CMPT-413
Computational Linguistics

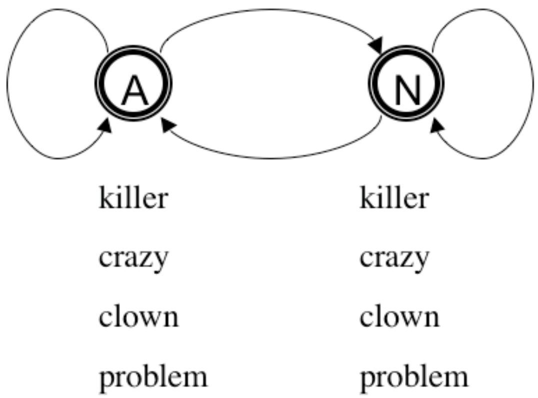
Anoop Sarkar
`http://www.cs.sfu.ca/~anoop`

March 7, 2012

Outline

Hidden Markov Model

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

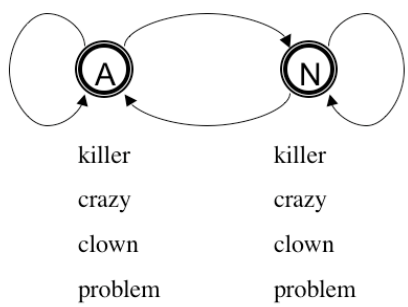


Hidden Markov Model Algorithms

- 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

HMM as Parser



$\pi =$

A	*N*
0.25	0.75

$a =$

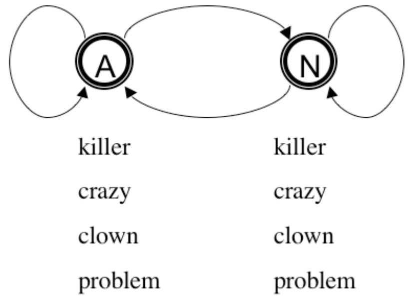
$a_{i,j}$	*A*	*N*
N	0.5	0.5
A	0.0	1.0

$b =$

$b_i(o)$	*A*	*N*
clown	0.0	0.4
killer	0.0	0.3
problem	0.0	0.3
crazy	1.0	0.0

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(killer clown, AA) $= \pi_A \cdot b_A(killer) \cdot a_{A,A} \cdot b_A(clown) = 0.0$
  - P(killer clown, AN) $= \pi_A \cdot b_A(killer) \cdot a_{A,N} \cdot b_N(clown) = 0.0$
  - P(killer clown, NN) $= \pi_N \cdot b_N(killer) \cdot a_{N,N} \cdot b_N(clown) = 0.75 \cdot 0.3 \cdot 0.5 \cdot 0.4 = 0.045$
  - P(killer clown, NA) $= \pi_N \cdot b_N(killer) \cdot a_{N,A} \cdot b_A(clown) = 0.0$
- Pick the state sequence with highest probability (NN=0.045).

HMM as Parser

- 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 *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 **Viterbi algorithm** uses dynamic programming to provide the best state sequence with a time complexity of $q^2 \cdot T$

Outline

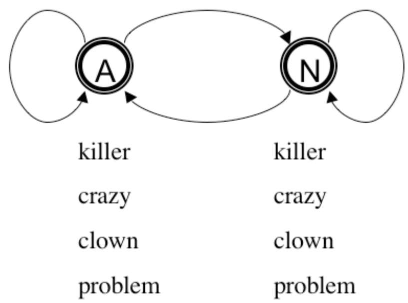
Viterbi Algorithm for HMMs

- 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:

$$\underset{s_1,\ldots,s_T}{\operatorname{argmax}} P(o_1, \ldots, o_T, s_1, \ldots, s_T)$$

- 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 $1, \ldots, T$.

Viterbi Algorithm for HMMs



- Consider the input *killer crazy clown problem*
- So the task is to find the most likely sequence of states:

$$\underset{s_1,s_2,s_3,s_4}{\operatorname{argmax}} P(\textit{killer crazy clown problem}, s_1, s_2, s_3, s_4)$$

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

$$\underset{s_1,s_2,s_3}{\operatorname{argmax}} P(\textit{killer crazy clown}, s_1, s_2, s_3)$$

Viterbi Algorithm for HMMs

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

$$\max_{s_1,\ldots,s_4} P(\textit{killer crazy clown problem}, s_1, s_2, s_3, s_4) =$$

$$\max\left\{\max_{s_1,s_2,s_3} P(\textit{killer crazy clown problem}, s_1, s_2, s_3, N),\right.$$

$$\left.\max_{s_1,s_2,s_3} P(\textit{killer crazy clown problem}, s_1, s_2, s_3, A)\right\}$$

