) = \sum_k \alpha_1^k \beta_1^k$$

Hidden Markov Models

Example

x = 1, 2, 1, 5, 6, 2, 1, 6, 2, 4



P(1|F) = 1/6	P(1|L) = 1/10
P(2|F) = 1/6	P(2|L) = 1/10
P(3|F) = 1/6	P(3|L) = 1/10
P(4|F) = 1/6	P(4|L) = 1/10
P(5|F) = 1/6	P(5|L) = 1/10
P(6|F) = 1/6	P(6|L) = 1/2

Alpha (actual)		Beta (actual)	
0.0833	0.0500	0.0000	0.0000
0.0136	0.0052	0.0000	0.0000
0.0022	0.0006	0.0000	0.0000
0.0004	0.0001	0.0000	0.0000
0.0001	0.0000	0.0001	0.0001
0.0000	0.0000	0.0007	0.0006
0.0000	0.0000	0.0045	0.0055
0.0000	0.0000	0.0264	0.0112
0.0000	0.0000	0.1633	0.1033
0.0000	0.0000	1.0000	1.0000

$$\alpha_t^k = P(x_t \mid y_t^k = 1)\sum_i \alpha_{t-1}^i a_{i,k}$$

$$\beta_t^k = \sum_i a_{k,i} P(x_{t+1} \mid y_{t+1}^i = 1)\beta_{t+1}^i$$

Hidden Markov Models

What is the probability of a hidden state prediction?

- A single state:

$$P(\mathbf{y}_t|\mathbf{X})$$

- What about a hidden state sequence ?

$$P(\mathbf{y}_1, \cdots, \mathbf{y}_T)|\mathbf{X})$$

Hidden Markov Models

Posterior decoding

- We can now calculate

$$P(y_t^k = 1 \mid \mathbf{x}) = \frac{P(y_t^k = 1, \mathbf{x})}{P(\mathbf{x})} = \frac{\alpha_t^k \beta_t^k}{P(\mathbf{x})}$$

- Then, we can ask
  - What is the most likely state at position *t* of sequence $\mathbf{x}$:

$$k_t^* = \arg\max_k P(y_t^k = 1 \mid \mathbf{x})$$

  - Note that this is an MPA of a single hidden state, what if we want to a MPA of a whole hidden state sequence?
  - Posterior Decoding: $\left\{ y_t^{k_t^*} = 1 : t = 1 \cdots T \right\}$
  - This is different from MPA of a whole sequence of hidden states
  - This can be understood as *bit error rate* vs. *word error rate*

Example:
MPA of *X* ?
MPA of *(X, Y)* ?

x	y	$P(x,y)$
0	0	0.35
0	1	0.05
1	0	0.3
1	1	0.3

Hidden Markov Models

Viterbi decoding

- GIVEN $\mathbf{x} = x_1, \ldots, x_T$, we want to find $\mathbf{y} = y_1, \ldots, y_T$, such that $P(\mathbf{y}|\mathbf{x})$ is maximized:

$$\mathbf{y}^* = \text{argmax}_{\mathbf{y}} P(\mathbf{y}|\mathbf{x}) = \text{argmax}_{\pi} P(\mathbf{y},\mathbf{x})$$

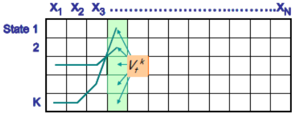
  - Let

$$V_t^k = \max_{\{y_1, \ldots y_{t-1}\}} P(x_1, \ldots, x_{t-1}, y_1, \ldots, y_{t-1}, x_t, y_t^k = 1)$$

= Probability of most likely **sequence of states** ending at state $y_t = k$

  - The recursion:

$$V_t^k = p(x_t \mid y_t^k = 1) \max_i a_{i,k} V_{t-1}^i$$



  - Underflows are a significant problem

$$p(x_1, \ldots, x_t, y_1, \ldots, y_t) = \pi_{y_1} a_{y_1, y_2} \cdots a_{y_{t-1}, y_t} b_{y_1, x_1} \cdots b_{y_t, x_t}$$

    - These numbers become extremely small – underflow
    - Solution: Take the logs of all values: $V_t^k = \log p(x_t \mid y_t^k = 1) + \max_i \left(\log(a_{i,k}) + V_{t-1}^i \right)$

