ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO ĐỒ ÁN MÔN HỌC

TOÁN ỨNG DỤNG VÀ THỐNG KÊ

Chủ đề: Mô hình Markov ẩn

Giảng viên lý thuyết: PGS. TS. Nguyễn Đình Thúc
Giảng viên hướng dẫn thực hành:

- Thầy Nguyễn Văn Quang Huy
- Cô Võ Nam Thục Đoan

Lớp: 20TN
Thành viên thực hiện:

- 20120131 – Nguyễn Văn Lộc
- 20120536 – Võ Trọng Nghĩa
- 20120572 – Nguyễn Kiều Minh Tâm

THÀNH PHỐ HỒ CHÍ MINH, THÁNG 4-5 NĂM 2022

Lời nói đầu

Mục lục

1 Lý thuyết mô hình Markov ẩn

Mô hình Markov ẩn (Hidden Markov model – HMM) là một mô hình máy học cổ điển thông dụng trong việc xử lý chuỗi.

1.1 Các thành phần của một mô hình Markov ẩn là gì? Chúng khác gì với mô hình Markov?

Một mô hình Markov ẩn có cấu tạo như sau[angrybird, hmm2021]:

- $Q = q_1 q_2 \dots q_N$: tập hợp N trạng thái (states).
- $A = a_{11} \dots a_{ij} \dots a_{NN}$: ma trận xác suất chuyển (transition probability matrix) A, a_{ij} là xác suất chuyển từ trạng thái i sang trạng thái j, $\sum_{j=1}^{N} a_{ij} = 1, \forall i$.
- $O = o_1 o_2 \dots o_T$: chuỗi các quan sát (observations), lấy từ bộ từ vựng (vocabulary) $V = v_1, v_2, \dots, v_V$.
- $B = b_i(o_t)$: ma trận xác suất phụ thuộc trạng thái (emission probabilities), thể hiện xác suất một quan sát o_t được tạo thành từ trạng thái i.
- $\pi = \pi_1, \pi_2, \dots, \pi_N$: phân phối xác suất ban đầu theo trạng thái, có nghĩa là π_i thể hiện xác suất xích Markov bắt đầu ở trạng thái i. Một số trạng thái j có thể có $\pi_j = 0$, do chúng không thể là trạng thái ban đầu của xích Markov.

So với xích Markov (Markov chains), mô hình Markov ẩn có thêm 2 thành phần là O – chuỗi các quan sát và B – ma trận xác suất phụ thuộc trạng thái.

1.2 Các giả thiết (assumptions) đặt ra cho mô hình Markov ẩn là gì? Tìm ví dụ các bài toán mà các giả thiết này hợp lý và bất hợp lý.

Các giả thiết (assumptions) đặt ra cho mô hình Markov ẩn.
Một mô hình Markov ẩn có 2 giả thiết chính:

- Xác suất của một trạng thái cụ thể chỉ phụ thuộc vào trạng thái ngay trước đó (Markov Assumptions):

$$P(q_i | q_1 \dots q_{i-1}) = P(q_i | q_{i-1}). \tag{1}$$

- Xác suất của một quan sát o_i chỉ phụ thuộc vào trạng thái tương ứng sinh ra nó q_i chứ không phụ thuộc vào bất kỳ trạng thái q_j nào khác hay bất kỳ quan sát o_k nào.

$$P(o_i | q_1, \dots q_i \dots, q_T, o_1, \dots, o_i, \dots, o_T) = P(o_i | q_i). \tag{2}$$

Ví dụ bài toán mà giả thiết cho mô hình Markov ẩn hợp lý:
Ví dụ bài toán mà giả thiết cho mô hình Markov ẩn bất hợp lý:

1.3 Cho một mô hình Markov ẩn với các tham số đã biết, thuật toán tiến trước (forward algorithm) được dùng để xác định độ hợp lý (likelihood) của một chuỗi quan sát (observation). Mô tả và đánh giá độ phức tạp của thuật toán tiến trước.

Xét mô hình Markov ẩn $\lambda = (A, B)$ và chuỗi các quan sát O, hãy xác định độ hợp lý (likelihood) của nó $P(O|\lambda)$.

Thuật toán tiến trước sử dụng một bảng để lưu trữ các giá trị trung gian và tính toán xác suất xảy ra một quan sát nào đó thông qua việc lấy tổng của tất cả các trường hợp có thể xảy ra của tập các trạng thái "ẩn" có thể đưa đến quan sát tương ứng, thực hiện tính toán trên tập các đường đi tạo thành một "lưới".

