Hidden Markov Models

Outline

- (Visible) Markov Models (VMMs)
- Hidden Markov Models (HMMs)
- The Three Fundamental Questions for HMMs
- Forward Algorithm & Backward Algorithm
- Viterbi Algorithm
- Baum-Welch Algorithm

Markov Models

- แบบจำลองมาคอฟ คือ แบบจำลองที่ใช้ในการจำลองความน่าจะเป็นของลำดับของเหตุการณ์ (a sequence of events) ที่เราสนใจ
  - เช่น เราสนใจลำดับของสภาวะอากาศในแต่ละวัน จึงต้องการสร้างแบบจำลองของน่าจะเป็น เพื่อใช้ในการพยากรณ์อากาศล่วงหน้า
    - $X = (X_1, ..., X_T)$ ตัวแปรสุ่มแทนวันแต่ละวัน
    - $S = \{\text{raining}, \text{hot}, \text{cold}\}$ ค่าของตัวแปรสุ่มคือสภาวะอากาศ
    - $P(X_t|X_1, ..., X_{t-1})$ คือความน่าจะเป็นของสภาวะอากาศโดยคาดเดาจากสภาวะอากาศในวันก่อนหน้า

Markov Models

- โดยแบบจำลองมาคอฟจะตั้งอยู่บนสมมุติฐานว่า
  - Limited horizon:

$$P(X_{t+1})|X_1, ..., X_t) = P(X_{t+1}|X_t)$$

  - Time invariant: สมมุติฐาน Limited horizon เป็นจริงสำหรับทุกๆ ช่วงเวลา (ทุกๆ ค่า t)

$$P(X_3|X_1, ..., X_2) = P(X_3|X_2)$$
$$P(X_4|X_1, ..., X_3) = P(X_4|X_3)$$
$$P(X_5|X_1, ..., X_4) = P(X_5|X_4)$$
$$\vdots$$

Markov Models

- เราสามารถอธิบาย Markov chain ได้ใช้ stochastic transition matrix A:

$$a_{ij} = P(X_{t+1} = s_j \mid X_t = s_i)$$

  - โดยที $a_{ij} \geqslant 0, \forall i, j$ and $\sum_{j=1}^{N} a_{ij} = 1, \forall i$

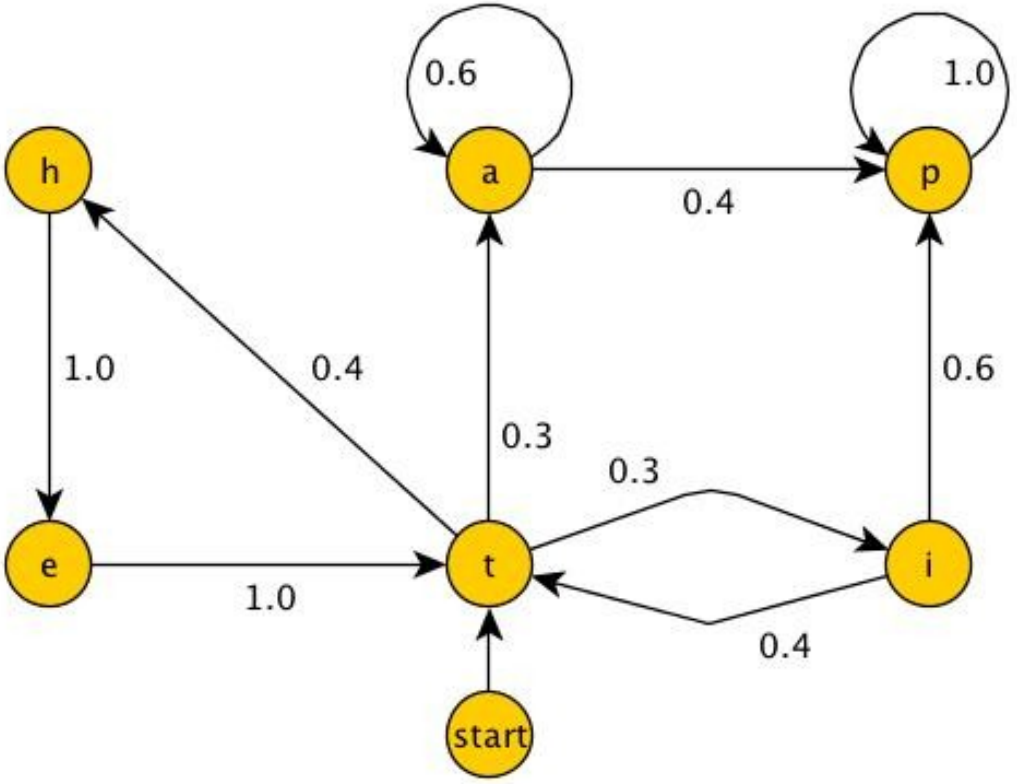
- และเวกเตอร์ Π Markov chain

$$\Pi_i = P(X_1 = s_i)$$

  - $\sum_{i=1}^{N} \Pi_i = 1$

A Markov model

ในกรณีนี state (หรือ deterministic function ใดๆ) คือ output



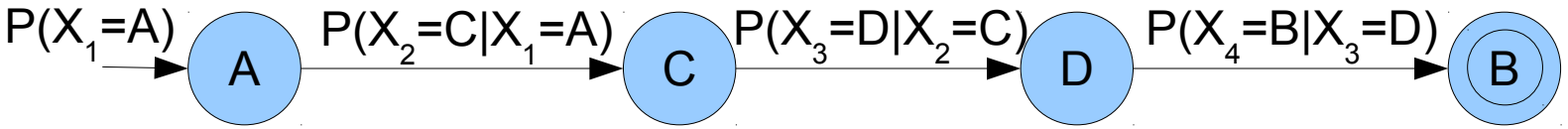
$$P(X_1, \ldots, X_T) = \Pi_1 \prod_{t=1}^{T-1} a_{t,t+1}$$

Markov Models

- ชัดเจนว่าแบบจำลองไบแกรม (Bigram models) คือแบบจำลองมาคอฟ
- แล้ว N-gram คือแบบจำลองมาคอฟหรือเปล่า ?
  - Trigram: $P(X_t|X_1,...,X_{t-1}) = P(X_t|X_{t-2},X_{t-1})$
- เราสามารถเปลียนรูปแบบจำลอง n-gram ใดๆ ให้เป็นแบบจำลองไบแกรมได้
- แบบจำลอง n-gram คือ (n-1)th order Markov model

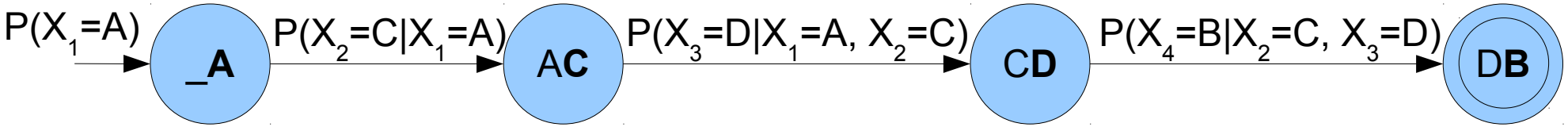
ตัวอย่าง

- ตัวอย่างการเปลียนแบบจำลองไตรแกรมให้อยู่ในรูปของแบบจำลองมาคอฟ



$P(X_1=A,X_2=C,X_3=D,X_4=B) = P(X_1=A)P(X_2=C|X_1=A)P(X_3=D|X_2=C)P(X_4=B|X_3=D)$

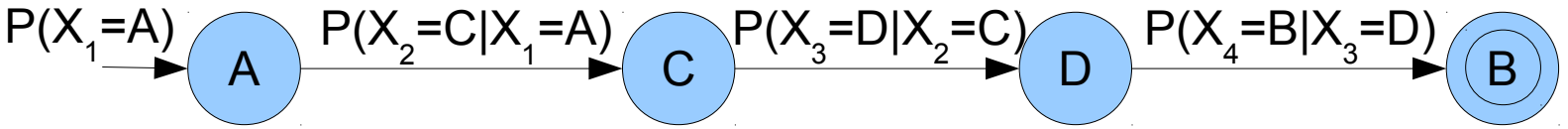
สร้างสถานะใหม่จากการทำ cross product ของสถานะเดิม $S_{new} = S \times S$



