Part IV. Genome Annotation:
2. Learning Hidden Markov models

Joint probability distribution
- Joint distribution over all $S = \{S_0, S_1, S_2, S_3, \ldots\}$ and $O = \{O_0, O_1, O_2, O_3, \ldots\}$
- $P(S, O) = P(S_0) P(S_1|S_0) P(O_1|S_0) P(S_2|S_1) P(O_2|S_1) P(O_3|S_2) \cdots$
  \[= P(S_0) \prod P(S_{i+1}|S_i) P(O_i|S_i) \cdots \]

Hidden Markov Models
- Two types of variables
  - $S$: Fair die or loaded die?
  - $O$: Each outcome of a die roll
- Conditional probabilities:
  - $P(S_i|S_{i-1}), P(O_i|S_i)$

Joint probability distribution
- Decoding problem
  - Find the most likely parse of a sequence
- Learning problem
  - Update the parameters of the model based on training data
Decoding

GIVEN \( O = O^1 O^2 \ldots O^N \)
Find \( S = S^1, \ldots, S^N \),
to maximize \( P(O, S) \)

\( S^* = \text{argmax}_S P(O, S) \)
Maximizes \( a_{S^1 S^2}(O^1) a_{S^2 S^3} \ldots a_{S^N S^1 N} e_{SN}(O^N) \)

**Dynamic Programming**
\( V_i(i) = \max_{S_1 \ldots S_i} P(O^1 \ldots O^i, S^1, \ldots, S^i, O^i, S^i = k) \)
= Prob. of most likely sequence of states ending at state \( S^i = k \)

Decoding – main idea

**Inductive assumption:** Given that for all states \( k \), and for a fixed position \( i \),

\[ V_i(i) = \max_{S_1 \ldots S_i} P(O^1 \ldots O^i, S^1, \ldots, S^i, O^i, S^i = k) \]

What is \( V_{i+1}(i+1) \)?

From definition,
\[ V_{i+1}(i+1) = \max_{S_1 \ldots S_{i+1}} P(O^1 \ldots O^{i+1}, S^1, \ldots, S^{i+1}, O^{i+1}, S^{i+1} = l) \]
= \( \max_{S_1 \ldots S_i} P(O^1 \ldots O^i, S^1, \ldots, S^i, O^i, S^i = l) \) \( \max_{S_{i+1}} P(O^{i+1}, S^{i+1} = l | S^i = l) \)
= \( \max_{S_1 \ldots S_i} \max_{S_{i+1}} P(O^1 \ldots O^i, S^1, \ldots, S^i, O^i, S^i = l) P(O^{i+1}, S^{i+1} = l | S^i = l) \)
= \( \max_{S_1 \ldots S_i} \max_{S_{i+1}} P(O^1 \ldots O^i, S^1, \ldots, S^i, O^i, S^i = l) \)
= \( \max_{S_1 \ldots S_i} \max_{S_{i+1}} \max_{S^{i+1}} P(O^1 \ldots O^i, S^1, \ldots, S^i, O^i, S^i = k) \)

= \( \max_{S_1 \ldots S_i} \max_{S_{i+1}} P(O^1 \ldots O^i, S^1, \ldots, S^i, O^i, S^i = k) \) \( V_i(i) \)
= \( e_i(a_{i+1}) \max_{S_{i+1}} \max_{S^{i+1}} V_{i+1}(i) \)

The Viterbi Algorithm

**State 1**

\( O^1 \quad O^2 \quad O^3 \quad \ldots \quad O^N \)

**State 2**

\( K \)

**State 3**

Similar to “aligning” a set of states to a sequence

**Time:** \( O(K^2 N) \)

**Space:** \( O(KN) \)
The Viterbi Algorithm

**Input:** \( O = O^1, O^2, \ldots, O^N \)

**Initialization:**
- \( V_i(0) = 1 \) (0 is the imaginary first position)
- \( V_i(0) = 0, \) for all \( k > 0 \)

**Iteration:**
- \( V_j(i) = e_j(O) \times \max_k a_{kj} V_k(i-1) \)
- \( \text{Ptr}(j)(i) = \arg\max_k a_{kj} V_k(i-1) \)

