Markov Models

- can be used for continuous or discrete sequences
- continuous data can be turned into discrete by vector quantization

Markov models are generative models of sequences!

W_t = weather over time $\in \{r, s\}$ (rain / sun)

1	2	3	4	5	...
r	r	s	s	r	

ex: $P(W_7 = s \mid W_1 \cdots W_6)$ → dependence on previous states

$P(W_t \mid W_{t-1})$ → 1st order Markov

ex: n = sunny r = rainy

π → starting probabilities

	n	r
π =	0.9	0.1

W_{t-1} \ W_t

$$a = \begin{array}{c} \\ n \\ r \end{array} \begin{array}{c} \begin{array}{cc} n & r \end{array} \\ \begin{bmatrix} 0.9 & 0.1 \\ 0.5 & 0.5 \end{bmatrix} \end{array}$$

$S = \{n, r\}$

$$a_{ij} = P(W_t = S_j \mid W_{t-1} = S_i)$$

State S_j

Generative Markov Sequence

1. select $s \in S$ according to starting probability π
2. select S_t from row corresponding to S_{t-1}

π = [n: 0.9, r: 0.1]

a = rows W_{t-1} (n, r), columns W_t (n, r): [[0.9, 0.1], [0.5, 0.5]]

$S = \{n, r\}$

ex:

t	1	2	3
s	n	n	r

*code and simulate!

In reality...

- you calculate the probabilities over all possible transition states

ex:

$P(W_0 = n) = \pi_1$

$P(W_1 = n) = \pi_1 \cdot a_{nn}$

$= 0.9 \cdot 0.9$

$= 0.81$

two possible outcomes each have a probability

$P(W_1 = r) = \pi_1 \cdot a_{nr}$

$= 0.9 \cdot 0.1$

$= 0.09$

$$a = \begin{matrix} & \begin{matrix} n & r \end{matrix} \\ \begin{matrix} n \\ r \end{matrix} & \begin{bmatrix} 0.9 & 0.1 \\ 0.5 & 0.5 \end{bmatrix} \end{matrix} \quad \begin{matrix} W_t \\ W_{t-1} \end{matrix}$$

$$\pi = \begin{matrix} \begin{matrix} n & r \end{matrix} \\ \begin{bmatrix} 0.9 & 0.1 \end{bmatrix} \end{matrix}$$

in the general case... (first order)

$P(W_0) = \pi$ (1×2)

$P(W_1) = P(W_0) \cdot a$ (1×2 · 2×2)

$P(W_t) = P(W_{t-1}) \cdot a = P(W_{t-2}) \cdot a^2 = \dots = \underbrace{P(W_0)}_{\pi} \cdot a^t$

in python: (try it out)

```
pi = np.array([1.0, 0.0])
a = np.array([[0.9, 0.1], [0.5, 0.5]])
P_wt = np.dot(pi, np.linalg.
    matrix_power(a, t))
```

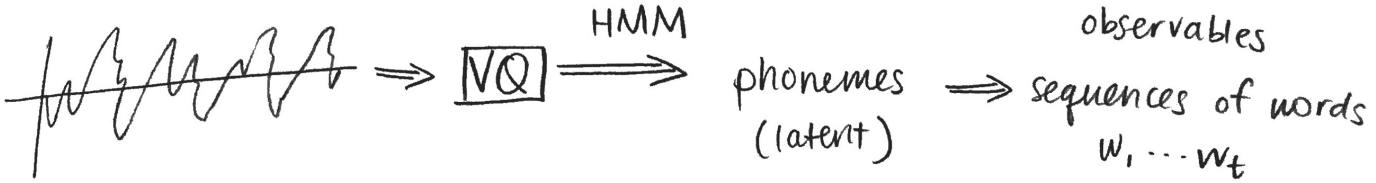
t is the # time points

* markov models converge onto a fixed point depending on the trans kernel a (regardless of π) yay eigenvalues and dynamical systems

Hidden Markov Models

first used in domain of speech recognition (Rabiner)

continuous, acoustic signal



- calculate P(phoneme sequence | acoustic sequence)

ex: P(sequence) = 0.1 × 0.5 × 0.5 × 0.9 × 0.9 ... etc

HMM continued...

