CSCI567 Machine Learning (Fall 2021)

Prof. Haipeng Luo

U of Southern California

Nov 11, 2021

Administration

HW5 will be released today (due on Nov 23).

Outline

1. Review of last lecture
2. (Hidden) Markov models
3. Review of HW4

Outline

Naive Bayes

Assume: **conditioning on a label, features are independent**

$$p(\boldsymbol{x}, y) = p(y)p(\boldsymbol{x} \mid y) = p(y) \prod_{d=1}^{\mathsf{D}} p(x_d \mid y = c)$$

For a label $c \in [\mathsf{C}]$,

$$p(y = c) = \frac{|\{n : y_n = c\}|}{N}$$

For each possible value k of a discrete feature d,

$$p(x_d = k \mid y = c) = \frac{|\{n : x_{nd} = k, y_n = c\}|}{|\{n : y_n = c\}|}$$

Naive Bayes: continuous features

If the feature is continuous, we can do

- parametric estimation, e.g. via a Gaussian

$$p(x_d = x \mid y = c) = \frac{1}{\sqrt{2\pi}\sigma_{cd}} \exp\left(-\frac{(x - \mu_{cd})^2}{2\sigma_{cd}^2}\right)$$

  where μ_{cd} and σ_{cd}^2 are the empirical mean and variance of feature d among all examples with label c.

- or nonparametric estimation, e.g. via a Kernel K and bandwidth h:

$$p(x_d = x \mid y = c) = \frac{1}{|\{n : y_n = c\}|} \sum_{n:y_n=c} K_h(x - x_{nd})$$

PCA

Input: a dataset represented as $\boldsymbol{X}$, $\#$components p we want

Step 1 Center the data by subtracting the mean

Step 2 Find the top p eigenvectors (with unit norm) of the covariance matrix $\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}$, denoted by $\boldsymbol{V} \in \mathbb{R}^{\mathsf{D} \times p}$

Step 3 Construct the new compressed dataset $\boldsymbol{X}\boldsymbol{V} \in \mathbb{R}^{N \times p}$

KPCA

Input: a dataset $\boldsymbol{X}$, #components p we want, **a kernel fucntion** k

Step 1 Compute the Gram matrix $\boldsymbol{K}$ and the centered Gram matrix

$$\bar{\boldsymbol{K}} = \boldsymbol{K} - \boldsymbol{E}\boldsymbol{K} - \boldsymbol{K}\boldsymbol{E} + \boldsymbol{E}\boldsymbol{K}\boldsymbol{E} \qquad \text{(implicitly centering } \boldsymbol{\Phi}\text{)}$$

Step 2 Find the top p eigenvectors of $\bar{\boldsymbol{K}}$ with the appropriate scaling, denoted by $\boldsymbol{A} \in \mathbb{R}^{\mathsf{N}\times p}$

(implicitly finding unit eigenvectors of $\bar{\boldsymbol{\Phi}}^{\mathrm{T}}\bar{\boldsymbol{\Phi}}$: $\boldsymbol{V} = \bar{\boldsymbol{\Phi}}^{\mathrm{T}}\boldsymbol{A} \in \mathbb{R}^{\mathsf{M}\times p}$)

Step 3 Construct the new dataset $\bar{\boldsymbol{K}}\boldsymbol{A} \in \mathbb{R}^{\mathsf{N}\times p}$

(implicitly/equivalently computing $\bar{\boldsymbol{\Phi}}\boldsymbol{V} = \bar{\boldsymbol{\Phi}}\bar{\boldsymbol{\Phi}}^{\mathrm{T}}\boldsymbol{A}$)

Outline

Markov Models

Markov models are powerful probabilistic tools to analyze **sequential data**:

- text or speech data
- stock market data
- gene data
- ...

Definition

A **Markov chain** is a stochastic process with **Markov property**: a sequence of random variables $Z_1, Z_2, \cdots$ s.t.

$$P(Z_{t+1} \mid Z_{1:t}) = P(Z_{t+1} \mid Z_t) \qquad \text{(Markov property)}$$

i.e. *the current state only depends on the most recent state* (notation $Z_{1:t}$ denotes the sequence $Z_1, \ldots, Z_t$).

