Markov Chains and Hidden Markov Models

COMP 571
Luay Nakhleh, Rice University

1

Markov Chains and Hidden Markov Models

- Modeling the statistical properties of biological sequences and distinguishing regions based on these models
- For the alignment problem, they provide a probabilistic framework for aligning sequences

2

Example: CpG Islands

- Regions that are rich in CG dinucleotide
- Promoter and "start" regions of many genes are characterized by high frequency of CG dinucleotides (in fact, more C and G nucleotides in general)

3

CpG Islands: Two Questions

4

- Q1: Given a short sequence, does it come from a CpG island?
- Q2: Given a long sequence, how would we find the CpG islands in it?

CpG Islands

5

- Answer to Q1:
  - Given sequence x and probabilistic model M of CpG islands, compute p=P(x|M)
  - If p is "significant", then x comes from a CpG island; otherwise, x does not come from a CpG island

CpG Islands

6

- Answer to Q1:
  - Given sequence x, probabilistic model M_1 of CpG islands, and probabilistic model M_2 of non-CpG islands, compute $p_1=P(x|M_1)$ and $p_2=P(x|M_2)$
  - If $p_1>p_2$, then x comes from a CpG island
  - If $p_1<p_2$, then x does not come from a CpG island

CpG Islands

7

- Answer to Q2:
  - As before, use the models M_1 and M_2, calculate the scores for a window of, say, 100 nucleotides around every nucleotide in the sequence
  - Not satisfactory
  - A more satisfactory approach is to build a single model for the entire sequence that incorporates both Markov chains

Difference Between the Two Solutions

8

- Solution to Q1:
  - One "state" for each nucleotide, since we have only one region
  - 1-1 correspondence between "state" and "nucleotide"
- Solution to Q2:
  - Two "states" for each nucleotide (one for the nucleotide in a CpG island, and another for the same nucleotide in a non-CpG island)
  - No 1-1 correspondence between "state" and "nucleotide"

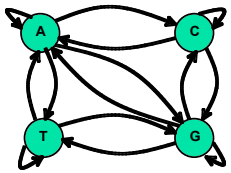
Markov Chains vs. HMMs

9

- When we have a 1-1 correspondence between alphabet letters and states, we have a Markov chain
- When such a correspondence does not hold, we only know the letters (observed data), and the states are "hidden"; hence, we have a hidden Markov model, or HMM

10

Markov Chains

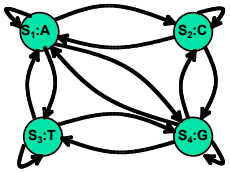


Associated with each edge is a transition probability

11

Markov Chains: The 1-1 Correspondence

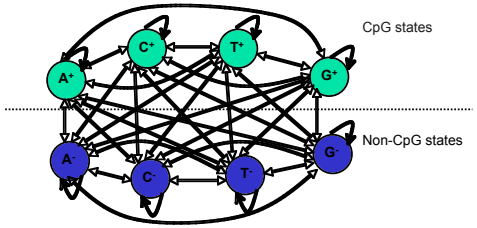
Sequence: GAGCGCGTAC



States: $S_4S_1S_4S_2S_4S_2S_4S_3S_1S_2$

12

HMMs: No 1-1 Correspondence (2 States Per Nucleotide)



What's Hidden?

- We can "see" the nucleotide sequence
- We cannot see the sequence of states, or path, that generated the nucleotide sequence
- Hence, the state sequence (path) that generated the data is hidden

13

Markov Chains and HMMs

- In Markov chains and hidden Markov models, the probability of being in a state depends solely on the previous state
- Dependence on more than the previous state necessitates higher order Markov models

14

Sequence Annotation Using Markov Chains

- The annotation is straightforward: given the input sequence, we have a unique annotation (mapping between sequence letters and model states)
- The outcome is the probability of the sequence given the model

15

Sequence Annotation Using HMMs

16

- For every input sequence, there are many possible annotations (paths in the HMM)
- Annotation corresponds to finding the best mapping between sequence letters and model states (i.e., the path of highest probability that corresponds to the input sequence)

Markov Chains: Formal Definition

17

- A set Q of states
- Transition probabilities
  - $a_{st}=P(x_i=t|x_{i-1}=s)$: probability of state t given that the previous state was s
- In this model, the probability of sequence $x=x_1x_2...x_L$ is

$$\mathbf{P}(x) = \mathbf{P}(x_L|x_{L-1})\mathbf{P}(x_{L-1}|x_{L-2})\cdots\mathbf{P}(x_2|x_1)\mathbf{P}(x_1) = \mathbf{P}(x_1)\prod_{i=2}^{L} a_{x_{i-1}x_i}$$

Markov Chains: Formal Definition

18

- Usually, two states "start" and "end" are added to the Markov chain to model the beginning and end of sequences, respectively
- Adding these two states, the model defines a probability distribution on all possible sequences (of any length)

HMMs: Formal Definition

19

- A set Q of states
- An alphabet Σ
- Transition probability a_{st} for every two states s and t
- Emission probability $e_k(b)$ for every letter b and state k (the probability of emitting letter b in state k)