$P(X_1=A,X_2=C,X_3=D,X_4=B) = P(X_1=A)P(X_2=C|X_1=A)P(X_3=D|X_1=A, X_2=C)P(X_4=B|X_2=C, X_3=D)$

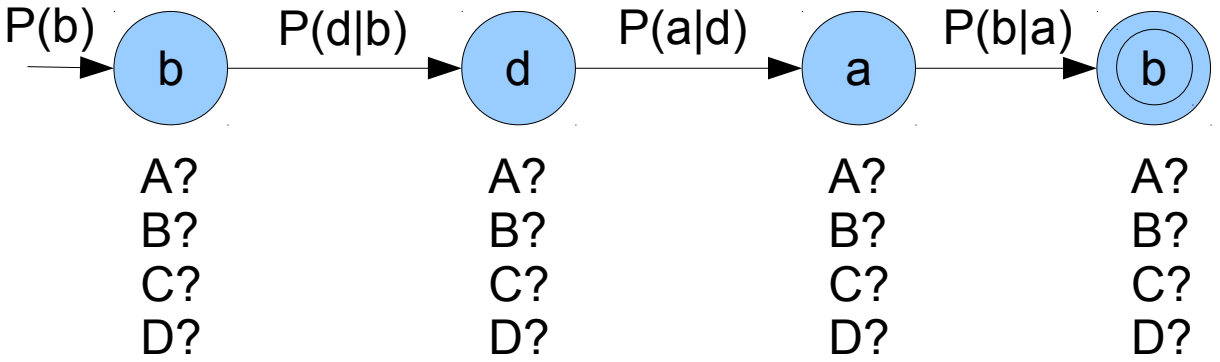
Hidden Markov Models

- HMMs คือแบบจำลองมาคอฟที**เราไม่รู้ว่าลำดับของสถานะคืออะไร** แต่รู้เพียงฟังก์ชันความน่าจะเป็นบางส่วนของแบบจำลองนั้น



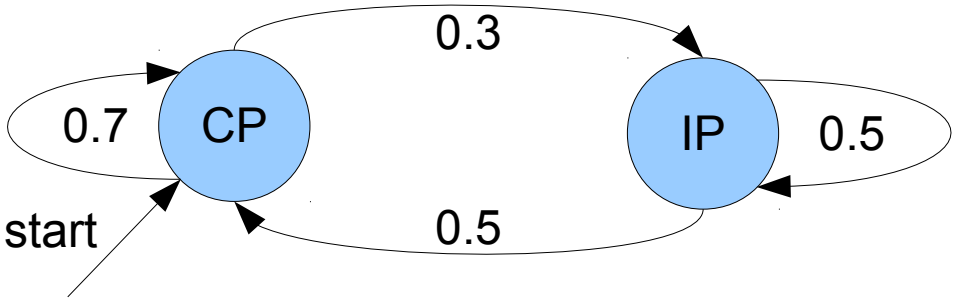
สมมุติให้ A = {a,b,c}, B = {a,b,d}, C = {a,c,d}, D= {b,c,d}

เรารู้เพียงความน่าจะเป็นของผลลัพธ์เท่านั้น



ตัวอย่าง

เครื่องขายน้ำตึงต้องมี 2 สถานะคือสถานะทีมักจะให้โคล่า (CP) กับ สถานะทีมักจะให้ชามะนาว (IP)



ในเครื่องขายน้ำปกติ ถ้าอยู่ในสถานะ CP จะให้โคล่าเสมอ และถ้าอยู่ในสถานะ IP จะให้ชามะนาวเสมอ แต่สำหรับเครื่องขายน้ำตึงต้องในแต่ละสถานะมีโอกาสจะให้น้ำอะไรก็ได้ ดังนั้นจำเป็นต้องมี emission probabilities สำหรับการสังเกต

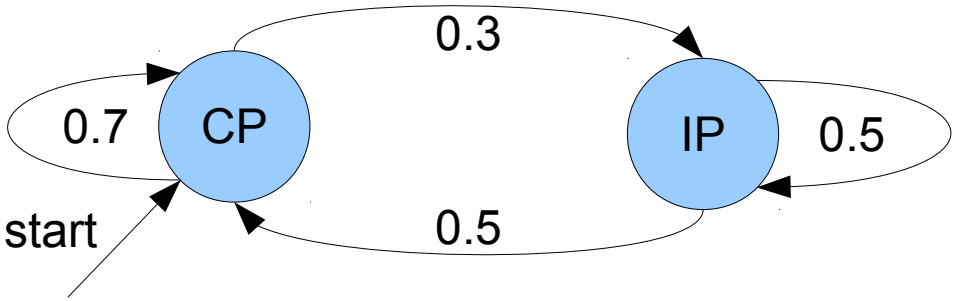
	โคล่า (cl)	ชามะนาว (it)	น้ำมะนาว (lm)
CP	0.6	0.1	0.3
IP	0.1	0.7	0.2

$$P(O_t = k | X_t = s_i, X_{t+1} = s_j) = b_{ijk}$$

P(cl | IP, CP) = P(cl | CP) = 0.6

ในตัวอย่างนีขึนอยู่กับสถานะปัจจุบันเท่านัน

ตัวอย่าง



	โคล่า (cl)	ชามะนาว (it)	น้ำมะนาว (lm)
CP	0.6	0.1	0.3
IP	0.1	0.7	0.2

จากแบบจำลองนี้ ค่าความน่าจะเป็นที่จะเกิดเหตุการณ์ {lm, it} เป็นเท่าไร (ครั้งแรกได้น้ำมะนาวและครั้งที่สองได้ชามะนาว)

= P(CP|CP)P(lm|CP) P(CP|CP)P(it|CP) + P(CP|CP)P(lm|CP) P(IP|CP)P(it|CP) + P(IP|CP)P(lm|CP) P(CP|IP)P(it|IP) + P(IP|CP)P(lm|CP) P(IP|IP)P(it|IP)

