# 확률통계 및 공통 알고리즘

## 6. 공통 알고리즘 -II

임채영

서울대학교

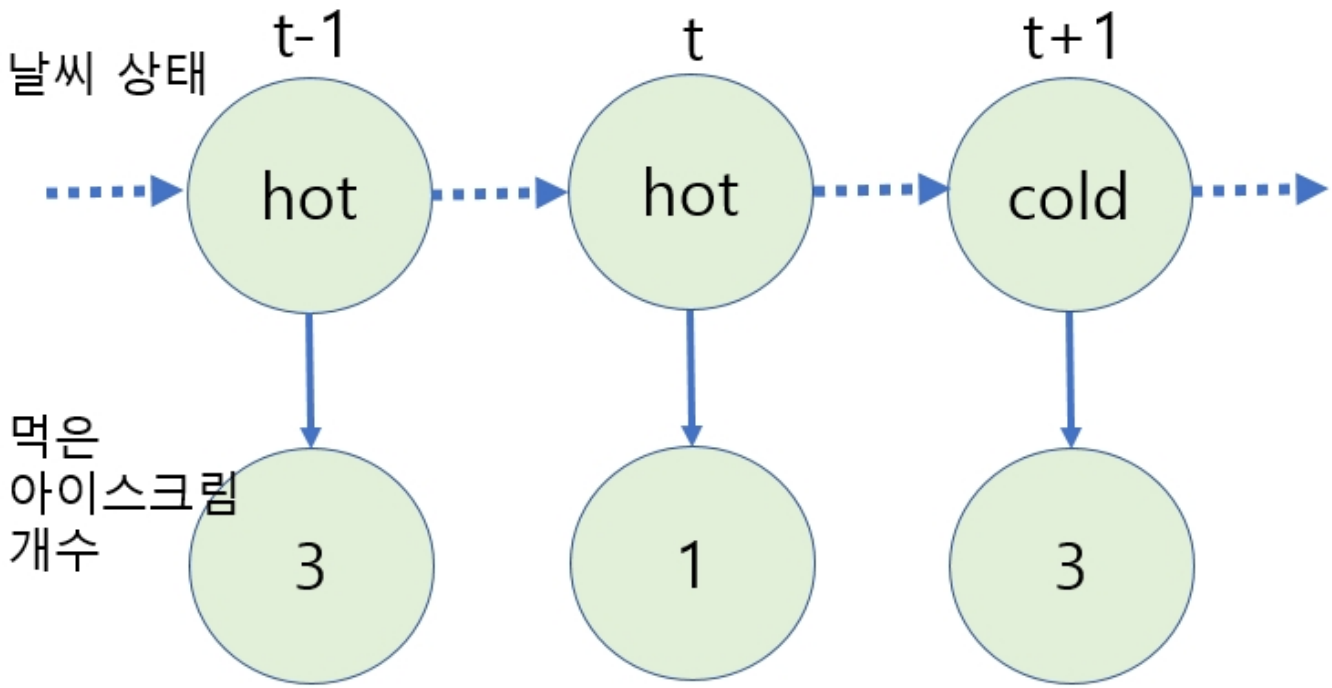
# 이번 강의에서 다룰 내용

- 은닉 마코프 모형(HMM)

# 은닉 마코프 모형 (Hidden Markov Model, HMM)

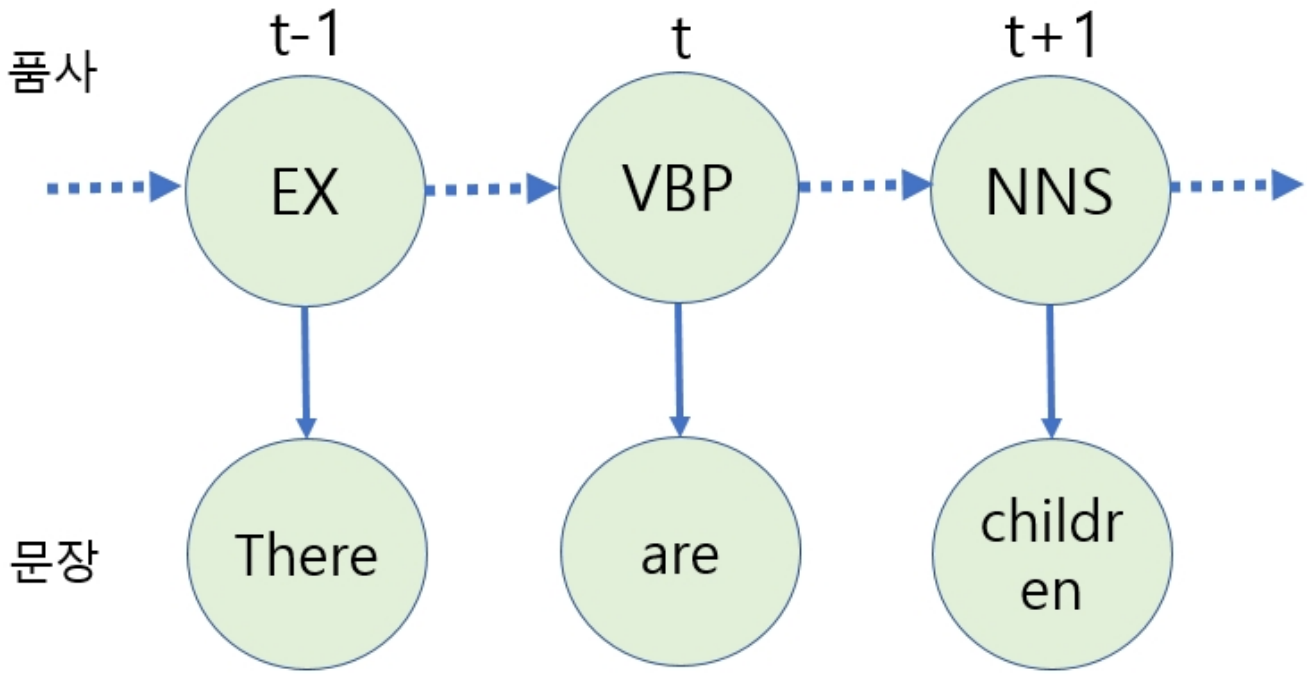
- 결과값들이 시간(또는 순서)에 따라 관측이 되었을때, 이러한 결과값을 주는 관측되지 않은 잠재변수열(또는 은닉변수열, hidden variable sequence)이 있다고 가정하는 모형이다.
- 결과값은 은닉변수들의 상태 (state)에 의해서만 결정된다고 가정한다.
- 은닉변수열은 마코프 성질을 가지는것으로 가정한다.
- 음성인식, 동작인식, 품사태킹, 생물정보학, 기상학등 다양한 분야에서 사용되고 있다.
- HMM과 알고리즘에 대한 소개는 Jurafsky 와 Martin의 "Speech and Language Processing" 책 참고.

