The Gene Finding Task

Given: an uncharacterized DNA sequence
Do: locate the genes in the sequence, including the coordinates of individual exons and introns

image from the UCSC Genome Browser
http://genome.ucsc.edu/
Gene Expression Revisited

Sources of Evidence for Gene Finding

- **signals**: the sequence *signals* (e.g. splice junctions) involved in gene expression
- **content**: statistical properties that distinguish protein-coding DNA from non-coding DNA
- **conservation**: signal and content properties that are conserved across related sequences (e.g. syntenic regions of the mouse and human genome)
Gene Finding: Search by Content

- encoding a protein affects the statistical properties of a DNA sequence
  - some amino acids are used more frequently than others (Leu more popular than Trp)
  - different numbers of codons for different amino acids (Leu has 6, Trp has 1)
  - for a given amino acid, usually one codon is used more frequently than others
    - this is termed codon preference
    - these preferences vary by species

<table>
<thead>
<tr>
<th>AA</th>
<th>codon</th>
<th>/1000</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gly</td>
<td>GGG</td>
<td>1.89</td>
</tr>
<tr>
<td>Gly</td>
<td>GGA</td>
<td>0.44</td>
</tr>
<tr>
<td>Gly</td>
<td>GGU</td>
<td>52.99</td>
</tr>
<tr>
<td>Gly</td>
<td>GGC</td>
<td>34.55</td>
</tr>
<tr>
<td></td>
<td>GAG</td>
<td>15.68</td>
</tr>
<tr>
<td>Glu</td>
<td>GAA</td>
<td>57.20</td>
</tr>
<tr>
<td>Asp</td>
<td>GAU</td>
<td>21.63</td>
</tr>
<tr>
<td>Asp</td>
<td>GAC</td>
<td>43.26</td>
</tr>
</tbody>
</table>
Reading Frames

- a given sequence may encode a protein in any of the six reading frames

Open Reading Frames (ORFs)

- an ORF is a sequence that
  - starts with a potential start codon
  - ends with a potential stop codon, *in the same reading frame*
  - doesn’t contain another stop codon in-frame
  - and is sufficiently long (say > 100 bases)

- an ORF meets the minimal requirements to be a protein-coding gene in an organism without introns
Markov Models & Reading Frames

• consider modeling a given coding sequence
• for each “word” we evaluate, we’ll want to consider its position with respect to the reading frame we’re assuming

<table>
<thead>
<tr>
<th>G C T A C G</th>
<th>G A G C T T C G G A G C</th>
<th>reading frame</th>
</tr>
</thead>
<tbody>
<tr>
<td>G C T A C G</td>
<td>G is in 3rd codon position</td>
<td></td>
</tr>
<tr>
<td>C T A C G G</td>
<td>G is in 1st position</td>
<td></td>
</tr>
<tr>
<td>T A C G G A</td>
<td>A is in 2nd position</td>
<td></td>
</tr>
</tbody>
</table>

• can do this using an inhomogenous model

A Fifth Order Inhomogenous Markov Chain

<table>
<thead>
<tr>
<th>position 2</th>
<th>position 3</th>
<th>position 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>AA AAAA</td>
<td>AA AAAA</td>
<td>AA AAAA</td>
</tr>
<tr>
<td>CTACA</td>
<td>CTACA</td>
<td>TACAA</td>
</tr>
<tr>
<td>CTACC</td>
<td>CTACC</td>
<td>TACAC</td>
</tr>
<tr>
<td>CTACG</td>
<td>CTACG</td>
<td>TACAG</td>
</tr>
<tr>
<td>CTACT</td>
<td>CTACT</td>
<td>TACAT</td>
</tr>
<tr>
<td>GCTAC</td>
<td>GCTAC</td>
<td></td>
</tr>
<tr>
<td>TTTTT</td>
<td>TTTTT</td>
<td>TTTTT</td>
</tr>
</tbody>
</table>

transition: to states in pos 2
Selecting the Order of a Markov Chain Model

- higher order models remember more “history”
- additional history can have predictive value
- example:
  - predict the next word in this sentence fragment
    “…ends ___” (up, it, well, of, …?)
  - now predict it given more history
    “…that ends ___”
    “…well that ends ___”
    “All’s well that ends ___”

Selecting the Order of a Markov Chain Model

- but the number of parameters we need to estimate grows exponentially with the order
  - for modeling DNA we need $O(4^{n+1})$ parameters for an $n$th order model
- the higher the order, the less reliable we can expect our parameter estimates to be
  - estimating the parameters of a 2nd order homogenous Markov chain from the complete genome of E. Coli, we’d see each word > 72,000 times on average
  - estimating the parameters of an 8th order chain, we’d see each word ~ 5 times on average
Interpolated Markov Models

• the IMM idea: manage this trade-off by interpolating among models of various orders
• *simple* linear interpolation:

\[
\Pr_{\text{IMM}}(x_i \mid x_{i-n}, \ldots, x_{i-1}) = \lambda_0 \Pr(x_i) \\
+ \lambda_1 \Pr(x_i \mid x_{i-1}) \\
+ \lambda_n \Pr(x_i \mid x_{i-n}, \ldots, x_{i-1})
\]

