BMI/CS 776
Lecture 11
Prokaryotic Gene Finding

Colin Dewey
February 26, 2008
Gene Expression Revisited

eukaryotes

prokaryotes
Approaches to Finding Genes

• **search by sequence similarity**: find genes by looking for matches to previously identified genes

• **search by signal**: find genes by identifying the sequence signals involved in gene expression

• **search by content**: find genes by statistical properties that distinguish protein-coding DNA from non-coding DNA

• **combined**: state-of-the-art systems for gene finding combine these strategies
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
### Codon Preference in E. Coli

<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>Glu</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

G C T A C G G A G C T T T C G G A G C
C G A T G C C T C G A A G C C T C G
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

**reading frame**

![Reading Frame Diagram]

**G** is in 3\textsuperscript{rd} codon position

**G** is in 1\textsuperscript{st} codon position

**A** is in 2\textsuperscript{nd} codon position

**can do this using an inhomogenous model**
A Fifth Order Inhomogeneous Markov Chain
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 nth 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 $\sim 17$ times on average
Interpolated Markov Models

- the IMM idea: manage this trade-off by interpolating among models of various orders

- simple linear interpolation:

\[
\Pr_{\text{IMM}}(x_i \mid x_{i-n}, \ldots, x_{i-1}) = \lambda_0 \Pr(x_i) + \lambda_1 \Pr(x_i \mid x_{i-1}) + \lambda_2 \Pr(x_i \mid x_{i-2}) + \cdots + \lambda_n \Pr(x_i \mid x_{i-n}, \ldots, x_{i-1})
\]

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

- \textit{general} linear interpolation

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

$$\Pr_{\text{IMM},n}(x_i \mid x_{i-n}, \ldots, x_{i-1}) =$$

$$\lambda_n(x_{i-n}, \ldots, x_{i-1}) \Pr(x_i \mid x_{i-n}, \ldots, x_{i-1}) +$$

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

• 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

$$\lambda_n(x_{i-n}, \ldots, x_{i-1}) = 1 \text{ if } c(x_{i-n}, \ldots, x_{i-1}) > 400$$
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:

\[
\begin{array}{c|c}
\text{nth order history + base} & \text{(n-1)th order history + base} \\
\hline
x_{i-n}, \ldots, x_{i-1}, a & x_{i-n+1}, \ldots, x_{i-1}, a \\
x_{i-n}, \ldots, x_{i-1}, c & x_{i-n+1}, \ldots, x_{i-1}, c \\
x_{i-n}, \ldots, x_{i-1}, g & x_{i-n+1}, \ldots, x_{i-1}, g \\
x_{i-n}, \ldots, x_{i-1}, t & x_{i-n+1}, \ldots, x_{i-1}, t \\
\end{array}
\]

• 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.
IMMs in GLIMMER

putting it all together

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

where \( d \in (0,1) \)
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>count</td>
<td>25</td>
<td>100</td>
<td>175</td>
</tr>
<tr>
<td></td>
<td>ACGC</td>
<td>CGC</td>
<td>GC</td>
</tr>
<tr>
<td>count</td>
<td>40</td>
<td>90</td>
<td>140</td>
</tr>
<tr>
<td></td>
<td>ACGG</td>
<td>CGG</td>
<td>GG</td>
</tr>
<tr>
<td>count</td>
<td>15</td>
<td>35</td>
<td>65</td>
</tr>
<tr>
<td></td>
<td>ACGT</td>
<td>CGT</td>
<td>GT</td>
</tr>
<tr>
<td>count</td>
<td>20</td>
<td>75</td>
<td>120</td>
</tr>
</tbody>
</table>

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

\[\lambda_3(\text{ACG}) = 0.857 \times \frac{100}{400} = 0.214\]

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

\[\lambda_1(\text{G}) = 1 \quad (c(\text{G}) > 400)\]
IMM Example (Continued)

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

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

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

\[
\Pr_{IMM,3}(T|ACG) = \lambda_3(ACG) \Pr(T|ACG) + (1 - \lambda_3(ACG)) \Pr_{IMM,2}(T|CG)
\]
\[
= (0.214) \Pr(T|ACG) + (1 - 0.214) \Pr(T|G)
\]
Gene Recognition in GLIMMER

- essentially ORF classification

- for each ORF
  - calculate the prob of the ORF sequence in each of the six 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
## Accuracy Metrics

### Confusion Matrix

<table>
<thead>
<tr>
<th></th>
<th>Positive</th>
<th>Negative</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>True Positives</strong> (TP)</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>False Negatives</strong> (FN)</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>False Positives</strong> (FP)</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>True Negatives</strong> (TN)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Metrics

**Sensitivity (Recall)**

\[
sensitivity = \frac{TP}{TP + FN}
\]

**Specificity**

\[
specificity = \frac{TN}{TN + FP}
\]

**Precision**

\[
precision = \frac{TP}{TP + FP}
\]
## GLIMMER Results

<table>
<thead>
<tr>
<th>Model</th>
<th>TP</th>
<th>FN</th>
<th>FP &amp; TP?</th>
</tr>
</thead>
<tbody>
<tr>
<td>GLIMMER IMM</td>
<td>1680</td>
<td>37</td>
<td>209</td>
</tr>
<tr>
<td>5(^{th})-Order Markov</td>
<td>1574</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

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

- 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

• \( \delta \) and \( \lambda \) generally vary depending on the history. That is, we have \( \lambda(x_{i-n}, \ldots, x_{i-1}) \) and \( \delta(x_{i-n}, \ldots, x_{i-1}) \)
Simple Back-off Example

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

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

\[
\begin{align*}
\Pr_{BACK}(A) &= \frac{4}{10} \\
\Pr_{BACK}(C) &= \frac{3}{10} \\
\Pr_{BACK}(G) &= \frac{2}{10} \\
\Pr_{BACK}(T) &= \frac{1}{10}
\end{align*}
\]

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