Hidden Markov Models (HMM)

Motivating Example

Assume you have a little robot that is trying to ***estimate*** *the* ***posterior*** *probability that you are* ***happy*** *or* ***sad****, given that the robot has observed whether you are* ***watching Game of Thrones (w)****,* ***sleeping (s)****,* ***crying (c)*** *or* ***face booking (f)****.*

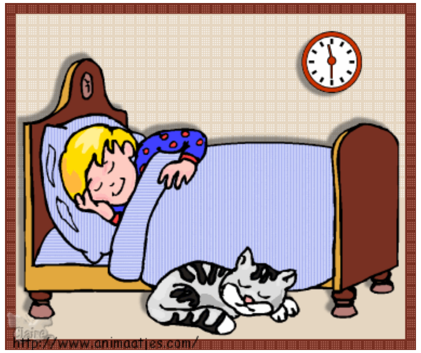
Let the ***unknown state*** *be* ***X=h*** *if you're happy and* ***X=s*** *if you're sad.*

Let ***Y*** *denote the* ***observation****, which can be* ***w****,* ***s****,* ***c*** *or* ***f*** *.*

Motivating Example

- The robot watches you over time, performing various activities (observables) and tries to infer your mood (state) at each instance of time.

W



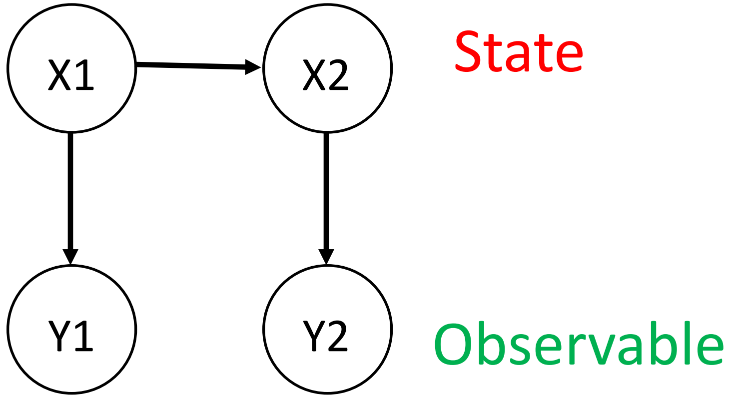
S

C

F

Based on these time series observations, robot needs to infer your mood

Motivating Example



- What data does the robot have?

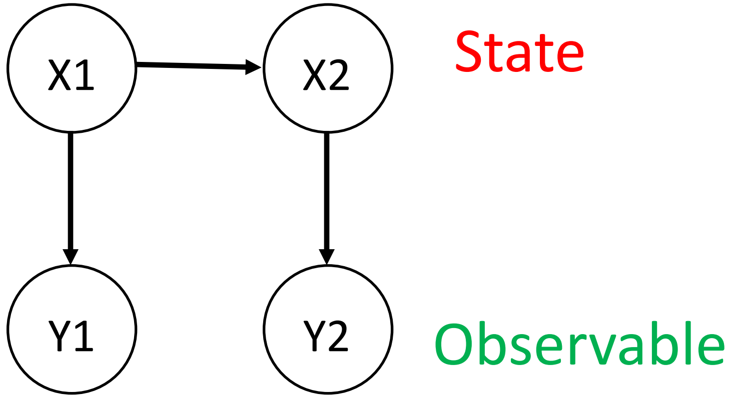
- Initial state probabilities (Prior probabilities). Let X represent state.

X = S	X = H
0.2	0.8

- Emission probability: Probability of doing something when you are in a certain state P(Y|X) e.g. if you are sad, how likely are you to be Facebooking?

	Y = W	Y = S	Y = C	Y = F
X = S	0.1	0.3	0.5	0.1
Y= H	0.3	0.4	0.1	0.2

Motivating Example



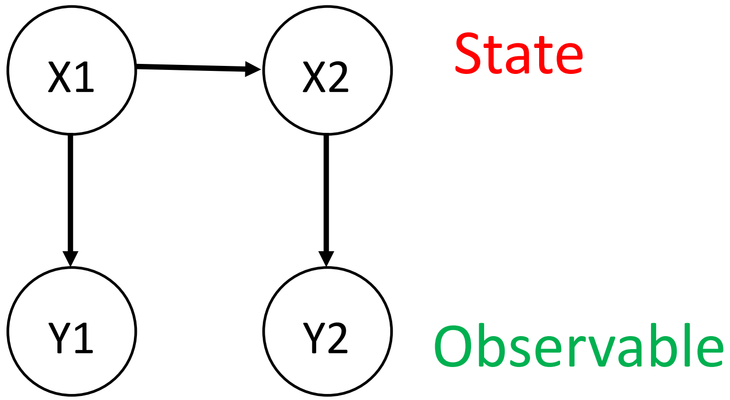
- What data does the robot have?

- Transition probabilities: Given you are in a certain state (X1), how likely are you to change states to X2 i.e. P(X = X2|X = X1)

	X2 = S	X2 = H
X1 = S	0.99	0.01
X1= H	0.1	0.9

Motivating Example

- With the above data, can you answer question below:



Robot observes you doing the following:
Y1 = W, Y2 = F, Y3 = C,
how likely are you to be in state X3 = H?
how likely is the following sequence - {H, S, H}

- This is the idea behind Hidden Markov Models

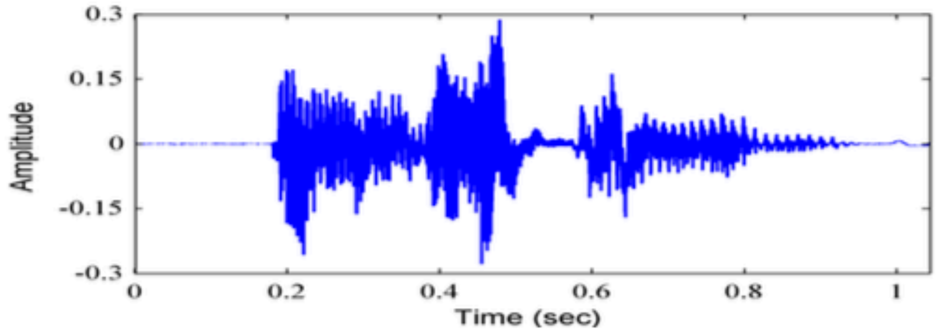
Application Areas for HMM

- Speech Recognition
  http://www.ece.ucsb.edu/Faculty/Rabiner/ece259/Reprints/tutorial%20on%20hmm%20and%20applications.pdf
- Natural Language Processing
  http://www3.cs.stonybrook.edu/~ychoi/cse628/lecture/06-hmm.pdf
- Handwriting (Character) Recognition
  https://www.cs.cmu.edu/~ekrevat/docs/HMM.pdf
- Analysis of genomic sequences and many other Bioinformatics tasks
  http://www.cs.columbia.edu/4761/notes07/chapter4.4-HMM.pdf
- Stock Market Prediction
  http://www.cs.cmu.edu/~bdhingra/papers/stock_hmm.pdf

i.i.d to sequential data

- So far we assumed independent, identically distributed data $\{X_i\}_{i=1}^n \overset{iid}{\sim} p(\mathbf{X})$
- Sequential data
  - Time-series data