- Similarly:

$$\underset{s_1,\ldots,s_3}{\operatorname{argmax}} P(\textit{killer crazy clown}, s_1, s_2, s_3) =$$

$$\underset{N,V}{\operatorname{argmax}}\left\{\max_{s_1,s_2} P(\textit{killer crazy clown}, s_1, s_2, N),\right.$$

$$\left.\max_{s_1,s_2} P(\textit{killer crazy clown}, s_1, s_2, A)\right\}$$

Viterbi Algorithm for HMMs

- Putting them together:

$$P(\textit{killer crazy clown problem}, s_1, s_2, s_3, N) = \\ \max\{P(\textit{killer crazy clown}, s_1, s_2, N) \cdot a_{N,N} \cdot b_N(\textit{problem}), \\ P(\textit{killer crazy clown}, s_1, s_2, A) \cdot a_{A,N} \cdot b_N(\textit{problem})\}$$

$$P(\textit{killer crazy clown problem}, s_1, s_2, s_3, A) = \\ \max\{P(\textit{killer crazy clown}, s_1, s_2, N) \cdot a_{N,A} \cdot b_A(\textit{problem}), \\ P(\textit{killer crazy clown}, s_1, s_2, A) \cdot a_{A,A} \cdot b_A(\textit{problem})\}$$

- The best score is given by:

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

Viterbi Algorithm for HMMs

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

$$V[N,3] = \max_{s_1,s_2} P(\textit{killer crazy clown}, s_1, s_2, N)$$
$$V[N,4] = \max_{s_1,s_2,s_3} P(\textit{killer crazy clown problem}, s_1, s_2, s_3, N)$$

- Putting them together:

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

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

- 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

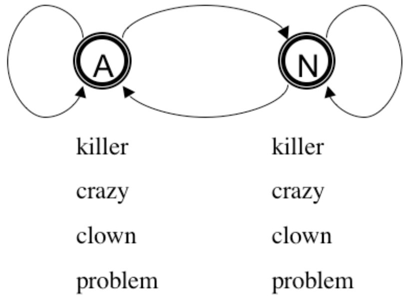
- For input of length T: $o_1, \ldots, o_T$, we want to find the sequence of states $s_1, \ldots, s_T$
- Each s_t in this sequence is one of the states in the HMM.
- For each state q we initialize our table: $V[q, 1] = \pi_q \cdot b_q(o_1)$
- Then compute recursively for $t = 1 \ldots T - 1$ for each 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 $\max_q V[q, T]$

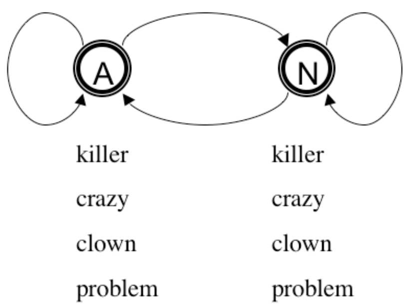
Outline

HMM as Language Model



- Find $P(\textit{killer clown}) = \sum_y P(y, \textit{killer clown})$
- $P(\textit{killer clown}) = P(AA, \textit{killer clown}) + P(AN, \textit{killer clown}) + P(NN, \textit{killer clown}) + P(NA, \textit{killer clown})$

HMM as Language Model



- Consider the input *killer crazy clown problem*
- So the task is to find the sum over all sequences of states:

$$\sum_{s_1,s_2,s_3,s_4} P(\textit{killer crazy clown problem}, s_1, s_2, s_3, s_4)$$

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

$$\sum_{s_1,s_2,s_3} P(\textit{killer crazy clown}, s_1, s_2, s_3)$$

HMM as Language Model

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

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

- 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

$$\begin{aligned} V[N,3] &= \sum_{s_1,s_2} P(\textit{killer crazy clown}, s_1, s_2, N) \\ V[N,4] &= \sum_{s_1,s_2,s_3} P(\textit{killer crazy clown problem}, s_1, s_2, s_3, N) \end{aligned}$$

- Putting them together:

$$\begin{aligned} V[N,4] &= V[N,3] \cdot a_{N,N} \cdot b_N(\textit{problem}) + \\ &\quad V[A,3] \cdot a_{A,N} \cdot b_N(\textit{problem}) \end{aligned}$$

$$\begin{aligned} V[A,4] &= V[N,3] \cdot a_{N,A} \cdot b_A(\textit{problem}) + \\ &\quad V[A,3] \cdot a_{A,A} \cdot b_A(\textit{problem}) \end{aligned}$$

- 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:

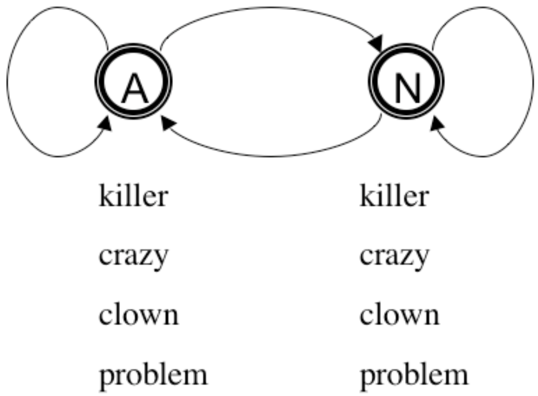
$$V[q, t+1] = \sum_{q'} \left\{ V[q', t] \cdot a_{q',q} \cdot b_q(o_{t+1}) \right\}$$

- 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*.

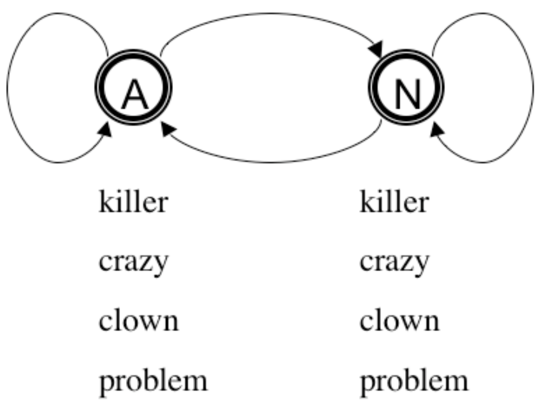
Outline

HMM Learning from Labeled Data

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



HMM Learning from Labeled Data



- The task: to find the values for the parameters of the HMM:
  - π_A, π_N
  - $a_{A,A}, a_{A,N}, a_{N,N}, a_{N,A}$
  - $b_A(killer), b_A(crazy), b_A(clown), b_A(problem)$
  - $b_N(killer), b_N(crazy), b_N(clown), b_N(problem)$

Learning from Fully Observed Data

- Labeled Data L:

```
x1,y1: killer/N clown/N       (x1 = killer,clown; y1 = N,N)
x2,y2: killer/N problem/N     (x2 = killer,problem; y2 = N,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
```


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_ℓ, y_ℓ) we can compute the probability using the HMM:
  - $(x_1 = killer, clown; y_1 = N, N)$: $P(x_1, y_1) = \pi_N \cdot b_N(killer) \cdot a_{N,N} \cdot b_N(clown)$
  - $(x_2 = killer, problem; y_2 = N, N)$: $P(x_2, y_2) = \pi_N \cdot b_N(killer) \cdot a_{N,N} \cdot b_N(problem)$
  - $(x_3 = crazy, problem; y_3 = A, N)$: $P(x_3, y_3) = \pi_A \cdot b_A(crazy) \cdot a_{A,N} \cdot b_N(problem)$
  - $(x_4 = crazy, clown; y_4 = A, N)$: $P(x_4, y_4) = \pi_A \cdot b_A(crazy) \cdot a_{A,N} \cdot b_N(clown)$
  - $(x_5 = problem, crazy, clown; y_5 = N, A, N)$: $P(x_5, y_5) = \pi_N \cdot b_N(problem) \cdot a_{N,A} \cdot b_A(crazy) \cdot a_{A,N} \cdot b_N(clown)$
  - $(x_6 = clown, crazy, killer; y_6 = A, A, N)$: $P(x_6, y_6) = \pi_N \cdot b_N(clown) \cdot a_{N,A} \cdot b_A(crazy) \cdot a_{A,N} \cdot b_N(killer)$
- $\prod_\ell P(x_\ell, y_\ell) = {\pi_N}^4 \cdot {\pi_A}^2 \cdot {a_{N,N}}^2 \cdot {a_{N,A}}^2 \cdot {a_{A,N}}^4 \cdot {a_{A,A}}^0 \cdot b_N(killer)^3 \cdot b_N(clown)^4 \cdot b_N(problem)^3 \cdot b_A(crazy)^4$

Learning from Fully Observed Data

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

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

Learning from Fully Observed Data

- 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 $L = (x_1, y_1), \ldots, (x_\ell, y_\ell), \ldots, (x_m, y_m)$

