Lecture 4
Modeling Biological Sequences using Hidden Markov Models

Module 1: Aligning and modeling genomes

- Module 1: Computational foundations
  - Dynamic programming: exploring exponential spaces in poly-time
  - Linear-time string matching, Hashing, Content-based indexing
  - Hidden Markov Models: decoding, evaluation, parsing, learning
- Last week: Sequence alignment / comparative genomics
  - Local/global alignment: infer nucleotide-level evolutionary events
  - Database search: scan for regions that may have common ancestry
- This week: Modeling genomes / exon / CpG island finding
  - Modeling class of elements, recognizing members of a class
  - Application to gene finding, conservation islands, CpG islands

We have learned how to align sequences to other sequences

- L2: Sequence alignment
  - Dynamic programming, duality path ⇆ alignment
  - Global / local alignment, general gap penalties
- L3: Rapid string search
  - Exact string match, semi-numerical matching
  - Database search: Hashing, BLAST, variations
- L15: Comparative genomics: evolutionary signatures
  - Tell me how you evolve, I’ll tell you what you are
  - Identifying conserved elements through evolution
- L16: Whole-genome assembly/alignment/duplication:
  - Finding all common substrings within/across species
  - Contigs/scaffolds, string graphs, glocal alignment paths
- Problem set 1, project planning, Problem set 2 out

Today: apply these ideas to model DNA sequences...

- GTACTCACCGGGTTACAGGATTATGGGTTACAGGTAACCGTT...

Modeling biological sequences with HMMs
(a.k.a. What to do with big unlabeled chunks of DNA)

- 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?

- 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)

Markov Chains and Hidden Markov Models

Predicting tomorrow’s weather

- Markov Chain
- Hidden Markov Model

Components of a 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 \(s, t \in Q\).
  - For each \(s, t \in Q\) the transition probability is: \(a_{st} = P(x_t = t | x_{t-1} = s)\)

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 \(x_{t-1}\):
\[
P(x_t | x_{t-1}, \ldots, x_1) = P(x_t | x_{t-1}) \prod_{j=1}^{t-1} P(x_j | x_{j-1})
\]

Formula: The probability of the sequence:
\[
P(x) = P(x_1, x_2, \ldots, x_N) = P(x_1) P(x_2 | x_1) P(x_3 | x_2) \ldots P(x_N | x_{N-1})
\]

Components of an HMM (Hidden Markov Model)

Definition: An HMM is a 5-tuple \((Q, V, p, A, E)\), where:
- \(Q\) is a finite set of states, \(|Q| = N\)
- \(V\) is a finite set of observation symbols per state, \(|V| = M\)
- \(p\) 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: \(a_{st} = P(x_t = t | x_{t-1} = s)\)
- \(E\) is a probability emission matrix, \(e_{sk} = P(v_t = k | x_t = s)\)

Output: Only emitted symbols are observable by the system but not the underlying random walk between states \(\Rightarrow\) "hidden"

Property: Emissions and transitions are dependent on the current state only and not on the past.
### The six algorithmic settings for HMMs

<table>
<thead>
<tr>
<th>Scoring</th>
<th>Decoding</th>
<th>Learning</th>
</tr>
</thead>
</table>
| 1. Scoring x, one path  
\[ P(x, \pi) \]  
Prob of a path, emissions | 3. Viterbi decoding  
\[ \pi^* = \arg \max_{\pi} P(x, \pi) \]  
Most likely path | 5. Supervised learning, given \( \pi \)  
\[ \Lambda^* = \arg \max_{\lambda} P(x, \pi|\lambda) \] | 1. Scoring x, all paths  
\[ P(x) = \sum_{\pi} P(x, \pi) \]  
Prob of emissions, over all paths | 6. Unsupervised learning  
\[ \Lambda^* = \arg \max_{\lambda} \max_{\pi} P(x, \pi|\lambda) \]  
Viterbi training, best path |

### Examples of HMMs

The dishonest casino  
The dishonest genome  
... and many more

### Example: The Dishonest Casino

A casino has two dice:
- Fair die  
  \[ P(1) = P(2) = P(3) = P(5) = P(6) = 1/6 \]  
- Loaded die  
  \[ P(1) = P(2) = P(3) = P(5) = 1/10 \]  
  \[ P(6) = 0.5 \]

Casino player switches between fair and loaded die on average once every 20 turns.