- Define the viterbi probability:

$$
\begin{aligned}
V_{t+1}^{k} &= \max_{\{y_1,\dots,y_t\}} P(x_1,\dots,x_t,y_1,\dots,y_t,x_{t+1},y_{t+1}^{k}=1) \\
&= \max_{\{y_1,\dots,y_t\}} P(x_{t+1},y_{t+1}^{k}=1 \mid x_1,\dots,x_t,y_1,\dots,y_t) P(x_1,\dots,x_t,y_1,\dots,y_t) \\
&= \max_{\{y_1,\dots,y_t\}} P(x_{t+1},y_{t+1}^{k}=1 \mid y_t) P(x_1,\dots,x_{t-1},y_1,\dots,y_{t-1},x_t,y_t) \\
&= \max_{i} P(x_{t+1},y_{t+1}^{k}=1 \mid y_t^{i}=1) \max_{\{y_1,\dots,y_{t-1}\}} P(x_1,\dots,x_{t-1},y_1,\dots,y_{t-1},x_t,y_t^{i}=1) \\
&= \max_{i} P(x_{t+1} \mid y_{t+1}^{k}=1) a_{i,k} V_t^{i} \\
&= P(x_{t+1} \mid y_{t+1}^{k}=1) \max_{i} a_{i,k} V_t^{i}
\end{aligned}
$$

- Input: $\mathbf{x} = \mathbf{x}_1, \cdots, \mathbf{x}_T$

Initialization:

$$V_1^k = P(x_1 \mid y_1^k = 1)\pi_k$$

Iteration:

$$V_t^k = P(x_t \mid y_t^k = 1)\max_i a_{i,k} V_{t-1}^i$$

$$\mathrm{Ptr}(k,t) = \arg\max_i a_{i,k} V_{t-1}^i$$

Termination:

$$P(\mathbf{x}, \mathbf{y}^*) = \max_k V_T^k$$

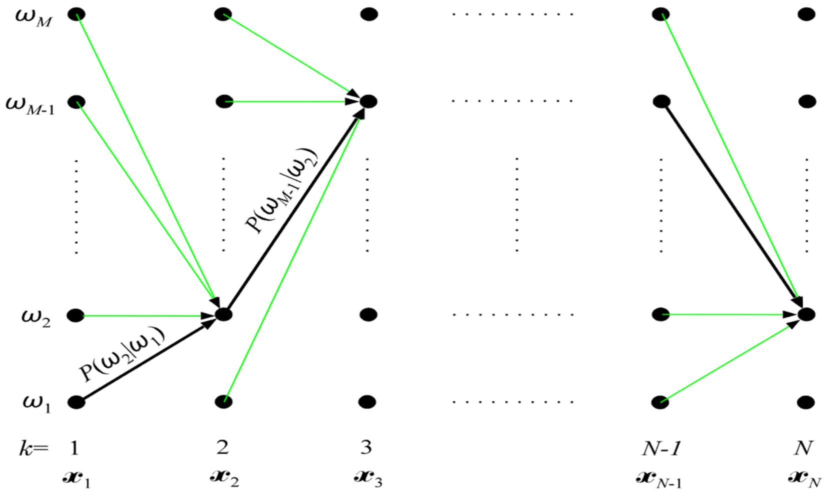
TraceBack:

$$y_T^* = \arg\max_k V_T^k$$

$$y_{t-1}^* = \mathrm{Ptr}(y_t^*, t)$$

Context Dependent Classification

The Viterbi Algorithm



Hidden Markov Models

Computational Complexity and implementation details

- What is the running time, and space required, for Forward, and Backward?

$$\alpha_t^k = p(x_t \mid y_t^k = 1) \sum_i \alpha_{t-1}^i a_{i,k}$$

$$\beta_t^k = \sum_i a_{k,i} p(x_{t+1} \mid y_{t+1}^i = 1) \beta_{t+1}^i$$

$$V_t^k = p(x_t \mid y_t^k = 1) \max_i a_{i,k} V_{t-1}^i$$

Time: $O(K^2 N)$; Space: $O(KN)$.

