Criterio de implementación de un HMM usando LogProb

Jorge Luis Guevara Díaz
Escuela de Informática

Introducción

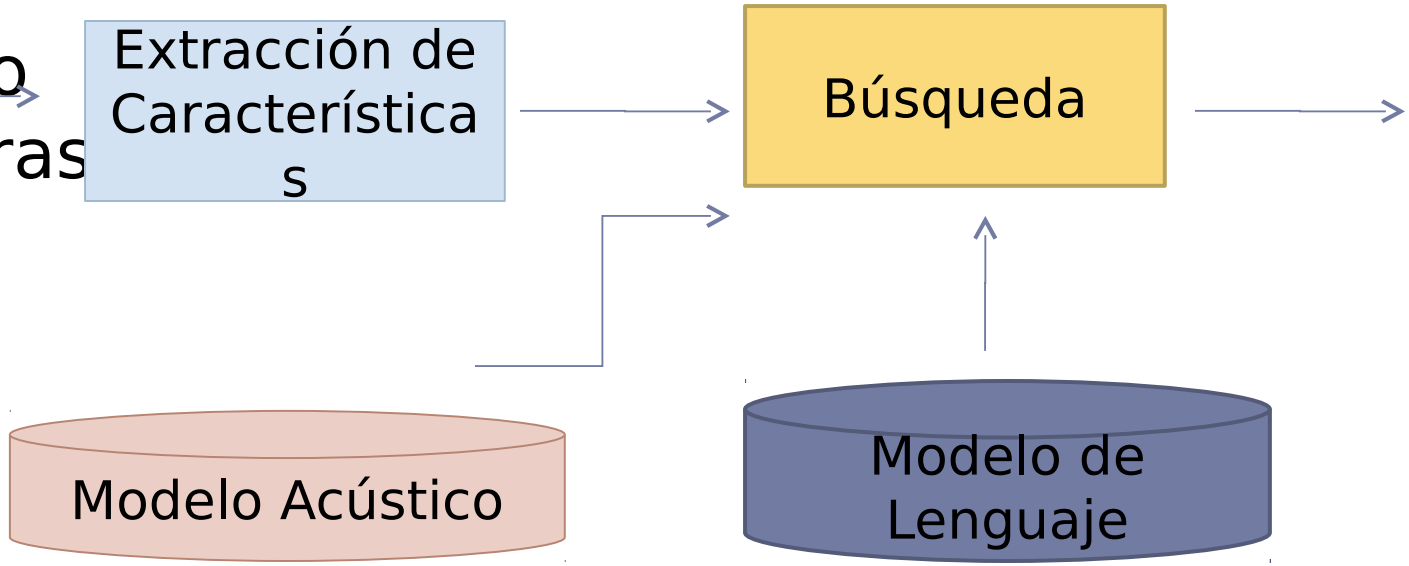
- Introducción
  - Motivación
- Modelos Ocultos de Markov - HMM
  - Definición
  - Algoritmo forward-backward
  - Algoritmo de Viterbi
  - Algoritmo Baum-Welch
- Criterio de implementación
  - Algoritmo f*
  - Algoritmo h*
  - Análisis y prueba de correctitud

Motivación

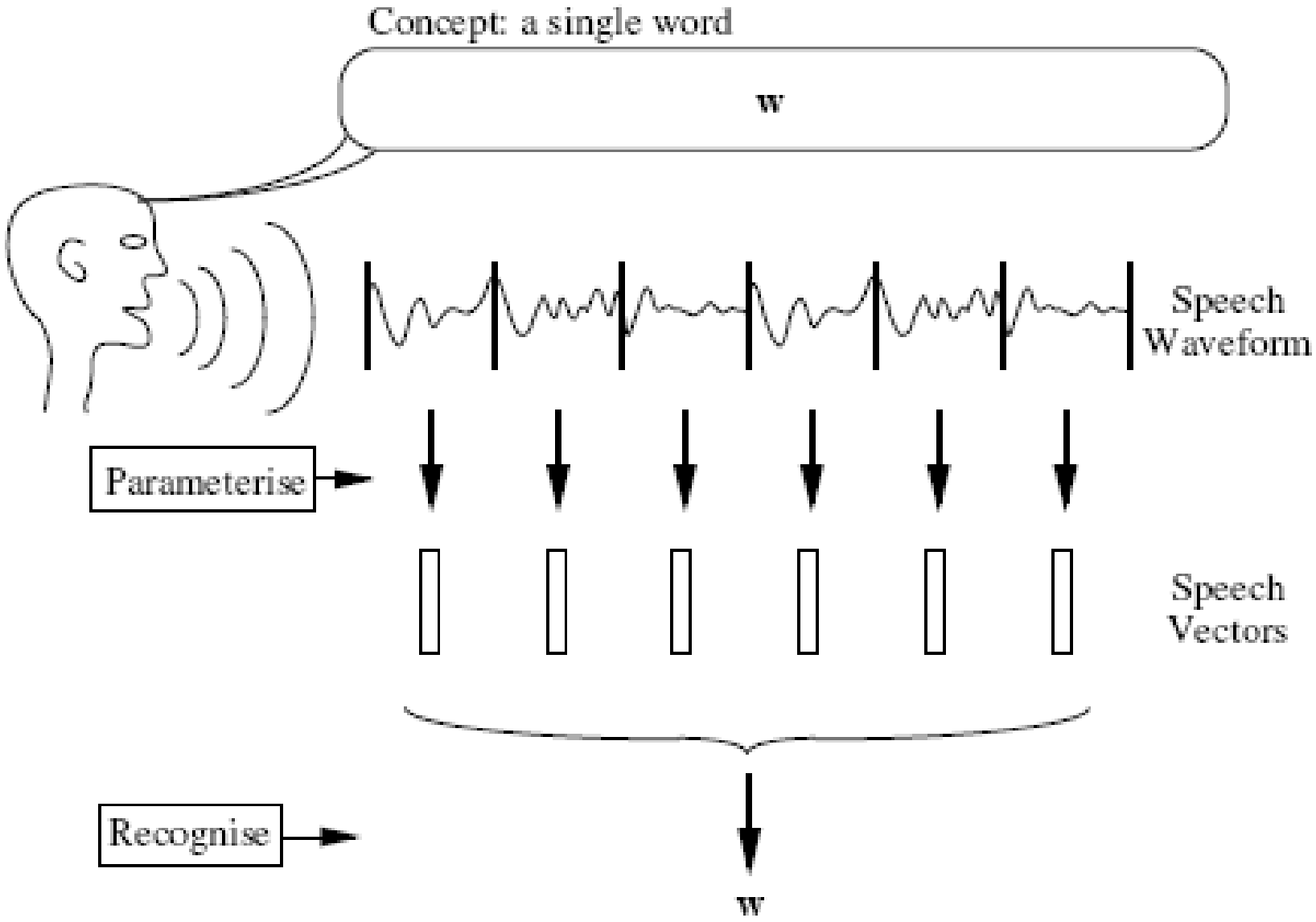
- Por que estudiar HMM?
  - Tiene un rango de aplicaciones muy interesantes como:
    - Reconocimiento automático del Habla
    - Reconocimiento de escritura a mano
    - Reconocimiento de gesturas
    - Reconocimiento de firmas
    - Procesamiento y etiquetado automático de secuencias musicales
    - Bioinformática para modelar secuencias DNA y proteinas
    - Es un ejercicio de programación divertido!!