Mỗi mắt xích trong "lưới" của thuật toán này, $\alpha_t(j)$ thể hiện xác suất ở trạng thái j sau khi trải qua t quan sát đầu tiên, với mô hình λ đã biết, nói cách khác,

$$\alpha_t(j) = P(o_1, o_2, \ldots, o_t, q_t = j|\lambda)$$

, và giá trị này được tính dựa trên các đường đi dẫn đến mắt xích này và lấy tổng, tức là

$$\alpha_t(j) = \sum_{i=1}^{N} \alpha_{t-1}(i) a_{ij} b_j(o_t)$$

Trong đó: $\alpha_{t-1}(i)$ chính là xác suất trên đoạn đường đến ô này, a_{ij} là xác suất chuyển tiếp từ trạng thái q_i sang trạng thái hiện tại q_j, và $b_j(o_t)$ là độ hợp lý của trạng thái quan sát o_t ứng với trạng thái hiện tại j.

Sau đây là mô tả thuật toán tiến trước:

Algorithm 1 Thuật toán tiến trước

function FORWARD(*observations* of len T, *stategraph* of len N) **return** $forwardprob$
  tạo một ma trân xác suất đường đi $forward[N, T]$
  ▷ khởi tạo
  for mỗi trạng thái s **from** 1 **to** N **do**
    $forward[s, 1] \leftarrow \pi_s * b_s(o_1)$
  ▷ đệ quy
  for mỗi bước t **from** 2 **to** T **do**
    for mỗi trạng thái s **from** 1 **to** N **do**
      $forward[s, t] \leftarrow \sum_{s'=1}^{N} \left(forward\left[s', t-1\right] * a_{s',s} * b_s(o_t) \right)$
  $forwardprob \leftarrow \sum_{s=1}^{N} forward[s, T]$
  ▷ Kết thúc
  return $forwardprob$

Trong giải thuật nói trên, giá trị $forward[s, t]$ chính là giá trị $\alpha_t(s)$ trình bày ở nội dung trước cùng đề mục. Các bước thực hiện được tóm lại như sau:

1. Khởi tạo: gán các

$$\alpha_1(j) = \pi_j b_j(o_1), \forall 1 \leq j \leq N$$

(các dòng 4, 5 trên mã giả)

2. Tính toán truy hồi: tính các

$$\alpha_t(j) = \sum_{i=1}^{N} \alpha_{t-1}(i) a_{ij} b_j(o_t); \forall 1 \leq j \leq N, 1 < t \leq T$$

(các dòng 7, 8, 9 trên mã giả)

3. Kết thúc: giá trị cần tính

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

(chính là giá trị $forwardprob$ đề cập trong thuật toán)

Giải thuật có độ phức tạp thời gian là $O(N^2T)$ trong đó N, T tương ứng là độ dài của tập trạng thái và độ dài của dãy quan sát cần tính toán giá trị độ hợp lý (likelihood) này; và là $O(NT)$ cho không gian lưu trữ (bảng tạm $forward$).

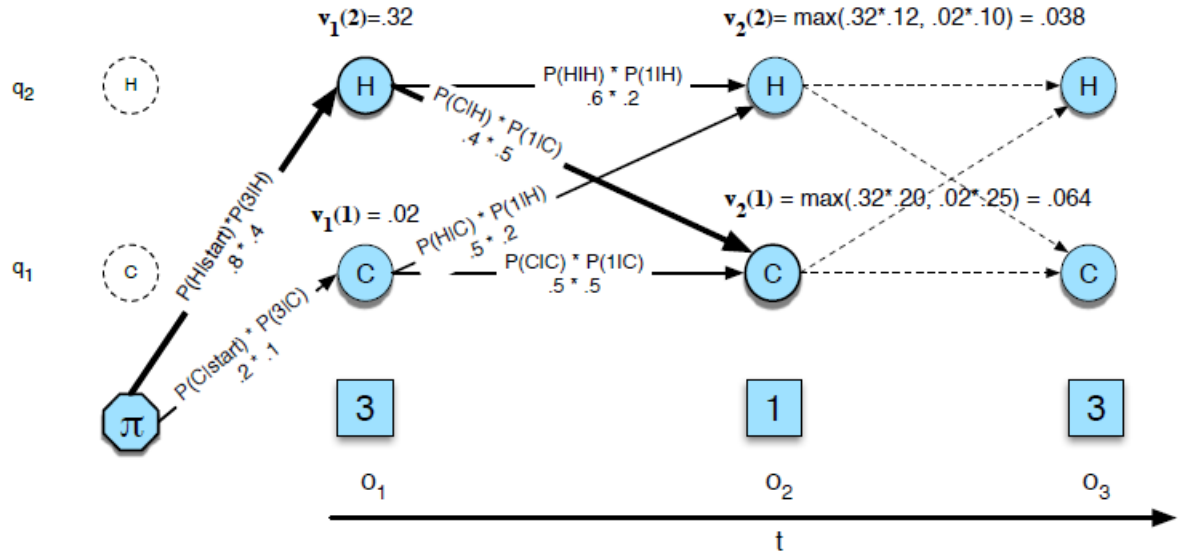
1.4 Cho một mô hình Markov ẩn với các tham số đã biết, thuật toán Viterbi được dùng để xác định chuỗi trạng thái (state) khả dĩ nhất. Mô tả và đánh giá độ phức tạp của thuật toán Viterbi.

