10-701
Machine Learning

Hidden Markov models (HMMs)
What’s wrong with Bayesian networks

• Bayesian networks are very useful for modeling joint distributions
• But they have their limitations:
  - Cannot account for temporal / sequence models
  - DAG’s (no self or any other loops)

This is not a valid Bayesian network!
Hidden Markov models

• Model a set of observation with a set of hidden states
  - Robot movement
    Observations: range sensor, visual sensor
    Hidden states: location (on a map)
  - Speech processing
    Observations: sound signals
    Hidden states: parts of speech, words
  - Biology
    Observations: DNA base pairs
    Hidden states: Genes
Hidden Markov models

- Model a set of observation with a set of hidden states
  - Robot movement
    - Observations: range sensor, visual sensor
    - Hidden states: location (on a map)
  1. Hidden states generate observations
  2. Hidden states transition to other hidden states
Examples: Speech processing

<table>
<thead>
<tr>
<th>sil</th>
<th>acht</th>
<th>negen</th>
<th>sil</th>
<th>drie</th>
<th>een</th>
</tr>
</thead>
<tbody>
<tr>
<td>sil</td>
<td>spk</td>
<td>spk</td>
<td>sil</td>
<td>spk</td>
<td>spk</td>
</tr>
</tbody>
</table>
Example: Biological data

ATGAAGCTACTGTCTTCTATCGAACAAGCATGCG
ATATTTGCCGACTTAAAAAGCTCAAG
TGCTCCAAAGAAAAACCGAAGTGCCGAAGTGT
CTGAAGAACAACGTGGGAGTGTCGCTAC
TCTCCAAAACCAAAAAGGTCTCCGCTGACTAGG
GCACATCTGACAGAAGTGGAATCAAGG
CTAGAAAGACTGGAACAGCTATTTTCTACTGATTTT
TCCTCGAGAAGACCTTGACATGATT
# Contents

<table>
<thead>
<tr>
<th>Preface</th>
<th>page ix</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Introduction</td>
</tr>
<tr>
<td>1.1</td>
<td>Sequence similarity, homology, and alignment</td>
</tr>
<tr>
<td>1.2</td>
<td>Overview of the book</td>
</tr>
<tr>
<td>1.3</td>
<td>Probabilities and probabilistic models</td>
</tr>
<tr>
<td>1.4</td>
<td>Further reading</td>
</tr>
<tr>
<td>2</td>
<td>Pairwise alignment</td>
</tr>
<tr>
<td>2.1</td>
<td>Introduction</td>
</tr>
<tr>
<td>2.2</td>
<td>The scoring model</td>
</tr>
<tr>
<td>2.3</td>
<td>Alignment algorithms</td>
</tr>
<tr>
<td>2.4</td>
<td>Dynamic programming with more complex models</td>
</tr>
<tr>
<td>2.5</td>
<td>Heuristic alignment algorithms</td>
</tr>
<tr>
<td>2.6</td>
<td>Linear space alignments</td>
</tr>
<tr>
<td>2.7</td>
<td>Significance of scores</td>
</tr>
<tr>
<td>2.8</td>
<td>Deriving score parameters from alignment data</td>
</tr>
<tr>
<td>2.9</td>
<td>Further reading</td>
</tr>
<tr>
<td>3</td>
<td>Markov chains and hidden Markov models</td>
</tr>
<tr>
<td>3.1</td>
<td>Markov chains</td>
</tr>
<tr>
<td>3.2</td>
<td>Hidden Markov models</td>
</tr>
<tr>
<td>3.3</td>
<td>Parameter estimation for HMMs</td>
</tr>
<tr>
<td>3.4</td>
<td>HMM model structure</td>
</tr>
<tr>
<td>3.5</td>
<td>More complex Markov chains</td>
</tr>
<tr>
<td>3.6</td>
<td>Numerical stability of HMM algorithms</td>
</tr>
<tr>
<td>3.7</td>
<td>Further reading</td>
</tr>
<tr>
<td>4</td>
<td>Pairwise alignment using HMMs</td>
</tr>
<tr>
<td>4.1</td>
<td>Pair HMMs</td>
</tr>
<tr>
<td>4.2</td>
<td>The full probability of $x$ and $y$, summing over all paths</td>
</tr>
<tr>
<td>4.3</td>
<td>Suboptimal alignment</td>
</tr>
<tr>
<td>4.4</td>
<td>The posterior probability that $x_i$ is aligned to $y_j$</td>
</tr>
<tr>
<td>4.5</td>
<td>Pair HMMs versus FSAs for searching</td>
</tr>
</tbody>
</table>
Example: Gambling on dice outcome

- Two dices, both skewed (output model).
- Can either stay with the same dice or switch to the second dice (transition mode).

