

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ



بازشناسی الگو

درس ۱۱

طبقه‌بندی وابسته به مضمون

Context-Dependent Classification

کاظم فولادی قلعه
دانشکده مهندسی، پردیس فارابی
دانشگاه تهران

<http://courses.fouladi.ir/pr>

CONTEXT DEPENDENT CLASSIFICATION

❖ Remember: Bayes rule

$$P(\omega_i|\underline{x}) > P(\omega_j|\underline{x}), \quad \forall j \neq i$$

❖ Here: The class to which a feature vector belongs depends on:

- Its own value
- The values of the other features
- An existing relation among the various classes

- ❖ This interrelation **demands** the classification to be performed **simultaneously** for **all available** feature vectors
- ❖ Thus, we will assume that the training vectors $\underline{x}_1, \underline{x}_2, \dots, \underline{x}_N$ occur in **sequence, one after the other** and we will refer to them as **observations**

❖ The Context Dependent Bayesian Classifier

➤ Let $X : \{\underline{x}_1, \underline{x}_2, \dots, \underline{x}_N\}$

➤ Let $\omega_i, i = 1, 2, \dots, M$

➤ Let Ω_i be a sequence of classes, that is

$$\Omega_i : \omega_{i1} \omega_{i2} \dots \omega_{iN}$$

There are M^N of those

➤ Thus, the Bayesian rule can equivalently be stated as

$$X \rightarrow \Omega_i : P(\Omega_i | X) > P(\Omega_j | X) \quad \forall i \neq j, \quad i, j = 1, 2, \dots, M^N$$

❖ Markov Chain Models (for class dependence)

$$P(\omega_{i_k} \mid \omega_{i_{k-1}}, \omega_{i_{k-2}}, \dots, \omega_{i_1}) = P(\omega_{i_k} \mid \omega_{i_{k-1}})$$

NOW remember:

$$\begin{aligned} P(\Omega_i) &= P(\omega_{i_1}, \omega_{i_2}, \dots, \omega_{i_N}) \\ &= P(\omega_{i_N} \mid \omega_{i_{N-1}}, \dots, \omega_{i_1}) \cdot P(\omega_{i_{N-1}} \mid \omega_{i_{N-2}}, \dots, \omega_{i_1}) \dots P(\omega_{i_1}) \end{aligned}$$

$$P(\Omega_i) = \left(\prod_{k=2}^N P(\omega_{i_k} \mid \omega_{i_{k-1}}) \right) P(\omega_{i_1})$$

❖ Assume:

- \underline{x}_i statistically mutually independent
- The pdf in one class independent of the others, then

$$p(X|\Omega_i) = \prod_{k=1}^N p(\underline{x}_k|\omega_{i_k})$$

- ❖ From the above, the Bayes rule is readily seen to be equivalent to:

$$P(\Omega_i|X)(\succ)P(\Omega_j|X)$$

$$P(\Omega_i)p(X|\Omega_i)(\succ)P(\Omega_j)p(X|\Omega_j)$$

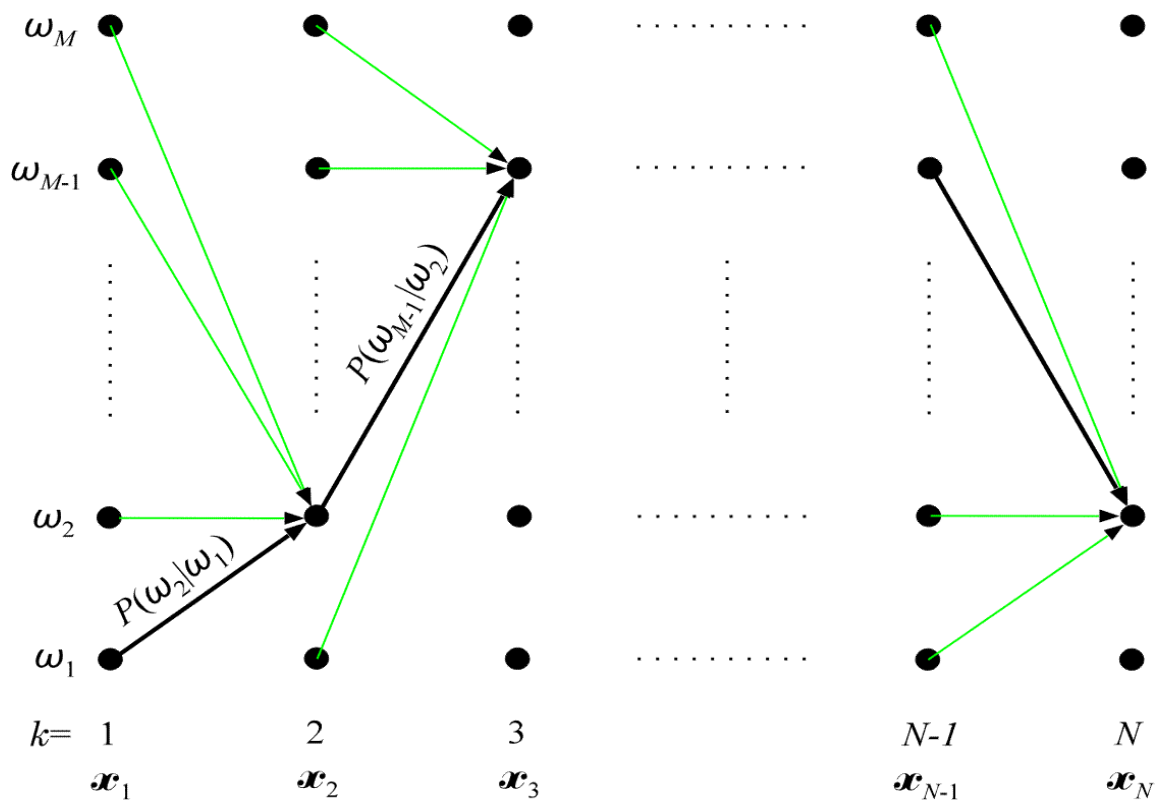
that is, it rests on

$$p(X|\Omega_i)P(\Omega_i) = P(\omega_{i_1})p(\underline{x}_1|\omega_{i_1}).$$

$$\prod_{k=2}^N P(\omega_{i_k}|\omega_{i_{k-1}})p(\underline{x}_k|\omega_{i_k})$$

- ❖ To find the above maximum in brute-force task we need $O(NM^N)$ operations!!

❖ The Viterbi Algorithm



➤ Thus, each Ω_i corresponds to one path through the trellis diagram. One of them is the optimum (e.g., black). The classes along the optimal path determine the classes to which ω_i are assigned.

➤ To each transition corresponds a cost. For our case

- $\hat{d}(\omega_{i_k}, \omega_{i_{k-1}}) = P(\omega_{i_k} | \omega_{i_{k-1}}) \cdot p(\underline{x}_k | \omega_{i_k})$

- $\hat{d}(\omega_{i_1}, \omega_{i_0}) \equiv P(\omega_{i_1}) p(\underline{x}_{i_1} | \omega_{i_1})$

- $\hat{D} = \prod_{k=1}^N \hat{d}(\omega_{i_k}, \omega_{i_{k-1}}) = p(X | \Omega_i) P(\Omega_i)$

- Equivalently

$$\ln \hat{D} = \sum_{k=1}^N \ln \hat{d}(.,.) \equiv D = \sum_{k=1}^N d(.,.)$$

where,

$$d(\omega_{i_k}, \omega_{i_{k-1}}) = \ln \hat{d}(\omega_{i_k}, \omega_{i_{k-1}})$$

- Define the cost up to a node , k ,

$$D(\omega_{i_k}) = \sum_{r=1}^k d(\omega_{i_r}, \omega_{i_{r-1}})$$

► Bellman's principle now states

$$D_{\max}(\omega_{i_k}) = \max_{i_{k-1}} \left[D_{\max}(\omega_{i_{k-1}}) + d(\omega_{i_k}, \omega_{i_{k-1}}) \right]$$

$$i_k, i_{k-1} = 1, 2, \dots, M$$

$$D_{\max}(\omega_{i_0}) = 0$$

► The optimal path terminates at $\omega_{i_N}^*$:

$$\omega_{i_N}^* = \arg \max_{\omega_{i_N}} D_{\max}(\omega_{i_N})$$

- Complexity $O(NM^2)$

❖ Channel Equalization

➤ The problem

- $x_k = f(I_k, I_{k-1}, \dots, I_{k-n+1}) + n_k$

- $\underline{x}_k \equiv [x_k, x_{k-1}, \dots, x_{k-l+1}]^T$

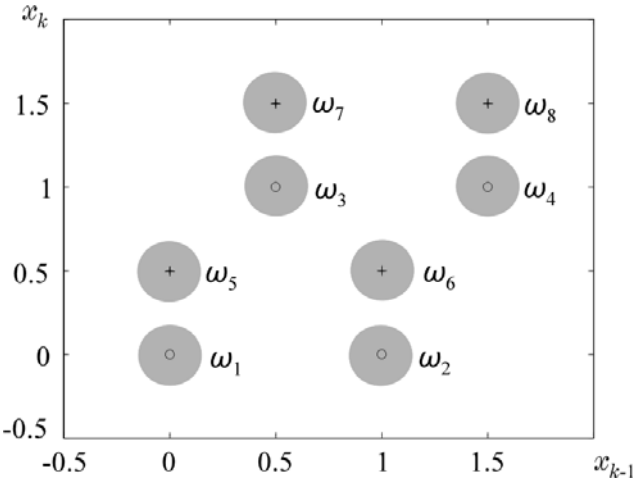
- $\underline{x}_k \rightarrow \hat{I}_k \text{ or } \hat{I}_{k-r}$

$$\underline{x}_k \rightarrow \boxed{\text{equalizer}} \rightarrow \hat{I}_{k-r}$$

► Example

- $x_k = 0.5I_k + I_{k-1} + n_k$
- $\underline{x}_k = \begin{bmatrix} x_k \\ x_{k-1} \end{bmatrix}, l = 2$
- In \underline{x}_k three input symbols are involved:

$$I_k, I_{k-1}, I_{k-2}$$



I_k	I_{k-1}	I_{k-2}	x_k	x_{k-1}	
0	0	0	0	0	ω_1
0	0	1	0	1	ω_2
0	1	0	1	0.5	ω_3
0	1	1	1	1.5	ω_4
1	0	0	0.5	0	ω_5
1	0	1	0.5	1	ω_6
1	1	0	1.5	0.5	ω_7
1	1	1	1.5	1.5	ω_8

► Not all transitions are allowed

- $(I_k, I_{k-1}, I_{k-2}) = (0, 0, 1)$

- Then

$$(I_{k+1}, I_k, I_{k-1}) \begin{cases} (1, 0, 0) \\ (0, 0, 0) \end{cases}$$

- $\omega_2 \begin{cases} \omega_5 \\ \omega_1 \end{cases}$

$$P(\omega_2 | \omega_i) = \begin{cases} 0.5, i = 5, 1 \\ 0, \text{otherwise} \end{cases}$$

- In this context, ω_i are related to **states**. Given the current state and the transmitted bit, I_k , we determine the next state. The probabilities $P(\omega_i|\omega_j)$ define the state dependence model.

➤ The transition cost

- $$d(\omega_{i_k}, \omega_{i_{k-1}}) = d_{\omega_{i_k}}(\underline{x})$$

$$= \left\| \underline{x}_k - \underline{\mu}_{i_k} \right\| = ((\underline{x}_k - \underline{\mu}_{i_k})^T \sum_{i_k}^{-1} (\underline{x}_k - \underline{\mu}_{i_k}))^{\frac{1}{2}}$$

for all allowable transitions

➤ Assume:

- Noise white and Gaussian
- A channel impulse response \hat{f} estimate to be available
- $(x_k - \hat{f}(I_k, \dots, I_{k-n+1})) \approx n_k$
- $d(\omega_{i_k}, \omega_{i_{k-1}}) = \ln p(x_k | \omega_{i_k}) = \ln p(n_k)$

$$\propto -(x_k - \hat{f}(I_k, \dots, I_{k-n+1}))^2$$
- The states are determined by the values of the binary variables

