深蓝学院
shenlanxueyuan.com

《语音识别：从入门到精通》
第四章作业讲解

助教 彭震东

HMM模型三个基本问题

1. **概率计算问题**
   - 已知模型 $\lambda = (A, B, \pi)$ 和观测序列 $O = (o_1, o_2, \dots, o_T)$
   - 计算概率 $P(O|\lambda)$

   → 前向算法、后向算法

2. **预测问题（解码问题）**
   - 已知模型 $\lambda = (A, B, \pi)$ 和观测序列 $O = (o_1, o_2, \dots, o_T)$
   - 计算使概率 $P(I|O)$ 最大的状态序列 $I = (i_1, i_2, \dots, i_T)$

   → 维特比算法

3. **学习问题**
   - 已知观测序列 $O = (o_1, o_2, \dots, o_T)$
   - 估计模型 λ，使 $P(O|\lambda)$ 最大

   → 维特比学习算法
   Baum-Welch学习算法

前向算法

- **前向概率定义**：给定隐马尔可夫模型 λ ，定义到时刻 t 部分观测序列为 $o_1, o_2, \dots, o_t$ 且状态为 q_i 的概率为前向概率，记作（可省略 λ ）

$$\alpha_t(i) = P(o_1, o_2, \dots, o_t, i_t = q_i | \lambda)$$

- **算法 10.2（观测序列概率的前向算法）**

  输入：隐马尔可夫模型 λ，观测序列 O；

  输出：观测序列概率 $P(O|\lambda)$．

（1）初值

$$\alpha_1(i) = \pi_i b_i(o_1), \quad i = 1, 2, \cdots, N \qquad (10.15)$$

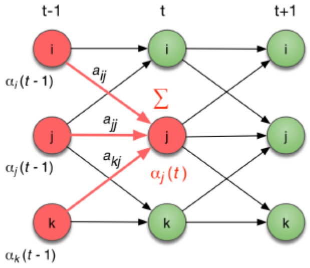
（2）递推　对 $t = 1, 2, \cdots, T-1$，

$$\alpha_{t+1}(i) = \left[\sum_{j=1}^{N} \alpha_t(j) a_{ji} \right] b_i(o_{t+1}), \quad i = 1, 2, \cdots, N \qquad (10.16)$$

（3）终止

$$P(O|\lambda) = \sum_{i=1}^{N} \alpha_T(i) \qquad (10.17)$$

■



14

时间复杂度：O(TN^2)

空间复杂度：O(NT)

作业代码（前向算法）

深蓝学院
shenlanxueyuan.com

```python
pi, A, B = HMM_model
T = len(O)
N = len(pi)
prob = 0.0
# Begin Assignment

alpha = [[0] * T for _ in range(N)]

# Initialize dp(alpha)
for j in range(N):
    alpha[j][0] = pi[j] * B[j][O[0]]

# following the formula
for i in range(1, T):
    for j in range(N):
        tmp = 0
        for k in range(N):
            tmp += alpha[k][i-1] * A[k][j]
        alpha[j][i] = tmp * B[j][O[i]]

# calculate the prob of observation from first column
for j in range(N):
    prob += alpha[j][T-1]
```

$$\alpha_1(i) = \pi_i b_i(o_1), \qquad i = 1,2,\cdots,N$$

递推　对 $t = 1,2,\cdots,T-1$，

$$\alpha_{t+1}(i) = \left[\sum_{j=1}^{N} \alpha_t(j) a_{ji}\right] b_i(o_{t+1}), \qquad i = 1,2,\cdots,N$$

$$P(O \mid \lambda) = \sum_{i=1}^{N} \alpha_T(i)$$

后向算法

后向概率定义：给定隐马尔可夫模型 λ ，定义在时刻 t状态为 q_i 的条件下，从 $t+1$ 到 T 的部分观测序列为 $o_{t+1}, o_{t+2}, \dots, o_T$ 的概率为后向概率，记作（可省略 λ ）

$$\beta_t(i) = P(o_{t+1}, o_{t+2}, \dots, o_T | i_t = q_i, \lambda)$$

算法 10.3 (观测序列概率的后向算法)

输入：隐马尔可夫模型 λ，观测序列 O；

输出：观测序列概率 $P(O|\lambda)$.

（1）

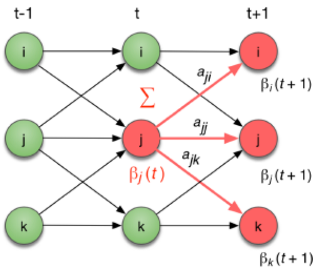
$$\beta_T(i) = 1, \quad i = 1, 2, \cdots, N \tag{10.19}$$

（2）对 $t = T-1, T-2, \cdots, 1$

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(o_{t+1}) \beta_{t+1}(j), \quad i = 1, 2, \cdots, N \tag{10.20}$$

（3）

$$P(O|\lambda) = \sum_{i=1}^{N} \pi_i b_i(o_1) \beta_1(i) \tag{10.21}$$ ■



时间复杂度：O(TN^2)

空间复杂度：O(NT)