    E.g. Speech

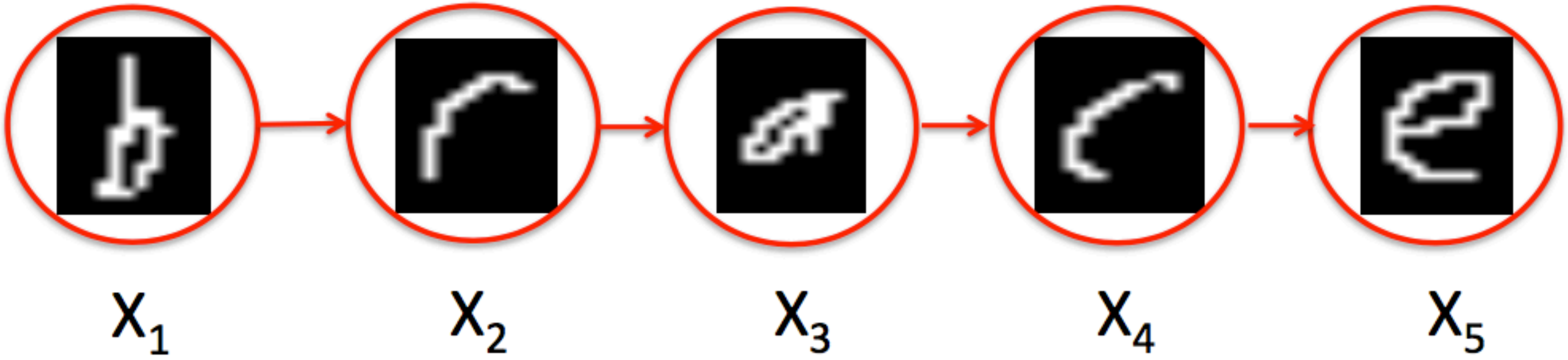


  - Characters in a sentence

  - Base pairs along a DNA strand

Representing sequential data

- How do we represent and learn $P(X_1, X_2, \ldots, X_n)$?



- Every variable depends on past few variables.

Markov Models

- Joint Distribution

$$\begin{aligned} p(\mathbf{X}) &= p(X_1, X_2, \ldots, X_n) \\ &= p(X_1)p(X_2|X_1)p(X_3|X_2, X_1)\ldots p(X_n|X_{n-1}, ..., X_1) \\ &= \prod_{i=1}^{n} p(X_i|X_{i-1}, \ldots, X_1) \end{aligned}$$

Chain rule

- Markov Assumption (m^{th} order)

$$p(\mathbf{X}) = \prod_{i=1}^{n} p(X_i|X_{i-1}, \ldots, X_{n-m})$$

Current observation only depends on past m observations

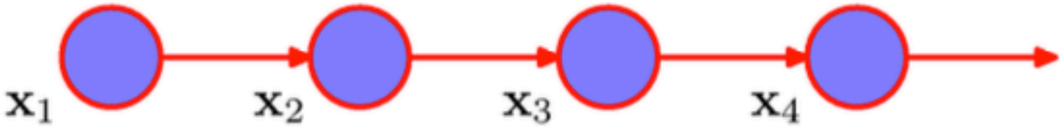
- Special case of Bayes Nets $p(\mathbf{X}) = \prod_{i=1}^{n} p(X_i|pa(X_i))$

Markov Models

- Markov Assumption

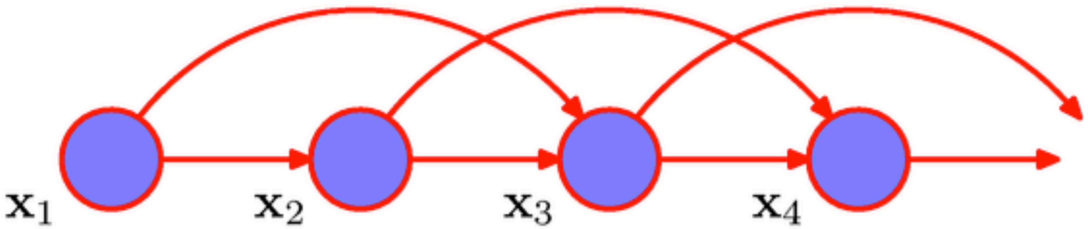
1st order

$$p(\mathbf{X}) = \prod_{i=1}^{n} p(X_n | X_{n-1})$$



2nd order

$$p(\mathbf{X}) = \prod_{i=1}^{n} p(X_n | X_{n-1}, X_{n-2})$$



Markov Models

parameters in stationary model K-ary variables

- Markov Assumption

1st order $p(\mathbf{X}) = \prod_{i=1}^{n} p(X_n|X_{n-1})$ $O(K^2)$

mth order $p(\mathbf{X}) = \prod_{i=1}^{n} p(X_n|X_{n-1}, \ldots, X_{n-m})$ $O(K^{m+1})$

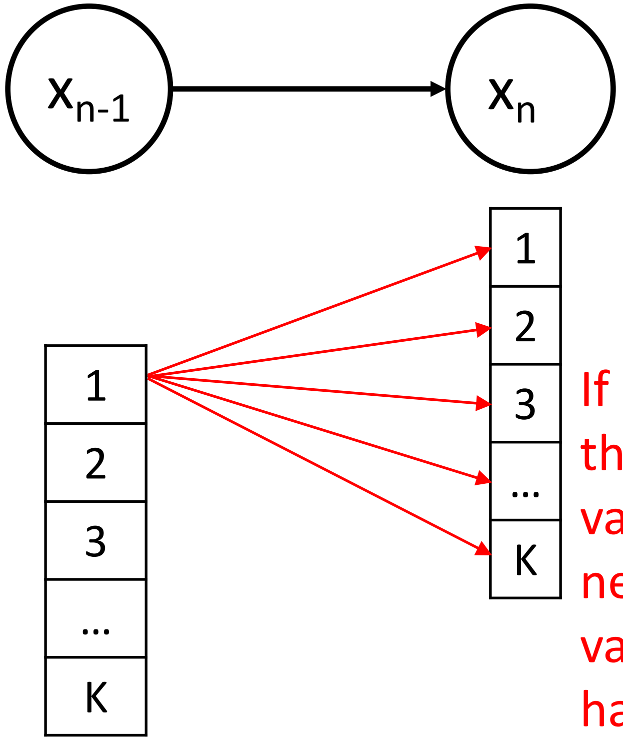
n-1th order $p(\mathbf{X}) = \prod_{i=1}^{n} p(X_n|X_{n-1}, \ldots, X_1)$ $O(K^n)$

≡ no assumptions – complete (but directed) graph

Homogeneous/stationary Markov model (probabilities don't depend on n)

How do we prove this

- For first order Markov model, x_{n-1} may be in any of the K states 1…K
- If I am in state i at node x_{n-1}, I can jump to any of the K states at node x_n. They have to sum to 1.0
- For each state of $x_{n-1} = i$, we need (K-1) probabilities for j = 1, 2, …,(K-1)
  $a_{ij} = p(x_n = j \mid x_{n-1} = i)$
- Total of K(K-1) parameters



If I have each of the first K-1 values, I don't need the last value as they have to sum to 1.0.

(K-1) parameters for each state of x_{n-1}. Total of K(K-1) parameters.