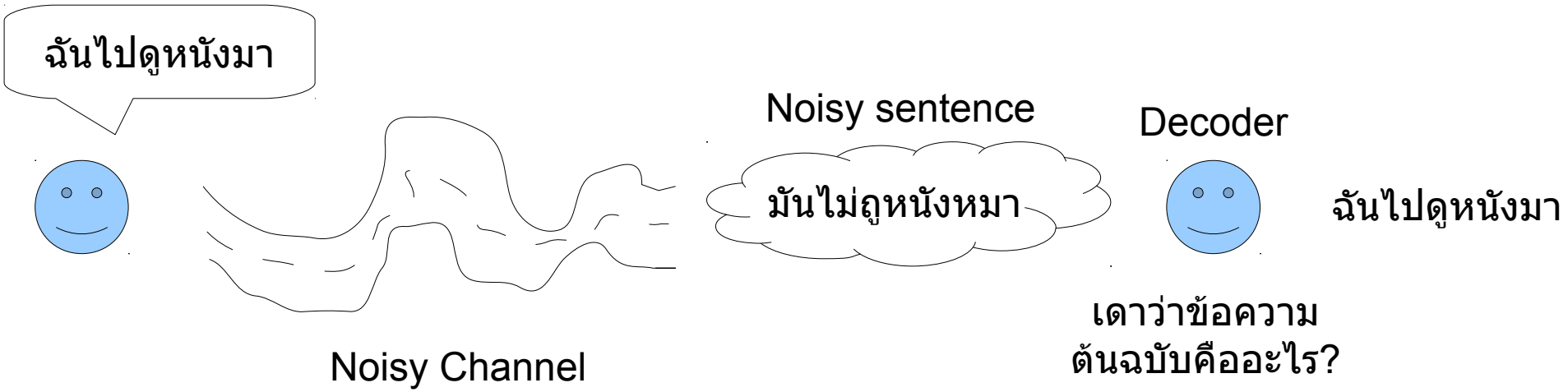
= 0.7x0.3x0.7x0.1 + 0.7x0.3x0.3x0.1 + 0.3x0.3x0.5x0.7 + 0.3x0.3x0.5x0.7

= 0.084

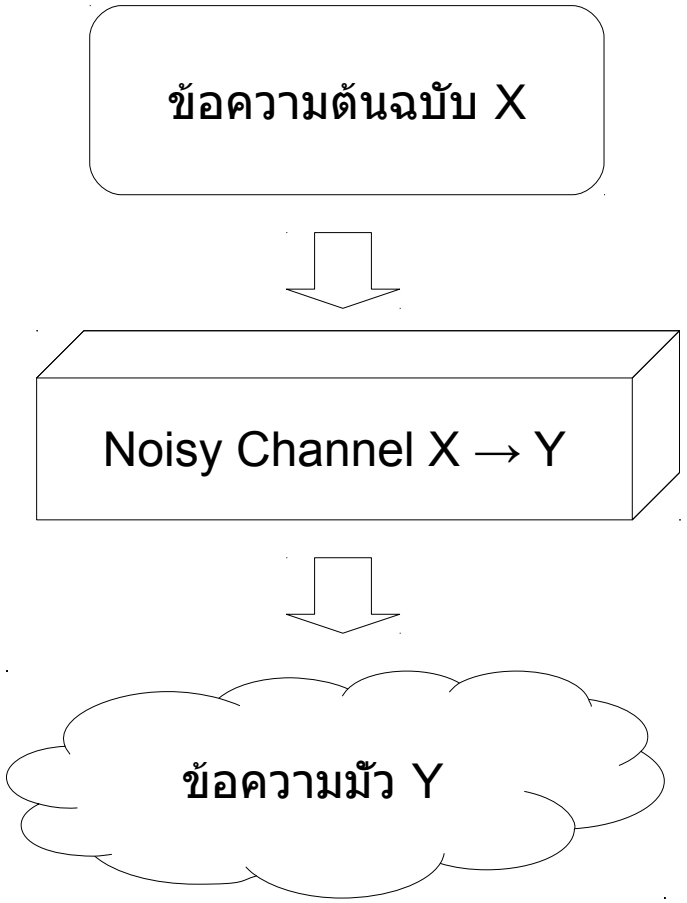
Why use HMMs?

- มีสถานะการณ์จำนวนมากทีสามารถจำลองได้ว่ามีเหตุการณ์ทีซ่อนอยู่ (underlying events) ซึงเหตุการณ์เหล่านั้น จะสร้างเหตุการณ์อืนทีเราสามารถสังเกตได้ออกมา เช่น การกำกับ POS
- เราสามารถมองเหตุการณ์เหล่านีได้ว่าเป็น noisy channel model