We only consider the following case:

- All Z_t's take value from the same discrete set $\{1, \ldots, S\}$
- $P(Z_{t+1} = s' \mid Z_t = s) = a_{s,s'}$, known as transition probability
- $P(Z_1 = s) = \pi_s$
- $(\{\pi_s\}, \{a_{s,s'}\}) = (\boldsymbol{\pi}, \boldsymbol{A})$ are parameters of the model

Examples

- Example 1 (**Language model**)

  States $[S]$ represent a dictionary of words,

  $$a_{\text{ice,cream}} = P(Z_{t+1} = \text{cream} \mid Z_t = \text{ice})$$

  is an example of the transition probability.

- Example 2 (**Weather**)

  States $[S]$ represent weather at each day

  $$a_{\text{sunny,rainy}} = P(Z_{t+1} = \text{rainy} \mid Z_t = \text{sunny})$$

High-order Markov chain

Is the Markov assumption reasonable? Not completely for the language model for example.

Higher order Markov chains make it more reasonable, e.g.

$$P(Z_{t+1} \mid Z_{1:t}) = P(Z_{t+1} \mid Z_t, Z_{t-1}) \qquad \textbf{(second-order Markov)}$$

i.e. the current word only depends on the last two words.

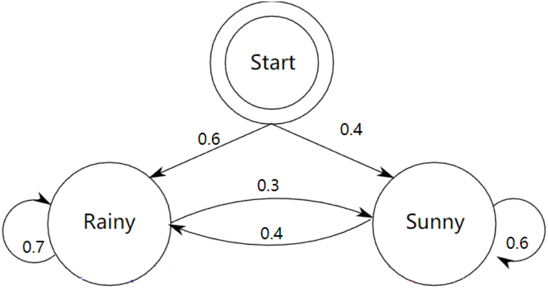
Learning higher order Markov chains is similar, but more expensive.

We only consider standard Markov chains.

Graph Representation

picture from Wikipedia

It is intuitive to represent a Markov model as a **graph**



Learning from examples

Now suppose we have observed N **sequences of examples**:

- $z_{1,1}, \ldots, z_{1,T}$
- $\cdots$
- $z_{n,1}, \ldots, z_{n,T}$
- $\cdots$
- $z_{N,1}, \ldots, z_{N,T}$

where

- for simplicity we assume each sequence has the same length T
- lower case $z_{n,t}$ represents the value of the random variable $Z_{n,t}$

From these observations how do we *learn the model parameters* $(\boldsymbol{\pi}, \boldsymbol{A})$?

Finding the MLE

Same story, find the **MLE**. The log-likelihood of a sequence $z_1, \ldots, z_T$ is

$$\begin{aligned}
&\ln P(Z_{1:T} = z_{1:T}) \\
&= \sum_{t=1}^{T} \ln P(Z_t = z_t \mid Z_{1:t-1} = z_{1:t-1}) && \text{(always true)} \\
&= \sum_{t=1}^{T} \ln P(Z_t = z_t \mid Z_{t-1} = z_{t-1}) && \text{(Markov property)} \\
&= \ln \pi_{z_1} + \sum_{t=2}^{T} \ln a_{z_{t-1}, z_t} \\
&= \sum_{s} \mathbb{I}[z_1 = s] \ln \pi_s + \sum_{s,s'} \left(\sum_{t=2}^{T} \mathbb{I}[z_{t-1} = s, z_t = s'] \right) \ln a_{s,s'}
\end{aligned}$$

Finding the MLE

So MLE is

$$\operatorname*{argmax}_{\boldsymbol{\pi}, \boldsymbol{A}} \sum_{s} (\textbf{\#initial states with value } s) \ln \pi_s + \sum_{s,s'} (\textbf{\#transitions from } s \textbf{ to } s') \ln a_{s,s'}$$

We have seen this many times. The solution is:

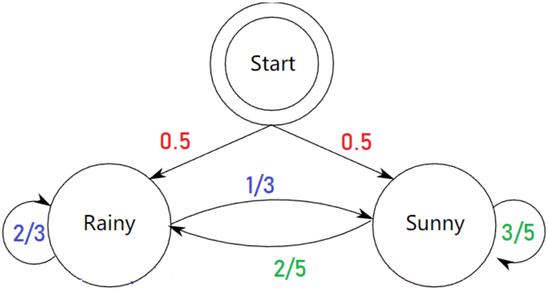
$$\pi_s \propto \#\text{initial states with value } s$$
$$a_{s,s'} \propto \#\text{transitions from } s \text{ to } s'$$

Example

Suppose we observed the following 2 sequences of length 5

- sunny, sunny, rainy, rainy, rainy
- rainy, sunny, sunny, sunny, rainy

MLE is the following model



Markov Model with outcomes

Now suppose each state Z_t also "emits" some **outcome** $X_t \in [O]$ based on the following model

$$P(X_t = o \mid Z_t = s) = b_{s,o} \qquad \text{(emission probability)}$$

independent of anything else.