How do we prove this

- For m^{th} order Markov model, we need to take previous m nodes. Total of K^m combination of previous states.
- If I am in state i at node x_{n-1}, I can jump to any of the K states at node x_n. They have to sum to 1.0
- Total of $K^m(K-1)$ parameters. So $O(K^{m+1})$ parameters.

m columns

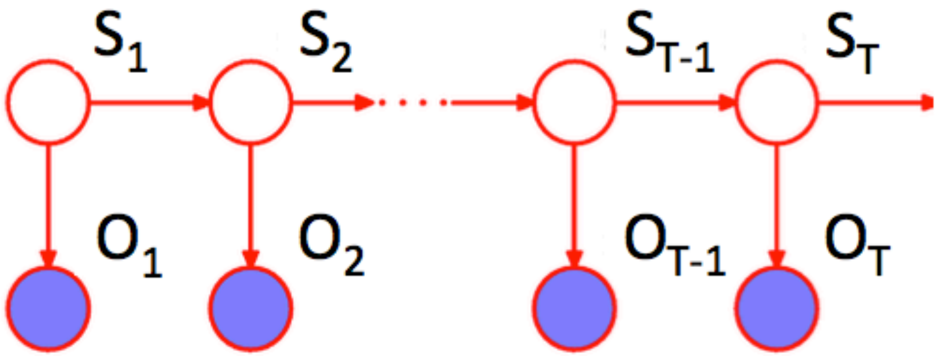
x_{n-1}	x_{n-2}	…	x_{n-m}	x_n
0	0	0	0	(K-1) params
..	…	..	..	
K	K	K	K	(K-1) params

K^m rows

Total of $K^m(K-1)$ parameters

Hidden Markov Models

- Parameters – stationary/homogeneous markov model (independent of time t)



Initial probabilities

$$p(S_1 = i) = \pi_i$$

Transition probabilities

$$p(S_t = j \mid S_{t-1} = i) = p_{ij}$$

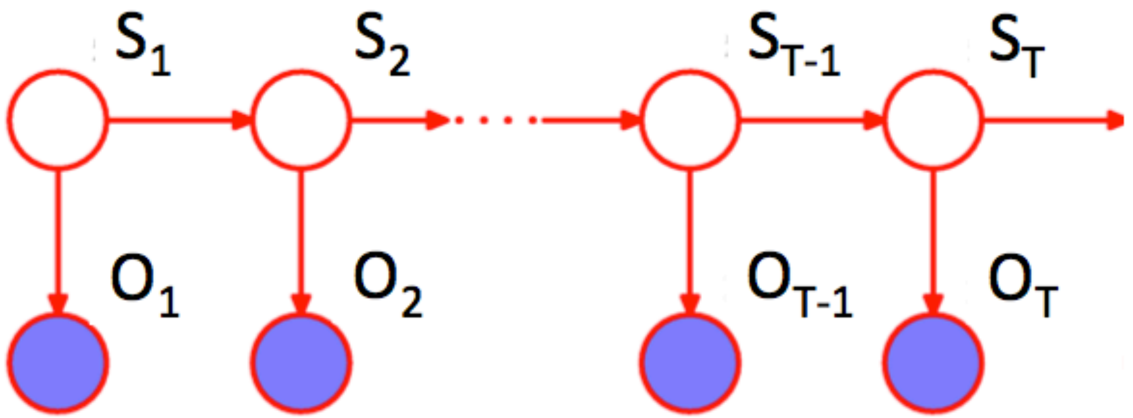
Emission probabilities

$$p(O_t = y \mid S_t = i) = q_i^y$$

$$p(\{S_t\}_{t=1}^T, \{O_t\}_{t=1}^T) = p(S_1)\prod_{t=2}^{T} p(S_t|S_{t-1})\prod_{t=1}^{T} p(O_t|S_t)$$

Hidden Markov Models

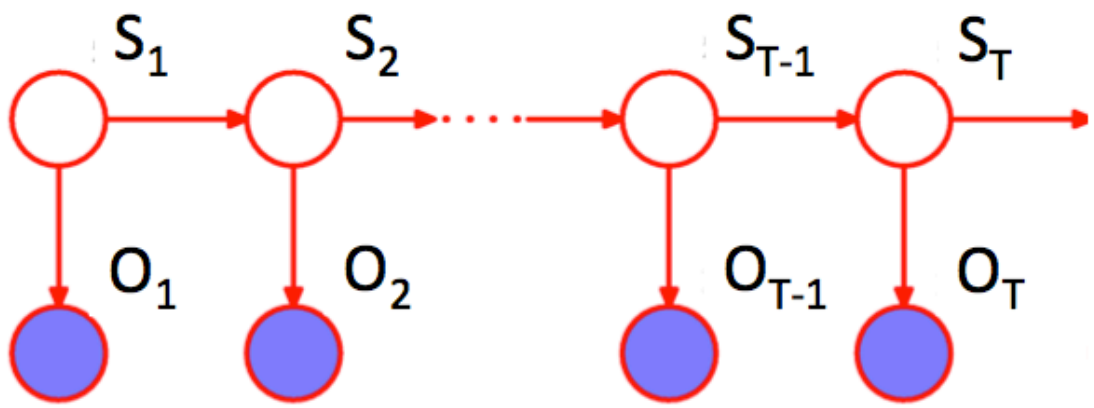
- Distributions that characterize sequential data with few parameters but are not limited by strong Markov assumptions.



Observation space $O_t \in \{y_1, y_2, \ldots, y_K\}$ e.g. pixels in 

Hidden states $S_t \in \{1, \ldots, I\}$ e.g. {a,b,c,…z}

Hidden Markov Models



$$p(S_1,\ldots,S_T,O_1,\ldots,O_T) = \prod_{t=1}^{T} p(O_t|S_t) \prod_{t=1}^{T} p(S_t|S_{t-1})$$

Note: The above equation allows use to evaluate the joint probability as a step-wise conditional probability

1st order Markov assumption on hidden states $\{S_t\}$ $t = 1, \ldots, T$

(can be extended to higher order).

Note: O_t depends on all previous observations $\{O_{t-1},\ldots O_1\}$

HMM Example

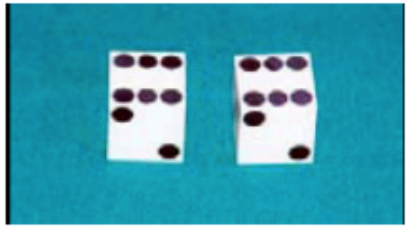
- The Dishonest Casino

A casino has two die:

Fair dice

Loaded dice

Casino player switches back-&-forth between fair and loaded die



HMM Example

- The Dishonest Casino

A casino has two dices:

Fair dice

P(1) = P(2) = P(3) = P(5) = P(6) = 1/6

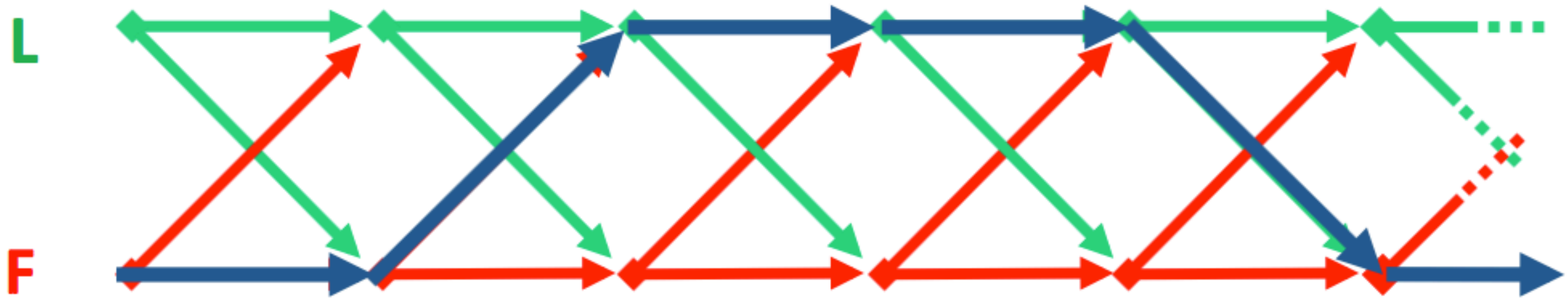
Loaded dice

P(1) = P(2) = P(3) = P(5) = 1/10
P(6) = ½

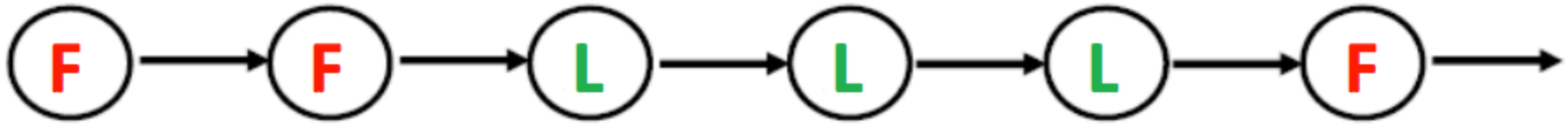
Casino player switches back-&-forth between fair and loaded die with 5% probability

