# Machine Learning (CS 181): 17. Hidden Markov Models

David C. Parkes and Sasha Rush

Spring 2017



## Contents

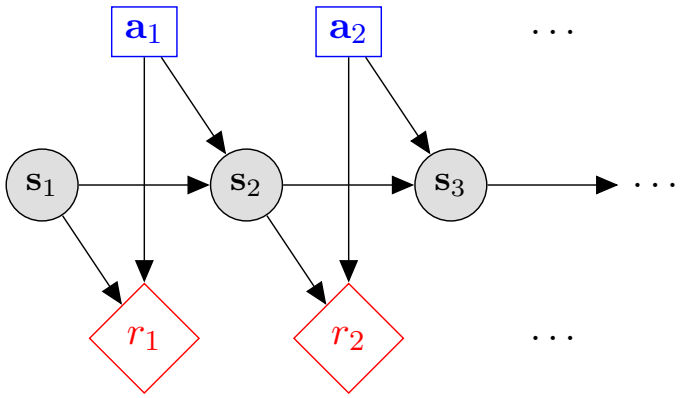
# Contents





# Learning to Act

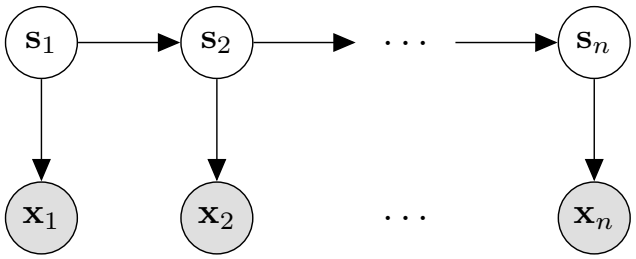
- So far we have been a passive learner. Observing, learning to predict, or finding structure in data.
- For the rest of the course, we'll work towards learning to act; get to Markov decision processes (MDPs) and learning.




Graphical model of MDP: in each state $\mathbf{s}_t$, take action $\mathbf{a}_t$, receiving reward $r_t$ and transitioning to state $\mathbf{s}_{t+1}$. Initial state $\mathbf{s}_1$.

## Today: Hidden Markov Models

A natural model of a sequence of observations in a dynamic system.

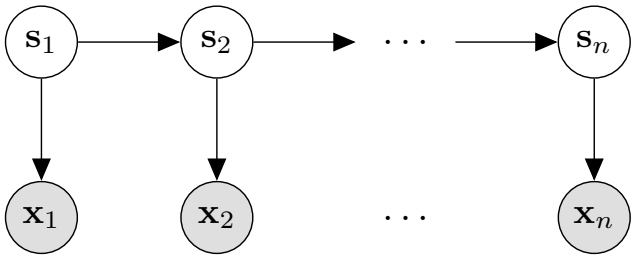



Each node is conditioned on its parents, gray nodes are observed. State $\mathbf{s}_t$ (hidden), observation $\mathbf{x}_t$, next state $\mathbf{s}_{t+1}$.

- A mixture of sequences. Finite set of states. Observations may be continuous or discrete.
- Each latent state generates a distr. on observations. May stay in the same state (i.e., $\mathbf{s}_{t+1} = \mathbf{s}_t$), before transitioning to new state.



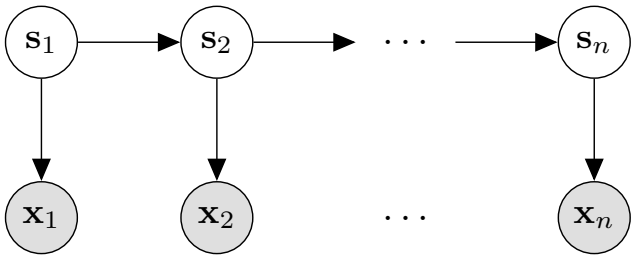
## Application: eCommerce Intention Recognition




- State (discrete): browse, at-work, shopping.
- Observation (discrete): navigation action, social action, add-to-basket, purchase; typically modeled via a state-conditional categorical distribution.

Of interest: what is the probability that the user is shopping, given the sequence of observations?

## Application: Speech Recognition




- State (discrete): phoneme /e/ (elf), /m/ (mum), /n/ (name), /k/ (cat)
- Observation (continuous): frequency (e.g., a 10-dimensional, real vector); typically modeled via a state-conditional mixture of Gaussians

Of interest: what is the most likely sequence of phonemes, given the observations?



## Rabiner's 1989 paper

**A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition**

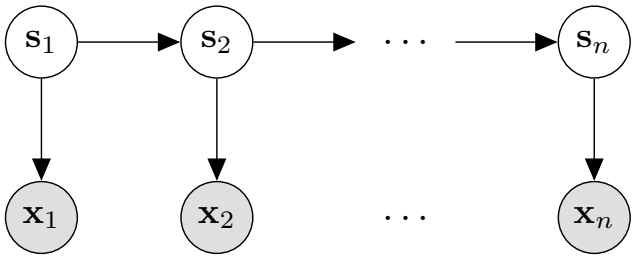
LAWRENCE R. RABINER, FELLOW, IEEE

*Although initially introduced and studied in the late 1960s and early 1970s, statistical methods of Markov source or hidden Markov modeling have become increasingly popular in the last several years. There are two strong reasons why this has occurred. First the models are very rich in mathematical structure and hence can form the theoretical basis for use in a wide range of applications. Second the models, when applied properly, work very well in practice for several important applications. In this paper we attempt to carefully and methodically review the theoretical aspects of this type of statistical modeling and show how they have been applied to selected problems in machine recognition of speech.*

In this case, with a good signal model, we can simulate the source and learn as much as possible via simulations. Finally, the most important reason why signal models are important is that they often work extremely well in practice, and enable us to realize important practical systems—e.g., prediction systems, recognition systems, identification systems, etc., in a very efficient manner.