![Diagram of dice outcomes](image_url)
A Hidden Markov model

- A set of states \( \{s_1 \ldots s_n\} \)
  - In each time point we are in exactly one of these states denoted by \( q_t \)
- \( \Pi_i \), the probability that we \textit{start} at state \( s_i \)
- A transition probability model, \( P(q_t = s_i \mid q_{t-1} = s_j) \)
- A set of possible outputs \( \Sigma \)
  - At time \( t \) we emit a symbol \( \sigma \in \Sigma \)
- An emission probability model, \( p(o_t = \sigma \mid s_i) \)
The Markov property

- A set of states \( \{s_1 \ldots s_n\} \)
  - In each time point we are in exactly one of these states denoted by \( q_t \)
- \( \Pi_i \), the probability that we start at state \( s_i \)
- A transition probability model, \( P(q_t = s_i \mid q_{t-1} = s_j) \)

An important aspect of this definition is the Markov property: \( q_{t+1} \) is conditionally independent of \( q_{t-1} \) (and any earlier time points) given \( q_t \)

More formally \( P(q_{t+1} = s_i \mid q_t = s_j) = P(q_{t+1} = s_i \mid q_t = s_j, q_{t-1} = s_j) \)
What can we ask when using a HMM?

A few examples:

• “What dice is currently being used?”
• “What is the probability of a 6 in the next role?”
• “What is the probability of 6 in any of the next 3 roles?”
Inference in HMMs

• Computing $P(Q)$ and $P(q_t = s_i)$
  - If we cannot look at observations
• Computing $P(Q | O)$ and $P(q_t = s_i | O)$
  - When we have observation and care about the last state only
• Computing $\arg\max_Q P(Q | O)$
  - When we care about the entire path
What dice is currently being used?

- We played $t$ rounds so far
- We want to determine $P(q_t = A)$
- Let's assume for now that we cannot observe any outputs (we are blind folded)
- How can we compute this?
\[ P(q_t = A) ? \]

- Simple answer:
  Lets determine \( P(Q) \) where \( Q \) is any path that ends in \( A \)
  \[ Q = q_1, \ldots, q_{t-1}, A \]
  \[ P(Q) = P(q_1, \ldots, q_{t-1}, A) = P(A \mid q_1, \ldots, q_{t-1}) \cdot P(q_1, \ldots, q_{t-1}) = P(A \mid q_{t-1}) \cdot P(q_1, \ldots, q_{t-1}) = \ldots = P(A \mid q_{t-1}) \cdot P(q_2 \mid q_1) \cdot P(q_1) \]

Markov property!

Initial probability
\[ \mathbf{P}(q_t = A) ? \]

• Simple answer:
  1. Let's determine \( \mathbf{P}(Q) \) where \( Q \) is any path that ends in \( A \)
     \( Q = q_1, \ldots, q_{t-1}, A \)
     \[ \mathbf{P}(Q) = \mathbf{P}(q_1, \ldots, q_{t-1}, A) = \mathbf{P}(A | q_1, \ldots, q_{t-1}) \mathbf{P}(q_1, \ldots, q_{t-1}) = \]
     \[ \mathbf{P}(A | q_{t-1}) \mathbf{P}(q_1, \ldots, q_{t-1}) = \ldots = \mathbf{P}(A | q_{t-1}) \ldots \mathbf{P}(q_2 | q_1) \mathbf{P}(q_1) \]
  2. \( \mathbf{P}(q_t = A) = \Sigma \mathbf{P}(Q) \)
     where the sum is over all sets of \( t \) states that end in \( A \)
P(q_t = A)?