Với mỗi mô hình chứa những biến ẩn (hidden variables), chẳng hạn như mô hình Markov ẩn, việc quyết định chuỗi biến nào là nền tảng (underlying source) của một (vài) chuỗi quan sát gọi là **giải mã** (decode). Công việc này được thực hiện bởi một **decoder**.

Giải mã (Decoding): cho một mô hình Markov ẩn $\lambda = (A, B)$ và một chuỗi các quan sát $O = o_1, o_2, ..., o_T$, tìm chuỗi các trạng thái $Q = q_1 q_2 q_3 ... q_T$ có khả năng xảy ra nhất.

Ta có thể đưa ra chuỗi tốt nhất như sau: với mỗi chuỗi trạng thái ẩn có thể, ta có thể sử dụng thuật toán tiến trước và tính độ hợp lý của chuỗi quan sát có được từ chuỗi trạng thái ẩn đó. Sau đó, ta có thể chọn chuỗi với độ hợp lý cao nhất. Dễ thấy từ phần trước rằng ta không thể làm theo cách này vì số lượng chuỗi trạng thái có thể là hàm mũ.

Thay vào đó, thuật toán giải mã phổ biến nhất là **thuật toán Viterbi** (Viterbi algorithm). Giống như thuật toán tiến trước, Viterbi là một thuật toán quy hoạch động sử dụng dynamic programming trellis. Viterbi cũng thể hiện một biến thể của quy hoạch động, thuật toán minimum edit distance.



Hình 1: Viterbi trellis dùng cho việc tìm đường đi tốt nhất qua các không gian trạng thái ẩn cho việc ăn kem. Các trạng thái ẩn được đóng khung trong vòng tròn, các quan sát được đóng khung hình vuông. Các hình tròn nét đứt màu trắng thể hiện. Hình trên thể hiện quá trình tính $v_i(j)$ cho 2 trạng thái ở 2 bước. Quá trình tính toán trong mỗi ô sử dụng công thức (4) $v_t(j) = max_{1 \leqslant N-1} v_{t-1} a_{ij} b_j(o_t)$. Xác suất trong mỗi ô được tính theo công thức (3) $v_t(j) = P(q_0, q_1, ..., q_{t-1}, o_1, o_2, ..., o_t, q_t = j | \lambda)$.

Hình 1 cho ta một ví dụ về Viterbi trellis về tìm chuỗi trạng thái ẩn tốt nhất cho chuỗi quan sát 3 1 3. Ý tưởng là xử lý chuỗi quan sát từ trái sang phải, điền vào những trellis. Mỗi ô trong trellis, $v_t(j)$, thể hiện xác suất mô hình Markov ẩn ở trạng thái j sau khi trải qua t quan sát và đã duyệt qua các trạng thái có thể $q_1, q_2, ..., q_{t-1}$, với tự động hóa λ biết trước. Giá trị mỗi ô $v_t(j)$ được tính đệ quy bằng cách sử dụng đường đi có thể xảy ra nhất dẫn tới ô đó. Mỗi ô thể hiện xác suất được tính bằng công thức sau:

$$v_t(j) = \max_{q_1 \ldots q_{t-1}} P(q_1 \ldots q_{t-1}, o_1, o_2 \ldots o_t, q_t = j | \lambda). \tag{3}$$

