제 **1**회 컴퓨터비전 및 패턴인식 겨울학교**, 2006.2.1~3, KAIST**

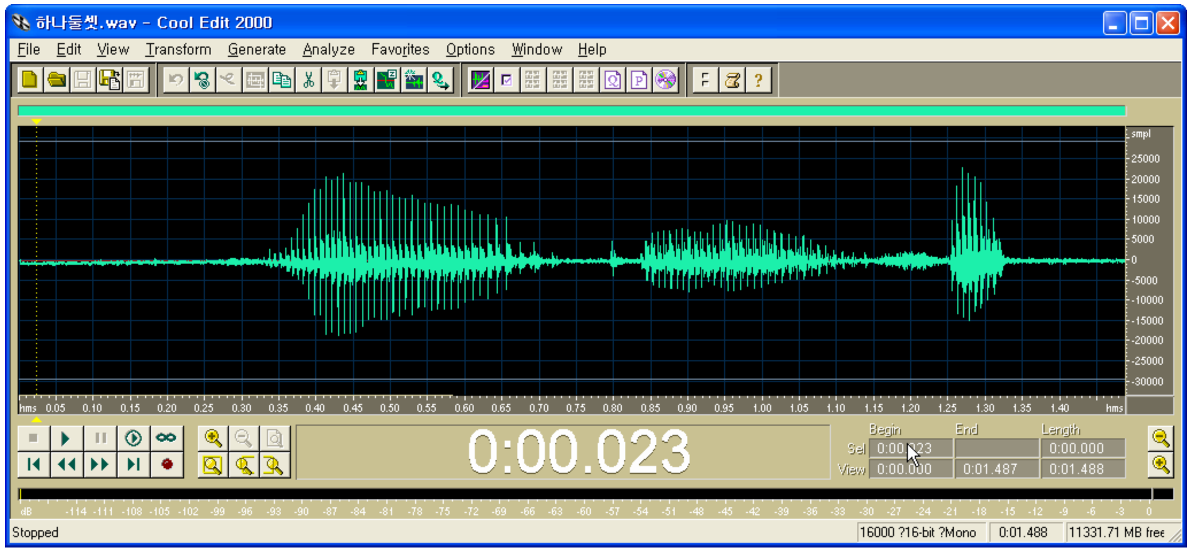
A Tutorial on Hidden Markov Models

2006년 2월 2일

하진영
jyha@kangwon.ac.kr
강원대학교 컴퓨터학부

Sequential Data

- Examples

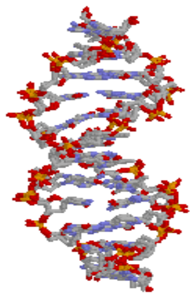

하나둘셋.wav - Cool Edit 2000
File Edit View Transform Generate Analyze Favorites Options Window Help
0:00.023
Begin End Length
Sel 0:00.023 0:00.000
View 0:00.000 0:01.487 0:01.488
Stopped
16000 ?16-bit ?Mono
0:01.488
11331.71 MB free

Speech data
("하나 둘 셋")

한글과 나라사랑
the land of morning calm
國民教育憲章

Handwriting data

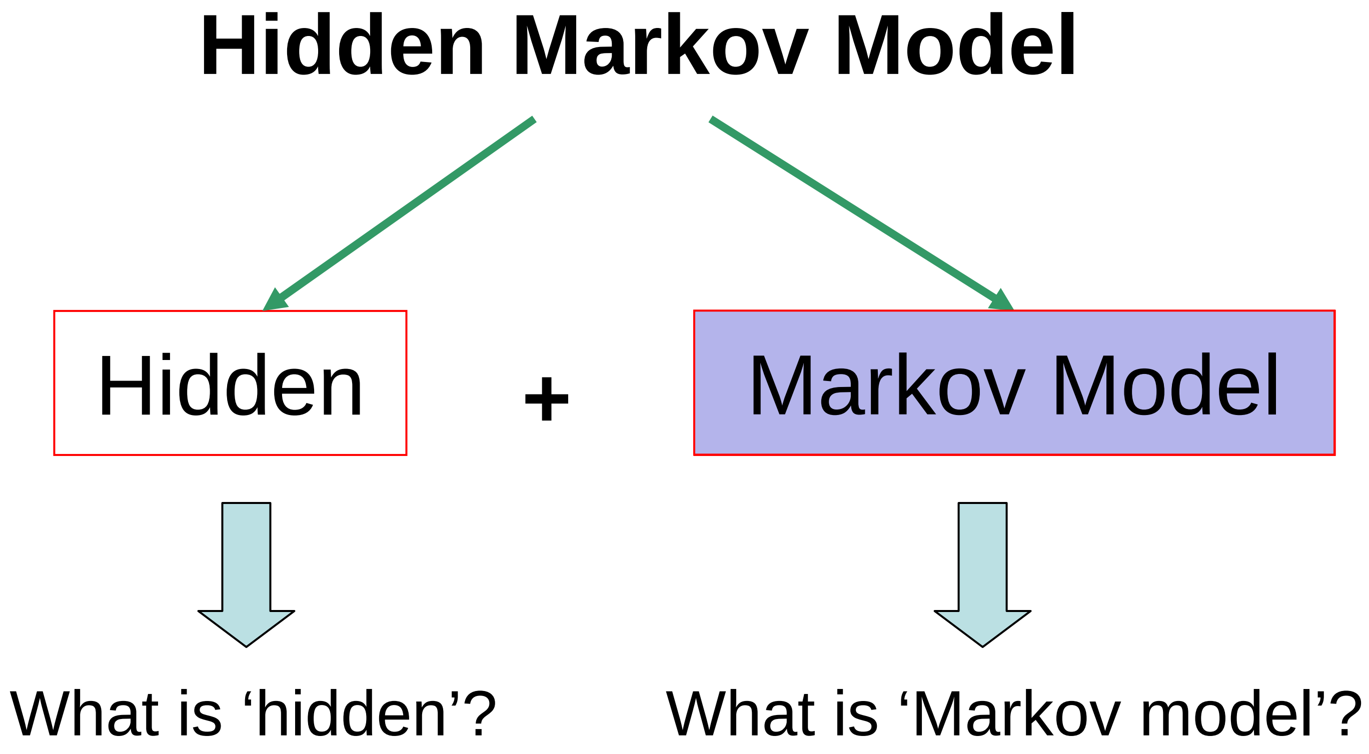
DNA



```
AAAAGAAAAGGTTAGAAAGATGAGAGATGATAAAGGGTCCATTTG
AGGTTAGGTAATATGGTTTGGTATCCCTGTAGTTAAAAGTTTTTG
TCTTATTTTAGAATACTGTGACTATTTCTTTAGTATTAATTTTTC
CTTCTGTTTTCCTCATCTAGGGAACCCCAAGAGCATCCAATAGAA
GCTGTGCAATTATGTAAAATTTTCAACTGTCTTCCTCAAAATAAA
GAAGTATGGTAATCTTTACCTGTATACAGTGCAGAGCCTTCTCAG
AAGCACAGAATATTTTTATATTTCCTTTATGTGAATTTTTAAGCT
GCAAATCTGATGGCCTTAATTTCCTTTTTGACACTGAAAGTTTTG
TAAAAGAAATCATGTCCATACACTTTGTTGCAAGATGTGAATTAT
TGACACTGAACTTAATAACTGTGTACTGTTCGGAAGGGGTTCCTC
AAATTTTTTGACTTTTTTTGTATGTGTGTTTTTTCTTTTTTTTTA
AGTTCTTATGAGGAGGGAGGGTAAATAAACCACTGTGCGTCTTGG
TGTAATTTGAAGATTGCCCCATCTAGACTAGCAATCTCTTCATTA
TTCTCTGCTATATATAAAACGGTGCTGTGAGGGAGGGGAAAAGCA
TTTTTCAATATATTGAACTTTTGTACTGAATTTTTTTGTAATAAG
CAATCAAGGTTATAATTTTTTTTAAAATAGAAATTTTGTAAGAAG
GCAATATTAACCTAATCACCATGTAAGCACTCTGGATGATGGATT
CCACAAAACTTGGTTTTATGGTTACTTCTTCTCTTAGATTCTTAA
TTCATGAGGAGGGTGGGGAGGGAGGTGGAGGGAGGGAAGGGTTT
CTCTATTAAAATGCATTCGTTGTGTTTTTTAAGATAGTGTAACTT
GCTAAATTTCTTATGTGACATTAACAAATAAAAAAGCTCTTTTAA
TATTAGATAA
```

18

What's HMM?

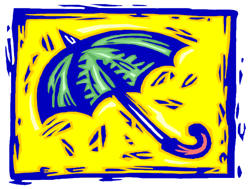
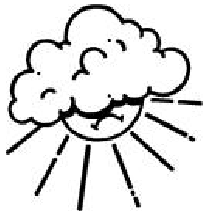
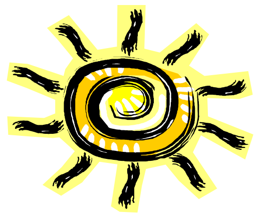

Hidden Markov Model
Hidden
+
Markov Model
What is 'hidden'?
What is 'Markov model'?

Markov Model

- Scenario
- Graphical representation
- Definition
- Sequence probability
- State probability

Markov Model: Scenario

- Classify a weather into three states
  - State 1: rain or snow
  - State 2: cloudy
  - State 3: sunny

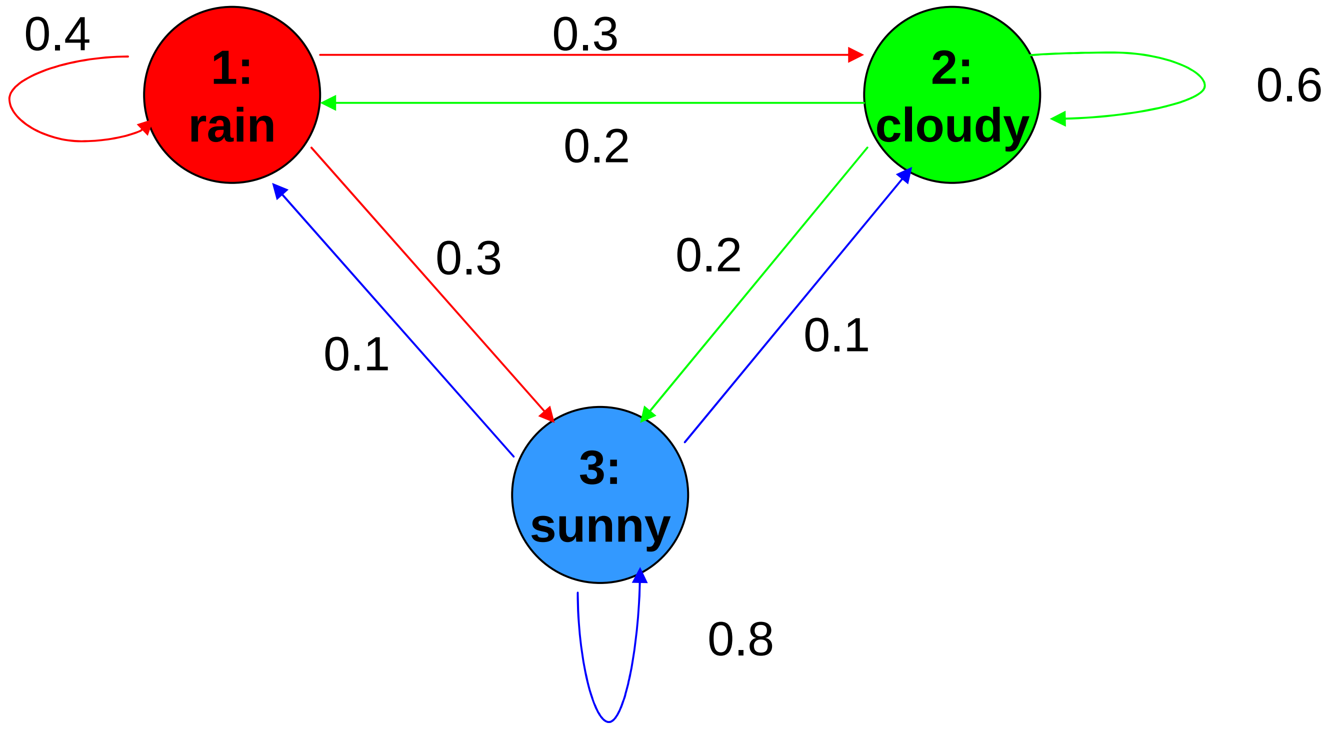
- By carefully examining the weather of some city for a long time, we found following weather change pattern

		Tomorrow		
		Rain/snow	Cloudy	Sunny
Today	Rain/Snow	0.4	0.3	0.3
	Cloudy	0.2	0.6	0.2
	Sunny	0.1	0.1	0.8

Assumption: tomorrow weather depends only on today's weather!