Noisy Channel Models



Noisy Channel Models



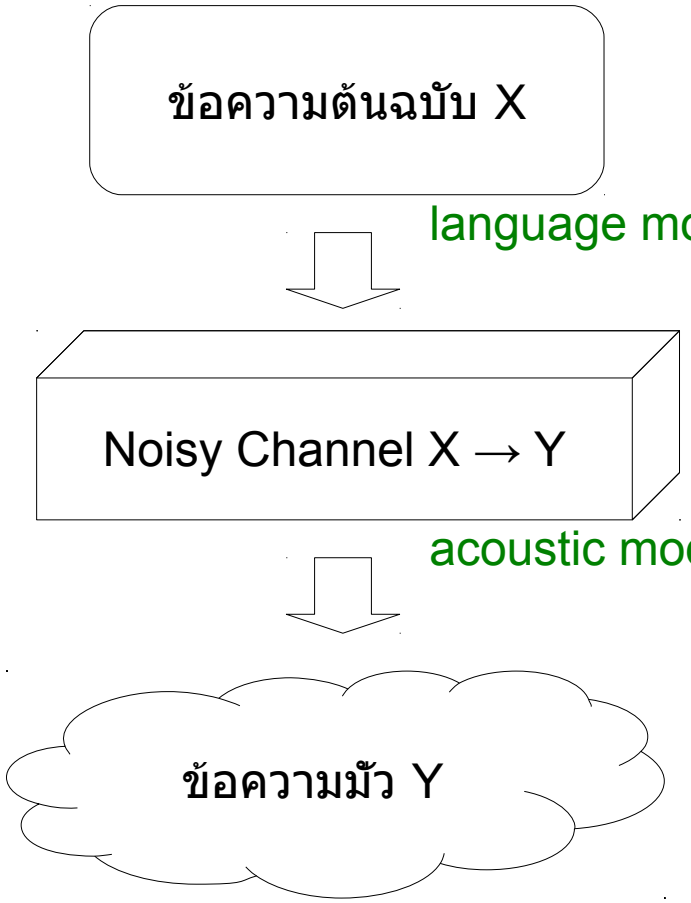
ข้อความทีพิมพ์ถูก

การพิมพ์ผิด

ข้อความทีพิมพ์ผิด

เราต้องการหา X จาก Y

Noisy Channel Models



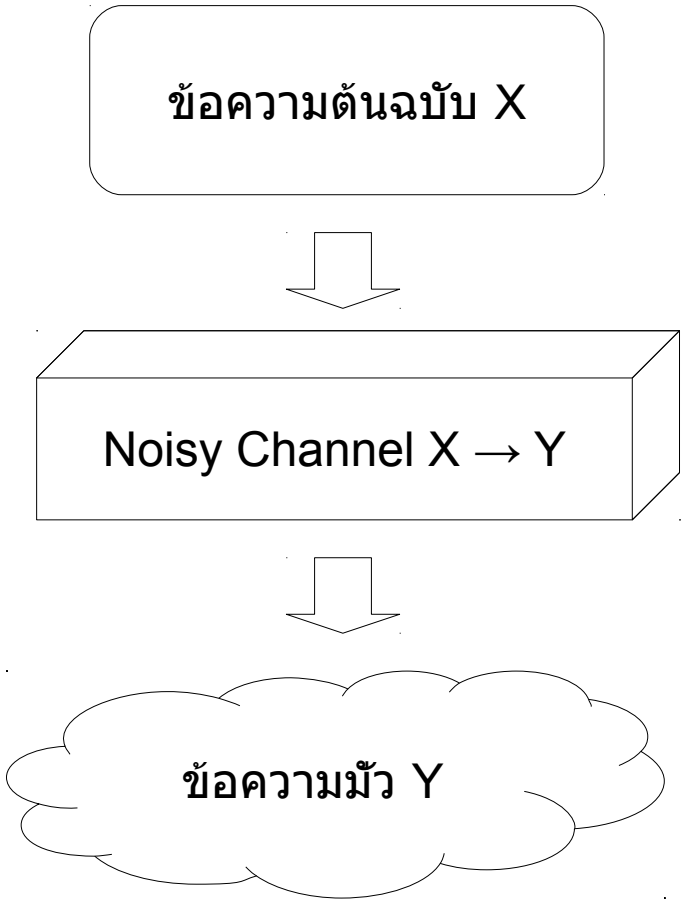
คำศัพท์ทั้งหมด

การออกเสียง

เสียงพูดที่ได้ยิน

เราต้องการหา X จาก Y

Noisy Channel Models



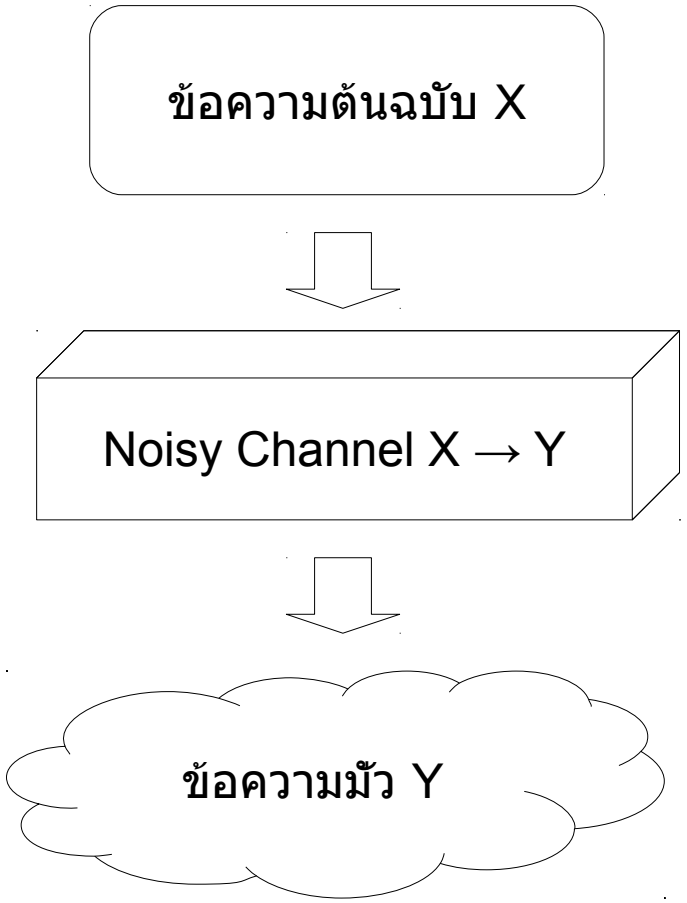
ภาษาไทย

การแปลไทย-อังกฤษ

ภาษาอังกฤษ

เราต้องการหา X จาก Y

Noisy Channel Models



ชนิดของคำทั้งหมด

การแทนชนิดคำด้วยคำ

ข้อความ

เราต้องการหา X จาก Y

General form of an HMM

HMM สามารถกำหนดโดยใช้ (S,K,Π,A,B) **โดยที**

S **คือเซตของสถานะ**: $\{s_1,...,s_N\}$

K **คือผลลัพธ์ทีเป็นไปได้ทั้งหมด**: $\{k_1,...,k_M\}$

Π **คือ** initial state probabilities: $\{\pi_i\}$, $i \in S$

A **คือ** state transition probabilities: $\{a_{ij}\}$, $i,j \in S$

B **คือ** symbol emission probabilities: $\{b_{ijk}\}$, $i,j \in S$, $k \in K$

ส่วน state sequence **แทนโดย** $X = (X_1,...,X_{T+1})$, $X_t: S \rightarrow \{1,...,N\}$

และ output sequence **แทนโดย** $O = (o_1,...,o_T)$, $o_t \in K$

<u>ข้อสังเกต</u> จากตัวอย่างเครื่องขายน้ำดื่งต้องค่าของ symbol emission probabilities จะขึนอยู่กับสถานะ t เท่านัน ซึง
เรียกว่า state-emission HMM ส่วน HMM ปกติ จะขึนกับ t+1 ด้วย ซึงเรียกว่า arc-emission HMM

The Three Fundamental Questions for HMMs

มีสามปัญหาหลักที่เราต้องการจะรู้เกียวกับ HMM

1)ถ้าให้แบบจำลอง $\mu = (A,B,\Pi)$ มา เราจะหาคำนวณหาค่าความน่าจะเป็นของผลลัพธ์ที่กำหนดได้อย่างไร?

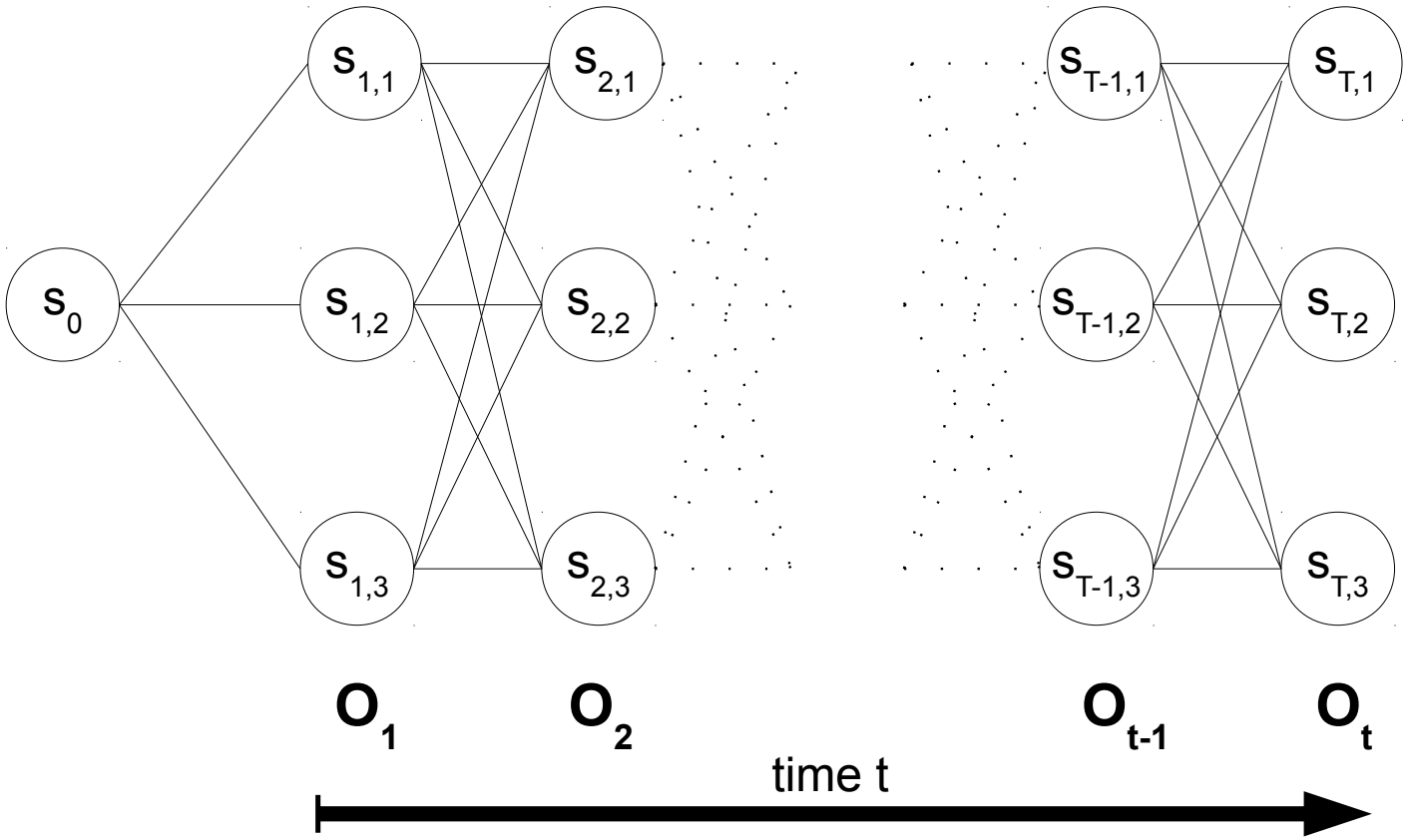
2)ถ้าให้ลำดับของการสังเกต (ผลลัพธ์) O และแบบจำลอง μ เราจะหา state sequence $(X_1,...,X_{T+1})$ที่ดีที่สุดในการอธิบายลำดับของการสังเกตได้อย่างไร?