- Useful implementation technique to avoid underflows
  - Viterbi: sum of logs
  - Forward/Backward: rescaling at each position by multiplying by a constant

1. Evaluation

GIVEN an HMM M, and a sequence x,
FIND Prob (x | M)
ALGO. Forward

2. Decoding

GIVEN an HMM M, and a sequence x,
FIND the sequence y of states that maximizes, e.g., P(y | x, M), or the most probable subsequence of states
ALGO. Viterbi, Forward-backward

3. Learning

GIVEN an HMM M, with unspecified transition/emission probs., and a sequence x,
FIND parameters $\theta = (\pi_i, a_{ij}, \eta_{ik})$ that maximize P(x | θ)
ALGO. Baum-Welch (EM)

- Supervised learning: estimation when the "right answer" is known
  - Examples: GIVEN: a genomic region $x = x_1 \cdots x_{1\ 000\ 000}$ where we have good (experimental) annotations of the CpG islands
    GIVEN: the casino player allows us to observe him one evening, as he changes dice and produces 10 000 rolls
- Unsupervised learning: estimation when the "right answer" is unknown
  - Examples: GIVEN: the porcupine genome; we don't know how frequent are the CpG islands there, neither do we know their composition
    GIVEN: 10000 rolls of the casino player, but we don't see when he changes dice
- QUESTION: Update the parameters θ of the model to maximize $P(\mathbf{X}|\theta)$ — Maximal likelihood (ML) estimation

Hidden Markov Models

Supervised ML estimation

- Given $x = x_1 \ldots x_N$ for which the true state path $y = y_1 \ldots y_N$ is known,
  - **Define:**

    A_{ij} = # times state transition $i \to j$ occurs in $\mathbf{y}$

    B_{ik} = # times state i in $\mathbf{y}$ emits k in $\mathbf{x}$

  - **We can show that the maximum likelihood parameters θ are:**

$$a_{ij}^{ML} = \frac{\#(i \to j)}{\#(i \to \bullet)} = \frac{\sum_n \sum_{t=2}^{T} y_{n,t-1}^i y_{n,t}^j}{\sum_n \sum_{t=2}^{T} y_{n,t-1}^i} = \frac{A_{ij}}{\sum_{j'} A_{ij'}}$$

$$b_{ik}^{ML} = \frac{\#(i \to k)}{\#(i \to \bullet)} = \frac{\sum_n \sum_{t=1}^{T} y_{n,t}^i x_{n,t}^k}{\sum_n \sum_{t=1}^{T} y_{n,t}^i} = \frac{B_{ik}}{\sum_{k'} B_{ik'}}$$

  - **What if y is continuous? We can treat $\{(x_{n,t}, y_{n,t}) : t = 1:T, n = 1:N\}$ as $N \times T$ observations of, e.g., a Gaussian, and apply learning rules for Gaussian …**

Hidden Markov Models

Supervised ML estimation

- **Intuition:**
  - When we know the underlying states, the best estimate of θ is the average frequency of transitions & emissions that occur in the training data

- **Drawback:**
  - Given little data, there may be **overfitting**:
    - $P(x|\theta)$ is maximized, but θ is unreasonable

      0 probabilities – VERY BAD

- **Example:**
  - Given 10 casino rolls, we observe

    x = 2, 1, 5, 6, 1, 2, 3, 6, 2, 3

    y = F, F, F, F, F, F, F, F, F, F

  - Then: $a_{FF} = 1$; $a_{FL} = 0$

    $b_{F1} = b_{F3} = .2$;

    $b_{F2} = .3$; $b_{F4} = 0$; $b_{F5} = b_{F6} = .1$

Hidden Markov Models

Pseudocounts

- Solution for small training sets:
  - Add pseudocounts

    A_{ij} = # times state transition $i \rightarrow j$ occurs in $\mathbf{y}$ + R_{ij}

    B_{ik} = # times state i in $\mathbf{y}$ emits k in $\mathbf{x}$ + S_{ik}

  - R_{ij}, S_{ij} are pseudocounts representing our prior belief
  - Total pseudocounts: $R_i = \Sigma_j R_{ij}$, $S_i = \Sigma_k S_{ik}$,
    - --- "strength" of prior belief,
    - --- total number of imaginary instances in the prior
- Larger total pseudocounts $\Rightarrow$ strong prior belief
- Small total pseudocounts: just to avoid 0 probabilities --- smoothing