HMMs: Sequences and Paths

20

- Due to the lack of a 1-1 correspondence, we need to distinguish between the sequence of letters (e.g., DNA sequences) and the sequence of states (path)
- For every sequence (of letters) there are many paths for generating it, each occurring with its probability
- We use x to denote a (DNA) sequence, and π to denote a (state) path

HMMs: The Model Probabilities

21

- Transition probability $a_{k\ell} = \mathbf{P}(\pi_i = \ell | \pi_{i-1} = k)$
- Emission probability $e_k(b) = \mathbf{P}(x_i = b | \pi_i = k)$

HMMs: The Sequence Probabilities

22

- The joint probability of an observed sequence and a path is

$$\mathbf{P}(x, \pi) = a_{0\pi_1} \prod_{i=1}^{L} e_{\pi_i}(x_i) a_{\pi_i \pi_{i+1}}$$

- The probability of a sequence is

$$\mathbf{P}(x) = \sum_{\pi} \mathbf{P}(x, \pi)$$

HMMs: The Parsing Problem

23

- Find the most probable state path that generates a given a sequence

$$\pi^* = \text{argmax}_{\pi} \mathbf{P}(x, \pi)$$

HMMs: The Posterior Decoding Problem

24

- Compute "confidence" for the states on a path

$$\mathbf{P}(\pi_i = k | x)$$

HMMs: The Parameter Estimation Problem

25

* Compute the transition and emission probabilities of an HMM (from a given training data set)

A Toy Example: 5' Splice Site Recognition

26

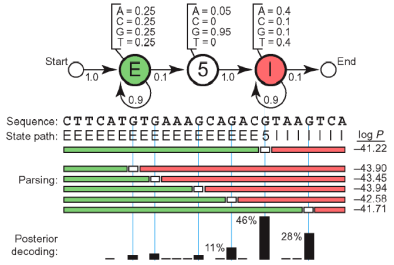
* From "What is a hidden Markov model?", by Sean R. Eddy
* 5' splice site indicates the "switch" from an exon to an intron

A Toy Example: 5' Splice Site Recognition

27

* Assumptions
  * Uniform base composition on average in exons
  * Introns are A/T rich (40% A/T, 10% G/C)
  * The 5' splice site consensus nucleotide is almost always a G (say, 95% G and 5% A)

A Toy Example: 5' Splice Site Recognition



28

HMMs: A DP Algorithm for the Parsing Problem

- Let $v_k(i)$ denote the probability of the most probable path ending in state k with observation x_i
- The DP structure:

$$v_\ell(i+1) = e_\ell(x_{i+1}) \max_k (v_k(i) a_{k\ell})$$

29

The Viterbi Algrorithm

Initialization

Recursion $i = 1 \ldots L$

Termination

Traceback $i = 1 \ldots L$

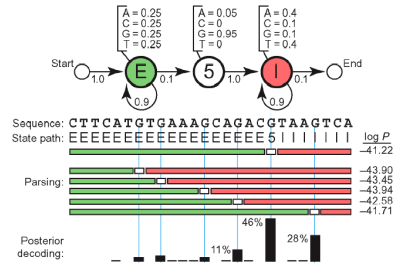
30

The Viterbi Algrorithm

- Usually, the algorithm is implemented to work with logarithms of probabilities so that the multiplication turns into addition
- The algorithm takes $O(Lq^2)$ time and $O(Lq)$ space, where L is the sequence length and q is the number of states

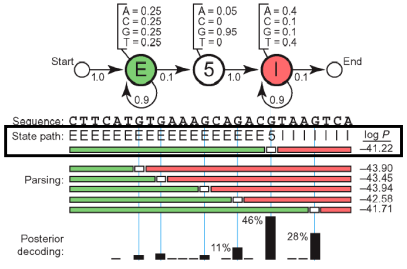
31

A Toy Example: 5′ Splice Site Recognition



32-1

A Toy Example: 5′ Splice Site Recognition



32-2

Other Values of Interest

33

- The probability of a sequence, P(x)
- Posterior decoding: $\mathbf{P}(\pi_i = k|x)$
- Efficient DP algorithms for both using the forward and backward algorithms

The Forward Algorithm

34

- $f_k(i)$: the probability of the observed sequence up to and including x_i, requiring that $\pi_i=k$
- In other words, $f_k(i)=P(x_1,\ldots,x_i, \pi_i=k)$
- The structure of the DP algorithm:

$$f_\ell(i+1) = e_\ell(x_{i+1}) \sum_k f_k(i) a_{k\ell}$$

The Forward Algorithm

35

- Initialization: $f_0(0) = 1, \quad f_k(0) = 0 \; \forall k > 0$
- Recursion: $f_\ell(i) = e_\ell(x_i) \sum_k f_k(i-1) a_{k\ell} \quad i = 1 \ldots L$
- Termination: $\mathbf{P}(x) = \sum_k f_k(L) a_{k0}$

The Backward Algorithm

36

- $b_k(i)$: the probability of the last observed L-i letters, requiring that $\pi_i=k$
- In other words, $b_k(i)=P(x_L,\ldots,x_{i+1}| \pi_i=k)$
- The structure of the DP algorithm:

$$b_\ell(i) = \sum_k a_{\ell k} e_k(x_{i+1}) b_k(i+1)$$

The Backward Algorithm

37

- **Initialization:** $b_k(L) = a_{k0} \quad \forall k$
- **Recursion:** $b_\ell(i) = \sum_k a_{\ell k} e_\ell(x_{i+1}) b_k(i+1) \quad i = L-1, \ldots, 1$
- **Termination:** $\mathbf{P}(x) = \sum_k a_{0k} e_k(x_1) b_k(1)$

