STAT 534
Lecture 19
Markov Random Field and Baum-Welch Algorithm
June 4, 2019

mmp@stat.washington.edu
Scribes: Michael Hellstern, Qinqing Liao, Geoffrey Wang

1 Project

- Write project as if reader has prior knowledge. No need to explain α, β, etc
- May allow late projects by a few days
- Use any project outline you want

2 Iterative Proportional Fitting (IPF)

- Consider a Markov random field: $V = A, B, C, D$, $E = AB, AD, BC, DC$. We know that the joint probability can be broken down as follows:

$$P_v(a, b, c, d) = \frac{1}{Z}\phi_{AB}\phi_{BC}\phi_{CD}\phi_{DA}$$

- `IPF Algorithm`:

Given: Data $= x^1, x^2, ..., X^N$ samples;
Model = (V, ϵ) with maximal cliques $C, ... = \mathcal{C}$;
Each maximal clique has a relative probability function ϕ_c.

Initialization: $\phi_c > 0$, for $c \in \mathcal{C}$. $\hat{P}_c(X_c) = \frac{N_c(X_c)}{N}$ for $x_c \in \Omega_c, c \in \mathcal{C}$. These are emipirical marginals.

```
for t = 1,2,...
    for c ∈ 𝒞
        estimate P_c by MCMC
        φ_c^new = φ_c^current P̂_c / P_c
        run until convergence
```

- **Example**:

  Recall our clique marginals are $P_C(X_c) = \sum_{x\in\Omega, X_c=X_c^0} P_v(X)$.

  Using our P_{ABCD} model as an example, if we wanted to calculate the P_{AB}, the joint probability distribution of A, B we might do:

$$P_{AB}(1,1) = \sum_{c,d} P_{ABCD}(1,1,c,d)$$

$$P_{AB}(1,-1) = \sum_{c,d} P_{ABCD}(1,-1,c,d)$$

  For our initialization step, we would calculate $N_c(X_c)$ as something like:

$$N_{AD}(-1,-1) = \#\{a^i = d^i = -1\}$$

  At Maximum likelihood,

$$P_{AB}^{ML} = \hat{P}_{AB}$$

$$\phi_{AB}(1,1) = \phi_{AB}(1,1)\frac{\hat{P}_{AB}(1,1)}{P_{AB}(1,1)}$$

- **Theorem**: At maximum likelihood: $P_c = \hat{P}_c$ for all $c \in \mathcal{C}$. Note, we only match on things we parameterize in the model. That is, $P_{AC}^{ML} \neq \hat{P}_{AC}$ because we that is not something we are estimating in our model.

  Now let's compute the gradient of the joint probability: Start with log likelihood:

$$l = lnP_v(data)$$

  Then

$$\frac{1}{N}l = \sum_{i=1}^{n}[\sum_{c}\sum_{\Omega_c} ln\phi_c(x_c)\hat{P}_c(x_c)] - lnZ$$

- **Lemma 1**: Gives formula for partial derivative of the normalizing constant, Z, respect to ϕc.

$$\frac{dln(Z)}{d\phi_c(x_c)} = \frac{P_c(X_c)}{\phi_c(X_c)}$$

- **Lemma 2**: The partial derivative of the complete log likelihood is:

$$\frac{d}{d\phi_c(x_c)}(\frac{1}{N}l) = \frac{d}{d\phi_c(x_c)}[ln(\phi_c(x_c)\hat{P}_c(X_c) - lnZ] = \frac{\hat{P}_c(x_c) - P_c(x_c)}{\phi_c(x_c)}$$

Setting $\frac{d}{d\phi_c(x_c)}(\frac{1}{N}l) = 0$ we get $\frac{\hat{P}_c(x_c)}{\phi_c^{new}(x_c)} = \frac{P_c(x_c)}{\phi_c^{current}(x_c)}$ this is called fixed point iteration.

Note: $\phi_c^{new} \leftarrow \phi_c^{current} \frac{\hat{P}_c}{P_c}$

- **Lemma 3**: $P_c^{t+1} = \frac{Z^t}{Z_{t+1}} \hat{P}_c$.