**Termination:**
- \( P(O, S^*) = \max_k V_k(N) \)

**Traceback:**
- \( S^N = \arg\max_k V_k(N) \)
- \( S_{i-1}^* = \text{Ptr}_{S_i}(i) \)

---

Viterbi Algorithm – a practical detail

Underflows are a significant problem

\[
P[ O^1, \ldots, O^i, S^1, \ldots, S^i ] = a_{0S1} a_{S1S2} \ldots a_{S_i} e_{S_i}(O^1) \ldots e_{S_i}(O^i)
\]

These numbers become extremely small – underflow

**Solution:** Take the logs of all values

\[
V_i(i) = \log e_i(O^i) + \max_k [ V_k(i-1) + \log a_{ki} ]
\]

---

Example

Let \( O \) be a long sequence with a portion of \( \sim 1/6 \) 6’s, followed by a portion of \( \sim \frac{1}{6} \) 6’s...

\( O = 123456123456 \ldots 12345 \ 6626364656 \ldots 1626364656 \)

Then, it is not hard to show that optimal parse is (exercise):

\[
\begin{align*}
\text{FFF} & \quad \cdots \quad \text{F} \\
\text{LLL} & \quad \cdots \quad \text{L}
\end{align*}
\]

A modeling example:

CpG islands in DNA sequences
What is CpG island?

- **CpG sites**
  - Regions of DNA where a cytosine nucleotide (C) occurs next to a guanine nucleotide (G) in the linear sequence
  - Shorthand for "–C–phosphate–G–", that is, C and G separated by only one phosphate (phosphate links any two nucleotide together in DNA)
  - "CpG" notation is used to distinguish this linear sequence from the CG base-pairing of C and G

- **CpG islands**
  - Regions with a high frequency of CpG sites
  - Other pairs (AA, AG, AT...) have different frequencies
  - Many genes in mammalian genomes have CpG islands associated with the start of the gene

**Methylation & Silencing**

- **Methylation**
  - Addition of CH$_3$ in “C”
  - Methylation is inherited during cell division
  - Silences genes in region
  - C in CpG is often methylated
    - In mammals, 70-80% of CpG sites are methylated
  - CG often mutates to TG, when methylated

**Example: CpG Islands**

- CG often mutates to TG, when methylated
  - C → methyl-C → T

CpG sites in the genome are frequently methylated
→ CpG sites are very rare

Methylation often suppressed around genes, promoters
→ CpG sites are much more than elsewhere (CpG islands)

The presence of a CpG island is used to help in the prediction and annotation of genes.
HMM to model CpG islands

- **Goal:** Detect the CpG islands!
  - $S_i$: CpG island or not
  - $O_i$: nucleotide sequence

A model of CpG Islands – (1)

**Architecture**

Not CpG → CpG → CpG → Genomic position

DNA seq. A T C G C G ...

A model of CpG Islands – (2)

**Transitions**

How do we estimate parameters of the model?

<table>
<thead>
<tr>
<th>Emission probabilities:</th>
<th>+</th>
<th>A</th>
<th>C</th>
<th>G</th>
<th>T</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transition probabilities within CpG islands</td>
<td>1</td>
<td>.180</td>
<td>.274</td>
<td>.426</td>
<td>.120</td>
</tr>
<tr>
<td>Established from known CpG islands (Training Set)</td>
<td>1</td>
<td>.171</td>
<td>.368</td>
<td>.274</td>
<td>.188</td>
</tr>
<tr>
<td>Transition probabilities within other regions</td>
<td>1</td>
<td>.161</td>
<td>.339</td>
<td>.375</td>
<td>.125</td>
</tr>
<tr>
<td>Established from known non-CpG islands (Training Set)</td>
<td>1</td>
<td>.079</td>
<td>.355</td>
<td>.384</td>
<td>.182</td>
</tr>
</tbody>
</table>