HMM Example

- Observed sequence: $\{O_t\}_{t=1}^{T}$

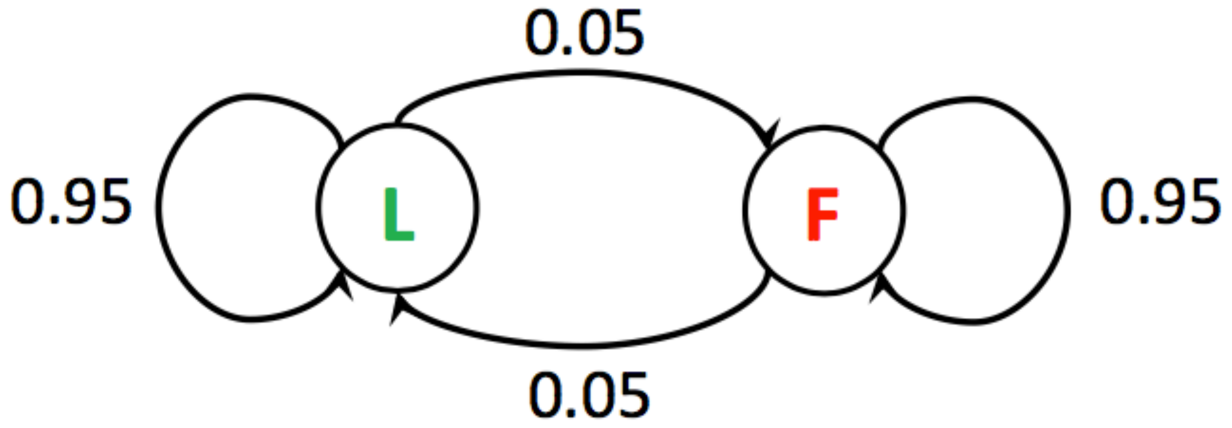


- Hidden sequence $\{S_t\}_{t=1}^{T}$ or segmentation):



State Space Representation

- Switch between F and L once every 20 turns (1/20 = 0.05)



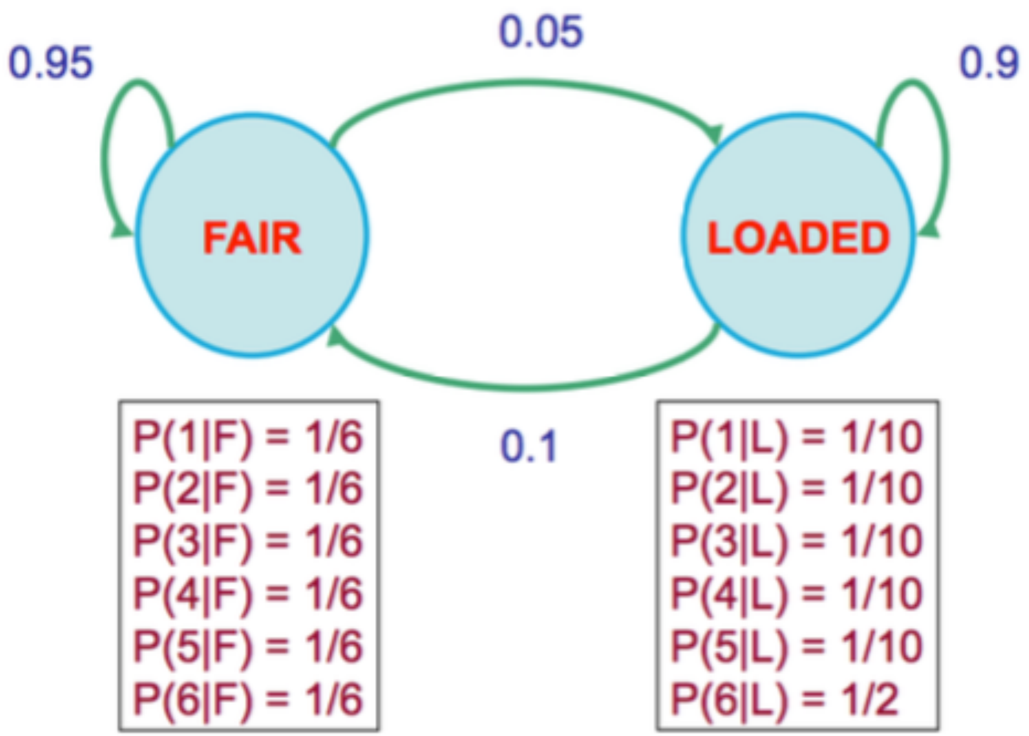
- HMM Parameters

Initial probs	$P(S_1 = L) = 0.5 = P(S_1 = F)$
Transition probs	$P(S_t = L/F \mid S_{t-1} = L/F) = 0.95$
	$P(S_t = F/L \mid S_{t-1} = L/F) = 0.05$
Emission probabilities	$P(O_t = y \mid S_t = F) = 1/6 \qquad y = 1,2,3,4,5,6$
	$P(O_t = y \mid S_t = L) = 1/10 \qquad y = 1,2,3,4,5$
	$= 1/2 \qquad y = 6$

Joint Probability Questions

You are given all the information (observation + states), you have to compute joint probability.

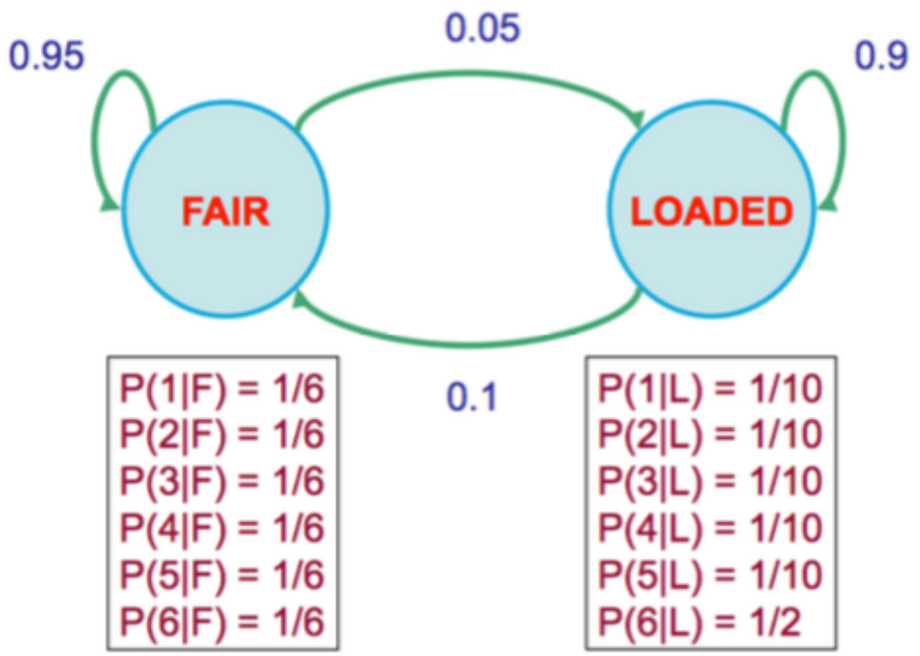
- Example:



Initial probability:
$a_{0,Fair}=a_{0,Loaded}= 0.5$

Find the probability of observable:
1, 2, 6, 6
Given the respective states:
F, F, L, L
i.e. P(1, 2, 6, 6, F, F, L, L)

You are given all information,
just need to find the joint probability



Initial probability:
$a_{0,Fair}=a_{0,Loaded}= 0.5$

Find the probability of observable:
1, 2, 6, 6
Given the respective states:
F, F, L, L
i.e P({1, 2, 6, 6}, {F, F, L, L})

Answer:

$=a_{0,Fair}*P(1|Fair) * a_{Fair,Fair}* P(2|Fair) * a_{Fair,Loaded}* P(6|Loaded) * a_{Loaded,Loaded}* P(6|Loaded)$

$= (0.5) * (1/6) * (0.95) * (1/6) * (0.05) * (1/2) * (0.95) * (1/2)$

$= 0.00016$

HMM Problems

GIVEN: A sequence of rolls by the casino player

124552646214614613613666166466163661636616361651561511514612356234

QUESTION

- How likely is this sequence, given our model of how the casino works?
  - This is the **EVALUATION** problem in HMMs
- What portion of the sequence was generated with the fair die, and what portion with the loaded die?
  - This is the **DECODING** question in HMMs
- 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 in HMMs

Three main problems in HMMs

- Evaluation – Given HMM parameters & observation seqn $\{O_t\}_{t=1}^T$

  find $p(\{O_t\}_{t=1}^T)$ prob of observed sequence

- Decoding – Given HMM parameters & observation seqn $\{O_t\}_{t=1}^T$

  find $\arg\max_{s_1,\ldots,s_T} p(\{S_t\}_{t=1}^T | \{O_t\}_{t=1}^T)$ most probable sequence of hidden states

- Learning – Given HMM with unknown parameters and $\{O_t\}_{t=1}^T$ observation sequence

  find $\arg\max_\theta p(\{O_t\}_{t=1}^T | \theta)$ parameters that maximize likelihood of observed data

HMM Algorithms

- Evaluation – What is the probability of the observed sequence? Forward Algorithm

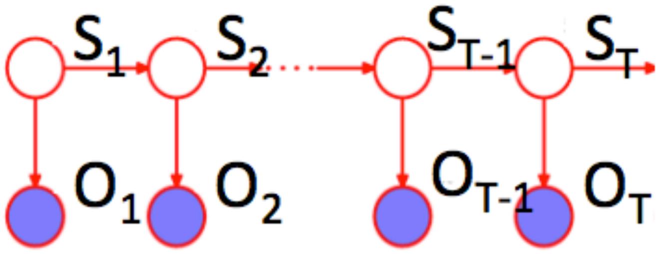
- Decoding – What is the probability that the third roll was loaded given the observed sequence? Forward-Backward Algorithm

  – What is the most likely die sequence given the observed sequence? Viterbi Algorithm

- Learning – Under what parameterization is the observed sequence most probable? Baum-Welch Algorithm (EM)

Evaluation Problem

- Given HMM parameters $p(S_1), p(S_t|S_{t-1}), p(O_t|S_t)$ & observation sequence $\{O_t\}_{t=1}^T$



find probability of observed sequence

Note: In this case, you don't know anything about the states. So you have to consider all possibilities.

$$p(\{O_t\}_{t=1}^T) = \sum_{S_1,\ldots,S_T} p(\{O_t\}_{t=1}^T, \{S_t\}_{t=1}^T)$$

requires summing over all possible hidden state values at all times – K^T exponential # terms! See next slide for explanation

Instead:

$$p(\{O_t\}_{t=1}^T) = \sum_k \underbrace{p(\{O_t\}_{t=1}^T, S_T = k)}_{\alpha_T^k}$$

Compute recursively

Explanation

- If you wanted to find $P(O_1, O_2, \ldots, O_T)$ using brute force approach, you would have to sum out (marginalize) over all possible values of state variables $(S_1, S_2, \ldots, S_T)$.
- Each S_i can be in one of K states.
- You have to take all possible combinations of these variables to sum out.
- Brute Force:

$$\sum_{s_1=1}^{K} \sum_{s_{2=1}}^{K} \ldots \sum_{s_T=1}^{K} P(S_1, S_2, \ldots, S_T, O_1, O_2, \ldots, O_T)$$

How many terms would there be in above summation? K^T

Forward Algorithm

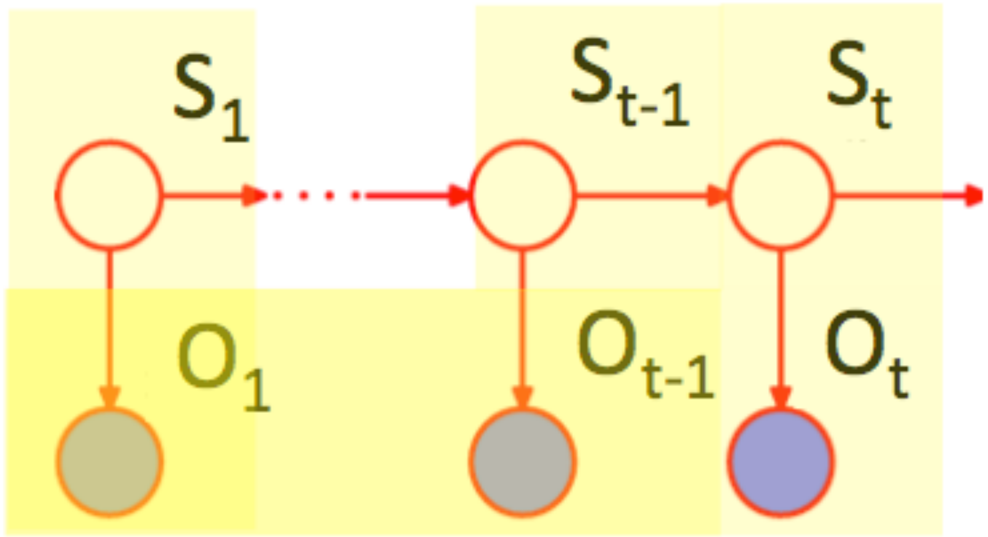
- Luckily, there is a better way than summing up K^T terms.
- We use the idea of dynamic programming.
- We introduce a term for forward probability as follows:
  α_t^k - probability of observing $\{O_1, O_2, \ldots, O_t\}$ and being in state k after t steps

$$\alpha_t^k = P(O_1, O_2, \ldots, O_t, S_t = k)$$

What would be α_1^k?

$$\alpha_1^k = P(O_1, S_1 =$$

$$= \underbrace{P(S_1 = k)}_{\text{Obtained from initial probability}} \underbrace{P(O_1 | S_1 = k)}_{\text{Obtained from emission probability}}$$