Markov Model: Graphical Representation

- Visual illustration with diagram



- Each state corresponds to one observation
- Sum of outgoing edge weights is one

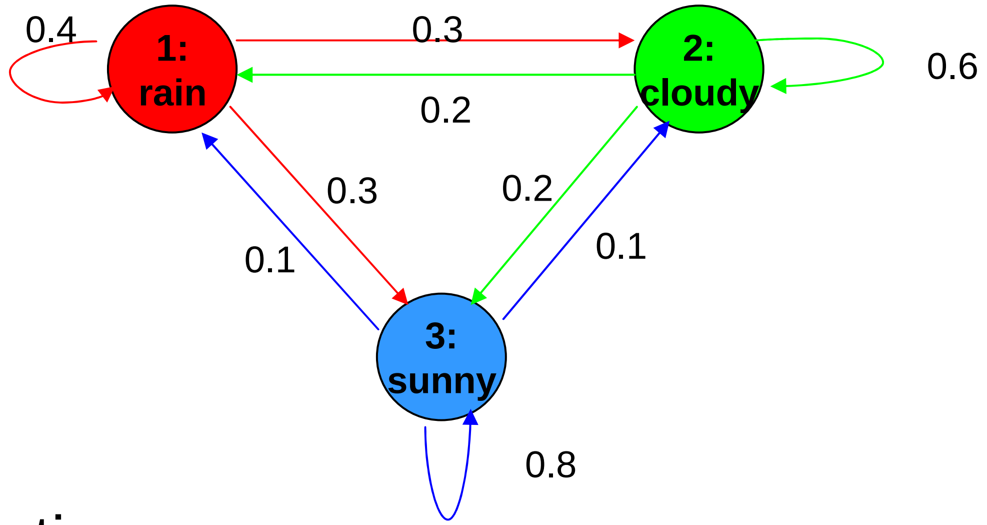
Markov Model: Definition

- Observable states

$$\{1,\ 2,\ \cdots,\ N\}$$

- Observed sequence

$$q_1, q_2, \cdots, q_T$$



- 1st order Markov assumption

$$P(q_t = j \mid q_{t-1} = i, q_{t-2} = k, \cdots) = P(q_t = j \mid q_{t-1} = i)$$

$$\begin{tikzcd} q_1 \arrow[rrrr, bend right, ""] & q_2 \arrow[rrr, bend right] & \cdots & q_{t-1} \arrow[r] & q_t & \Longrightarrow & q_{t-1} \arrow[r] & q_t \end{tikzcd}$$

Bayesian network representation

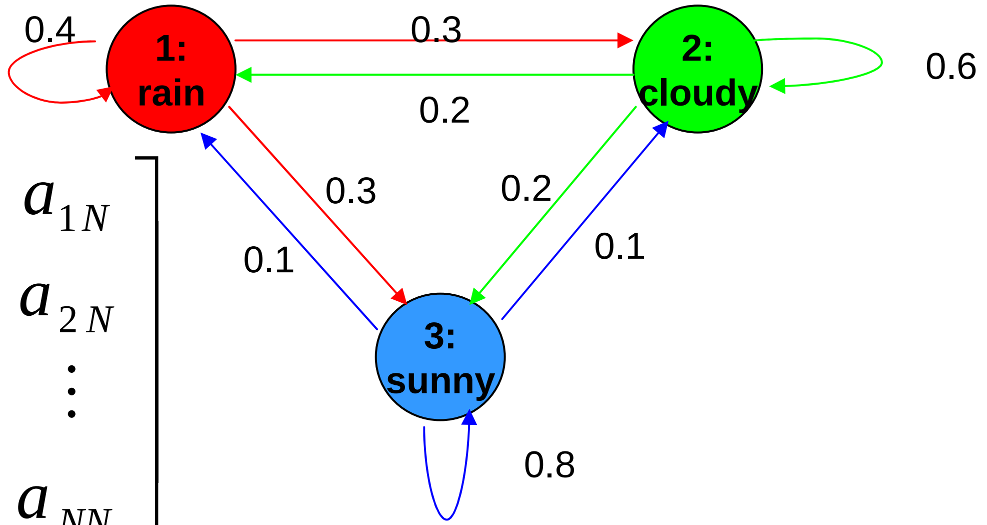
- Stationary

$$P(q_t = j \mid q_{t-1} = i) = P(q_{t+l} = j \mid q_{t+l-1} = i)$$

Markov Model: Definition (Cont.)

- State transition matrix

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1N} \\ a_{21} & a_{22} & \cdots & a_{2N} \\ \vdots & \vdots & \vdots & \vdots \\ a_{N1} & a_{NN} & \cdots & a_{NN} \end{bmatrix}$$



  - Where

$$a_{ij} = P(q_t = j \mid q_{t-1} = i), \qquad 1 \le i, j \le N$$

  - With constraints

$$a_{ij} \ge 0, \qquad \sum_{j=1}^{N} a_{ij} = 1$$

- Initial state probability

$$\pi_i = P(q_1 = i), \qquad 1 \le i \le N$$

Markov Model: Sequence Prob.

- Conditional probability

$$P(A,B) = P(A \mid B)P(B)$$

- Sequence probability of Markov model

Chain rule

$$\Downarrow$$

$$\begin{aligned}
&P(q_1, q_2, \cdots, q_T) \\
&= P(q_1)P(q_2 \mid q_1)\cdots P(q_{T-1} \mid q_1, \cdots, q_{T-2})P(q_T \mid q_1, \cdots, q_{T-1}) \\
&= P(q_1)P(q_2 \mid q_1)\cdots P(q_{T-1} \mid q_{T-2})P(q_T \mid q_{T-1})
\end{aligned}$$

$$\Uparrow$$

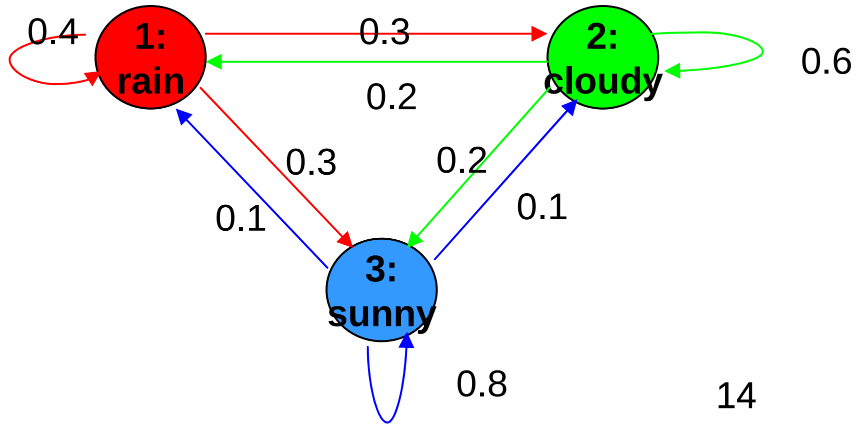
1st order Markov assumption

Markov Model: Sequence Prob. (Cont.)

- Question: What is the probability that the weather for the next 7 days will be “sun-sun-rain-rain-sun-cloudy-sun” when today is sunny?

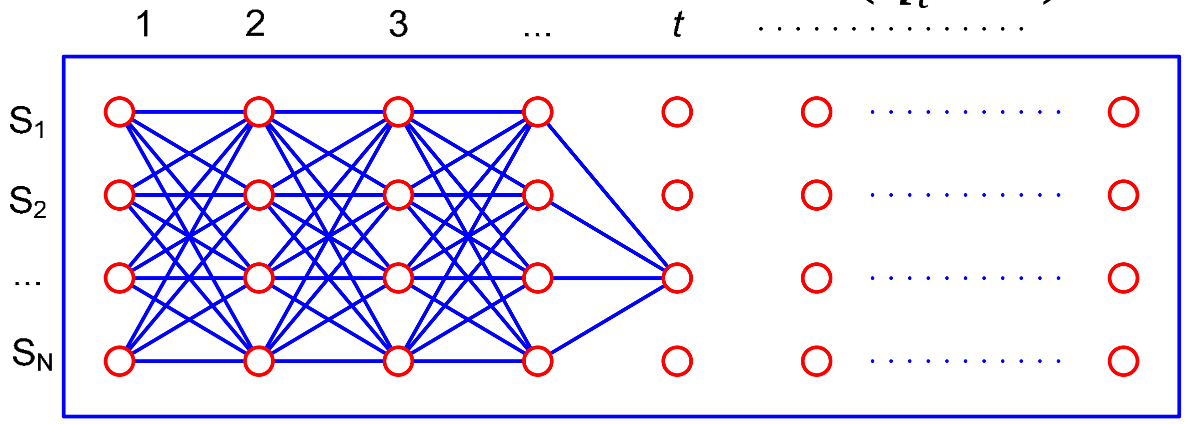
$$S_1 : rain, \quad S_2 : cloudy, \quad S_3 : sunny$$

$$\begin{aligned}
P(O \mid \text{model}) &= P(S_3, S_3, S_3, S_1, S_1, S_3, S_2, S_3 \mid \text{model}) \\
&= P(S_3) \cdot P(S_3 \mid S_3) \cdot P(S_3 \mid S_3) \cdot P(S_1 \mid S_3) \\
&\quad \cdot P(S_1 \mid S_1) P(S_3 \mid S_1) P(S_2 \mid S_3) P(S_3 \mid S_2) \\
&= \pi_3 \cdot a_{33} \cdot a_{33} \cdot a_{31} \cdot a_{11} \cdot a_{13} \cdot a_{32} \cdot a_{23} \\
&= 1 \cdot (0.8)(0.8)(0.1)(0.4)(0.3)(0.1)(0.2) \\
&= 1.536 \times 10^{-4}
\end{aligned}$$



Markov Model: State Probability

- State probability at time *t* : $P(q_t = i)$



- Simple but slow algorithm:
  - Probability of a path that ends to state *i* at time *t*:

$$Q_t(i) = (q_1, q_2, \cdots, q_t = i)$$

$$P(Q_t(i)) = \pi_{q_1} \prod_{k=2}^{t} P(q_k \mid q_{k-1})$$

  - Summation of probabilities of all the paths that ends to *i* at *t*

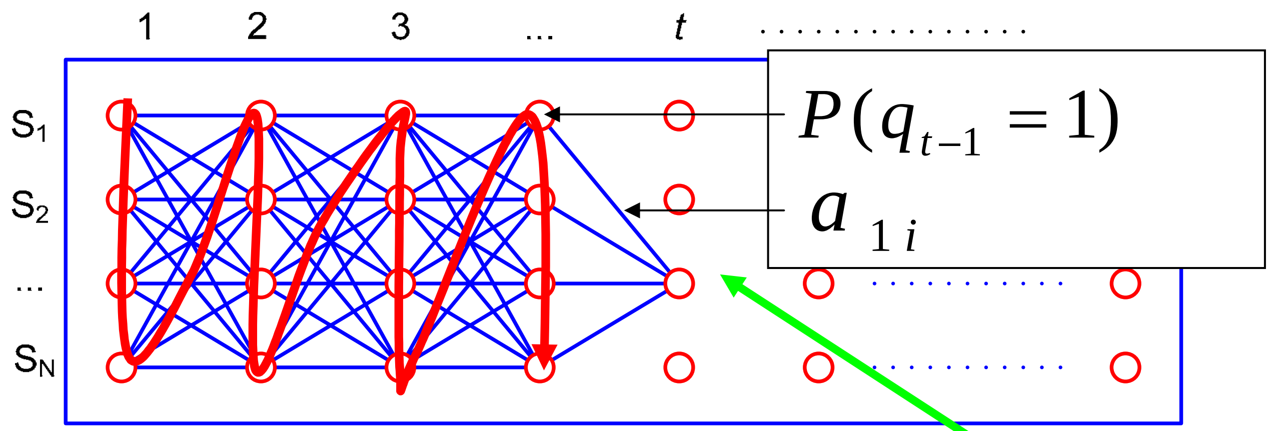
$$P(q_t = i) = \sum_{all \;\; Q_t(i)'s} P(Q_t(i))$$

Exponential time complexity:

$O(N^t)$

Markov Model: State Prob. (Cont.)