The Posterior Probability

38

$$\begin{aligned} f_k(i) b_k(i) &= \mathbf{P}(x, \pi_i = k) \\ &= \mathbf{P}(\pi_i = k|x)\mathbf{P}(x) \end{aligned}$$

$$\Rightarrow \mathbf{P}(\pi_i = k|x) = \frac{f_k(i) b_k(i)}{\mathbf{P}(x)}$$

The Probability of a Sequence

$$\mathbf{P}(x) = \sum_k f_k(L) a_{k0}$$

$$\mathbf{P}(x) = \sum_k a_{0k} e_k(x_1) b_k(1)$$

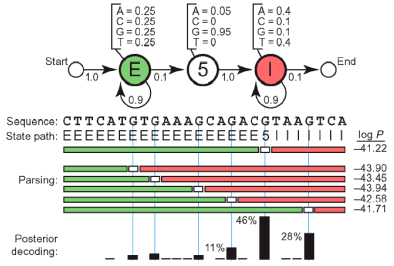
39

Computational Requirements of the Algorithms

- Each of the algorithms takes $O(Lq^2)$ time and $O(Lq)$ space, where L is the sequence length and q is the number of states

40

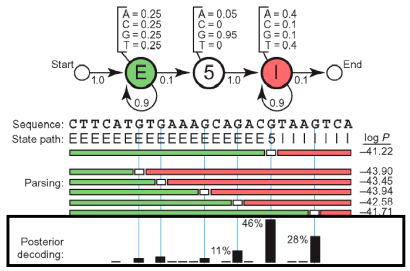
A Toy Example: 5′ Splice Site Recognition



41-1

A Toy Example: 5' Splice Site Recognition

41-2



Applications of Posterior Decoding (1)

42

- Find the sequence π' of states where $\pi'_i = \text{argmax}_k \mathbf{P}(\pi_i = k|x)$
- This is a more appropriate path when we are interested in the state assignment at a particular point i (however, this sequence of states may not be a legitimate path!)

Applications of Posterior Decoding (2)

43

- Assume function g(k) is defined on the set of states
- We can consider $G(i|x) = \sum_k \mathbf{P}(\pi_i = k|x) g(k)$
- For example, for the CpG island problem, setting g(k)=1 for the "+" states, and g(k)=0 for the "-" states, G(i|x) is precisely the posterior probability according to the model that base i is in a CpG island

Parameter Estimation for HMMs

44

- Two components:
  - the probabilities (emission and transition): there is a well-developed theory
  - the structure (states): more of an "art"
- We'll focus on estimating the probabilities

Estimating HMM Emission and Transition Probabilities

45

- Given the structure of an HMM, and a set of training sequences, we'd want to estimate the probabilities from the training data set
- There are two cases
  - The training sequences are already annotated (i.e., the state sequences are known)
  - The training sequences are not annotated (i.e., the state sequences are not known)

Estimating HMM Emission and Transition Probabilities

46

- Maximum likelihood estimation:
  - The data: n sequences $x^1,\ldots,x^n$
  - The model θ: transition and emission probabilities

$$\theta^* = \text{argmax}_\theta \log P(x^1,\ldots,x^n|\theta) = \sum_{j=1}^{n} \log P(x^j|\theta)$$

Estimating HMM Probabilities: Known State Sequences

47

- Given a training data set, count the number of times each particular transition or emission is used; denote these by A_{kl} and $E_k(b)$
- Then, MLEs for a and e are:

(1) $$a_{k\ell} = \frac{A_{k\ell}}{\sum_{\ell'} A_{k\ell'}} \qquad e_k(b) = \frac{E_k(b)}{\sum_{b'} E_k(b')}$$

Estimating HMM Probabilities: Known State Sequences

48

- MLEs are vulnerable to overfitting if there are insufficient data (e.g., what happens if a state k is never used in the set of example sequences?).
- To avoid such problems, it is preferable to add predetermined pseudo-counts to the A and E values.

Estimating HMM Probabilities: Known State Sequences

49

- A_{kl} = number of transitions k to l in training data + r_{kl}
- $E_k(b)$ = number of emissions of b from k in training data + $r_k(b)$

Estimating HMM Probabilities: Known State Sequences

50

- The pseudo-counts r_{kl} and $r_k(b)$ should reflect prior biases about the probability values.
- Small total values $\Sigma_l r_{kl}$ or $\Sigma_b r_k(b)$ indicate weak prior knowledge, whereas larger total values indicate more definite prior knowledge, which requires more data to modify it.

Estimating HMM Probabilities: Unknown State Sequences

51

- The Baum-Welch algorithm, which is an expectation-maximization (EM) algorithm
- EM is a general algorithm for ML estimation with "missing data."