These are several possible choices for what type of signal model is used for characterizing the properties of a given signal. Broadly one can dichotomize the types of signal

## Application: Automated Vehicles

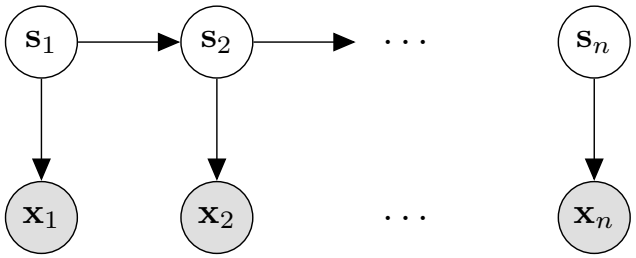



- State (discrete): empty, parked, waiting, turning.
- Observation (discrete, continuous): (position, velocity, size, color, #riders, map)

Of interest: what is the probability that another vehicle is "parked," given the sequence of observations?



## Application: Medical Diagnosis




- State: ill, healthy, hungry, tired.
- Observation (discrete): (sleep, eat, walk, sneeze)

Of interest: what is the most likely diagnosis, given the sequence of observations?

## New emphasis when working with HMMs

- The need for inference about latent variables (e.g., what is the most likely sequence of states, what is the distribution over the next state?)
- Then need for estimation of model parameters even when some variables are latent (i.e., the states in HMMs)
- The role of graphical models in reasoning about probabilistic models, in developing methods for inference.



## Contents

## A Markov Model (or 'Markov chain')


- States: $\mathbf{s}_t \in \{S_k\}_{k=1}^{c}$ (one-hot coded)
- Transition model (captures the Markovian assumption):

$$p(\mathbf{s}_{t+1} = S_\ell \,|\, \mathbf{s}_t = S_k; \mathbf{T}) = T_{kl}.$$

- Initial state distribution:

$$p(\mathbf{s}_1 = S_k; \boldsymbol{\theta}) = \theta_k.$$

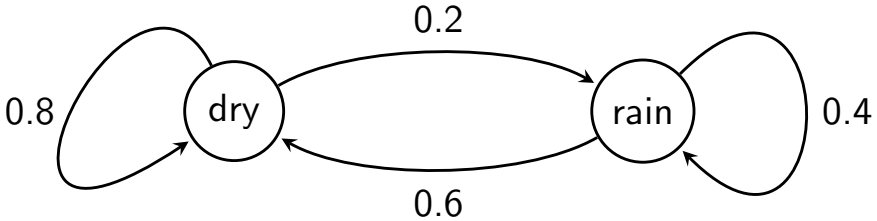
- Parameters transition matrix $\mathbf{T}$ $(c \times c)$, initial state distr. $\boldsymbol{\theta}$ $(c \times 1)$, and number periods $n$ (may be infinite).



## Example: Weather

- Probability 0.8 dry tomorrow if dry today, 0.6 if rains today.
- $s_t = \begin{cases} S_1 & \text{if day } t \text{ is dry} \\ S_2 & \text{if day } t \text{ is rainy} \end{cases}$
- $p(s_{t+1} = S_1 \,|\, s_t = S_1) = 0.8$
- $p(s_{t+1} = S_1 \,|\, s_t = S_2) = 0.6$

$$\text{transition matrix} \quad \mathbf{T} = \begin{pmatrix} p_{11} & p_{12} \\ p_{21} & p_{22} \end{pmatrix} = \begin{pmatrix} 0.8 & 0.2 \\ 0.6 & 0.4 \end{pmatrix}$$



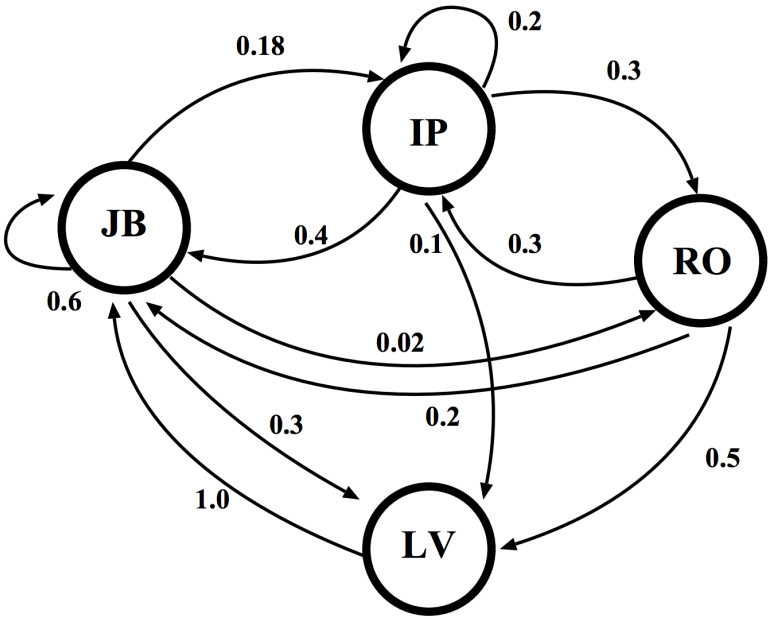

**State Transition Diagram**

## Example: IRS audit

- Often have to **increase the state space** to achieve the Markovian property
- E.g., if the probability of an IRS tax audit depends on the last two audits, then we can introduce four states:
- $\mathcal{S} = \left\{ \begin{array}{ll} S_1 & :: \text{ if last two audits were "no no"} \\ S_2 & :: \text{ if last two audits were "yes no"} \\ S_3 & :: \text{ if last two audits were "no yes"} \\ S_4 & :: \text{ if last two audits were "yes yes"} \end{array} \right\}$



## Example: Markov chain for eCommerce




4 states. JB = just browsing (start state), IP = interesting product, RO = ready to order, LV = leaving (end of session). (de Souze e Silva and Muntz, 11)

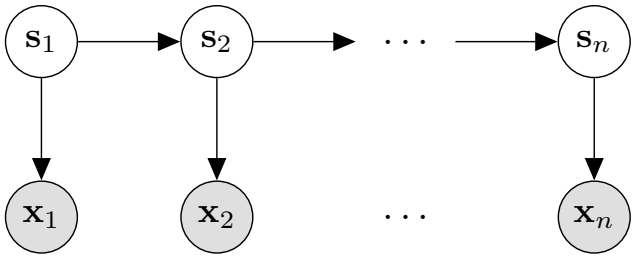
## Modeling exercise: College Student (Part I)

Amanda goes to class at most once (C), works in bursts (W), and is always happy to hang out (H) or eat a meal (M). At some point she goes to bed (B).