Reconocimiento automático del habla

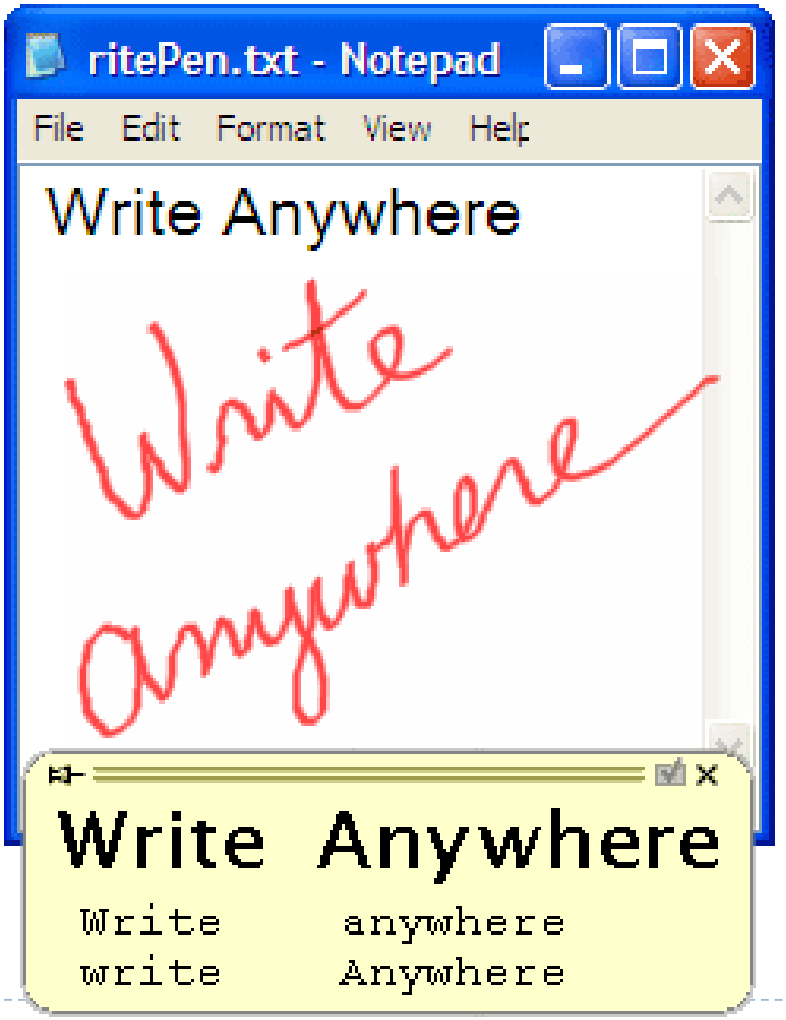
- audio palabras



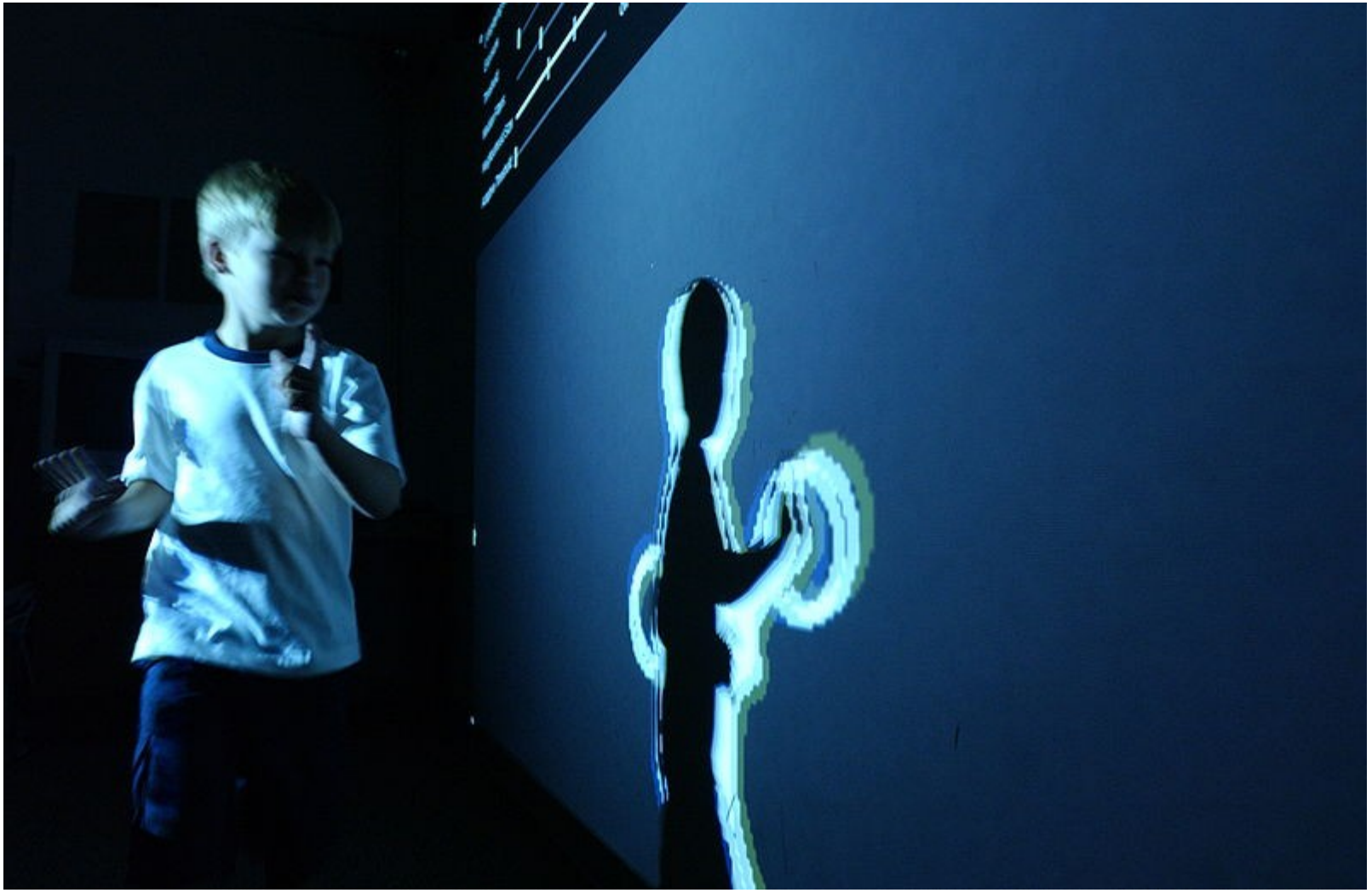
- $W = \arg\max_W P(O|W)\,P(W)$


Concept: a single word
w
Speech Waveform
Parameterise
Speech Vectors
Recognise
w

Reconocimiento de escritura a mano

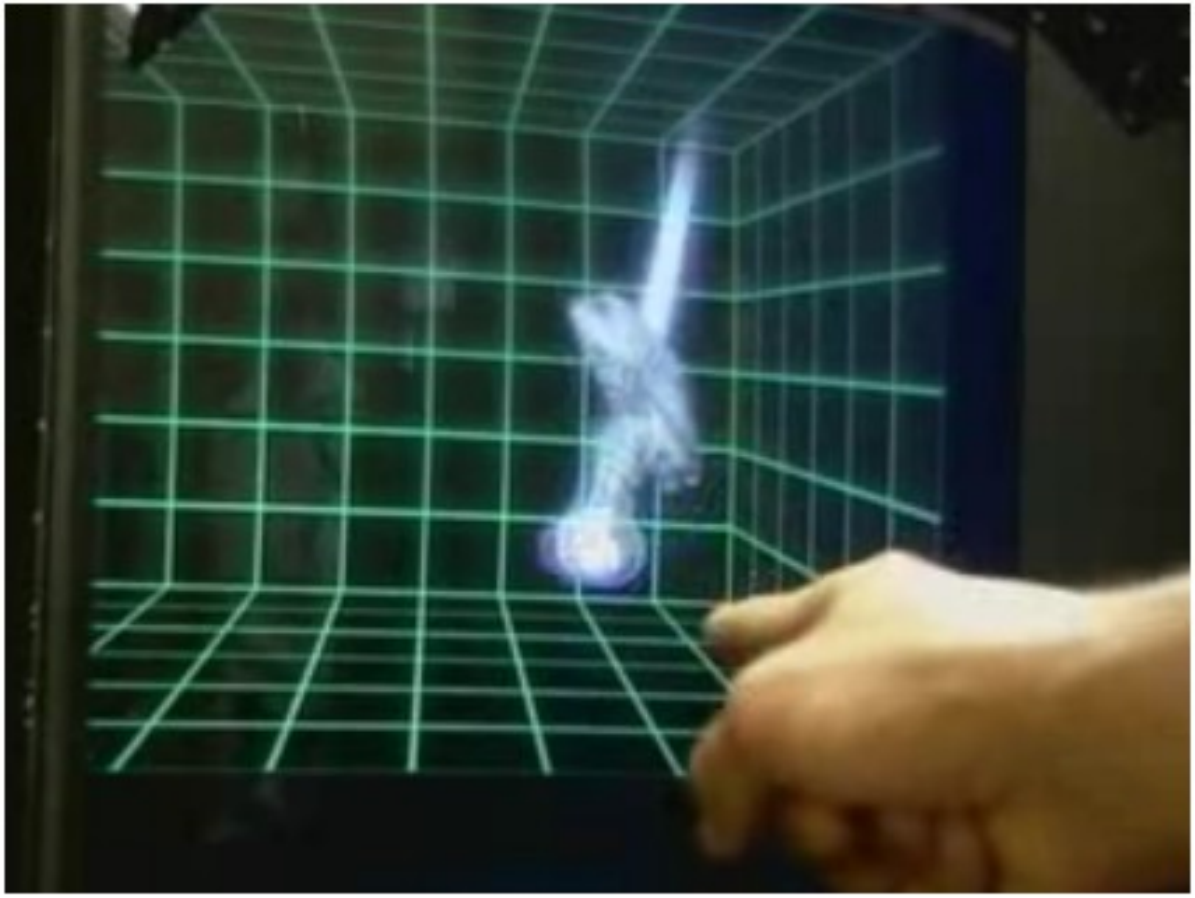


Reconocimiento de gestura



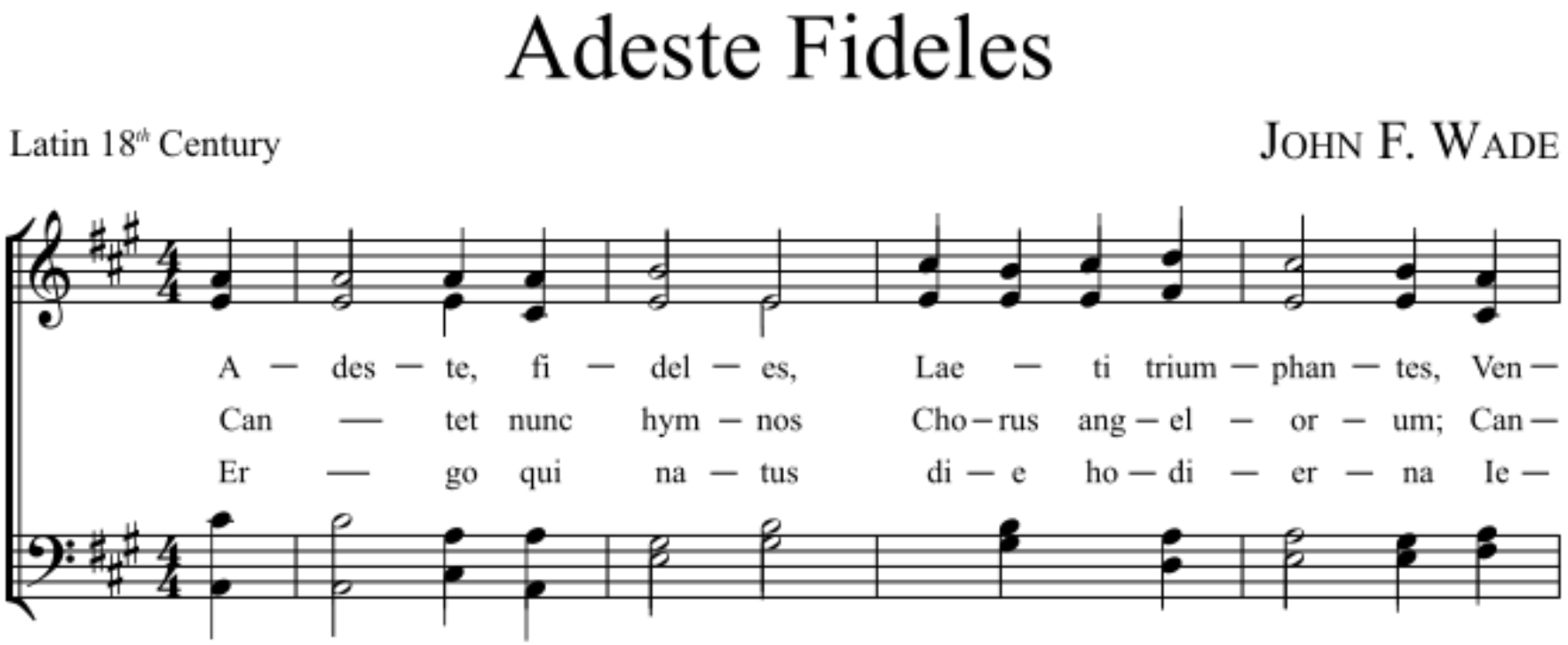
Hand and Face Tracking

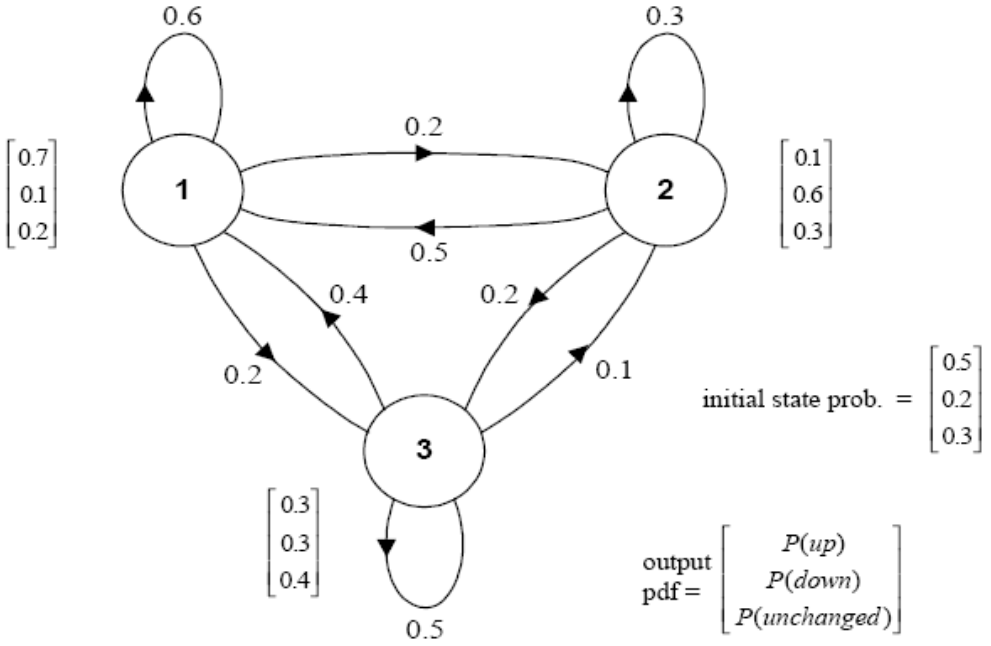
Finger tracking



Reconocimiento de firmas