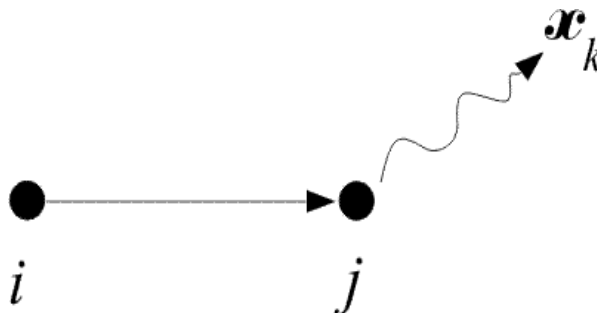
$$I_{k-1}, \dots, I_{k-n+1}$$

For $n = 3$, there will be 4 states

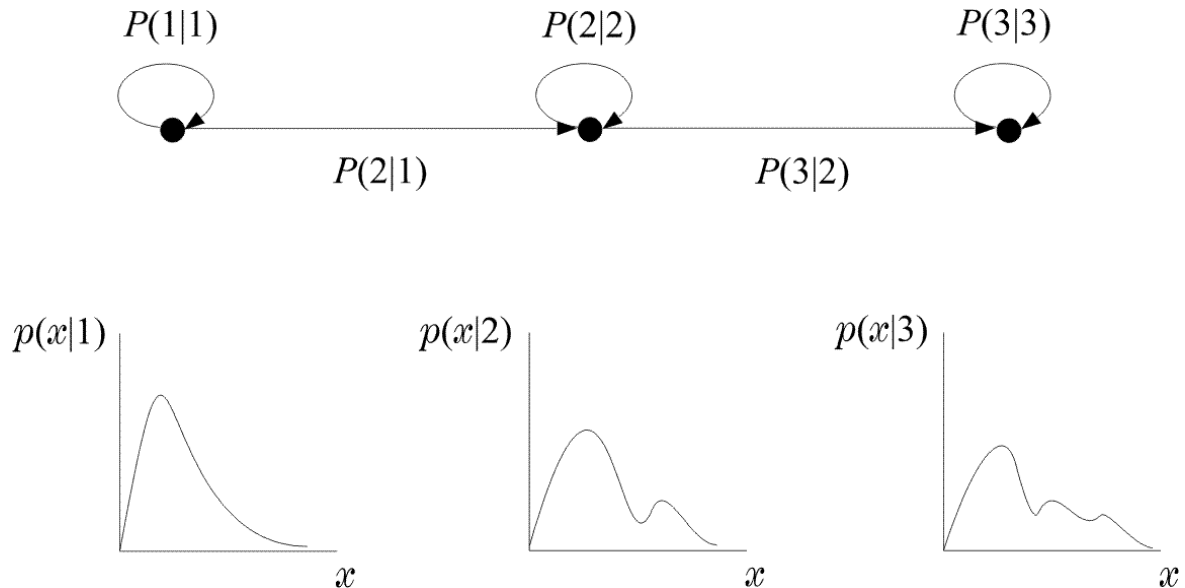
❖ Hidden Markov Models

- In the channel equalization problem, the states are **observable** and can be “learned” during the training period
- Now we shall assume that states **are not observable** and can only be **inferred** from the training data
- Applications:
 - Speech and Music Recognition
 - OCR
 - Blind Equalization
 - Bioinformatics

- An HMM is a **stochastic finite state automaton**, that generates the observation sequence, $\underline{x}_1, \underline{x}_2, \dots, \underline{x}_N$
- We assume that: **The observation sequence is produced as a result of successive transitions between states, upon arrival at a state:**



- This type of modeling is used for **nonstationary stochastic processes** that undergo **distinct** transitions among a set of different stationary processes.



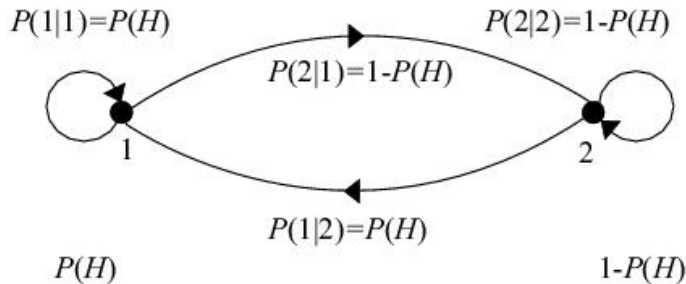
➤ Examples of HMM:

- The single coin case: Assume a coin that is tossed behind a curtain. All it is available to us is the outcome, i.e., H or T . Assume the two states to be:

$$S = 1 \rightarrow H \qquad S = 2 \rightarrow T$$

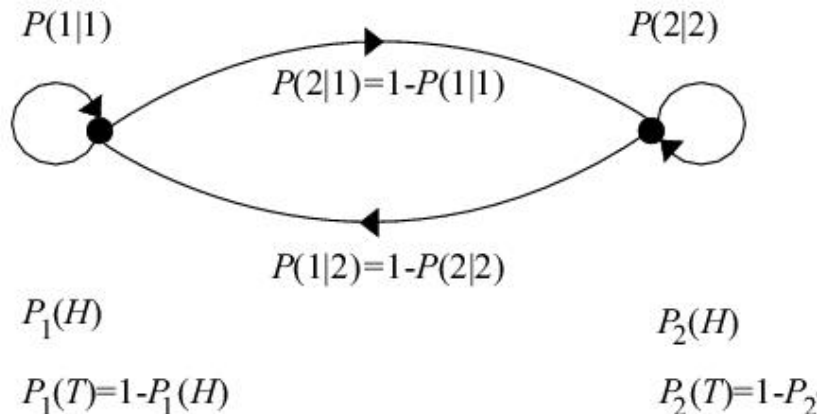
This is also an example of a random experiment with observable states. The model is characterized by a single parameter, e.g., $P(H)$. Note that

