Outline

Algorithms for Hidden Markov Models

Main HMM Algorithms
HMM as Parser
Viterbi Algorithm for HMMs
HMM as Language Model
HMM Learning: Fully Observed Case
Learning from Unlabeled Data
Hidden Markov Model

Model $\theta = \begin{cases} 
\pi_i & \text{probability of starting at state } i \\
 a_{i,j} & \text{probability of transition from state } i \text{ to state } j \\
 b_i(o) & \text{probability of output } o \text{ at state } i 
\end{cases}$

- killer
- crazy
- clown
- problem
- killer
- crazy
- clown
- problem
Hidden Markov Model Algorithms

- HMM as parser: compute the best sequence of states for a given observation sequence.
- HMM as language model: compute probability of given observation sequence.
- HMM as learner: given a corpus of observation sequences, learn its distribution, i.e. learn the parameters of the HMM from the corpus.
  - Learning from a set of observations with the sequence of states provided (states are not hidden) [Supervised Learning]
  - Learning from a set of observations without any state information. [Unsupervised Learning]
Outline

Algorithms for Hidden Markov Models
  Main HMM Algorithms
HMM as Parser
Viterbi Algorithm for HMMs
HMM as Language Model
HMM Learning: Fully Observed Case
Learning from Unlabeled Data
HMM as Parser

The task: for a given observation sequence find the most likely state sequence.
HMM as Parser

Find most likely sequence of states for *killer clown*

Score every possible sequence of states: AA, AN, NN, NA

- $P(\text{killer clown, AA}) = \pi_A \cdot b_A(\text{killer}) \cdot a_{A,A} \cdot b_A(\text{clown}) = 0.0$
- $P(\text{killer clown, AN}) = \pi_A \cdot b_A(\text{killer}) \cdot a_{A,N} \cdot b_N(\text{clown}) = 0.0$
- $P(\text{killer clown, NN}) = \pi_N \cdot b_N(\text{killer}) \cdot a_{N,N} \cdot b_N(\text{clown}) = 0.75 \cdot 0.3 \cdot 0.5 \cdot 0.4 = 0.045$
- $P(\text{killer clown, NA}) = \pi_N \cdot b_N(\text{killer}) \cdot a_{N,A} \cdot b_A(\text{clown}) = 0.0$

Pick the state sequence with highest probability (NN=0.045).
As we have seen, for input of length 2, and a HMM with 2 states there are $2^2$ possible state sequences.

In general, if we have $q$ states and input of length $T$ there are $q^T$ possible state sequences.

Using our example HMM, for input \textit{killer crazy clown problem} we will have $2^4$ possible state sequences to score.

Our naive algorithm takes exponential time to find the best state sequence for a given input.

The \textbf{Viterbi algorithm} uses dynamic programming to provide the best state sequence with a time complexity of $q^2 \cdot T$. 

Outline

Algorithms for Hidden Markov Models
  Main HMM Algorithms
  HMM as Parser
Viterbi Algorithm for HMMs
  HMM as Language Model
  HMM Learning: Fully Observed Case
  Learning from Unlabeled Data
Viterbi Algorithm for HMMs

- For input of length $T$: $o_1, \ldots, o_T$, we want to find the sequence of states $s_1, \ldots, s_T$
- Each $s_t$ in this sequence is one of the states in the HMM.
- So the task is to find the most likely sequence of states:

$$\arg\max_{s_1, \ldots, s_T} P(o_1, \ldots, o_T, s_1, \ldots, s_T)$$

- The Viterbi algorithm solves this by creating a table $V[s, t]$ where $s$ is one of the states, and $t$ is an index between $1, \ldots, T$. 
Consider the input \textit{killer crazy clown problem}

So the task is to find the most likely sequence of states:

$$\text{argmax}_{s_1, s_2, s_3, s_4} P(\text{killer crazy clown problem}, s_1, s_2, s_3, s_4)$$