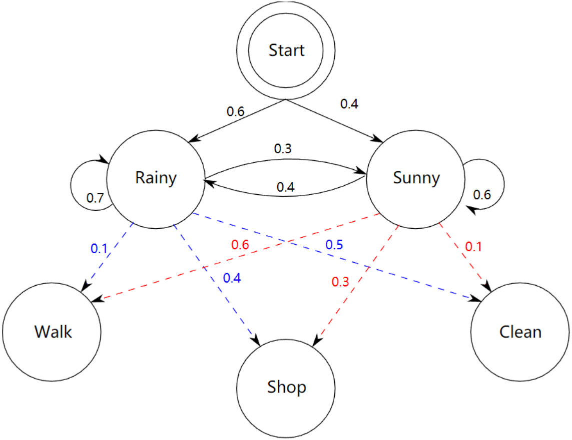
For example, in the language model, X_t is the speech signal for the underlying word Z_t (very useful for speech recognition).

Now the model parameters are $(\{\pi_s\}, \{a_{s,s'}\}, \{b_{s,o}\}) = (\boldsymbol{\pi}, \boldsymbol{A}, \boldsymbol{B})$.

Another example

picture from Wikipedia

On each day, we also observe **Bob's activity: walk, shop, or clean**, which only depends on the weather of that day.



Joint likelihood

The joint log-likelihood of a state-outcome sequence $z_1, x_1, \ldots, z_T, x_T$ is

$$
\begin{aligned}
&\ln P(Z_{1:T} = z_{1:T}, X_{1:T} = x_{1:T}) \\
&= \ln P(Z_{1:T} = z_{1:T}) + \ln P(X_{1:T} = x_{1:T} \mid Z_{1:T} = z_{1:T}) \quad \text{(always true)} \\
&= \sum_{t=1}^{T} \ln P(Z_t = z_t \mid Z_{t-1} = z_{t-1}) + \sum_{t=1}^{T} \ln P(X_t = x_t \mid Z_t = z_t)
\end{aligned}
$$

(due to all the independence)

$$
= \ln \pi_{z_1} + \sum_{t=2}^{T} \ln a_{z_{t-1}, z_t} + \sum_{t=1}^{T} \ln b_{z_t, x_t}
$$

Learning the model

If we observe N state-outcome sequences: $z_{n,1}, x_{n,1}, \dots, z_{n,T}, x_{n,T}$ for $n = 1, \dots, N$, the MLE is again very simple (verify yourself):

$$\pi_s \propto \textbf{\#initial states with value } s$$
$$a_{s,s'} \propto \textbf{\#transitions from } s \textbf{ to } s'$$
$$b_{s,o} \propto \textbf{\#state-outcome pairs } (s,o)$$

Learning the model

However, *most often we do not observe the states!* Think about the speech recognition example.

This is called **Hidden Markov Model (HMM)**, widely used in practice

How to learn HMMs? **Roadmap**:

- first discuss how to **infer** when the model is known (key: dynamic programming)
- then discuss how to **learn** the model (key: EM)

What can we infer about an HMM?

Knowing the parameter of an HMM, we can infer

- **the probability of observing some sequence**

$$P(X_{1:T} = x_{1:T})$$

  e.g. prob. of observing Bob's activities "walk, walk, shop, clean, walk, shop, shop" for one week

- **the state at some point, given an observation sequence**

$$P(Z_t = s \mid X_{1:T} = x_{1:T})$$

  e.g. given Bob's activities for one week, how was the weather like on Wed?

What can we infer for a known HMM?