Hidden Markov Models

Unsupervised ML estimation

- Given $x = x_1 \ldots x_N$ for which the true state path $y = y_1 \ldots y_N$ is unknown,

- **EXPECTATION MAXIMIZATION**

0. Starting with our best guess of a model M, parameters θ.
1. Estimate A_{ij} , B_{ik} in the training data
   - How? $A_{ij} = \sum_{n,t} \langle y^i_{n,t-1} y^j_{n,t} \rangle$ $\quad B_{ik} = \sum_{n,t} \langle y^i_{n,t} \rangle x^k_{n,t}$,
   - Update θ according to A_{ij} , B_{ik}
   - Now a "supervised learning" problem
2. Repeat 1 & 2, until convergence

This is called the Baum-Welch Algorithm

We can get to a provably more (or equally) likely parameter set θ each iteration

Hidden Markov Models

Unsupervised ML estimation

The Baum Welch algorithm

- The complete log likelihood

$$\ell_c(\boldsymbol{\theta}; \mathbf{x}, \mathbf{y}) = \log p(\mathbf{x}, \mathbf{y}) = \log \prod_n \left(p(y_{n,1}) \prod_{t=2}^{T} p(y_{n,t} \mid y_{n,t-1}) \prod_{t=1}^{T} p(x_{n,t} \mid x_{n,t}) \right)$$

- The expected complete log likelihood

$$\langle \ell_c(\boldsymbol{\theta}; \mathbf{x}, \mathbf{y}) \rangle = \sum_n \left(\langle y_{n,1}^i \rangle_{p(y_{n,1} \mid \mathbf{x}_n)} \log \pi_i \right) + \sum_n \sum_{t=2}^{T} \left(\langle y_{n,t-1}^i y_{n,t}^j \rangle_{p(y_{n,t-1}, y_{n,t} \mid \mathbf{x}_n)} \log a_{i,j} \right) + \sum_n \sum_{t=1}^{T} \left(x_{n,t}^k \langle y_{n,t}^i \rangle_{p(y_{n,t} \mid \mathbf{x}_n)} \log b_{i,k} \right)$$

- EM
  - The **E** step

$$\gamma_{n,t}^i = \langle y_{n,t}^i \rangle = p(y_{n,t}^i = 1 \mid \mathbf{x}_n)$$

$$\xi_{n,t}^{i,j} = \langle y_{n,t-1}^i y_{n,t}^j \rangle = p(y_{n,t-1}^i = 1, y_{n,t}^j = 1 \mid \mathbf{x}_n)$$

  - The **M** step ("symbolically" identical to MLE)

$$\pi_i^{ML} = \frac{\sum_n \gamma_{n,1}^i}{N} \qquad a_{ij}^{ML} = \frac{\sum_n \sum_{t=2}^{T} \xi_{n,t}^{i,j}}{\sum_n \sum_{t=1}^{T-1} \gamma_{n,t}^i} \qquad b_{ik}^{ML} = \frac{\sum_n \sum_{t=1}^{T} \gamma_{n,t}^i x_{n,t}^k}{\sum_n \sum_{t=1}^{T-1} \gamma_{n,t}^i}$$

The Baum Welch algorithm

Maximum likelihood for the HMM

- We have observed a data set $\{\mathbf{x}_1, \cdots, \mathbf{x}_T\}$ (or possibly multiples sequences)
- so we can determine the parameters of an HMM

$$\theta = \{\pi, A, \phi\}$$

by using maximum likelihood (ϕ represent the parameters of the emission probabilities).

- The likelihood function is

$$p(\mathbf{X}|\theta) = \sum_{\mathbf{y}} p(\mathbf{X}, \mathbf{y}|\theta)$$

The Baum Welch algorithm

Maximizing the likelihood function

Expectation maximization algorithm (EM)

- Initial selection for the model parameters: θ^{old}
- E step:
  - Posterior distribution of the latent variables $p(\mathbf{y}|\mathbf{X}, \theta^{\text{old}})$

$$Q(\theta, \theta^{\text{old}}) = \sum_{\mathbf{y}} p(\mathbf{y}|\mathbf{X}, \theta^{\text{old}}) \ln p(\mathbf{X}, \mathbf{y}|\theta)$$

