Lecture 7

Hidden Markov Models

Additional Reference

- *Biological Sequence Analysis* by R. Durbin, S. Eddy, A. Krogh, and G. Mitchison

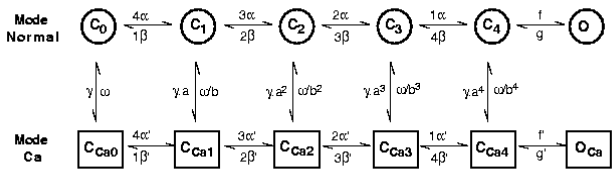
Uses of Hidden Markov Models

- Modeling stochastic processes
- Sequence alignment
- Phylogenetic tree construction
- Microarray data analysis (clustering)
- Protein secondary structure prediction
- RNA secondary structure prediction
- Ion channel modeling

Markov Model

- A process is Markov if it has no memory, that is, if the next state it assumes, depends only on its present state and not on any previous states.
- The states can be observed and the transition probabilities between states is known
- Example – rolling a die has 6 possible states each with a probability of 1/6

Markov Model – Example 2
Ion Channel



From Jafri et al. 1998

- Conversion between states determined by transition probabilities.
- Multiple states to reflect properties of channel.

Markov Model – Example 3
CpG Islands

- In the human genome, the dinucleotide CG occurs (called CpG) is often methylated.
- The methylated C often mutates into a T resulting in a lower frequency of CpG dinucleotides than would be expected.
- In regions such as the start region of many genes, the methylation process seems to be suppressed resulting in a higer frequency of CG dinucleotides.
- These are called CpG Island.

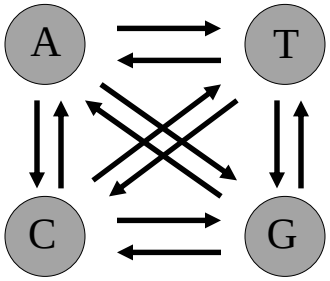
CpG Islands

- Hence, the CpG Island might be a good indicator of the start of a gene.
- This leads to two questions:
  - How can we tell if a region is a CpG Island or not?
  - How do we identify CpG Islands in a long sequence of DNA?

CpG Islands

- We can create a Markov model to generate the CpG Island regions starting with a Markov model that generates a DNA sequence (Markov chain).
- On the arrows are transition probabilities:

$a_{st}=P(x_i=t|x_{i-1}=s)$



DNA sequence model

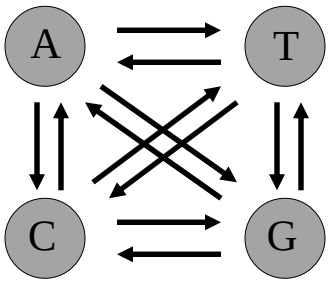
- The probability of the sequence is

$P(x) = P(x_L, x_{L-1}, \ldots, x_1)$

$= P(x_L \mid x_{L-1}, \ldots, x_1) P(x_{L-1} \mid x_{L-2}, \ldots, x_1) \ldots P(x_1)$

$= P(x_L \mid x_{L-1}) P(x_{L-1} \mid x_{L-2}) \ldots P(x_1)$ [Why?]

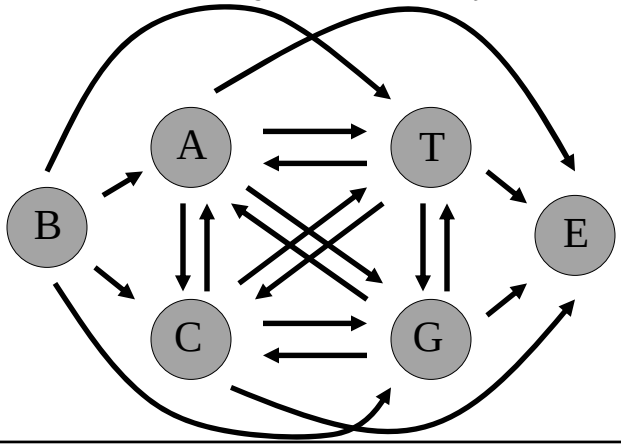
$= P(x_1) \Pi_{i=1}^{l} a_{x_{i-1} x_i}$



