CSE 250A: Assignment 3

Jiaxu Zhu A53094655

October 19, 2015

3.1 Inference

(a) Suppose that $P(X_t = j|X_1 = i) = [A^{t-1}]_{ij}$ is true for $t \geq 2$, then we prove that $P(X_{t+1} = j|X_1 = i) = [A^t]_{ij}$ is also true:

$$\begin{aligned}
P(X_{t+1} = j|X_1 = i) &= \sum_{k=1}^{m} P(X_{t+1} = j, X_t = k|X_1 = i) \quad (marginalization) \\
&= \sum_{k=1}^{m} P(X_{t+1} = j|X_t = k, X_1 = i)P(X_t = k|X_1 = i) \quad (product\ rule) \\
&= \sum_{k=1}^{m} P(X_{t+1} = j|X_t = k)P(X_t = k|X_1 = i) \quad (d - separationI) \\
&= \sum_{k=1}^{m} A_{kj}[A^{t-1}]_{ik} \\
&= [A^t]_{ij}
\end{aligned}$$

For $t = 2$, we have $P(X_2 = j|X_1 = i) = A_{ij}$, therefore we can say that $P(X_{t+1} = j|X_1 = i) = [A^t]_{ij}$ is true for $t \geq 1$;

(b) As we all know, for matrix multiplication $AA, A \in \mathbb{R}^{m\times m}$ is usually done in $O(m^3)$. But we find that, for a given i, j, we only needs the j-th column of $A^t, t \geq 1$(or i-th, here we pick j-th column) for further computation. Therefore we propose a simple algorithm show in Alg. 1.

As we can see, A' is the j-th column of A, so the running time of one matrix multiplication is $O(m^2)$. And we do the multiplication t times, so the overall running time is $O(m^2t)$.

Algorithm 1 3.1(b) Inference

```
function INFERENCE(A,i,j,t)
    A' = A_{*j}
    for iter = 1 to t do
        A' = [A × A']_{*j}
    return A'_i
```

(c) We also notice that to get A^t, we can always compute it using $A, A^2, A^4, ..., A^{2^n}, ...$ according to the binary form of t, instead of doing matrix multiplication t times. we shows the algorithm in Alg. 2

As we can see, so the running time of one matrix multiplication is $O(m^3)$. And we do the multiplication $\log_2 t$ times, so the overall running time is $O(m^3 \log_2 t)$.

(d) when matrix A is sparse, knowing positions of notn-zero elements helps. Suppose that non-zero elemonts in row i is store in a list P_i in the form of {j,value}.Then we slightly modify the Alg. 1 to get the Alg. 3

Algorithm 2 3.1(c) Inference

```
function INFERENCE(A,i,j,t)
    R = I
    while t > 0 do
        if t mod 2 = 1 then
            R = R × A
        A = A × A
        t = ⌊t/2⌋
    return R_ij
```

As we can see, A' is the j-th column of A, so the running time of one matrix multiplication is $O(sm)$. Because there are at most s non-zero elements per row. And we do the multiplication t times, so the overall running time is $O(smt)$.

Algorithm 3 3.1(d) Inference

```
function INFERENCE(A,P,i,j,t)
    A' = A_*j
    for iter = 1 to t do
        tmp = vector(m, 1)
        for row = 1 to m do
            tmp_row = 0
            for each p in P_row do
                tmp_row = tmp_row + p.value × A'_p.j
        A' = tmp
    return A'_j
```

3.2 Stochastic simulation

(a)

$$\begin{aligned}
\sum_{z\in[-\infty,+\infty]} P(Z=z|B_1,B_2,...,B_n) &= \sum_{z\in[-\infty,+\infty]} \left(\frac{1-\alpha}{1+\alpha}\right)\alpha^{|Z-f(B)|} \\
&= \left(\frac{1-\alpha}{1+\alpha}\right)\left(\alpha^0+2\sum_{k=1}^{\infty}\alpha^k\right) \\
&= \left(\frac{1-\alpha}{1+\alpha}\right)\left(1+2\lim_{k\to\infty}\frac{\alpha-\alpha^{k+1}}{1-\alpha}\right) \\
&= \lim_{k\to\infty}\left(1-\frac{2\alpha^{k+1}}{1+\alpha}\right) \\
&= 1
\end{aligned}$$

(b)

$$P(B_7=1|Z=64)=0.74$$

(c) As shown in Fig. 1, we plot estimated probability every 2×10^4 samples and we have 1×10^6 samples in total. And we can tell that our estimate has converged to a good degree of precision (two significant digits).