3)ถ้าให้ลำดับของการสังเกต O และแบบจำลองต่างๆ ที่เกิดจากการปรับตัวแปรต่างๆ $\mu = (A,B,\Pi)$ เราจะหาแบบจำลองที่ดีที่สุดในการอธิบายลำดับของการสังเกตได้อย่างไร?

ปัญหาที่ 1: Finding the probability of an observation

- สิ่งที่ต้องการหาคือ $P(O \mid \mu)$
- หาได้จากผลรวมของสถานะที่เป็นไปได้ทั้งหมดในแบบจำลอง (เหมือนในตัวอย่างเครื่องขายน้ำ)
- ถ้าใช้วิธีหาแบบโง่ๆ จะใช้เวลาในการคำนวณมาก เพราะจำนวนเส้นทางที่เป็นไปได้คือ N^T โดยที่ N คือ จำนวนของสถานะ และ T คือจำนวนของการสังเกต
- แม้แบบจำลองขนาดเล็กที่ N=10 และ T=10 จะมีจำนวนเส้นทาง 10,000,000,000 แบบ

The Trellis (Lattice)

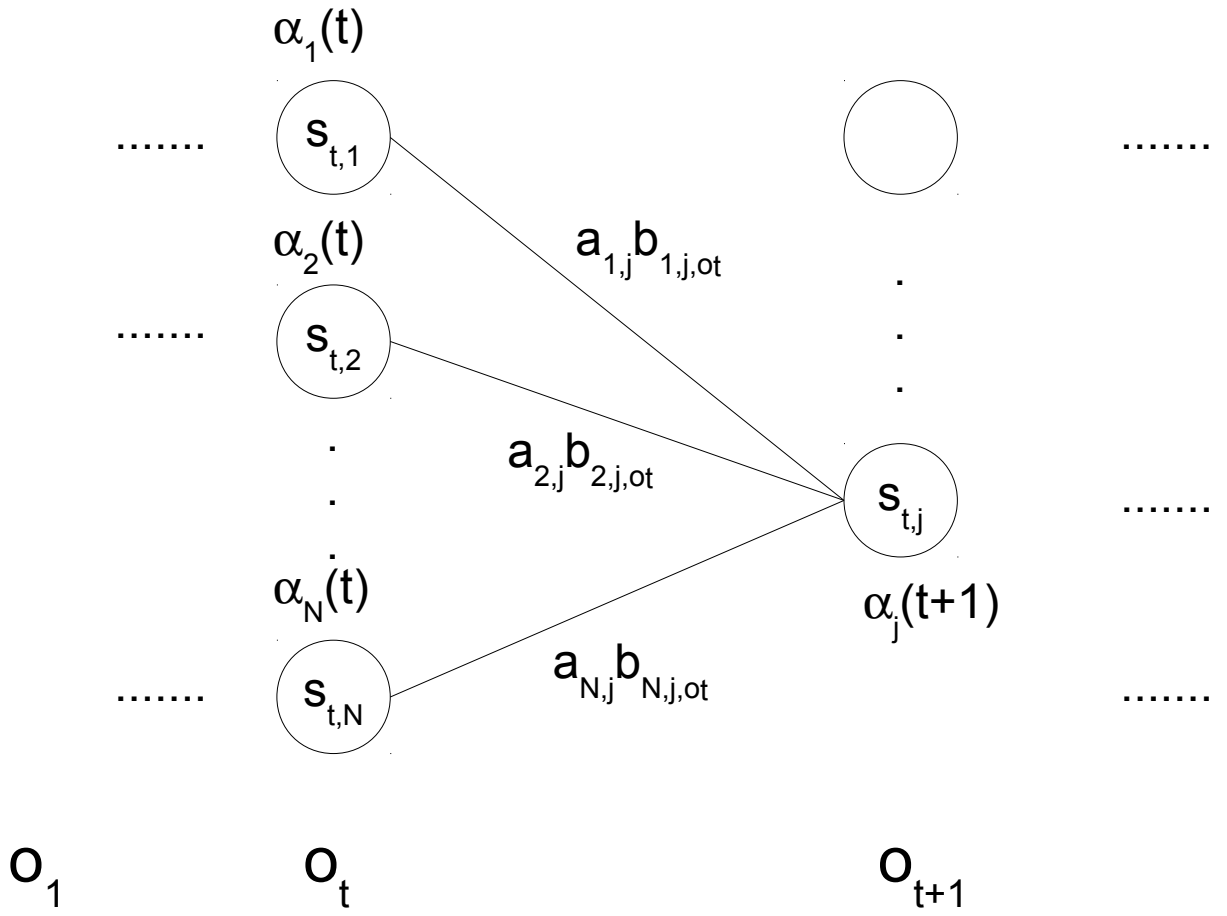


Forward Probabilities

- ให้แบบจำลอง μ มา ความน่าจะเป็น ณ เวลา t ที่สถานะ i สำหรับการสังเกตบางส่วน $o_1,\ldots,o_{t-1}$ คือ

$$\alpha_i(t) = P(o_1 \ldots o_{t-1}, X_t = i|\mu)$$

Forward Probabilities



$$\alpha_j(t+1) = \sum_{i=1}^{N} \alpha_i(t) \times a_{ij} \times b_{ijo_t}$$

Forward Algorithm

- Initialization:

$$\alpha_i(1) = \pi_i, \ 1 \leq i \leq N$$

- Induction:

$$\alpha_j(t+1) = \sum_{i=1}^{N} \alpha_i(t) \times a_{ij} \times b_{ijo_t}, \ 1 \leq t \leq T, 1 \leq j \leq N$$

- Total:

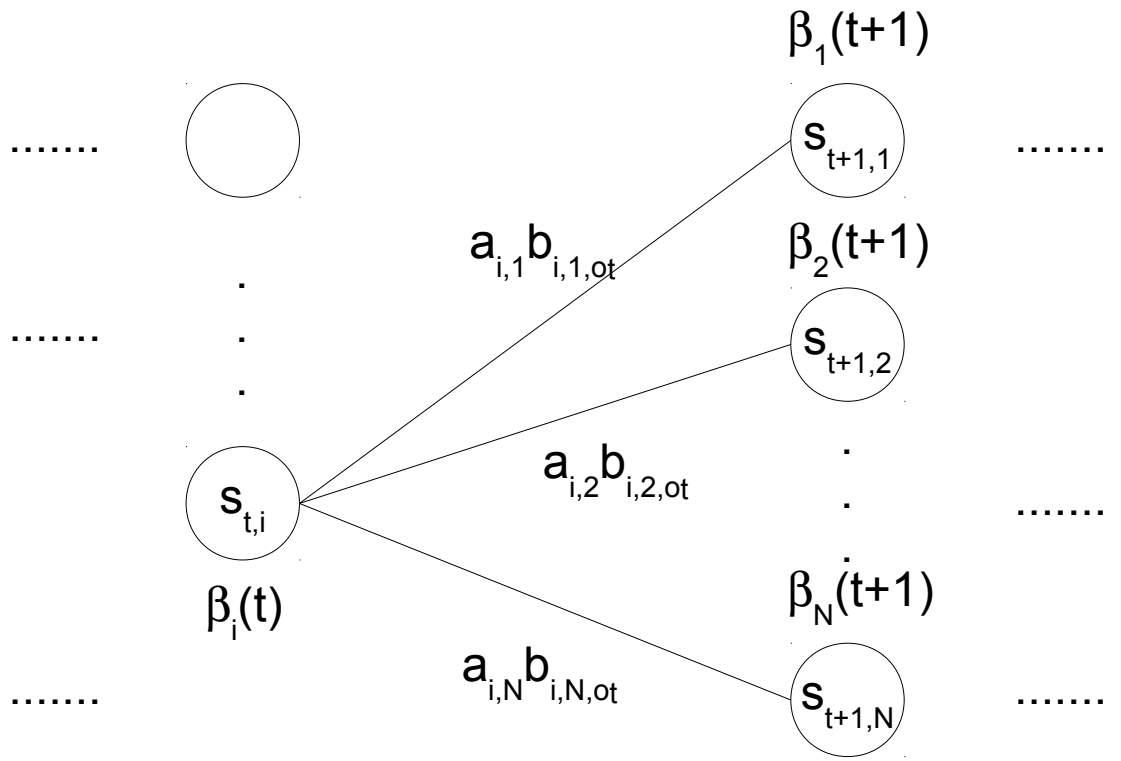
$$P(O|\mu) = \sum_{i=1}^{N} \alpha_i(T+1)$$

Backward Probabilities

- ให้แบบจำลอง μ มา ความน่าจะเป็น ณ เวลา t ที่สถานะ i สำหรับการสังเกตบางส่วน $o_t,\ldots,o_T$ คือ

$$\beta_i(t) = P(o_t \ldots o_T, X_t = i|\mu)$$

Backward Probabilities



$$\beta_i(t) = \sum_{j=1}^{N} a_{ij} \times b_{ijo_t} \times \beta_j(t+1)$$

Backward Algorithm

- Initialization:

$$\beta_i(T+1) = 1,\ 1 \leq i \leq N$$

- Induction:

$$\beta_i(t) = \sum_{j=1}^{N} a_{ij} \times b_{ijo_t} \times \beta_j(t+1),\ 1 \leq t \leq T, 1 \leq j \leq N$$

- Total:

$$P(O|\mu) = \sum_{i=1}^{N} \pi_i \times \beta_i(1)$$

Combining Forward and Backward

- เราสามารถใช้การรวมกันของ forward probabilities กับ backward probabilities เพื่อหาความน่าจะเป็นของลำดับการสังเกต $o_1,...,o_T$

$$
\begin{aligned}
P(O, X_t = i|\mu) &= P(o_1...o_T, X_t = i|\mu) \\
&= P(o_1...o_{t-1}, X_t = i, o_t...o_T|\mu) \\
&= P(o_1...o_{t-1}, X_t = i|\mu) \\
&\qquad \times P(o_t...o_T|o_1...o_{t-1}, X_t = i, \mu) \\
&= P(o_1...o_{t-1}, X_t = i|\mu)P(o_t...o_T|X_t = i, \mu) \\
&= \alpha_i(t)\beta_i(t)
\end{aligned}
$$

$$
P(O|\mu) = \sum_{i=1}^{N} \alpha_i(t)\beta_i(t), \ 1 \leq t \leq T+1
$$

ปัญหาที่ 2: Finding the best state sequence

- เมื่อให้แบบจำลอง μ และลำดับการสังเกต $o_1 \ldots o_T$
  - Forward algorithm จะให้ความน่าจะเป็นรวมของทุกเส้นทาง
  - สำหรับปัญหาที่ 2 เราต้องการหาเพียงเส้นทางที่ดีที่สุด (ความน่าจะเป็นสูงสุด)
  - เราต้องการลำดับของสถานะที

$$X^* = \arg\max_X P(X|O,\mu)$$

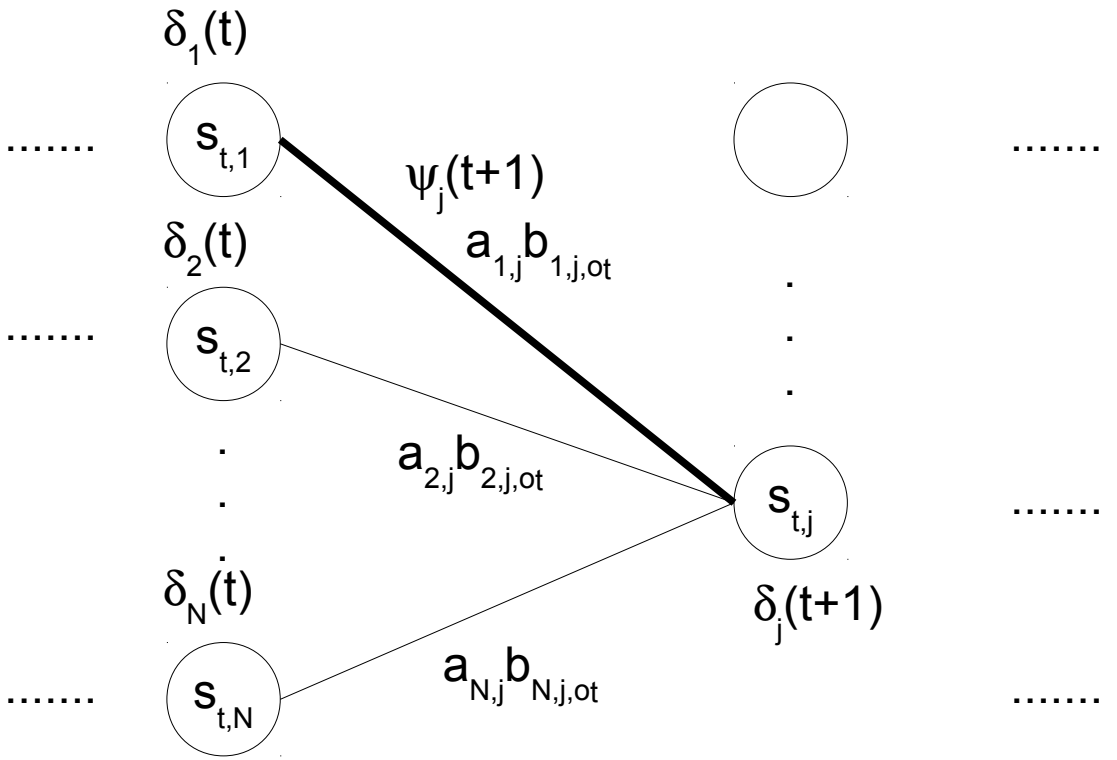
Viterbi Algorithm

- หลักการทำงานเหมือนกับ Forward algorithm แต่จะแตกต่างกันตรงทีเปลียนจากการหาผลรวม เป็นการหาค่าสูงสุดแทน

