Modeling Biological Sequence using Hidden Markov Models

Challenges in Computational Biology

Course Outline

What have we learned so far?

• Sequence alignment
  – Dynamic programming, duality path \( \Rightarrow \) alignment
  – Global / local alignment, general gap penalties
• Rapid string search
  – Exact string match, semi-numerical matching
  – Database search: Hashing, BLAST, variations
• Clustering / Modeling expression profiles
  – Algorithmic view: Clustering (k-means, hierarchical)
  – ML view: Unsupervised learning (Expectation-Maxi)
• Classification: Supervised learning
  – Model-based approach (Bayesian classification)
  – Discriminative approach (SVM and Kernel mapping)
• Problem set 1, Problem set 2

Today: apply these ideas to model DNA sequences

...GTACTCACCACGGTTACAGGATTACGTTACAGGTTACAGGTAACCGTT...

• What to do with a completely new piece of DNA
  – Align it to things we know about (database search)
  – Align it to things we don’t know about (assembly)
• Stare at it
  – Non-standard nucleotide composition?
  – Interesting k-mer frequencies?
  – Recurrent patterns?
• Model it
  – Make some hypotheses about it
  – Build a ‘generative model’ to describe it
  – Find sequences of similar type
  ➔ How do we model DNA sequences?

This week: Modeling biological sequences
(a.k.a. What to do with big unlabelled chunks 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
- Our tools:
  - Markov Chains and Hidden Markov Models (HMMs)

### Definitions: HMM (Hidden Markov Model)

**Definition:** An HMM is a 5-tuple \((Q, V, A, E, \pi)\), where:
- \(Q\) is a finite set of states, \(|Q|=N\)
- \(V\) is a finite set of observation symbols per state, \(|V|=M\)
- \(\pi\) is the initial state probabilities
- \(A\) is the state transition probabilities, denoted by \(a_{st}\) for each \(s, t \in Q\)
  - For each \(s, t \in Q\) the transition probability is: \(a_{st} = P(X_t = x_t | X_{t-1} = x_{t-1})\)
- \(E\) is a probability emission matrix, \(e_{ux} = P(v_x \text{ at time } t | q_t = s)\)

**Output:** Only emitted symbols are observable by the system but not the underlying random walk between states \(\rightarrow \text{"hidden"}\)

**Property:** Emissions and transitions are dependent on the current state only and not on the past.

### The six algorithmic settings for HMMs

#### One path

1. **Scoring**
   - \(P(x, \pi)\) Prob of a path, emissions

2. **Decoding**
   - \(\pi^* = \argmax_{\pi} P(x, \pi)\) Path containing the most likely state at any time point.

3. **Learning**
   - \(A^* = \argmax_A P(x, \pi(A))\)
   - \(\Lambda^* = \argmax_{\Lambda} \Sigma_{\pi(\Lambda)}\)
   - Baum-Welch training, over all paths

#### All paths

4. **Decoding**
   - \(P(x) = \sum_{\pi} P(x, \pi)\) Prob of emissions, over all paths

5. **Learning**
   - \(A^* = \argmax_A P(x, \pi(A))\)
   - \(\Lambda^* = \argmax_{\Lambda} \Sigma_{\pi(\Lambda)}\)
   - Baum-Welch training, over all paths

---

**Example 1: Finding GC-rich regions**

- Promoter regions frequently have higher counts of Gs and Cs
- Model genome as nucleotides drawn independently from two distributions: Background (B) and Promoters (P).  
- Emission probabilities based on nucleotide composition in each.
- Transition probabilities based on relative abundance & avg. length

---

### HMM as a Generative Model

- \(P(S|P)\)
- \(P(S|B)\)
- \(P(S|A)\)
- \(P(L_0|L)\)

---

**HMM as a Generative Model**

- \(S\): G, C, A, T

---

**Definition: Markov Chain**

**Definition:** A Markov chain is a triplet \((Q, p, A)\), where:
- \(Q\) is a finite set of states. Each state corresponds to a symbol in the alphabet \(\Sigma\)
- \(p\) is the initial state probabilities.
- \(A\) is the state transition probabilities, denoted by \(a_{st}\) for each \(s, t \in Q\).