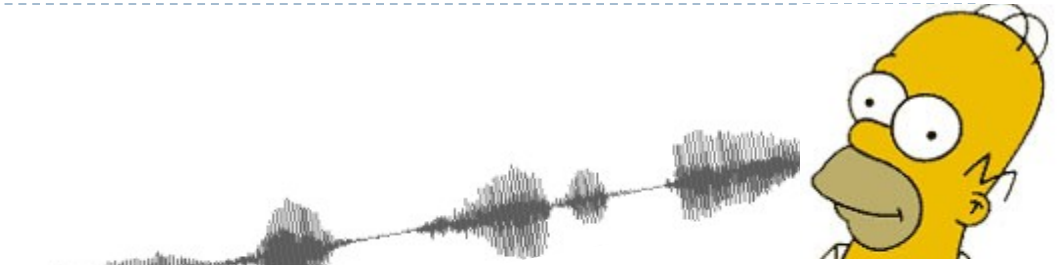
Procesamiento y etiquetado automático de secuencias musicales





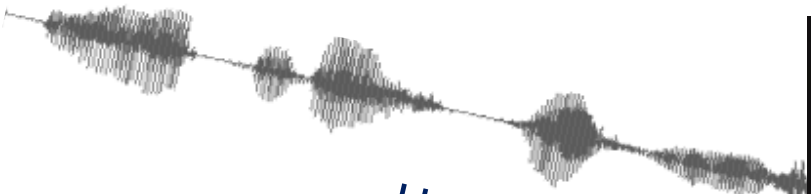
Modelos Ocultos de Markov

Modelos Ocultos de Markov (HMM)



Holaaa

Hooolaaa



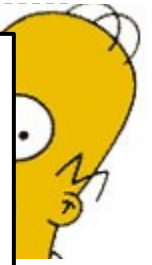
Hooola

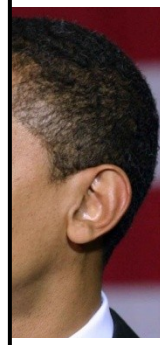
Modelos Ocultos de Markov (HMM)

En nuestro cerebro tenemos un "modelo" que nos permite identificar la palabra hola sea dicho por cualquier persona y de cualquier manera