**Game:**
1. You bet $1
2. You roll (always with a fair die)
3. Casino player rolls (maybe with fair die, maybe with loaded die)
4. Highest number wins $2

### The dishonest casino model

**Observed (world)**

**Hidden (model)**

### Examples of HMMs for genome annotation

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Topology / Transitions</td>
<td>2 states, different nucleotide composition</td>
<td>2 states, different conservation levels</td>
<td>2 states, different tri-nucleotide composition</td>
<td>2 states, different evolutionary signatures</td>
<td>~20 states, different composition/conservation, specific structure</td>
<td>40 states, different chromatin mark combinations</td>
</tr>
<tr>
<td>Hidden States / Annotation</td>
<td>GC-rich / AT-rich</td>
<td>Conserved / non-conserved Coding exon / non-coding (intron or intergenic) Coding exon / non-coding (intron or intergenic) First/last/middle coding exon, UTRs, intron1/2/3, intergenic, (+/- strand)</td>
<td>Enhancer / promoter / transcribed / repressed / repetitive</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Emissions / Observations</td>
<td>Nucleotides Level of conservation Triplet of nucleotides 64x64 matrix of codon substitution frequencies Codons, nucleotides, splice sites, start/stop codons</td>
<td>Vector of chromatin mark frequencies</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
What is the joint probability of observing \( x \) and a specific path \( \pi \): 
\[
P(\pi, x) = P(x|\pi)P(\pi) = P(\text{emissions} | \pi)P(\pi)
\]
and rolls
\[
x = 1, 2, 1, 5, 6, 2, 1, 6, 2, 4
\]
Joined probability 
\[
P(x, \pi) = \frac{1}{6} \times P(1 | \text{Fair}) P(\text{Fair} | 1) P(2 | \text{Fair}) P(\text{Fair} | 2) ... P(4 | \text{Fair}) = \frac{1}{6} \times (\frac{1}{10})^9 \times (0.95)^9 = 0.5 \times 10^{-9}
\]
Why is \( p \) so small?

Likelihood ratio:
\[
P(\pi, \text{all-Fair}) = 6.59 \times 10^{-10} \quad \text{(very very small)}
\]

It is 6.59 times more likely that the die is fair all the way, than loaded all the way.

Model comparison

Let the sequence of rolls be:
\[
x = 1, 6, 6, 5, 6, 2, 6, 6, 3, 6
\]
Now, what is the likelihood \( \pi = F, F, ..., F \)?
\[
\frac{1}{6} \times (\frac{1}{10})^9 \times (0.95)^9 = 0.5 \times 10^{-9} \quad \text{same as before}
\]
What is the likelihood \( \pi = L, L, ..., L \)?
\[
\frac{1}{2} \times (\frac{1}{10})^9 \times (\frac{1}{2})^6 \times (0.95)^9 = 0.5 \times 10^{-7}
\]
So, it is 100 times more likely the die is loaded.

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

3. Viterbi decoding
   \[\pi^* = \text{argmax}_\pi P(x, \pi)\]
   Most likely path

4. Posterior decoding
   \[\pi^* = \{\pi_i | \pi_i = \text{argmax}_\pi \sum \pi P(\pi | x)\}\]
   Path containing the most likely state at any time point.

**All paths**

5. Supervised learning, given \( \pi \)
   \[\Lambda^* = \text{argmax}_\lambda \sum \pi P(\pi | \lambda)\]
   Viterbi training, best path

6. Unsupervised learning
   \[\Lambda^* = \text{argmax}_\lambda \sum P(\pi | \lambda)\]
   Baum-Welch training, over all paths

Running the model: Probability of a sequence

Comparing the two paths

What about partial runs and die switching

The six algorithmic settings for HMMs

Model evaluation
3. DECODING:
What was the sequence of hidden states?

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

$\rightarrow$ Viterbi algorithm
  - Define $V_i(i) =$ Probability of the most likely path through state $\pi=i$
  - Compute $V_i(i+1) =$ as a function of $\max \{ V_i(i) \}$
  - $V_i(i+1) = e_i(x_{i+1}) \times \max_j a_{ij} V_j(i)$

$\rightarrow$ Dynamic Programming

Finding the most likely path

- Find path $\pi^*$ that maximizes total joint probability $P[ x, \pi ]$
  - $P(x, \pi) = a_{0i_1} \times \prod_i e_i(x_i) \times a_{i_{i+1}i}$

The Viterbi Algorithm

Input: $x = x_1 \ldots x_N$

Initialization:
$V_i(0) = 1, V_i(0) = 0$, for all $k > 0$

Iteration:
$V_i(i) = e_i(x_i) \times \max_j a_{ij} V_j(i-1) $

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

Traceback:
Follow max pointers back
Similar to aligning states to seq

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

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

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

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

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

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

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

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

Given a HMM, we can generate a sequence of length $n$ as follows:
1. Start at state $\pi_1$ according to prob $a_{01}$
2. Emit letter $x_1$ according to prob $e_1(x_1)$
3. Go to state $\pi_2$ according to prob $a_{12}$
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)?

