I529: Machine Learning in Bioinformatics (Spring 2017)

HMM: Parameter Estimation

Yuzhen Ye
School of Informatics and Computing
Indiana University, Bloomington
Spring 2017

Content

- Review
  - HMM: three problems
  - The forward & backward algorithms; will be used again for the training of HMM
- When the training sequences are annotated (with known states)—MLE estimations
- When the states are unknown—Baum Welch training
  - An EM algorithm
  - E step—calculate A_{kl} and $E_k(b)$
  - M step

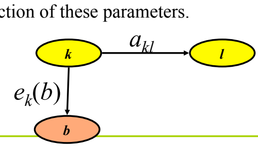
Parameters defining a HMM

HMM consists of:

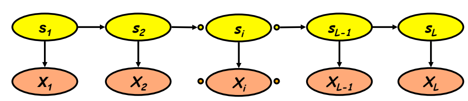
A Markov chain over a set of (hidden) states, and for each state s and observable symbol x, an emission probability $p(X_i=x|S_i=s)$.

An HMM model is defined by the ***parameters***: a_{kl} (transition probabilities) and $e_k(b)$ (emission probabilities), for all states k,l and all symbols b.

Let θ denote the collection of these parameters.



Parameter estimation for HMM



To determine the values of (the parameters in) θ, use a ***training set*** = $\{x^1,\dots,x^n\}$, where each x^j is a sequence which is assumed to fit the model.
Given the parameters θ, each sequence x^j has an assigned probability $p(x^j|\theta)$.

Data for HMM learning

To determine the values of (the parameters in) θ, use a ***training set*** = $\{x^1,\dots,x^n\}$, where each x^j is a sequence which is assumed to fit the model.
Given the parameters θ, each sequence x^j has an assigned probability $p(x^j|\theta)$.

Properties of (the sequences in) the training set:
1. For each x^j, the information on the states s^j_i
   The input sequences are annotated by the corresponding hidden sequences.
2. The size (number of sequences) of the training set

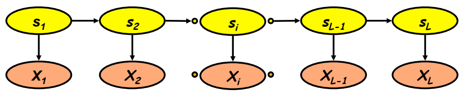
Maximum likelihood parameter estimation for HMM

The elements of the training set $\{x^1,\dots,x^n\}$, are assumed to be independent,
$p(x^1,\dots,x^n|\theta) = \prod_j p(x^j|\theta)$.

ML parameter estimation looks for θ which maximizes the above.

The exact method for finding or approximating this θ depends on the nature of the training set used.

Case 1: State paths are fully known



The training set $\{\mathbf{x^1},\ldots,\mathbf{x^n}\}$

By the ML method, we look for parameters θ^* (a_{kl} and $e_k(b)$) which maximize the probability of the sample set: $p(\mathbf{x^1},\ldots,\mathbf{x^n}|\ \theta^*) = \mathrm{MAX}_\theta\, p(\mathbf{x^1},\ldots,\mathbf{x^n}|\ \theta)$.

Case 1: State paths are fully known

For a sequence $\mathbf{x^j}$: $$p(x^j \mid \theta) = \prod_{i=1}^{L} a_{s_{i-1}s_i} e_{s_i}(x_i^j)$$

m_{kl}= #(transitions from k to l) in sequence $\mathbf{x^j}$.
$m_k(b)$=#(emissions of symbol b from state k) in sequence $\mathbf{x^j}$.

$$p(x^j \mid \theta) = \prod_{(k,l)} a_{kl}^{m_{kl}} \prod_{(k,b)} [e_k(b)]^{m_k(b)}$$

For the entire training set:
A_{kl} = #(transitions from k to l) in the training set.
$E_k(b)$ = #(emissions of symbol b from state k) in the training set.

We need to maximize: $$\prod_{(k,l)} a_{kl}^{A_{kl}} \prod_{(k,b)} [e_k(b)]^{E_k(b)}$$

Subject to: for all states k, $\sum_l a_{kl} = 1$, and $\sum_b \mathrm{e_k}(b) = 1$, $a_{kl}, e_k(b) \geq 0$.