A sub-problem is to find the most likely sequence of states for \textit{killer crazy clown}:

$$\text{argmax}_{s_1, s_2, s_3} P(\text{killer crazy clown}, s_1, s_2, s_3)$$
Viterbi Algorithm for HMMs

- In our example there are two possible values for $s_4$:

$$
\max P(\text{killer crazy clown problem}, s_1, s_2, s_3, s_4) = \max \left\{ \max P(\text{killer crazy clown problem}, s_1, s_2, s_3, N), \max P(\text{killer crazy clown problem}, s_1, s_2, s_3, A) \right\}
$$

- Similarly:

$$
\arg\max P(\text{killer crazy clown}, s_1, s_2, s_3) = \arg\max \left\{ \max P(\text{killer crazy clown}, s_1, s_2, N), \max P(\text{killer crazy clown}, s_1, s_2, A) \right\}
$$
Viterbi Algorithm for HMMs

Putting them together:

\[
P(\text{killer crazy clown problem}, s_1, s_2, s_3, N) = \max \left\{ P(\text{killer crazy clown, } s_1, s_2, N) \cdot a_{N,N} \cdot b_N(\text{problem}), P(\text{killer crazy clown, } s_1, s_2, A) \cdot a_{A,N} \cdot b_N(\text{problem}) \right\}
\]

\[
P(\text{killer crazy clown problem, } s_1, s_2, s_3, A) = \max \left\{ P(\text{killer crazy clown, } s_1, s_2, N) \cdot a_{N,A} \cdot b_A(\text{problem}), P(\text{killer crazy clown, } s_1, s_2, A) \cdot a_{A,A} \cdot b_A(\text{problem}) \right\}
\]

The best score is given by:

\[
\max_{s_1, \ldots, s_4} P(\text{killer crazy clown problem, } s_1, s_2, s_3, s_4) = \max_{N,A} \left\{ \max_{s_1,s_2,s_3} P(\text{killer crazy clown problem, } s_1, s_2, s_3, N), \max_{s_1,s_2,s_3} P(\text{killer crazy clown problem, } s_1, s_2, s_3, A) \right\}
\]
Viterbi Algorithm for HMMs

- Provide an index for each input symbol:
  1: killer 2: crazy 3: clown 4: problem

\[
V[N, 3] = \max_{s_1, s_2} P(\text{killer crazy clown}, s_1, s_2, N)
\]

\[
V[N, 4] = \max_{s_1, s_2, s_3} P(\text{killer crazy clown problem}, s_1, s_2, s_3, N)
\]

- Putting them together:

\[
V[N, 4] = \max \{ V[N, 3] \cdot a_{N,N} \cdot b_N(\text{problem}), V[A, 3] \cdot a_{A,N} \cdot b_N(\text{problem}) \}
\]

\[
V[A, 4] = \max \{ V[N, 3] \cdot a_{N,A} \cdot b_A(\text{problem}), V[A, 3] \cdot a_{A,A} \cdot b_A(\text{problem}) \}
\]

- The best score for the input is given by:

\[
\max \{ V[N, 4], V[A, 4] \}
\]

- To extract the best sequence of states we backtrack (same trick as obtaining alignments from minimum edit distance)
Viterbi Algorithm for HMMs

- For input of length $T$: $o_1, \ldots, o_T$, we want to find the sequence of states $s_1, \ldots, s_T$
- Each $s_t$ in this sequence is one of the states in the HMM.
- For each state $q$ we initialize our table: $V[q, 1] = \pi_q \cdot b_q(o_1)$
- Then compute recursively for $t = 1 \ldots T - 1$ for each state $q$:
  \[ V[q, t + 1] = \max_{q'} \left\{ V[q', t] \cdot a_{q', q} \cdot b_q(o_{t+1}) \right\} \]
- After the loop terminates, the best score is $\max_q V[q, T]$
Outline