Draw a transition diagram that models Amanda's day.



## A Hidden Markov Model




- States $\mathbf{s}_t \in \{S_k\}_{k=1}^c$, transitions $p(\mathbf{s}_{t+1} \,|\, \mathbf{s}_t; \mathbf{T})$, initial state $p(\mathbf{s}_1; \boldsymbol{\theta})$.
- Observations (assumed discrete going forward): $\mathbf{x}_t \in \{O_j\}_{j=1}^m$ (one-hot coded)
- Observation model:

$$p(\mathbf{x}_t = O_j \,|\, \mathbf{s}_t = S_k; \{\boldsymbol{\pi}\}) = \pi_{kj}.$$

- Parameters transition matrix $\mathbf{T}$ $(c \times c)$, initial state distr. $\boldsymbol{\theta}$ $(c \times 1)$, state-conditional observations $\{\boldsymbol{\pi}_k\}_{k=1}^c$ $(m \times 1)$, and num. periods $n$.

## Modeling Exercise: College Student (Part II)

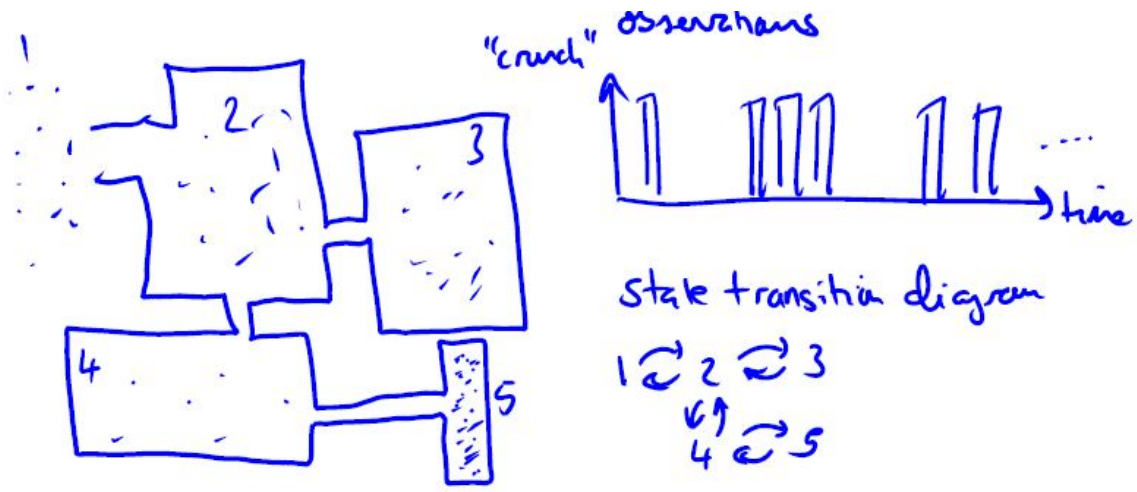
Amanda goes to class at most once (C), works in bursts (W), and is always happy to hang out (H) or eat a meal (M). At some point she goes to bed (B). Suppose we observe whether she's sleeping (S), eating (E), or awake (not eating) (A). Augment your Markov model, creating an HMM.



## The Kinds of Questions we'll be interested in

- 'filtering': track the hidden state $p(\mathbf{s}_t \mid \mathbf{x}_1, \ldots, \mathbf{x}_t)$
- 'smoothing': compute $p(\mathbf{s}_t \mid \mathbf{x}_1, \ldots, \mathbf{x}_n)$
- Compute prob. of a seq. of observations $p(\mathbf{x}_1, \ldots, \mathbf{x}_n)$
  - 'classification': Imagine having multiple models (e.g., phoneme model for word 'Two', phoneme model for word 'One', etc.) Can use $p(\mathbf{x}_1, \ldots, \mathbf{x}_n; \text{one})$, $p(\mathbf{x}_1, \ldots, \mathbf{x}_n; \text{two})$, etc. and Bayes rule to compute the most likely word.
- 'explanation': compute the most likely 'explanation' for a sequence of observations $\arg\max_{\mathbf{s}_1,\ldots,\mathbf{s}_n} p(\mathbf{s}_1, \ldots, \mathbf{s}_n \mid \mathbf{x}_1, \ldots, \mathbf{x}_n)$
- 'prediction': given $\mathbf{x}_1, \ldots, \mathbf{x}_n$, compute $P(\mathbf{x}_{n+1} \mid \mathbf{x}_1, \ldots, \mathbf{x}_n)$.

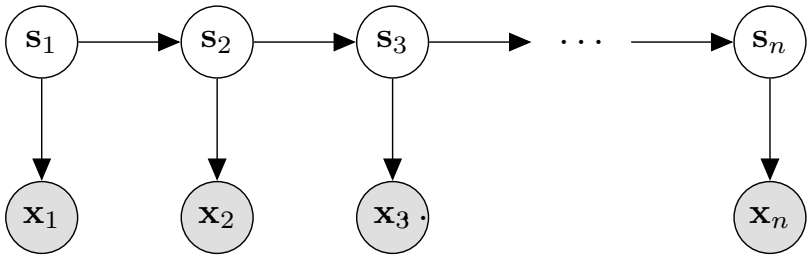
## Example: Tracking A Mouse






## Contents

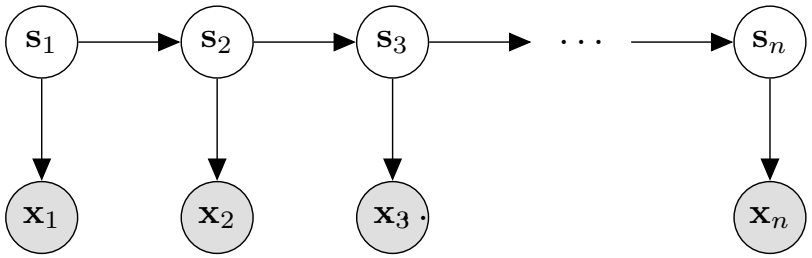
## Working with HMMs (1 of 2)