# HMM 예제 1



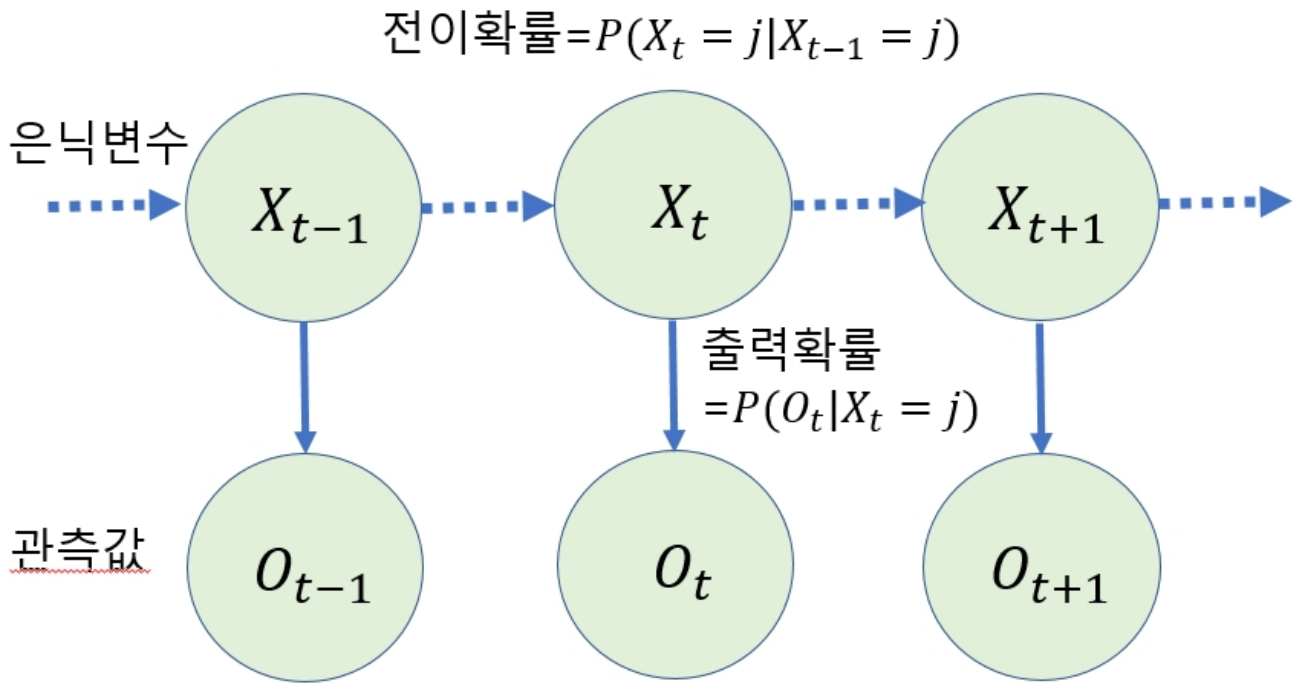

- 데이터 (결과값,관측값): 먹은 아이스크림 개수
- 숨겨진 정보(은닉변수): 날씨 (hot/cold)
- 먹은 아이스크림 개수로 날씨가 어떤지 추측
- 주어진 날씨에 따라 얼마의 아이스크림을 먹을지 확률계산
- 전날의 날씨에서 다음날의 날씨에 대한 확률계산

# HMM 예제 2




- 데이터 (결과값,관측값): 문장
- 숨겨진 정보(은닉변수): 품사 (EX: existential, VBP: verb present, NNS: noun, plural )
- 주어진 문장으로 품사가 어떤지 추측

# HMM 형태




- 주어진 정보
  - $\boldsymbol{O}_1^T = (O_1, \cdots, O_T)$ (결과값, 데이터, observations)
  - $\{S_1, \cdots, S_N\}$: 은닉변수의 상태공간 (State space for hidden variables). 편의상 $\{1, \cdots, N\}$으로 표현.

- 모르는 정보

  - 은닉변수(hidden variables):
    $\boldsymbol{X}_1^T = (X_1, \cdots, X_T), X_t \in \{1, \cdots, N\}$.
  - 전이확률 (transition probability):
    $a_{ij} = P(X_t = j | X_{t-1} = i), i, j = 1, \cdots, N$.
    $\boldsymbol{A}$: $a_{ij}$로 구성된 $N \times N$ 전이확률 행렬
  - 출력확률 (emission probability):
    $b_j(O_t) = P(O_t | X_t = j), j = 1, \cdots, N, t = 1, \cdots, T$.
    $O_t$는 연속 확률변수 또는 이산 확률변수이다.
    본 강의에서는 이산확률변수이면서 가능한 값들이 유한한 경우로 가정한다. 즉, $O_t \in \{o_1, \cdots, o_L\}$로 가정한다.
    $\boldsymbol{B}$: $b_i(o_j)$로 구성된 $N \times L$ 출력확률 행렬

- $\boldsymbol{\pi} = (\pi_1, \cdots, \pi_N)$: 은닉변수의 초기 상태에 대한 확률분포 (주어진 정보라고 가정 할 수도 있다.)

- $\boldsymbol{\lambda} = (\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{\pi})$를 데이터로부터 추정해야 하는 모수로 본다. (전이확률이나 출력확률을 몇개의 모수로 표현되는 확률분포로 가정하는경우, $\boldsymbol{A}, \boldsymbol{B}$는 그러한 모수들의 집합으로 볼수 있다.)

# HMM의 가정

- 마코프 가정 (Markov property):
  $P(X_t|X_{t-1}, \cdots, X_1) = P(X_t|X_{t-1})$ (first-order)
- 출력확률의 독립가정:
  $P(O_t|X_1, \cdots, X_T, O_1, \cdots, O_T) = P(O_t|X_t)$

# HMM의 학습과 추론

- 주어진 데이터 $\boldsymbol{O}_1^T$를 이용하여 HMM의 모수를 추정하고, 가장 최적의 은닉변수열 $\boldsymbol{X}_1^T$를 찾는다.