$$P(1|1) = P(H) \qquad P(2|1) = P(T) = 1 - P(H)$$



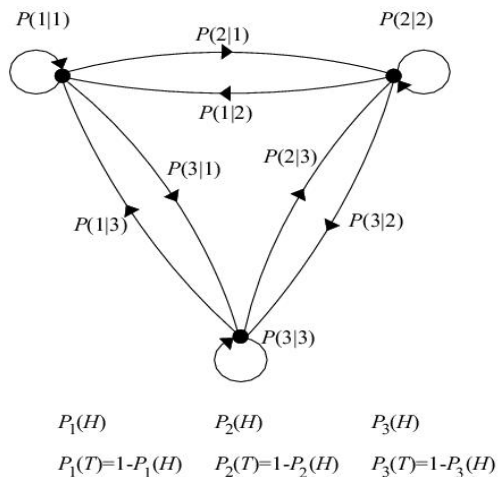
- The two-coins case: For this case, we observe a sequence of H or T . However, we have no access to know **which coin was tossed**. Identify one state for each coin. This is an example where **states are not observable**. H or T can be emitted from either state. The model depends on **four** parameters.

$$P_1(H), P_2(H), P(1|1), P(2|2)$$



(b)

- The three-coins case example is shown below:



- Note that in all previous examples, specifying the model is equivalent to knowing:
 - The probability of each observation (H, T) to be emitted from each state.
 - The transition probabilities among states: $P(i|j)$.

- A general HMM model is characterized by the following set of parameters
- K , number of states
 - $P(i|j), i, j = 1, 2, \dots, K$
 - $p(\underline{x}|i), i = 1, 2, \dots, K$
 - $P(i), i = 1, 2, \dots, K$, initial state probabilities, $P(.)$

That is:

$$S = \{P(i|j), p(\underline{x}|i), P(i), K\}$$

- What is the problem in Pattern Recognition
 - Given M reference patterns, each described by an HMM, find the parameters, S , for each of them (training)
 - Given an unknown pattern, find to which one of the M , known patterns, matches best (recognition)

► Recognition: Any path method

- Assume the M models to be known (M classes).
- A sequence of observations, X , is given.
- Assume observations to be **emissions** upon the **arrival** on successive states
- Decide in favor of the model S^* (from the M available) according to the **Bayes rule**

$$S^* = \arg \max_S P(S|X)$$

for **equiprobable patterns**

$$S^* = \arg \max_S p(X|S)$$

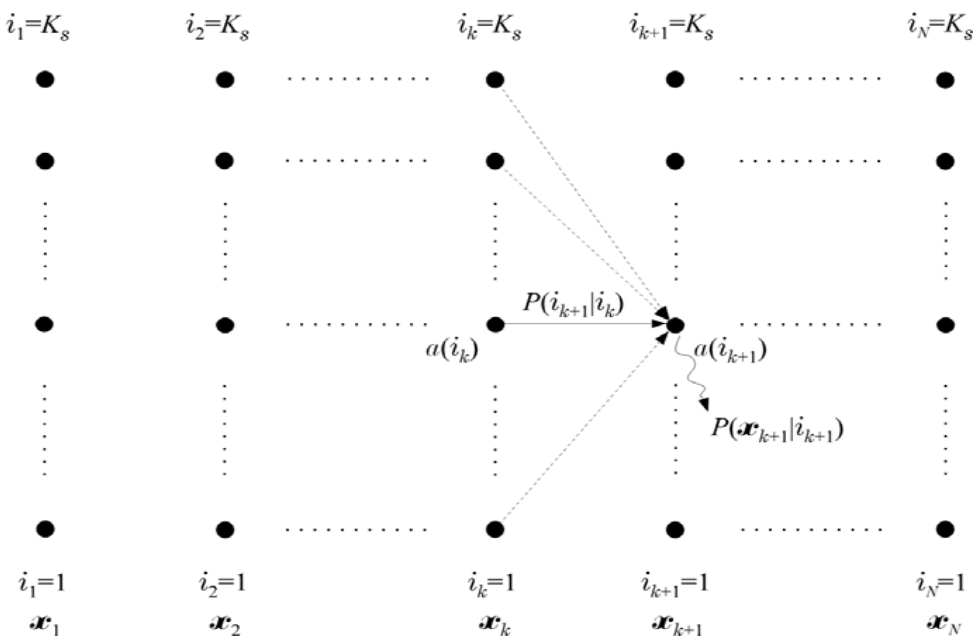
- For each model S there is more than one possible sets of successive state transitions Ω_i , each with probability $P(\Omega_i|S)$