Knowing the parameter of an HMM, we can infer

- **the transition at some point, given an observation sequence**

$$P(Z_t = s, Z_{t+1} = s' \mid X_{1:T} = x_{1:T})$$

  e.g. given Bob's activities for one week, how was the weather like on Wed and Thu?

- **most likely hidden states path, given an observation sequence**

$$\operatorname*{argmax}_{z_{1:T}} P(Z_{1:T} = z_{1:T} \mid X_{1:T} = x_{1:T})$$

  e.g. given Bob's activities for one week, what's the most likely weather for this week?

Forward and backward messages

The key to infer all these is to compute two things:

- **forward messages**: for each s and t

$$\alpha_s(t) = P(Z_t = s, X_{1:t} = x_{1:t})$$

- **backward messages**: for each s and t

$$\beta_s(t) = P(X_{t+1:T} = x_{t+1:T} \mid Z_t = s)$$

Computing forward messages

Key: *establish a recursive formula*

$$
\begin{aligned}
&\alpha_s(t) \\
&= P(Z_t = s, X_{1:t} = x_{1:t}) \\
&= P(X_t = x_t \mid Z_t = s, X_{1:t-1} = x_{1:t-1}) P(Z_t = s, X_{1:t-1} = x_{1:t-1}) \\
&= b_{s,x_t} \sum_{s'} P(Z_t = s, Z_{t-1} = s', X_{1:t-1} = x_{1:t-1}) \qquad \text{(marginalizing)} \\
&= b_{s,x_t} \sum_{s'} P(Z_t = s | Z_{t-1} = s', X_{1:t-1} = x_{1:t-1}) P(Z_{t-1} = s', X_{1:t-1} = x_{1:t-1}) \\
&= b_{s,x_t} \sum_{s'} a_{s',s} \alpha_{s'}(t-1) \qquad \text{(recursive form!)}
\end{aligned}
$$

Base case: $\alpha_s(1) = P(Z_1 = s, X_1 = x_1) = \pi_s b_{s,x_1}$

Forward procedure

Forward procedure

For all $s \in [S]$, compute $\alpha_s(1) = \pi_s b_{s,x_1}$.

For $t = 2, \ldots, T$

- for each $s \in [S]$, compute

$$\alpha_s(t) = b_{s,x_t} \sum_{s'} a_{s',s} \alpha_{s'}(t-1)$$

It takes $O(S^2 T)$ time and $O(ST)$ space.

Computing backward messages

Again establish a recursive formula

$$\begin{aligned}
&\beta_s(t) \\
&= P(X_{t+1:T} = x_{t+1:T} \mid Z_t = s) \\
&= \sum_{s'} P(X_{t+1:T} = x_{t+1:T}, Z_{t+1} = s' \mid Z_t = s) && \text{(marginalizing)} \\
&= \sum_{s'} P(Z_{t+1} = s' \mid Z_t = s) P(X_{t+1:T} = x_{t+1:T} \mid Z_{t+1} = s', Z_t = s) \\
&= \sum_{s'} a_{s,s'} P(X_{t+1} = x_{t+1} \mid Z_{t+1} = s') P(X_{t+2:T} = x_{t+2:T} \mid Z_{t+1} = s') \\
&= \sum_{s'} a_{s,s'} b_{s',x_{t+1}} \beta_{s'}(t+1) && \text{(\textit{recursive form!})}
\end{aligned}$$

Base case: $\beta_s(T) = 1$

Backward procedure

Backward procedure

For all $s \in [S]$, set $\beta_s(T) = 1$.

For $t = T-1, \ldots, 1$

- for each $s \in [S]$, compute

$$\beta_s(t) = \sum_{s'} a_{s,s'} b_{s',x_{t+1}} \beta_{s'}(t+1)$$

Again it takes $O(S^2T)$ time and $O(ST)$ space.

Using forward and backward messages

With forward and backward messages, we can easily infer many things, e.g.

$$\begin{aligned}
\gamma_s(t) &= P(Z_t = s \mid X_{1:T} = x_{1:T}) \\
&\propto P(Z_t = s, X_{1:T} = x_{1:T}) \\
&= P(Z_t = s, X_{1:t} = x_{1:t}) P(X_{t+1:T} = x_{t+1:T} \mid Z_t = s, X_{1:t} = x_{1:t}) \\
&= \alpha_s(t)\beta_s(t)
\end{aligned}$$

What constant are we omitting in "$\propto$"? It is exactly

$$P(X_{1:T} = x_{1:T}) = \sum_s \alpha_s(t)\beta_s(t),$$

the probability of observing the sequence $x_{1:T}$.