(1) 학습 (Learning): $\boldsymbol{\lambda}$ 추정

(2) 디코딩 (decoding): 최적의 $\boldsymbol{X}_1^T$ 찾기

# (1) 학습

- 학습 단계에서는 주어진 $\boldsymbol{O}_1^T$에 대하여 모수 $\boldsymbol{\lambda} = (\boldsymbol{A}, \boldsymbol{B}, \pi)$를 찾는다(학습시킨다).
- $\boldsymbol{A}$는 전이확률로 구성된 행렬이고, $\boldsymbol{B}$는 출력확률로 구성된 행렬, $\pi$는 초기상태에 대한 확률로 이루어진 벡터이다.
- 최대가능도 방법으로 학습시킬수 있다. 즉, $\max \mathit{Likelihood}(\boldsymbol{\lambda}) = \max P(\boldsymbol{O}_1^T|\boldsymbol{\lambda})$인 $\hat{\boldsymbol{\lambda}}$를 찾는것을 말한다.
- 본 강의에서는 은닉변수열을 이용한 EM알고리즘 형태인 바움-웰치(Baum-Welch) 알고리즘을 소개한다.

# 바움-웰치 알고리즘

- 잠재변수(은닉변수)가 있는 모형에서 Likelihood (가능도)를 최대화 시키는 방법인 EM 알고리즘에서 E-step을 회상해보자.
- $Q(\theta|\theta_{(r)}) = E(\log(f(\boldsymbol{Y}_o, \boldsymbol{Y}_m|\theta)))$
- HMM모형에서는

  $\theta = \lambda = (\boldsymbol{A}, \boldsymbol{B}, \pi)$,

  $\boldsymbol{Y}_o = \boldsymbol{O}_1^T$,

  $\boldsymbol{Y}_m = \boldsymbol{X}_1^T$,

  $f(\boldsymbol{Y}_o, \boldsymbol{Y}_m|\theta) = P(\boldsymbol{O}_1^T, \boldsymbol{X}_1^T|\lambda)$가 된다.

# E-step

- $Q(\lambda|\lambda_{(r)}) = E(\log P(\boldsymbol{O}_1^T, \boldsymbol{X}_1^T|\lambda)) = \sum_{\boldsymbol{X}_1^T} \log P(\boldsymbol{O}_1^T, \boldsymbol{X}_1^T|\lambda) P(\boldsymbol{X}_1^T|\boldsymbol{O}_1^T, \lambda_{(r)})$
- 한편 $Q(\lambda|\lambda_{(r)})$를 최대화 시키는 $\lambda$를 찾는 의미에서 $\sum_{\boldsymbol{X}_1^T} \log P(\boldsymbol{O}_1^T, \boldsymbol{X}_1^T|\lambda) P(\boldsymbol{X}_1^T, \boldsymbol{O}_1^T|\lambda_{(r)})$를 최대화 시키는 $\lambda$를 찾는것과 동일하다.
- 따라서, $Q(\lambda|\lambda_{(r)}) = \sum_{\boldsymbol{X}_1^T} \log P(\boldsymbol{O}_1^T, \boldsymbol{X}_1^T|\lambda) P(\boldsymbol{X}_1^T, \boldsymbol{O}_1^T|\lambda_{(r)})$로 생각하자.

$$
\begin{aligned}
P(\boldsymbol{O}_1^T, \boldsymbol{X}_1^T|\lambda) &= P(O_T|\boldsymbol{X}_1^T, \boldsymbol{O}_1^{T-1}, \lambda)P(\boldsymbol{X}_1^T, \boldsymbol{O}_1^{T-1}|\lambda) \\
&= P(O_T|X_T, \lambda)P(X_T|\boldsymbol{X}_1^{T-1}, \boldsymbol{O}_1^{T-1}, \lambda)P(\boldsymbol{X}_1^{T-1}, \boldsymbol{O}_1^{T-1}|\lambda) \\
&= b_{X_T}(O_T)P(X_T|X_{T-1}, \lambda)P(\boldsymbol{X}_1^{T-1}, \boldsymbol{O}_1^{T-1}|\lambda) \\
&= b_{X_T}(O_T)a_{X_{T-1}\ X_T}P(\boldsymbol{X}_1^{T-1}, \boldsymbol{O}_1^{T-1}|\lambda)
\end{aligned}
$$

반복하면

$$
P(\boldsymbol{O}_1^T, \boldsymbol{X}_1^T|\lambda) = \pi_{X_0}\prod_{t=1}^{T} b_{X_t}(O_t)a_{X_{t-1}\ X_t},
$$

$a_{X_0\, X_1} = \pi_{X_1}$.

$$
\begin{aligned}
Q(\lambda|\lambda_{(r)}) &= \sum_{\boldsymbol{X}_1^T} \log P(\boldsymbol{O}_1^T, \boldsymbol{X}_1^T|\lambda) P(\boldsymbol{X}_1^T, \boldsymbol{O}_1^T|\lambda_{(r)}) \\
&= \sum_{\boldsymbol{X}_1^T} \left(\log \pi_{X_1}\right) P(\boldsymbol{X}_1^T, \boldsymbol{O}_1^T|\lambda_{(r)}) \\
&\quad + \sum_{\boldsymbol{X}_1^T} \left( \sum_{t=2}^{T} \log a_{X_{t-1}\, X_t} \right) P(\boldsymbol{X}_1^T, \boldsymbol{O}_1^T|\lambda_{(r)}) \\
&\quad + \sum_{\boldsymbol{X}_1^T} \left( \sum_{t=1}^{T} \log b_{X_t}(O_t) \right) P(\boldsymbol{X}_1^T, \boldsymbol{O}_1^T|\lambda_{(r)})
\end{aligned}
$$

# M-step

- $Q(\boldsymbol{\lambda}|\boldsymbol{\lambda}_{(r)})$는 세 부분으로 나뉘어 진다.
- 첫번째, 두번째, 세번째 항들은 $\{\pi_i\}$, $\{a_{ij}\}$, $\{b_j(\cdot)\}$를 각각 포함하고 있어 따로 최대화 시킬수 있다.
- $\boldsymbol{\pi}$는 주어진 경우는 두번째 세번째 항만 이용하면 된다.

- $Q(\lambda|\lambda_{(r)})$의 첫번째 항:

$$\begin{aligned} Q_1 &:= \sum_{\boldsymbol{x}_1^T} \left(\log \pi_{X_1}\right) P(\boldsymbol{X}_1^T, \boldsymbol{O}_1^T|\lambda_{(r)}) \\ &= \sum_{X_1} \left(\log \pi_{X_1}\right) P(X_1, \boldsymbol{O}_1^T|\lambda_{(r)}) \end{aligned}$$

- $\sum_i \pi_i = 1$이어야 하므로, Lagrange multiplier $\gamma$를 도입한

$$Q_{1\,\gamma} = \sum_{i=1}^{N} (\log \pi_i)\, P(X_1 = i, \boldsymbol{O}_1^T|\lambda_{(r)}) + \gamma \left(\sum_i \pi_i - 1\right)$$

를 최대화 하는 $\pi_i$들을 찾는다.