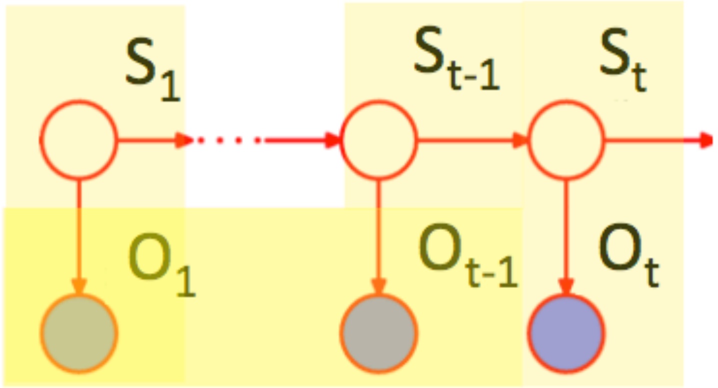


Forward Probability

At step T, the game finishes and we could be in any of the k states, so the probability of observables is sum of the probabilities of being in each of these states

$$p(\{O_t\}_{t=1}^T) = \sum_k p(\{O_t\}_{t=1}^T, S_T = k) = \sum_k \alpha_T^k$$

Compute forward probability α_t^k recursively over t



$$\alpha_t^k \quad := \quad p(O_1, \ldots, O_t, S_t = k)$$

Introduce S_{t-1}

Chain rule

Markov assumption

$$= \quad p(O_t | S_t = k) \sum_i \alpha_{t-1}^i p(S_t = k | S_{t-1} = i)$$

Forward Algorithm

Can compute α_t^k for all k, t using dynamic programming:

- Initialize: $\alpha_1^k = p(O_1 | S_1 = k)\, p(S_1 = k)$ for all k

  This is known from initial probability data

- Iterate: for t = 2, ..., T

  $$\alpha_t^k = p(O_t | S_t = k) \sum_i \alpha_{t-1}^i\, p(S_t = k | S_{t-1} = i) \quad \text{for all k}$$

- Termination: $p(\{O_t\}_{t=1}^T) = \sum_k \alpha_T^k$

  Since the paths are disjoint, you are summing them up

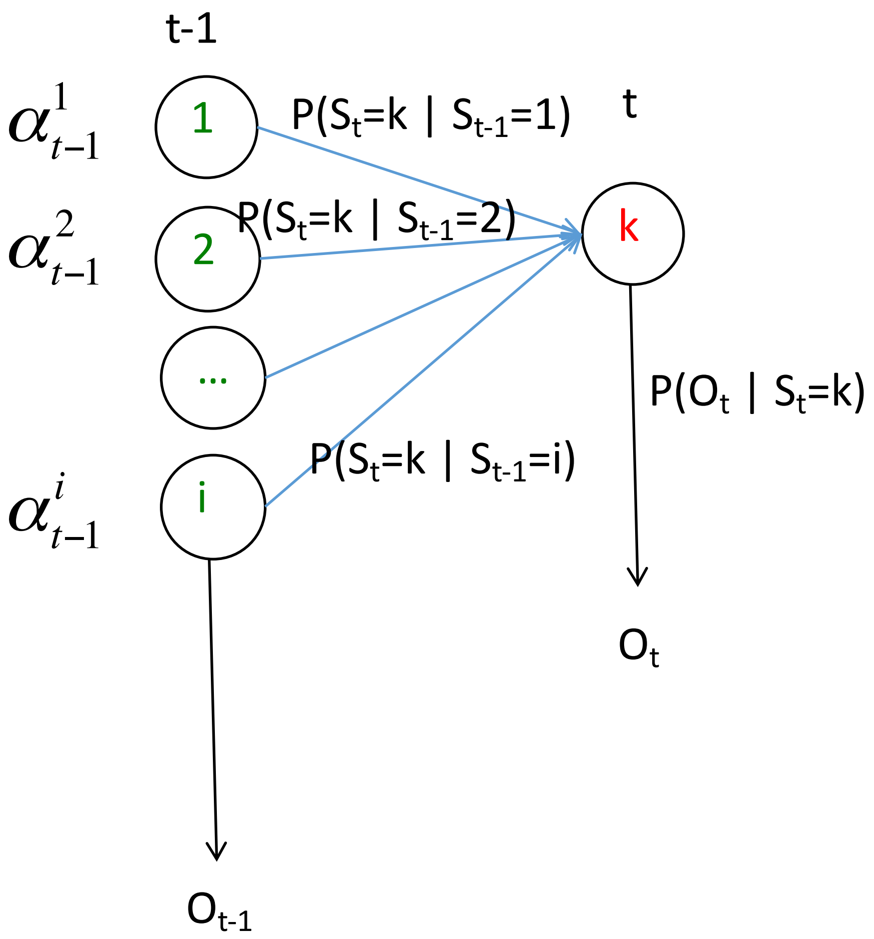
Forward Algorithm

Let us define:

$$\alpha_t^k \quad := \quad p(O_1, \ldots, O_t, S_t = k)$$

At time t, the model is in state k

In this notation, being in state i at time t-1 is α_{t-1}^{i}



Forward Algorithm

- We are just concerned with transition from t-1 to t and then emitting Ot.
- Since they are different paths, you need to sum them up.

$$\alpha_t^k \quad := \quad p(O_1, \ldots, O_t, S_t = k)$$

$$\vdots$$

Introduce S_{t-1}

Chain rule

Markov assumption

$$= \quad p(O_t | S_t = k) \sum \alpha_{t-1}^i p(S_t = k | S_{t-1} = i)$$

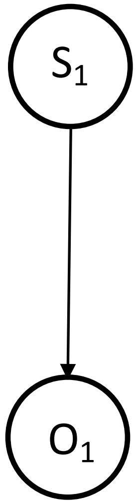
Forward Algorithm

For any state i

$\alpha_1^i = P(O_1|S_1 = i)P(S_1 = i)$

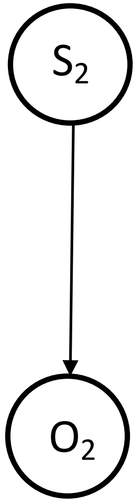
For any state k

$$\alpha_2^k = P(O_2|S_2 = k) \underbrace{\sum_i \alpha_1^i \, P(S_2 = k|S_1 = i)}_{\text{This is a sum over all previous states}}$$



We know how to go from α_1^i to α_2^k

In fact, we know how to go from any α_t^i to α_{t+1}^k



- Forward algorithm helps us go from a previous state to the next state one step at a time.

Example of Forward Algorithm

Transition from\to	S1	S2
S1	.6	.4
S2	.3	.7

(a) Initial Transition Probability Matrix $A_{i,j}$

Output Prob	R	W	B
S1	.3	.4	.3
S2	.4	.3	.3

(b) Output Probabilities

S1	.8
S2	.2

(c) Initial State Probability

Find the probability of being in state S1 at t=2 and observable being R, W

$\alpha_2^1 = P(Q_2 = S1, O_1 = R, O_2 = W)$

First compute

$\alpha_1^1 = P(Q_1 = S1)P(O_1 = R|Q_1 = S1) = 0.8 \times 0.3 = 0.24$
$\alpha_1^2 = P(Q_1 = S2)P(O_1 = R|Q_1 = S2) = 0.2 \times 0.4 = 0.08$

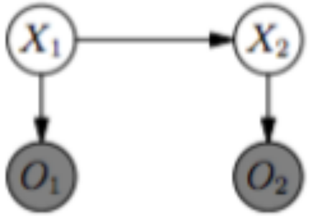
Then, find

$\alpha_2^1 = P(O_2 = W|Q_2 = S1) \sum_i \alpha_1^i \, P(Q_2 = S1|Q_1 = i) = 0.4\times[0.24\times0.6 + 0.08 \times0.3]$ = 0.0672

$\alpha_2^2 = P(O_2 = W|Q_2 = S2) \sum_i \alpha_1^i \, P(Q_2 = S2|Q_1 = i) = 0.3\times[0.24\times0.4 + 0.08 \times0.7]$ = 0.0456

So, the probability of observables i.e. P(R, W) = $\alpha_2^1 + \alpha_2^2 = 0.0672 + 0.0456 = 0.1128$

Example of Forward Algorithm-2



X_1	$\Pr(X_1)$
0	0.3
1	0.7

X_t	X_{t+1}	$\Pr(X_{t+1}\|X_t)$
0	0	0.4
0	1	0.6
1	0	0.8
1	1	0.2

X_t	O_t	$\Pr(O_t\|X_t)$
0	A	0.9
0	B	0.1
1	A	0.5
1	B	0.5

Find the probability: $\Pr(X_2 = 0, O_1 = A, O_2 = B)$

Decoding Problem 1

- In the forward algorithm, you stated with the first step and moved forward.
- In this part, you find the probability of an intermediate state.

e.g. in a game of casino that went for 10 rolls, which dice was used in the third roll.