The EM Algorithm

52

- Assume some statistical model is determined by parameters θ.
- The observed quantities are x, and the probability of X is determined by some missing data z.
- For HMMs: θ is the transition/emission probabilities, X is the sequence data, and z represents the path through the model.

The EM Algorithm

- The goal is to find the model parameters θ that maximize the likelihood

$$\mathcal{P}(X|\theta)$$

$$\theta^* \leftarrow \text{argmax}_\theta \ln \mathcal{P}(X|\theta)$$

53-1

The EM Algorithm

- The goal is to find the model parameters θ that maximize the likelihood

$$\mathcal{P}(X|\theta)$$

'log' is a strictly increasing function, so θ that maximizes log P(x| θ) also maximizes P(x| θ).

So, the goal is to find

$$\theta^* \leftarrow \text{argmax}_\theta \ln \mathcal{P}(X|\theta)$$

53-2

The EM Algorithm

- The EM algorithm is an iterative procedure for maximizing

$$L(\theta) = \ln \mathcal{P}(X|\theta)$$

54

The EM Algorithm

- Assume that after the n^{th} iteration, the current estimator for θ is given by θ_n.
- Since the objective is to maximize $L(\theta)$, we seek a new estimator θ to maximize the difference

$$L(\theta) - L(\theta_n) = \ln \mathcal{P}(\mathbf{X}|\theta) - \ln \mathcal{P}(\mathbf{X}|\theta_n)$$

55

The EM Algorithm

- When there is missing/hidden data Z, the total probability $P(X|\theta)$ can be written as

$$\mathcal{P}(\mathbf{X}|\theta) = \sum_{\mathbf{z}} \mathcal{P}(\mathbf{X}|\mathbf{z}, \theta)\mathcal{P}(\mathbf{z}|\theta)$$

56

The EM Algorithm

- Using the formula for the total probability while accounting for missing/hidden data, we seek to maximize

$$L(\theta) - L(\theta_n) = \ln \sum_{\mathbf{z}} \mathcal{P}(\mathbf{X}|\mathbf{z}, \theta)\mathcal{P}(\mathbf{z}|\theta) - \ln \mathcal{P}(\mathbf{X}|\theta_n)$$

57-1

The EM Algorithm

- Using the formula for the total probability while accounting for missing/hidden data, we seek to maximize

$$L(\theta) - L(\theta_n) = \ln \sum_{\mathbf{z}} \mathcal{P}(\mathbf{X}|\mathbf{z},\theta)\mathcal{P}(\mathbf{z}|\theta) - \ln \mathcal{P}(\mathbf{X}|\theta_n)$$

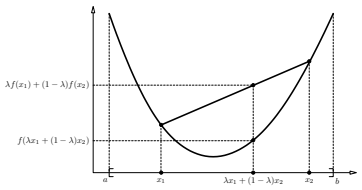
logarithm of sum

57-2

The EM Algorithm

Definition 1 *Let f be a real valued function defined on an interval $I = [a, b]$. f is said to be* convex *on I if $\forall x_1, x_2 \in I, \lambda \in [0, 1]$,*

$$f(\lambda x_1 + (1-\lambda)x_2) \leq \lambda f(x_1) + (1-\lambda) f(x_2).$$



58

The EM Algorithm

- Jensen's Inequality: Let f be a convex function defined on an interval B. If $x_1, x_2, \ldots, x_n \in B$ and $\lambda_1, \lambda_2, \ldots, \lambda_n \geq 0$ with $\Sigma\lambda_i = 1$, then

$$f\left(\sum_{i=1}^{n} \lambda_i x_i\right) \leq \sum_{i=1}^{n} \lambda_i f(x_i)$$

59

The EM Algorithm

- -ln x is strictly convex on (0,∞).
- Therefore,

$$-\ln \sum_{i=1}^{n} \lambda_i x_i \leq \sum_{i=1}^{n} \lambda_i (-\ln(x_i))$$

- Equivalently,

$$\ln \sum_{i=1}^{n} \lambda_i x_i \geq \sum_{i=1}^{n} \lambda_i \ln(x_i)$$

60

The EM Algorithm

- Back to...

$$L(\theta) - L(\theta_n) = \ln \sum_{\mathbf{z}} \mathcal{P}(\mathbf{X}|\mathbf{z}, \theta)\mathcal{P}(\mathbf{z}|\theta) - \ln \mathcal{P}(\mathbf{X}|\theta_n)$$

61-1

The EM Algorithm

- Back to...

$$L(\theta) - L(\theta_n) = \ln \sum_{\mathbf{z}} \mathcal{P}(\mathbf{X}|\mathbf{z}, \theta)\mathcal{P}(\mathbf{z}|\theta) - \ln \mathcal{P}(\mathbf{X}|\theta_n)$$

to use Jensen's inequality, we need to identify the constants (the λ_i's)

61-2

The EM Algorithm

* Back to...

$$L(\theta) - L(\theta_n) = \ln \sum_{\mathbf{z}} \mathcal{P}(\mathbf{X}|\mathbf{z},\theta)\mathcal{P}(\mathbf{z}|\theta) - \ln \mathcal{P}(\mathbf{X}|\theta_n)$$

to use Jensen's inequality, we need to identify the constants (the λ_i's)

If we consider constants of the form $P(z|X,\theta_n)$, we know that

$$\sum_{\mathbf{z}} \mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n) = 1$$