• where \( \sum_i \lambda_i = 1 \)

Interpolated Markov Models

• we can make the weights depend on the history
  – for a given order, we may have significantly more data to estimate some words than others
• *general* linear interpolation

\[
\Pr_{\text{IMM}}(x_i \mid x_{i-n}, \ldots, x_{i-1}) = \lambda_0 \Pr(x_i) \\
+ \lambda_1(x_{i-1}) \Pr(x_i \mid x_{i-1}) \\
+ \lambda_n(x_{i-n}, \ldots, x_{i-1}) \Pr(x_i \mid x_{i-n}, \ldots, x_{i-1})
\]
The GLIMMER System

• Salzberg et al., 1998
• system for identifying genes in bacterial genomes
• uses 8th order, inhomogeneous, interpolated Markov chain models

IMMs in GLIMMER

• how does GLIMMER determine the $\lambda$ values?
• first, let’s express the IMM probability calculation recursively

$$
\Pr_{IMM,n} (x_i \mid x_{i-n}, \ldots, x_{i-1}) = \\
\lambda_n (x_{i-n}, \ldots, x_{i-1}) \Pr(x_i \mid x_{i-n}, \ldots, x_{i-1}) + \\
[1 - \lambda_n (x_{i-n}, \ldots, x_{i-1})] \Pr_{IMM,n-1} (x_i \mid x_{i-n+1}, \ldots, x_{i-1})
$$

• let $c(x_{i-n}, \ldots, x_{i-1})$ be the number of times we see the history $x_{i-n}, \ldots, x_{i-1}$ in our training set

$$
\lambda_n (x_{i-n}, \ldots, x_{i-1}) = 1 \quad \text{if} \quad c(x_{i-n}, \ldots, x_{i-1}) > 400
$$
**IMMs in GLIMMER**

- if we haven’t seen \( x_{i-n}, \ldots, x_{i-1} \) more than 400 times, then compare the counts for the following:

<table>
<thead>
<tr>
<th>( n )-th order history + base</th>
<th>( (n-1) )-th order history + base</th>
</tr>
</thead>
<tbody>
<tr>
<td>( x_{i-n}, \ldots, x_{i-1}, a )</td>
<td>( x_{i-n+1}, \ldots, x_{i-1}, a )</td>
</tr>
<tr>
<td>( x_{i-n}, \ldots, x_{i-1}, c )</td>
<td>( x_{i-n+1}, \ldots, x_{i-1}, c )</td>
</tr>
<tr>
<td>( x_{i-n}, \ldots, x_{i-1}, g )</td>
<td>( x_{i-n+1}, \ldots, x_{i-1}, g )</td>
</tr>
<tr>
<td>( x_{i-n}, \ldots, x_{i-1}, t )</td>
<td>( x_{i-n+1}, \ldots, x_{i-1}, t )</td>
</tr>
</tbody>
</table>

- use a statistical test (\( \chi^2 \)) to get a value \( d \) indicating our confidence that the distributions represented by the two sets of counts are different.

**Putting it all together**

\[
\lambda_n(x_{i-n}, \ldots, x_{i-1}) = \begin{cases} 
1 & \text{if } c(x_{i-n}, \ldots, x_{i-1}) > 400 \\
\frac{d \times c(x_{i-n}, \ldots, x_{i-1})}{400} & \text{else if } d \geq 0.5 \\
0 & \text{otherwise}
\end{cases}
\]

where \( d \in (0,1) \).
IMM Example

• suppose we have the following counts from our training set

<table>
<thead>
<tr>
<th></th>
<th>ACGA</th>
<th>CGA</th>
<th>GA</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACGA</td>
<td>25</td>
<td>100</td>
<td>175</td>
</tr>
<tr>
<td>ACGC</td>
<td>40</td>
<td>90</td>
<td>140</td>
</tr>
<tr>
<td>ACGG</td>
<td>15</td>
<td>35</td>
<td>65</td>
</tr>
<tr>
<td>ACGT</td>
<td>20</td>
<td>75</td>
<td>120</td>
</tr>
</tbody>
</table>

\[ \chi^2 \text{ test: } d = 0.857 \quad \chi^2 \text{ test: } d = 0.141 \]

\[ \lambda_3(ACG) = 0.857 \times \frac{100}{400} \]

\[ \lambda_2(CG) = 0 \quad (d < 0.5, \ c(CG) < 400) \]

\[ \lambda_1(G) = 1 \quad (c(G) > 400) \]

IMM Example (Continued)