Cần lưu ý rằng ta lấy đại diện con đường có thể xảy ra nhất bằng cách lấy giá trị lớn nhất của các chuỗi trạng thái trước đó $\max_{q_1 \ldots q_{t-1}}$. Giống như các thuật toán quy hoạch động khác, thuật toán Viterbi điền giá trị vào mỗi ô bằng phương pháp đệ quy. Biết rằng ta đã tính xác suất của mỗi trang thái ở thời điểm $t-1$, ta tính xác suất Viterbi bằng cách tìm mở rộng (extensions) có thể nhất của con đường dẫn đến ô hiện tại. VỚi mỗi trạng thái q_j cho trước ở thời điểm t, giá trị $v_t(j)$ được tính như sau:

$$v_t(j) = \max_{i=1}^{N} v_{t-1}(i) a_{ij} b_j(o_t). \tag{4}$$

3 nhân tử trong công thức (4) được sử dụng cho việc mở rộng con đường trước đó để tính xác suất Viterbi ở thời điểm t :

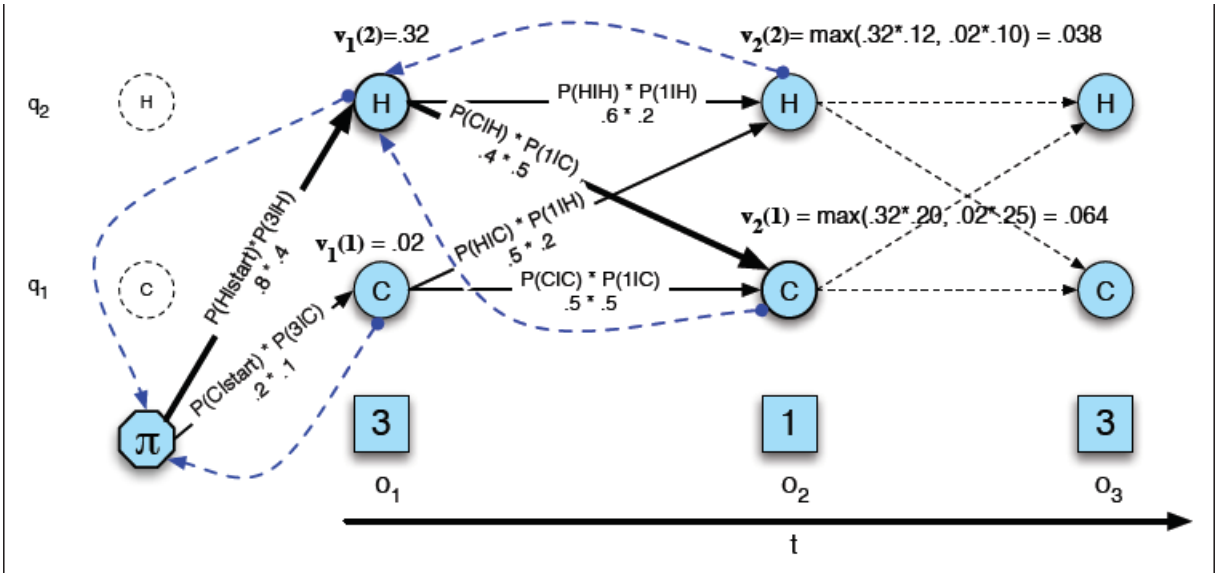
$v_{t-1}(i)$: xác suất con đường Viterbi (previous Viterbi path probability) từ bước trước đó.
a_{ij} : xác suất chuyển (transition probability) từ trạng thái q_i trước đó sang trạng thái q_j hiện tại.
$b_j(o_t)$: độ hợp lý (state observation likelihood) của quan sát o_t từ trạng thái j.

Mã giả của thuật toán Viterbi như sau:

Algorithm 2 Thuật toán Viterbi

function VITERBI(*observation* of len T, *state_graph* of len N)
    tạo một ma trân xác suất đường đi $viterbi[N, T]$
    ▷ khởi tạo
    for mỗi trạng thái s **from** 1 **to** N **do**
        $viterbi[s, 1] \leftarrow \pi_s * b_s(o_1)$
        $backpointer[s, 1] \leftarrow 0$
    ▷ đệ quy
    for mỗi bước t **from** 2 **to** T **do**
        for mỗi trạng thái s **from** 1 **to** N **do**
            $viterbi[s, t] \leftarrow \max_{s'=1}^{N} (viterbi[s', t-1] * a_{s',s} * b_s(o_t))$
            $backpointer[s, t] \leftarrow \arg\max_{s'=1}^{N} (viterbi[s', t-1] * a_{s',s} * b_s(o_t))$
    ▷ Kết thúc
    $best\_path\_prob \leftarrow \max_{s=1}^{N} (viterbi[s, T])$
    $best\_path\_pointer \leftarrow \arg\max_{s=1}^{N} (viterbi[s, T])$
    $best\_path \leftarrow$ con đường bắt đầu ở trạng thái $best\_path\_pointer$,
    đi theo $backpointer[]$ đến các trạng thái
    return $best\_path, best\_path\_prob$

Thuật toán Viterbi tương tự như thuật toán tiến trước, ngoại trừ việc thuật toán này lấy **giá trị lớn nhất** của các xác suất đường đi (path probabilities) trước đó trong khi thuật toán tiến trước lấy **tổng**. Ngoài ra, thuật toán Viterbi có thêm một thành phần so với thuật toán tiến trước: **backpointer**. Lý do cho sự khác biệt này là trong khi thuật toán tiến trước cần cung cấp một độ hợp lý quan sát (observation likelihood), thuật toán Viterbi cần cung cấp xác suất và chuỗi trạng thái hợp lý nhất. Ta tính chuỗi trạng thái tốt nhất này bằng cách lưu lại vết (keep track) của đường đi từ các trạng thái ẩn dẫn tới mỗi trạng thái, giống như trong hình 2, và sau đó, quay lại tìm đường đi tốt nhất tới trạng thái đầu (Viterbi backtrace).



Hình 2: Viterbi backtrace

Ta có định nghĩa của thuật toán Viterbi như sau:

1. Khởi tạo:

$$v_1(j) = \pi_j b_j \quad 1 \leqslant j \leqslant N$$
$$bt_1(j) = 0 \quad 1 \leqslant j \leqslant N$$

2. Đệ quy:

$$v_t(j) = \max_{i=1}^{N} (v_{t-1}(i)\, a_{ij} b_j(o_t)) \quad 1 \leqslant j \leqslant N, 1 < t \leqslant T$$
$$bt_t(j) = \max_{i=1}^{N} (v_{t-1}(i)\, a_{ij} b_j(o_t)) \quad 1 \leqslant j \leqslant N, 1 < t \leqslant T$$

3. Kết thúc:

$$\text{Xác suất tốt nhất: } P* = \max_{i=1}^{N} (v_T(i))$$
$$\text{Điểm bắt đầu của con đường tốt nhất: } q_T* = \arg\max_{i=1}^{N} (v_T(i))$$

Đánh giá độ phức tạp của thuật toán Viterbi:

1. Độ phức tạp thời gian:

   - Ở bước khởi tạo, ta duyệt qua lần lượt các trạng thái đánh số từ 1 đến N nên độ phức tạp thời gian ở bước này là $O(N)$.
   - Ở bước đệ quy, ta dùng 2 vòng lặp lồng nhau. Vòng lặp trong có độ phức tạp $O(N^2)$, còn vòng lặp ngoài có độ phức tạp $O(T)$. Như vậy, độ phức tạp của bước này là $O(T \cdot N^2)$.
   - Ở bước kết thúc, độ phức tạp của bước này là $O(N)$.

   Như vậy, độ phức tạp thời gian của thuật toán Viterbi là $O(T \cdot N^2)$.

2. Độ phức tạp không gian của thuật toán này là $O(T \cdot N)$.

1.5 Cho một chuỗi quan sát, giả sử ta cho rằng chuỗi quan sát này được sinh ra từ một mô hình Markov ẩn với tham số chưa biết, thuật toán Baum – Welch được dùng để ước lượng các tham số này. Thuật toán Baum – Welch là trường hợp đặc biệt của thuật toán Kỳ vọng – Tối ưu (Expectation – Maximization, hay EM). Thuật toán này gồm 2 bước: bước E (Expectation, hay Kỳ vọng) và bước M (Maximization, hay Tối ưu).