**Output:** The output of the model is the set of states at each instant time \(\rightarrow\) the set of states are observable

**Property:** The probability of each symbol \(x_t\) depends only on the value of the preceding symbol \(x_{t-1}\):

\[
P(x_t | x_{t-1}, ..., x_1) = P(x_t | x_{t-1})
\]

**Formula:** The probability of the sequence:

\[
P(s) = P(x_1, x_2, ..., x_t) = P(x_t | x_{t-1}) P(x_{t-1} | x_{t-2})... P(x_2 | x_1) P(x_1)
\]

---

**Example 1: Finding GC-rich regions**

- Promoter regions frequently have higher counts of Gs and Cs
- Model genome as nucleotides drawn independently from two distributions: Background (B) and Promoters (P).
- Emission probabilities based on nucleotide composition in each.
- Transition probabilities based on relative abundance & avg. length

---

### 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
- Our tools:
  - Markov Chains and Hidden Markov Models (HMMs)
**Sequence Classification**

**PROBLEM:** Given a sequence, is it a promoter region?

- We can calculate $P(S|MP)$, but what is a sufficient $P$ value?[1](#)

**SOLUTION:** compare to a null model and calculate log-likelihood ratio

- e.g. background DNA distribution model, $B$

$$\text{Score} = \log \frac{P(S|MP)}{P(S|B)}$$

---

**Finding GC-rich regions**

- Could use the log-likelihood ratio on windows of fixed size

- Downside: have to evaluate all islands of all lengths repeatedly

- Need: a way to easily find transitions

---

**Probability of seq, path if all promoter**

$$L:\quad P \quad P \quad P \quad P \quad P \quad P \quad P \quad P \quad S:\quad G \quad 0.30 \quad C \quad 0.42 \quad A \quad 0.15 \quad T \quad 0.15 \quad A \quad 0.15 \quad A \quad 0.15 \quad T \quad 0.13 \quad G \quad 0.30 \quad S:\quad G \quad 0.30$$

$$P(x,\pi) = a_P^x e_P(G)^x a_P^x e_P(G)^x a_P^x e_P(C)^x a_P^x e_P(A)^x a_P^x$$

$$= a_P^x (0.75)^x (0.15)^x (0.13)^x (0.30)^x (0.42)^x$$

$$= 0.3 \times 10^{-7}$$

**Why is this so small?**

---

**Probability of seq, path sequence if mixed**

$$L:\quad B \quad B \quad B \quad P \quad P \quad B \quad B \quad S:\quad G \quad 0.25 \quad C \quad 0.25 \quad A \quad 0.42 \quad A \quad 0.42 \quad A \quad 0.42 \quad T \quad 0.30 \quad G \quad 0.25$$

$$P(x,\pi) = a_P^x e_P(G)^x a_P^x e_P(B)^x a_P^x e_P(C)^x a_P^x e_P(A)^x a_P^x$$

$$= a_P^x (0.85)^x (0.25)^x (0.75)^x (0.42)^x (0.30)^x (0.25)^x$$

$$= 6.7 \times 10^{-7}$$

**Should we try all possibilities? What is the most likely path?**

---

**The six algorithmic settings for HMMs**

### One path

1. Scoring $x$, one path
   $$P(x,\pi)$$

2. Scoring $x$, all paths
   $$P(x) = \sum_{\pi} P(x,\pi)$$

### All paths

3. Viterbi decoding
   $$\pi^* = \arg\max_{\pi} P(x,\pi)$$

4. Posterior decoding
   $$\pi^* = \{\pi \mid \pi = \arg\max_{\pi} \sum_{x} P(x,\pi)\}$$

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

6. Unsupervised learning
   $$\Lambda^* = \arg\max_{\Lambda} \sum_{x} P(x,\pi)$$

---

**Path containing the most likely state at any time point.**
3. DECODING: What was the sequence of hidden states?

Given: Model parameters \( e_i(.) \), \( a_{ij} \)