Algorithms for Hidden Markov Models
- Main HMM Algorithms
- HMM as Parser
- Viterbi Algorithm for HMMs

HMM as Language Model
- HMM Learning: Fully Observed Case
- Learning from Unlabeled Data
HMM as Language Model

Find $P(\text{killer clown}) = \sum_y P(y, \text{killer clown})$

$P(\text{killer clown}) = P(\text{AA, killer clown}) + P(\text{AN, killer clown}) + P(\text{NN, killer clown}) + P(\text{NA, killer clown})$
HMM as Language Model

Consider the input *killer crazy clown problem*

So the task is to find the sum over all sequences of states:

\[
\sum_{s_1, s_2, s_3, s_4} P(\text{killer crazy clown problem, } s_1, s_2, s_3, s_4)
\]

A sub-problem is to find the most likely sequence of states for *killer crazy clown*:

\[
\sum_{s_1, s_2, s_3} P(\text{killer crazy clown, } s_1, s_2, s_3)
\]
In our example there are two possible values for $s_4$:

$$
\sum_{s_1, \ldots, s_4} P(killer\ crazy\ clown\ problem, s_1, s_2, s_3, s_4) =
\sum_{s_1, s_2, s_3} P(killer\ crazy\ clown\ problem, s_1, s_2, s_3, N) +
\sum_{s_1, s_2, s_3} P(killer\ crazy\ clown\ problem, s_1, s_2, s_3, A)
$$

Very similar to the Viterbi algorithm. Sum instead of max, and that’s the only difference!
HMM as Language Model

- Provide an index for each input symbol:
  1: killer 2: crazy 3: clown 4: problem

\[
V[N, 3] = \sum_{s_1, s_2} P(\text{killer crazy clown}, s_1, s_2, N)
\]

\[
V[N, 4] = \sum_{s_1, s_2, s_3} P(\text{killer crazy clown problem}, s_1, s_2, s_3, N)
\]

- Putting them together:

\[
V[N, 4] = V[N, 3] \cdot a_{N,N} \cdot b_N(\text{problem}) + V[A, 3] \cdot a_{A,N} \cdot b_N(\text{problem})
\]

\[
V[A, 4] = V[N, 3] \cdot a_{N,A} \cdot b_A(\text{problem}) + V[A, 3] \cdot a_{A,A} \cdot b_A(\text{problem})
\]

- The best score for the input is given by: \( V[N, 4] + V[A, 4] \)
HMM as Language Model

- For input of length $T$: $o_1, \ldots, o_T$, we want to find
  
  $$P(o_1, \ldots, o_T) = \sum_{y_1, \ldots, y_T} P(y_1, \ldots, y_T, o_1, \ldots, o_T)$$

- Each $y_t$ in this sequence is one of the states in the HMM.

- For each state $q$ we initialize our table: $V[q, 1] = \pi_q \cdot b_q(o_1)$

- Then compute recursively for $t = 1 \ldots T - 1$ for each state $q$:

  $$V[q, t + 1] = \sum_{q'} \{ V[q', t] \cdot a_{q', q} \cdot b_q(o_{t+1}) \}$$

- After the loop terminates, the best score is $\sum_q V[q, T]$

- So: Viterbi with sum instead of max gives us an algorithm for HMM as a language model.

- This algorithm is sometimes called the forward algorithm.
Outline

Algorithms for Hidden Markov Models
  Main HMM Algorithms
  HMM as Parser
  Viterbi Algorithm for HMMs
  HMM as Language Model
  HMM Learning: Fully Observed Case
  Learning from Unlabeled Data
HMM Learning from Labeled Data

Model $\theta = \begin{cases} 
\pi_i & \text{probability of starting at state } i \\
a_{i,j} & \text{probability of transition from state } i \text{ to state } j \\
b_i(o) & \text{probability of output } o \text{ at state } i 
\end{cases}$
The task: to find the values for the parameters of the HMM:

- $\pi_A, \pi_N$
- $a_{A,A}, a_{A,N}, a_{N,N}, a_{N,A}$
- $b_A(\text{killer}), b_A(\text{crazy}), b_A(\text{clown}), b_A(\text{problem})$
- $b_N(\text{killer}), b_N(\text{crazy}), b_N(\text{clown}), b_N(\text{problem})$
Learning from Fully Observed Data

- Labeled Data $L$:
  - $x_1, y_1$: killer/N clown/N \hspace{0.5cm} (x1 = killer, clown; y1 = N, N)
  - $x_2, y_2$: killer/N problem/N \hspace{0.5cm} (x2 = killer, problem; y2 = N, N)
  - $x_3, y_3$: crazy/A problem/N ... 
  - $x_4, y_4$: crazy/A clown/N
  - $x_5, y_5$: problem/N crazy/A clown/N
  - $x_6, y_6$: clown/N crazy/A killer/N
Learning from Fully Observed Data

- Let’s say we have \( m \) labeled examples:
  \[ L = (x_1, y_1), \ldots, (x_m, y_m) \]
- Each \((x_\ell, y_\ell) = \{o_1, \ldots, o_T, s_1, \ldots, s_T\}\)
- For each \((x_\ell, y_\ell)\) we can compute the probability using the HMM:
  
  \(\begin{align*}
  \text{\( (x_1 = \text{killer}, \text{clown}; y_1 = N, N) \)}: \\
  P(x_1, y_1) &= \pi_N \cdot b_N(\text{killer}) \cdot a_{N,N} \cdot b_N(\text{clown}) \\
  \text{\( (x_2 = \text{killer}, \text{problem}; y_2 = N, N) \)}: \\
  P(x_2, y_2) &= \pi_N \cdot b_N(\text{killer}) \cdot a_{N,N} \cdot b_N(\text{problem}) \\
  \text{\( (x_3 = \text{crazy}, \text{problem}; y_3 = A, N) \)}: \\
  P(x_3, y_3) &= \pi_A \cdot b_A(\text{crazy}) \cdot a_{A,N} \cdot b_N(\text{problem}) \\
  \text{\( (x_4 = \text{crazy}, \text{clown}; y_4 = A, N) \)}: \\
  P(x_4, y_4) &= \pi_A \cdot b_A(\text{crazy}) \cdot a_{A,N} \cdot b_N(\text{clown}) \\
  \text{\( (x_5 = \text{problem}, \text{crazy}, \text{clown}; y_5 = N, A, N) \)}: \\
  P(x_5, y_5) &= \pi_N \cdot b_N(\text{problem}) \cdot a_{N,A} \cdot b_A(\text{crazy}) \cdot a_{A,N} \cdot b_N(\text{clown}) \\
  \text{\( (x_6 = \text{clown}, \text{crazy}, \text{killer}; y_6 = A, A, N) \)}: \\
  P(x_6, y_6) &= \pi_N \cdot b_N(\text{clown}) \cdot a_{N,A} \cdot b_A(\text{crazy}) \cdot a_{A,N} \cdot b_N(\text{killer})
  \end{align*}\)

\[ \prod_{\ell} P(x_\ell, y_\ell) = \pi_N^4 \cdot \pi_A^2 \cdot a_{N,N}^2 \cdot a_{N,A}^2 \cdot a_{A,N}^4 \cdot a_{A,A}^0 \cdot b_N(\text{killer})^3 \cdot b_N(\text{clown})^4 \cdot b_N(\text{problem})^3 \cdot b_A(\text{crazy})^4 \]
Learning from Fully Observed Data