61-3

The EM Algorithm

$$\begin{aligned}
L(\theta) - L(\theta_n) &= \ln \sum_{\mathbf{z}} \mathcal{P}(\mathbf{X}|\mathbf{z},\theta)\mathcal{P}(\mathbf{z}|\theta) - \ln \mathcal{P}(\mathbf{X}|\theta_n) \\
&= \ln \sum_{\mathbf{z}} \mathcal{P}(\mathbf{X}|\mathbf{z},\theta)\mathcal{P}(\mathbf{z}|\theta) \cdot \frac{\mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n)}{\mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n)} - \ln \mathcal{P}(\mathbf{X}|\theta_n) \\
&= \ln \sum_{\mathbf{z}} \mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n) \left(\frac{\mathcal{P}(\mathbf{X}|\mathbf{z},\theta)\mathcal{P}(\mathbf{z}|\theta)}{\mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n)} \right) - \ln \mathcal{P}(\mathbf{X}|\theta_n) \\
&\geq \sum_{\mathbf{z}} \mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n) \ln \left(\frac{\mathcal{P}(\mathbf{X}|\mathbf{z},\theta)\mathcal{P}(\mathbf{z}|\theta)}{\mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n)} \right) - \ln \mathcal{P}(\mathbf{X}|\theta_n) \\
&= \sum_{\mathbf{z}} \mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n) \ln \left(\frac{\mathcal{P}(\mathbf{X}|\mathbf{z},\theta)\mathcal{P}(\mathbf{z}|\theta)}{\mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n)\mathcal{P}(\mathbf{X}|\theta_n)} \right) \\
&\triangleq \Delta(\theta|\theta_n)
\end{aligned}$$

62-1

The EM Algorithm

$$\begin{aligned}
L(\theta) - L(\theta_n) &= \ln \sum_{\mathbf{z}} \mathcal{P}(\mathbf{X}|\mathbf{z},\theta)\mathcal{P}(\mathbf{z}|\theta) - \ln \mathcal{P}(\mathbf{X}|\theta_n) \\
&= \ln \sum_{\mathbf{z}} \mathcal{P}(\mathbf{X}|\mathbf{z},\theta)\mathcal{P}(\mathbf{z}|\theta) \cdot \frac{\mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n)}{\mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n)} - \ln \mathcal{P}(\mathbf{X}|\theta_n) \\
&= \ln \sum_{\mathbf{z}} \mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n) \left(\frac{\mathcal{P}(\mathbf{X}|\mathbf{z},\theta)\mathcal{P}(\mathbf{z}|\theta)}{\mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n)} \right) - \ln \mathcal{P}(\mathbf{X}|\theta_n) \\
&\geq \sum_{\mathbf{z}} \mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n) \ln \left(\frac{\mathcal{P}(\mathbf{X}|\mathbf{z},\theta)\mathcal{P}(\mathbf{z}|\theta)}{\mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n)} \right) - \ln \mathcal{P}(\mathbf{X}|\theta_n) \\
&= \sum_{\mathbf{z}} \mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n) \ln \left(\frac{\mathcal{P}(\mathbf{X}|\mathbf{z},\theta)\mathcal{P}(\mathbf{z}|\theta)}{\mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n)\mathcal{P}(\mathbf{X}|\theta_n)} \right) \\
&\triangleq \Delta(\theta|\theta_n)
\end{aligned}$$

$$L(\theta) \geq L(\theta_n) + \Delta(\theta|\theta_n)$$

62-2

The EM Algorithm

$$
\begin{aligned}
L(\theta) - L(\theta_n) &= \ln \sum_{\mathbf{z}} \mathcal{P}(\mathbf{X}|\mathbf{z},\theta)\mathcal{P}(\mathbf{z}|\theta) - \ln \mathcal{P}(\mathbf{X}|\theta_n) \\
&= \ln \sum_{\mathbf{z}} \mathcal{P}(\mathbf{X}|\mathbf{z},\theta)\mathcal{P}(\mathbf{z}|\theta) \cdot \frac{\mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n)}{\mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n)} - \ln \mathcal{P}(\mathbf{X}|\theta_n) \\
&= \ln \sum_{\mathbf{z}} \mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n) \left(\frac{\mathcal{P}(\mathbf{X}|\mathbf{z},\theta)\mathcal{P}(\mathbf{z}|\theta)}{\mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n)} \right) - \ln \mathcal{P}(\mathbf{X}|\theta_n) \\
&\geq \sum_{\mathbf{z}} \mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n) \ln \left(\frac{\mathcal{P}(\mathbf{X}|\mathbf{z},\theta)\mathcal{P}(\mathbf{z}|\theta)}{\mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n)} \right) - \ln \mathcal{P}(\mathbf{X}|\theta_n) \\
&= \sum_{\mathbf{z}} \mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n) \ln \left(\frac{\mathcal{P}(\mathbf{X}|\mathbf{z},\theta)\mathcal{P}(\mathbf{z}|\theta)}{\mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n)\mathcal{P}(\mathbf{X}|\theta_n)} \right) \\
&\triangleq \Delta(\theta|\theta_n)
\end{aligned}
$$