- State probability at time *t* : $P(q_t = i)$



Each node stores the sum of probabilities of partial paths

- Efficient algorithm (Lattice algorithm)
  - Recursive path probability calculation

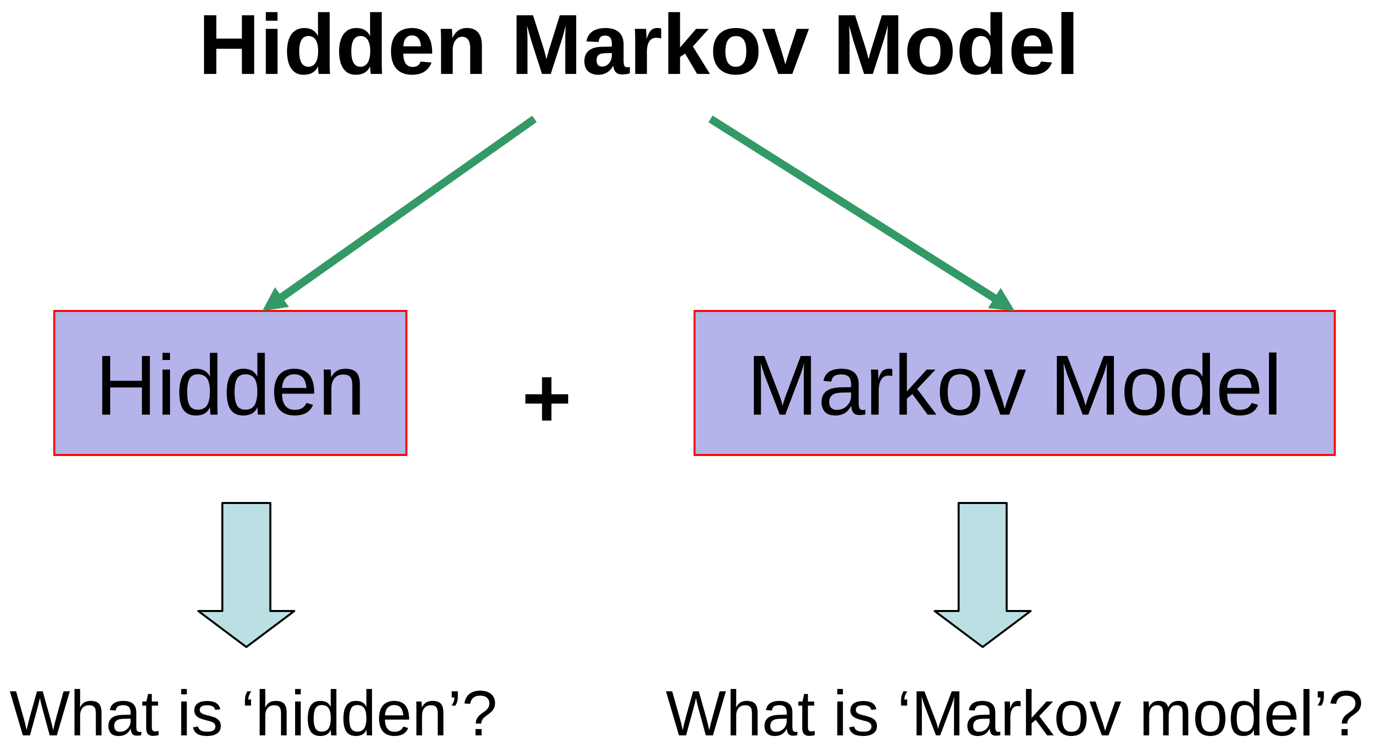
$$P(q_t = i) = \sum_{j=1}^{N} P(q_{t-1} = j, q_t = i)$$

$$= \sum_{j=1}^{N} P(q_{t-1} = j) P(q_t = i \mid q_{t-1} = j)$$

$$= \sum_{j=1}^{N} P(q_{t-1} = j) \cdot a_{ji}$$

Time complexity: $O(N^2 t)$

What’s HMM?


Hidden Markov Model
Hidden
+
Markov Model
What is ‘hidden’?
What is ‘Markov model’?

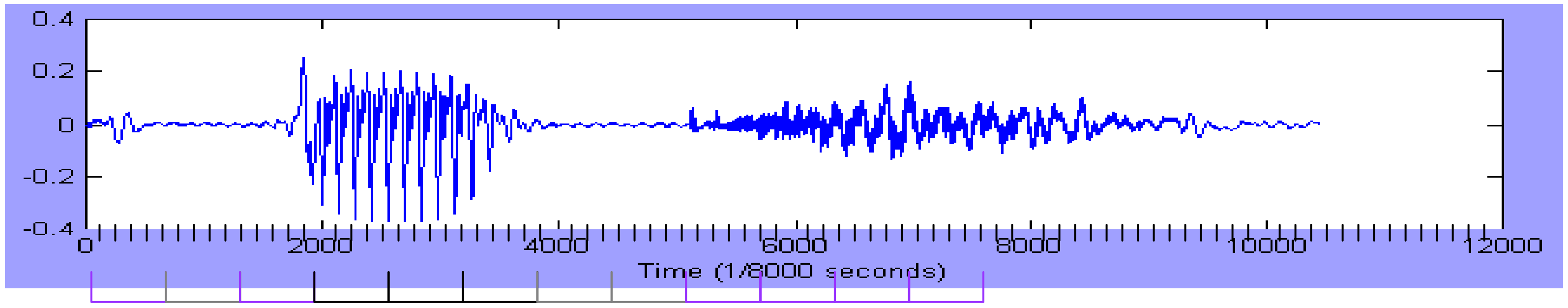
Hidden Markov Model

- Example
- Generation process
- Definition
- Model evaluation algorithm
- Path decoding algorithm
- Training algorithm

Time Series Example

- Representation
  - $\mathbf{X} = \mathbf{x}_1\ \mathbf{x}_2\ \mathbf{x}_3\ \mathbf{x}_4\ \mathbf{x}_5 \ldots \mathbf{x}_{T-1}\ \mathbf{x}_T$

    = s ϕ p iy iy iy ϕ ϕ ch ch ch ch



Analysis Methods

- Probability-based analysis?

$$P(\mathrm{s}\ \phi\ \mathrm{p}\ \mathrm{iy}\ \mathrm{iy}\ \mathrm{iy}\ \phi\ \phi\ \mathrm{ch}\ \mathrm{ch}\ \mathrm{ch}\ \mathrm{ch}) = ?$$

- Method I

$$\boxed{P(\mathrm{s})P(\phi)^3 P(\mathrm{p})P(\mathrm{iy})^3 P(\mathrm{ch})^4}$$

  – Observations are independent; no time/order

  – A poor model for temporal structure

    • Model size = $|V| = N$

Analysis methods

- Method II

$$P(\mathrm{s})P(\mathrm{s}\,|\,\mathrm{s})P(\phi\,|\,\mathrm{s})P(\mathrm{p}\,|\,\phi)P(\mathrm{iy}\,|\,\mathrm{p})P(\mathrm{iy}\,|\,\mathrm{iy})^2$$
$$\times P(\phi\,|\,\mathrm{iy})P(\phi\,|\,\phi)P(\mathrm{ch}\,|\,\phi)P(\mathrm{ch}\,|\,\mathrm{ch})^{2`}$$

– A simple model of ordered sequence

- A symbol is dependent only on the immediately preceding:

$$P(x_t \mid x_1 x_2 x_3 \cdots x_{t-1}) = P(x_t \mid x_{t-1})$$

- $|V| \times |V|$ matrix model
- 50×50 – not very bad …
- $10^5 \times 10^5$ – doubly outrageous!!

Another analysis method

- Method III
  - What you see is a clue to <u>what lies behind</u> and is not known *a priori*
    - The source that generated the observation
    - The source evolves and generates characteristic observation sequences

$$q_0 \to q_1 \to q_2 \to \quad \cdots \quad \to q_T$$

$$P(\mathrm{s}, q_1)P(\mathrm{s}, q_2 \mid q_1)P(\phi, q_3 \mid q_2)\cdots P(\mathrm{ch}, q_T \mid q_{T-1}) = \prod_t P(x_t, q_t \mid q_{t-1})$$

$$\sum_Q P(\mathrm{s}, q_1)P(\mathrm{s}, q_2 \mid q_1)P(\phi, q_3 \mid q_2)\cdots P(\mathrm{ch}, q_T \mid q_{T-1}) = \sum_Q \prod_t P(x_t, q_t \mid q_{t-1})$$

The Auxiliary Variable

$$q_t \in S = \{1, \ \ldots, \ N\}$$

- N is also conjectured
- $\{q_t : t \geq 0\}$ is conjectured, not visible
  - is $Q = q_1 q_2 \cdots q_T$
  - is Markovian

$$P(q_1 q_2 \cdots q_T) = P(q_1) P(q_2 \mid q_1) \cdots P(q_T \mid q_{T-1})$$

  - “Markov chain”

Summary of the Concept

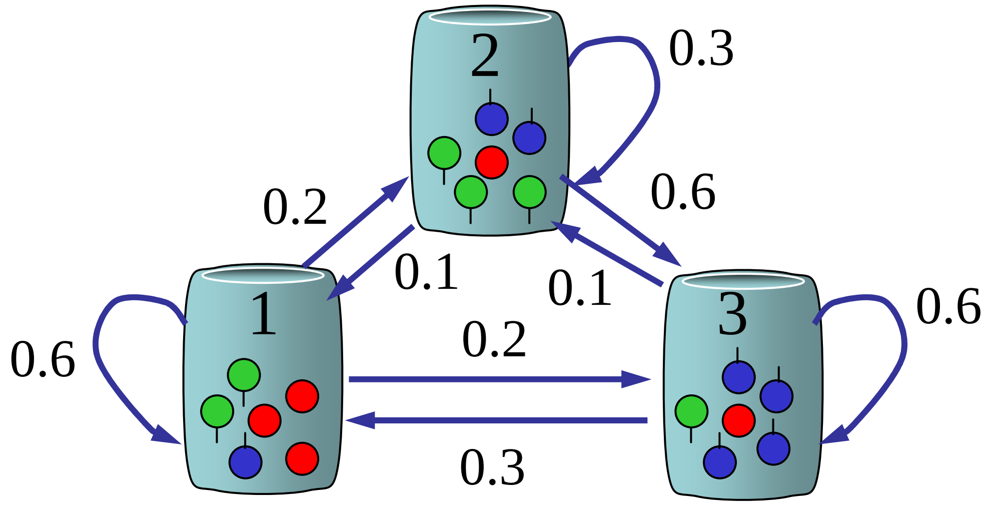
$$
\begin{aligned}
P(X) &= \sum_{Q} P(X, Q) \\
&= \sum_{Q} P(Q) P(X \mid Q) \\
&= \sum_{Q} P(q_1 q_2 \cdots q_T) P(x_1 x_2 \cdots x_T \mid q_1 q_2 \cdots q_T) \\
&= \sum_{Q} \underbrace{\prod_{t=1}^{T} P(q_t \mid q_{t-1})}_{\text{Markov chain process}} \underbrace{\prod_{t=1}^{T} p(x_t \mid q_t)}_{\text{Output process}}
\end{aligned}
$$