- Simple answer:
  1. Let's determine P(Q) where Q is any path that ends in A
     \[ Q = q_1, \ldots q_{t-1}, A \]
     \[ P(Q) = P(q_1, \ldots q_{t-1}, A) = P(A \mid q_1, \ldots q_{t-1}) P(q_1, \ldots q_{t-1}) = P(A \mid q_{t-1}) P(q_1, \ldots q_{t-1}) = \ldots = P(A \mid q_{t-1}) \ldots P(q_2 \mid q_1) P(q_1) \]

  2. \( P(q_t = A) = \sum P(Q) \)
     where the sum is over all sets of states that end in A

Q: How many sets Q are there?
A: A lot! \( 2^{t-1} \)
Not a feasible solution
\[ P(q_t = A), \text{ the smart way} \]

- Lets define \( p_t(i) \) as the probability of being in state \( i \) at time \( t \):
  \[ p_t(i) = p(q_t = s_i) \]
- We can determine \( p_t(i) \) by induction
  1. \( p_1(i) = \Pi_i \)
  2. \( p_t(i) = ? \)
P(q_t = A), the smart way

- Lets define $p_t(i) = \text{probability state } i \text{ at time } t = p(q_t = s_i)$
- We can determine $p_t(i)$ by induction
  1. $p_1(i) = \Pi_i$
  2. $p_t(i) = \sum_j p(q_t = s_i | q_{t-1} = s_j)p_{t-1}(j)$
P(q_t = A), the smart way

- Lets define p_t(i) = probability state i at time t = p(q_t = s_i)
- We can determine p_t(i) by induction
  1. p_1(i) = Π_i
  2. p_t(i) = Σ_j p(q_t = s_i | q_{t-1} = s_j)p_{t-1}(j)

This type of computation is called dynamic programming

Complexity: O(n^2*t)

Number of states in our HMM

<table>
<thead>
<tr>
<th>Time / state</th>
<th>t1</th>
<th>t2</th>
<th>t3</th>
</tr>
</thead>
<tbody>
<tr>
<td>s1</td>
<td>.3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>s2</td>
<td>.7</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Limit theorem for Markov transitions

- If we do not see any observations and if the transition matrix is strictly positive (no zeros) than:
  \[ \lim_{k \to \infty} (P^k)_{i,j} = \theta_j \]

- In other words, at the limit the starting point does not really matter and there is a fix probability for being at any state.
Inference in HMMs

- Computing $P(Q)$ and $P(q_t = s_i)$
- Computing $P(Q | O)$ and $P(q_t = s_i | O)$
- Computing $\arg\max_Q P(Q)$
But what if we observe outputs?

- So far, we assumed that we could not observe the outputs.
- In reality, we almost always can.

| v  | P(v |A) | P(v |B) |
|----|-------|-------|
| 1  | 0.3   | 0.1   |
| 2  | 0.2   | 0.1   |
| 3  | 0.2   | 0.1   |
| 4  | 0.1   | 0.2   |
| 5  | 0.1   | 0.2   |
| 6  | 0.1   | 0.3   |
But what if we observe outputs?

- So far, we assumed that we could not observe the outputs.
- In reality, we almost always can.

Does observing the sequence 5, 6, 4, 5, 6, 6 change our belief about the state?

| v | P(v |A) | P(v |B) |
|---|------|------|
| 1 | .3   | .1   |
| 2 | .2   | .1   |
| 3 | .2   | .1   |
| 4 | .1   | .2   |
| 5 | .1   | .2   |
| 6 | .1   | .3   |
P(q_t = A) when outputs are observed

• We want to compute P(q_t = A | O_1 ... O_t)
• For ease of writing we will use the following notations (commonly used in the literature)
• \( a_{j,i} = P(q_t = s_i | q_{t-1} = s_j) \)
• \( b_i(o_t) = P(o_t | s_i) \)
P(q_t = A) when outputs are observed

• We want to compute P(q_t = A | O_1 ... O_t)
• Let's start with a simpler question. Given a sequence of states Q, what is P(Q | O_1 ... O_t) = P(Q | O)?
  - It is pretty simple to move from P(Q) to P(q_t = A)
  - In some cases P(Q) is the more important question
    - Speech processing
    - NLP
\[ P(Q \mid O) \]

- We can use Bayes rule:

\[ P(Q \mid O) = \frac{P(O \mid Q)P(Q)}{P(O)} \]

Easy, \( P(O \mid Q) = P(o_1 \mid q_1) P(o_2 \mid q_2) \ldots P(o_t \mid q_t) \)
\[ P(Q \mid O) \]

- We can use Bayes rule:

\[
P(Q \mid O) = \frac{P(O \mid Q)P(Q)}{P(O)}
\]