• now suppose we want to calculate \( \Pr_{\text{IMM,3}}(T \mid ACG) \)

\[ \Pr_{\text{IMM,1}}(T \mid G) = \lambda_1(G) \Pr(T \mid G) + (1 - \lambda_1(G)) \Pr_{\text{IMM,0}}(T) \]

\[ = \Pr(T \mid G) \]

\[ \Pr_{\text{IMM,2}}(T \mid CG) = \lambda_2(CG) \Pr(T \mid CG) + (1 - \lambda_2(CG)) \Pr_{\text{IMM,1}}(T \mid G) \]

\[ = \Pr(T \mid G) \]

\[ \Pr_{\text{IMM,3}}(T \mid ACG) = \lambda_3(ACG) \Pr(T \mid ACG) + (1 - \lambda_3(ACG)) \Pr_{\text{IMM,2}}(T \mid CG) \]

\[ = 0.214 \times \Pr(T \mid ACG) + (1 - 0.214) \times \Pr(T \mid G) \]
Gene Recognition in GLIMMER

- essentially ORF classification
- for each ORF
  - calculate the prob of the ORF sequence in each of the 6 possible reading frames
  - if the highest scoring frame corresponds to the reading frame of the ORF, mark the ORF as a gene
- for overlapping ORFs that look like genes
  - score overlapping region separately
  - predict only one of the ORFs as a gene

GLIMMER Experiment

- 8th order IMM vs. 5th order Markov model
- trained on 1168 genes (ORFs really)
- tested on 1717 annotated (more or less known) genes
GLIMMER Results

<table>
<thead>
<tr>
<th>Model</th>
<th>Genes found</th>
<th>Genes missed</th>
<th>Additional genes</th>
</tr>
</thead>
<tbody>
<tr>
<td>GLIMMER IMM</td>
<td>1680 (97.8%)</td>
<td>37</td>
<td>209</td>
</tr>
<tr>
<td>5th-Order Markov</td>
<td>1574 (91.7%)</td>
<td>143</td>
<td>104</td>
</tr>
</tbody>
</table>

The first column indicates how many of the 1717 annotated genes in *H. influenzae* were found by each algorithm. The ‘additional genes’ column shows how many extra genes, not included in the 1717 annotated entries, were called genes by each method.

- GLIMMER has greater sensitivity than the baseline
- It’s not clear if its precision/specificity is better

An Alternative Approach: Back-off Models

- Devised for language modeling

\[
\Pr_{\text{BACK}}(x_i | x_{i-n}, \ldots, x_{i-1}) = \begin{cases} 
(1 - \delta) \frac{c(x_{i-n}, \ldots, x_i)}{c(x_{i-n}, \ldots, x_{i-1})}, & \text{if } c(x_{i-n}, \ldots, x_i) > k \\
\lambda \Pr_{\text{BACK}}(x_i | x_{i-n+1}, \ldots, x_{i-1}), & \text{otherwise}
\end{cases}
\]

- Use *n*th order probability if we’ve seen this sequence (*history + current character*) *k* times
- Otherwise back off to lower-order
An Alternative Approach: Back-off Models

\[
\Pr_{\text{BACK}}(x_i \mid x_{i-n}, \ldots, x_{i-1}) = \begin{cases} 
(1 - \delta) \frac{c(x_{i-n}, \ldots, x_i)}{c(x_{i-n}, \ldots, x_{i-1})}, & \text{if } c(x_{i-n}, \ldots, x_i) > k \\
\lambda \Pr_{\text{BACK}}(x_i \mid x_{i-n+1}, \ldots, x_{i-1}), & \text{otherwise}
\end{cases}
\]

- why do we need \( \delta \) and \( \lambda \) ?
- \( \delta \): save some probability mass for sequences we haven’t seen
- \( \lambda \): distribute this saved mass to lower-order sequences (different \( \lambda \) for each history; really \( \lambda(x_{i-n+1}, \ldots, x_{i-1}) \))
- this is important for natural language, where there are many words that could follow a particular history

Simple Back-off Example

- given training sequence: TAACGACACG
- suppose \( \delta = 0.2 \) and \( k = 0 \)

\[
\begin{align*}
\Pr_{\text{BACK}}(A) &= \frac{4}{10} & \Pr_{\text{BACK}}(A \mid A) &= (1 - \delta) \frac{1}{4} = 0.2 \\
\Pr_{\text{BACK}}(C) &= \frac{3}{10} & \Pr_{\text{BACK}}(C \mid A) &= (1 - \delta) \frac{3}{4} = 0.6 \\
\Pr_{\text{BACK}}(G) &= \frac{2}{10} & \Pr_{\text{BACK}}(G \mid A) &= \left[ \frac{\delta}{\Pr_{\text{BACK}}(G) + \Pr_{\text{BACK}}(T)} \right] \times \Pr_{\text{BACK}}(G) = \frac{0.2}{0.3} \times 0.2 \\
\Pr_{\text{BACK}}(T) &= \frac{1}{10} & \Pr_{\text{BACK}}(T \mid A) &= \left[ \frac{\delta}{\Pr_{\text{BACK}}(G) + \Pr_{\text{BACK}}(T)} \right] \times \Pr_{\text{BACK}}(T) = \frac{0.2}{0.3} \times 0.1
\end{align*}
\]