- Forward probability: $\alpha_j(t+1) = \sum_{i=1}^{N} \alpha_i(t) \times a_{ij} \times b_{ijo_t}$

- Viterbi recursion: $\delta_j(t+1) = \max_{1 \le i \le N} \delta_i(t) \times a_{ij} \times b_{ijo_t}$

Viterbi Recursion



o_1 o_t o_{t+1}

$$\delta_j(t+1) = \max_{1 \le i \le N} \delta_i(t) \times a_{ij} \times b_{ijo_t}$$

สำหรับจำเส้นทาง

$$\psi_j(t+1) = \arg\max_{1 \le i \le N} \delta_i(t) \times a_{ij} \times b_{ijo_t}$$

Viterbi Algorithm

- Initialization: $\delta_j(1) = \pi_j,\ 1 \leq j \leq N$
- Induction:

$$\delta_j(t+1) = \max_{1 \leq i \leq N} \delta_i(t) \times a_{ij} \times b_{ijo_t},\ 1 \leq j \leq N$$

$$\psi_j(t+1) = \arg\max_{1 \leq i \leq N} \delta_i(t) \times a_{ij} \times b_{ijo_t},\ 1 \leq j \leq N$$

- Total:

$$P(X^*) = \max_{1 \leq i \leq N} \delta_i(T+1) \qquad X^*_{T+1} = \arg\max_{1 \leq i \leq N} \delta_i(T+1)$$

- Path readout:

$$X^*_t = \psi_{X^*_{t+1}}(t+1),\ t = T-1, ..., 1$$

ปัญหาที่ 3: Parameter estimation

- ในปัญหาที่ 1 และ 2 เรารู้ตัวแปรต่างๆ ของแบบจำลอง $\mu = (A,B,\Pi)$
- ตามปกติตัวแปรต่างๆ จะถูกประมาณจากข้อมูลที่ผ่านการกำกับ (annotated data) ซึ่งถูกเตรียมไว้
  - การกำกับข้อมูลมักจะยากและใช้เวลานาน
  - ข้อมูลที่กำกับแล้วอาจไม่ตรงกับข้อมูลที่เรากำลังทำ
- ในปัญหาที่ 3 เราจะหาตัวแปรที่ดีที่สุดจากลำดับการสังเกต O เท่านั้น นั่นคือ เรากำลังหา μ^* ที่ซึ่ง

$$\mu^* = \arg\max_{\mu} P(O|\mu)$$

ปัญหาที่ 3: Parameter estimation

- แต่ในความเป็นจริง เราไม่สามารถหาค่าตัวแปรที่ดีที่สุดได้ (global maximum)
- แต่เราสามารถหาค่าตัวแปรที่ทำให้แบบจำลองดีขึ้นกว่าเดิมได้ (local maximum)
- นั่นคือ จากแบบจำลองเดิม μ เราสามารถหาแบบจำลองใหม่ μ^* ที่ทำให้ $P(O|\mu^*) \geq P(O|\mu)$

Baum-Welch algorithm

- หลักการคือการประมาณค่าตัวแปรใหม่โดยใช้วิธี hill-climbing (local search)
- เริ่มจากกำหนดค่าเริ่มต้นให้กับตัวแปรในแบบจำลอง μ จากนั้น Baum-Welch algorithm จะหาค่าตัวแปรใหม่ μ^* เพื่อทำให้ความน่าจะเป็นสำหรับลำดับการสังเกต O มีค่ามากเพิ่มขึ้น
- การคำนวณจากทำตามจำนวนรอบที่ถูกกำหนดไว้ หรือ ทำจนกระทั่งแบบจำลองใหม่ไม่ดีกว่าเดิม

Parameter Re-estimation

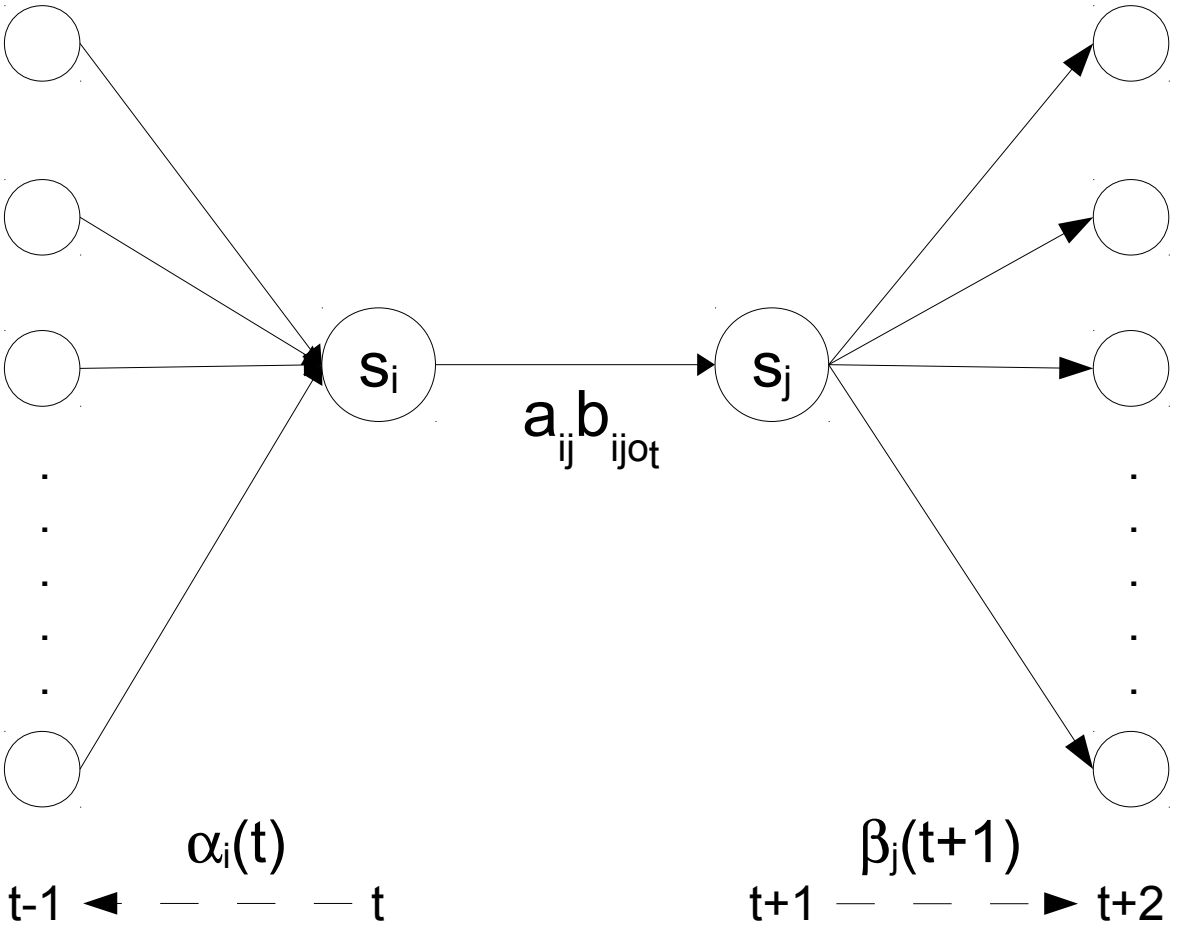
- ตัวแปร 3 ตัวที่ต้องประมาณค่าใหม่คือ
  - Initial state probabilities: Π
  - Transition probabilities: A
  - Symbol emission probabilities: B