**Note:** these transitions out of each state sum to 1: no room for transitions between (+) and (-) states

| Log likelihood ratios – Telling “CpG Island” from “Non-CpG Island” |
|-------------------------|----|----|----|----|----|
| $L(u, v) = \log \frac{P(\mathbf{x} \mid +)}{P(\mathbf{x} \mid -)}$ |
| 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 |

Given a region $\mathbf{x} = x_1 \ldots x_n$
A quick-&-dirty way to decide whether entire $\mathbf{x}$ is CpG

$P(\mathbf{x} \text{ is CpG}) > P(\mathbf{x} \text{ is not CpG}) \Rightarrow \sum_i L(x_i, x_{i+1}) > 0$
A model of CpG Islands – (2)

Transitions

- What about transitions between (+) and (-) states?
- They affect
  - Avg. length of CpG island
  - Avg. separation between two CpG islands

Length distribution of region X:

\[ P[l_x = 1] = 1-p \]
\[ P[l_x = 2] = p(1-p) \]
\[ \vdots \]
\[ P[l_x = k] = p^{k-1}(1-p) \]

\[ E[l_x] = 1/(1-p) \]

Geometric distribution, with
mean \( 1/(1-p) \)

What if a new genome comes?

- We just sequenced the porcupine genome
- We know CpG islands play the same role in this genome
- However, we have no known CpG islands for porcupines
- We suspect the frequency and characteristics of CpG islands are quite different in porcupines

How do we adjust the parameters in our model?

LEARNING

Hidden Markov Models

- Decoding problem
  - Find the most likely parse of a sequence

- Learning problem
  - Update the parameters of the model based on training data

Two learning scenarios

- Estimation when the "right answer" is known
  Example:
  GIVEN: a genomic region \( x = x_1, \ldots, x_{1,000,000} \) where we have good (experimental) annotations of the CpG islands
  GIVEN: the casino player allows us to observe him one evening, as he changes dice and produces 10,000 rolls

- Estimation when the "right answer" is unknown
  Example:
  GIVEN: the porcupine genome; we don’t know how frequent are the CpG islands there, neither do we know their composition
  GIVEN: the 10,000 rolls of the casino player, but we don’t see when he changes dice

Question:
Update the parameters \( \theta \) of the model to maximize \( P(O|\theta) \)
1. When the states are known

Given $O = O^1 \cdots O^N$ for which the true $S = S^1 \cdots S^N$ is known,

Define:

- $A_{kl} = \# \text{ times } k \rightarrow l \text{ transition occurs in } S$
- $E_k(b) = \# \text{ times state } k \text{ in } S \text{ emits } b \text{ in } O$

We can show that the maximum likelihood parameters $\theta$ (maximize $P(O|\theta)$) are:

$a_{kl} = \frac{A_{kl}}{\sum_i A_{ki}}$;
$e_k(b) = \frac{E_k(b)}{\sum_c E_k(c)}$

Intuition: When we know the underlying states, best estimate is the normalized frequency of transitions & emissions that occur in the training data.

Example:
Given 10 casino rolls, we observe
$O = 2, 1, 5, 6, 1, 2, 3, 6, 2, 3$

Then:
$a_{FF} = 5/7$;
$a_{FL} = 2/7$;
$e_F(1) = e_F(3) = 2/8$;
$e_F(2) = 1/8$; $e_F(4) = 0$; $e_F(5) = e_F(6) = 1/8$
$e_L(2) = 1$; $e_L(1) = e_L(3) = e_L(4) = e_L(5) = e_L(6) = 0$

1. When the states are known

Drawback: Given little data, there may be overfitting: $P(O|\theta)$ is maximized, but $\theta$ is unreasonable
0 probabilities – BAD