def: $\theta = [S, \pi, a]$

$\pi_t = P(X_0 = S_i)$

$a_{ij} = P(X_{t+1} = S_d \mid X_t = S_t$

you can find $P(seq; \theta)$ using NLL function: $L(\theta; seq)$

can also estimate params from data

$$\hat{\pi}_i = \frac{\#\text{ times } S_i \text{ occurs at start of sequence}}{m \text{ sequences}}$$

$$\hat{a}_{ij} = \frac{\#\text{ times ij transition occurs}}{\#\text{ times in state i}}$$

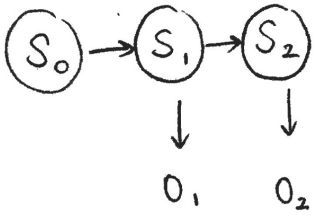
$$\frac{\partial L(\mathcal{D}; \theta)}{\partial \pi} = 0 \quad , \quad \frac{\partial L(\mathcal{D}; \theta)}{a} = 0$$

HMM setup has 2 more parameters:

$$\theta = [\underset{1\cdots n}{S}, \underset{1\cdots K}{O}, \underset{1 \times n}{\pi}, \underset{n \times n}{a}, \underset{n \times K}{b}]$$

- S: n hidden states
- O: K observed states
- π: initial prob state of S
- a: transition kernel
- b: emission probabilities = $P(\text{observe a state} \mid \text{hidden state})$

* S and O also go from $1 \le t \le T$ time steps!!



$$a_{ij} = P(S_{t+1} = S_j \mid S_t = S_i)$$

$$b_{iK} = P(O_t = O_k \mid S_t = S_i)$$

(probably the most important page)

Inference in HMMs and Estimation in HMMs

Inference problems:

1) Filtering: inferring hidden state S from observations O
2) Smoothing: using future observations O to smooth predictions of past hidden states
3) Decoding: find choice of hidden variable assignments that has highest probability given known observations

Filtering $P(S_t \mid O_1 \cdots O_t)$

Dishonest Casino

$S = \{F, L\}$ $\quad O = \{h, t\}$ $\quad \pi = [1, 0]$ $\quad b = \begin{matrix} & h & t \\ F & 0.5 & 0.5 \\ L & 0.1 & 0.9 \end{matrix}$ $(n \times k)$

$a = \begin{matrix} & F & L \\ F & 0.95 & 0.05 \\ L & 0.1 & 0.9 \end{matrix}$ S_{t-1} (rows), S_t (columns)

ex: $P(S_0 = F) = 1$, $P(S_0 = L) = 0$ $\}$ π

find $P(S_1 = F) = \sum_{S_0} P(S_1 = F \mid S_0) P(S_0) \Rightarrow P(S_1 = F \mid O_1 = h) = \dfrac{P(O_1 = h \mid S_1 = F) P(S_1 = F)}{P(O_1 = h)}$

we use forward algorithm to estimate all the probabilities:

Small example:

$P(S_1 = F \mid O_1 = h) = 0.5 \times 0.95 = 0.475$

$P(S_1 = L \mid O_1 = h) = 0.1 \times 0.05 = 0.0005$

normalize!! $\sum_i P_i = 0.48$

$P(S_1 \mid O_1 = h) = [0.475 \quad 0.0005] ./ 0.48$

$= [0.99 \quad 0.01]$

$P(S_1 = F \mid O_1 = t) = 0.5 \times 0.1 = 0.05$

$P(S_1 = L \mid O_1 = t) = 0.9 \times 0.9 = 0.81$ $\quad \sum_i P_i = 0.86$

$P(S_1 \mid O_1 = t) = [0.058 \quad 0.942] \Rightarrow$ these are $\alpha_1(i)$ (defined on next page)

for our example:

$T = 2$ states, $n = 2$ states

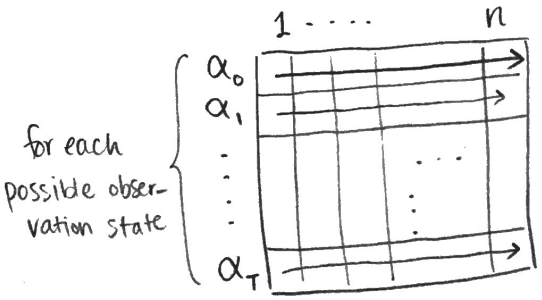
O		F	L
	α_0	1	0
heads	α_1	0.99	0.01
tails	α_2	0.058	0.942

generalization of forward algorithm

Define $\alpha_t(i) = P(S_t = s_i ; O_1 \cdots O_T)$

$$\alpha_{t+1}(j) = b_j(O_{t+1}) \sum_{i=1}^{n \text{ states}} a_{ij} \alpha_t(i) \quad \text{with } \alpha_0(i) = \pi_i$$

n states (hidden) $\quad 1 \le i, j \le n$



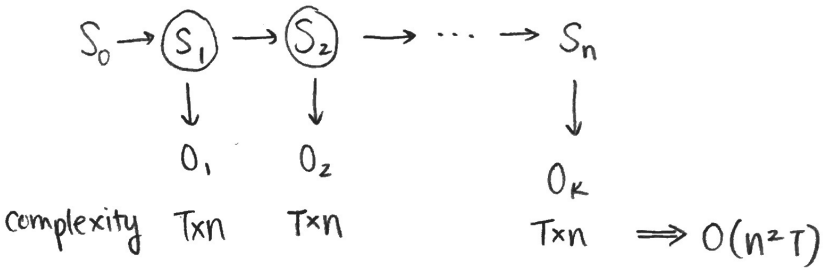
fill one row, then normalize by $\sum_i p_i$

Complexity $O(n^2 T)$

because need to fill $T \times n$ matrix n times!

$0 \le j \le n$

$1 \le t \le T$



Smoothing

$(k < t)$

$$P(S_k | O_1 \cdots O_t) = \frac{P(O_{k+1} \cdots O_t | S_k, O_1 \cdots O_k) P(S_k ; O_1 \cdots O_k)}{P(O_1 \cdots O_t)}$$

$$\propto \underbrace{P(O_{k+1} \cdots O_t | S_k)}_{\text{backward algorithm}} \underbrace{P(S_k ; O_1 \cdots O_k)}_{\alpha_k \text{ (forward algorithm/filtering!!)}}$$

backward algorithm

Define $\beta_k = P(O_{k+1} \cdots O_t | S_k) \quad \beta_T(i) = 1 \quad 1 \le i \le n$

$$\beta(t) = \sum_{j=1}^{n} a_{ij} \beta_{t+1}(j) b_j(O_{t+1})$$

$O(n^2 T)$

Posterior Decoding

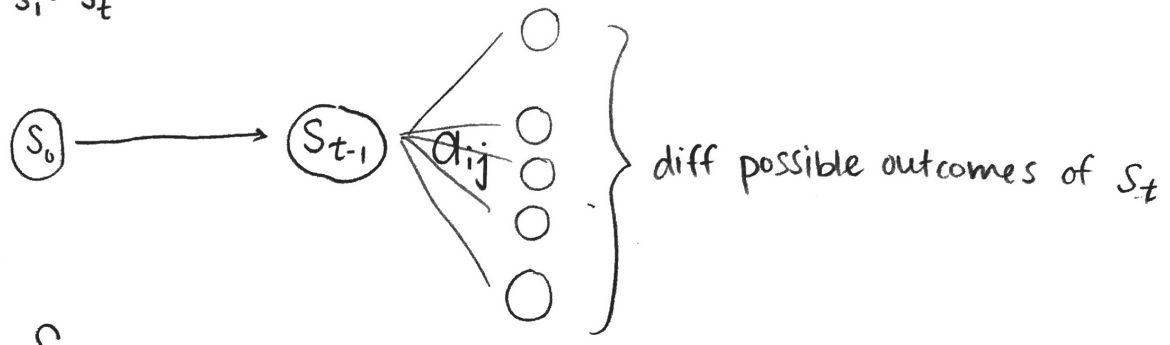
$\alpha_t(i)\beta_t(i) \propto P(S_t = s_i \mid O_i \cdots O_T)$ combine forward/backward