This is true for any t; a good way to check correctness of your code.

Using forward and backward messages

Another example: the conditional probability of transition s to s' at time t

$$
\begin{aligned}
&\xi_{s,s'}(t) \\
&= P(Z_t = s, Z_{t+1} = s' \mid X_{1:T} = x_{1:T}) \\
&\propto P(Z_t = s, Z_{t+1} = s', X_{1:T} = x_{1:T}) \\
&= P(Z_t = s, X_{1:t} = x_{1:t}) P(Z_{t+1} = s', X_{t+1:T} = x_{t+1:T} \mid Z_t = s, X_{1:t} = x_{1:t}) \\
&= \alpha_s(t) P(Z_{t+1} = s' \mid Z_t = s) P(X_{t+1:T} = x_{t+1:T} \mid Z_{t+1} = s') \\
&= \alpha_s(t) a_{s,s'} P(X_{t+1} = x_{t+1} \mid Z_{t+1} = s') P(X_{t+2:T} = x_{t+2:T} \mid Z_{t+1} = s') \\
&= \alpha_s(t) a_{s,s'} b_{s',x_{t+1}} \beta_{s'}(t+1)
\end{aligned}
$$

The normalization constant is in fact again $P(X_{1:T} = x_{1:T})$

Finding the most likely path

Though can't use forward and backward messages directly to find the most likely path, it is very similar to the forward procedure. Key: compute

$$\delta_s(t) = \max_{z_{1:t-1}} P(Z_t = s, Z_{1:t-1} = z_{1:t-1}, X_{1:t} = x_{1:t})$$

the probability of the most likely path for time $1 : t$ ending at state s

Computing $\delta_s(t)$

Observe

$$
\begin{aligned}
\delta_s(t) &= \max_{z_{1:t-1}} P(Z_t = s, Z_{1:t-1} = z_{1:t-1}, X_{1:t} = x_{1:t}) \\
&= \max_{s'} \max_{z_{1:t-2}} P(Z_t = s, Z_{t-1} = s', Z_{1:t-2} = z_{1:t-2}, X_{1:t} = x_{1:t}) \\
&= \max_{s'} P(Z_t = s \mid Z_{t-1} = s') P(X_t = x_t \mid Z_t = s) \cdot \\
&\qquad \max_{z_{1:t-2}} P(Z_{t-1} = s', Z_{1:t-2} = z_{1:t-2}, X_{1:t-1} = x_{1:t-1}) \\
&= b_{s,x_t} \max_{s'} a_{s',s} \delta_{s'}(t-1) && (\textit{recursive form!})
\end{aligned}
$$

Base case: $\delta_s(1) = P(Z_1 = s, X_1 = x_1) = \pi_s b_{s,x_1}$

Exactly the same as forward messages except replacing "sum" by "max"!

Viterbi Algorithm (!)

Viterbi Algorithm

For each $s \in [S]$, compute $\delta_s(1) = \pi_s b_{s,x_1}$.

For each $t = 2, \ldots, T$,

- for each $s \in [S]$, compute

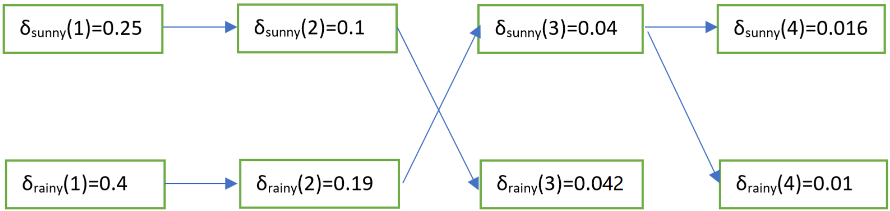
$$\delta_s(t) = b_{s,x_t} \max_{s'} a_{s',s} \delta_{s'}(t-1),$$
$$\Delta_s(t) = \operatorname*{argmax}_{s'} a_{s',s} \delta_{s'}(t-1).$$

Backtracking: let $z_T^* = \operatorname{argmax}_s \delta_s(T)$.
For each $t = T, \ldots, 2$: set $z_{t-1}^* = \Delta_{z_t^*}(t)$.