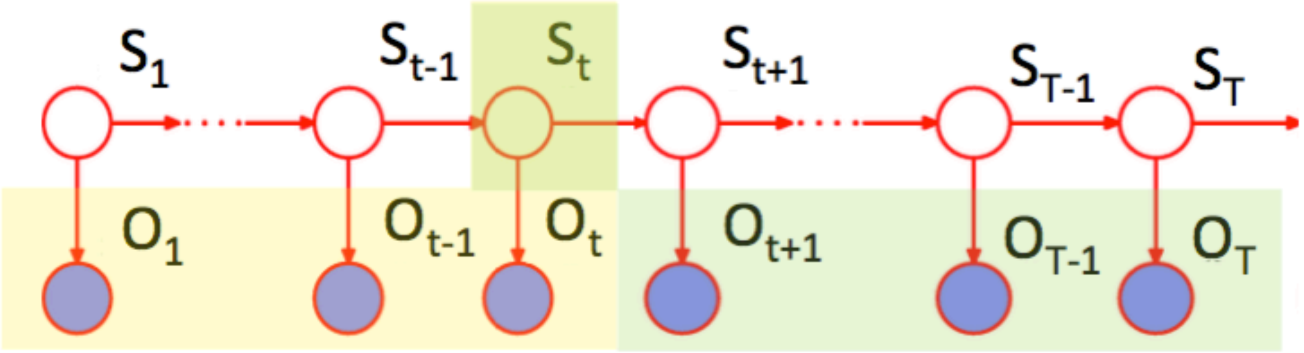
Decoding Problem 1

- Given HMM parameters $p(S_1), p(S_t|S_{t-1}), p(O_t|S_t)$ & observation sequence $\{O_t\}_{t=1}^T$

  find probability that hidden state at time t was k $p(S_t = k|\{O_t\}_{t=1}^T)$

$$
\begin{aligned}
p(S_t = k, \{O_t\}_{t=1}^T) &= p(O_1, \ldots, O_t, S_t = k, O_{t+1}, \ldots, O_T) \\
&= \underbrace{p(O_1, \ldots, O_t, S_t = k)}_{\alpha_t^k}\underbrace{p(O_{t+1}, \ldots, O_T|S_t = k)}_{\beta_t^k}
\end{aligned}
$$

Compute recursively



Decoding Problem 1

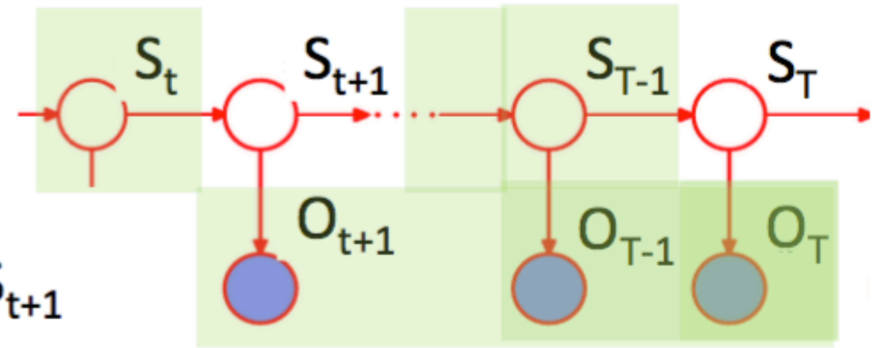
- You already know how to find α_t^k
- Now how to find the other part β_t^k
- As before, you know that you are in state k at time t, you can go to any of the i states at time t+1. The backward of any state at time t+1 would be β_{t+1}^i

Backward Probability

$$p(S_t = k, \{O_t\}_{t=1}^{T}) = p(O_1, \ldots, O_t, S_t = k) p(O_{t+1}, \ldots, O_T | S_t = k) = \alpha_t^k \beta_t^k$$

Compute backward probability β_t^k recursively over t

$$\beta_t^k \quad := \quad p(O_{t+1}, \ldots, O_T | S_t = k)$$



Introduce S_{t+1}

$\vdots$ Chain rule

Markov assumption

$$= \quad \sum_i p(S_{t+1} = i | S_t = k) p(O_{t+1} | S_{t+1} = i) \beta_{t+1}^i$$

Backward Algorithm

Can compute β_t^k for all k, t using dynamic programming:

- Initialize: $\beta_T^k = 1$ for all k

- Iterate: for t = T-1, …, 1

$$\beta_t^k = \sum_i p(S_{t+1} = i | S_t = k) p(O_{t+1} | S_{t+1} = i) \beta_{t+1}^i \quad \text{for all k}$$

- Termination: $p(S_t = k, \{O_t\}_{t=1}^T) = \alpha_t^k \beta_t^k$

$$p(S_t = k | \{O_t\}_{t=1}^T) = \frac{p(S_t = k, \{O_t\}_{t=1}^T)}{p(\{O_t\}_{t=1}^T)} = \frac{\alpha_t^k \beta_t^k}{\sum_i \alpha_t^i \beta_t^i}$$

Backward Algorithm

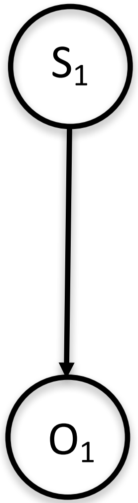
For any state i

$$\beta_{T-1}^{i} = \sum_{k} P(S_T = k \,|S_{T-1} = i)P(O_T|S_T = k)\beta_T^{k}$$

This is a sum over all next states

For any state k

$\beta_T^k = 1$

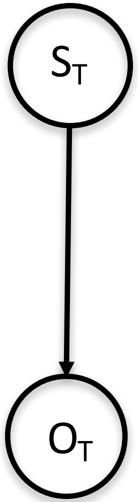


We know how to go from

β_T^k to β_{T-1}^i

In fact, we know how to go from any

β_{t+1}^k to β_t^i



This is the last step.

- Backward algorithm helps us go from a given state to the previous state one step at a time.

Backward Example

Transition from\to	S1	S2
S1	.6	.4
S2	.3	.7

(a) Initial Transition Probability Matrix $A_{i,j}$

Output Prob	R	W	B
S1	.3	.4	.3
S2	.4	.3	.3

(b) Output Probabilities

S1	.8
S2	.2

(c) Initial State Probability

Backward example: Find the backward probability of O = {RWBB}

Start with the last time t = 4, with β of each state being 1

$\beta_3^1 = p(X_4 = 1|X_3 = 1) * p(O_4 = B|X_4 = 1) * 1 + p(X_4 = 2|X_3 = 1) * p(O_4 = B|X_4 = 2) * 1$
$= 0.6 * 0.3 * 1 + 0.4 \ * 0.3 * 1 = 0.3$

$\beta_3^2 = p(X_4 = 1|X_3 = 2) * p(O_4 = B|X_4 = 1) * 1 + p(X_4 = 2|X_3 = 2) * p(O_4 = B|X_4 = 2) * 1$
$= 0.3 * 0.3 * 1 + 0.7 \ * 0.3 * 1 = 0.3$

State	t = 0	t = 1; O = R	t = 2; O = W	t = 3; O = B	t = 4; O = B
S1	0.0078	0.0324	0.09	0.3	1
S2	0.0024	0.0297	0.09	0.3	1

Most likely state vs. Most likely sequence

- Most likely state assignment at time t

$$\arg\max_k p(S_t = k | \{O_t\}_{t=1}^T) = \arg\max_k \alpha_t^k \beta_t^k$$

E.g. Which die was most likely used by the casino in the third roll given the observed sequence?