Easy, \( P(Q) = P(q_1) P(q_2 \mid q_1) \ldots P(q_t \mid q_{t-1}) \)
We can use Bayes rule:

\[ P(Q \mid O) = \frac{P(O \mid Q)P(Q)}{P(O)} \]

Hard!
P(O)

• What is the probability of seeing a set of observations:
  - An important question in its own rights, for example classification using two HMMs
• Define $\alpha_t(i) = P(o_1, o_2 \ldots, o_t \land q_t = s_i)$
• $\alpha_t(i)$ is the probability that we:
  1. Observe $o_1, o_2 \ldots, o_t$
  2. End up at state $i$

How do we compute $\alpha_t(i)$?
Computing $\alpha_t(i)$

- $\alpha_1(i) = P(o_1 \land q_1 = i) = P(o_1 \mid q_1 = s_i)$$I_{i}$

We must be at a state in time t

Chain rule

Markov property
Example: Computing $\alpha_3(B)$

- We observed 2,3,6

$\alpha_1(A) = P(2 \land q_1 = A) = P(2 \mid q_1 = A) \Pi_A = .2 \cdot .7 = .14$, $\alpha_1(B) = .1 \cdot .3 = .03$

$\alpha_2(A) = \Sigma_{j=A,B} b_A(3) a_{j,A} \alpha_1(j) = .2 \cdot .8 \cdot .14 + .2 \cdot .2 \cdot .03 = 0.0236$, $\alpha_2(B) = 0.0052$

$\alpha_3(B) = \Sigma_{j=A,B} b_B(6) a_{j,B} \alpha_2(j) = .3 \cdot .2 \cdot 0.0236 + .3 \cdot .8 \cdot .0052 = 0.00264$

<table>
<thead>
<tr>
<th>$v$</th>
<th>$P(v \mid A)$</th>
<th>$P(v \mid B)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>.3</td>
<td>.1</td>
</tr>
<tr>
<td>2</td>
<td>.2</td>
<td>.1</td>
</tr>
<tr>
<td>3</td>
<td>.2</td>
<td>.1</td>
</tr>
<tr>
<td>4</td>
<td>.1</td>
<td>.2</td>
</tr>
<tr>
<td>5</td>
<td>.1</td>
<td>.2</td>
</tr>
<tr>
<td>6</td>
<td>.1</td>
<td>.3</td>
</tr>
</tbody>
</table>

$\Pi_A = 0.7$

$\Pi_b = 0.3$
Where we are

- We want to compute $P(Q \mid O)$
- For this, we only need to compute $P(O)$
- We know how to compute $\alpha_t(i)$

From now its easy

$\alpha_t(i) = P(o_1, o_2 \ldots, o_t \land q_t = s_i)$

so

$P(O) = P(o_1, o_2 \ldots, o_t) = \sum_i P(o_1, o_2 \ldots, o_t \land q_t = s_i) = \sum_i \alpha_t(i)$

note that

$p(q_t = s_i \mid o_1, o_2 \ldots, o_t) = \frac{\alpha_t(i)}{\sum_j \alpha_t(j)}$

$P(A \mid B) = \frac{P(A \land B)}{P(B)}$
Complexity

• How long does it take to compute \( P(Q \mid O) \)?
• \( P(Q) \): \( O(t) \)
• \( P(O \mid Q) \): \( O(t) \)
• \( P(O) \): \( O(n^2t) \)
Inference in HMMs

• Computing $P(Q)$ and $P(q_t = s_i)$

• Computing $P(Q | O)$ and $P(q_t = s_i | O)$

• Computing $\text{argmax}_Q P(Q)$
Most probable path

• We are almost done …
• One final question remains
  How do we find the most probable path, that is $Q^*$ such that

$$P(Q^* \mid O) = \arg\max_Q P(Q|O)?$$

• This is an important path
  - The words in speech processing
  - The set of genes in the genome
  - etc.
Example

- What is the most probable set of states leading to the sequence:
  
  $1,2,2,5,6,5,1,2,3$ ?