3.3 Node clustering

CPTs for the polytree is shown in Tab. 1.

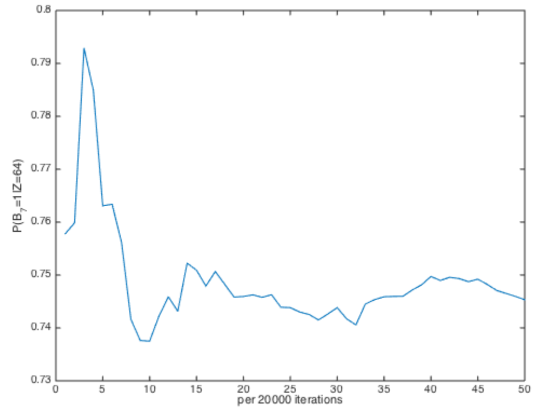

Figure 1: $P(B_7 = 1|Z = 64)$ as a function of the number of samples

Y_1	Y_2	Y_3	Y	$P(Y\|X = 0)$	$P(Y\|X = 1)$	$P(Z_1 = 1\|Y)$	$P(Z_2 = 1\|Y)$
0	0	0	1	0.09375	0.09375	0.9	0.1
1	0	0	2	0.28125	0.09375	0.8	0.2
0	1	0	3	0.09375	0.03125	0.7	0.3
0	0	1	4	0.03125	0.28125	0.6	0.4
1	1	0	5	0.28125	0.03125	0.5	0.5
1	0	1	6	0.09375	0.28125	0.4	0.6
0	1	1	7	0.03125	0.09375	0.3	0.7
1	1	1	8	0.09375	0.09375	0.2	0.8

Table 1: CPTs for the polytree

3.4 Maximum likelihood estimation

(a)

$$P(X_{t+1} = x'|X_t = x) = \frac{COUNT_t(x, x')}{COUNT_t(x)} \quad 1 \le t < T$$

(b)

$$P(X_t = x|X_{t+1} = x') = \frac{COUNT_t(x, x')}{COUNT_{t+1}(x')} \quad 1 \le t < T$$

(c) We first derive joint distribution from G1:

$$\begin{aligned} P(X_1 = x_1, X_2 = x_2, ..., X_T = x_T) &= P(X_1 = x_1) \prod_{i=2}^{T} P(X_i = x_i|X_1 = x_1, ..., X_{i-1} = x_{i-1}) \;\; (productrule) \\ &= P(X_1 = x_1) \prod_{i=2}^{T} P(X_i = x_i|X_{i-1} = x_{i-1}) \;\; (d - separationI) \\ &= \frac{COUNT_1(x_1)}{|data|} \prod_{i=1}^{T-1} \frac{COUNT_i(x_i, x_{i+1})}{COUNT_i(x_i)} \\ &= \frac{\prod_{i=1}^{T-1} COUNT_i(x_i, x_{i+1})}{|data| \prod_{i=2}^{T-1} COUNT_i(x_i)} \end{aligned}$$

Token	$P_u(w)$
MILLION	0.002073
MORE	0.001709
MR.	0.001442
MOST	0.000788
MARKET	0.000780
MAY	0.000730
M.	0.000703
MANY	0.000697
MADE	0.000560
MUCH	0.000515
MAKE	0.000514
MONTH	0.000445
MONEY	0.000437
MONTHS	0.000406
MY	0.000400
MONDAY	0.000382
MAJOR	0.000371
MILITARY	0.000352
MEMBERS	0.000336
MIGHT	0.000274
MEETING	0.000266
MUST	0.000267
ME	0.000264
MARCH	0.000260
MAN	0.000253
MS.	0.000239
MINISTER	0.000240
MAKING	0.000212
MOVE	0.000210
MILES	0.000206