- $\partial Q_{1\,\gamma}/\partial \pi_i = \frac{1}{\pi_i} P(X_1 = i, \boldsymbol{O}_1^T | \lambda_{(r)}) + \gamma = 0$
- $\partial Q_{1\,\gamma}/\partial \gamma = \sum_i \pi_i - 1 = 0$
- 위 두 식을 정리하면

$$
\begin{aligned}
\hat{\pi}_i^{(r+1)} &= P(X_1 = i, \boldsymbol{O}_1^T | \lambda_{(r)}) / \sum_j P(X_1 = j, \boldsymbol{O}_1^T | \lambda_{(r)}) \\
&= P(X_1 = i, \boldsymbol{O}_1^T | \lambda_{(r)}) / P(\boldsymbol{O}_1^T | \lambda_{(r)}) \\
&= P(X_1 = i | \boldsymbol{O}_1^T, \lambda_{(r)}) P(\boldsymbol{O}_1^T | \lambda_{(r)}) / P(\boldsymbol{O}_1^T | \lambda_{(r)}) \\
&= P(X_1 = i | \boldsymbol{O}_1^T, \lambda_{(r)}) \qquad (1)
\end{aligned}
$$

- $Q(\lambda|\lambda_{(r)})$의 두번째 항:

$$
\begin{aligned}
Q_2 &:= \sum_{\boldsymbol{x}_1^T} \left( \sum_{t=2}^{T} \log a_{x_{t-1}\ x_t} \right) P(\boldsymbol{x}_1^T, \boldsymbol{o}_1^T|\lambda_{(r)}) \\
&= \sum_{\boldsymbol{x}_1^T} \log a_{x_1\ x_2} P(\boldsymbol{x}_1^T, \boldsymbol{o}_1^T|\lambda_{(r)}) + \cdots \\
&\qquad + \sum_{\boldsymbol{x}_1^T} \log a_{x_{T-1}\ x_T} P(\boldsymbol{x}_1^T, \boldsymbol{o}_1^T|\lambda_{(r)}) \\
&= \sum_{i,j=1}^{N} \log a_{ij} \sum_{\boldsymbol{x}_3^T} P(X_1 = i, X_2 = j, \boldsymbol{x}_3^T, \boldsymbol{o}_1^T|\lambda_{(r)}) + \cdots \\
&\qquad + \sum_{i,j=1}^{N} \log a_{ij} \sum_{\boldsymbol{x}_1^{T-2}} P(X_{T-1} = i, X_T = j, \boldsymbol{x}_1^{T-2}, \boldsymbol{o}_1^T|\lambda_{(r)})
\end{aligned}
$$

$$
\begin{aligned}
Q_2 &= \sum_{i,j=1}^{N} \log a_{ij} P(X_1 = i, X_2 = j, \boldsymbol{O}_1^T | \lambda_{(r)}) + \cdots \\
&\quad + \sum_{i,j=1}^{N} \log a_{ij} P(X_{T-1} = i, X_T = j, \boldsymbol{O}_1^T | \lambda_{(r)}) \\
&= \sum_{i,j=1}^{N} \sum_{t=2}^{T} \log a_{ij} P(X_{t-1} = i, X_t = j, \boldsymbol{O}_1^T | \lambda_{(r)})
\end{aligned}
$$

- $a_{ij} = P(X_t = j | X_{t-1} = i, \lambda)$이므로 $\sum_j a_{ij} = 1, i = 1, \cdots, N$ 을 만족한다.
- 따라서, 첫번째 항의 경우와 마찬가지로 Lagrange multiplier를 도입하여 최대화시킨다.

즉,

$$
\begin{aligned}
Q_{2\,\gamma} &= \sum_{i,j=1}^{N}\sum_{t=2}^{T} \log a_{ij} P(X_{t-1}=i, X_t=j, \boldsymbol{O}_1^T|\boldsymbol{\lambda}_{(r)}) \\
&\quad + \sum_i \gamma_i \Big(\sum_j a_{ij} - 1\Big)
\end{aligned}
$$

를 최대화 하는 $a_{ij}$를 찾는다.

- $\partial Q_{2\,\gamma}/\partial a_{ij} = \frac{1}{a_{ij}} \sum_{t=2}^{T} P(X_{t-1}=i, X_t=j, \boldsymbol{O}_1^T|\boldsymbol{\lambda}_{(r)}) + \gamma_i = 0$
- $\partial Q_{2\,\gamma}/\partial \gamma_i = \sum_j a_{ij} - 1 = 0,\ i = 1, \cdots, N.$
- 위 두 식을 정리하면

$$
\begin{aligned}
\hat{a}_{ij}^{(r+1)} &= \frac{\sum_{t=2}^{T} P(X_{t-1}=i, X_t=j, \boldsymbol{O}_1^T|\boldsymbol{\lambda}_{(r)})}{\sum_k \sum_{t=2}^{T} P(X_{t-1}=i, X_t=k, \boldsymbol{O}_1^T|\boldsymbol{\lambda}_{(r)})} \\
&= \frac{\sum_{t=2}^{T} P(X_{t-1}=i, X_t=j, \boldsymbol{O}_1^T|\boldsymbol{\lambda}_{(r)})}{\sum_{t=2}^{T} P(X_{t-1}=i, \boldsymbol{O}_1^T|\boldsymbol{\lambda}_{(r)})} \qquad (2)
\end{aligned}
$$

- $Q(\lambda|\lambda_{(r)})$의 세번째 항:

$$
\begin{aligned}
Q_3 &:= \sum_{\boldsymbol{x}_1^T} \left( \sum_{t=1}^{T} \log b_{X_t}(O_t) \right) P(\boldsymbol{X}_1^T, \boldsymbol{O}_1^T | \lambda_{(r)}) \\
&= \sum_{\boldsymbol{x}_1^T} \log b_{X_1}(O_1) P(\boldsymbol{X}_1^T, \boldsymbol{O}_1^T | \lambda_{(r)}) + \cdots \\
&\quad + \sum_{\boldsymbol{x}_1^T} \log b_{X_T}(O_T) P(\boldsymbol{X}_1^T, \boldsymbol{O}_1^T | \lambda_{(r)}) \\
&= \sum_{j=1}^{N} \sum_{\boldsymbol{x}_2^T} \log b_i(O_1) P(X_1 = j, \boldsymbol{X}_2^T, \boldsymbol{O}_1^T | \lambda_{(r)}) + \cdots \\
&\quad + \sum_{j=1}^{N} \sum_{\boldsymbol{x}_1^{T-1}} \log b_j(O_T) P(\boldsymbol{X}_1^{T-1}, X_T = j, \boldsymbol{O}_1^T | \lambda_{(r)})
\end{aligned}
$$

$$
\begin{aligned}
Q_3 &= \sum_{j=1}^{N} \log b_i(O_1) P(X_1 = j, \boldsymbol{O}_1^T | \boldsymbol{\lambda}_{(r)}) + \cdots \\
&\quad + \sum_{j=1}^{N} \log b_j(O_T) P(X_T = j, \boldsymbol{O}_1^T | \boldsymbol{\lambda}_{(r)}) \\
&= \sum_{j=1}^{N} \sum_{t=1}^{T} \log b_j(O_t) P(X_t = j, \boldsymbol{O}_1^T | \boldsymbol{\lambda}_{(r)})
\end{aligned}
$$