  From Lemma 3 it follows that $\sum_{x_c \in \Omega} P_c^{t+1}(x_c) = 1 = \sum_{\Omega_c} \hat{P}_c(x_c)$ so this means: $Z^t = Z^{t+1}$

- Important notes:

  1. IPF algorithm maximizes likelihood of data w.r.t $\phi_c, c \in \mathcal{C}$.
  2. We obtain $\hat{P}_c(x_c)$ from the data. This is part of the initialization step.
  3. In our IPF algorithm, P_c must be estimated by MCMC. It is not known.
  4. This is a concave problem so the maximum we converge on is a global max!
  5. This algorithm converges fast relative to other methods.

3 HMM and Baum-Welch

- Convergence: Use $\frac{l^{t+1}}{l^t} - 1 \leq tol$
  Tol $= 10^{-4}$ is reasonable

- Consider our data with sequences sampled independently:

  seq 1: $O_1^1, O_2^1, ..., O_{T_1}^1$

  seq 2: $O_1^2, O_2^2, ..., O_{T_2}^2$

  ...

  Our model is always defined by $\lambda = (A, B, \pi)$. As usual, if we wanted $P(O_{1:T}|\lambda)$ we find it via forward-backward algorithm.

  Also note that

$$likelihood(data|\lambda) = \Pi_{k=1}^{n} P(O_{1:T_k}^k|\lambda)$$

$$\ell(\lambda) = \sum_k (lnP(O^k_{1:T_k})|\lambda)$$

The complete likelihood is given by

$$P(O_{1:T}, q_{1:T}|\lambda) = \pi(q_1)a_{q_1q_2}a_{q_2q_2}...b_{q_1O_1}b_{q_2O_2}...$$

The complete log likelihood is then:

$$\ell_c(\bar{\lambda}) = log(\pi(q_1)) + \sum_{t=1}^{T-1} log(a_{q_tq_{t+1}}) + \sum_{t=1}^{T} ln(b_{q_tO_t})$$

Define indicator variables $Z_t(i) = 1_{q_t=i}$ for $t = 1 : T, i = 1 : N$ and rewrite the previous equation using them:

$$\ell_c(\bar{\lambda}) = \sum_{i=1}^{N} z_1(i)ln(\bar{\pi}_i) + \sum_{t=1}^{T-1}\sum_{i=1}^{N}\sum_{j=1}^{N} z_t(i)z_{t+1}(j)ln(\bar{a_{ij}}) + \sum_{t=1}^{T}\sum_{i=1}^{N} z_t(i)lnb^{-}_{iO_t}$$

Idea of the Baum-Welch algorithm: Use current λ to estimate $E_\lambda(l_c) = Q(\lambda, \overline{\lambda})$

To start, lets find $E(Z)$s. $E_\lambda(Z_1(i)) = \gamma_1(i)$ because expected value of indicator is simply the probability of the event. We also have that: $E_\lambda(Z_t(i)Z_{t+1}(j)) = \xi_{ij}(t)$

Now we have all the information to do the expectation step:

$$E_\lambda(l_c(\overline{\lambda})) = \sum_i \gamma_1(i)ln(\overline{\pi}_i) + \sum_{t=1}^{T-1}\sum_{i,j} \xi_t(i,j)ln(\overline{a}_{ij}) + \sum_t\sum_i \gamma_t(i)ln(\overline{b}_{iO_t})$$

- The result computed was for one sequence, now to incorporate multiple sequences $k = 1 : n$

$$P(O^k_{1:t_k}, q^k_{1:T_k}, k = 1 : n) = \Pi^n_{k=1}P(O^k_{1:t_k}, q^k_{1:T_k})$$

$$\ell_c\bar{\lambda} = \sum_{k=1}^{n}[\sum_{i=1}^{N} z_1(i)ln(\bar{\pi}_i) + \sum_{t=1}^{T-1}\sum_{i=1}^{N}\sum_{j=1}^{N} z_t(i)z_{t+1}(j)ln(\bar{a_{ij}}) + \sum_{t=1}^{T}\sum_{i=1}^{N} z_t(i)lnb^{-}_{iO_t}]$$

$$Q(\lambda, \bar{\lambda}) = \sum_{k=1}^{n}[\sum_i \gamma^k_1(i)ln\bar{\pi}_i + \sum_t\sum_{ij} \xi^k_t(i,j)ln\bar{a_{ij}} + \sum_t\sum_i \xi^k_t(i)ln\bar{b}_{iO^k_t}]$$

So taking the log and Q we just sum over the k values and the maximization will follow.