Este modelo lo tenemos gracias a un proceso de aprendizaje

Aprendimos a entender "hola" pues escuchamos esa palabra muchas veces

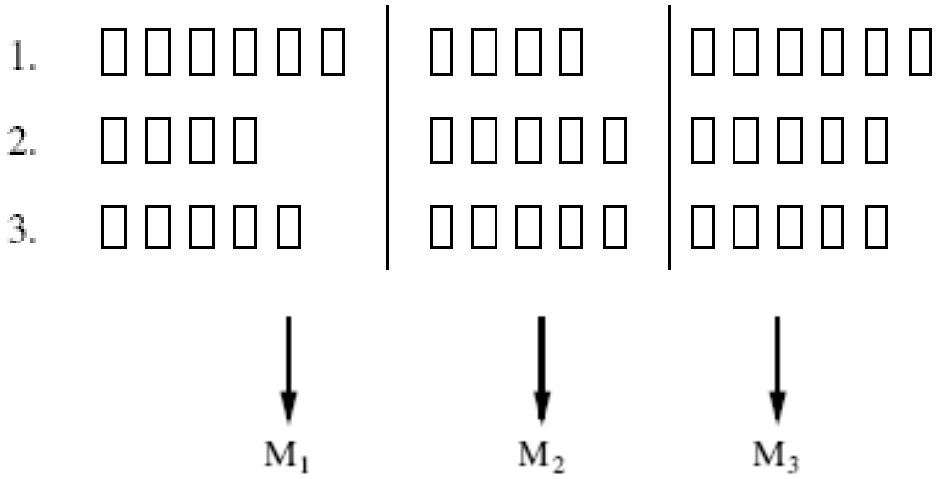




Hooola

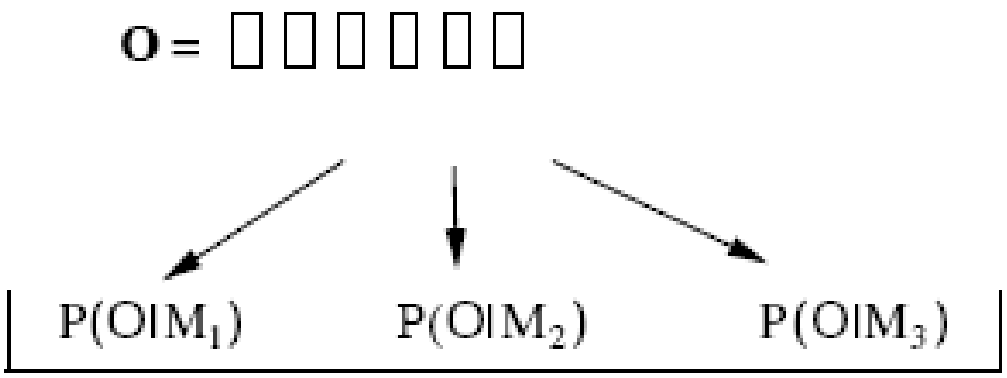
Modelos Ocultos de Markov - HMM

- Entrenamiento



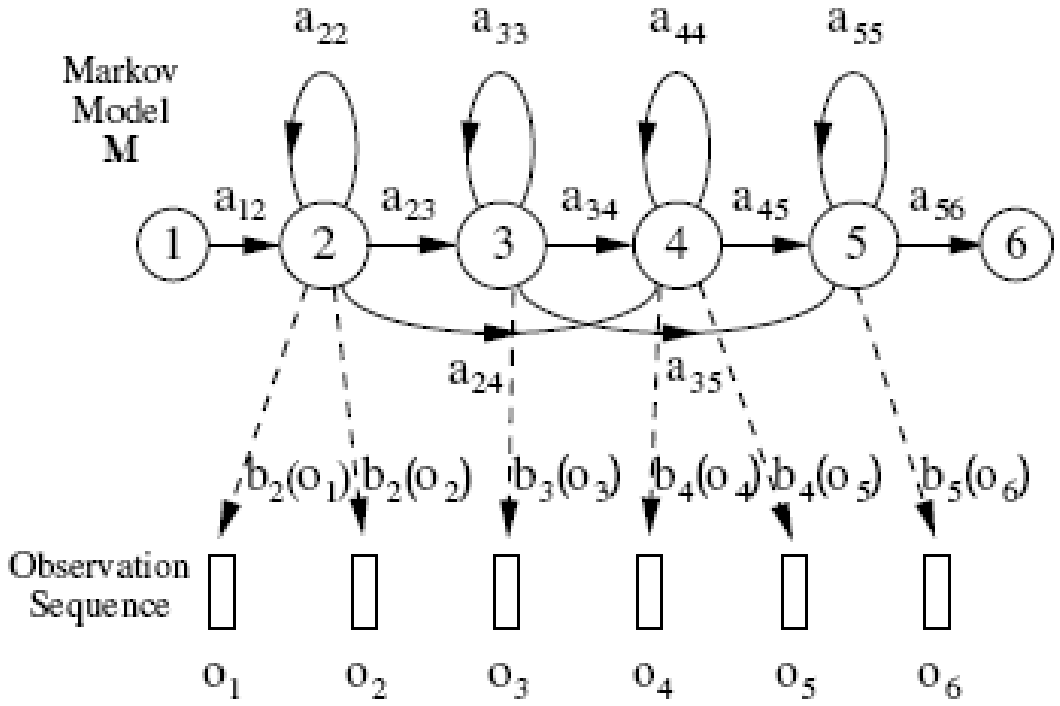
Modelos Ocultos de Markov - HMM

- Reconocimiento

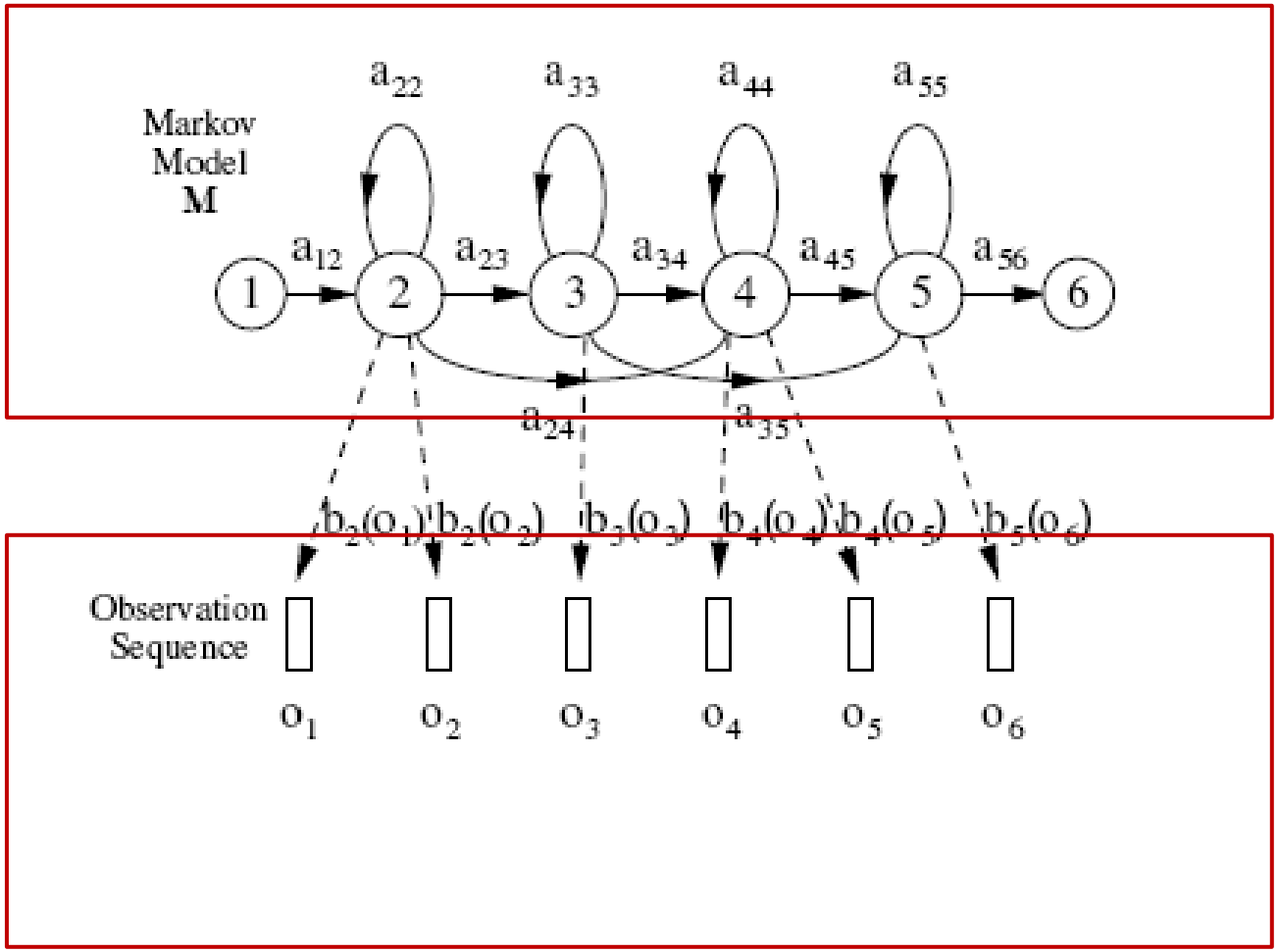


escoger el modelo con probabilidad mas alta

Modelos Ocultos de Markov - HMM



Modelos Ocultos de Markov - HMM



Parte oculta

Parte observable

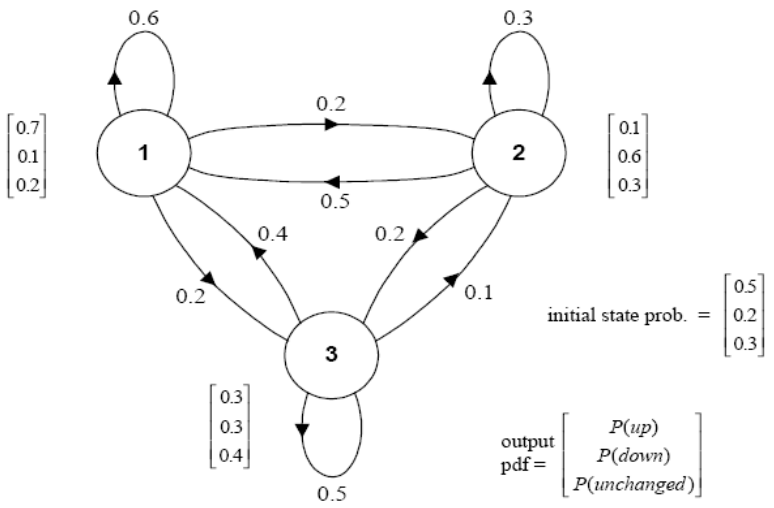
Modelos Ocultos de Markov - HMM

- **Definición HMM discreto**
  - Un HMM es una quintupla (S, V, A,B,Π), donde:

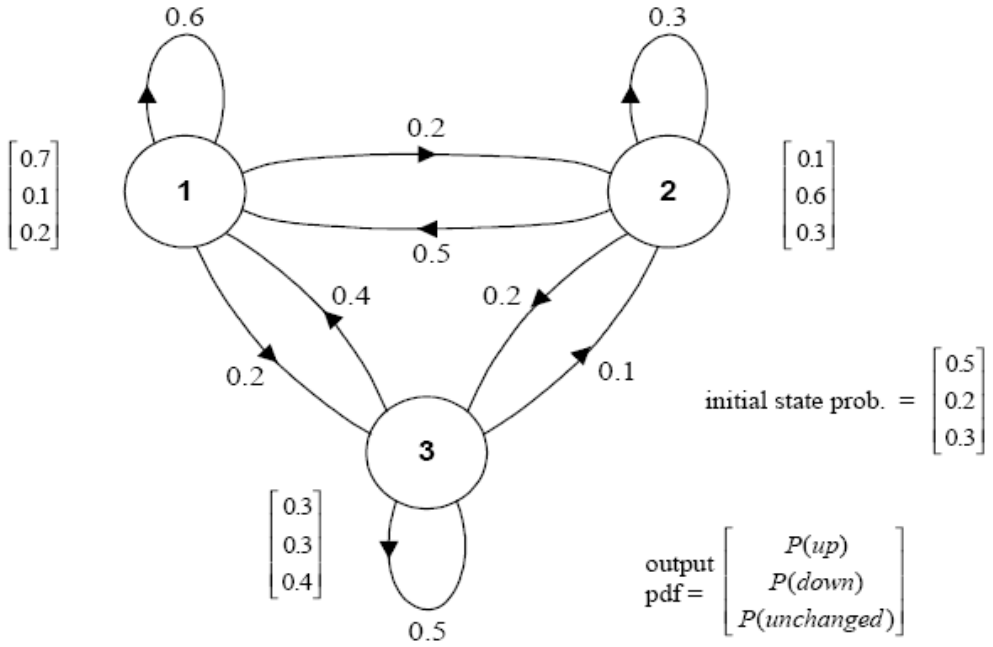
  - **S** = {Si,...SN} conjunto de N estados
  - **V**= {V1,... VK} conjunto de K símbolos de observación,
  - **A**= probabilidades de transición entre estados
  - **B** =probabilidades de los símbolos de observación
  - **Π** = distribución inicial de los estados