Hidden Markov Models

Unsupervised ML estimation

The Baum Welch algorithm

Maximizing the likelihood function: EM

E step:

$$Q(\theta, \theta^{\text{old}}) = \sum_{k=1}^{K} \gamma(y_{1k}) \ln \pi_k + \sum_{t=2}^{T} \sum_{j=1}^{K} \sum_{k=1}^{K} \xi(\mathbf{y}_{t-1,j}, \mathbf{y}_{tk}) \ln A_{jk}$$

$$+ \sum_{t=1}^{T} \sum_{k=1}^{K} \gamma(\mathbf{y}_{tk}) \ln p(\mathbf{x}_t | \phi_k)$$

- The marginal posterior distribution of a latent variable γ and the joint posterior distribution of two successive latent variables ξ

$$\gamma(\mathbf{y}_t) = p(\mathbf{y}_t | \mathbf{X}, \theta^{\text{old}})$$

$$\xi(\mathbf{y}_{t-1}, \mathbf{y}_t) = p(\mathbf{y}_{t-1}, \mathbf{y}_t | \mathbf{X}, \theta^{\text{old}})$$

The Baum Welch algorithm

Maximizing the likelihood function: EM

M step:

- Maximize $Q(\theta, \theta^{\text{old}})$ with respect to parameters $\theta = \{\pi, A, \phi\}$, treat $\gamma(\mathbf{y}_t)$ and $\xi(\mathbf{y}_{t-1}, \mathbf{y}_t)$ as constant. By using Lagrange multipliers

$$\pi_k = \frac{\gamma(y_{1k})}{\sum_{j=1}^{K} \gamma(y_{1j})}$$

$$A_{jk} = \frac{\sum_{t=2}^{T} \xi(y_{t-1,j}, y_{tk})}{\sum_{l=1}^{K} \sum_{t=2}^{T} \xi(y_{t-1,j}, y_{tl})}$$

The Baum Welch algorithm

Maximizing the likelihood function: EM

M step:

- Parameters ϕ_k independent
  $\rightarrow$ for Gaussian emission densities $p(\mathbf{x}|\phi_k) = \mathcal{N}(\mathbf{x}|\mu_k, \Sigma_k)$

$$\mu_k = \frac{\sum_{t=1}^{T} \gamma(y_{tk})\mathbf{x}_t}{\sum_{t=1}^{T} \gamma(y_{tk})}$$

$$\Sigma_k = \frac{\sum_{t=1}^{T} \gamma(y_{tk})(\mathbf{x}_t - \mu_k)(\mathbf{x}_t - \mu_k)^T}{\sum_{t=1}^{T} \gamma(y_{tk})}$$

The Baum Welch algorithm

$$\gamma(\mathbf{y}_t) = p(\mathbf{y}_t|\mathbf{X}) = \frac{p(\mathbf{X}|\mathbf{y}_t)p(\mathbf{y}_t)}{p(\mathbf{X})} = \frac{\alpha(\mathbf{y}_t)\beta(\mathbf{y}_t)}{p(\mathbf{X})}$$

$$\xi(\mathbf{y}_{t-1}, \mathbf{y}_t) = p(\mathbf{y}_{t-1}, \mathbf{y}_t|\mathbf{X}) = \frac{p(\mathbf{X}|y_{t-1}, \mathbf{y}_t)p(\mathbf{y}_{t-1}, \mathbf{y}_t)}{p(\mathbf{X})} =$$

$$\frac{\alpha(\mathbf{y}_{t-1})p(\mathbf{x}_t|\mathbf{y}_t)p(\mathbf{y}_t|\mathbf{y}_{t-1})\beta(\mathbf{y}_t)}{p(\mathbf{X})}$$

The Baum Welch algorithm - comments

Time Complexity: # iterations $\times \mathcal{O}(K^2 N)$

- Guaranteed to increase the log likelihood of the model
- Not guaranteed to find globally best parameters
- Converges to local optimum, depending on initial conditions
- Too many parameters / too large model: Over-fitting

References

Bishop
Pattern recognition and machine learning,
Book.

Trevor Hastie and Robert Tibshirani and Jerome Friedman
The elements of statistical learning,
Springer.

Eric Xing' Homepage
http://www.cs.cmu.edu/~epxing/
http://www.cs.cmu.edu/~epxing/Class/10701-08s/