- $O_t \in \{o_1, \cdots, o_L\}$로 가정했으므로,
  $\sum_{l=1}^{L} b_j(o_l) = \sum_{l=1}^{L} P(O_t = o_l | X_t = j) = 1$이 된다.
- $b_j(o_l)$에 대한 조건을 만족하는 $b_j(o_l)$을 추정해야 하므로,
  Lagrange multiplier를 도입하여 최대화 문제를 푼다.

즉,

$$
\begin{aligned}
Q_{3\,\gamma} &= \sum_{j=1}^{N}\sum_{t=1}^{T} \log b_j(O_t) P(X_t = j, \boldsymbol{O}_1^T | \boldsymbol{\lambda}_{(r)}) \\
&\quad + \sum_j \gamma_j \left( \sum_{l=1}^{L} b_j(o_l) - 1 \right)
\end{aligned}
$$

를 최대화 하는 $b_j(o_l)$을 찾는다.

- $\partial Q_{3\,\gamma} / \partial b_j(o_l) = \sum_{t=1}^{T} \frac{1}{b_j(O_t)} P(X_t = j, \boldsymbol{O}_1^T | \boldsymbol{\lambda}_{(r)}) \delta_{o_l}(O_t) + \gamma_j = 0$. $\delta_{o_l}(O_t)$는 $O_t = o_l$이면 1의 값을 아니면 0의 값을 갖는 dirac delta함수이다.
- $\partial Q_{3\,\gamma} / \partial \gamma_j = \sum_{l=1}^{L} b_j(o_l) - 1 = 0$

- 앞의 두 식을 정리하면

$$\begin{aligned}
\hat{b}_j^{(r+1)}(o_l) &= \frac{\sum_{t=1}^{T} P(X_t = j, \boldsymbol{O}_1^T | \boldsymbol{\lambda}_{(r)}) \delta_{o_l}(O_t)}{\sum_{l=1}^{L} \sum_{t=1}^{T} P(X_t = j, \boldsymbol{O}_1^T | \boldsymbol{\lambda}_{(r)}) \delta_{o_l}(O_t)} \\
&= \frac{\sum_{t=1}^{T} P(X_t = j, \boldsymbol{O}_1^T | \boldsymbol{\lambda}_{(r)}) \delta_{o_l}(O_t)}{\sum_{t=1}^{T} P(X_t = j, \boldsymbol{O}_1^T | \boldsymbol{\lambda}_{(r)})} \qquad (3)
\end{aligned}$$

- 지금까지 구한 $\hat{\pi}_i^{(r+1)}$, $\hat{a}_{ij}^{(r+1)}$, $\hat{b}_j^{(r+1)}(o_l)$ 식 (1), (2), (3)을 정리하면 다음과 같다.

$$\begin{aligned}
\hat{\pi}_i^{(r+1)} &= P(X_1 = i | \boldsymbol{O}_1^T, \boldsymbol{\lambda}_{(r)}) \\
\hat{a}_{ij}^{(r+1)} &= \frac{\sum_{t=2}^{T} P(X_{t-1} = i, X_t = j, \boldsymbol{O}_1^T | \boldsymbol{\lambda}_{(r)})}{\sum_{t=2}^{T} P(X_{t-1} = i, \boldsymbol{O}_1^T | \boldsymbol{\lambda}_{(r)})} \\
\hat{b}_j^{(r+1)}(o_l) &= \frac{\sum_{t=1}^{T} P(X_t = j, \boldsymbol{O}_1^T | \boldsymbol{\lambda}_{(r)}) \delta_{o_l}(O_t)}{\sum_{t=1}^{T} P(X_t = j, \boldsymbol{O}_1^T | \boldsymbol{\lambda}_{(r)})}
\end{aligned}$$

- $\boldsymbol{\lambda}_{(r)}$은 EM알고리즘의 전단계($(r)$)에서 구한 $\boldsymbol{\lambda}$값을 의미함을 상기하자.

EM알고리즘은 다음과 같이 정리할 수 있다.

1. 초기화 (Initialization)
   $\boldsymbol{\lambda}_{(1)} = (\hat{a}_{ij}^{(1)}, \hat{b}_j^{(1)}(o_l), \hat{\pi}_i^{(1)})$를 적당히 정한다.
2. 반복 (Iteration), $r = 2, \cdots$

$$
\begin{aligned}
\hat{\pi}_i^{(r+1)} &= P(X_1 = i | \boldsymbol{O}_1^T, \boldsymbol{\lambda}_{(r)}) \\
\hat{a}_{ij}^{(r+1)} &= \frac{\sum_{t=2}^T P(X_{t-1} = i, X_t = j, \boldsymbol{O}_1^T | \boldsymbol{\lambda}_{(r)})}{\sum_{t=2}^T P(X_{t-1} = i, \boldsymbol{O}_1^T | \boldsymbol{\lambda}_{(r)})} \\
\hat{b}_j^{(r+1)}(o_l) &= \frac{\sum_{t=1}^T P(X_t = j, \boldsymbol{O}_1^T | \boldsymbol{\lambda}_{(r)}) \delta_{o_l}(O_t)}{\sum_{t=1}^T P(X_t = j, \boldsymbol{O}_1^T | \boldsymbol{\lambda}_{(r)})}
\end{aligned}
$$

3. $\|\boldsymbol{\lambda}_{(r+1)} - \boldsymbol{\lambda}_{(r)}\| \leq \epsilon$ 이면 멈춤

- 바움-웰치 알고리즘은 앞 슬라이드의 식들을 좀 더 세분화한 후 각 수식을 반복알고리즘을 통해 구한다.
- 다음은 해당 수식들을 세분화 해서 구하는 과정을 소개한다.