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

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 θ is:

$$\begin{aligned} L(\theta) &= \sum_{\ell=1}^{m} \log P(x_\ell, y_\ell) \\ &= \sum_{\ell=1}^{m} \sum_i f(i, x_\ell, y_\ell) \log \pi_i + \\ &\quad \sum_{i,j} f(i, j, x_\ell, y_\ell) \log a_{i,j} + \\ &\quad \sum_{i,o} f(i, o, x_\ell, y_\ell) \log b_i(o) \end{aligned}$$

Learning from Fully Observed Data

$$L(\theta) = \sum_{\ell=1}^{m} \sum_i f(i, x_\ell, y_\ell) \log \pi_i + \sum_{i,j} f(i, j, x_\ell, y_\ell) \log a_{i,j} + \sum_{i,o} f(i, o, x_\ell, y_\ell) \log b_i(o)$$

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

Learning from Fully Observed Data

$$L(\theta) = \sum_{\ell=1}^{m}$$
$$\sum_i f(i, x_\ell, y_\ell) \log \pi_i + \sum_{i,j} f(i, j, x_\ell, y_\ell) \log a_{i,j} + \sum_{i,o} f(i, o, x_\ell, y_\ell) \log b_i(o)$$

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

$$\begin{aligned}
\pi_i &= \frac{\sum_\ell f(i, x_\ell, y_\ell)}{\sum_\ell \sum_k f(k, x_\ell, y_\ell)} \\
a_{i,j} &= \frac{\sum_\ell f(i, j, x_\ell, y_\ell)}{\sum_\ell \sum_k f(i, k, x_\ell, y_\ell)} \\
b_i(o) &= \frac{\sum_\ell f(i, o, x_\ell, y_\ell)}{\sum_\ell \sum_{o' \in V} f(i, o', x_\ell, y_\ell)}
\end{aligned}$$

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
```

Learning from Fully Observed Data

- The values of π_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:

$$\begin{aligned} \sum_\ell f(N, x_\ell, y_\ell) &= 4 \\ \sum_\ell f(A, x_\ell, y_\ell) &= 2 \end{aligned}$$

Learning from Fully Observed Data

- 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:

$$\begin{aligned} \sum_\ell f(N, N, x_\ell, y_\ell) &= 2 & \sum_\ell f(A, N, x_\ell, y_\ell) &= 4 \\ \sum_\ell f(N, A, x_\ell, y_\ell) &= 2 & \sum_\ell f(A, A, x_\ell, y_\ell) &= 0 \end{aligned}$$

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:

$$\begin{aligned}
\sum_\ell f(N, killer, x_\ell, y_\ell) &= 3 & \sum_\ell f(A, killer, x_\ell, y_\ell) &= 0 \\
\sum_\ell f(N, clown, x_\ell, y_\ell) &= 4 & \sum_\ell f(A, clown, x_\ell, y_\ell) &= 0 \\
\sum_\ell f(N, crazy, x_\ell, y_\ell) &= 0 & \sum_\ell f(A, crazy, x_\ell, y_\ell) &= 4 \\
\sum_\ell f(N, problem, x_\ell, y_\ell) &= 3 & \sum_\ell f(A, problem, x_\ell, y_\ell) &= 0
\end{aligned}$$

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
```

$\pi =$

A	N
0.25	0.75

$a =$

$a_{i,j}$	A	N
N	0.5	0.5
A	0.0	1.0

$b =$

$b_i(o)$	A	N
clown	0.0	0.4
killer	0.0	0.3
problem	0.0	0.3
crazy	1.0	0.0

Outline

Learning from Unlabeled Data

- Unlabeled Data $U = x_1, \ldots, x_m$:

```
x1: killer clown
x2: killer problem
x3: crazy problem
x4: crazy clown
```

- `y1, y2, y3, y4` are unknown.
- But we can enumerate all possible values for `y1, y2, y3, y4`
- For example, for `x1: killer clown`

`x1,y1,1: killer/A clown/A` $p_1 = \pi_A \cdot b_A(killer) \cdot a_{A,A} \cdot b_A(clown)$
`x1,y1,2: killer/A clown/N` $p_2 = \pi_A \cdot b_A(killer) \cdot a_{A,N} \cdot b_N(clown)$
`x1,y1,3: killer/N clown/N` $p_3 = \pi_N \cdot b_N(killer) \cdot a_{N,N} \cdot b_N(clown)$
`x1,y1,4: killer/N clown/A` $p_4 = \pi_N \cdot b_N(killer) \cdot a_{N,A} \cdot b_A(clown)$

Learning from Unlabeled Data

- Assume some values for $\theta = \pi, a, b$
- We can compute $P(y \mid x_\ell, \theta)$ for any y for a given x_ℓ