1.5.1 Mô tả thuật toán Kỳ vọng – Tối ưu tổng quát.

Thuật toán EM là một thuật toán tuần tự (iterative algorithm), dùng để tìm kỳ vọng ban đầu cho các xác suất, sau đó sử dụng những kỳ vọng này để tìm những kỳ vọng tốt hơn, cứ tiếp tục như vậy, cải thiện giá trị xác suất học được một cách tuần tự.
Mô tả thuật toán Kỳ vọng – Tối ưu:**g4gem**

1. Cho một tập hợp dữ liệu chưa hoàn chỉnh (incomplete data), xét một tập hợp các tham số khởi đầu (starting parameters).
2. Bước Kỳ vọng (bước E): sử dụng các dữ liệu đã được quan sát sẵn từ tập dataset, dự đoán giá trị các dữ liệu bị khuyết.
3. Bước Tối ưu (bước M): Dữ liệu hoàn chỉnh từ bước Kỳ vọng được sử dụng để cập nhật giá trị các tham số.
4. Lặp lại bước 2 và bước 3 cho đến khi hội tụ về giá trị tối ưu.

1.5.2 Mô tả và đánh giá độ phức tạp của bước E và bước M của thuật toán Baum – Welch

Trong thực tế, nhiều trường hợp ta không thể biết số lượng của bất kỳ một trạng thái ẩn nào. Thuật toán Baum – Welch xử lý vấn đề này bằng cách dự đoán các số lượng này một cách tuần tự. Ta bắt đầu bằng việc dự đoán xác suất chuyển và quan sát, sau đó sử dụng các kết quả này để tìm những xác suất tốt hơn. Ta tiếp tục bằng việc tính xác suất tiến trước (forward probability) cho một quan sát, sau đó chia khối lượng xác suất (probability mass) cho tất cả con đường dẫn tới xác suất tiến trước này.
Trước hết, ta cần định nghĩa khái niệm **xác suất quay lại** (backward probability). Khái niệm này chỉ xác suất nhìn thấy quan sát này từ thời điểm $t+1$ đến cuối, biết rằng ta đang ở trạng thái i tại thời điểm t (và một tự động hóa λ có sẵn):

$$\beta_t(i) = P(o_{t+1}, o_{t+2}, \ldots, o_T | q_t = i, \lambda) \tag{5}$$

Xác suất này được tính bằng quy nạp tương tự như thuật toán tiến trước.

1. Khởi tạo:

$$\beta_T(i) = 1, \quad 1 \leqslant i \leqslant N.$$

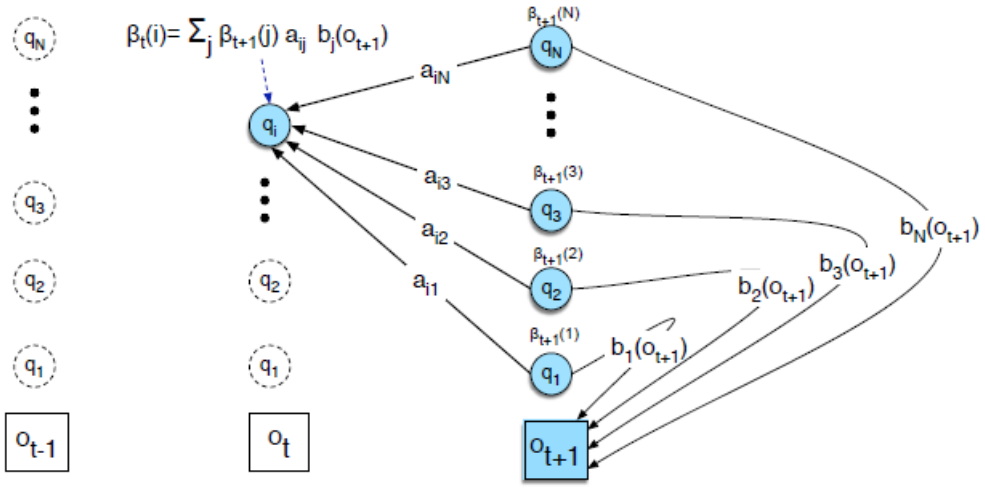
2. Đệ quy:

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(o_{t+1}) \beta_{t+1}(j), \quad 1 \leqslant i \leqslant N, 1 \leqslant t < T .$$

3. Kết thúc:

$$P(O|\lambda) = \sum_{j=1}^{N} \pi_j b_j(o_1) \beta_1(j).$$

Hình 3 sau đây minh họa các bước quy nạp tính xác suất quay lui.



