Computational Genomics Overview

What have we learned so far?

- String searching and counting
  - Brute-force algorithm
  - W-mer indexing
- Sequence alignment
  - Dynamic programming, duality path alignment
  - Global / local alignment, general gap penalties
- String comparison
  - Exact string match, semi-numerical matching
- Rapid database search
  - Exact matching: Hashing, BLAST
  - Inexact matching: neighborhood search, projections
- Problem set 1

What do you do?

...GTACTCACGGGTTACAGGATTATGGGTTACAGGTAACCGTT...

- Non-standard nucleotide composition?
- Interesting k-mer frequencies?
- Recurring patterns?

• Align it to things we know about
• Align it to things we don’t know about
• Stare at it
  - Non-standard nucleotide composition?
  - Interesting k-mer frequencies?
  - Recurring patterns?
• Model it
  - Make some hypotheses about it
  - Build a ‘generative model’ to describe it
  - Find sequences of similar type

This week: Modeling biological sequences
(a.k.a. What to do with a huge chunk of DNA)

- Ability to emit DNA sequences of a certain type
  - Not exact alignment to previously known gene
  - Preserving ‘properties’ of type, not identical sequence
- Ability to recognize DNA sequences of a certain type (state)
  - What (hidden) state is most likely to have generated observations
  - Find set of states and transitions that generated a long sequence
- Ability to learn distinguishing characteristics of each state
  - Training our generative models on large datasets
  - Learn to classify unlabelled data

Why Probabilistic Sequence Modeling?

- Biological data is noisy
- Probability provides a calculus for manipulating models
- Not limited to yes/no answers – can provide “degrees of belief”
- Many common computational tools based on probabilistic models
First example: GC-rich regions

- Promoter regions frequently have higher counts of Gs and Cs
- Calculate sequence distribution from known regulatory regions
  - Count occurrences of A,T,G,C
- Model islands as nucleotides drawn independently from this distribution

A: 0.15
T: 0.13
G: 0.30
C: 0.42

The Probability of a Sequence

- Can calculate the probability of a particular sequence (S) according to the model for promoter regions (MP)

\[
P(S \mid MP) = P(S_1, S_2, ..., S_n \mid MP) = \prod_i P(S_i \mid MP)
\]

Example

S = AAATCGACTTCAAA

\[
P(S \mid MP) = P(A)^4 \times P(T)^4 \times P(G)^3 \times P(C)^2
\]

\[= (0.15)^4 \times (0.13)^4 \times (0.30)^3 \times (0.42)^2
\]

\[= 1.55 \times 10^{-11}
\]

Sequence Classification

PROBLEM: Given a sequence, is it an island?
- We can calculate \(P(S \mid MP)\), but what is a sufficient \(P\) value?

SOLUTION: compare to a null model and calculate log-likelihood ratio
- e.g. background DNA distribution model, B

\[
\text{Score} = \log \frac{P(S \mid MP)}{P(S \mid B)}
\]

Finding GC-rich regions

- Could use the log-likelihood ratio on windows of fixed size
  - What if islands have variable length?
- We prefer a model for entire sequence

A More Complex Model

A Generative Model
We would like to find the most likely path.

Exhaustive search? ... no

The three main questions on HMMs

1. Decoding

GIVEN a HMM \((e_i, a_{ij})\), and a sequence \(x\),
FIND \(\pi\) of states that maximizes \(P(x, \pi | M)\).

Viterbi algorithm

2. Evaluation

GIVEN a HMM \((e_i, a_{ij})\), and a sequence \(x\),
FIND Prob \(x | M\).

Forward algorithm

3. Learning

GIVEN a HMM \((\theta), \pi\), and a sequence \(x\),
FIND parameters \(\theta = (e_i, a_{ij})\) that maximize \(P(x | \theta)\).

Expectation Maximization

Finding the optimal path

- We can now evaluate any path through hidden states, given the emitted sequences.
- How do we find the best path?
- Optimal substructure! Best path through a given state is:
  - Best path to previous state
  - Best transition from previous state to this state
  - Best path to the end state

\(\text{Viterbi algorithm}\)

- Define \(V_k(i) = \text{Probability of the most likely path through state } x_i \ldots x_k\)
- Compute \(V_k(i+1)\) as a function of \(\max_j a_{ik} V_j(i)\)
- \(V_k(\pi) = a_{0k} x_{\pi k} \max_{a_{ik}} V_k(i)\)

\(\text{Dynamic Programming}\)

Finding the most likely path

- Find path \(\pi^*\) that maximizes total joint probability \(P(x, \pi)\)

\[ P(x, \pi) = \prod_{i=0}^{N-1} e_i(x_i) \times a_{\pi i, \pi i+1} \]
Calculate maximum $P(x, \pi)$ recursively

- Assume we know $V_j$ for the previous time step $(i-1)$
- Calculate $V_i(i) = e_k(x_i) \times \max_j (V_{j}(i-1) \times a_{jk})$

The Viterbi Algorithm

- Input: $x = x_1, \ldots, x_n$
- Initialization: $V_j(0) = 0$, for all $k > 0$
- Iteration: $V_j(i) = e_k(x_i) \times \max_j a_{jk} \times V_j(i-1)$
- Traceback: Follow max pointers back
- In practice: Use log scores for computation
- Time: $O(MN)$
- Space: $O(MN)$

What have we learned?

- Modeling biological sequences
  - Recognize a type of sequence
- Simple examples
  - The dishonest casino
  - Finding CpG islands
- Definitions
  - Markov Chains
  - Hidden Markov Models (HMMs)
- Our first computations
  - Evaluation: know model, emissions, states $\rightarrow p$?
  - Viterbi: know model, emissions $\rightarrow$ find optimal path

Next time: More on HMMs

- HMM algorithms
  - Viterbi algorithm
  - Forward / backward
- Working with HMMs
  - Posterior decoding
  - Baum-Welch training
  - Viterbi training
- Applications
  - HMMs for Gene Finding

One path

1. Scoring $x$, one path
   $P(x, \pi) = \sum_k P(x, \pi)$
   Prob of a path, emissions

2. Scoring $x$, all paths
   $P(x) = \sum_\pi P(x, \pi)$
   Prob of emissions, over all paths

3. Viterbi decoding
   $\pi^* = \arg\max_\pi P(x, \pi)$
   Most likely path

4. Posterior decoding
   $\pi^* = \arg\max_{\pi, k} P(\pi, k|x)$
   Path containing the most likely state at any time point

5. Supervised learning, given $\pi$
   $\Lambda^* = \arg\max_\Lambda P(x, \pi|\Lambda)$

6. Unsupervised learning
   $\Lambda^* = \arg\max_\Lambda \sum_k P(x, \pi|\Lambda)$
   Viterbi training, best path

Learning

Decoding

Scoring