- We can easily collect frequency of observing a word with a state (tag)
  - $f(i, x, y) = \text{number of times } i \text{ is the initial state in } (x, y)$
  - $f(i, j, x, y) = \text{number of times } j \text{ follows } i \text{ in } (x, y)$
  - $f(i, o, x, y) = \text{number of times } i \text{ is paired with observation } o$
- Then according to our HMM the probability of $x, y$ is:

$$P(x, y) = \prod_i \pi_i^{f(i, x, y)} \cdot \prod_{i,j} a_{i,j}^{f(i, j, x, y)} \cdot \prod_{i,o} b_i(o)^{f(i, o, x, y)}$$
According to our HMM the probability of $x, y$ is:

$$P(x, y) = \prod_i \pi_i^{f(i,x,y)} \cdot \prod_{i,j} a_{i,j}^{f(i,j,x,y)} \cdot \prod_{i,o} b_i(o)^{f(i,o,x,y)}$$

For the labeled data $L = (x_1, y_1), \ldots, (x_\ell, y_\ell), \ldots, (x_m, y_m)$

$$P(L) = \prod_{\ell=1}^{m} P(x_\ell, y_\ell)$$

$$= \prod_{\ell=1}^{m} \left( \prod_i \pi_i^{f(i,x_\ell,y_\ell)} \cdot \prod_{i,j} a_{i,j}^{f(i,j,x_\ell,y_\ell)} \cdot \prod_{i,o} b_i(o)^{f(i,o,x_\ell,y_\ell)} \right)$$
Learning from Fully Observed Data

According to our HMM the probability of $x, y$ is:

$$P(L) = \prod_{\ell=1}^{m} \left( \prod_i \pi_i^f(i, x_\ell, y_\ell) \cdot \prod_{i,j} a_{i,j}^f(i,j, x_\ell, y_\ell) \cdot \prod_{i,o} b_{i}(o)^f(i, o, x_\ell, y_\ell) \right)$$

The log probability of the labeled data $(x_1, y_1), \ldots, (x_m, y_m)$ according to HMM with parameters $\theta$ is:

$$L(\theta) = \sum_{\ell=1}^{m} \log P(x_\ell, y_\ell)$$

$$= \sum_{\ell=1}^{m} \sum_i f(i, x_\ell, y_\ell) \log \pi_i + \sum_{i,j} f(i, j, x_\ell, y_\ell) \log a_{i,j} + \sum_{i,o} f(i, o, x_\ell, y_\ell) \log b_{i}(o)$$
Learning from Fully Observed Data

\[ L(\theta) = \sum_{\ell=1}^{m} \left( \sum_{i} f(i, x_\ell, y_\ell) \log \pi_i + \sum_{i,j} f(i, j, x_\ell, y_\ell) \log a_{i,j} + \sum_{i,o} f(i, o, x_\ell, y_\ell) \log b_{i,o} \right) \]

- \( L(\theta) \) is the probability of the labeled data \((x_1, y_1), \ldots, (x_m, y_m)\)
- We want to find a \( \theta \) that will give us the maximum value of \( L(\theta) \)
- We find the \( \theta \) such that \( \frac{dL(\theta)}{d\theta} = 0 \)
Learning from Fully Observed Data

\[ L(\theta) = \sum_{\ell=1}^{m} \left( \sum_i f(i, x_{\ell}, y_{\ell}) \log \pi_i + \sum_{i,j} f(i, j, x_{\ell}, y_{\ell}) \log a_{i,j} + \sum_{i,o} f(i, o, x_{\ell}, y_{\ell}) \log b_{i}(o) \right) \]

- The values of \( \pi_i, a_{i,j}, b_{i}(o) \) that maximize \( L(\theta) \) are:

\[
\begin{align*}
\pi_i &= \frac{\sum_{\ell} f(i, x_{\ell}, y_{\ell})}{\sum_{\ell} \sum_k f(k, x_{\ell}, y_{\ell})} \\
a_{i,j} &= \frac{\sum_{\ell} f(i, j, x_{\ell}, y_{\ell})}{\sum_{\ell} \sum_k f(i, k, x_{\ell}, y_{\ell})} \\
b_{i}(o) &= \frac{\sum_{\ell} f(i, o, x_{\ell}, y_{\ell})}{\sum_{\ell} \sum_{o' \in V} f(i, o', x_{\ell}, y_{\ell})}
\end{align*}
\]
Learning from Fully Observed Data