$L(\theta) \geq L(\theta_n) + \Delta(\theta|\theta_n)$

$l(\theta|\theta_n)$

62-3

The EM Algorithm

$$
\begin{aligned}
L(\theta) - L(\theta_n) &= \ln \sum_{\mathbf{z}} \mathcal{P}(\mathbf{X}|\mathbf{z},\theta)\mathcal{P}(\mathbf{z}|\theta) - \ln \mathcal{P}(\mathbf{X}|\theta_n) \\
&= \ln \sum_{\mathbf{z}} \mathcal{P}(\mathbf{X}|\mathbf{z},\theta)\mathcal{P}(\mathbf{z}|\theta) \cdot \frac{\mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n)}{\mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n)} - \ln \mathcal{P}(\mathbf{X}|\theta_n) \\
&= \ln \sum_{\mathbf{z}} \mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n) \left(\frac{\mathcal{P}(\mathbf{X}|\mathbf{z},\theta)\mathcal{P}(\mathbf{z}|\theta)}{\mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n)} \right) - \ln \mathcal{P}(\mathbf{X}|\theta_n) \\
&\geq \sum_{\mathbf{z}} \mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n) \ln \left(\frac{\mathcal{P}(\mathbf{X}|\mathbf{z},\theta)\mathcal{P}(\mathbf{z}|\theta)}{\mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n)} \right) - \ln \mathcal{P}(\mathbf{X}|\theta_n) \\
&= \sum_{\mathbf{z}} \mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n) \ln \left(\frac{\mathcal{P}(\mathbf{X}|\mathbf{z},\theta)\mathcal{P}(\mathbf{z}|\theta)}{\mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n)\mathcal{P}(\mathbf{X}|\theta_n)} \right) \\
&\triangleq \Delta(\theta|\theta_n)
\end{aligned}
$$

$L(\theta) \geq L(\theta_n) + \Delta(\theta|\theta_n)$

$l(\theta|\theta_n)$

$$L(\theta) \geq l(\theta|\theta_n)$$

62-4

The EM Algorithm

* So, we have

$$L(\theta) \geq l(\theta|\theta_n)$$

and

$$
\begin{aligned}
l(\theta_n|\theta_n) &= L(\theta_n) + \Delta(\theta_n|\theta_n) \\
&= L(\theta_n) + \sum_{\mathbf{z}} \mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n) \ln \frac{\mathcal{P}(\mathbf{X}|\mathbf{z},\theta_n)\mathcal{P}(\mathbf{z}|\theta_n)}{\mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n)\mathcal{P}(\mathbf{X}|\theta_n)} \\
&= L(\theta_n) + \sum_{\mathbf{z}} \mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n) \ln \frac{\mathcal{P}(\mathbf{X},\mathbf{z}|\theta_n)}{\mathcal{P}(\mathbf{X},\mathbf{z}|\theta_n)} \\
&= L(\theta_n) + \sum_{\mathbf{z}} \mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n) \ln 1 \\
&= L(\theta_n),
\end{aligned}
$$

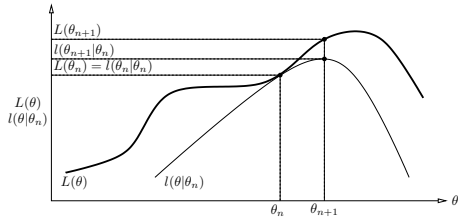
63

The EM Algorithm

- In other words, the function $l(\theta|\theta_n)$ is
  - bounded from above by the likelihood $L(\theta)$, and
  - equals the likelihood $L(\theta_n)$ when $\theta=\theta_n$.
- Therefore, any θ that increases $l(\theta|\theta_n)$ in turn increases the likelihood $L(\theta)$.
- So, the EM Algorithm finds θ_{n+1} that maximizes $l(\theta|\theta_n)$

64

The EM Algorithm



65

The EM Algorithm

$$
\begin{aligned}
\theta_{n+1} &= \arg\max_\theta \{l(\theta|\theta_n)\} \\
&= \arg\max_\theta \left\{ L(\theta_n) + \sum_{\mathbf{z}} \mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n) \ln \frac{\mathcal{P}(\mathbf{X}|\mathbf{z},\theta)\mathcal{P}(\mathbf{z}|\theta)}{\mathcal{P}(\mathbf{X}|\theta_n)\mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n)} \right\} \\
&\quad \text{Now drop terms which are constant w.r.t. } \theta \\
&= \arg\max_\theta \left\{ \sum_{\mathbf{z}} \mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n) \ln \mathcal{P}(\mathbf{X}|\mathbf{z},\theta)\mathcal{P}(\mathbf{z}|\theta) \right\} \\
&= \arg\max_\theta \left\{ \sum_{\mathbf{z}} \mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n) \ln \frac{\mathcal{P}(\mathbf{X},\mathbf{z},\theta)}{\mathcal{P}(\mathbf{z},\theta)} \frac{\mathcal{P}(\mathbf{z},\theta)}{\mathcal{P}(\theta)} \right\} \\
&= \arg\max_\theta \left\{ \sum_{\mathbf{z}} \mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n) \ln \mathcal{P}(\mathbf{X},\mathbf{z}|\theta) \right\} \\
&= \arg\max_\theta \left\{ \mathrm{E}_{\mathbf{z}|\mathbf{X},\theta_n} \{\ln \mathcal{P}(\mathbf{X},\mathbf{z}|\theta)\} \right\}
\end{aligned}
$$