DNA Sequence Model

- We can model the beginning and end of the sequences by adding a beginning state and end state.

with $P(x_1 = s) = a_{Bs}$ and $P(E \mid x_L = a_{tE})$



Terminology

- The state sequence is called the path π.
- The i^{th} state in the path is called π_i.
- The chain is characterized by parameters called transition probabilities

  $a_{kl} = P(\pi_i=l| \pi_i=k)$
- The transition probability a_{ok} from the begin state to k can be thought of as the probability of starting in state k.
- In addition to having different states, the chain consists of symbols b. There is an emission probability $e_k(b) = P(x_i=b| \pi_i=k)$.

Example

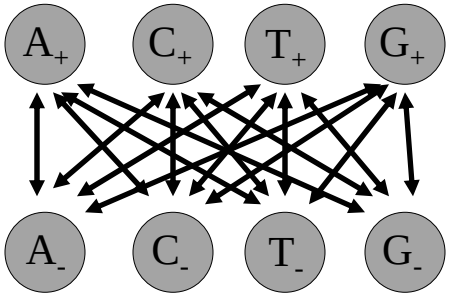
Assume three coins are used each with a different bias for heads. The first coin is fair with P(H)=0.5, the second coin has P(H)=0.75 and the third coin has P(H)=0.1. Also assume that the first coin is chosen at random. If the first coin is chosen than either the second or third is chosen next with equal probability. If the second or third coin is chosen, any of the three are chosen next with equal probability.

$$\pi = \begin{pmatrix} 1/3 \\ 1/3 \\ 1/3 \end{pmatrix} \quad a = \begin{pmatrix} 0 & 1/2 & 1/2 \\ 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \end{pmatrix} \quad b = \begin{pmatrix} 0.5 & 0.5 \\ 0.75 & 0.25 \\ 0.1 & 0.9 \end{pmatrix}$$

initial probability transition probability emission probability

Example - CpG Islands

- We can create a hidden Markov model to generate the CpG Island regions.
- The "+" states are for nucleotides found in the CpG Islands.
- The "-" states are nucleotides not found in the CpG Islands.
- There is a complete set of transitions within each set of states.



Example - CpG Islands

- Using the hidden Markov Model we can compute the transition probabilities by

$$a_{st}^{+} = \frac{c_{st}^{+}}{\sum_{t'} c_{st'}^{+}}$$

+	A	C	G	T
A	0.180	0.274	0.426	0.120
C	0.171	0.368	0.274	0.188
G	0.161	0.339	0.375	0.125
T	0.079	0.355	0.384	0.182

-	A	C	G	T
A	0.300	0.205	0.285	0.210
C	0.322	0.298	0.078	0.302
G	0.248	0.246	0.298	0.208
T	0.177	0.239	0.292	0.292

Example - CpG Islands

- For discrimination, the log liklihood ratio is calculated by

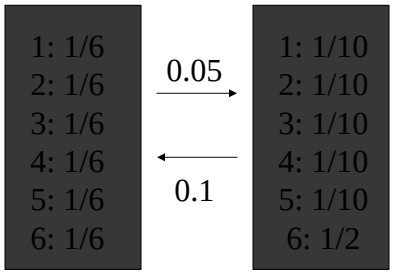
$$S(x) = \log \frac{P(x|+)}{P(x|-)} = \sum_{i=1}^{L} \log \frac{a^{+}_{x_{i-1}x_i}}{a^{-}_{x_{i-1}x_i}} = \sum_{i=1}^{L} \beta_{x_{i-1}x_i}$$

β/L	A	C	G	T
A	-0.740	0.419	0.580	-0.803
C	-0.913	0.302	1.812	-0.685
G	-0.624	0.461	0.331	-0.730
T	-1.169	0.573	0.393	0.679