作业代码（后向算法）

深蓝学院 shenlanxueyuan.com

```python
pi, A, B = HMM_model
T = len(O)
N = len(pi)
prob = 0.0
# Begin Assignment

beta = [[0] * T for _ in range(N)]

# Initialize backward dp(Beta)
for j in range(N):
    beta[j][T-1] = 1

# following the formula
for i in range(T-2, -1, -1):
    for j in range(N):
        tmp = 0
        for k in range(N):
            tmp += A[j][k] * B[k][O[i+1]] * beta[k][i+1]
        beta[j][i] = tmp

# calculate the prob of observation from first column
for j in range(N):
    prob += pi[j] * B[j][O[0]] * beta[j][0]
```

$$\beta_T(i)=1, \quad i=1,2,\cdots,N$$

对 $t=T-1,T-2,\cdots,1$

$$\beta_t(i)=\sum_{j=1}^{N}a_{ij}b_j(o_{t+1})\beta_{t+1}(j), \quad i=1,2,\cdots,N$$

$$P(O\mid\lambda)=\sum_{i=1}^{N}\pi_i b_i(o_1)\beta_1(i)$$

维特比算法

首先导入两个变量δ和ψ．定义在时刻t状态为i的所有单个路径$(i_1,i_2,\cdots,i_t)$中概率最大值为

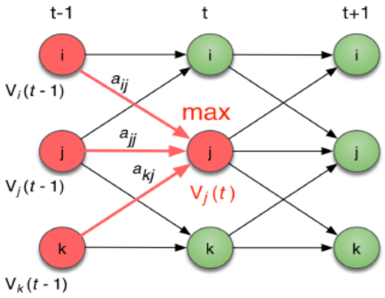
$$\delta_t(i)=\max_{i_1,i_2,\cdots,i_{t-1}} P(i_t=i,i_{t-1},\cdots,i_1,o_t,\cdots,o_1\mid\lambda),\quad i=1,2,\cdots,N \tag{10.44}$$

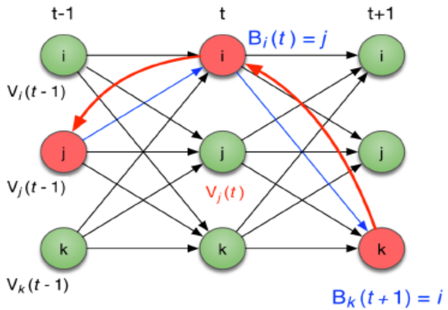
由定义可得变量δ的递推公式：

$$\begin{aligned}\delta_{t+1}(i)&=\max_{i_1,i_2,\cdots,i_t} P(i_{t+1}=i,i_t,\cdots,i_1,o_{t+1},\cdots,o_1\mid\lambda)\\&=\max_{1\leqslant j\leqslant N}[\delta_t(j)a_{ji}]b_i(o_{t+1}),\quad i=1,2,\cdots,N;\ t=1,2,\cdots,T-1\end{aligned} \tag{10.45}$$

定义在时刻t状态为i的所有单个路径$(i_1,i_2,\cdots,i_{t-1},i)$中概率最大的路径的第$t-1$个结点为

$$\psi_t(i)=\arg\max_{1\leqslant j\leqslant N}[\delta_{t-1}(j)a_{ji}],\quad i=1,2,\cdots,N \tag{10.46}$$





作业代码（维特比算法）

```python
pi, A, B = HMM_model
T = len(O)
N = len(pi)
best_prob = 0.0
best_path = [0] * T
# Begin Assignment

delta = [[0] * T for _ in range(N)]
phi = [[0] * T for _ in range(N)]

# Initialize for both dp and the best path record matrix
for j in range(N):
    delta[j][0] = pi[j] * B[j][O[0]]
    phi[j][0] = -1

# calculate the max prob for each time and for each state
# store the best state that goes form t to t+1
for i in range(1, N):
    for j in range(N):
        tmp = []
        for k in range(N):
            tmp.append(delta[k][i-1] * A[k][j])
        max_i, max_v = max_(tmp)
        delta[j][i] = max_v * B[j][O[i]]
        phi[j][i] = max_i

tmp = []
for j in range(N):
    tmp.append(delta[j][T-1])
```

delta: 记录最大概率

phi: 记录最优路径

$$\delta_{t+1}(i) = \max_{i_1, i_2, \cdots, i_t} P(i_{t+1} = i, i_t, \cdots, i_1, o_{t+1}, \cdots, o_1 \mid \lambda)$$
$$= \max_{1 \leqslant j \leqslant N} [\delta_t(j) a_{ji}] b_i(o_{t+1}), \qquad i = 1, 2, \cdots, N;\ t = 1, 2, \cdots, T-1$$

$$\psi_t(i) = \arg\max_{1 \leqslant j \leqslant N} [\delta_{t-1}(j) a_{ji}], \qquad i = 1, 2, \cdots, N$$

作业代码（维特比算法）

深蓝学院 shenlanxueyuan.com

```python
tmp = []
for j in range(N):
    tmp.append(delta[j][T-1])

# get the best state for final point
m_i, m_v = max_(tmp)
best_prob = m_v
best_path[T-1] = m_i

# backtrack for the rest time
for i in range(T-2, -1, -1):
    best_path[i] = phi[best_path[i+1]][i+1]

# transform the state as index in python start from "0"
best_path = [(val+1) for val in best_path]
```

回溯过程：
寻找最优路径的终点
从t=T-1回溯到t=0

一个小细节：
状态索引从"1"开始，
python数组索引从"0"开始，
需要做一个调整

参考资料

- 第四章课件：HMM算法

深蓝学院
shenlanxueyuan.com

感谢各位聆听！

Thanks for Listening