Modelos Ocultos de Markov - HMM

- Ejemplo
  - **S** = {1,2,3} conjunto de N estados
  - **V**= {up,down,unchangend} conjunto de K símbolos de observación,
  - **A**= probabilidades de transición entre estados
  - **B** =probabilidades de los símbolos de observación
  - **Π** = distribución inicial de los estados



Modelos Ocultos de Markov - HMM



1. Problema 1 (evaluación)
2. Problema 2 (decodificación)
3. Problema 3 (aprendizaje)

Modelos Ocultos de Markov - HMM

1. **Problema 1 (evaluación)**
   1. Calcular eficientemente ***P(O/ λ)*** ***algoritmo forward***

2. **Problema 2 (decodificación)**
   1. Escoger una correspondiente secuencia de estados ***Q = q1…qT*** que sea óptima, ***algoritmo de Viterbi***

3. **Problema 3 (aprendizaje-entrenamiento)**
   1. Ajustar los parámetros del modelo ***λ = (A,B,π)*** para maximizar ***P(O/ λ)*** ? ***-algoritmo forward-backward (Baum-Welch)***

Solución al problema I Algoritmo *forward*

1. Inicialización

$$\alpha_1(i) = \pi_i b_i(O_1), \qquad 1 \le i \le N$$

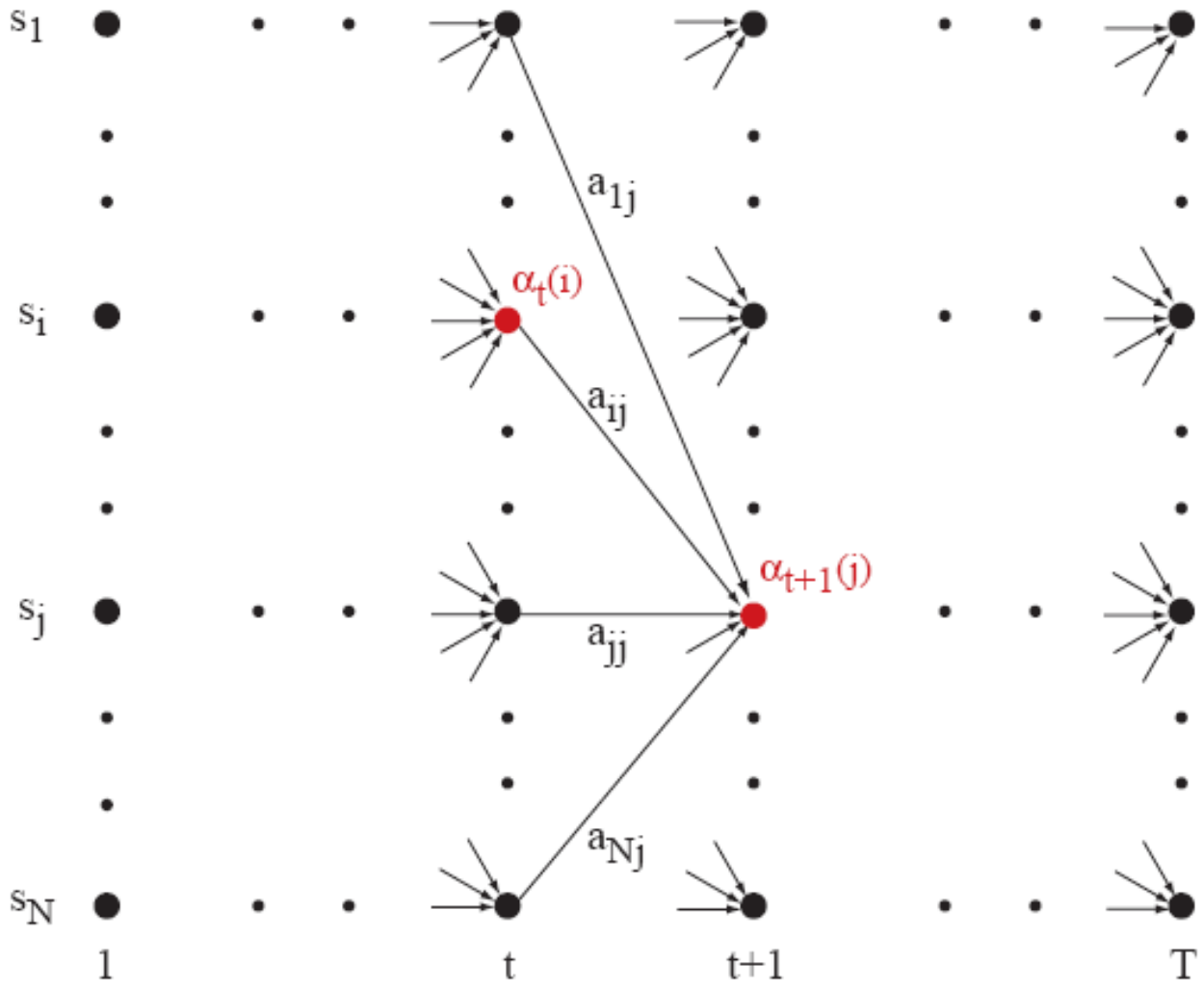
2. Inducción

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij}\right] b_j(O_{t+1}), \qquad 1 \le t \le T-1$$

$$1 \le j \le N.$$

3. Terminación

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


s_1
s_i
s_j
s_N
a_{1j}
$\alpha_t(i)$
a_{ij}
$\alpha_{t+1}(j)$
a_{jj}
a_{Nj}
1
t
t+1
T

Solución al problema II Algoritmo *Viterbi*

1. Inicialización

$$\delta_1(i) = \pi_i b_i(O_1), \qquad 1 \le i \le N$$

$$\psi_1(i) = 0.$$

2. Inducción

$$\delta_t(j) = \max_{1 \le i \le N} [\delta_{t-1}(i) a_{ij}] b_j(O_t), \qquad 2 \le t \le T$$

$$1 \le j \le N$$

$$\psi_t(j) = \operatorname*{argmax}_{1 \le i \le N} [\delta_{t-1}(i) a_{ij}], \qquad 2 \le t \le T$$