MLE for *n* outcomes

The MLE is given by the relative frequencies:

$$\theta_i = \frac{n_i}{n} \quad i = 1,..,k$$

MLE applied to HMM

We apply the previous technique to get for each k the parameters $\{a_{kl}|l=1,..,m\}$ and $\{e_k(b)|b\in\Sigma\}$:

$$a_{kl} = \frac{A_{kl}}{\sum_{l'} A_{kl'}}, \text{ and } e_k(b) = \frac{E_k(b)}{\sum_{b'} E_k(b')}$$

Which gives the optimal ML parameters

Adding pseudo counts in HMM

If the sample set is too small, we may get a biased result. In this case we modify the actual count by our prior knowledge/belief:
r_{kl} is our prior belief and transitions from k to l.
$r_k(b)$ is our prior belief on emissions of b from state k.

$$\text{then } a_{kl} = \frac{A_{kl} + r_{kl}}{\sum_{l'}(A_{kl'} + r_{kl'})}, \text{ and } e_k(b) = \frac{E_k(b) + r_k(b)}{\sum_{b'}(E_k(b') + r_k(b))}$$

Fair casino problem: the sequences are annotated

- Consider the fair casino, where the dealer may use two coins (First and Second).
- HMM: the hidden states are {F(air), B(iased)}, observation symbols are {H(head), T(ail)}. We want to approximate the HMM parameters, the initial probabilities a_{0F} and a_{0B}, the transition probabilities a_{FF}, a_{FB}, a_{BF}, and a_{BB}, the emission probabilities $e_F(T)$, $e_F(H)$, $e_B(T)$ and $e_S(H)$.
- When the training set contains annotated sequences, we can simply compute the frequency for each of these cases to estimate the corresponding probabilities, which proved to be the Maximum Likelihood model parameters.

Fair casino problem: learning

Training sequences

```
Seq1                        Seq2
Obs:  TTHTHHTTHH      Obs:  THHTHHHHHHTTHH
Hid:  FFFFBBBBBB      Hid:  FFFFFBBBBBFFFF
```

MLE

a_{0F}=#F/2=1.0, a_{0B}=#B/2=0.0

a_{FF} = #(FF)/#(Fx) = 10/12 = 0.83; a_{FB} = #(FB)/#(Fx)=2/12=0.17

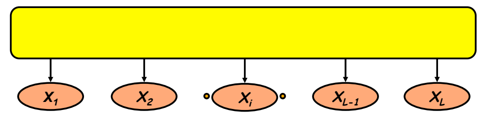
a_{BF} = #(BF)/#(Bx) = 1/10=0.1; a_{BB} = #(BB)/#(Bx)= 9/10=0.9

Fx means the di-hidden states with F as the first state.

e_F(T) = #(T,F)/#(F)= 7/13=0.53; e_F(H)= #(H,F)/#(F)=6/13=0.47

e_B(T) = #(T,B)/#(B) = 2/11=0.18; e_B(H) = #(H,B)/#(B)=9/11=0.82

Case 2: State paths are unknown



For a given θ we have:

$$p(\mathbf{x}^1,\ldots,\mathbf{x}^n|\theta)= p(\mathbf{x}^1|\theta)\cdots p(\mathbf{x}^n|\theta)$$

(since the $\mathbf{x}^j$ are independent)

For each sequence $\mathbf{x}$: $p(\mathbf{x}|\theta)=\sum_s p(\mathbf{x},\mathbf{s}|\theta)$

The sum taken over **all hidden state paths s!**

Finding θ^* which maximizes $\sum_s p(\mathbf{x},\mathbf{s}|\theta)$ is hard.

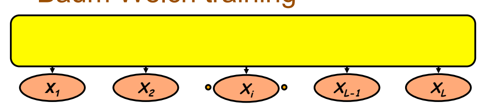
ML Parameter Estimation for HMM

The general process for finding θ in this case is

1. Start with an initial value of θ.
2. Find θ' so that $p(\mathbf{x}|\theta') > p(\mathbf{x}|\theta)$
3. set $\theta = \theta'$.
4. Repeat until some convergence criterion is met.

A general algorithm of this type is the Expectation Maximization algorithm, which we will meet later. For the specific case of HMM, it is the Baum-Welch training.

Baum Welch training



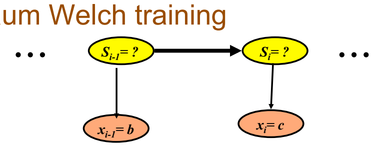
We start with some values of a_{kl} and $e_k(b)$, which define prior values of θ.

Then we use an iterative algorithm which attempts to replace θ by a θ^* s.t.

$$p(\mathbf{x}|\theta^*) > p(\mathbf{x}|\theta)$$

This is done by "imitating" the algorithm for Case 1, where all states are known:

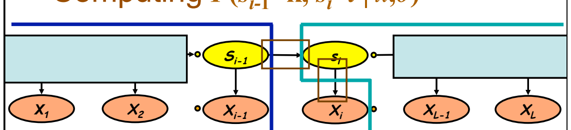
Baum Welch training



When the states are known, we can simply count. However, when the states are unknown, the "counting" process is a little trickier; instead, we use averaging process.