66-1

The EM Algorithm

$$
\begin{aligned}
\theta_{n+1} &= \arg\max_{\theta} \{l(\theta|\theta_n)\} \\
&= \arg\max_{\theta} \left\{ L(\theta_n) + \sum_{\mathbf{z}} \mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n) \ln \frac{\mathcal{P}(\mathbf{X}|\mathbf{z},\theta)\mathcal{P}(\mathbf{z}|\theta)}{\mathcal{P}(\mathbf{X}|\theta_n)\mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n)} \right\} \\
&\quad \text{Now drop terms which are constant w.r.t. } \theta \\
&= \arg\max_{\theta} \left\{ \sum_{\mathbf{z}} \mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n) \ln \mathcal{P}(\mathbf{X}|\mathbf{z},\theta)\mathcal{P}(\mathbf{z}|\theta) \right\} \\
&= \arg\max_{\theta} \left\{ \sum_{\mathbf{z}} \mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n) \ln \frac{\mathcal{P}(\mathbf{X},\mathbf{z},\theta)}{\mathcal{P}(\mathbf{z},\theta)} \frac{\mathcal{P}(\mathbf{z},\theta)}{\mathcal{P}(\theta)} \right\} \\
&= \arg\max_{\theta} \left\{ \sum_{\mathbf{z}} \mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n) \ln \mathcal{P}(\mathbf{X},\mathbf{z}|\theta) \right\} \\
&= \arg\max_{\theta} \{ \mathrm{E}_{\mathbf{z}|\mathbf{X},\theta_n} \{\ln \mathcal{P}(\mathbf{X},\mathbf{z}|\theta)\} \}
\end{aligned}
$$

expectation

66-2

The EM Algorithm

$$
\begin{aligned}
\theta_{n+1} &= \arg\max_{\theta} \{l(\theta|\theta_n)\} \\
&= \arg\max_{\theta} \left\{ L(\theta_n) + \sum_{\mathbf{z}} \mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n) \ln \frac{\mathcal{P}(\mathbf{X}|\mathbf{z},\theta)\mathcal{P}(\mathbf{z}|\theta)}{\mathcal{P}(\mathbf{X}|\theta_n)\mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n)} \right\} \\
&\quad \text{Now drop terms which are constant w.r.t. } \theta \\
&= \arg\max_{\theta} \left\{ \sum_{\mathbf{z}} \mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n) \ln \mathcal{P}(\mathbf{X}|\mathbf{z},\theta)\mathcal{P}(\mathbf{z}|\theta) \right\} \\
&= \arg\max_{\theta} \left\{ \sum_{\mathbf{z}} \mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n) \ln \frac{\mathcal{P}(\mathbf{X},\mathbf{z},\theta)}{\mathcal{P}(\mathbf{z},\theta)} \frac{\mathcal{P}(\mathbf{z},\theta)}{\mathcal{P}(\theta)} \right\} \\
&= \arg\max_{\theta} \left\{ \sum_{\mathbf{z}} \mathcal{P}(\mathbf{z}|\mathbf{X},\theta_n) \ln \mathcal{P}(\mathbf{X},\mathbf{z}|\theta) \right\} \\
&= \arg\max_{\theta} \{ \mathrm{E}_{\mathbf{z}|\mathbf{X},\theta_n} \{\ln \mathcal{P}(\mathbf{X},\mathbf{z}|\theta)\} \}
\end{aligned}
$$

maximization

expectation

66-3

The EM Algorithm and HMMs

- If π denotes a state path, we want to maximize

$$\log P(x|\theta) = \sum_{\pi} \log P(x,\pi|\theta)$$

67

The EM Algorithm and HMMs

68

* The Q function (as the expectation function is often referred to in the context of EM) is given by

$$Q(\theta|\theta^t) = \sum_{\pi} P(\pi|x, \theta^t) \log P(x, \pi|\theta)$$

(using θ^t instead of θ_n)

The EM Algorithm and HMMs

69

* Denoting by $A_{kl}(\pi)$ and $E_k(b,\pi)$ the number of times transition k to l is observed for path π and number of times character b is observed in state k for path π, respectively, then

$$P(x, \pi|\theta) = \prod_{k=1}^{M} \prod_{b} [e_k(b)]^{E_k(b,\pi)} \prod_{k=0}^{M} \prod_{l=1}^{M} [a_{kl}]^{A_{kl}(\pi)}$$

The EM Algorithm and HMMs

70-1

* Taking the logarithm, we have

$$Q(\theta|\theta^t) = \sum_{\pi} P(\pi|x, \theta^t) \times \left[\sum_{k=1}^{M} \sum_{b} E_k(b, \pi) \log e_k(b) + \sum_{k=0}^{M} \sum_{l=1}^{M} A_{kl}(\pi) \log a_{kl} \right]$$