- Future independent of past given the present:
  $p(\mathbf{s}_{t+1} \,|\, \mathbf{s}_1, \ldots \mathbf{s}_t, \mathbf{x}_1, \ldots, \mathbf{x}_t) = p(\mathbf{s}_{t+1} \,|\, \mathbf{s}_t)$
  - What about $p(\mathbf{x}_{t+1} \,|\, \mathbf{x}_1, \ldots, \mathbf{x}_t)$?
  - Is all history needed to make decisions?
- Observations only depend on the present:
  $p(\mathbf{x}_t \,|\, \mathbf{s}_1, \ldots, \mathbf{s}_t, \mathbf{x}_1, \ldots, \mathbf{x}_{t-1}) = p(\mathbf{x}_t \,|\, \mathbf{s}_t)$



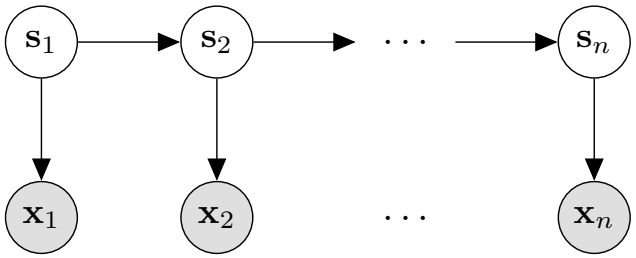
## Working with HMMs (2 of 2)




Distribution defined by HMM is

$$p(\mathbf{s}_1, \ldots, \mathbf{s}_n, \mathbf{x}_1, \ldots, \mathbf{x}_n) = p(\mathbf{s}_1, \ldots, \mathbf{s}_n) p(\mathbf{x}_1, \ldots, \mathbf{x}_n \,|\, \mathbf{s}_1, \ldots, \mathbf{s}_n)$$
$$= p(\mathbf{s}_1) \prod_{t=1}^{n-1} p(\mathbf{s}_{t+1} \,|\, \mathbf{s}_t) \prod_{t=1}^{n} p(\mathbf{x}_t \,|\, \mathbf{s}_t)$$

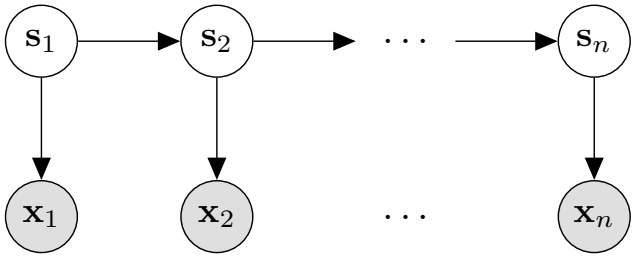
## The Training Problem in HMMs




- Data is collection of sequences $\{\mathbf{x}^i\}_{i=1}^N$, where $\mathbf{x}^i = (\mathbf{x}_1^i, \ldots, \mathbf{x}_n^i)$
  - note: a sequence of observations is a row vector!
  - assume all sequences of same length $n$
- Goal is to learn parameters $\{\mathbf{T}, \boldsymbol{\theta}, \{\boldsymbol{\pi}_k\}\}$.
- Via maximum likelihood?



## Maximum Likelihood Estimation




- Observed data (given): sequences $\mathbf{x}^i$. State sequences (latent): $\mathbf{s}^i$. Parameters $\mathbf{w} = \{\mathbf{T}, \boldsymbol{\theta}, \{\boldsymbol{\pi}\}_k\}$
- Consider one sequence, $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$. Log likelihood:

$$\ln(p(\mathbf{x}; \mathbf{w})) = \ln \sum_{\mathbf{s}=(\mathbf{s}_1,\ldots,\mathbf{s}_n)} p(\mathbf{s}, \mathbf{x}; \mathbf{w}) = \ln \sum_{\mathbf{s}=(\mathbf{s}_1,\ldots,\mathbf{s}_n)} p(\mathbf{s}; \mathbf{w}) p(\mathbf{x} \,|\, \mathbf{s}; \mathbf{w})$$

- The summation inside the log means that the terms do not nicely decompose by parameters.

## Aside: Complete information version (1 of 2)

- Suppose we had the states. Can write out the probability for a single sequence:

$$\begin{aligned} p(\mathbf{x}, \mathbf{s}; \mathbf{w}) &= p(\mathbf{s}; \mathbf{w}) p(\mathbf{x} \,|\, \mathbf{s}; \mathbf{w}) \\ &= p(\mathbf{s}_1; \boldsymbol{\theta}) \prod_{t=1}^{n-1} p(\mathbf{s}_{t+1} \,|\, \mathbf{s}_t; \mathbf{T}) \prod_{t=1}^{n} p(\mathbf{x}_t \,|\, \mathbf{s}_t; \{\boldsymbol{\pi}\}) \end{aligned}$$

- Via one-hot coding of $\mathbf{x}_t$ and $\mathbf{s}_t$, can show

$$\ln(p(\mathbf{x}, \mathbf{s}; \mathbf{w})) = \sum_{k=1}^{c} s_{1k} \ln \theta_k + \sum_{t=1}^{n-1} \sum_{k=1}^{c} \sum_{\ell=1}^{c} s_{tk} s_{t+1,\ell} t_{k\ell} + \sum_{t=1}^{n} \sum_{k=1}^{c} \sum_{j=1}^{m} s_{tk} x_{tj} \pi_{kj}$$