$$1 \le j \le N.$$

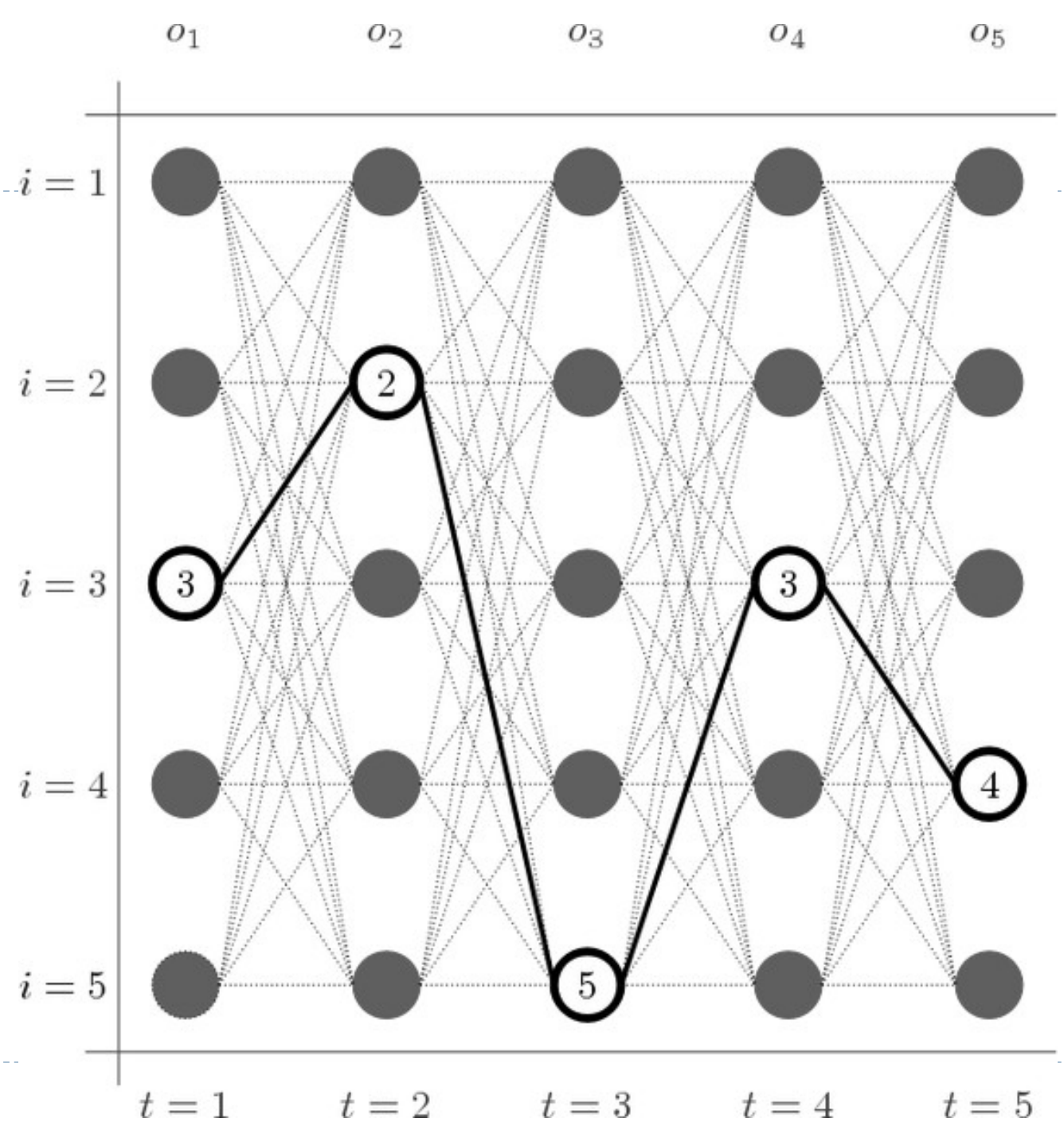
3. Terminación

$$P^* = \max_{1 \le i \le N} [\delta_T(i)]$$

$$q_T^* = \operatorname*{argmax}_{1 \le i \le N} [\delta_T(i)].$$

4. Path backtraking

$$q_t^* = \psi_{t+1}(q_{t+1}^*), \qquad t = T-1, T-2, \cdots, 1.$$


o_1
o_2
o_3
o_4
o_5
$i = 1$
$i = 2$
$i = 3$
$i = 4$
$i = 5$
2
3
3
4
5
$t = 1$
$t = 2$
$t = 3$
$t = 4$
$t = 5$

Solución al problema III Algoritmo *Baum Welch*

1. Inicializar $\lambda=(A,B,\pi)$
2. Calcular α, β, ξ, γ
   1. Donde

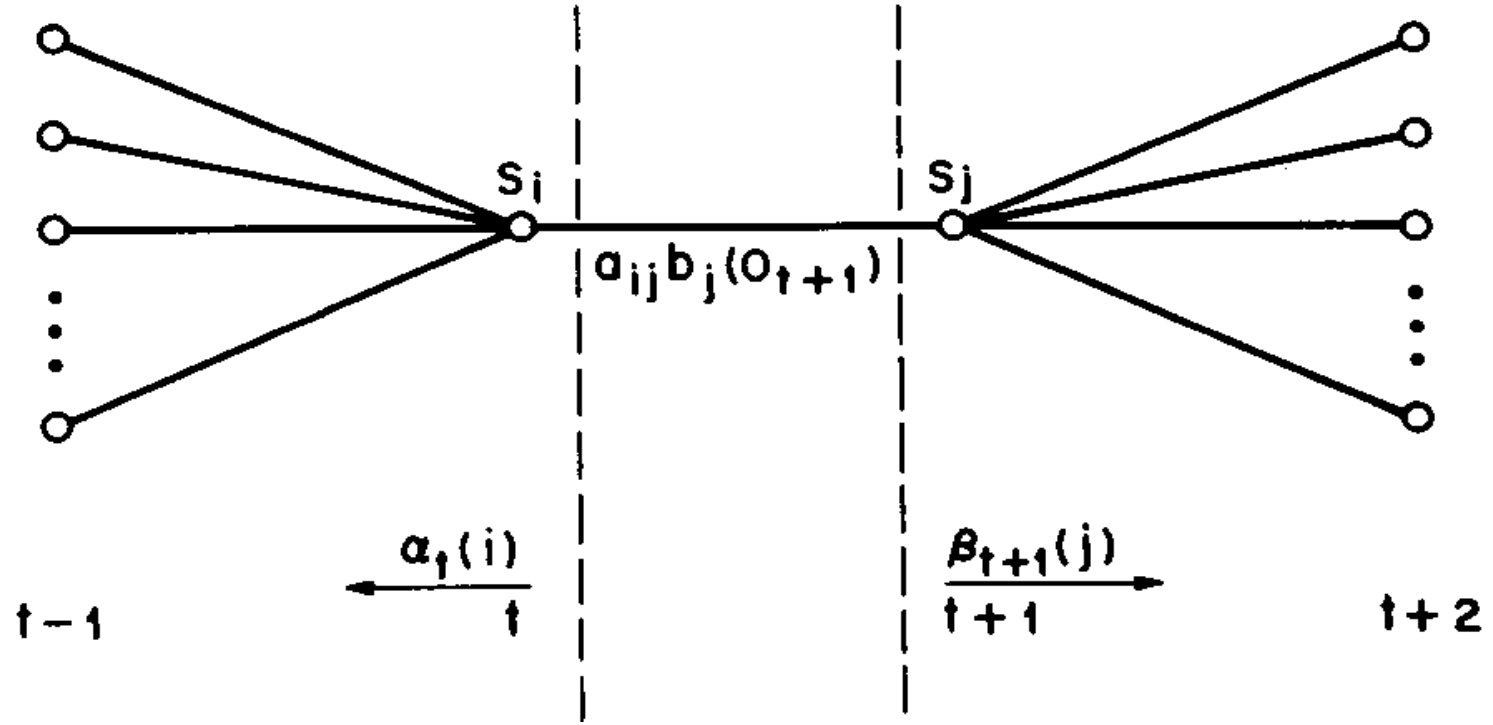
$$\xi_t(i, j) = P(q_t = S_i, q_{t+1} = S_j | O, \lambda).$$

$$\xi_t(i, j) = \frac{\alpha_t(i)\, a_{ij} b_j(O_{t+1})\, \beta_{t+1}(j)}{P(O|\lambda)}$$

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

$$\gamma_t(i) = P(q_t = S_i | O, \lambda) \qquad \gamma_t(i) = \frac{\alpha_t(i)\, \beta_t(i)}{P(O|\lambda)} = \frac{\alpha_t(i)\, \beta_t(i)}{\sum_{i=1}^{N} \alpha_t(i)\, \beta_t(i)}$$

3. Estimar nuevo $\lambda'=(A,B,\pi)$
4. Remplazar λ con λ'
5. Si no converge ir a etapa 2
6. Fin


s_i
s_j
$a_{ij} b_j(O_{t+1})$
$\alpha_t(i)$
t
$t-1$
$\beta_{t+1}(j)$
$t+1$
$t+2$

Solución al problema III Algoritmo *Baum Welch*

1. Inicializar $\lambda=(A,B,\pi)$
2. Calcular α, β, ξ, γ
3. Estimar nuevo $\lambda'=(A,B,\pi)$

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

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

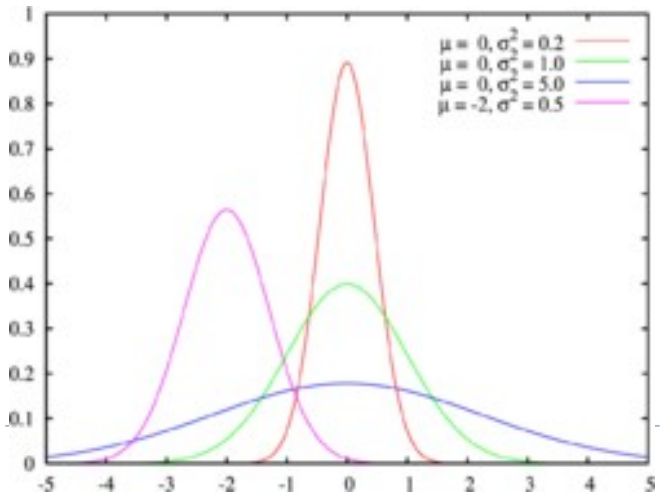
$$\bar{b}_j(k) = \frac{\sum_{t=1 \text{ s.t. } O_t = v_k}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)}.$$

4. Remplazar λ con λ'
5. Si no converge ir a etapa 2
6. Fin

Modelos continuos

- Muchos problemas reales, los más interesantes no tienen símbolos discretos si no continuos, ejemplo el habla, para esto se utiliza en lugar de símbolos discretos una función de densidad de probabilidad gener anas GMM por es

$$b_j(\mathbf{O}) = \sum_{m=1}^{M} c_{jm} \mathfrak{N}[\mathbf{O}, \boldsymbol{\mu}_{jm}, \boldsymbol{U}_{jm}],$$



Gaussian Mixture Models

- Cada gaussiana multidimesional se construye mediante la expresión

$$b_j(y) = \frac{1}{|\Sigma_j|^{1/2}(2\pi)^{K/2}} \exp\left(\frac{-(y-\mu_j)^T \Sigma_j^{-1}(y-\mu_j)}{2}\right),$$

- donde

$$\hat{\mu}_j = \frac{1}{T}\sum_{t=1}^{T} y_t ,$$

$$\hat{\Sigma}_j = \frac{1}{T}\sum_{t=1}^{T}(y_t - \hat{\mu}_j)(y_t - \hat{\mu}_j)^T .$$

Gaussian Mixture Models

- Para todos los patrones de entrenamiento se tiene:

$$\bar{\mu}_j = \frac{\sum_{e=1}^{E}\sum_{t=1}^{T_e}\gamma_j(t,e)\,y_{te}}{\sum_{e=1}^{E}\sum_{t=1}^{T_e}\gamma_j(t,e)},$$

$$\bar{\Sigma}_j = \frac{\sum_{e=1}^{E}\sum_{t=1}^{T_e}\gamma_j(t,e)\,(y_{te}-\bar{\mu}_j)(y_{te}-\bar{\mu}_j)^T}{\sum_{e=1}^{E}\sum_{t=1}^{T_e}\gamma_j(t,e)},$$