Example:
Given 10 casino rolls, we observe
$O = 2, 1, 5, 6, 1, 2, 3, 6, 2, 3$

Then:
$a_{FF} = 1$; $a_{FL} = 0$;
$e_F(1) = e_F(3) = .2$;
$e_F(2) = .3$; $e_F(4) = 0$; $e_F(5) = e_F(6) = .1$

Pseudocounts

Solution for small training sets:

Add pseudocounts

$A_{kl} = \# \text{ times } k \rightarrow l \text{ transition occurs in } S + r_{kl}$
$E_k(b) = \# \text{ times state } k \text{ in } x \text{ emits } b \text{ in } x + r_k(b)$

$r_{kl}$, $r_k(b)$ are pseudocounts representing our prior belief

Larger pseudocounts $\Rightarrow$ Strong prior belief

Small pseudocounts ($\epsilon < 1$): just to avoid 0 probabilities
Pseudocounts

**Example:** dishonest casino

We will observe player for one day, 600 rolls

Reasonable pseudocounts:

- \( r_{FF} = r_{FL} = r_{LF} = r_{LL} = 1; \)
- \( r_{F1} = r_{L1} = 1; \)
- \( r_{F1} = r_{F2} = \ldots = r_{F6} = 20 \) (strong belief fair is fair)
- \( r_{L1} = r_{L2} = \ldots = r_{L6} = 5 \) (wait and see for loaded)

Above #s are arbitrary – assigning priors is an art

2. When the states are hidden

- We do not know the true \( A_{kl}, E_k(b) \)

  **Idea:**
  
  - We estimate our "best guess" on what \( A_{kl}, E_k(b) \) are
  - Or, we start with random / uniform values
  
  - We update the parameters of the model, based on our guess
  
  - Estimate the states based on the parameters
  
  - Repeat

Use the Viterbi algorithm

- Initialization: Pick the best-guess for model parameters (or arbitrary)

- Iteration:
  1. Apply the Viterbi algorithm to find \( S^* \)
  2. Calculate \( A_{kl}, E_k(b) \) according to \( S^* \)
  3. Calculate the new parameters \( a_{kl}, e_k(b) \)
  4. Until convergence

- Notes:
  - Not guaranteed to increase \( P(O | \theta) \)
  - Guaranteed to increase \( P(O | \theta, S^*) \)

Gene Recognition
Gene Finding

- **Gene:**
  A sequence of nucleotides coding for protein

- **Gene finding problem:**
  Determine the beginning and end positions of genes in a genome

Genes & proteins

<table>
<thead>
<tr>
<th>gene</th>
<th>transcription</th>
<th>translation</th>
<th>protein</th>
</tr>
</thead>
</table>

Double-stranded DNA

Single-stranded RNA

Transcription video!

http://www.youtube.com/watch?v=DA2t5N72mgw

RNA processing (splicing)

5' cap, polyadenylation, exon, intron, splicing, UTR, mRNA

Splicing video!

http://www.youtube.com/watch?v=aVgwr0QpYNE
Gene structure

Transcription Start Site (TSS)

5' UTR
promoter
exons
3' UTR

5' 3'

introns
coding region (~5%)
non-coding (~95%)

Needles in a Haystack

Signals for gene finding

1. Regular gene structure
2. Exon/intron lengths

In more detail
(color ~state)

Start codon codons
Donor site

5' UTR
Promoter
Exon (Left)

CCTGCCAC

5' UTR
Intron (Removed)