- remember: $c$ states, $n$ time periods, $m$ possible outputs



## Aside: Complete information version (2 of 2)

- MLE for complete-data log likelihood has an analytical solution:

$$\hat{\theta}_k = \frac{N_{1k}}{N}, \quad \hat{t}_{k\ell} = \frac{N_{kl}}{N_{-n,k}}, \quad \hat{\pi}_{kj} = \frac{N_{kj}}{N_k}$$

where

$$N_{1k} = \sum_{i=1}^{N} s_{1k}^i; \quad N_{-n,k} = \sum_{i=1}^{N} \sum_{t=1}^{n-1} s_{tk}^i; \quad N_k = \sum_{i=1}^{N} \sum_{t=1}^{n} s_{tk}^i.$$

$\{$various counts of state $S_k\}$

$$N_{k\ell} = \sum_{i=1}^{N} \sum_{t=1}^{n-1} s_{tk}^i s_{t+1,\ell}^i \quad \{\text{num. transitions from } S_k \text{ to state } S_l\}$$

$$N_{kj} = \sum_{i=1}^{N} \sum_{t=1}^{n} s_{tk}^i x_{tj}^i \quad \{\text{num. observations } O_j \text{ in state } S_k\}$$

## How to Proceed? EM!

Initialize parameters, then repeat two steps:

- (E step): for each sequence $\mathbf{x}^i$, estimate the state distribution conditioned on $\mathbf{x}^i$, for each state $\mathbf{s}_1^i, \ldots, \mathbf{s}_n^i$.
- (M step): update parameters $\{\boldsymbol{\theta}, \mathbf{T}, \{\boldsymbol{\pi}\}\}$, to maximize the expected complete-data log likelihood.

For sequence $\mathbf{x}$, the expected complete-data log likelihood is

$$\mathbf{E}_{\mathbf{S}}[\ln(p(\mathbf{x}, \mathbf{S}; \mathbf{w}))],$$

where $\mathbf{S}$ is the set of random variables corresponding to states $\mathbf{s}_1, \ldots, \mathbf{s}_n$.



## Instantiating EM for HMMs: The E-Step

For each sequence $\mathbf{x}^i$, the E-step will use the current parameters $\mathbf{w}$ to estimate the following ( this is inference! see this soon):

- the probability that the state is $S_k$:

$$\forall t, k: \quad q_{tk}^i \triangleq p(\mathbf{s}_t^i = S_k \,|\, \mathbf{x}^i; \mathbf{w})$$

- the probability a transition from state $S_k$ to $S_\ell$ in period $t$:

$$\forall t, k, \ell: \quad q_{t,t+1,k,\ell}^i \triangleq p(\mathbf{s}_t^i = S_k, \mathbf{s}_{t+1}^i = S_\ell \,|\, \mathbf{x}^i; \mathbf{w})$$

## Instantiating EM for HMMs: The M-Step

Given the $\mathbf{q}_t^i$ vectors $(c \times 1)$ and the $\mathbf{Q}_{t,t+1}^i$ matrices $(c \times c)$, the M-step computes estimated counts, adopting the estimated state distributions in place of observed states. Estimated counts:

$$\hat{N}_{1k} = \sum_{i=1}^{N} q_{1k}^i; \quad \hat{N}_{-n,k} = \sum_{i=1}^{N} \sum_{t=1}^{n-1} q_{tk}^i; \quad \hat{N}_k = \sum_{i=1}^{N} \sum_{t=1}^{n} q_{tk}^i$$

$$\hat{N}_{k\ell} = \sum_{i=1}^{N} \sum_{t=1}^{n-1} q_{t,t+1,k,\ell}^i; \quad \hat{N}_{kj} = \sum_{i=1}^{N} \sum_{t=1}^{n} q_{tk}^i x_{tj}^i$$

Given this, the parameter updates are:

$$\hat{\theta}_k = \frac{\hat{N}_{1k}}{N}, \quad \hat{t}_{k\ell} = \frac{\hat{N}_{kl}}{\hat{N}_{-n,k}}, \quad \hat{\pi}_{kj} = \frac{\hat{N}_{kj}}{\hat{N}_k}$$



## Discussion

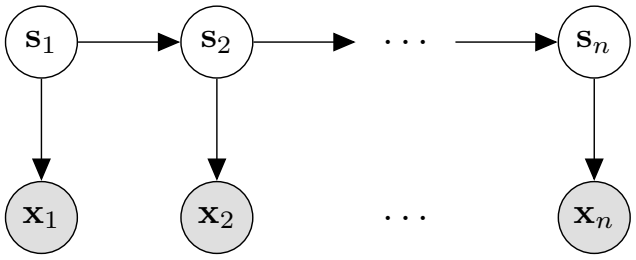
- EM for HMMs when used with a particular 'forward-backward' inference algorithm (see this soon!) is the Baum-Welch algorithm
- With too few hidden states, HMM may not be able to model the domain with enough fidelity. With too many hidden states, HMM may overfit.
- Validation can be used to determine the number of hidden states—for each $c$, train and evaluate its likelihood on the validation set.

# Contents





# Inference about Latent States

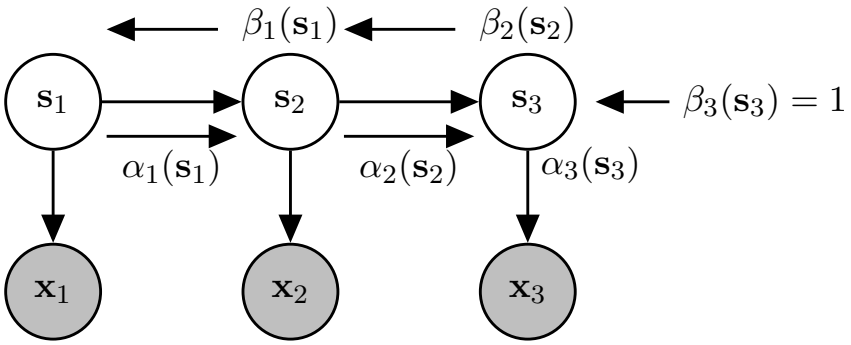



Questions we're interested in:

- $p(\mathbf{x}_1, \ldots, \mathbf{x}_n) = \sum_{\mathbf{s}_1, \ldots, \mathbf{s}_n} p(\mathbf{x}_1, \ldots, \mathbf{x}_n, \mathbf{s}_1, \ldots, \mathbf{s}_n)$
- smoothing: $p(\mathbf{s}_t \,|\, \mathbf{x}_1, \ldots, \mathbf{x}_n)$
- prediction $p(\mathbf{x}_{n+1} \,|\, \mathbf{x}_1, \ldots, \mathbf{x}_n)$
- transitions $p(\mathbf{s}_t, s_{t+1} \,|\, \mathbf{x}_1, \ldots, \mathbf{x}_n)$

All about 'marginalization.' Need to find an efficient way to do this.