- Labeled Data:
  - x1, y1: killer/N clown/N
  - x2, y2: killer/N problem/N
  - x3, y3: crazy/A problem/N
  - x4, y4: crazy/A clown/N
  - x5, y5: problem/N crazy/A clown/N
  - x6, y6: clown/N crazy/A killer/N
The values of $\pi_i$ that maximize $L(\theta)$ are:

$$\pi_i = \frac{\sum_{\ell} f(i, x_\ell, y_\ell)}{\sum_{\ell} \sum_{k} f(k, x_\ell, y_\ell)}$$

$\pi_N = \frac{2}{3}$ and $\pi_A = \frac{1}{3}$ because:

$$\sum_{\ell} f(N, x_\ell, y_\ell) = 4$$

$$\sum_{\ell} f(A, x_\ell, y_\ell) = 2$$
The values of $a_{i,j}$ that maximize $L(\theta)$ are:

$$a_{i,j} = \frac{\sum_\ell f(i, j, x_\ell, y_\ell)}{\sum_\ell \sum_k f(i, k, x_\ell, y_\ell)}$$

$a_{N,N} = \frac{1}{2}$; $a_{N,A} = \frac{1}{2}$; $a_{A,N} = 1$ and $a_{A,A} = 0$ because:

$$\sum_\ell f(N, N, x_\ell, y_\ell) = 2 \quad \sum_\ell f(A, N, x_\ell, y_\ell) = 4$$
$$\sum_\ell f(N, A, x_\ell, y_\ell) = 2 \quad \sum_\ell f(A, A, x_\ell, y_\ell) = 0$$
Learning from Fully Observed Data

- The values of $b_i(o)$ that maximize $L(\theta)$ are:

$$ b_i(o) = \frac{\sum_{\ell} f(i, o, x_\ell, y_\ell)}{\sum_{\ell} \sum_{o' \in V} f(i, o', x_\ell, y_\ell)} $$

- $b_N(killer) = \frac{3}{10}$; $b_N(clown) = \frac{4}{10}$; $b_N(problem) = \frac{3}{10}$ and $b_A(crazy) = 1$ because:

$$ \sum_{\ell} f(N, killer, x_\ell, y_\ell) = 3 \quad \sum_{\ell} f(A, killer, x_\ell, y_\ell) = 0 $$
$$ \sum_{\ell} f(N, clown, x_\ell, y_\ell) = 4 \quad \sum_{\ell} f(A, clown, x_\ell, y_\ell) = 0 $$
$$ \sum_{\ell} f(N, crazy, x_\ell, y_\ell) = 0 \quad \sum_{\ell} f(A, crazy, x_\ell, y_\ell) = 4 $$
$$ \sum_{\ell} f(N, problem, x_\ell, y_\ell) = 3 \quad \sum_{\ell} f(A, problem, x_\ell, y_\ell) = 0 $$
Learning from Fully Observed Data

x1,y1: killer/N clown/N
x2,y2: killer/N problem/N
x3,y3: crazy/A problem/N
x4,y4: crazy/A clown/N
x5,y5: problem/N crazy/A clown/N
x6,y6: clown/N crazy/A killer/N

\[
\pi = \begin{pmatrix} A & N \\ 0.25 & 0.75 \end{pmatrix} \quad a = \begin{pmatrix} a_{i,j} & A & N \\ N & 0.5 & 0.5 \\ A & 0.0 & 1.0 \end{pmatrix} \quad b = \begin{pmatrix} b_i(o) & A & N \\ clown & 0.0 & 0.4 \\ killer & 0.0 & 0.3 \\ problem & 0.0 & 0.3 \\ crazy & 1.0 & 0.0 \end{pmatrix}
\]
Outline