- 필요한 수식들은 다음과 같다.
  - $P(X_1 = i|\boldsymbol{O}_1^T, \boldsymbol{\lambda}_{(r)})$
  - $P(X_{t-1} = i, X_t = j, \boldsymbol{O}_1^T|\boldsymbol{\lambda}_{(r)})$
  - $P(X_t = j, \boldsymbol{O}_1^T|\boldsymbol{\lambda}_{(r)})$
- 이를 위해 다음의 수식들을 정의하자.
  - $\alpha_t^{(r)}(j) = P(X_t = j, \boldsymbol{O}_1^t|\boldsymbol{\lambda}_{(r)})$
  - $\beta_t^{(r)}(i) = P(\boldsymbol{O}_{t+1}^T|X_t = i, \boldsymbol{\lambda}_{(r)})$
  - $\gamma_t^{(r)}(j) = P(X_t = j|\boldsymbol{O}_1^T, \boldsymbol{\lambda}_{(r)})$
- 첫번째 필요한 수식은 $P(X_1 = i|\boldsymbol{O}_1^T, \boldsymbol{\lambda}_{(r)}) = \gamma_1^{(r)}(i)$로 표현된다.

두번째 필요한 수식은 다음과 같이 표현된다.

$$
\begin{aligned}
P(X_{t-1}=i, X_t=j, \boldsymbol{O}_1^T|\boldsymbol{\lambda}_{(r)}) &= P(X_{t-1}=i, X_t=j, \boldsymbol{O}_1^{t-1}, O_t, \boldsymbol{O}_{t+1}^T|\boldsymbol{\lambda}_{(r)}) \\
&= P(\boldsymbol{O}_{t+1}^T|X_t=j, X_{t-1}=i, O_t, \boldsymbol{O}_1^{t-1}, \boldsymbol{\lambda}_{(r)}) \\
&\quad \times P(O_t|X_t=j, X_{t-1}=i, \boldsymbol{O}_1^{t-1}, \boldsymbol{\lambda}_{(r)}) \\
&\quad \times P(X_t=j|X_{t-1}=i, \boldsymbol{O}_1^{t-1}, \boldsymbol{\lambda}_{(r)}) \times P(X_{t-1}=i, \boldsymbol{O}_1^{t-1}|\boldsymbol{\lambda}_{(r)}) \\
&= P(\boldsymbol{O}_{t+1}^T|X_t=j, \boldsymbol{\lambda}_{(r)}) \times P(O_t|X_t=j, \boldsymbol{\lambda}_{(r)}) \\
&\quad \times P(X_t=j|X_{t-1}=i, \boldsymbol{\lambda}_{(r)}) \times P(X_{t-1}=i, \boldsymbol{O}_1^{t-1}|\boldsymbol{\lambda}_{(r)}) \\
&= \beta_t^{(r)}(j) b_j^{(r)}(O_t) a_{ij}^{(r)} \alpha_{t-1}^{(r)}(i)
\end{aligned}
$$

세번째 필요한 수식은 다음과 같이 표현된다.

$$
\begin{aligned}
P(X_t=j, \boldsymbol{O}_1^T|\boldsymbol{\lambda}_{(r)}) &= P(X_t=j|\boldsymbol{O}_1^T, \boldsymbol{\lambda}_{(r)}) P(\boldsymbol{O}_1^T|\boldsymbol{\lambda}_{(r)}) \\
&= \gamma_t^{(r)}(j) P(\boldsymbol{O}_1^T|\boldsymbol{\lambda}_{(r)})
\end{aligned}
$$

- $P(\boldsymbol{O}_1^T|\lambda_{(r)})$는 다음과 같이 $\alpha_t(i), \beta_t(i)$로 표현할수 있다.

$$
\begin{aligned}
P(\boldsymbol{O}_1^T|\lambda_{(r)}) &= \sum_{i=1}^{N} P(X_t = i, \boldsymbol{O}_1^T|\lambda_{(r)}) \\
&= \sum_{i=1}^{N} P(X_t = i, \boldsymbol{O}_1^t, \boldsymbol{O}_{t+1}^T|\lambda_{(r)}) \\
&= \sum_{i=1}^{N} P(\boldsymbol{O}_{t+1}^T|X_t = i, \boldsymbol{O}_1^t, \lambda_{(r)}) P(X_t = i, \boldsymbol{O}_1^t|\lambda_{(r)}) \\
&= \sum_{i=1}^{N} \beta_t^{(r)}(i)\alpha_t^{(r)}(i)
\end{aligned}
$$

- 참고로 $P(\boldsymbol{O}_1^T|\lambda_{(r)})$은 관측값($\boldsymbol{O}_1^T$)의 가능도(Likelihood)이다.

정리하면

$$
\begin{aligned}
\hat{\pi}_i^{(r+1)} &= P(X_1 = i | \boldsymbol{O}_1^T, \boldsymbol{\lambda}_{(r)}) = \gamma_1^{(r)}(i) \\
\hat{a}_{ij}^{(r+1)} &= \frac{\sum_{t=2}^{T} P(X_{t-1} = i, X_t = j, \boldsymbol{O}_1^T | \boldsymbol{\lambda}_{(r)})}{\sum_{t=2}^{T} P(X_{t-1} = i, \boldsymbol{O}_1^T | \boldsymbol{\lambda}_{(r)})} \\
&= \frac{\sum_{t=2}^{T} \alpha_{t-1}^{(r)}(i) a_{ij}^{(r)} b_j^{(r)}(O_t) \beta_t^{(r)}(j)}{\sum_{t=2}^{T} \gamma_{t-1}^{(r)}(i) P(\boldsymbol{O}_1^T | \boldsymbol{\lambda}_{(r)})} \\
&= \frac{\sum_{t=2}^{T} \alpha_{t-1}^{(r)}(i) a_{ij}^{(r)} b_j^{(r)}(O_t) \beta_t^{(r)}(j)}{\sum_{t=2}^{T} \gamma_{t-1}^{(r)}(i) \left( \sum_{k=1}^{N} \alpha_t^{(r)}(k) \beta_t^{(r)}(k) \right)}
\end{aligned}
$$

$$
\begin{aligned}
\hat{b}_j^{(r+1)}(o_l) &= \frac{\sum_{t=1}^{T} P(X_t = j, \boldsymbol{O}_1^T | \boldsymbol{\lambda}_{(r)}) \delta_{o_l}(O_t)}{\sum_{t=1}^{T} P(X_t = j, \boldsymbol{O}_1^T | \boldsymbol{\lambda}_{(r)})} \\
&= \frac{\sum_{t=1}^{T} \gamma_t^{(r)}(j) P(\boldsymbol{O}_1^T | \boldsymbol{\lambda}_{(r)}) \delta_{o_l}(O_t)}{\sum_{t=1}^{T} \gamma_t^{(r)}(j) P(\boldsymbol{O}_1^T | \boldsymbol{\lambda}_{(r)})} \\
&= \frac{\sum_{t=1}^{T} \gamma_t^{(r)}(j) \delta_{o_l}(O_t)}{\sum_{t=1}^{T} \gamma_t^{(r)}(j)}
\end{aligned}
$$