# Forward-backward algorithm (1 of 5)

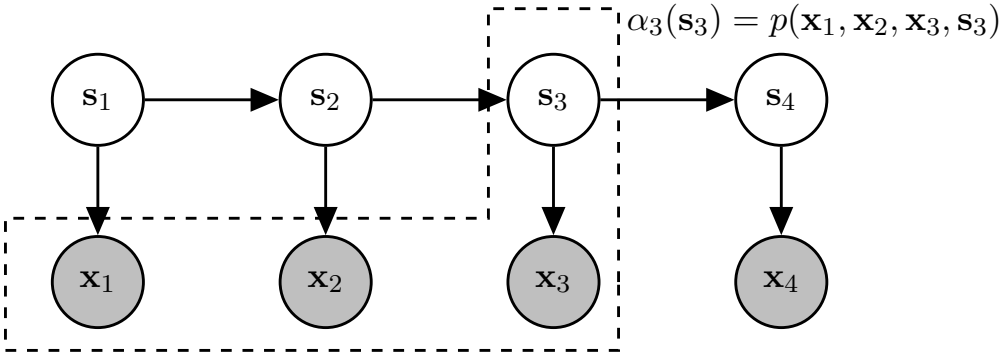



- Compute "$\alpha$" values (forward pass)
- Compute "$\beta$" values (backward pass)
- We will then be able to use $\alpha$s and $\beta$s to answer all the inference tasks we care about!

Can think about this as a 'message passing' algorithm, which makes it easy to distribute.

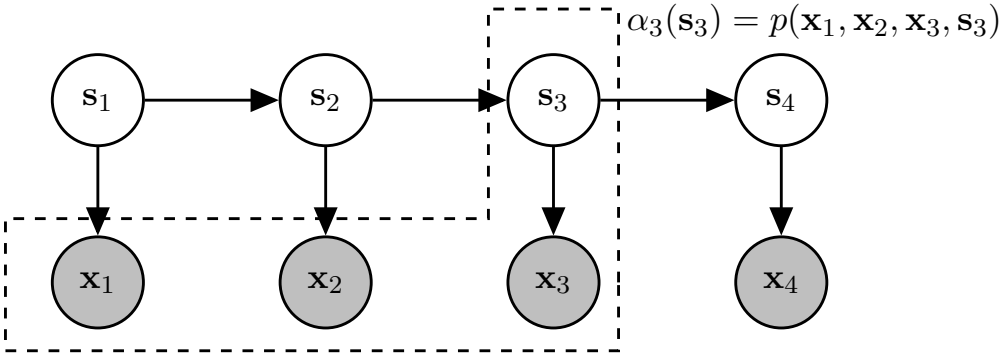


# Forward-backward algorithm: alpha values (2 of 5)




$$\begin{aligned}
\forall \mathbf{s}_3: \quad \alpha_3(\mathbf{s}_3) &\triangleq p(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{s}_3) = \sum_{\mathbf{s}_2} p(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{s}_2, \mathbf{s}_3) \\
&= \sum_{\mathbf{s}_2} p(\mathbf{x}_3 \,|\, \mathbf{x}_1, \mathbf{x}_2, \mathbf{s}_2, \mathbf{s}_3) p(\mathbf{x}_1, \mathbf{x}_2, \mathbf{s}_2, \mathbf{s}_3) \\
&= p(\mathbf{x}_3 \,|\, \mathbf{s}_3) \sum_{\mathbf{s}_2} p(\mathbf{s}_3 \,|\, \mathbf{x}_1, \mathbf{x}_2, \mathbf{s}_2) p(\mathbf{x}_1, \mathbf{x}_2, \mathbf{s}_2) \\
&= p(\mathbf{x}_3 \,|\, \mathbf{s}_3) \sum_{\mathbf{s}_2} p(\mathbf{s}_3 \,|\, \mathbf{s}_2) \alpha_2(\mathbf{s}_2)
\end{aligned}$$

## Forward-backward algorithm: alpha values (3 of 5)




We found $\quad \forall \mathbf{s}_3 : \quad \alpha_3(\mathbf{s}_3) = p(\mathbf{x}_3 \,|\, \mathbf{s}_3) \sum_{\mathbf{s}_2} p(\mathbf{s}_3 \,|\, \mathbf{s}_2) \alpha_2(\mathbf{s}_2)$

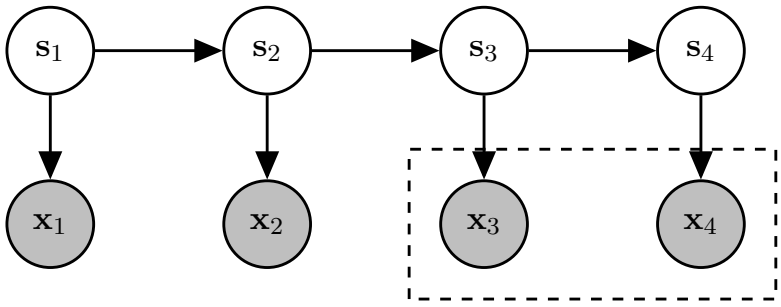
Generalizing, we have

$$\forall \mathbf{s}_t : \quad \alpha_t(\mathbf{s}_t) = \begin{cases} p(\mathbf{x}_t \,|\, \mathbf{s}_t) \sum_{\mathbf{s}_{t-1}} p(\mathbf{s}_t \,|\, \mathbf{s}_{t-1}) \alpha_{t-1}(\mathbf{s}_{t-1}) & \text{if } 1 < t \leq n \\ p(\mathbf{x}_1 \,|\, \mathbf{s}_1) p(\mathbf{s}_1) & \text{o.w.} \end{cases}$$

Computation $O(c)$ for each state $\mathbf{s}_t$, and thus $O(c^2)$ per time period. Total is $O(c^2 n)$. Big win!