The occasionally dishonest casino

- Uses a fair die most of the time but switches to a loaded die
- The loaded die has probability of 0.5 of a 6 and 0.1 for the other outcomes
- The fair to loaded transition probability is 0.05.
- The loaded to fair transition probability is 0.1

The Occasionally Dishonest Casino



The Occasionally Dishonest Casino

- This is a hidden Markov model because, while we can see the rolls of the die, we do not know which rolls are with a fair die and which rolls are with a loaded die.
- All we can see are the die rolls that are emitted by the model.
- Hence we do not know the sequence of states.
- The joint probability of an observed sequence x and a state sequence π is

$P(x, \pi) = a_{0\pi^1} \Pi^l_{i=1} e_{\pi i} (x_i) a_{\pi i+1 \pi 1}$

Most Probable State Path

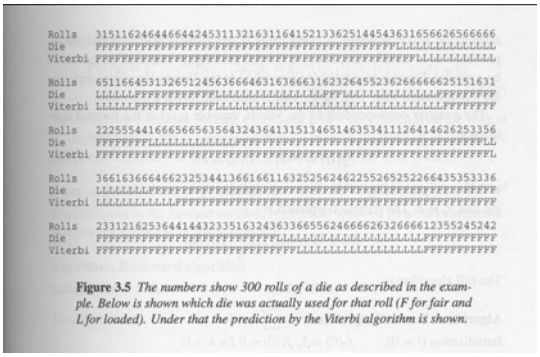


Figure 3.5 *The numbers show 300 rolls of a die as described in the example. Below is shown which die was actually used for that roll (F for fair and L for loaded). Under that the prediction by the Viterbi algorithm is shown.*

From Durbin et al

Given a sequence of emissions, we want to find the most probable sequence of states that would yield the emitted sequence.

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

Viterbi Algorithm

Suppose that the probability $v_k(i)$ (the Viterbi variable) of the most probably path ending in state k with observation i is known for all states k.

Then these probabilities can be calculated for observation x_{i+1} as

$$v_l(i+1) = e_l(x_{i+1}) \max_k(v_k(i)\, a_{kl})$$

All paths start in the begin state so $v_0(0) = 1$.

Viterbi Algorithm

Initialization: (i=0): $v_0(0)=1$, $v_k(0)=0$ for $k > 0$

Recursion: (i=1…L): $v_l(i)=e_l(x_i)\max_k v_k(i-1)a_{kl}$);
$ptr_i=\text{argmax}_k v_k(i-1)a_{kl}$).

Termination: $P(x,\pi^*)=\max_k(v_k(L)a_{k0})$;
$\pi^*_L=\text{argmax}_k(v_k(L)a_{k0})$.

Traceback (i=L…1): $\pi^*_{i-1}=ptr_i(\pi^*_i)$.

Viterbi Algorithm

- Multiplying many probabilities together results in very small numbers that will give underflow.
- This and other algorithms should be done in log space ($\log(v_l(i))$ so that the products become sums and the numbers stay reasonable.

The Forward Algorithm

- The number of paths π increases exponentially with the length of the sequence.
- The forward algorithm efficiently calculates the probability of a sequence by assuming the most probably path π^* is the only path with significant probability.
- The probability of the observed sequence up to and including x_i with $\pi_i=k$ is

  $f_k(i)=P(x_1 \ldots x_i, \pi_i=k)$

  (the forward variable)
- The recursion equation is

  $f_l(i+1) = e_l(x_{i+1})\Sigma_k f_k(i)a_{kl}$

The Forward Algorithm

Initialization (i=0): $f_0(0)=1$, $f_k(0)=0$ for $k>0$

Recursion (i=1…L): $f_l(i) = e_l(x_i)\Sigma_k f_k(i-1)a_{kl}$

Termination: $P(x) = \Sigma_k f_k(L)a_{k0}$

Backward Algorithm and Posterior State Probabilities

- While the Viterbi algorithm finds the most probable path through the model, we might want to know what the most probable state is for an observation x_i.
- The probability that an observation x_i came from a state k given the observed sequence is the *posterior probability*. (i.e. $P(\pi_i=k|x)$)
- $P(\pi_i=k|x) = P(\pi_i=k, x) /P(x)$
- $P(\pi_i=k, x) = f_k(i)\, b_k(i)$

  where $b_k(i)=P(x_{i+1}\ldots x_L| \pi_i=k)$

  (the backward variable)

