

SPR Day 19

Inferring the Most Likely State Sequence under an HMM using Viterbi algorithm

Reading: Bishop PRML 13.2.5

- Topics:
- Motivation to find most likely hidden state sequence
 - Formal Problem Statement (+ why brute force is hard)
 - Intuition for a recursive algorithm
 - Viterbi algorithm

Motivation: Inferring Sequences of Hidden States

Common task in many sequential data applications

given data x_1, x_2, \dots, x_T

produce a "good" guess about system state overtime

z_1, z_2, \dots, z_T

$z_t \in \{1, 2, \dots, K\}$

Examples

E1: Messages over noisy channels

true states: English alphabet $\{a, b, c, d, \dots, x, y, z\}$

observed data: binary codes

x_1	x_2	x_3	\dots	x_T
0	0	0		1
0	1	0		1
0	0	1		1
0	1	0		0
0	1	0		0

$P(x/z)$
some chance
this is a 0 instead
but got corrupted

Inference goal:

Which is more likely the message?

a: the british will strike at midnight

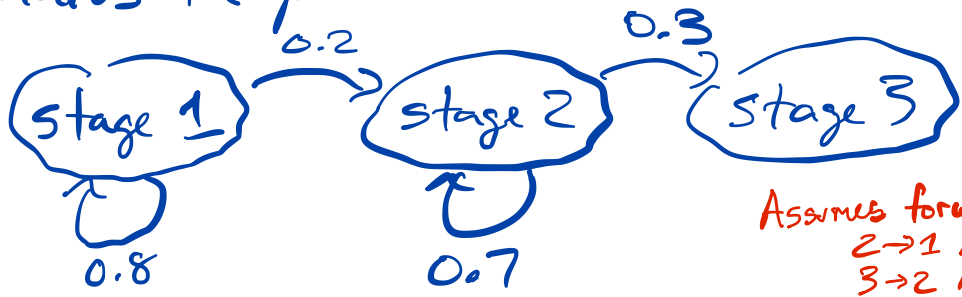
b: the british will strike at midnight

Known probabilities of transitions between letters useful!

E2: Disease status in patient over time

Model

Markov model
for disease
progression



Assumes forward progression
2 → 1 NOT allowed
3 → 2 NOT allowed

Observations

symptoms at each time x_t

Goal:

Given symptom history, what stage was patient at in each timestep?

$$z_1, z_2, \dots, z_T$$

$$z_{1:T} = 1, 1, 2, 2, 2, 2, 3, 3$$

1 = stage 1
2 = stage 2
3 = stage 3

Key idea: need to do inference of all z_1, z_2, \dots, z_T values jointly, will give better, more useful results.

Want to avoid implausible configurations, e.g.

$$z_{1:T} = 1, 2, \boxed{2, 1}, 2, 2, 3, 3$$

Our Markov model says cannot go back from stage 2 to stage 1

Formal Problem Statement

Inference of Most Likely Hidden State Seq.

Given: Data at each timestep: x_1, x_2, \dots, x_T

HMM Parameters (assume Gaussian data-given-state distributions)

- $\pi \in \Delta^K$ initial state probs

- $A = \{A_j\}_{j=1}^K$ $A_j \in \Delta^K$ transition probs

- $\mu = \{\mu_k\}_{k=1}^K$ means and variances

$\sigma = \{\sigma_k\}_{k=1}^K$

Infer: $\hat{z}_1, \hat{z}_2, \dots, \hat{z}_T$, a state sequence that satisfies:

- $\hat{z}_{1:T} = \underset{z_{1:T} \in \Omega}{\operatorname{argmax}} p(z_{1:T} | x_{1:T}, \theta)$
↑
space of all possible sequences of size T w/ K possible symbols

We could call this "MAP estimation" because we are finding the sequence $z_{1:T}$ that is most likely under posterior $p(z|x, \theta)$

Brute Force Method

We could enumerate all $z_{1:T} \in \Omega$, and for each one compute $\log p(x_{1:T}, z_{1:T} | \theta)$ [the complete log likelihood] \Rightarrow costs $O(TK^2)$ runtime for each $z_{1:T}$

E.g. for $T=4, K=3$ we would do

$z_{1:T}$	$\log p(x_{1:T}, z_{1:T} \theta)$
-----------	-------------------------------------

1, 1, 1, 1

0.001

1, 1, 1, 2

0.003

1, 1, 1, 3

0.023

⋮

⋮

2, 1, 1, 1

-0.001

2, 1, 1, 2

0.001

⋮

⋮

4, 4, 4, 3

-0.001

4, 4, 4, 4

-0.002

return $z_{1:T}$ with largest value

Equivalent goals

$$\max_{z_{1:T}} p(z_{1:T} | x_{1:T})$$

$$= \max_{z_{1:T}} \frac{p(z_{1:T}, x_{1:T})}{\text{const}} \quad \text{Bayes rule}$$

$$= \max_{z_{1:T}} \log p(z_{1:T}, x_{1:T}) \quad \text{log is monotonic}$$

Problem: Each row costs $O(TK^2)$

And there are K^T possible rows.

Once $K > 10$ and $T > 10$, this brute force becomes way too impractical

Idea: Approach like FORWARD and BACKWARD algo.