GATCCCATGCCGAGGGCCCCTG

Stop codon

GGGAGAACAAATAAGCAC

Poly-A site

3' UTR
Exon and Intron Lengths

Signals for gene finding

1. Regular gene structure
2. Exon/intron lengths
3. Codon composition

Nucleotide Composition

- Base composition in exons is characteristic due to the genetic code

<table>
<thead>
<tr>
<th>Amino Acid</th>
<th>SLC</th>
<th>DNA Codons</th>
</tr>
</thead>
<tbody>
<tr>
<td>Isoleucine</td>
<td>I</td>
<td>ATT, ATC, ATA</td>
</tr>
<tr>
<td>Leucine</td>
<td>L</td>
<td>CTT, CTC, CTA, TTA, TTG</td>
</tr>
<tr>
<td>Valine</td>
<td>V</td>
<td>GTT, GTC, GTA, GTG</td>
</tr>
<tr>
<td>Phenylalanine</td>
<td>F</td>
<td>TTT, TTC</td>
</tr>
<tr>
<td>Methionine</td>
<td>M</td>
<td>ATG</td>
</tr>
<tr>
<td>Cysteine</td>
<td>C</td>
<td>TGT, TGC</td>
</tr>
<tr>
<td>Alanine</td>
<td>A</td>
<td>GCT, GCC, GCA, GCG</td>
</tr>
<tr>
<td>Glycine</td>
<td>G</td>
<td>GGT, GGC, GGA, GGG</td>
</tr>
<tr>
<td>Proline</td>
<td>P</td>
<td>CCT, CCC, CCA, CGG</td>
</tr>
<tr>
<td>Threonine</td>
<td>T</td>
<td>ACT, ACC, ACA, ACG</td>
</tr>
<tr>
<td>Serine</td>
<td>S</td>
<td>TCT, TCC, TCA, TTC, TCG, AGT, AGC</td>
</tr>
<tr>
<td>Tyrosine</td>
<td>Y</td>
<td>TAT, TAC</td>
</tr>
<tr>
<td>Tryptophan</td>
<td>W</td>
<td>TGG</td>
</tr>
<tr>
<td>Glutamine</td>
<td>Q</td>
<td>CAA, CAG</td>
</tr>
<tr>
<td>Asparagine</td>
<td>N</td>
<td>AAT, AAC</td>
</tr>
<tr>
<td>Histidine</td>
<td>H</td>
<td>CAT, CAC</td>
</tr>
<tr>
<td>Glutamic acid</td>
<td>E</td>
<td>GAA, GAG</td>
</tr>
<tr>
<td>Aspartic acid</td>
<td>D</td>
<td>GAT, GAC</td>
</tr>
<tr>
<td>Lysine</td>
<td>K</td>
<td>AAA, AAG</td>
</tr>
<tr>
<td>Arginine</td>
<td>R</td>
<td>CGT, CGC, CGA, CGG, AGA, AGG</td>
</tr>
</tbody>
</table>
Signals for gene finding

1. Regular gene structure
2. Exon/intron lengths
3. Codon composition
4. Motifs at the boundaries of exons, introns, etc

Splice Sites

From http://www-lmmb.ncifcrf.gov/~toms/sequencelogo.html

Signals for gene finding

1. Regular gene structure
2. Exon/intron lengths
3. Codon composition
4. Motifs at the boundaries of exons, introns, etc
5. Patterns of conservation
6. Sequenced mRNAs
**HMMs for Gene Recognition**

- **Intergene State**
- **First Exon State**
- **Intron State**

```
GTCAGATGAGCAAAGTAGACACTCCAGTAACGCGGTGAGTACATTAA
```

**Genscan**

- N: Intergenic region
- P: promoter
- F: 5’ UTR
- \( E_{\text{sing}} \): single-exon (intronless) gene (start codon \( \rightarrow \) stop codon)
- \( E_{\text{int}} \): initial exon (start codon \( \rightarrow \) donor splice site)
- \( E_{\text{ssngl}} \): phase k internal exon (acceptor \( \rightarrow \) donor splice site)
- \( E_{\text{term}} \): terminal exon (acceptor \( \rightarrow \) stop codon)
- T: 3’ UTR

**Representation of the human genome**

- The representation of the whole human genome
  - 1990’s: A sequence of nucleotides (discrete features)
  - 2000’s: A collection of numerical data tracts with values almost every part of the genome

- Epigenetic modification
  - DNA methylation
  - Histone modification

*In the next lecture*
Acknowledgement

- The lecture slides were generated based on Prof. Serafim Batzhglou’s slides