Hidden Markov Model

- is a doubly stochastic process
  - stochastic chain process : $\{ q(t) \}$
  - output process : $\{ f(x|q) \}$

- is also called as
  - *Hidden Markov chain*
  - *Probabilistic function of Markov chain*

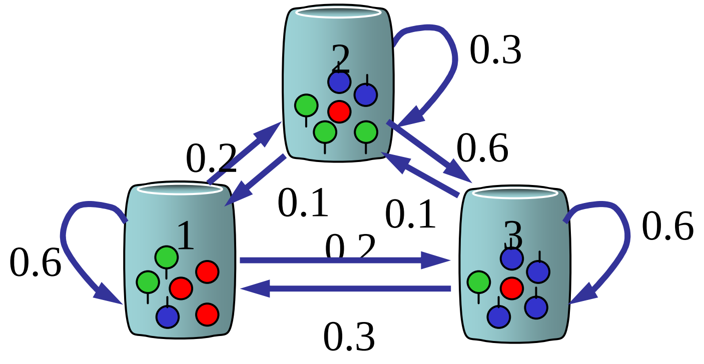
Hidden Markov Model: Example

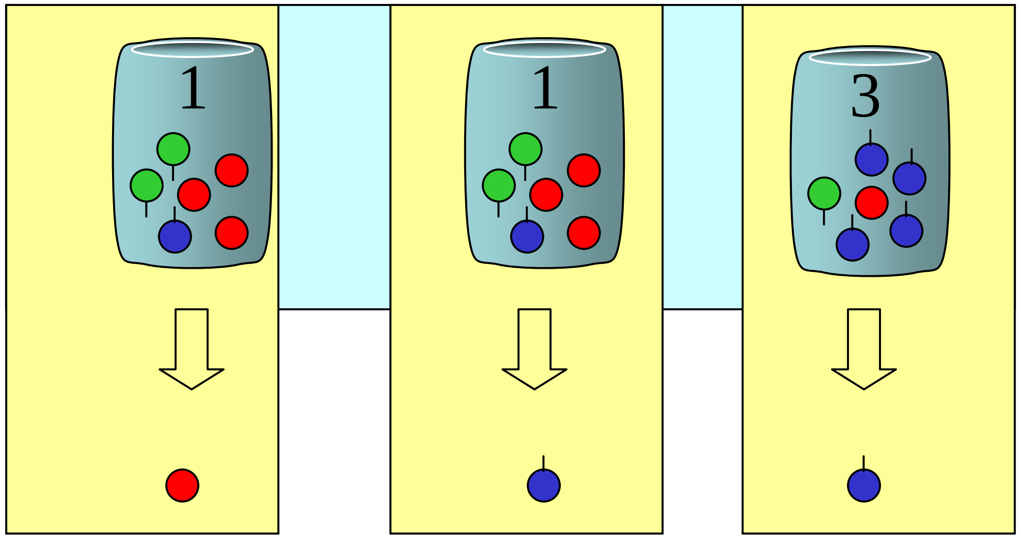


- *N* pots containing color balls
- *M* distinct colors
- Each pot contains different number of color balls

HMM: Generation Process

- Sequence generating algorithm
  - Step 1: Pick initial pot according to some random process
  - Step 2: Randomly pick a ball from the pot and then replace it
  - Step 3: Select another pot according to a random selection process
  - Step 4: Repeat steps 2 and 3



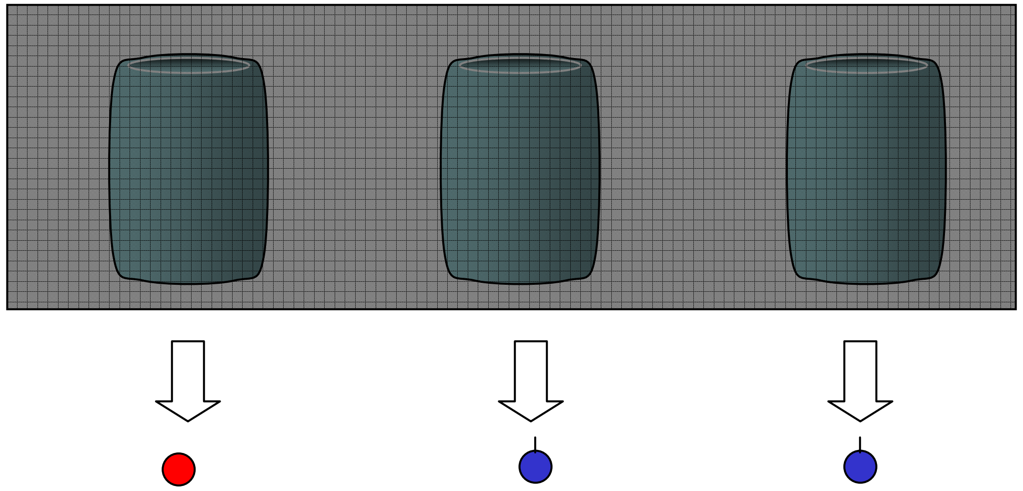


Markov process: $\{q(t)\}$

Output process: $\{f(x|q)\}$

HMM: Hidden Information

- Now, what is hidden?



  - We can just see the chosen balls
  - We can't see which pot is selected at a time
  - So, pot selection (state transition) information is hidden

HMM: Formal Definition

- Notation: $\lambda = (\mathrm{A}, \mathrm{B}, \pi)$

  (1) *N* : Number of states

  (2) *M* : Number of symbols observable in states

$$V = \{v_1, \cdots, v_M\}$$

  (3) *A* : State transition probability distribution

$$A = \{a_{ij}\}, \qquad 1 \le i, j \le N$$

  (4) *B* : Observation symbol probability distribution

$$B = \{b_i(v_k)\}, \qquad 1 \le i \le N, 1 \le j \le M$$

  (5) π : Initial state distribution

$$\pi_i = P(q_1 = i), \qquad 1 \le i \le N$$

Three Problems

1. Model evaluation problem
   - What is the probability of the observation?
   - Forward algorithm

2. Path decoding problem
   - What is the best state sequence for the observation?
   - Viterbi algorithm

3. Model training problem
   - How to estimate the model parameters?
   - Baum-Welch reestimation algorithm

Solution to Model Evaluation Problem

Forward algorithm

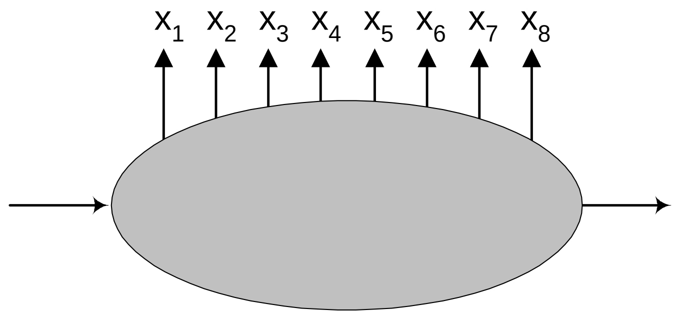
Backward algorithm

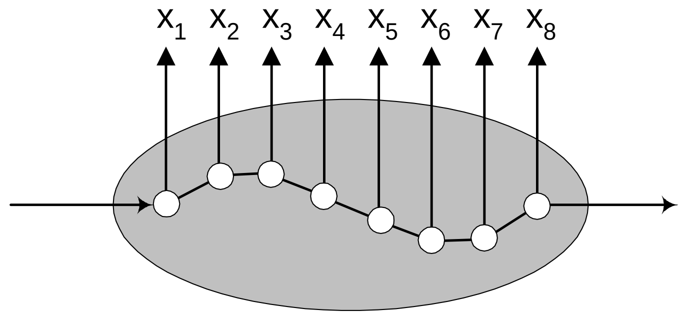
Definition

- Given a model λ
- Observation sequence: $X = x_1, x_2, \cdots, x_T$
- $P(X|\lambda) = ?$

- $P(X \mid \lambda) = \sum_Q P(X, Q \mid \lambda) = \sum_Q P(X \mid Q, \lambda) P(Q \mid \lambda)$

(A path or state sequence: $Q = q_1, \cdots, q_T$)





Solution

- Easy but slow solution: exhaustive enumeration

$$P(X \mid \lambda) = \sum_Q P(X, Q \mid \lambda) = \sum_Q P(X \mid Q, \lambda) P(Q \mid \lambda)$$

$$= \sum_Q b_{q_1}(x_1) b_{q_2}(x_2) \cdots b_{q_T}(x_T) \pi_{q_1} a_{q_1 q_2} a_{q_2 q_3} \cdots a_{q_{T-1} q_T}$$

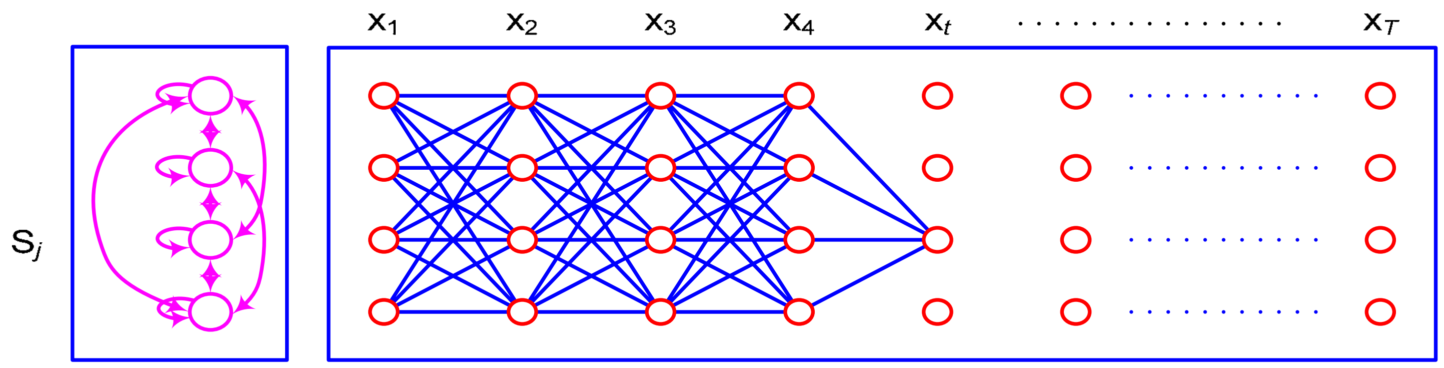
  - Exhaustive enumeration = combinational explosion!

$$O(N^T)$$

- Smart solution exists?
  - Yes!
  - Dynamic Programming technique
  - Lattice structure based computation
  - Highly efficient -- linear in frame length

Forward Algorithm

- Key idea
  - Span a lattice of *N* states and *T* times
  - Keep the sum of probabilities of all the paths coming to each state *i* at time *t*



- Forward probability

$$\alpha_t(j) = P(x_1 x_2 \ldots x_t, q_t = S_j \mid \lambda)$$
$$= \sum_{Q_t} P(x_1 x_2 \ldots x_t, Q_t = q_1 \ldots q_t \mid \lambda)$$
$$= \sum_{i=1}^{N} \alpha_{t-1}(i) a_{ij} b_j(x_t)$$