Gaussian Mixture Models

- Finalmente se construye un GMM (aca se tiene k=M)

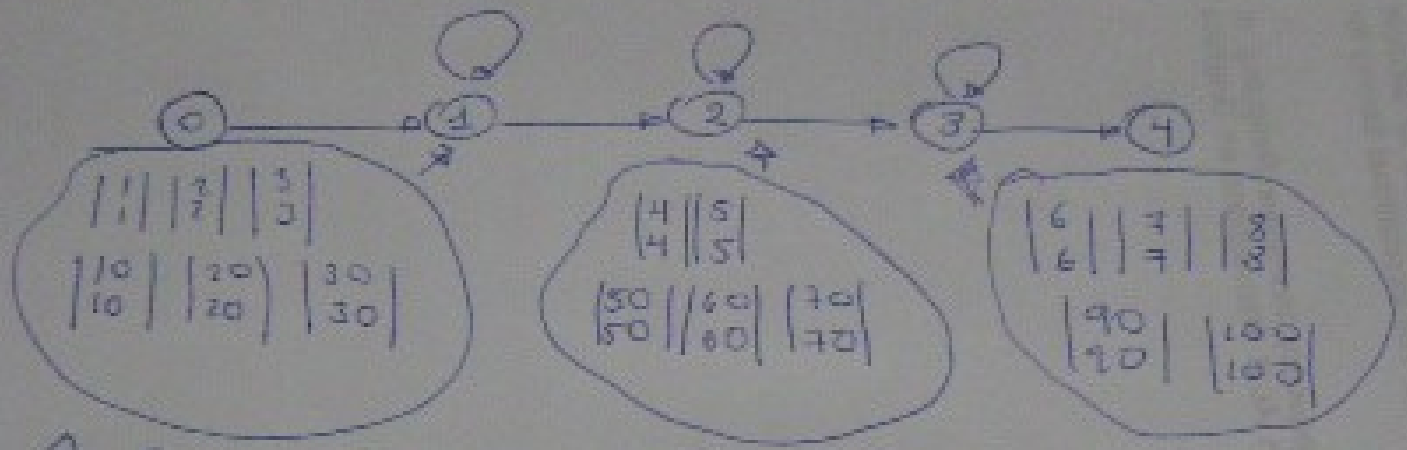
$$b_j(\boldsymbol{y}) = \sum_{m=1}^{M} c_{jm} b_{jm}(\boldsymbol{y}) ,$$

donde $$b_{jm}(\boldsymbol{y}) = N\left(\boldsymbol{y}; \boldsymbol{\mu}_{jm}, \boldsymbol{\Sigma}_{jm}\right) .$$

$$\sum_{m=1}^{M} c_{jm} = 1 .$$

y $$\bar{c}_{jm} = \frac{n_{jm}}{n_j} ,$$

HMM



Gaussiana 0

$\mu = \begin{vmatrix} 30 \\ 30 \end{vmatrix}$

$\Sigma = \begin{vmatrix} 100 & 0 \\ 0 & 100 \end{vmatrix}$

$c_1 = 0.42$

Gaussiana 1

$\mu \quad \begin{vmatrix} 4 \\ 4 \end{vmatrix}$

$\Sigma \quad \begin{vmatrix} 16.7 & 0 \\ 0 & 16.7 \end{vmatrix}$

$c_2 = 0.57$

Gaussiana 0

$\mu = \begin{vmatrix} 60 \\ 60 \end{vmatrix}$

$\Sigma = \begin{vmatrix} 100 & 0 \\ 0 & 100 \end{vmatrix}$

$c_1 = 0.6$

Gaussiana 1

$\mu = \begin{vmatrix} 4.5 \\ 4.5 \end{vmatrix}$

$\Sigma = \begin{vmatrix} 0.5 & 0 \\ 0 & 0.5 \end{vmatrix}$

$c_2 = 0.4$

Gaussiana 0

$\mu = \begin{vmatrix} 85 \\ 85 \end{vmatrix}$

$\Sigma = \begin{vmatrix} 0.5 & 0 \\ 0 & 0.5 \end{vmatrix}$

$c_1 = 0.4$

Gaussiana 1

$\mu = \begin{vmatrix} 90 \\ 90 \end{vmatrix}$

$\Sigma = \begin{vmatrix} 100 & 0 \\ 0 & 100 \end{vmatrix}$

$c_2 = 0.6$

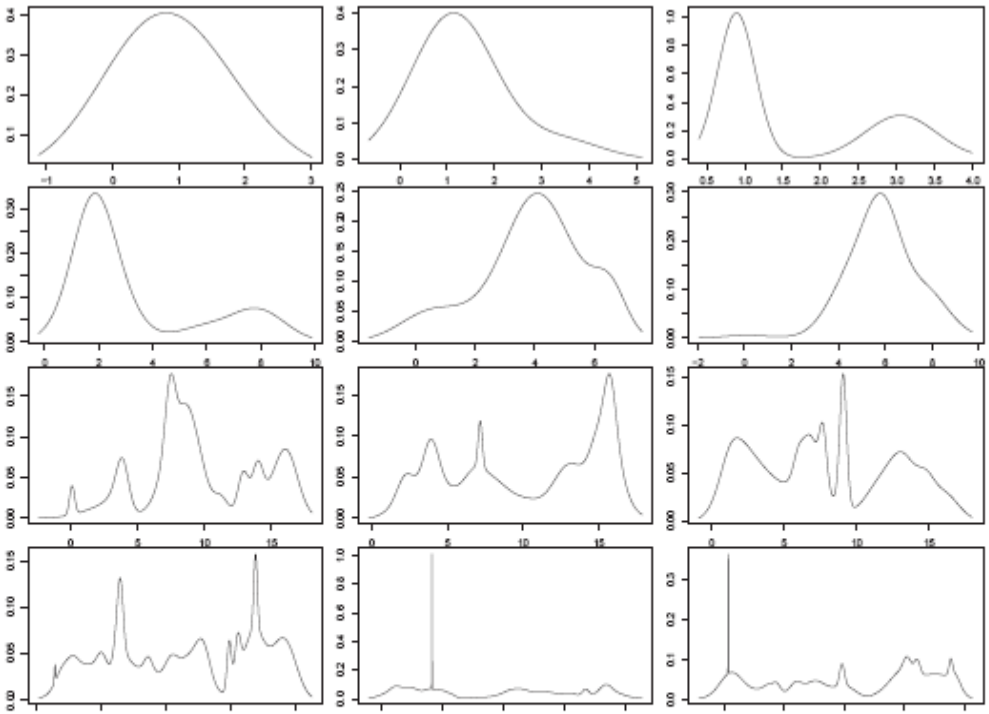
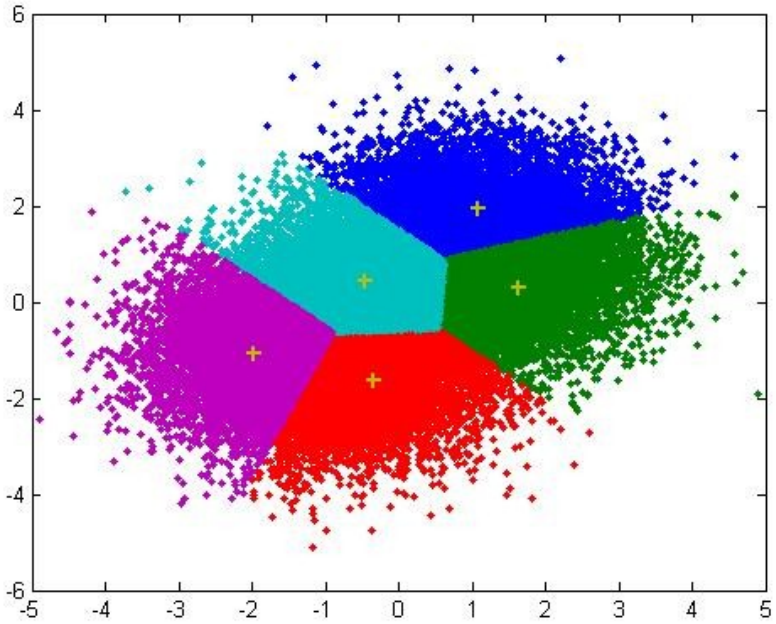
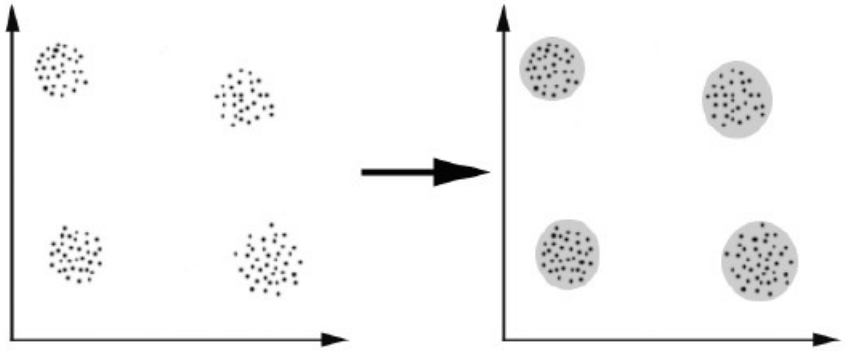


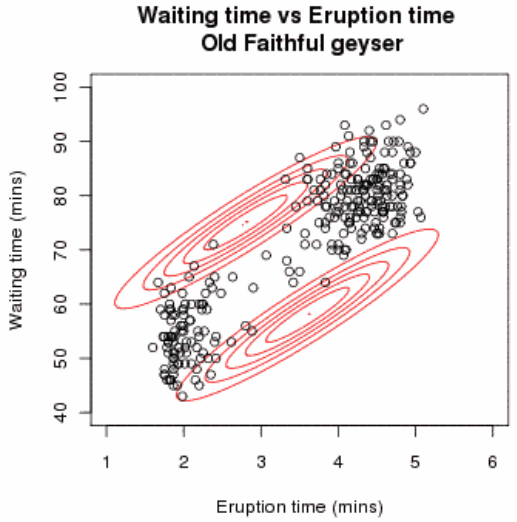
FIGURE 1. Some normal mixture densities for $K = 2$ *(first row)*, $K = 5$ *(second row)*, $K = 25$ *(third row)* and $K = 50$ *(last row)*.