Output the most likely path $z_1^*, \ldots, z_T^*$.

Example

Arrows represent the "argmax", i.e. $\Delta_s(t)$.



The most likely path is **"rainy, rainy, sunny, sunny"**.

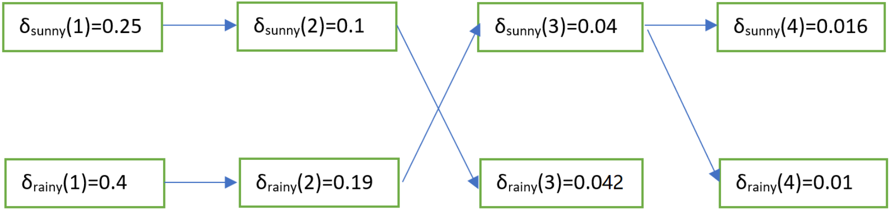
Exercise 1

What is the most likely sequence $z^*_{1:T_0}$ given $x_{1:T_0}$ for some $T_0 < T$?

- Is it the first T_0 outputs of the Viterbi algorithm (with all data)?

No. It should be

- $z^*_{T_0} = \text{argmax}_s \, \delta_s(T_0)$
- for each $t = T_0, \ldots, 2$: $z^*_{t-1} = \Delta_{z^*_t}(t)$



The answer for $T_0 = 3$ is: **"sunny, sunny, rainy"**.

Exercise 2

What is the most likely sequence $z^*_{1:T_0}$ given $x_{1:T}$ for some $T_0 < T$?

- Is it the same as Exercise 1?
- Is it the first T_0 outputs of the Viterbi algorithm (with all data)?

Neither. It should be

- $z^*_{T_0} = \operatorname{argmax}_s \delta_s(T_0)\beta_s(T_0)$
- for each $t = T_0, \ldots, 2$: $z^*_{t-1} = \Delta_{z^*_t}(t)$

Exercise 2 (cont.)

Reasoning:

$$
\begin{aligned}
z^*_{T_0} &= \underset{s}{\operatorname{argmax}} \max_{z_{1:T_0-1}} P(Z_{T_0} = s, Z_{1:T_0-1} = z_{1:T_0-1}, X_{1:T} = x_{1:T}) \\
&= \underset{s}{\operatorname{argmax}} \max_{z_{1:T_0-1}} P(Z_{T_0} = s, Z_{1:T_0-1} = z_{1:T_0-1}, X_{1:T_0} = x_{1:T_0}) \cdot \\
&\qquad P(X_{T_0+1,T} = x_{T_0+1:T} \mid Z_{T_0} = s, Z_{1:T_0-1} = z_{1:T_0-1}, X_{1:T_0} = x_{1:T_0}) \\
&= \underset{s}{\operatorname{argmax}} \left(\max_{z_{1:T_0-1}} P(Z_{T_0} = s, Z_{1:T_0-1} = z_{1:T_0-1}, X_{1:T_0} = x_{1:T_0}) \right) \cdot \\
&\qquad P(X_{T_0+1,T} = x_{T_0+1:T} \mid Z_{T_0} = s) \\
&= \underset{s}{\operatorname{argmax}}\ \delta_s(T_0)\beta_s(T_0)
\end{aligned}
$$

Exercise 3

What is the most likely sequence $z^*_{1:T}$ given $x_{1:T_0}$ for some $T_0 < T$?

- Is it the same as the Viterbi algorithm (with all data)?
- Are the first T_0 states the same as Exercise 1?

Again, neither is true.

Exercise 3 (cont.)

Viterbi Algorithm with partial data $x_{1:T_0}$

For each $s \in [S]$, compute $\delta_s(1) = \pi_s b_{s,x_1}$.

For each $t = 2, \ldots, T$,

- for each $s \in [S]$, compute

$$\delta_s(t) = \begin{cases} b_{s,x_t} \max_{s'} a_{s',s}\delta_{s'}(t-1) & \text{if } t \leq T_0 \\ \max_{s'} a_{s',s}\delta_{s'}(t-1) & \text{else} \end{cases}$$

$$\Delta_s(t) = \operatorname*{argmax}_{s'} a_{s',s}\delta_{s'}(t-1).$$

Backtracking: let $z_T^* = \operatorname{argmax}_s \delta_s(T)$.
For each $t = T, \ldots, 2$: set $z_{t-1}^* = \Delta_{z_t^*}(t)$.