The Backward Algorithm

Initialization (i=L): $b_k(L)=a_{k0}$ for all k

Recursion (i=L-1…1): $b_l(i) = \Sigma_l\, a_{kl} e_l(x_{i+1})\, b_k(i+1)$

Termination: $P(x) = \Sigma_l\, a_{0l}\, e_l(x_1)\, b_l(1)$

Posterior Probabilities

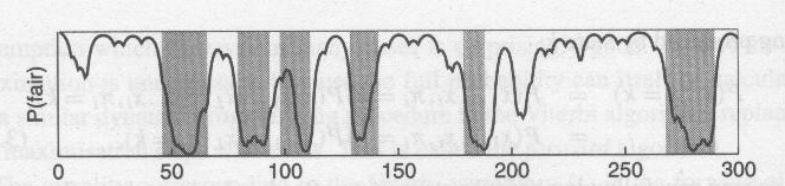


Figure 3.6 *The posterior probability of being in the state corresponding to the fair die in the casino example. The x axis shows the number of the roll. The shaded areas show when the roll was generated by the loaded die.*

From Durbin et al

Posterior Decoding

- The posterior probability is useful for two different forms of decoding (in additiion the Viterbi decoding discussed previously).
- These are used when there are more than one path that has similar probability as the most probable one.
- One approach is to use an alternative path to look at a state assignment at a particular point i.

Posterior Decoding

- The second approach can be used when something other than the state sequence might be of interest.
- For example, supposed that the probability of switching from fair to loaded is 0.01 (no switch 0.99).
- The Viterbi algorithm does not visit the loaded die state, but posterior decoding can determine when the loaded die state might be visited.

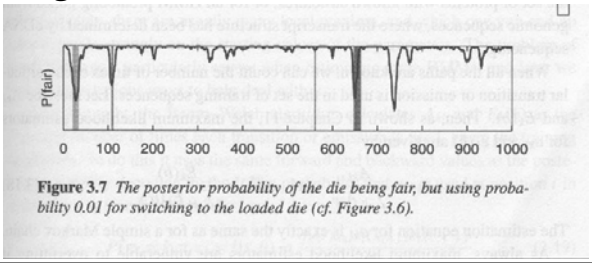


Figure 3.7 *The posterior probability of the die being fair, but using probability 0.01 for switching to the loaded die (cf. Figure 3.6).*

From Durbin et al

Parameter Estimation for HMMs

- In the example given, the initial, transition and emission probabilities are known.
- In specifying a model to describe data, these are usually unknown.
- Furthermore, the structure of the HMM is also unknown.

Training and Testing the HMM

- The parameters of the model are fit on a training set, ie., the parameters are chosen so that the training set is the most likely outcome for the model.
- A test set is used to make sure the model is well-trained.
- If so, the model can be used on new data.

Parameter Estimation when the state sequence is known

- If we know all the paths, we can count the number times a particular transition or emission occurs out of the total number of times that it was possible to get a maximum likelihood estimate for the transition and emission probabilities.

$$a_{kl} = A_{kl} / \Sigma_{l'} A_{kl'}$$
$$e_k(b) = E_k(b) / \Sigma_{b'} E_k(b')$$

- This can be proven to be the MLE.