The EM Algorithm and HMMs

70-2

* Taking the logarithm, we have

$$Q(\theta|\theta^t) = \boxed{\sum_{\pi} P(\pi|x,\theta^t)} \times \left[\sum_{k=1}^{M} \sum_{b} \boxed{E_k(b,\pi)} \log e_k(b) + \sum_{k=0}^{M} \sum_{l=1}^{M} \boxed{A_{kl}(\pi)} \log a_{kl} \right]$$

$E_k(b)$ A_{kl}

The EM Algorithm and HMMs

71

$$Q(\theta|\theta^t) = \sum_{k=1}^{M} \sum_{b} E_k(b) \log e_k(b) + \sum_{k=0}^{M} \sum_{l=1}^{M} A_{kl} \log a_{kl}$$

The EM Algorithm and HMMs

72

* To maximize Q:

(1) $$\boxed{a_{k\ell} = \frac{A_{k\ell}}{\sum_{\ell'} A_{k\ell'}} \qquad e_k(b) = \frac{E_k(b)}{\sum_{b'} E_k(b')}}$$

Estimating HMM Probabilities: Unknown State Sequences

73

- The Baum-Welch algorithm, which is an expectation-maximization (EM) algorithm
- Informally, the algorithm first estimates the A_{kl} and $E_k(b)$ by considering probable paths for the training sequences using the current values of a_{kl} and $e_k(b)$
- Then, new values of the as and es are derived using the equations on the previous slide
- This process is iterated until some stopping criterion is reached

The Baum-Welch Algorithm

74

- It is possible to show that the overall log likelihood of the model is increased by the iteration, and hence that the process will converge to a local maximum
- Unfortunately, there are usually many local maxima, and which one you end up with depends strongly on the starting values of the parameters
- The problem of local maxima is particularly severe when estimating large HMMs

The Baum-Welch Algorithm

75

- More formally, the Baum-Welch algorithm calculates A_{kl} and $E_k(b)$ as the expected number of times each transition or emission is used, given the training sequences
- To do this, it uses the forward and backward values

The Baum-Welch Algorithm

76

* The probability that transition "k to l" is used at position i in sequence x is

$$\mathbf{P}(\pi_i = k, \pi_{i+1} = \ell | x, \theta) = \frac{f_k(i) a_{k\ell} e_\ell(x_{i+1}) b_\ell(i+1)}{\mathbf{P}(x)}$$

The Baum-Welch Algorithm

77

* From this we derive the expected number of times that transition "k to l" is used by summing over all positions and over all training data sets

(2) $$A_{k\ell} = \sum_j \frac{1}{\mathbf{P}(x^j)} \sum_i f_k^j(i) a_{k\ell} e_\ell(x_{i+1}^j) b_\ell^j(i+1)$$

(f^j and b^j are the forward and backward values for sequence x^j)

The Baum-Welch Algorithm

78

* Similarly, we can find the expected number of times that letter b appears in state k

(3) $$E_k(b) = \sum_j \frac{1}{\mathbf{P}(x^j)} \sum_{\{i | x_i^j = b\}} f_k^j(i) b_k^j(i)$$

The Baum-Welch Algorithm

79

- Having calculated these expectations, the new model parameters are calculated just as before, using (1)
- We can iterate using the new values of the parameters to obtain new values of the As and Es as before, but in this case we are converging in a continuous-values space, and so will never in fact reach the maximum
- It is therefore necessary to set a convergence criterion, typically stopping when the change in total log likelihood is sufficiently small

The Baum-Welch Algorithm

80

1. Initialization: Pick arbitrary model parameters (θ)
2. Recurrence:
    1. Set all the A and E variables to their pseudocount values r (or to zero)
    2. For each sequence j=1,...,n
        1. Calculate $f_k(i)$ for sequence j using the forward algorithm
        2. Calculate $b_k(i)$ for sequence j using the backward algorithm
        3. Add the contribution of sequence j to A (2) and E (3)
    3. Calculate the new model parameters using (1)
    4. Calculate the new log likelihood of the model
3. Termination: Stop if the change in the log likelihood is less than some predefined threshold or the maximum number of iterations is exceeded

Viterbi Training

81

- An alternative to the Baum-Welch algorithm is frequently used, which is called Viterbi training
- In this approach, the most probable paths for the training sequences are derived using the Viterbi algorithm, and these are used in the re-estimation process
- Again, the process is iterated when the new parameter values are obtained
- In this case, the algorithm converges precisely, because the assignment of paths is a discrete process, and we can continue until none of the paths change
- At this point the parameter estimates will not change either, because they are determined completely by the paths

Viterbi Training

82

- Unlike Baum-Welch, this procedure does not maximize the true likelihood (the probability of the sequences, given the parameters)
- Instead, it finds the value of θ that maximizes the contribution to the likelihood $P(x^1,\ldots,x^n|\theta,\pi^*(x^1),\ldots,\pi^*(x^n))$ from the most probable paths for all the sequences
- This is a probable reason for why Viterbi training performs less well in general than Baum-Welch
- However, it is widely used, and it can be argued that when the primary use of the HMM is to produce decodings via Viterbi alignments, then it is good to train using them

Acknowledgments

83

- In addition to Durbin et al.'s "Biological Sequence Analysis", materials are also based on
  - "The Expectation Maximization Algorithm: A Short Tutorial" by Sean Borman

Questions?

84