- $P(x) = \sum_{\pi} P(x, \pi) = \sum_{\pi} P(x|\pi) P(\pi)$
- (weighted average of conditional probability, summed over all paths, weighted by each path’s probability)

- Challenge: exponential number of paths

The Forward Algorithm – derivation

Define the forward probability:

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

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

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

$$= \sum_{\pi_i = k} f(i-1)_k a_{kl} \phi(x_i)$$

$$= \phi(x_i) \sum_k f(i-1)_k a_{kl}$$

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

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

Calculate $f(i) = \phi(x_i) \sum_j (f(i-1)_j \times a_{jk})$

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

Calculate probability of emission over all paths

- Each path has associated probability
  - Some paths are likely, others unlikely: sum them all up
  - Return total probability that emissions are observed, summed over all paths
  - Viterbi path is the most likely one
    - How much ‘probability mass’ does it contain?

- (cheap) alternative:
  - Calculate probability over maximum (Viterbi) path $\pi^*$
  - Good approximation if Viterbi has highest density
  - BUT: incorrect

- (real) solution
  - Calculate the exact sum iteratively
    - $P(x) = \sum_\pi P(x, \pi)$
  - Can use dynamic programming
The Forward Algorithm

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

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

**Iteration:**
\[
f_k(i) = e_k(x_i) \sum_j a_{jk} f_{i-1}(j)
\]

**Termination:**
\[
P(x, \pi^*) = \sum_k f_k(N)
\]

**In practice:**

- Sum of log scores is difficult
- Approximate \( \exp(1+p+q) \)
- Scaling of probabilities

**Running time and space:**

- **Time:** \( O(KN) \)
- **Space:** \( O(KN) \)

---

### Examples of HMMs for genome annotation

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Topology / Transitions</td>
<td>2 states, different nucleotide composition</td>
<td>2 states, different conservation levels</td>
<td>2 states, different tri-nucleotide composition</td>
<td>~20 states, different evolutionary signatures</td>
<td>40 states, different chromatin mark combinations</td>
<td></td>
</tr>
<tr>
<td>Hidden States / Annotation</td>
<td>GC-rich / AT-rich</td>
<td>Conserved / non-conserved</td>
<td>Coding exon / non-coding (intronic or intergenic)</td>
<td>Coding exon / non-coding (intronic or intergenic)</td>
<td>First/last/middle coding exon, UTRs, intron1/2/3, intergenic, &quot;+/- strand&quot;</td>
<td>Enhancer / promoter / transcribed / repressed / repetitive</td>
</tr>
<tr>
<td>Emissions / Observations</td>
<td>Nucleotides</td>
<td>Level of conservation</td>
<td>Triplets of nucleotides</td>
<td>64x64 matrix of codon substitution frequencies</td>
<td>Codons, nucleotides, splice sites, start/stop codons</td>
<td>Vector of chromatin mark frequencies</td>
</tr>
</tbody>
</table>

---

### The six algorithmic settings for HMMs

<table>
<thead>
<tr>
<th>Scoring</th>
<th>Decoding</th>
<th>Learning</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Scoring ( x ), one path</td>
<td>3. Viterbi decoding</td>
<td>5. Supervised learning, given ( \pi^* )</td>
</tr>
<tr>
<td>( P(x, \pi) )</td>
<td>( \pi^* = \underset{\pi}{\text{argmax}} P(x, \pi) )</td>
<td>( \Lambda^* = \underset{\pi}{\text{argmax}} \sum_{\pi} P(x, \pi</td>
</tr>
<tr>
<td>Prob of a path, emissions</td>
<td>Most likely path</td>
<td>Path containing the most likely state at any time point.</td>
</tr>
<tr>
<td>Prob of emissions, all paths</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

### What have we learned?

- **Modeling sequential data**
  - Recognize a \textit{type} of sequence, genomic, oral, verbal, visual, etc...
- **Definitions**
  - Markov Chains
  - Hidden Markov Models (HMMs)
- **Examples of HMMs**
  - Recognizing GC-rich regions, preferentially-conserved elements, coding exons, protein-coding gene structures, chromatin states
- **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

---

### What have we learned?

- **Modeling sequential data**
  - Recognize a \textit{type} of sequence, genomic, oral, verbal, visual, etc...
- **Definitions**
  - Markov Chains
  - Hidden Markov Models (HMMs)
- **Examples of HMMs**
  - Recognizing GC-rich regions, preferentially-conserved elements, coding exons, protein-coding gene structures, chromatin states
- **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