Forward Algorithm

- Initialization

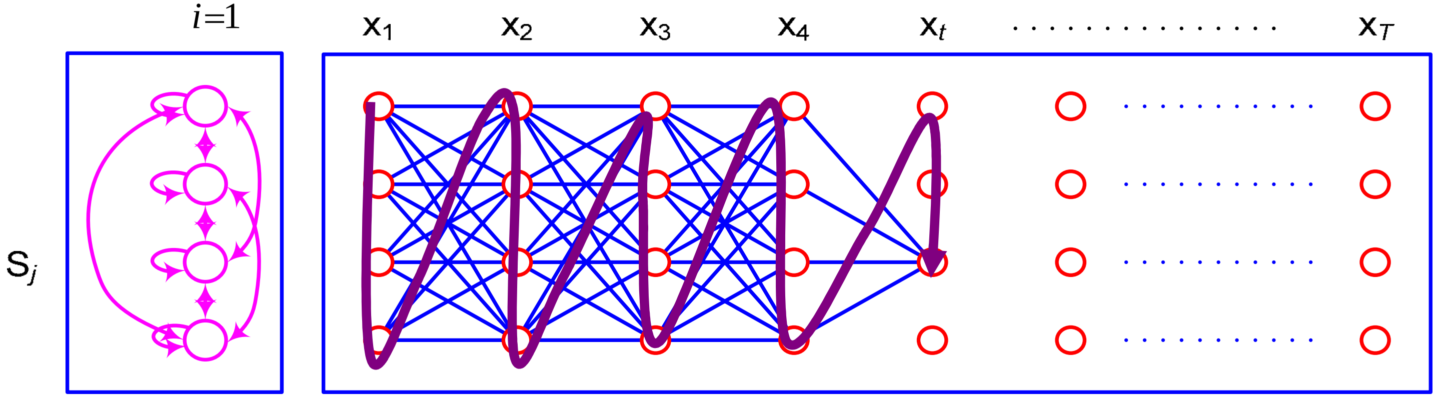
$$\alpha_1(i) = \pi_i b_i(\mathbf{x}_1) \qquad 1 \le i \le N$$

- Induction

$$\alpha_t(j) = \sum_{i=1}^{N} \alpha_{t-1}(i) a_{ij} b_j(\mathbf{x}_t) \qquad 1 \le j \le N,\ t = 2, 3, \cdots, T$$

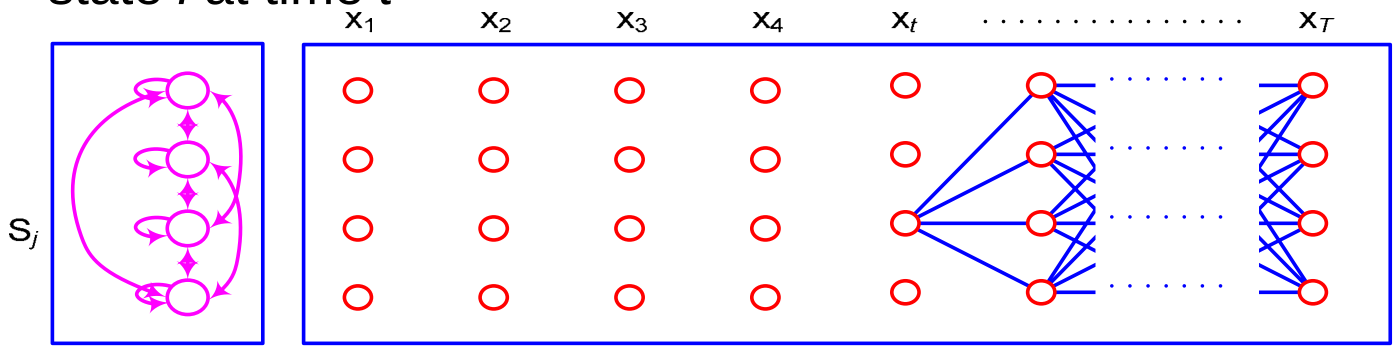
- Termination

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



Backward Algorithm (1)

- Key Idea
  - Span a lattice of *N* states and *T* times
  - Keep the sum of probabilities of all the outgoing paths at each state *i* at time *t*



- Backward probability

$$\beta_t(i) = P(x_{t+1}x_{t+2}...x_T \mid q_t = S_i, \lambda)$$

$$= \sum_{Q_{t+1}} P(x_{t+1}x_{t+2}...x_T, Q_{t+1} = q_{t+1}...q_T \mid q_t = S_i, \lambda)$$

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

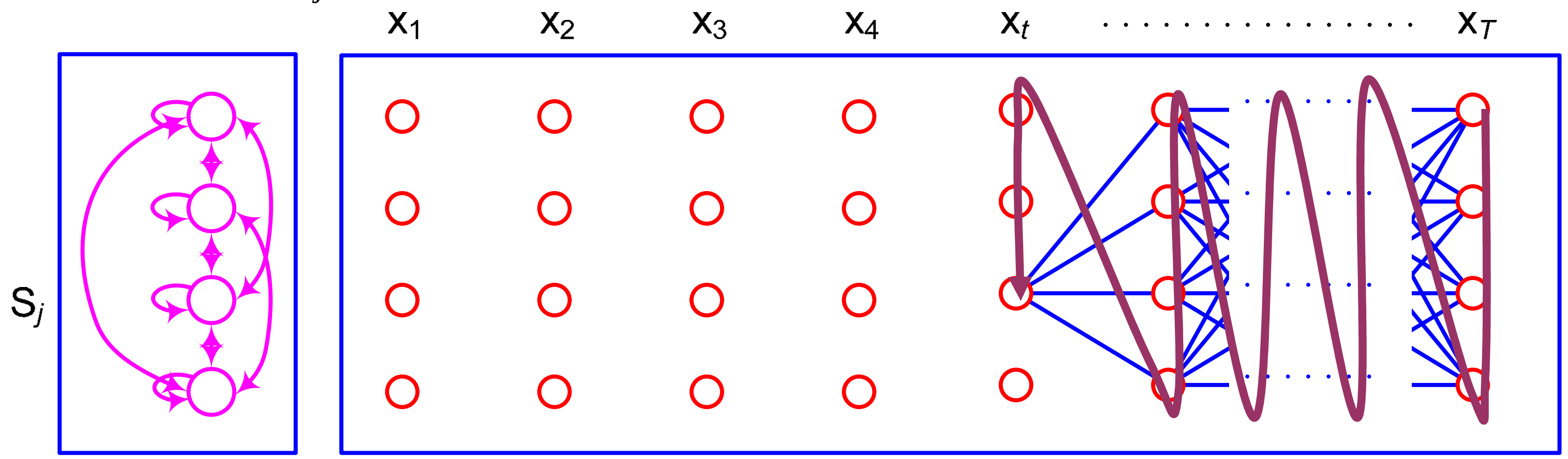
Backward Algorithm (2)

- Initialization

$$\beta_T(i) = 1 \qquad 1 \le i \le N$$

- Induction

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(\mathbf{x}_{t+1}) \beta_{t+1}(j) \qquad 1 \le i \le N, \quad t = T-1, T-2, \cdots, 1$$



Solution to Path Decoding Problem

State sequence

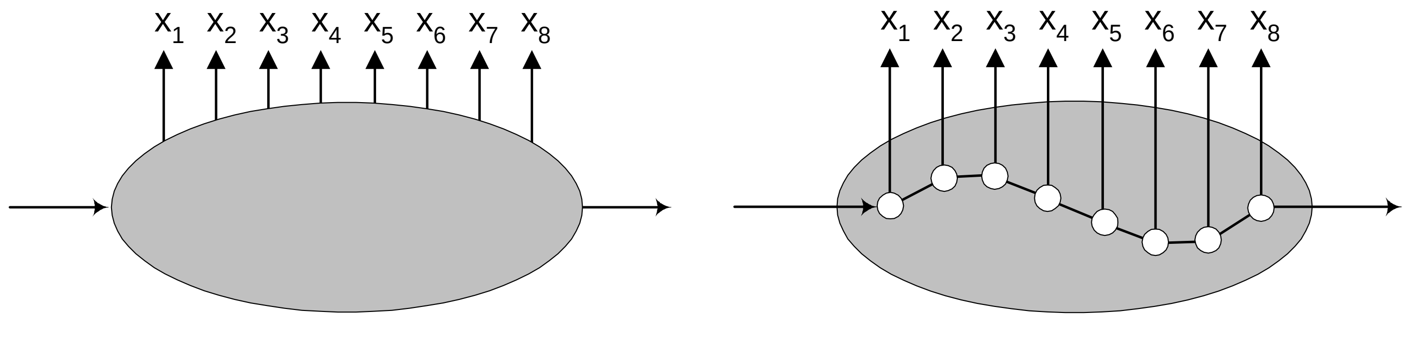
Optimal path

Viterbi algorithm

Sequence segmentation

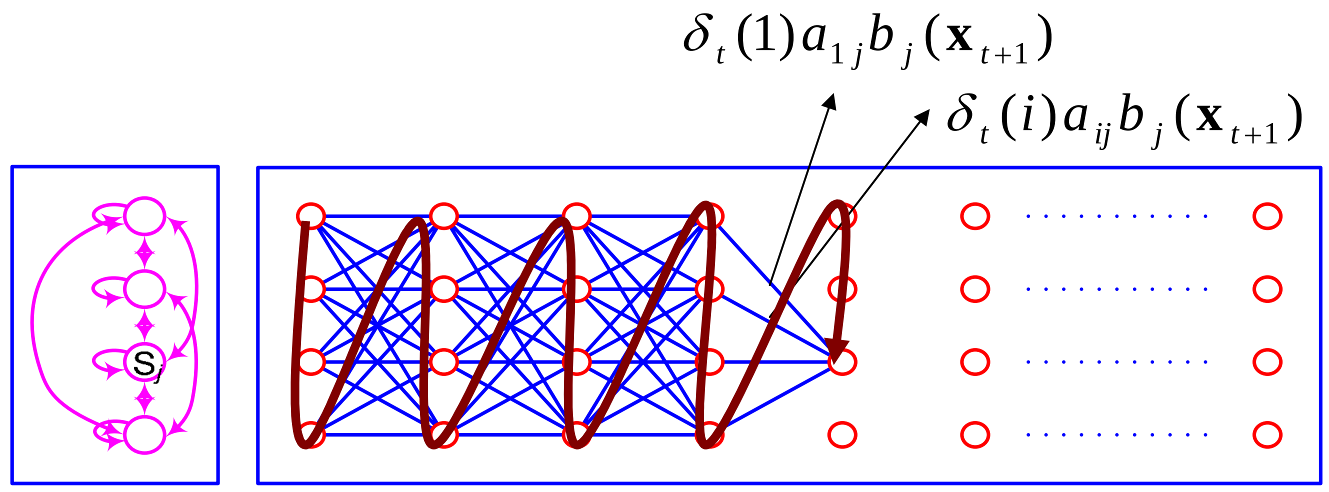
The Most Probable Path

- Given a model λ
- Observation sequence: $X = \mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_T$
- $P(X, Q \mid \lambda)$ = ?
- $Q^* = \arg\max_Q P(X, Q \mid \lambda) = \arg\max_Q P(X \mid Q, \lambda) P(Q \mid \lambda)$
  - (A path or state sequence: $Q = q_1, \cdots, q_T$)



Viterbi Path Idea

- Key idea
  - Span a lattice of *N* states and *T* times
  - Keep the probability and the previous node of the most probable path coming to each state *i* at time *t*
- Recursive path selection
  - Path probability: $\delta_{t+1}(j) = \max_{1 \le i \le N} \delta_t(i) a_{ij} b_j(\mathbf{x}_{t+1})$
  - Path node: $\psi_{t+1}(j) = \arg\max_{1 \le i \le N} \delta_t(i) a_{ij}$


$\delta_t(1) a_{1j} b_j(\mathbf{x}_{t+1})$
$\delta_t(i) a_{ij} b_j(\mathbf{x}_{t+1})$
S_j

Viterbi Algorithm

- Introduction: $\delta_1(i) = \pi_i b_i(\mathbf{x}_1), \quad 1 \le i \le N$

  $\psi_1(i) = 0$

- Recursion: $\delta_{t+1}(j) = \max\limits_{1 \le i \le N} \delta_t(i) a_{ij} b_j(\mathbf{x}_{t+1}), \quad 1 \le t \le T-1$