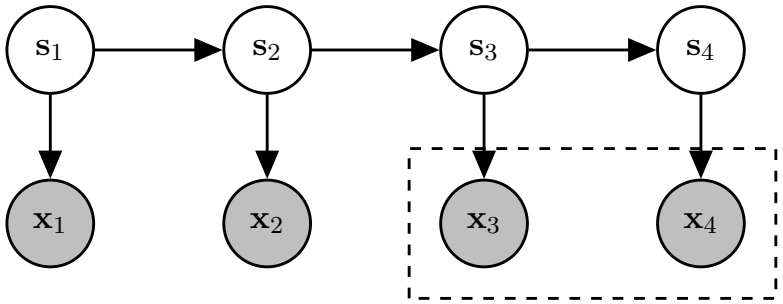
## Forward-backward algorithm: beta values (4 of 5)




$\beta_2(\mathbf{s}_2) = p(\mathbf{x}_3, \mathbf{x}_4 | \mathbf{s}_2)$

$$\begin{aligned} \forall \mathbf{s}_2 : \quad \beta_2(\mathbf{s}_2) \triangleq p(\mathbf{x}_3, \mathbf{x}_4 \,|\, \mathbf{s}_2) &= \sum_{s_3} p(\mathbf{x}_3, \mathbf{x}_4, \mathbf{s}_3 \,|\, \mathbf{s}_2) \\ &= \sum_{s_3} p(\mathbf{s}_3 \,|\, \mathbf{s}_2) p(\mathbf{x}_3, \mathbf{x}_4 \,|\, \mathbf{s}_2, \mathbf{s}_3) \\ &= \sum_{s_3} p(\mathbf{s}_3 \,|\, \mathbf{s}_2) p(\mathbf{x}_3 \,|\, \mathbf{s}_3) p(\mathbf{x}_4 \,|\, \mathbf{s}_3) \\ &= \sum_{s_3} p(\mathbf{s}_3 \,|\, \mathbf{s}_2) p(\mathbf{x}_3 \,|\, \mathbf{s}_3) \beta_3(\mathbf{s}_3) \end{aligned}$$

## Forward-backward algorithm (5 of 5)




$$\beta_2(\mathbf{s}_2) = p(\mathbf{x}_3, \mathbf{x}_4 | \mathbf{s}_2)$$

We found $\quad \forall \mathbf{s}_2 : \quad \beta_2(\mathbf{s}_2) = \sum_{s_3} p(\mathbf{s}_3 \,|\, \mathbf{s}_2) p(\mathbf{x}_3 \,|\, \mathbf{s}_3) \beta_3(\mathbf{s}_3)$

Generalizing, we have

$$\forall \mathbf{s}_t : \quad \beta_t(\mathbf{s}_t) = \begin{cases} \sum_{\mathbf{s}_{t+1}} p(\mathbf{s}_{t+1} \,|\, \mathbf{s}_t) p(\mathbf{x}_{t+1} \,|\, \mathbf{s}_{t+1}) \beta_{t+1}(\mathbf{s}_{t+1}) & \text{if } 1 \leq t < n \\ 1 & \text{o.w.} \end{cases}$$

Computation $O(c)$ for each state $\mathbf{s}_t$, and thus $O(c^2)$ per time period. Total is $O(c^2 n)$.



## Working with $\alpha$- and $\beta$-values

We have:

$$\begin{aligned} \forall \mathbf{s}_t : \quad \alpha_t(\mathbf{s}_t)\beta_t(\mathbf{s}_t) &= p(\mathbf{x}_1, \ldots, \mathbf{x}_t, \mathbf{s}_t) p(\mathbf{x}_{t+1}, \ldots, \mathbf{x}_n \,|\, \mathbf{s}_t) \\ &= p(\mathbf{s}_t) p(\mathbf{x}_1, \ldots, \mathbf{x}_t \,|\, \mathbf{s}_t) p(\mathbf{x}_{t+1}, \ldots, \mathbf{x}_n \,|\, \mathbf{s}_t) \\ &= p(\mathbf{s}_t) p(\mathbf{x}_1, \ldots, \mathbf{x}_n \,|\, \mathbf{s}_t) \qquad (1) \\ &= p(\mathbf{x}_1, \ldots, \mathbf{x}_n, s_t) \end{aligned}$$

where (1) follows by the independence of $\mathbf{x}_1, \ldots, \mathbf{x}_t$ and $\mathbf{x}_{t+1}, \ldots, \mathbf{x}_n$, conditioned on $\mathbf{s}_t$. (Looking to lecture 19, this is by 'd-separation'; e.g., the path $\mathbf{x}_1$ to $\mathbf{x}_n$ is blocked by $\mathbf{s}_t$.)

## Using $\alpha$- and $\beta$- values for Inference

Based on

$$\forall \mathbf{s}_t: \quad \alpha_t(\mathbf{s}_t)\beta_t(\mathbf{s}_t) = p(\mathbf{s}_t)p(\mathbf{x}_1, \ldots, \mathbf{x}_n \,|\, \mathbf{s}_t) = p(\mathbf{x}_1, \ldots, \mathbf{x}_n, s_t),$$

we can compute:

- $p(\mathbf{x}_1, \ldots, \mathbf{x}_n) = \sum_{\mathbf{s}_t} \alpha_t(\mathbf{s}_t)\beta_t(\mathbf{s}_t)$ (for any $t$)
- 'smoothing': $p(\mathbf{s}_t \,|\, \mathbf{x}_1, \ldots, \mathbf{x}_n) \propto p(\mathbf{x}_1, \ldots, \mathbf{x}_n, \mathbf{s}_t) = \alpha_t(\mathbf{s}_t)\beta_t(\mathbf{s}_t)$