Estimation when the paths are unknown

- In this case there is no closed form estimate for the transition and emission probabilities.
- Instead, an iterative method must be used to estimated the values for the transition and emission probabilities using current values.
- The new values replace the old values and the iteration continues until convergence occurs.
- This procedure is called the *Baum-Welch Algorithm.*

Baum-Welch Algorithm

- Calculates the transition and emission matric as the expected number of times each transition or emission is used given the training sequence.
- To do this, the forward and backward values are used.
- The probability that a_{kl} is used at position i in sequence x is

$$P(\pi_i=k,\pi_{i+1}=l|x,\theta) = f_k(i)a_{kl}e_l(x_{i+1})b_l(i+1)/P(x)$$

Baum-Welch Algorithm

The expected number of times a_{kl} is used is determined by summing over all positions and all training sequences

$$A_{kl} = \Sigma_j \, 1/P(x^j) \, \Sigma_i \, f^j_k(i) \, a_{kl} \, e_l(x^j_{i+1}) b^j_l(i+1)$$

Where f^j is the forward variable calculated for sequence j and b^j is the backward variable calculated for sequence j

The new model parameters are calculated by

$$a_{kl} = A_{kl} / \Sigma_{l'} \, A_{kl'}$$

Baum-Welch Algorithm

The expected number of times that the letter b appears in state k is

$$E_k(b) = \Sigma_j \, 1/P(x^j) \, \Sigma_i \, f^j_k(i) \, b^j_l(i)$$

Where the inner sum is only over positions I for which the symbol emitted is b.

The new model parameters are calculated by

$$e_k(b) = E_k(b) / \Sigma_{b'} E_k(b')$$

Baum-Welch Algorithm

Initialization: Pick arbitrary model parameters

Recurrence:

Set all the A and E variables to their pseudocount values r (or to zero)

For each sequence j = 1…n:

- Calculate $f_k(i)$ for sequence j using the forward algorithm
- Calculate $b_k(i)$ for sequence j using the backward algorithm
- Add the contribution of sequence j to A and E

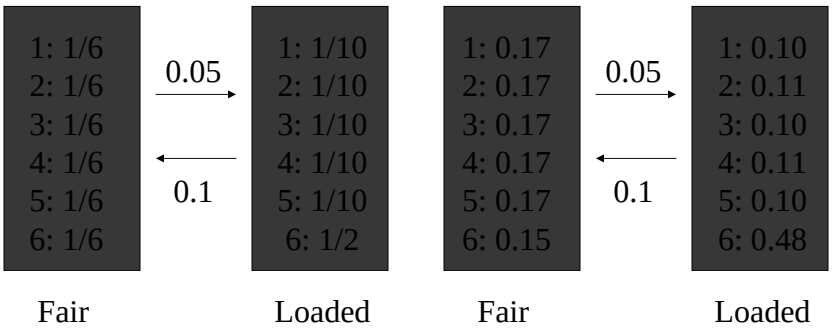
Calculate the new model parameters

Calculate the new log likelihood of the model

Termination: Stop if the change in the log likelihood is less than some predefined threshold or the maximum number of iterations is exceeded

Parameter Estimation with HMM

With hidden Markov models, we can calculate the most likely emission and transition probabilities from the sequence of outcomes.



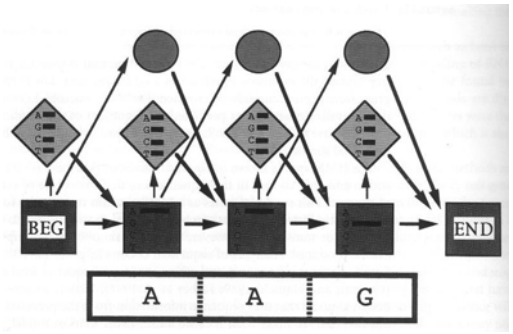
HMM Topology

- Thus far we have studied how to determine the unknown parameters for a model of know topology.
- Use knowledge about the process being described to decide the topology.
- Picking a very general topology and letting the model fit itself by reducing unused connections to low probability is not a good approach since the model gets caught in local maxima.

Silent States

- *Silent states* or *null states* are states that do not emit symbols. In the previous example the begin and end states were silent.
- These can be added anywhere in the model.