Given: Sequence of emissions \( x \)

Find: Sequence of hidden states \( \pi \)

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

→ Viterbi algorithm
  - Define \( V_k(i) = \text{Probability of the most likely path through state } \pi_k \)
  - Compute \( V_k(i+1) \) as a function of \( \max_{k'} \{ V_k'(i) \} \)
    \[
    V_k(i+1) = e_k(x_{i+1}) \times \max_j a_{jk} V_j(i)
    \]

→ Dynamic Programming

Finding the most likely path

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

\[
P(x, \pi) = \prod_{i=0}^{N-1} e_{\pi_i}(x_i) \times \max_{j} a_{\pi_i \pi_{i+1}}
\]

The Viterbi Algorithm

Input: \( x = x_1, \ldots, x_N \)

Initialization:
\( V_0(0) = 1, V_k(0) = 0, \text{ for all } k > 0 \)

Iteration:
\( V_k(i) = e_k(x_i) \times \max_j a_{jk} V_{j}(i-1) \)

Termination:
\( P(x, \pi^*) = \max_k V_k(N) \)

The six algorithmic settings for HMMs

**One path**

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

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

**All paths**

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

4. Posterior decoding
   \( \pi^* = (\pi_1, \ldots, \pi_N) = \arg \max_{\pi} \sum_{\pi} P(\pi; x) \)
   Path containing the most likely state at any time point.

5. Supervised learning, given \( \pi^* \)
   \( \Lambda^* = \arg \max_{\Lambda} P(x, \pi^*; \Lambda) \)
   Viterbi training, best path

6. Unsupervised learning
   \( \Lambda^* = \arg \max_{\Lambda} \sum_{\pi} P(x, \pi; \Lambda) \)
   Baum-Welch training, over all paths
2. EVALUATION
(how well does our model capture the world)

Given: Model parameters $e_i(.)$, $a_{ij}$
Given: Sequence of emissions $x$
Find: $P(x|M)$, summed over all possible paths $\pi$

Simple: Given the model, generate some sequence $x$

1. Start at state $\pi_1$ according to prob $a_{0\pi_1}$
2. Emit letter $x_1$ according to prob $e_{\pi_1}(x_1)$
3. Go to state $\pi_2$ according to prob $a_{\pi_1\pi_2}$
4. ... until emitting $x_n$

We have some sequence $x$ that can be emitted by $p$. Can calculate its likelihood. However, in general, many different paths may emit this same sequence $x$.

How do we find the total probability of generating a given $x$, over any path?

Complex: Given $x$, was it generated by the model?

Given a sequence $x$,
What is the probability that $x$ was generated by the model (using any path)?

- $P(x) = \sum_\pi P(x, \pi)$

- Challenge: exponential number of paths

The Forward Algorithm – derivation

Define the forward probability:

$f(i) = P(x_1, ..., x_i, \pi_i = l)$

$= \sum_{\pi_1, ..., \pi_{i-1}} P(x_1, ..., x_i, \pi_1, ..., \pi_{i-1}, \pi_i = l) e_\pi(x_i)$

$= \sum_{\pi_1, ..., \pi_{i-1}} P(x_1, ..., x_{i-1}, \pi_1, ..., \pi_{i-2}, \pi_{i-1}) a_{\pi_{i-1}\pi_i} e_\pi(x_i)$

$= \sum_{\pi_1, ..., \pi_{i-1}} f_{i-1}(\pi_{i-1}) a_{\pi_{i-1}\pi_i} e_\pi(x_i)$

$= e_\pi(x_i) \sum_{\pi_1, ..., \pi_{i-1}} f_{i-1}(\pi_{i-1}) a_{\pi_{i-1}\pi_i}$