- 이제 구해야 하는 수식들은 $\boldsymbol{\lambda}$가 주어졌을때 다음과 같다.
  - $\alpha_t(j) = P(X_t = i, \boldsymbol{O}_1^t | \boldsymbol{\lambda})$: 전향 확률 (forward probability)
  - $\beta_t(i) = P(\boldsymbol{O}_{t+1}^T | X_t = i, \boldsymbol{\lambda})$: 후향 확률 (backward probability)
  - $\gamma_t(j) = P(X_t = j | \boldsymbol{O}_1^T, \boldsymbol{\lambda})$

$\alpha_t(j)$구하기:

$$
\begin{aligned}
\alpha_t(j) &= P(\boldsymbol{O}_1^t, X_t = j|\boldsymbol{\lambda}) \\
&= \sum_{i=1}^{N} P(\boldsymbol{O}_1^t, X_{t-1} = i, X_t = j|\boldsymbol{\lambda}) \\
&= \sum_{i=1}^{N} P(O_t|\boldsymbol{O}_1^{t-1}, X_{t-1} = i, X_t = j, \boldsymbol{\lambda}) \\
&\quad \times P(X_t = j|\boldsymbol{O}_1^{t-1}, X_{t-1} = i, \boldsymbol{\lambda}) P(\boldsymbol{O}_1^{t-1}, X_{t-1} = i|\boldsymbol{\lambda}) \\
&= \sum_{i=1}^{N} P(O_t|X_t = j, \boldsymbol{\lambda}) P(X_t = j|X_{t-1} = i, \boldsymbol{\lambda}) \alpha_{t-1}(i) \\
&= \sum_{i=1}^{N} b_j(O_t)\, a_{ij}\, \alpha_{t-1}(i)
\end{aligned}
$$

$$
\begin{aligned}
\alpha_1(j) &= P(O_1, X_1 = j|\boldsymbol{\lambda}) = P(O_1|X_1 = j, \lambda) P(X_1 = j|\boldsymbol{\lambda}) \\
&= b_j(O_1)\pi_j
\end{aligned}
$$

$\alpha_1(j)$로부터 반복적으로 $\alpha_t(j)$를 구할수 있다.

1. 초기화(Initialization)

$$\alpha_1(j) = b_j(O_1)\pi_j,$$

$j = 1, \cdots, N$

2. 반복(Recursion), $k = 2, \cdots, t$,

$$\alpha_k(j) = \sum_{i=1}^{N} b_j(O_k)a_{ij}\alpha_{k-1}(i),$$

$j = 1, \cdots, N$.

참고로, $Likelihood(\lambda) = P(\boldsymbol{O}_1^T|\lambda) = \sum_{j=1}^{N} \alpha_T(j)$이므로 $k$를 $T$까지 구하는 경우 가능도 함수를 계산할수 있다. 이러한 알고리즘을 전향 알고리즘 (forward algorithm)이라 부른다.

$\beta_t(i)$구하기:

$$\beta_t(i) = P(\boldsymbol{O}_{t+1}^T | X_t = i, \boldsymbol{\lambda}) = \sum_{j=1}^{N} P(O_{t+1}, \boldsymbol{O}_{t+2}^T, X_{t+1} = j | X_t = i, \boldsymbol{\lambda})$$

$$= \sum_{j=1}^{N} \Big( P(\boldsymbol{O}_{t+2}^T | O_{t+1}, X_{t+1} = j, X_t = i, \boldsymbol{\lambda})$$

$$\times P(O_{t+1} | X_{t+1} = j, X_t = i, \boldsymbol{\lambda}) \times P(X_{t+1} = j | X_t = i, \boldsymbol{\lambda}) \Big)$$

$$= \sum_{j=1}^{N} P(\boldsymbol{O}_{t+2}^T | X_{t+1} = j, \boldsymbol{\lambda}) P(O_{t+1} | X_{t+1} = j, \boldsymbol{\lambda}) \times a_{ij}$$

$$= \sum_{j=1}^{N} \beta_{t+1}(j) b_j(O_{t+1}) a_{ij}$$

$\beta_T(i) = 1$

$\beta_T(i)$로부터 반복적으로 $\beta_t(i)$를 구할수 있다.

1. 초기화(Initialization)
   $\beta_T(i) = 1, i = 1, \cdots, N$
2. 반복(Recursion), $k = T-1, \cdots, t$,

$$\beta_k(i) = \sum_{j=1}^{N} a_{ij} b_j(O_{k+1}) \beta_{k+1}(j),$$

$i = 1, \cdots, N$.

참고로, $Likelihood(\lambda) = P(\boldsymbol{O}_1^T|\lambda) = \sum_{i=1}^{N} \beta_1(i) b_i(O_1)\pi_i$이므로 $k$를 1까지 구하는 경우 가능도 함수를 계산할수 있다. 이러한 알고리즘을 후향 알고리즘 (backward algorithm)이라 부른다.

$\gamma_t(i)$구하기:

$$\gamma_t(i) = P(X_t = i|\boldsymbol{O}_1^T, \boldsymbol{\lambda}) = \frac{P(X_t = i, \boldsymbol{O}_1^T|\boldsymbol{\lambda})}{P(\boldsymbol{O}_1^T|\boldsymbol{\lambda})} = \frac{P(X_t = i, \boldsymbol{O}_1^T|\boldsymbol{\lambda})}{\sum_{j=1}^N P(X_t = j, \boldsymbol{O}_1^T|\boldsymbol{\lambda})}$$

한편, 앞에서 $P(\boldsymbol{O}_1^T|\boldsymbol{\lambda}) = \sum_{j=1}^N \alpha_t(j)\beta_t(j)$임을 보였고, 이 과정에서 $P(X_t = i, \boldsymbol{O}_1^T|\boldsymbol{\lambda}) = \alpha_t(i)\beta_t(i)$임을 알 수 있었다.

따라서

$$\gamma_t(i) = P(X_t = i|\boldsymbol{O}_1^T, \boldsymbol{\lambda}) = \frac{\alpha_t(i)\beta_t(i)}{\sum_{j=1}^N \alpha_t(j)\beta_t(j)}$$

- $\alpha_t(j)$, $\beta_t(i)$, $\gamma_t(i)$들을 구할수 있으므로 해당식들을 이용하여 $\boldsymbol{\lambda}$를 구하는 EM알고리즘에 대입하여 최종 학습한 $\boldsymbol{\lambda}$를 구하게 된다.