Expected number of transitions

- เราสามารถหาความน่าจะเป็นทีผ่านกึงใดๆ ณ เวลา t เมือให้ลำดับการสังเกต O ได้

$$
\begin{aligned}
\xi_t(i,j) &= P(X_t = i, X_{t+1} = j | O, \mu) \\
&= \frac{P(X_t = i, X_{t+1} = j, O|\mu)}{P(O|\mu)} \\
&= \frac{\alpha_i(t) a_{ij} b_{ijo_t} \beta_j(t+1)}{\sum_{m=1}^{N} \alpha_m(t)\beta_m(t)} \\
&= \frac{\alpha_i(t) a_{ij} b_{ijo_t} \beta_j(t+1)}{\sum_{m=1}^{N} \sum_{n=1}^{N} \alpha_m(t) a_{mn} b_{mno_t} \beta_m(t+1)}
\end{aligned}
$$

Expected number of transitions

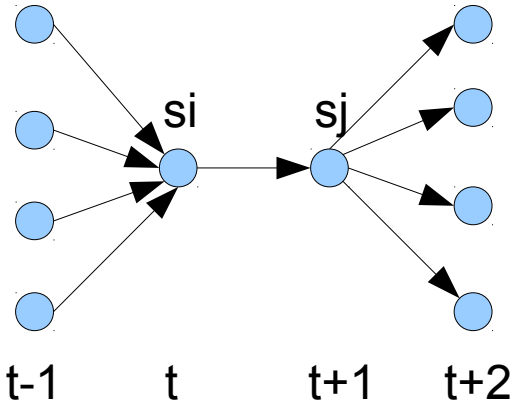


$$\xi_t(i,j) = \frac{\alpha_i(t)a_{ij}b_{ijo_t}\beta_j(t+1)}{\sum_{m=1}^{N}\sum_{n=1}^{N}\alpha_m(t)a_{mn}b_{mno_t}\beta_m(t+1)}$$

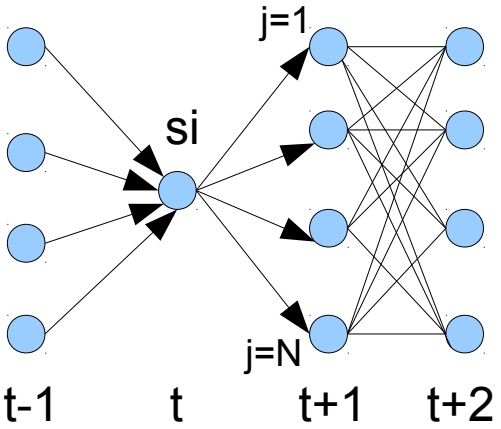
Expected number of transitions

- กำหนดให้ผลรวมของ $\xi_t(i,j)$ สำหรับทุกๆ j เป็น

$$\gamma_i(t) = \sum_{j=1}^{N} \xi_t(i,j)$$



$\xi_t(i,j)$



$\gamma_i(t)$

Expected number of transitions

- ดังนั้นหากเราหาผลรวมของ $\xi_t(i,j)$ และ $\gamma_i(t)$สำหรับทุก t เราจะได้

$$\sum_{t=1}^{T} \xi_t(i,j) = \text{expected number of transitions from state } i \text{ to } j \text{ in } O$$

$$\sum_{t=1}^{T} \gamma_i(t) = \quad \text{expected number of transitions from state } i \text{ in } O$$

Re-estimating Transition Probabilities

- เราสามารถประมาณค่า transition probabilities (A) ได้จาก

$$a_{ij}^{*} = \frac{\text{expected number of transitions from state } i \text{ to } j}{\text{expected number of transitions from state } i}$$

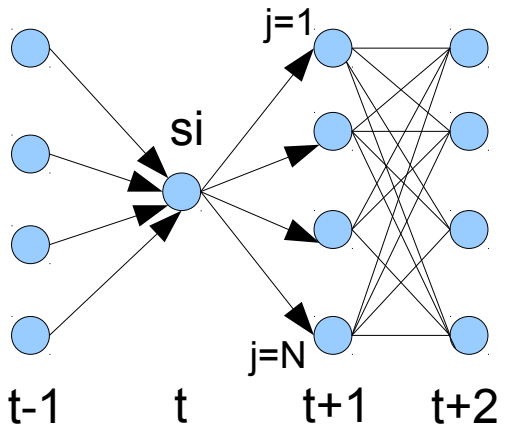
$$a_{ij}^{*} = \frac{\sum_{t=1}^{T} \xi_t(i,j)}{\sum_{t=1}^{T} \gamma_i(t)}$$

Re-estimating Transition Probabilities

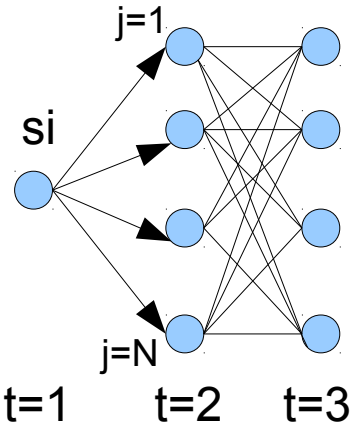
- เราสามารถประมาณค่า initial state probabilities ได้จาก

$$\pi_i^* = \text{expected frequency in state } i \text{ at time } t = 1$$

$$\pi_i^* = \gamma_i(1)$$



$\gamma_i(t)$



$\gamma_i(1)$

Re-estimating Transition Probabilities

- เราสามารถประมาณค่า emission probabilities ได้จาก

$$b^*_{ijk} = \frac{\text{expected number of transitions from } i \text{ to } j \text{ with k observed}}{\text{expected number of transitions from } i \text{ to } j}$$

$$b^*_{ijk} = \frac{\sum_{\{t:o_t=k,1\leq t\leq T\}} \xi_t(i,j)}{\sum_{t=1}^{T} \xi_t(i,j)}$$

การปรับปรุงแบบจำลอง

- จากแบบจำลองเดิม $\mu = (A,B,\Pi)$ เราสามารถหาแบบจำลองใหม่ $\mu^* = (A^*,B^*,\Pi^*)$ ได้จากสมการเหล่านี้

$$a_{ij}^* = \frac{\sum_{t=1}^{T} \xi_t(i,j)}{\sum_{t=1}^{T} \gamma_i(t)} \qquad b_{ijk}^* = \frac{\sum_{\{t:o_t=k,1\le t\le T\}} \xi_t(i,j)}{\sum_{t=1}^{T} \xi_t(i,j)} \qquad \pi_i^* = \gamma_i(1)$$

Expectation Maximization

- Baum-Welch algorithm เป็นตัวอย่างของ EM algorithm
- EM algorithm ประกอบด้วย 2 ขั้นตอนคือ
  - The E step: หาค่า expected ของแบบจำลองเดิม
  - The M step: นำค่า expected ไปคำนวณหาตัวแปรที่ทำให้ความน่าจะเป็นของแบบจำลองใหม่มีค่าสูงสุด

เอกสารอ้างอิง

- Manning & Schutz (2000), Foundation of Statistical NLP, The MIT Press, London.
- http://www.cis.upenn.edu/~cis530/slides-2009/530-Parts%20of%20Speech%20II%20-%202009.pdf
- http://www.comp.leeds.ac.uk/roger/HiddenMarkovModels/html_dev/main.html