Calculate total probability $\sum_\pi P(x, \pi)$ recursively

- Assume we know $f(i)$ for the previous time step $(i-1)$

- Calculate $f(i)$

  $f_i(l) = e_l(x_i) \sum_{\pi_{i-1}} f_{i-1}(\pi_{i-1}) a_{\pi_{i-1}l}$
The Forward Algorithm

Input: \( x = x_1 \ldots x_N \)

Initialization:
\( f_0(0) = 1, f_k(0) = 0, \text{ for all } k > 0 \)

Iteration:
\( f_k(i) = e^{K(x_i)} \times \sum_{j} a_{jk} f_{k-1}(j) \)

Termination:
\( P(x, \pi^*) = \sum_{k} f_k(N) \)

In practice:
- Sum of log scores is difficult
- \( \approx \) approximate \( \exp(1+p+q) \)
- Scaling of probabilities

Running time and space:
- Time: \( O(K^2N) \)
- Space: \( O(KN) \)

The six algorithmic settings for HMMs

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Scoring, one path</td>
<td>( P(x, \pi) )</td>
</tr>
<tr>
<td>2. Scoring, all paths</td>
<td>( P(x) = \sum_{\pi} P(x, \pi) )</td>
</tr>
<tr>
<td>3. Viterbi decoding</td>
<td>( \pi^* = \arg\max_{\pi} P(x, \pi) ) Most likely path</td>
</tr>
<tr>
<td>4. Posterior decoding</td>
<td>( \pi^* = { \pi_i</td>
</tr>
<tr>
<td>5. Supervised learning</td>
<td>( \Lambda^* = \arg\max_{\Lambda} \sum_{\pi} P(x, \pi</td>
</tr>
<tr>
<td>6. Unsupervised learning</td>
<td>( \Lambda^* = \arg\max_{\Lambda} \max_{\pi} P(x, \pi</td>
</tr>
</tbody>
</table>

Increasing the state of the system (looking back)

- **Markov Models are memory-less**
  - In other words, all memory is encoded in the states
  - To remember additional information, augment state

- **Our first HMM had minimal memory**
  - State, emissions, only depend on current state
  - Current state only encoded one previous nucleotide

- **How do you count di-nucleotide frequencies?**
  - CpG islands: di-nucleotides
  - Codon triplets: tri-nucleotides
  - Di-codon frequencies: six nucleotides

  ➔ Expanding the number of states

Counting nucleotide transitions: Markov/HMM

- **Markov Chain**
  - \( Q \): states
  - \( p \): initial state probabilities
  - \( A \): transition probabilities

- **HMM**
  - \( Q \): states
  - \( V \): observations
  - \( p \): initial state probabilities
  - \( A \): transition probabilities
  - \( E \): emission probabilities

Example 2: CpG islands: incorporating memory

- **Markov Chain**
  - \( Q \): states
  - \( p \): initial state probabilities
  - \( A \): transition probabilities

- **HMM**
  - \( Q \): states
  - \( V \): observations
  - \( p \): initial state probabilities
  - \( A \): transition probabilities
  - \( E \): emission probabilities

What have we learned?

- **Modeling sequential data**
  - Recognize a type of sequence, genomic, oral, verbal, visual, etc...

- **Definitions**
  - Markov Chains
  - Hidden Markov Models (HMMs)

- **Simple examples**
  - Recognizing GC-rich regions.
  - Recognizing CpG dinucleotides

- **Our first computations**
  - Running the model: know model \( \rightarrow \) generate sequence of a 'type'
  - Evaluation: know model, emissions, states \( \rightarrow \) \( p \)?
  - Viterbi: know model, emissions \( \rightarrow \) find optimal path
  - Forward: know model, emissions \( \rightarrow \) total \( p \) over all paths

- **Next time:**
  - Posterior decoding
  - Supervised learning
  - Unsupervised learning: Baum-Welch, Viterbi training