$\Pi_A = 0.7$

$\Pi_B = 0.3$

| v  | $P(v | A)$ | $P(v | B)$ |
|----|-----------|-----------|
| 1  | 0.3       | 0.1       |
| 2  | 0.2       | 0.1       |
| 3  | 0.2       | 0.1       |
| 4  | 0.1       | 0.2       |
| 5  | 0.1       | 0.2       |
| 6  | 0.1       | 0.3       |

\[ 0.8 \quad 0.2 \quad 0.8 \]

\[ 0.2 \quad 0.2 \]
Most probable path

\[
\text{arg max}_Q P(Q \mid O) = \text{arg max}_Q \frac{P(O \mid Q)P(Q)}{P(O)} = \text{arg max}_Q P(O \mid Q)P(Q)
\]

We will use the following definition:

\[
\delta_t(i) = \max_{q_1 \cdots q_{t-1}} p(q_1 \cdots q_{t-1} \land q_t = s_i \land O_1 \cdots O_t)
\]

In other words we are interested in the most likely path from 1 to t that:

1. Ends in $S_i$
2. Produces outputs $O_1 \ldots O_t$
Computing $\delta_t(i)$

$$\delta_1(i) = p(q_1 = s_i \land O_1) = p(q_1 = s_i)p(O_1 \mid q_1 = s_i) = \pi_i b_i(O_1)$$

$$\delta_t(i) = \max_{q_1 \ldots q_{t-1}} p(q_1 \ldots q_t \land q_t = s_i \land O_1 \ldots O_t)$$

Q: Given $\delta_t(i)$, how can we compute $\delta_{t+1}(i)$?

A: To get from $\delta_t(i)$ to $\delta_{t+1}(i)$ we need to

1. Add an emission for time $t+1$ ($O_{t+1}$)

2. Transition to state $s_i$

$$\delta_{t+1}(i) = \max_{q_1 \ldots q_t} p(q_1 \ldots q_t \land q_{t+1} = s_i \land O_1 \ldots O_{t+1})$$

$$= \max_j \delta_t(j) p(q_{t+1} = s_i \mid q_t = s_j) p(O_{t+1} \mid q_{t+1} = s_i)$$

$$= \max_j \delta_t(j) a_{j,i} b_i(O_{t+1})$$
The Viterbi algorithm

\[ \delta_{t+1}(i) = \max_{q_1 \ldots q_t} p(q_1 \ldots q_t \land q_{t+1} = s_i \land O_1 \ldots O_{t+1}) \]

\[ = \max_j \delta_t(j)p(q_{t+1} = s_i | q_t = s_j)p(O_{t+1} | q_{t+1} = s_i) \]

\[ = \max_j \delta_t(j)a_{j,i}b_i(O_{t+1}) \]

- Once again we use dynamic programming for solving \( \delta_t(i) \)
- Once we have \( \delta_t(i) \), we can solve for our \( P(Q^* | O) \)

By:

\[ P(Q^* | O) = \arg\max_Q P(Q | O) = \text{path defined by } \arg\max_j \delta_t(j), \]
**Inference in HMMs**

- Computing $P(Q)$ and $P(q_t = s_i)$  

- Computing $P(Q \mid O)$ and $P(q_t = s_i \mid O)$  

- Computing $\arg\max_Q P(Q)$
What you should know

• Why HMMs? Which applications are suitable?
• Inference in HMMs
  - No observations
  - Probability of next state w. observations
  - Maximum scoring path (Viterbi)
Computing $\alpha_t(i)$

\[
\alpha_t(i) = P(o_1, o_2, \ldots, o_t \land q_t = s_i)
\]

- $\alpha_1(i) = P(o_1 \land q_1 = i) = P(o_1 \mid q_1 = s_i) \Pi_i$

\[
\alpha_{t+1}(i) = P(O_1 \ldots O_{t+1} \land q_{t+1} = s_i) =
\]

\[
\sum_j P(O_1 \ldots O_t \land q_t = s_j \land O_{t+1} \land q_{t+1} = s_i) =
\]

\[
\sum_j P(O_{t+1} \land q_{t+1} = s_i \mid O_1 \ldots O_t \land q_t = s_j) P(O_1 \ldots O_t \land q_t = s_j) =
\]

\[
\sum_j P(O_{t+1} \land q_{t+1} = s_i \mid O_1 \ldots O_t \land q_t = s_j) \alpha_t(j) =
\]

\[
\sum_j P(O_{t+1} \mid q_{t+1} = s_i) P(q_{t+1} = s_i \mid q_t = s_j) \alpha_t(j) =
\]

\[
\sum_j b_i(O_{t+1}) a_{j,i} \alpha_t(j)
\]