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

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

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

- $P(y \mid x_\ell, \theta)$ 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` $P(\texttt{AA} \mid \texttt{killer clown}, \theta)$
  - `x1,y1,2: killer/A clown/N` $P(\texttt{AN} \mid \texttt{killer clown}, \theta)$
  - `x1,y1,3: killer/N clown/N` $P(\texttt{NN} \mid \texttt{killer clown}, \theta)$
  - `x1,y1,4: killer/N clown/A` $P(\texttt{NA} \mid \texttt{killer clown}, \theta)$
- For `x2`: `killer problem`
  - `x2,y2,1: killer/A problem/A` $P(\texttt{AA} \mid \texttt{killer problem}, \theta)$
  - `x2,y2,2: killer/A problem/N` $P(\texttt{AN} \mid \texttt{killer problem}, \theta)$
  - `x2,y2,3: killer/N problem/N` $P(\texttt{NN} \mid \texttt{killer problem}, \theta)$
  - `x2,y2,4: killer/N problem/A` $P(\texttt{NA} \mid \texttt{killer problem}, \theta)$
- Similarly for `x3`: `crazy problem`
- And `x4`: `crazy clown`

Learning from Unlabeled Data

- For unlabeled data, the log probability of the data given θ is:

$$
\begin{aligned}
L(\theta) &= \sum_{\ell=1}^{m} \log \sum_{y} P(x_\ell, y \mid \theta) \\
&= \sum_{\ell=1}^{m} \log \sum_{y} P(y \mid x_\ell, \theta) \cdot P(x_\ell \mid \theta)
\end{aligned}
$$

- Unlike the fully observed case there is no simple solution to finding θ to maximize $L(\theta)$
- We instead initialize θ to some values, and then iteratively find better values of θ: $\theta^0, \theta^1, \ldots$ using the following formula:

$$
\begin{aligned}
\theta^t &= \operatorname*{argmax}_{\theta} Q(\theta, \theta^{t-1}) \\
&= \sum_{\ell=1}^{m} \sum_{y} P(y \mid x_\ell, \theta^{t-1}) \cdot \log P(x_\ell, y \mid \theta)
\end{aligned}
$$

Learning from Unlabeled Data

$$
\begin{aligned}
\theta^t &= \operatorname*{argmax}_{\theta} Q(\theta, \theta^{t-1}) \\
Q(\theta, \theta^{t-1}) &= \sum_{\ell=1}^{m} \sum_{y} P(y \mid x_\ell, \theta^{t-1}) \cdot \log P(x_\ell, y \mid \theta) \\
&= \sum_{\ell=1}^{m} \sum_{y} P(y \mid x_\ell, \theta^{t-1}) \cdot \\
&\quad \left(\sum_{i} f(i, x_\ell, y) \cdot \log \pi_i \right. \\
&\quad + \sum_{i,j} f(i, j, x_\ell, y) \cdot \log a_{i,j} \\
&\quad \left. + \sum_{i,o} f(i, o, x_\ell, y) \cdot \log b_i(o) \right)
\end{aligned}
$$

Learning from Unlabeled Data

$$
\begin{aligned}
g(i, x_\ell) &= \sum_y P(y \mid x_\ell, \theta^{t-1}) \cdot f(i, x_\ell, y) \\
g(i, j, x_\ell) &= \sum_y P(y \mid x_\ell, \theta^{t-1}) \cdot f(i, j, x_\ell, y) \\
g(i, o, x_\ell) &= \sum_y P(y \mid x_\ell, \theta^{t-1}) \cdot f(i, o, x_\ell, y)
\end{aligned}
$$

$$
\begin{aligned}
\theta^t = \operatorname*{argmax}_{\pi, a, b} \sum_{\ell=1}^{m} & \sum_i g(i, x_\ell) \cdot \log \pi_i \\
& + \sum_{i,j} g(i, j, x_\ell) \cdot \log a_{i,j} \\
& + \sum_{i,o} g(i, o, x_\ell) \cdot \log b_j(o)
\end{aligned}
$$

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:

$$
\begin{aligned}
\pi_i &= \frac{\sum_\ell g(i, x_\ell)}{\sum_\ell \sum_k g(k, x_\ell)} \\
a_{i,j} &= \frac{\sum_\ell g(i, j, x_\ell)}{\sum_\ell \sum_k g(i, k, x_\ell)} \\
b_i(o) &= \frac{\sum_\ell g(i, o, x_\ell)}{\sum_\ell \sum_{o' \in V} g(i, o', x_\ell)}
\end{aligned}
$$

EM Algorithm for Learning HMMs

- Initialize θ^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 $\theta^t = \text{argmax}_\theta\, Q(\theta, \theta^{t-1})$
  - 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 θ^0!