- Most likely assignment of state sequence

$$\arg \max_{\{S_t\}_{t=1}^T} p(\{S_t\}_{t=1}^T | \{O_t\}_{t=1}^T)$$

E.g. What was the most likely sequence of die rolls used by the casino given the observed sequence?

Not the same solution !

MLA of x?
MLA of (x,y)?

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

Decoding Problem 2

- Given HMM parameters $p(S_1), p(S_t|S_{t-1}), p(O_t|S_t)$ & observation sequence $\{O_t\}_{t=1}^T$

  find most likely assignment of state sequence

$$\arg \max_{\{S_t\}_{t=1}^T} p(\{S_t\}_{t=1}^T | \{O_t\}_{t=1}^T) = \arg \max_{\{S_t\}_{t=1}^T} p(\{S_t\}_{t=1}^T, \{O_t\}_{t=1}^T)$$

$$= \arg\max_k \underbrace{\max_{\{S_t\}_{t=1}^{T-1}} p(S_T = k, \{S_t\}_{t=1}^{T-1}, \{O_t\}_{t=1}^T)}_{V_T^k}$$

Compute recursively

V_T^k - probability of most likely sequence of states ending at state $S_T = k$

Viterbi Decoding

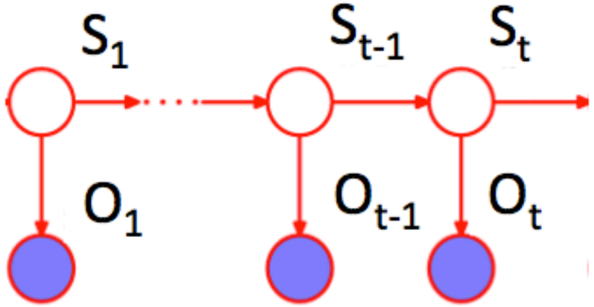
$$\max_{\{S_t\}_{t=1}^T} p(\{S_t\}_{t=1}^T, \{O_t\}_{t=1}^T) = \max_k V_T^k$$

Compute probability V_t^k recursively over t

$$V_t^k \quad := \quad \max_{S_1,\ldots,S_{t-1}} p(S_t = k, S_1, \ldots, S_{t-1}, O_1, \ldots, O_t)$$

$\vdots$ Bayes rule

Markov assumption



$$= \quad p(O_t|S_t = k) \max_i p(S_t = k|S_{t-1} = i) V_{t-1}^i$$

Viterbi Algorithm

Can compute V_t^k for all k, t using dynamic programming:

- Initialize: $V_1^k = p(O_1|S_1=k)p(S_1 = k)$ for all k

- Iterate: for t = 2, ..., T

$$V_t^k = p(O_t|S_t = k) \max_i p(S_t = k|S_{t-1} = i)V_{t-1}^i \quad \text{for all k}$$

- Termination: $\max_{\{S_t\}_{t=1}^T} p(\{S_t\}_{t=1}^T, \{O_t\}_{t=1}^T) = \max_k V_T^k$

Traceback:

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

$$S_{t-1}^* = \arg\max_i p(S_t^*|S_{t-1} = i)V_{t-1}^i$$

Viterbi Example

Given the following HMM:

1. N = 3, states S_1, S_2, S_3
2. M=4 symbols in alphabet {a, c, t, g}
3. **Initial probability distribution vector** π = {.25, .5, .25}

4. Transition probability matrix τ, where τ_{ij}**=probability of transition from state i to state j**

	S1	S2	S3
S1	.25	.5	.25
S2	.25	.25	.5
S3	.5	.5	0

5. Emission probabilities e where $e_i(c)$**=probability that state i emits character c**

	a	c	t	g
S1	1	0	0	0
S2	.25	.5	0	.25
S3	.25	.25	.25	.25

You observe {C C T}. What are the most likely states?

Viterbi Example

	S1	S2	S3
S1	.25	.5	.25
S2	.25	.25	.5
S3	.5	.5	0

	a	c	t	g
S1	1	0	0	0
S2	.25	.5	0	.25
S3	.25	.25	.25	.25

Initial probability distribution vector π = {.25, .5, .25}

	C	C	T
S1	$\frac{1}{4} * 0 = 0$	$0 * \max\{\ldots\} = 0$	$0 * \max\{\ldots\} = 0$
S2	$\frac{1}{2} * \frac{1}{2} = \frac{1}{4}$	$0 * \frac{1}{2}$ $\frac{1}{2} * \max\{ \frac{1}{4} * \frac{1}{4} \} = {}^{1}/_{32}$ ${}^{1}/_{16} * \frac{1}{2}$	$0 * \max\{\ldots\} = 0$
S3	$\frac{1}{4} * \frac{1}{4} = {}^{1}/_{16}$	$0 * \frac{1}{4}$ $\frac{1}{4} * \max\{ \frac{1}{4} * \frac{1}{2} \} = {}^{1}/_{32}$ ${}^{1}/_{16} * 0$	$0 * \frac{1}{4}$ $\frac{1}{4} * \max\{ {}^{1}/_{32} * \frac{1}{2} \} = {}^{1}/_{256}$ ${}^{1}/_{32} * 0$

A traceback of the above array yields this best path sequence of states through the HMM to produce the output string:

S2 S2 S3

Viterbi Example

```
states = ('Healthy', 'Fever')

observations = ('normal', 'cold', 'dizzy')

start_probability = {'Healthy': 0.6, 'Fever': 0.4}

transition_probability = {
   'Healthy' : {'Healthy': 0.7, 'Fever': 0.3},
   'Fever' : {'Healthy': 0.4, 'Fever': 0.6}
   }

emission_probability = {
   'Healthy' : {'normal': 0.5, 'cold': 0.4, 'dizzy': 0.1},
   'Fever' : {'normal': 0.1, 'cold': 0.3, 'dizzy': 0.6}
   }
```

You observe: {Normal, Cold, Dizzy}. What are the likely states?

Viterbi Example

```
states = ('Healthy', 'Fever')

observations = ('normal', 'cold', 'dizzy')

start_probability = {'Healthy': 0.6, 'Fever': 0.4}

transition_probability = {
   'Healthy' : {'Healthy': 0.7, 'Fever': 0.3},
   'Fever' : {'Healthy': 0.4, 'Fever': 0.6}
   }

emission_probability = {
   'Healthy' : {'normal': 0.5, 'cold': 0.4, 'dizzy': 0.1},
   'Fever' : {'normal': 0.1, 'cold': 0.3, 'dizzy': 0.6}
   }
```

You observe: {Normal, Cold, Dizzy}. What are the likely states?

Observable-> State	Normal	Cold	Dizzy
Healthy	0.6*0.5 = .30	0.4 x max{.30x0.7, 0.04x0.4} = 0.084	0.1 x max{.084x0.7, 0.027x0.4} = 0.00588
Fever	0.4*0.1 = 0.04	0.3 x max{.30x0.3, 0.04x0.6} = 0.027	0.6 x max{0.084x0.3, 0.027x0.6} = 0.01512

Most likely states: H H F