Thus:

$$\begin{aligned}
 P(X|S) &= \sum_i p(X, \Omega_i|S) \\
 &= \sum_i p(X|\Omega_i, S) P(\Omega_i|S)
 \end{aligned}$$

- For the efficient computation of the above DEFINE

$$\begin{aligned}
 - \quad \alpha(i_{k+1}) &= p(\underline{x}_1, \dots, \underline{x}_{k+1}, i_{k+1}|S) \\
 &= \underbrace{\sum_{i_k} \alpha(i_k)}_{\text{History}} \underbrace{P(i_{k+1}|i_k) p(\underline{x}_{k+1}|i_{k+1}))}_{\text{Local activity}}
 \end{aligned}$$



- Observe that

$$P(X|S) = \sum_{i_N=1}^{K_s} \alpha(i_N)$$

Compute this for each S

- Some more quantities

$$\begin{aligned} - \quad \beta(i_k) &= p(\underline{x}_{k+1}, \underline{x}_{k+2}, \dots, \underline{x}_N | i_k, S) \\ &= \sum_{i_{k+1}} \beta(i_{k+1}) P(i_{k+1} | i_k) p(\underline{x}_{k+1} | i_{k+1}) \end{aligned}$$

$$\begin{aligned} - \quad \gamma(i_k) &= p(\underline{x}_1, \dots, \underline{x}_N, i_k | S) \\ &= \alpha(i_k) \beta(i_k) \end{aligned}$$

► Training

- The philosophy:

Given a training set X , known to belong to the specific model, estimate the unknown parameters of S , so that the output of the model, e.g.

$$p(X|S) = \sum_{i_N=1}^{K_s} \alpha(i_N)$$

to be maximized

- This is a ML estimation problem with missing data

➤ Assumption: Data \underline{x} discrete

$$\underline{x} \in \{1, 2, \dots, r\} \Rightarrow p(\underline{x}|i) \equiv P(\underline{x}|i)$$

➤ Definitions:

- $$\xi_k(i, j) = \frac{\alpha(i_k = i)P(j|i)P(\underline{x}_{k+1}|j)\beta(i_{k+1} = j)}{P(X|S)}$$
- $$\gamma_k(i) = \frac{\alpha(i_k = i)\beta(i_k = i)}{P(X|S)}$$

► The Algorithm:

- Initial conditions for all the unknown parameters.

Compute $P(X|S)$

- Step 1: From the current estimates of the model parameters **reestimate** the new model S from

$$- \quad \bar{P}(j|i) = \frac{\sum_{k=1}^{N-1} \xi_k(i, j)}{\sum_{k=1}^{N-1} \gamma_k(i)} \quad \left(= \frac{\# \text{ of transitions from } i \text{ to } j}{\# \text{ of transitions from } i} \right)$$

$$- \quad \bar{P}_{\underline{x}}(r|i) = \frac{\sum_{(k=1 \text{ and } \underline{x} \rightarrow r)}^N \gamma_k(i)}{\sum_{k=1}^N \gamma_k(i)} \quad \left(= \frac{\text{at state } i \text{ and } \underline{x} = r}{\# \text{ of being at state } i} \right)$$

$$- \quad \bar{P}(i) = \gamma_1(i)$$

- Step 2: Compute $P(X|\bar{S})$. If $P(X|\bar{S}) - P(X|S) > \varepsilon$, $S = \bar{S}$ go to step 1. Otherwise stop
- Remarks:
 - Each iteration improves the model
$$\bar{S} : P(X|\bar{S}) > P(X|S)$$
 - The algorithm converges to a maximum (local or global)
 - The algorithm is an implementation of the EM algorithm