We can also use $\alpha$- and $\beta$- values to compute (left as exercise!):

- 'prediction'

$p(\mathbf{x}_{n+1} \,|\, \mathbf{x}_1, \ldots, \mathbf{x}_n) \propto \sum_{\mathbf{s}_n, \mathbf{s}_{n+1}} \alpha_n(\mathbf{s}_n)p(\mathbf{s}_{n+1} \,|\, \mathbf{s}_n)p(\mathbf{x}_{n+1} \,|\, \mathbf{s}_{n+1})$

- 'transition'

$p(\mathbf{s}_t, \mathbf{s}_{t+1} \,|\, \mathbf{x}_1, \ldots, \mathbf{x}_n) \propto \alpha_t(\mathbf{s}_t)p(\mathbf{s}_{t+1} \,|\, \mathbf{s}_t)p(\mathbf{x}_{t+1} \,|\, \mathbf{s}_{t+1})\beta_{t+1}(\mathbf{s}_{t+1})$



## EM for HMMs (Baum-Welch Algorithm)

To run EM for estimating the parameters of a HMM, we repeat:

- Given current parameters, run forward-backward algorithm for each sequence $\mathbf{x}^i$ to estimate $\alpha$- and $\beta$- values
- Use these $\alpha$- and $\beta$- values to compute, for each sequence $\mathbf{x}^i$,

$$\forall k, t: \quad q^i_{tk} \triangleq p(\mathbf{s}^i_t = S_k \,|\, \mathbf{x}^i; \mathbf{w}) \qquad \{\text{smoothing}\}$$
$$\forall t, k, \ell: \quad q^i_{t,t+1,k,\ell} \triangleq p(\mathbf{s}^i_t = S_k, \mathbf{s}^i_{t+1} = S_\ell \,|\, \mathbf{x}^i; \mathbf{w}) \qquad \{\text{transition}\}$$

- Use these $\mathbf{q}^i_t$ vectors and $\mathbf{Q}^i_{t,t+1}$ matrices to compute the estimated counts $\hat{N}_{1k}, \hat{N}_{-n,k}, \hat{N}_k, \hat{N}_{k\ell}$, and $\hat{N}_{kj}$.
- Solve the M-step to update parameters.

## Prediction of most likely state sequence (1 of 2)

We've been solving $\sum_{\mathbf{s}_1,\ldots,\mathbf{s}_n} p(\mathbf{s}_1,\ldots,\mathbf{s}_n,\mathbf{x}_1,\ldots,\mathbf{x}_n)$.
For this, we've used:

$$\alpha_t(\mathbf{s}_t) \triangleq \sum_{\mathbf{s}_1,\ldots,\mathbf{s}_{t-1}} p(\mathbf{s}_1,\ldots,\mathbf{s}_t,\mathbf{x}_1,\ldots,\mathbf{x}_t) = p(\mathbf{s}_t,\mathbf{x}_1,\ldots,\mathbf{x}_t)$$

What if we instead want to solve:

$$\arg\max_{\mathbf{s}_1,\ldots,\mathbf{s}_n} p(\mathbf{s}\,|\,\mathbf{x}) = \arg\max_{\mathbf{s}_1,\ldots,\mathbf{s}_n} p(\mathbf{s}_1,\ldots,\mathbf{s}_n,\mathbf{x}_1,\ldots,\mathbf{x}_n)$$

This is the decoding (or explanation) problem. For this, we can use:

$$v_t(\mathbf{s}_t) \triangleq \max_{\mathbf{s}_1,\ldots,\mathbf{s}_{t-1}} p(\mathbf{s}_1,\ldots,\mathbf{s}_t,\mathbf{x}_1,\ldots,\mathbf{x}_t)$$

This is the likelihood of $\mathbf{x}_1,\ldots,\mathbf{x}_t$, if the current state is $\mathbf{s}_t$, and under the best explanation so far.



## Prediction of most likely state sequence (2 of 2)

Recurrence for marginalization:

$$\forall \mathbf{s}_t: \quad \alpha_t(\mathbf{s}_t) = \begin{cases} p(\mathbf{x}_t\,|\,\mathbf{s}_t)\sum_{\mathbf{s}_{t-1}} p(\mathbf{s}_t\,|\,\mathbf{s}_{t-1})\alpha_{t-1}(\mathbf{s}_{t-1}) & \text{if } 1 < t \leq n \\ p(\mathbf{x}_1\,|\,\mathbf{s}_1)p(\mathbf{s}_1) & \text{o.w.} \end{cases}$$

Analogously, the recurrence for decoding can be shown to be:

$$\forall \mathbf{s}_t: \quad v_t(\mathbf{s}_t) = \begin{cases} p(\mathbf{x}_t\,|\,\mathbf{s}_t)\max_{\mathbf{s}_{t-1}} p(\mathbf{s}_t\,|\,\mathbf{s}_{t-1})v_{t-1}(\mathbf{s}_{t-1}) & \text{if } 1 < t \leq n \\ p(\mathbf{x}_1\,|\,\mathbf{s}_1)p(\mathbf{s}_1) & \text{o.w.} \end{cases}$$

To be able to find the optimal sequence, we also store, for each $\mathbf{s}_t$, the best choice of $\mathbf{s}_{t-1}$:

$$\mathbf{z}_t(\mathbf{s}_t) = \arg\max_{\mathbf{s}_{t-1}} p(\mathbf{s}_t\,|\,\mathbf{s}_{t-1})v_{t-1}(\mathbf{s}_{t-1})$$

This is the forward Viterbi algorithm.

# Contents





# Conclusion

- HMMs are a powerful modeling framework to reason about sequential data
- Based on the observation of one or more sequences, we can consider a particular number of hidden states, and estimate the parameters of a HMM via the EM algorithm
- For this, we use the forward-backward algorithm for the E-step
- One parameters are estimated, we can also solve lots of interesting inference tasks. Again, via forward-backward, or its cousin the 'Viterbi' algorithm.