Output the most likely path $z_1^*, \ldots, z_T^*$.

Learning the parameters of an HMM

All previous inferences depend on knowing the parameters $(\boldsymbol{\pi}, \boldsymbol{A}, \boldsymbol{B})$.

How do we learn the parameters based on N observation sequences $x_{n,1}, \ldots, x_{n,T}$ for $n = 1, \ldots, N$?

MLE is intractable due to the hidden variables $Z_{n,t}$'s (similar to GMMs)

Need to apply **EM** again! Known as the **Baum–Welch algorithm**.

Applying EM: E-Step

Recall in the E-Step we fix the parameters and find the **posterior distributions q of the hidden states** (for each sample n), which leads to the complete log-likelihood:

$$
\begin{aligned}
&\mathbb{E}_{z_{1:T}\sim q}\left[\ln(Z_{1:T} = z_{1:T}, X_{1:T} = x_{1:T})\right] \\
&= \mathbb{E}_{z_{1:T}\sim q}\left[\ln \pi_{z_1} + \sum_{t=1}^{T-1} \ln a_{z_t, z_{t+1}} + \sum_{t=1}^{T} \ln b_{z_t, x_t}\right] \\
&= \sum_{s} \gamma_s(1) \ln \pi_s + \sum_{t=1}^{T-1} \sum_{s,s'} \xi_{s,s'}(t) \ln a_{s,s'} + \sum_{t=1}^{T} \sum_{s} \gamma_s(t) \ln b_{s,x_t}
\end{aligned}
$$

We have discussed how to compute

$$
\begin{aligned}
\gamma_s(t) &= P(Z_t = s \mid X_{1:T} = x_{1:T}) \\
\xi_{s,s'}(t) &= P(Z_t = s, Z_{t+1} = s' \mid X_{1:T} = x_{1:T})
\end{aligned}
$$

Applying EM: M-Step

The maximizer of complete log-likelihood is simply doing **weighted counting** (compared to the unweighted counting on Slide 22):

$$\pi_s \propto \sum_n \gamma_s^{(n)}(1) = \mathbb{E}_q\left[\ \textbf{\#initial states with value } s\right]$$

$$a_{s,s'} \propto \sum_n \sum_{t=1}^{T-1} \xi_{s,s'}^{(n)}(t) = \mathbb{E}_q\left[\ \textbf{\#transitions from } s \textbf{ to } s'\right]$$

$$b_{s,o} \propto \sum_n \sum_{t:x_t=o} \gamma_s^{(n)}(t) = \mathbb{E}_q\left[\ \textbf{\#state-outcome pairs } (s,o)\right]$$

where

$$\gamma_s^{(n)}(t) = P(Z_{n,t} = s \mid X_{n,1:T} = x_{n,1:T})$$

$$\xi_{s,s'}^{(n)}(t) = P(Z_{n,t} = s, Z_{n,t+1} = s' \mid X_{n,1:T} = x_{n,1:T})$$

Baum–Welch algorithm

Step 0 Initialize the parameters $(\boldsymbol{\pi}, \boldsymbol{A}, \boldsymbol{B})$

Step 1 (E-Step) Fixing the parameters, compute forward and backward messages for all sample sequences, then use these to compute $\gamma_s^{(n)}(t)$ and $\xi_{s,s'}^{(n)}(t)$ for each n, t, s, s' (see Slides 31 and 32).

Step 2 (M-Step) Update parameters:

$$\pi_s \propto \sum_n \gamma_s^{(n)}(1), \quad a_{s,s'} \propto \sum_n \sum_{t=1}^{T-1} \xi_{s,s'}^{(n)}(t), \quad b_{s,o} \propto \sum_n \sum_{t:x_t=o} \gamma_s^{(n)}(t)$$

Step 3 Return to Step 1 if not converged

Summary

Very important models: **Markov chains**, **hidden Markov models**

Several algorithms:

- forward and backward procedures
- inferring HMMs based on forward and backward messages
- Viterbi algorithm
- Baum–Welch algorithm

Outline