HMM of *E. Coli* Gene



match

insert

delete

From Mount

- HMM for finding the most probable set of genes in *E. coli* gene sequences of unknown gene composition.
- A similar model exists for each of the 61 codons

HMMs for Multiple Sequence Alignment

- 3n states where n is the average sequence length.
- N matching states along the backbone with and additional insertion or deletion state so that variable length sequences can be accommodated.
- The training set of sequences to be aligned is treated as a collection of observation sequences.
- Once the HMM is trained, each sequence from the training set can be scored using the Viterbi algorithm which gives rise to the path of matching, inserted, and deleted states.

HMMs for Multiple Sequence Alignment

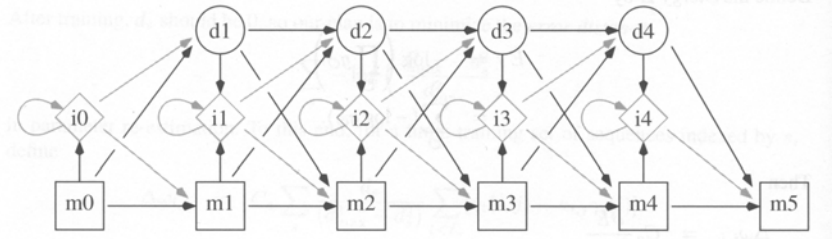


Figure 5.3 Linear HMM for multiple sequence alignment. Adapted from [DEKM98] and [Wat95] with permissions from Cambridge University Press and CRC Press, Boca Raton, Florida

From Clote and Backofen

- Sequence length 4
- 4 matching states, 5 insertion states, and 4 delete states.
- m0, m5, and the delete states are silent.

HMMs for Multiple Sequence Alignment

Consider the sequences GGCT, ACCGAT, and CT.

After convergence of Baum-Welch algorithm, the Viterbi path

GGCT	m0, m1, m2, m3, m4, m5
ACCGAT	m0, i0, m1, d2, m3, i3, i3, m4, m5
CT	m0, m1, d2, d3, m4, m5

This yields

a C -- C g a T	m0, i0, m1, d2, m3, i3, i3, m4, m5
. G G C . . T	m0, m1, m2, m3, m4, m5
. C -- -- . . T	m0, m1, d2, d3, m4, m5

Protein Motifs

- Mamitsuka used a HMM to identify sugar transport proteins with the PROSITE consensus sequence

[LIVMTSA]-[DE]-x-[LIVMFYWA]-G-R-[RK]-x(4,6)-G

- He compared training times and error distributiosn for 4 types of HMMS where each model had a different algorithm for parameter re-estimation.
- He used a target value for the calculated likelihood for postive and negative examples.
- Used two methods other than Baum-Welch to see if they worked better to improve sensitivity of the HMM.

Protein Motifs

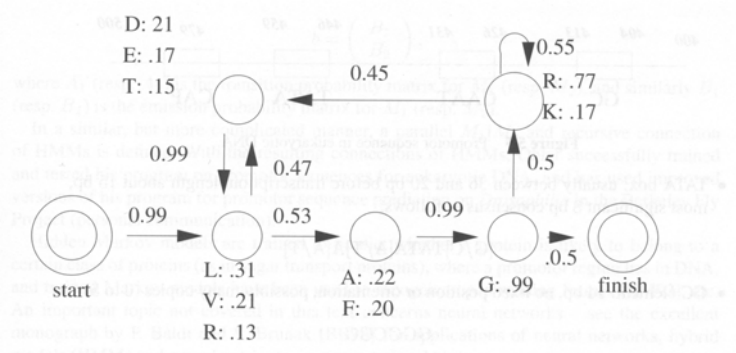


Figure 5.4 Mamitsuka's HMM for sugar transport proteins. Reprinted from [Mam96]. Copyright Mary Ann Liebert, Inc, New York.

Emission probabilities for various amino acids and Transition probabilities between the states are shown