Hình 3: Tính xác suất quay lui

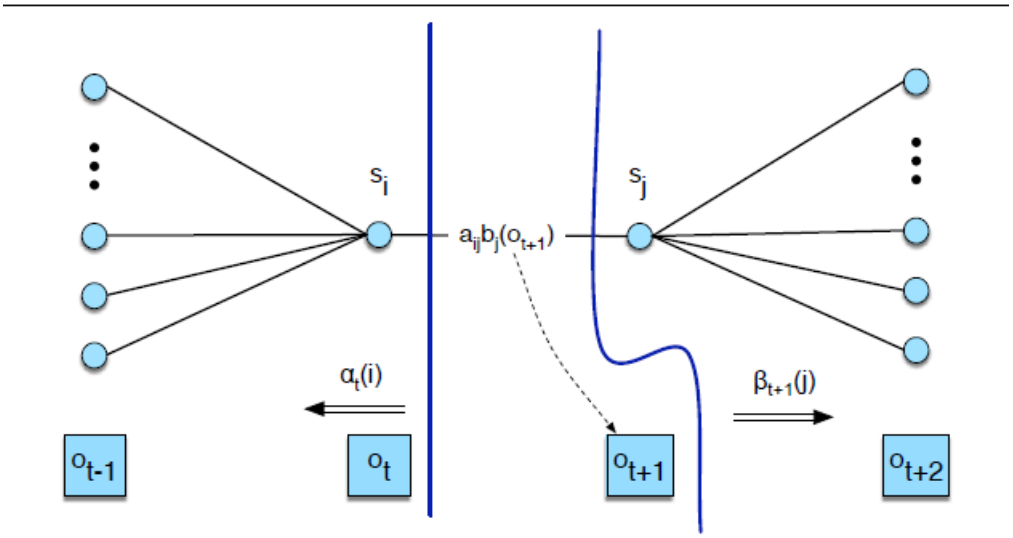
Ta sẽ xem xét cách mà xác suất tiến trước và xác suất quay lui giúp ta tính xác suất chuyển a_{ij} và xác suất quan sát $b_i(o_t)$ từ một chuỗi quan sát, ngay cả khi con đường thật sự trong mô hình bị ẩn.

$$\widehat{a_{ij}} = \frac{\text{giá trị kỳ vọng của số lượng chuyển đổi từ trạng thái } i \text{ sang trạng thái } j}{\text{giá trị kỳ vọng của số lượng chuyển đổi từ trạng thái } i}. \quad (6)$$

Ta định nghĩa ξ_t là xác suất xảy ra trạng thái i tại thời điểm t và trạng thái j tại thời điểm $t+1$, biết trước mô hình và chuỗi quan sát.

$$\xi_t(i,j) = P(q_t = i, q_{t+1} = j | O, \lambda). \quad (7)$$

Quá trình tính giá trị này được minh họa trong hình 4.



Hình 4: Tính xác suất $\xi_t(i,j)$. Rabiner ©1989 IEEE

Sau một vài bước biến đổi ([**hmm2021**]), ta được công thức tính ξ_t như sau:

$$\xi_t(i,j) = \frac{\alpha_t(i)\, a_{ij} b_j(o_t+1)\, \beta_{t+1}(j)}{\sum_{j=1}^{N} \alpha_t(j)\, \beta_t(j)}. \tag{8}$$

Số lượng chuyển đổi kỳ vọng từ trạng thái i sang trạng thái j là tổng theo t của các giá trị ξ. Với công thức (6), ta chỉ cần thêm tổng các chuyển đổi từ trạng thái i. Ta có thể thu được giá trị này bằng cách tìm tất cả giá trị chuyển ra từ i. Công thức cho $\widehat{a_{ij}}$ như sau:

$$\widehat{a_{ij}} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \sum_{k=1}^{N} \xi_t(i,k)}. \tag{9}$$

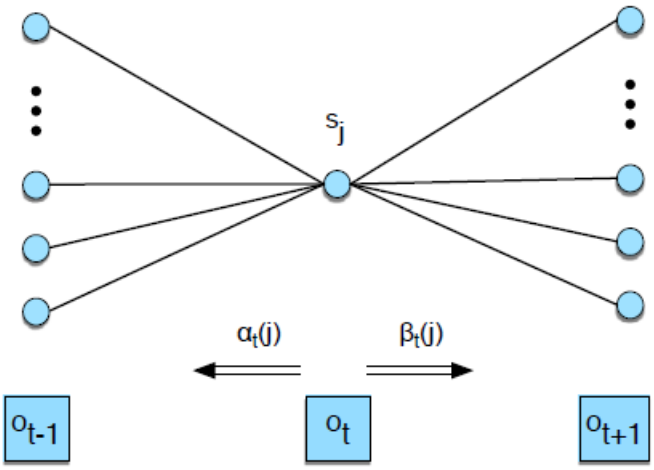
Ngoài ra, ta còn cần tìm xác suất quan sát. Đây là xác suất của một biểu tượng (symbol) v_k từ bộ từ vựng quan sát (observation vocabulary) V, với trạng thái j biết trước: $\widehat{b_j}(v_k)$. Ta sẽ tính:

$$\widehat{b_j}(v_k) = \frac{\text{số lượng kỳ vọng thời điểm có trạng thái } j \text{ và biểu tượng quan sát} v_k}{\text{số lượng các thời điểm có trạng thái } j}. \tag{10}$$

Ta định nghĩa $\gamma_t(j)$ là xác suất ở trạng thái j tại thời điểm t :

$$\gamma_t(j) = P(q_t = j | O, \lambda). \tag{11}$$

Quá trình tính giá trị này được minh họa trong hình 5.



Hình 5: Tính xác suất $\gamma_t(j)$. Rabiner ©1989 IEEE

Sau một vài bước biến đổi ([**hmm2021**]), ta được công thức tính $\gamma_t(j)$ như sau:

$$\gamma_t(j) = \frac{\alpha_t(j)\,\beta_t(j)}{P(O|\lambda)}. \tag{12}$$

Từ đó, ta có công thức tính $\widehat{b_j}(v_k)$ như sau:

$$\widehat{b_j}(v_k) = \frac{\sum_{t=1}^{T} \quad s.t. O_t = v_k \quad \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)}. \tag{13}$$

Giờ chúng ta có công thức (9) và công thức (13) để *dự đoán lại* (re-estimate) ma trận xác suất chuyển A và ma trận xác suất phụ thuộc trạng thái B từ một chuỗi quan sát O, giả sử ta đã có dự đoán trước đó của A và B.
Các thao tác dự đoán này là lõi (core) của thuật toán Baum – Welch. Thuật toán này bắt đầu từ một vài dự đoán ban đầu của các tham số mô hình Markov ẩn $\lambda = (A, B)$. Sau đó, ta sẽ chạy tuần tự 2 bước. Giống như các trường hợp khác của thuật toán EM, thuật toán Baum – Welch có 2 bước: bước E và bước M.
Ở bước E, ta tính giá trị chiếm giữ trạng thái kỳ vọng (expected state occupancy count) γ và giá trị chuyển trạng thái kỳ vọng (expected state transition count) ξ từ các xác suất A và B trước đó.
Ở bước M, ta sử dụng γ và ξ để tính giá trị A và B mới.

Algorithm 3 Thuật toán Baum – Welch

function BAUM-WELCH(*observation* of len T, *output vocabulary* V, *hidden state set* Q)
    khởi tạo A và B
    duyệt đến khi hội tụ
                ▷ Bước E
    $\gamma_t(j) = \frac{\alpha_t(j)\beta_t(j)}{\alpha_T(q_F)}, \quad \forall t, j$
    $\xi_t(i,j) = \frac{\alpha_t(i)a_{ij}b_j(o_{t+1})\beta_{t+1}(j)}{\alpha_T(q_F)}, \quad \forall t, i, j$
                ▷ Bước M

8: $\widehat{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T} \sum_{k=1}^{N} \xi_t(i,k)}$

9: $\widehat{b_j}(v_k) = \frac{\sum_{t=1}^{T} \quad s.t.O_t = v_k \quad ^{\gamma_t(j)}}{\sum_{t=1}^{T} \gamma_t(j)}$

10: **return** A, B

Đánh giá độ phức tạp của thuật toán Baum – Welch:

1. Độ phức tạp thời gian:
   - Bước E: bước E gồm 2 phần, phần đầu có độ phức tạp $O(NT)$, phần sau có độ phức tạp $O\left(N^2T\right)$ nên độ phức tạp của bước E là $O\left(N^2T\right)$.
   - Bước M: bước M gồm 2 phần, phần đầu có độ phức tạp $O\left(N^2T\right)$, phần sau có độ phức tạp $O(NVT)$ nên độ phức tạp của bước M là $O\left(N^2T\right)$.
2. Độ phức tạp không gian:
   - Bước E: $O\left(N^2T\right)$.
   - Bước M: $O\left(N^2\right)$.