# (2) 디코딩

- 디코딩 단계에서는 주어진 데이터 $\boldsymbol{O}_1^T$와 주어진 $\boldsymbol{\lambda}$(학습한 값)에서 최적의 은닉변수열 $\boldsymbol{X}_1^T$ 찾는다.
- 본 강의에서는 $P(\boldsymbol{X}_1^T|\boldsymbol{O}_1^T, \boldsymbol{\lambda})$를 최대화 시키는 $\boldsymbol{X}_1^T$찾는것으로 생각한다.
- $P(\boldsymbol{X}_1^T|\boldsymbol{O}_1^T, \boldsymbol{\lambda})$를 각 $\boldsymbol{X}_1^T$에 대해서 계산하여 최대가 되는 $\boldsymbol{X}_1^T$를 찾아야 하므로 계산 복잡도는 $O(N^T)$.
- 따라서 계산 복잡도를 줄이는 방법이 필요하다.

# 비터비 알고리즘 (Viterbi algorithm)

- 반복적으로 찾아나가는 접근 방식
- 먼저 다음의 성질을 고려

$$\text{argmax}_{\boldsymbol{X}_1^T} P(\boldsymbol{X}_1^T | \boldsymbol{O}_1^T, \lambda) = \text{argmax}_{\boldsymbol{X}_1^T} P(\boldsymbol{X}_1^T, \boldsymbol{O}_1^T | \lambda)$$

$$\max_{\boldsymbol{X}_1^T} P(\boldsymbol{X}_1^T, \boldsymbol{O}_1^T | \lambda) = \max_j \left( \max_{\boldsymbol{X}_1^{T-1}} P(\boldsymbol{X}_1^{T-1}, X_T = j, \boldsymbol{O}_1^T | \lambda) \right)$$

- $\nu_t(j) = \max_{\boldsymbol{x}_1^{t-1}} P(\boldsymbol{X}_1^{t-1}, X_t = j, \boldsymbol{O}_1^t | \lambda)$로 정의하면 최대화 시키는 값은 다음과 같이 표현된다.
- $\max_{\boldsymbol{x}_1^T} P(\boldsymbol{X}_1^T, \boldsymbol{O}_1^T | \lambda) = \max_{j=1,\cdots,N} \nu_T(j)$
- 이때, $\nu_T(j)$는 다음의 성질을 이용하여 반복적으로 (recursively) 구하게 된다.

$$
\begin{aligned}
\nu_T(j) &= \max_{\boldsymbol{X}_1^{T-1}} P(\boldsymbol{X}_1^{T-1}, X_T = j, \boldsymbol{O}_1^T|\lambda) \\
&= \max_{i=1}^N \max_{\boldsymbol{X}_1^{T-2}} P(\boldsymbol{X}_1^{T-2}, X_{T-1} = i, X_T = j, \boldsymbol{O}_1^T|\lambda) \\
&= \max_{i=1}^N \max_{\boldsymbol{X}_1^{T-2}} P(O_T|\boldsymbol{X}_1^{T-2}, X_{T-1} = i, X_T = j, \boldsymbol{O}_1^{T-1}, \lambda) \\
&\qquad \times P(X_T = j|\boldsymbol{X}_1^{T-2}, X_{T-1} = i, \boldsymbol{O}_1^{T-1}, \lambda) \\
&\qquad \times P(\boldsymbol{X}_1^{T-2}, X_{T-1} = i, \boldsymbol{O}_1^{T-1}|\lambda) \\
&= \max_{i=1}^N b_j(O_T)\, a_{ij} \max_{\boldsymbol{X}_1^{T-2}} P(\boldsymbol{X}_1^{T-2}, X_{T-1} = i, \boldsymbol{O}_1^{T-1}|\lambda) \\
&= \max_{i=1}^N b_j(O_T)\, a_{ij}\, \nu_{T-1}(i)
\end{aligned}
$$

- 반복적으로는 $\nu_t(j) = \max_{i=1}^N \nu_{t-1}(i)\, a_{ij}\, b_j(O_t)$이 된다.
- $\nu_1(j) = P(X_1 = j, O_1|\lambda) = P(O_1|X_1 = j, \lambda)P(X_1 = j|\lambda) = b_j(O_1)\pi_j$.

- 한편, $\max_{\boldsymbol{X}_1^T} P(\boldsymbol{X}_1^T, \boldsymbol{O}_1^T|\boldsymbol{\lambda}) = \max_{j=1,\cdots,N} \nu_T(j)$를 만족하는 $\boldsymbol{X}_1^T$를 찾는것이 목적이므로 각 단계마다 최대가 되는 $X_t$를 구하기 위한 정보(backtrace 또는 backpointer)를 저장해 두어야 한다.
- 즉, $\nu_t(j) = \max_{i=1}^N \nu_{t-1}(i)\, a_{ij}\, b_j(O_t)$를 구할때,
- $BT_t(j) = \text{argmax}_{i=1}^N \nu_{t-1}(i)\, a_{ij}\, b_j(O_t)$를 기록해 둔다.
- 마지막 스텝 $T$에서는 $\nu_T(j)$를 이용해서 최대가 되는 $x_T$를 찾는다. 즉 $\hat{X}_T = \text{argmax}_j\, \nu_T(j)$.

비터비 알고리즘 (Viterbi algorithm)

1. 초기화(Initialization)

   $\nu_1(j) = b_j(O_1)\pi_j, j = 1, \cdots, N$

   $BT_1(j) = 0, j = 1, \cdots, N$

2. 반복(Recursion), $t = 2, \cdots, T$,

   $\nu_t(j) = \max_{i=1}^{N} \nu_{t-1}(i)a_{ij}b_j(O_t), j = 1, \cdots, N.$

   $BT_t(j) = \text{argmax}_{i=1}^{N} \nu_{t-1}(i)a_{ij}b_j(O_t), j = 1, \cdots, N.$

3. 종료(Termination)

   Best score $= \max_{i=1}^{N} \nu_T(i)$

   $\hat{X}_T = \text{argmax}_{i=1}^{T} \nu_T(i)$

최적의 $\boldsymbol{X}_1^T$는 어떻게 찾을까?

- 마지막 스텝에서 Best score $\nu_T(i^*)$를 주는 인덱스 $i^*$가 정해지므로 $\hat{X}_T = i^*$를 찾게 된다.
- 해당 인덱스 $i^*$에 대하여 $\nu_T(i^*) = \max_{i=1}^{N} \nu_{T-1}(i) a_{ii^*} b_{i^*}(O_t)$ 값을 만족하는 $i$를 찾을수 있다.
- 해당 인덱스가 $BT_T(i^*)$이고, $T-1$ 스텝에서의 최적의 은닉변수의 상태는 $\hat{X}_{T-1} = i^{**}$가 된다.
- 이를 반복적으로 진행하면 모든 최적의 은닉변수열 $\hat{\boldsymbol{X}}_1^T$를 찾게 된다.