Algorithms for Hidden Markov Models
- Main HMM Algorithms
- HMM as Parser
- Viterbi Algorithm for HMMs
- HMM as Language Model
- HMM Learning: Fully Observed Case
- Learning from Unlabeled Data
Learning from Unlabeled Data

- Unlabeled Data $U = x_1, \ldots, x_m$:
  - $x_1$: killer clown
  - $x_2$: killer problem
  - $x_3$: crazy problem
  - $x_4$: crazy clown

- $y_1, y_2, y_3, y_4$ are unknown.

- But we can enumerate all possible values for $y_1, y_2, y_3, y_4$

- For example, for $x_1$: killer clown
  - $x_1, y_1, 1$: killer/A clown/A $p_1 = \pi_A \cdot b_A(\text{killer}) \cdot a_{A,A} \cdot b_A(\text{clown})$
  - $x_1, y_1, 2$: killer/A clown/N $p_2 = \pi_A \cdot b_A(\text{killer}) \cdot a_{A,N} \cdot b_N(\text{clown})$
  - $x_1, y_1, 3$: killer/N clown/N $p_3 = \pi_N \cdot b_N(\text{killer}) \cdot a_{N,N} \cdot b_N(\text{clown})$
  - $x_1, y_1, 4$: killer/N clown/A $p_4 = \pi_N \cdot b_N(\text{killer}) \cdot a_{N,A} \cdot b_A(\text{clown})$
Assume some values for $\theta = \pi, a, b$

We can compute $P(y \mid x_\ell, \theta)$ for any $y$ for a given $x_\ell$

$$P(y \mid x_\ell, \theta) = \frac{P(x, y \mid \theta)}{\sum_{y'} P(x, y' \mid \theta)}$$

For example, we can compute $P(\text{NN} \mid \text{killer clown}, \theta)$ as follows:

$$\frac{\pi_N \cdot b_N(\text{killer}) \cdot a_{N,N} \cdot b_N(\text{clown})}{\sum_{i,j} \pi_i \cdot b_i(\text{killer}) \cdot a_{i,j} \cdot b_j(\text{clown})}$$

$P(y \mid x_\ell, \theta)$ is called the *posterior probability*
Learning from Unlabeled Data

- Compute the posterior for all possible outputs for each example in training:
  - For x1: killer clown
    - x1,y1,1: killer/A clown/A \( P(\text{AA} | \text{killer clown}, \theta) \)
    - x1,y1,2: killer/A clown/N \( P(\text{AN} | \text{killer clown}, \theta) \)
    - x1,y1,3: killer/N clown/N \( P(\text{NN} | \text{killer clown}, \theta) \)
    - x1,y1,4: killer/N clown/A \( P(\text{NA} | \text{killer clown}, \theta) \)
  - For x2: killer problem
    - x2,y2,1: killer/A problem/A \( P(\text{AA} | \text{killer problem}, \theta) \)
    - x2,y2,2: killer/A problem/N \( P(\text{AN} | \text{killer problem}, \theta) \)
    - x2,y2,3: killer/N problem/N \( P(\text{NN} | \text{killer problem}, \theta) \)
    - x2,y2,4: killer/N problem/A \( P(\text{NA} | \text{killer problem}, \theta) \)
  - Similarly for x3: crazy problem
  - And x4: crazy clown
Learning from Unlabeled Data

- For unlabeled data, the log probability of the data given $\theta$ is:

\[
L(\theta) = \sum_{\ell=1}^{m} \log \sum_{y} P(x_{\ell}, y | \theta)
\]

\[
= \sum_{\ell=1}^{m} \log \sum_{y} P(y | x_{\ell}, \theta) \cdot P(x_{\ell} | \theta)
\]

- Unlike the fully observed case there is no simple solution to finding $\theta$ to maximize $L(\theta)$