Find subproblems easier to solve, use those to build up overall solution

Base case: $T=1$ This is doable!

$$\hat{z}_1 = \operatorname{argmax}_{z_1 \in \{1, 2, \dots, K\}} \log p(x_1, z_1 / \theta)$$

$$= \operatorname{argmax}_{k \in \{1, 2, \dots, K\}} \log p(z_1=k / \theta) + \log p(x_1 / z_1=k, \theta)$$

$\log \pi_k + \underbrace{\log \text{NormPDF}(x_1 / \mu_k, \sigma_k)}_{L_{1k}}$

Notation:

$$w_{1j} = \log p(z_1=j) + \log p(x_1 / z_1=j)$$

Joint log prob of x_1 and $z_1=j$

Lets define for timesteps $t \in 1 \dots T$

$$L_{tk} = \log p(x_t / z_t=k, \theta)$$

from here on, for easy notation

Next, consider $T=2$

$$\hat{z}_1, \hat{z}_2 = \operatorname{argmax}_{k \in \{1, 2, \dots, K\}} \operatorname{argmax}_{j \in \{1, 2, \dots, K\}} \log p(z_1=j, z_2=k, x_1, x_2)$$

$$= \operatorname{argmax}_k \log p(x_2 / z_2=k) + \operatorname{argmax}_j \log p(z_2=k / z_1=j) + w_{1j}$$

ignore dependence on θ for now...

Track score: Joint log prob of $x_{1:2}$, $z_2=k$, and best choice for z_1

$$w_{2k} = \log p(x_2 / z_2=k) + \max_j [\log p(z_2=k / z_1=j) + w_{1j}]$$

Can't do j argmax before k argmax

Next, consider $T=3$

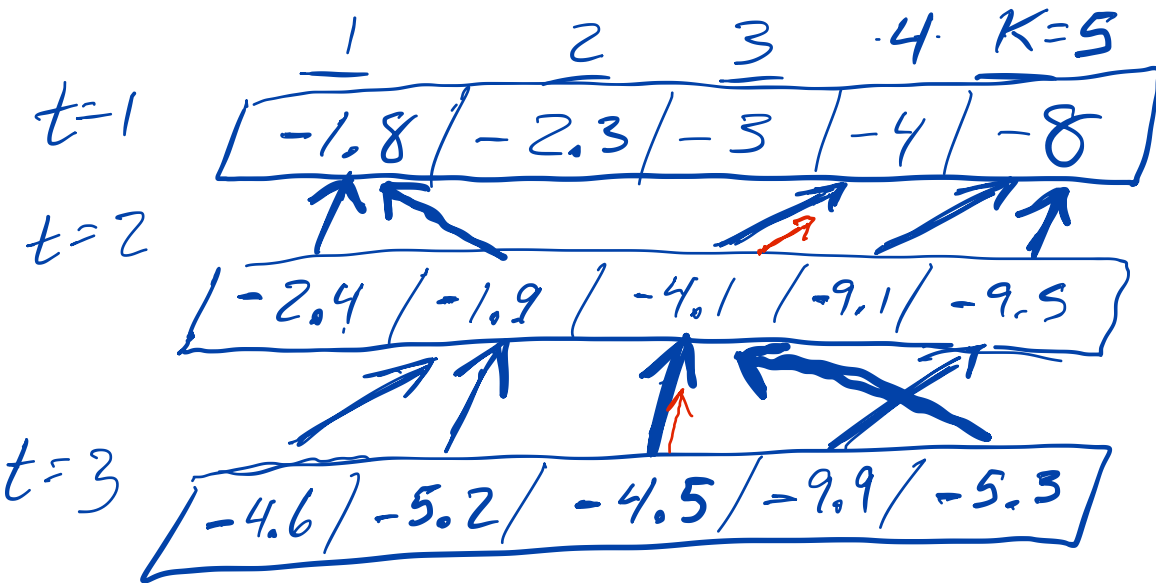
$$z_1, z_2, z_3 = \underset{j, k, l}{\operatorname{argmax}} \left[\log p(x_3 | z_3=l) + \underset{k \in \{1, \dots, K\}}{\operatorname{argmax}} \left(\log p(z_3=l | z_2=k) + \log p(z_1, z_2, x_1, x_2) \right) \right]$$

w_{2k} can tell us this

See the recursive structure?

For each possible timestep t and state k , we track:

- score $w_{tk} \in \mathbb{R}$, the complete log likelihood of best sequence that terminates at k at time t
- backpointer $b_{tk} \in \{1, 2, \dots, K\}$ which state j the best possible path uses at previous time $t-1$



numbers represent w_{tk} scores
arrows \uparrow represent backpointers

given this configuration, we find the best sequence by looking at last row ($T=3$) and picking overall highest score (highest joint log likelihood).

We pick -4.5 so $z_3=3$.

Then following back pointers, we have $z_2=3, z_1=4$

So given w_t and b_t , we can solve the problem

This insight gives us the

VITERBI Algorithm

Dynamic Programming

Input: π, A : HMM parameters

$L = T \times K$ array
where $L_{tk} = \log P(x_t / z_t = k, \theta)$

Think of L
as deterministic summing
of x, μ, σ^2

Initial: $w_{1k} = L_{1k} + \log \pi_k$ for $k \in 1 \dots K$

$t=1$
 $b_{1k} = -1$ (unused, no backpointer at $t=1$)

Recursive update: for t in $2, \dots, T$:

$t=2, 3, \dots, T$ ↓
score $w_{tk} = L_{tk} + \max_j (\log A_{jk} + w_{t-1,j})$ $\forall k$

backpointer $b_{tk} = \operatorname{argmax}_j \log A_{jk} + w_{t-1,j}$ $\forall k$

Backward sweep to identify state sequence

$\hat{z}_T = \operatorname{argmax}_k w_{Tk}$

for $t = T-1, T-2, \dots, 1$:

$\hat{z}_t = b_{t+1, \hat{z}_{t+1}}$

follow back pointer

Return $\hat{z}_1, \hat{z}_2, \dots, \hat{z}_{T-1}, \hat{z}_T$

Runtime: Linear in T

Quadratic in K

need to visit each k and perform $\max_{j \in \{1, \dots, K\}}$

$O(TK^2)$