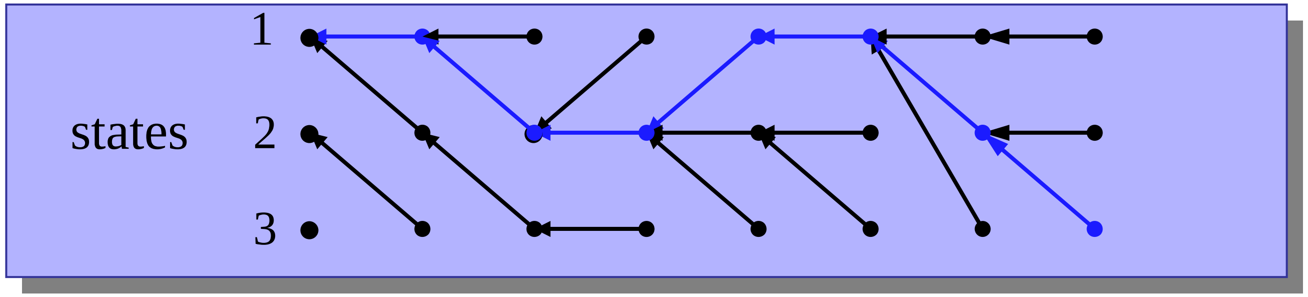
  $1 \le j \le N$

  $\psi_{t+1}(j) = \arg\max\limits_{1 \le i \le N} \delta_t(i) a_{ij}$

- Termination: $P^* = \max\limits_{1 \le i \le N} \delta_T(i)$

  $q_T^* = \arg\max\limits_{1 \le i \le N} \delta_T(i)$

- Path backtracking: $q_t^* = \psi_{t+1}(q_{t+1}^*), \quad t = T-1, \ldots, 1$



Solution to
Model training Problem

HMM training algorithm

Maximum likelihood estimation

Baum-Welch reestimation

HMM Training Algorithm

- Given an observation sequence $X = \mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_T$
- Find the model parameter $\lambda^* = (A, B, \pi)$

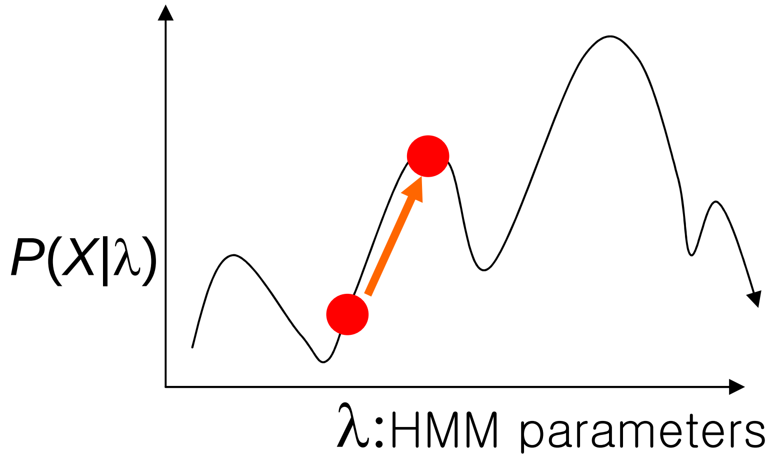
  s.t. $P(X \mid \lambda^*) \geq P(X \mid \lambda)$ for $\forall \lambda$

  - Adapt HMM parameters maximally to training samples
  - Likelihood of a sample

$$P(X \mid \lambda) = \sum_Q P(X \mid Q, \lambda) \boxed{P(Q \mid \lambda)}$$

State transition is hidden!

- NO analytical solution
- *Baum-Welch* reestimation (EM)
  - iterative procedures that locally maximizes $P(X|\lambda)$
  - convergence proven
  - MLE statistic estimation



EM Algorithm for Training

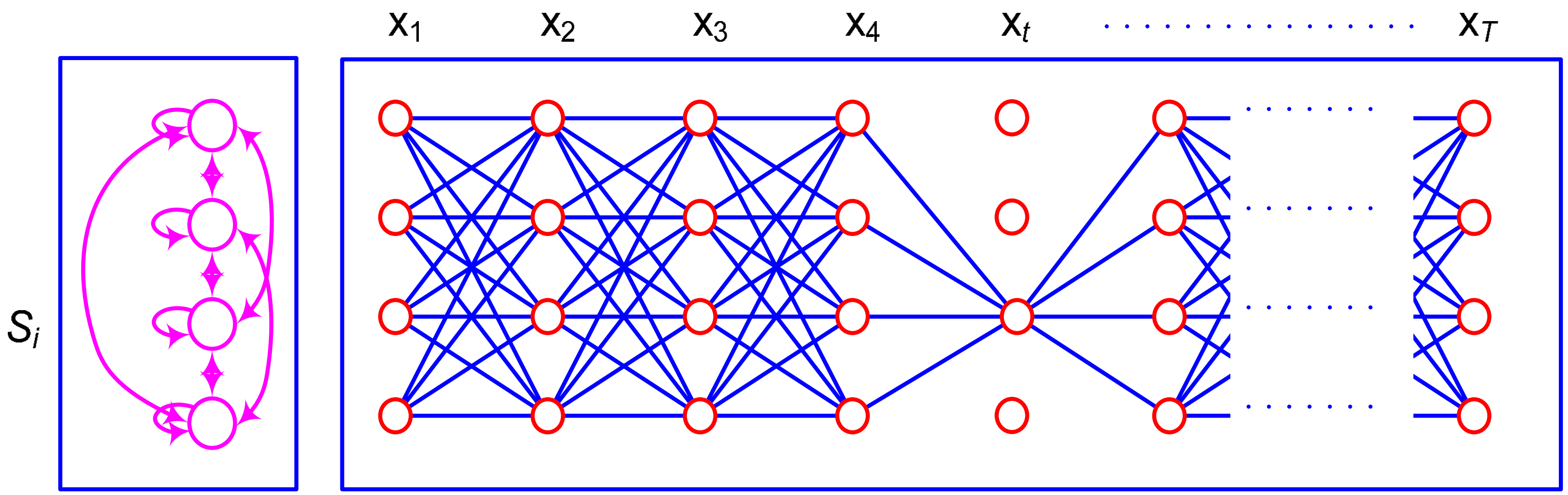
- With $\lambda^{(t)} = < \{a_{ij}\}, \{b_{ik}\}, \pi_i >$, estimate EXPECTATION of following quantities:
  - –Expected number of state *i* visiting
  - –Expected number of transitions from *i* to *j*

- With following quantities:
  - –Expected number of state *i* visiting
  - –Expected number of transitions from *i* to *j*
- Obtain the MAXIMUM LIKELIHOOD of

$$\lambda^{(t+1)} = < \{a'_{ij}\}, \{b'_{ik}\}, \pi'_i >$$

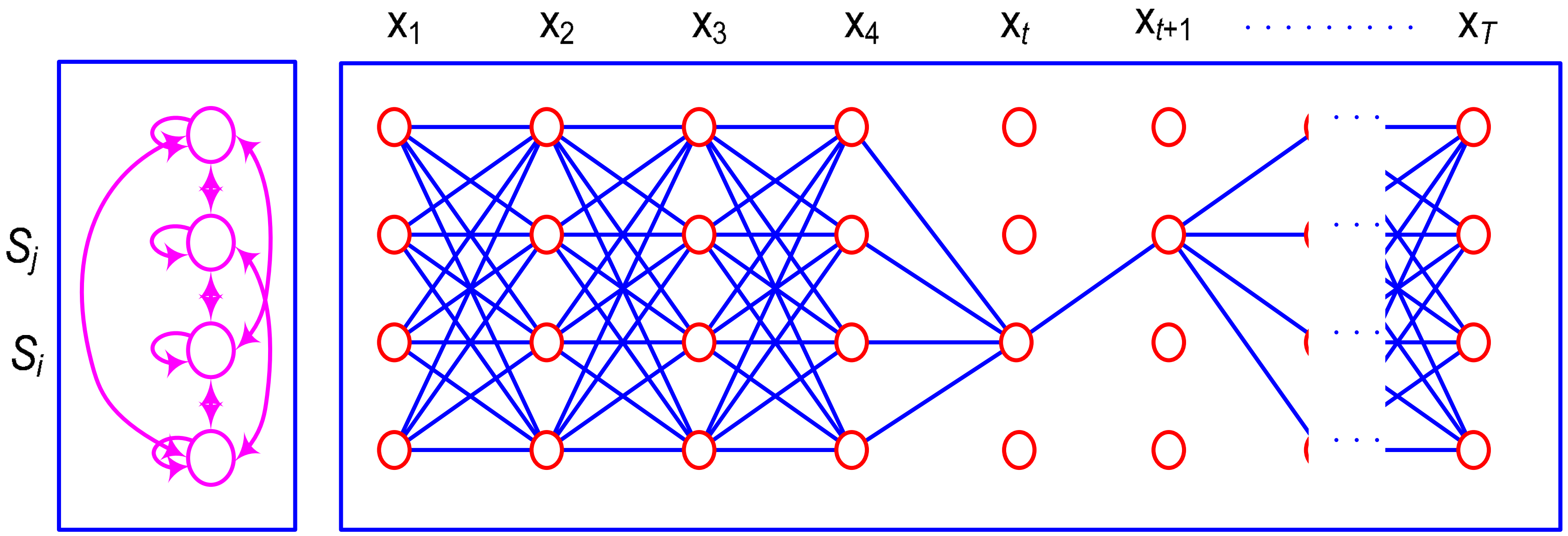
Expected Number of S_i Visiting

$$\begin{aligned}\gamma_t(i) &= P(q_t = S_i \mid X, \lambda) \\ &= \frac{P(q_t = S_i, X \mid \lambda)}{P(X \mid \lambda)} \\ &= \frac{\alpha_t(i)\beta_t(i)}{\sum_j \alpha_t(j)\beta_t(j)}\end{aligned}$$



Expected Number of Transition

$$\xi_t(i,j) = P(q_t = S_i, q_{t+1} = S_j \mid X, \lambda) = \frac{\alpha_t(i) a_{ij} b_j(x_{t+1}) \beta_{t+1}(j)}{\sum_i \sum_j \alpha_i(i) a_{ij} b_j(x_{t+1}) \beta_{t+1}(j)}$$



Parameter Reestimation

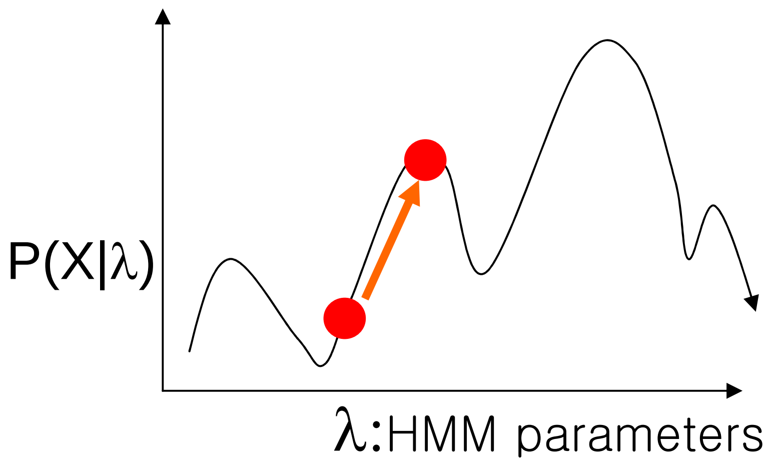
- MLE parameter estimation

$$\bar{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i, j)}{\sum_{t=1}^{T-1} \gamma_t(i)}$$

$$\bar{b}_j(v_k) = \frac{\sum_{t=1,\ s.t.\ x_t = v_k}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)}$$

$$\bar{\pi}_i = \gamma_1(i)$$

  - Iterative: $P(X \mid \lambda^{(t+1)}) \geq P(X \mid \lambda^{(t)})$
  - convergence proven:
  - arriving local optima