- We instead initialize $\theta$ to some values, and then iteratively find better values of $\theta$: $\theta^0, \theta^1, \ldots$ using the following formula:

\[
\theta^t = \arg\max_{\theta} Q(\theta, \theta^{t-1})
\]

\[
= \sum_{\ell=1}^{m} \sum_{y} P(y | x_{\ell}, \theta^{t-1}) \cdot \log P(x_{\ell}, y | \theta)
\]
Learning from Unlabeled Data

\[ \theta^t = \arg\max_{\theta} Q(\theta, \theta^{t-1}) \]

\[ Q(\theta, \theta^{t-1}) = \sum_{\ell=1}^{m} \sum_{y} P(y \mid x_\ell, \theta^{t-1}) \cdot \log P(x_\ell, y \mid \theta) \]

\[ = \sum_{\ell=1}^{m} \sum_{y} P(y \mid x_\ell, \theta^{t-1}) \cdot \left( \sum_{i} f(i, x_\ell, y) \cdot \log \pi_i \right) \]

\[ + \sum_{i,j} f(i, j, x_\ell, y) \cdot \log a_{i,j} \]

\[ + \sum_{i,o} f(i, o, x_\ell, y) \cdot \log b_i(o) \]
Learning from Unlabeled Data

\[ g(i, x_\ell) = \sum_y P(y \mid x_\ell, \theta^{t-1}) \cdot f(i, x_\ell, y) \]

\[ g(i, j, x_\ell) = \sum_y P(y \mid x_\ell, \theta^{t-1}) \cdot f(i, j, x_\ell, y) \]

\[ g(i, o, x_\ell) = \sum_y P(y \mid x_\ell, \theta^{t-1}) \cdot f(i, o, x_\ell, y) \]

\[ \theta^t = \arg\max_{\pi, a, b} \sum_{\ell=1}^{m} \sum_i g(i, x_\ell) \cdot \log \pi_i \]

\[ + \sum_{i,j} g(i, j, x_\ell) \cdot \log a_{i,j} \]

\[ + \sum_{i,o} g(i, o, x_\ell) \cdot \log b_{j(o)} \]
Learning from Unlabeled Data

\[ Q(\theta, \theta^{t-1}) = \sum_{\ell=1}^{m} \sum_{i} g(i, x_{\ell}) \log \pi_{i} + \sum_{i,j} g(i, j, x_{\ell}) \log a_{i,j} + \sum_{i,o} g(i, o, x_{\ell}) \log b_{i}(o) \]

▶ The values of \( \pi_{i}, a_{i,j}, b_{i}(o) \) that maximize \( L(\theta) \) are:

\[
\pi_{i} = \frac{\sum_{\ell} g(i, x_{\ell})}{\sum_{\ell} \sum_{k} g(k, x_{\ell})}
\]

\[
a_{i,j} = \frac{\sum_{\ell} g(i, j, x_{\ell})}{\sum_{\ell} \sum_{k} g(i, k, x_{\ell})}
\]

\[
b_{i}(o) = \frac{\sum_{\ell} g(i, o, x_{\ell})}{\sum_{\ell} \sum_{o' \in V} g(i, o', x_{\ell})}
\]
EM Algorithm for Learning HMMs

- Initialize $\theta^0$ at random. Let $t = 0$.
- The EM Algorithm:
  - E-step: compute expected values of $y$, $P(y \mid x, \theta)$ and calculate $g(i, x)$, $g(i, j, x)$, $g(i, o, x)$
  - M-step: compute $\theta^t = \arg\max_{\theta} Q(\theta, \theta^{t-1})$
  - Stop if $L(\theta^t)$ did not change much since last iteration. Else continue.
- The above algorithm is guaranteed to improve likelihood of the unlabeled data.
- In other words, $L(\theta^t) \geq L(\theta^{t-1})$
- But it all depends on $\theta^0$!