Table 2: Tokens that start with the letter M and their numerical unigram probabilities

where $|data|$ is the size of the data set. Then we derive joint distribution from G2:

$$\begin{aligned}
P(X_1 = x_1, X_2 = x_2, ..., X_T = x_T) &= P(X_T = x_T) \prod_{i=1}^{T-1} P(X_i = x_i | X_{i+1} = x_{i+1}, ..., X_T = x_T) \quad (productrule) \\
&= P(X_T = x_T) \prod_{i=1}^{T-1} P(X_i = x_i | X_{i+1} = x_{i+1}) \quad (d - separationI) \\
&= \frac{COUNT_T(x_T)}{|data|} \prod_{i=1}^{T-1} \frac{COUNT_i(x_i, x_{i+1})}{COUNT_{i+1}(x_{i+1})} \\
&= \frac{\prod_{i=1}^{T-1} COUNT_i(x_i, x_{i+1})}{|data| \prod_{i=2}^{T-1} COUNT_i(x_i)}
\end{aligned}$$

We find the derived joint distribution from G1 and G2 is the same.

3.5 Statistical language modeling

(a) The results are displayed in Tab. 2.

(b) The ten most likely words to follow the word THE, along with their numerical bigram probabilities, are shown in Tab. 3.

Token	$P_b(w\|\mathbf{THE})$
⟨UNK ⟩	0.615020
U.	0.013372
FIRST	0.011720
COMPANY	0.011659
NEW	0.009451
UNITED	0.008672
GOVERNMENT	0.006803
NINETEEN	0.006651
SAME	0.006287
TWO	0.006161

Table 3: 10 most probable token after **THE**

(c)

$$\begin{aligned}
\mathcal{L}_u &= \log[P_u(\mathbf{the})P_u(\mathbf{stock})P_u(\mathbf{market})...P_u(\mathbf{points})P_u(\mathbf{last})P_u(\mathbf{week})] \\
&= -64.5094403436 \\
\mathcal{L}_b &= \log[P_b(\mathbf{the}|\langle \mathbf{s}\rangle)P_b(\mathbf{stock}|\mathbf{the})P_b(\mathbf{market}|\mathbf{stock})...P_b(\mathbf{last}|\mathbf{points})P_b(\mathbf{week}|\mathbf{last})] \\
&= -40.9181321338
\end{aligned}$$

Bigram model yields the highest log-likelihood.

(d)

$$\begin{aligned}
\mathcal{L}_u &= \log[P_u(\mathbf{the})P_u(\mathbf{sixteen})P_u(\mathbf{officials})...P_u(\mathbf{sold})P_u(\mathbf{fire})P_u(\mathbf{insurance})] \\
&= -44.2919344731 \\
\mathcal{L}_b &= \log[P_b(\mathbf{the}| < \mathbf{s} >)P_b(\mathbf{sixteen}|\mathbf{the})P_b(\mathbf{officials}|\mathbf{sixteen})...P_b(\mathbf{fire}|\mathbf{sold})P_b(\mathbf{insurance}|\mathbf{fire})] \\
&= \log(0.0)
\end{aligned}$$

When the pairs (**sisteen, officials**) and (**sold, fire**) are not observed in the training corpus? This makes the estimated probability to log(0), which is meaningless.

(e) Fig. 2 shows the value of this log-likelihood $\mathcal{L}_m$ as a function of the parameter $\lambda \in [0, 1]$. And the optimal λ is 0.65 with probability -42.9642.

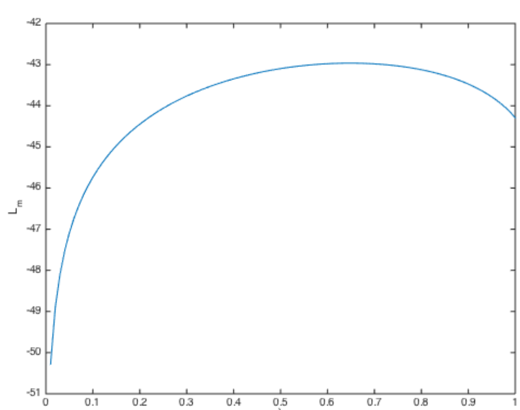

Figure 2: log-likelihood function $\mathcal{L}_m$