Pattern Classification

- Construct one HMM per each class *k*
  - $\lambda_1, \cdots, \lambda_N$
- Train each HMM λ_k with samples D_k
  - Baum-Welch reestimation algorithm

- Calculate model likelihood of $\lambda_1, \cdots, \lambda_N$ with observation X
  - Forward algorithm: $P(X \mid \lambda_k)$
- Find the model with maximum *a posteriori* probability

$$
\begin{aligned}
\lambda^* &= \operatorname{argmax}_{\lambda_k} \; P(\lambda_k \mid X) \\
&= \operatorname{argmax}_{\lambda_k} \; \frac{P(\lambda_k) P(X \mid \lambda_k)}{P(X)} \\
&= \operatorname{argmax}_{\lambda_k} P(\lambda_k) P(X \mid \lambda_k)
\end{aligned}
$$

HMM applications and Software

- On-line handwriting recognition
- Speech applications
- HMM toolbox for Matlab
- HTK (hidden Markov model Toolkit)

Software Tools for HMM

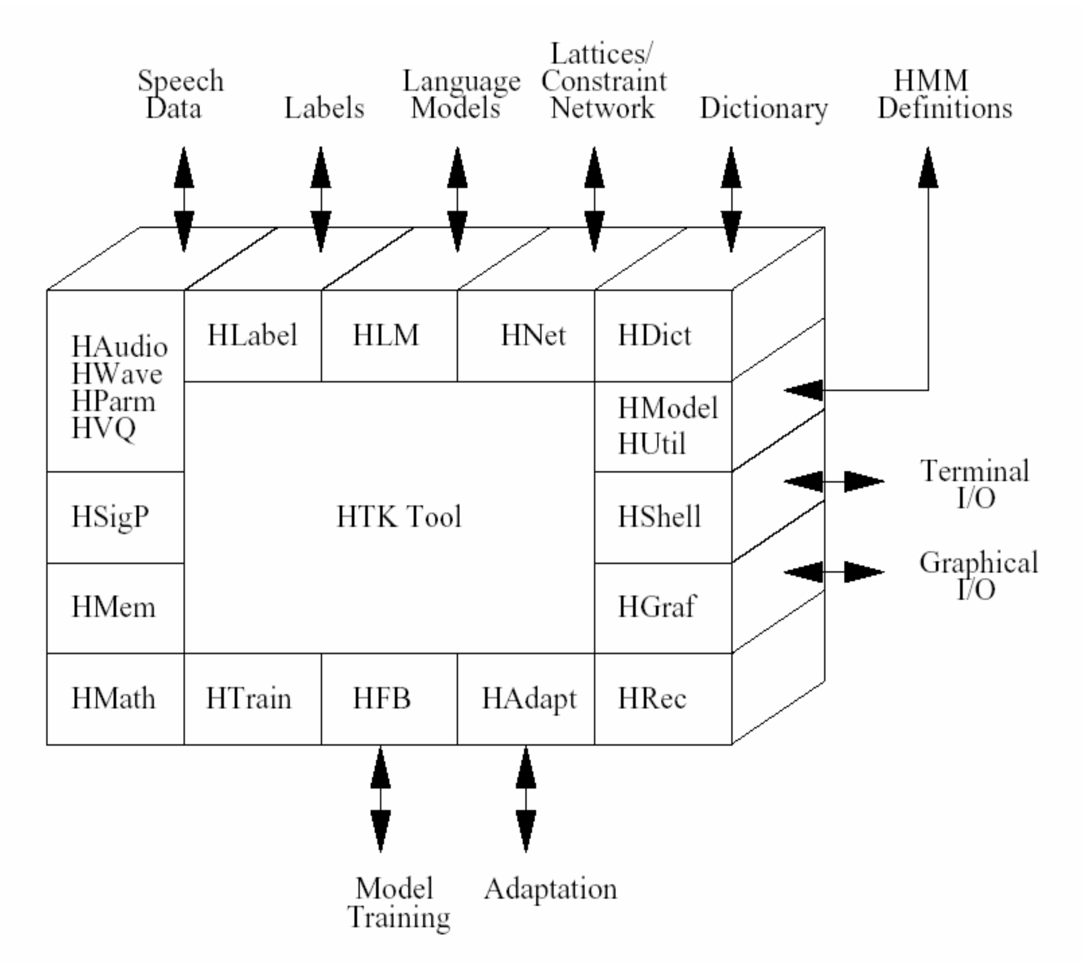
- **HMM toolbox for Matlab**
  - Developed by Kevin Murphy
  - Freely downloadable SW written in Matlab (Hmm… Matlab is not free!)
  - Easy-to-use: flexible data structure and fast prototyping by Matlab
  - Somewhat slow performance due to Matlab
  - Download: http://www.cs.ubc.ca/~murphyk/Software/HMM/hmm.html

- HTK (Hidden Markov toolkit)
  - Developed by Speech Vision and Robotics Group of Cambridge University
  - Freely downloadable SW written in C
  - Useful for speech recognition research: comprehensive set of programs for training, recognizing and analyzing speech signals
  - Powerful and comprehensive, but somewhat complicate
  - Download: http://htk.eng.cam.ac.uk/

What is HTK ?

- Hidden Markov Model Toolkit
- Set of tools for training and evaluation HMMs
- Primarily used in automatic speech recognition and economic modeling
- Modular implementation, (relatively) easy to extend

HTK Software Architecture



- HShell : User input/output & interaction with the OS
- HLabel : Label files
- HLM : Language model
- HNet : Network and lattices
- HDic : Dictionaries
- HVQ : VQ codebooks
- HModel : HMM definitions
- HMem : Memory management
- HGraf : Graphics
- HAdapt : Adaptation
- HRec : main recognition processing functions

Generic Properties of a HTK Tool

- Designed to run with a traditional command-line style interface
- Each tool has a number of required argument plus optional arguments

```
HFoo    -T    1    -f    34.3    -a    -s    myfile    file1   file2
```

  – This tool has two main arguments called file1 and file2 plus four optional arguments
  – -f : real number, -T : integer, -s : string, -a : no following value

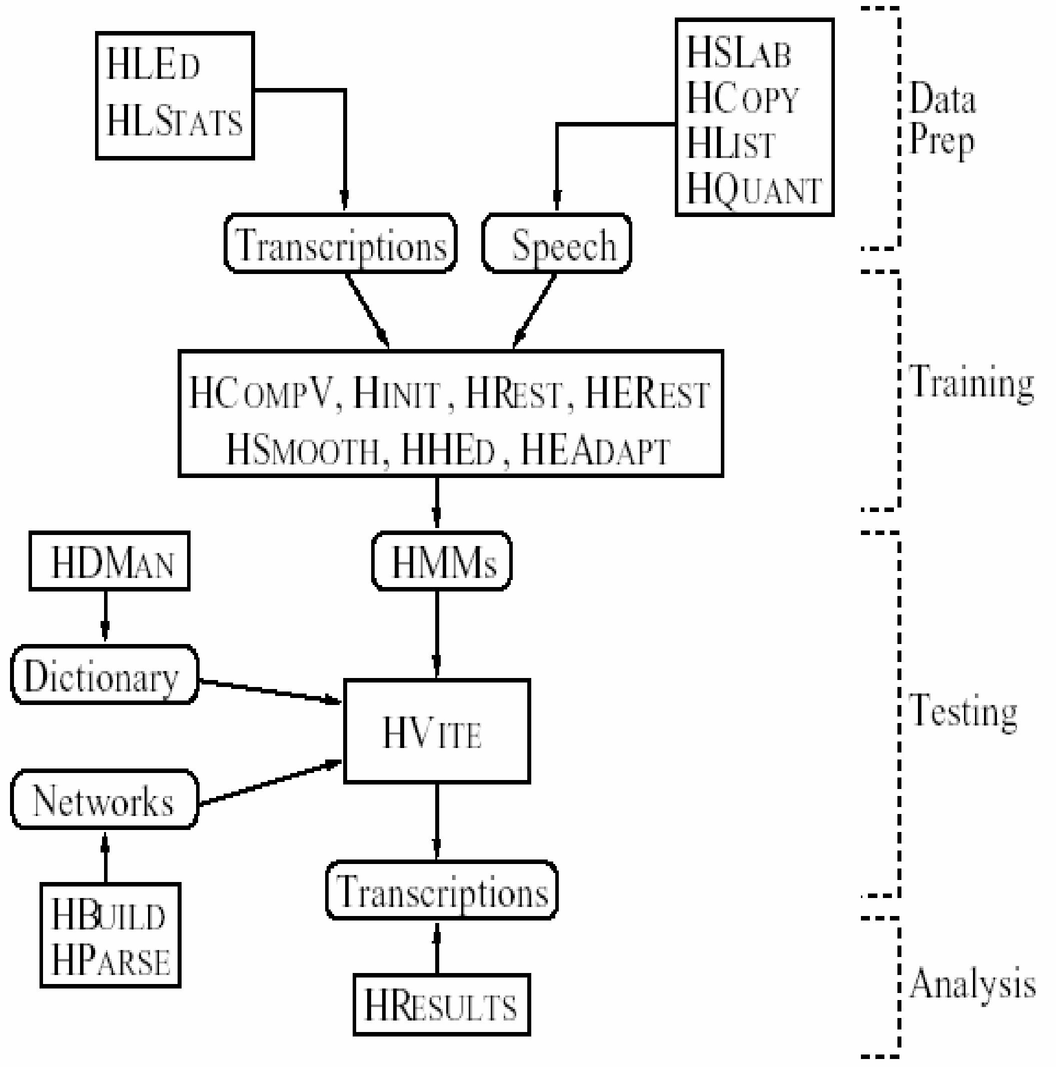
```
HFoo    -C    config    -f    34.3    -a    -s    myfile    file1    file2
```

  – HFoo will load the parameters stored in the configuration file config during its initialization procedures
  – Configuration parameters can sometimes by used as an alternative to using command line arguments

The Toolkit

- There are 4 main phases
  - data preparation, training, testing and analysis

- The Toolkit
  - Data Preparation Tools
  - Training Tools
  - Recognition Tools
  - Analysis Tools



< HTK Processing Stages >

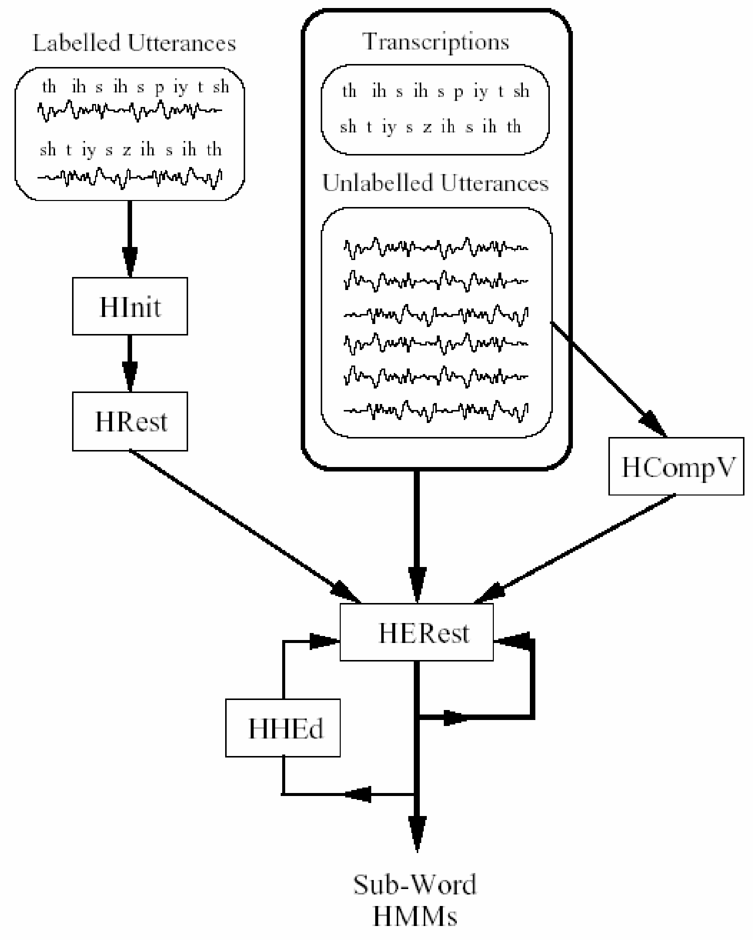
Data Preparation Tools

- A set of speech data file and their associated transcriptions are required
- It must by converted into the appropriate parametric form