Viterbi Algorithm: find most possible sequence of hidden variable assignments

recall there are 2^T possible sequences for a 2 state sequence

* but there is a special property of most likely sequences

$$\underset{s_1 \cdots s_t}{\text{argmax}} \; P(S_1 \cdots S_t \mid O_1 \cdots O_t)$$

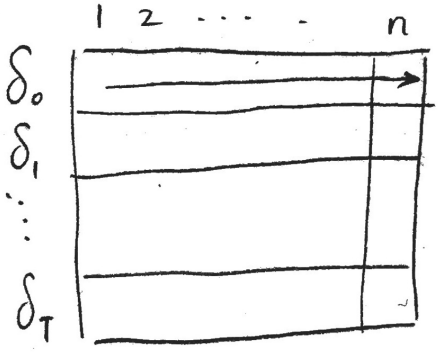


Define $\delta_t(i) = \underset{s_0, \cdots, s_{t-1}}{\text{argmax}} \; P(S_t = s_i, s_0 \cdots s_{t-1}, O_1 \cdots O_t)$

$\delta_0 = \pi_i \qquad 1 \le i \le n$

$O(n^2 T)$

$\delta_{t+1}(j) = b_j(O_{t+1}) \max_i a_{ij} \delta_t(i)$



Estimating HMMs from data

- given observation sequences and corresponding hidden states, find

O S $\theta = [\pi, a, b]$

$$\hat{\pi}_i = \frac{\#\text{ times } S_0 = i}{\#\text{ sequences}} \qquad \hat{a}_{ij} = \frac{\#\text{ times } i \to j \text{ transitions}}{\#\text{ times in state } i}$$

$$\hat{b}_k(i) = \frac{\#\text{ times } i \in O_k \text{ occurs}}{\#\text{ times in state } i}$$

Using EM to guess an algorithm:

1. Guess $\theta = [\pi, a, b]$
2. use sequence $O_1 \cdots O_T$ and θ to infer $\hat{a}_{ij}$, $\hat{b}_k(i)$, and $\hat{\pi}_i$
3. Update θ

(2–3 repeated until no change in θ)

Baum-Welch (EM) for HMMs

1. Guess $\hat{\theta} = [\hat{\pi}, \hat{a}, \hat{b}]$ **E-step**

→ What is probability of being in state i at time t and moving to state j

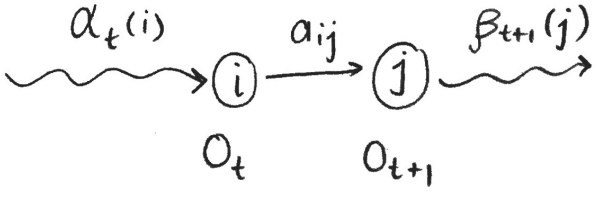
estimate of $\hat{a}_{ij}$ → $\xi_t(i,j) = P(S_t = i, S_{t+1} = j \mid O, \hat{\theta})$

$$= \frac{\alpha_t(i)\, a_{ij}\, b_j(O_{t+1})\, \beta_{t+1}(j)}{\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_t(i)\, a_{ij}\, b_j(O_{t+1})\, \beta_{t+1}(j)}$$

→ $\gamma_t(i)$ = expected times in i

$$= P(S_t = s_i \mid O, \hat{\theta})$$

$$= \sum_{j=1}^{n} \xi_t(i,j)$$



M-Step Given γ_t and ξ_t

re-estimate $\hat{a}_{ij} = \dfrac{\sum_{t}^{T-1} \xi_t(i,j)}{\sum_{t}^{T-1} \gamma_t(i)}$ $\qquad \hat{\pi}_i = \gamma_1(i)$

$$\hat{b}_i(k) = \frac{\sum_{t=1}^{T} \gamma_t(i)_{(o_t = k)}}{\sum_{t=1}^{T} \gamma_t(i)} \quad \frac{(\text{\# times in state } i \text{ and observed symbol } k)}{(\text{\# times in state } i)}$$