For each edge $s_{i-1} \rightarrow s_i$ we compute the **average** number of "k to l" transitions, for all possible pairs (k,l), over this edge. Then, for each k and l, we take A_{kl} to be the sum over all edges.

Computing $P(s_{i-1}=k, s_i=l \mid x,\theta)$



$$P(\underbrace{x_1,\ldots,x_L}_{\mathbf{x}},s_{i-1}=k,s_i=l|\theta) = P(x_1,\ldots,x_{i-1},s_{i-1}=k|\theta)\, a_{kl}e_l(x_i)\, P(x_{i+1},\ldots,x_L|s_i=l,\theta)$$

$$= f_k(i-1)\, a_{kl}e_l(x_i)\, b_l(i)$$

$$\boxed{p(s_{i-1}=k,s_i=l \mid \mathbf{x},\boldsymbol{\theta}) = \frac{f_k(i-1)\, a_{kl}e_l(x_i)\, b_l(i)}{p(x\mid\theta)}}$$

Compute A_{kl} for one sequence

For each pair (k,l), compute the expected number of state transitions from k to l, as the sum of the expected number of k to l transitions over all L edges :

$$A_{kl} = \frac{1}{p(x\,|\,\theta)} \sum_{i=1}^{L} p(s_{i-1} = k, s_i = l, x\,|\,\theta)$$

$$A_{kl} = \frac{1}{p(x\,|\,\theta)} \sum_{i=1}^{L} f_k(i-1)a_{kl}e_l(x_i)b_l(i)$$

Compute A_{kl} for many sequences

When we have n independent input sequences ($\mathbf{x}^1,\ldots,\mathbf{x}^n$), then A_{kl} is given by:

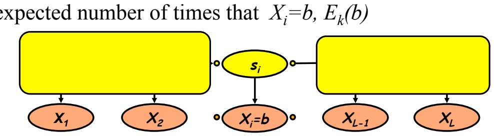
$$A_{kl} = \sum_{j=1}^{n} \frac{1}{p(x^j)} \sum_{i=1}^{L} p(s_{i-1}=k, s_i=l, x^j\,|\,\theta)$$

$$A_{kl} = \sum_{j=1}^{n} \frac{1}{p(x^j)} \sum_{i=1}^{L} f_k^j(i-1)a_{kl}e_l(x_i)b_l^j(i)$$

where f_k^j and b_l^j are the forward and backward algorithms for x^j under θ.

Compute expected number of symbol emissions

for state k and each symbol b, for each i, compute the expected number of times that $X_i=b$, $E_k(b)$



One edge

$$p(s_i = k \mid x_1,\ldots x_L)$$
$$= p(x_1\ldots x_L, s_i = k) / p(x_1,\ldots x_L)$$
$$= f_k(i)b_k(i) / p(x_1,\ldots,x_L)$$

One sequence

$$E_k(b) = \frac{1}{p(x\,|\,\theta)} \sum_{i:x_i=b} f_k(i)b_k(i)$$

(over all i's for which $x_i = b$)

n sequences

$$E_k(b) = \sum_{j=1}^{n} \frac{1}{p(x^j)} \sum_{i:x_i^j=b} f_k^j(i)f_k^j(i)$$

Summary of the E step

Task: compute the expected numbers A_{kl} of k,l transitions for all pairs of states k and l, and the expected numbers $E_k(b)$ of transmisions of symbol b from state k, for all states k and symbols b.

The next step is the M step, which is identical to the computation of optimal ML parameters when all states are known.

Baum Welch: M step

Use the A_{kl}' s, $E_k(b)$' s to compute the new values of a_{kl} and $e_k(b)$. These values define θ^*.

$$a_{kl} = \frac{A_{kl}}{\sum_{l'} A_{kl'}}, \text{ and } e_k(b) = \frac{E_k(b)}{\sum_{b'} E_k(b')}$$

The correctness of the EM algorithm implies that:

$$p(\mathbf{x}^1,\ldots,\mathbf{x}^n|\theta^*) \geq p(\mathbf{x}^1,\ldots,\mathbf{x}^n|\theta)$$

i.e, θ^* increases the probability of the data

This procedure is iterated, until some convergence criterion is met. Be aware of the local maximum (minimum) problem!

Viterbi learning

- Iterative improvement of model parameters (just like the Baum Welch algorithm)
- But only the single most likely hidden path is considered for each observation sequence on each iteration