- **HSlab** : Used both to record the speech and to manually annotate it with and required transcriptions
- **HCopy** : simply copying each file performs the required encoding
- **HList** : used to check the contents of any speech file
- **HLed** : output file to a single *Master Label file* MLF which is usually more convenient for subsequent processing
- **HLstats** : gather and display statistics on label files and where required
- **HQuant** : used to build a VQ codebook in preparation for building discrete probability HMM system

Training Tools

- If there is some speech data available for which the location of the sub-word boundaries have been marked, this can be used as *bootstrap data*
- **HInit** and **HRest** provide isolated word style training using the fully labeled bootstrap data
- Each of the required HMMs is generated individually



Training Tools (cont'd)

- **HInit** : iteratively compute an initial set of parameter values using a *segmental k-means* procedure
- **HRest** : process fully labeled bootstrap data using a Baum-Welch re-estimation procedure
- **HCompV** : all of the phone models are initialized to by identical and have state means and variances equal to the global speech mean and variance
- **HERest** : perform a single Baum-Welch re-estimation of the whole set of HMM phone models simultaneously
- **HHed** : apply a variety of parameter tying and increment the number of mixture components in specified distributions
- **HEadapt** : adapt HMMs to better model the characteristics of particular speakers using a small amount of training or adaptation data

Recognition Tools

- **HVite** : use the token passing algorithm to perform Viterbi-based speech recognition
- **HBuild** : allow sub-networks to be created and used within higher level networks
- **HParse** : convert EBNF into the equivalent word network
- **HSgen** : compute the empirical perplexity of the task
- **HDman** : dictionary management tool

Analysis Tools

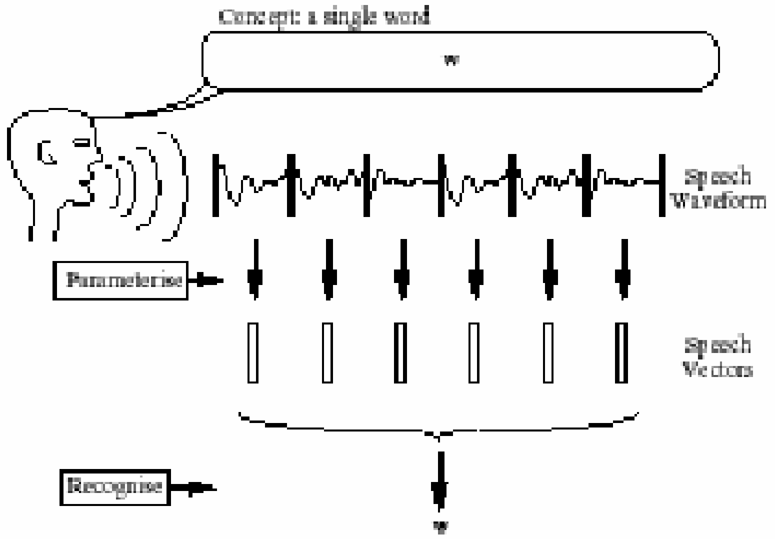
- **HResults**
  - Use dynamic programming to align the two transcriptions and count substitution, deletion and insertion errors
  - Provide speaker-by-speaker breakdowns, confusion matrices and time –aligned transcriptions
  - Compute *Figure of Merit* scores and *Receiver Operation Curve* information

HTK Example

- Isolated word recognition

$O = o_1, o_2, \cdots, o_T$ spoken word be represented by a sequence of vectors or *observations* O

$\arg\max_i \{P(w_i \mid O)\}$ Isolated word recognition
w_i : the *i*'th vocabulary word



$$\boxed{P(w_i \mid O)} = \frac{P(O \mid w_i) P(w_i)}{P(O)}$$

given set

depend on the likelihood $P(O|w_i)$

- Isolated word recognition (cont'd)

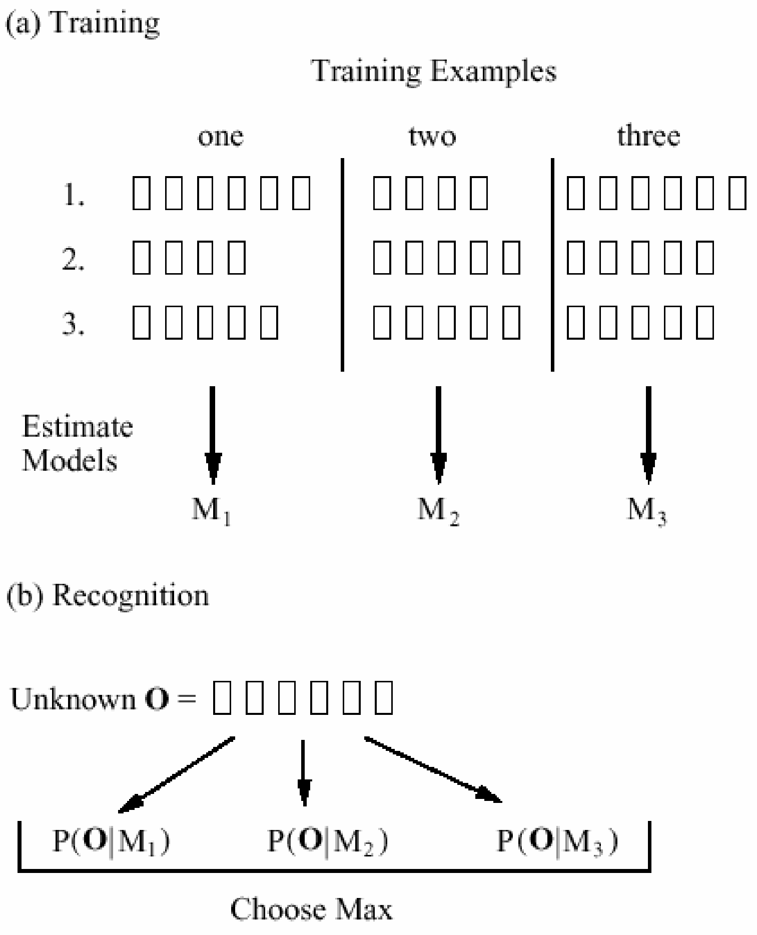
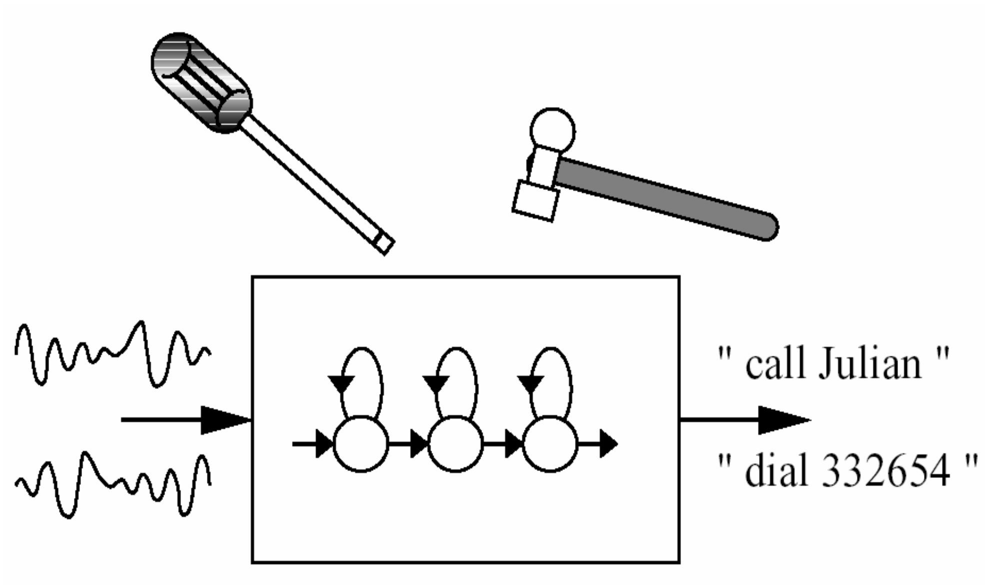


Fig. 1.4 Using HMMs for Isolated Word Recognition

Speech Recognition Example using HTK

- Recognizer for voice dialing application
  - Goal of our system
    - Provide a voice-operated interface for phone dialing
  - Recognizer
    - digit strings, limited set of names
    - sub-word based



References

- Hidden Markov Model
  - L.R. Rabiner, “A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition”, *IEEE Proc*. pp. 267-295, 1989.
  - L.R. Bahl et. al, “A Maximum Likelihood Approach to Continuous Speech Recognition”, *IEEE PAMI*, pp. 179-190, May. 1983.
  - M. Ostendorf, “From HMM’s to Segment Models: a Unified View of Stochastic Modeling for Speech Recognition”, *IEEE SPA*, pp 360-378, Sep., 1996.
- HMM Tutorials
  - 신봉기, “HMM Theory and Applications”, *2003컴퓨터비젼및패턴인식연구회 춘계워크샵 튜토리얼.*
  - 조성정, 한국정보과학회 ILVB Tutorial, 2005.04.16, 서울.
  - Sam Roweis, “Hidden Markov Models (SCIA Tutorial 2003)”, *http://www.cs.toronto.edu/~roweis/notes/scia03h.pdf*
  - Andrew Moore, “Hidden Markov Models”,

    http://www-2.cs.cmu.edu/~awm/tutorials/hmm.html

References (Cont.)

- HMM Applications
  - B.-K. Sin, J.-Y. Ha, S.-C. Oh, Jin H. Kim, “Network-Based Approach to Online Cursive Script Recognition”, *IEEE Transactions on Systems Man and Cybernetics Part B-Cybernetics,* Vol. 29, No. 2, pp.321-328, 1999.
  - J.-Y. Ha, "Structure Code for HMM Network-Based Hangul Recognition", *18th International Conference on Computer Processing of Oriental Languages*, pp.165-170, 1999.
  - 김무중, 김효숙, 김선주, 김병기, 하진영, 권철홍, “한국인을 위한 영어 발음 교정 시스템의 개발 및 성능 평가”, *말소리*, 제46호, pp.87-102, 2003.

- HMM Topology optimization
  - H. Singer, and M. Ostendorf, “Maximum likelihood successive state splitting,” ICASSP , 1996, pp. 601-604.
  - A. Stolcke, and S. Omohundro, “Hidden Markov model induction by Bayesian model merging,” Advances in NIPS. 1993, pp. 11-18. San Mateo, CA: Morgan Kaufmann.
  - 0 A. Biem, J.-Y. Ha, J. Subrahmonia, "A Bayesian Model Selection Criterion for HMM Topology Optimization", International Conference on Acoustics Speech and Signal Processing, pp.I989~I992, IEEE Signal Processing Society, 2002.
  - A. Biem, “A Model Selection Criterion for Classification: Application to HMM Topology Optimization,” ICDAR 2003, pp. 204-210, 2003.

- HMM Software
  - Kevin Murphy, “HMM toolbox for Matlab”, freely downloadable SW written in Matlab, http://www.cs.ubc.ca/~murphyk/Software/HMM/hmm.html
  - Speech Vision and Robotics Group of Cambridge University, “HTK (Hidden Markov toolkit)”, freely downloadable SW written in C, http://htk.eng.cam.ac.uk/
  - Sphinx at CMU

    http://cmusphinx.sourceforge.net/html/cmusphinx.php