Inicialización de un HMM

- Generalmente se usa K-means, algoritmo EM, o una red neuronal para inicializar los valores







Criterio de Implementación

Criterios de implementación

- Los modelos ocultos de Markov no se puede implementar directamente como dice la teoría, a menos que sean modelos con muy pocos estados

- **Problema**
  - Los valores del modelo son valores menores a 1 y las formulas anteriores involucran muchas multiplicaciones y usualmente los valores resultantes son mas pequeños que el número en punto flotante mas pequeño representable en la computadora

- Criterios
  - Escalamiento
  - Usando Logaritmos

Implementación usando logaritmos de las probabilidades

- Todas las formulas anteriores tienen que expresarse de manera logarítmica, por ejemplo

- En Vite

$$\hat{\alpha}_j(t) = \max_{\text{over } i} \left(\hat{\alpha}_i(t-1) a_{ij} \right) b_j(\boldsymbol{y}_t) \quad \text{for } 1 < t \leq T \ .$$

a

$$\hat{\alpha}_j^L(t) = \max_{\text{over } i} \left(\hat{\alpha}_i^L(t-1) + a_{ij}^L \right) + b_j^L(\boldsymbol{y}_t) \ ,$$

Implementación usando logaritmos de las probabilidades

- Todas las formulas anteriores tienen que expresarse de manera logarítmica, por ejemplo

- Cada gaussiana multidimensional pasa de:

$$b_j(\boldsymbol{y}) = \frac{1}{|\boldsymbol{\Sigma}_j|^{1/2}(2\pi)^{K/2}} \exp\left(\frac{-(\boldsymbol{y}-\boldsymbol{\mu}_j)^T \boldsymbol{\Sigma}_j^{-1}(\boldsymbol{y}-\boldsymbol{\mu}_j)}{2}\right),$$

a

$$b_j^L(\boldsymbol{y}_t) = \log(b_j(\boldsymbol{y}_t)) = -\frac{K}{2}\log(2\pi) - \sum_{k=1}^{K}\log(\sigma_{jk}) - \frac{1}{2}\sum_{k=1}^{K}\left(\frac{y_{kt}-\mu_{jk}}{\sigma_{jk}}\right)^2$$

Implementación usando logaritmos de las probabilidades

- Todo lo anterior es sencillo, pero como hacemos si tenemos que calcular

  Log(A+B) ?

  ó

  Log(A+B+…+N)?

recordar que todos los datos lo tenemos en forma logarítmica, se necesita una función donde

$$\log(A + B + \ldots + N) = F^*(\text{Log}(A) + \text{Log}(B) + \ldots + \log(N))$$

Por ejemplo el algoritmo forward requiere una operación de este tipo, así c

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij}\right] b_j(O_{t+1}), \qquad 1 \le t \le T-1$$

$$1 \le j \le N.$$

Implementación usando logaritmos de las probabilidades

- Solución para log(A+B)
  - Ordenar de tal manera que A>B
  - Reescribir

$$\log(A+B) = \log(A(1+B/A)) = \log(A) + \log(1+B/A)\,.$$

  - Si la diferencia entre B y A es mas grande que el numero mas pequeño representable enl a computadora, entonces

$$\log(A+B) = \log(A)$$

Implementación usando logaritmos de las probabilidades

// calcula log(A+B) $\log(A+B) = \log(A(1+B/A)) = \log(A) + \log(1+B/A)\,.$

Algoritmo f*(*log(A)*,log(B))

```
1  if log(B) > log(A)
3        then exchange log(A) ↔ log(B)
5  C ← log(B) – log(A) //calcula log(B/A)
7  if C ≤ numero mas pequeño representable en la computadora
9        then C ← 0
11 else
13    C ← log(1+eC)
15 return log(A) + C
```

Implementación usando logaritmos de las probabilidades

- Y como calculamos log(A+B+…+N)?

Algoritmo sumLog(*log(A)*,log(B),…, log(N))

1 *A* ← **Quicksort(***log(A)*,log(B),…, log(N)**)**

9 **return** *h*(A , A.length)*

Algoritmo h*(*A, n*)

3 **if** *A.length* ==2

5 **then return** f*(A[0], A[1])

7 **else**

8 C ← *h*(A, n-1)*

9 **return** *C+log(1+e(A[A.length-1] – C)*

Implementación usando logaritmos de las probabilidades

- **Análisis del algoritmo**
- El método de ordenación Quicksort

  $\Theta(n \log n)$
- El procedimiento h*

  $T(n)=T(n-1)+\Theta(1)$

  $es\ \Theta(n)$
- El procedimiento completo

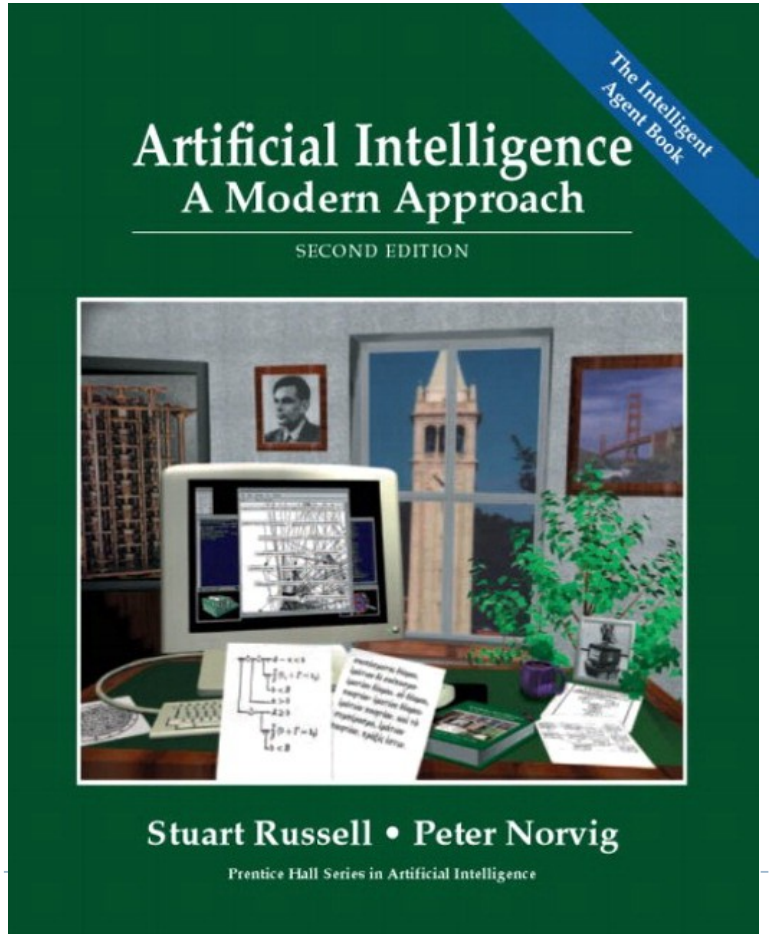
  $\Theta(n \log n) + \Theta(n)$
- La complejidad final es $\Theta(n \log n)$

Implementación usando logaritmos de las probabilidades

- **Correctitud**
  - Para probar la correctitud (h*)se define el siguiente invariante del bucle teniendo en cuenta la variable **C que contiene log(A+B+…+N-1)**
  - Inicializacion
    - C al llegar antes de tomar el primer valor de caso base tiene log(Θ)
  - Mantenimiento
    - C luego de devolver los valores tendrá log(A+B) y en cada sucesiva llamada tendrá log(A+B+C) , log(A+B+C+D), y así sucesivamente
  - Terminación
    - Al final del procedimiento C tiene log(A+B+C+D+…+N-1),

Referencias

- Por donde empezar?
  - Capitulo 15



Referencias

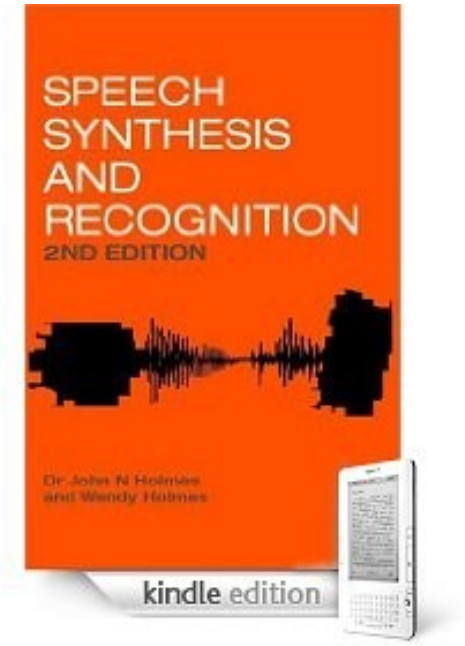
- Por donde empezar?

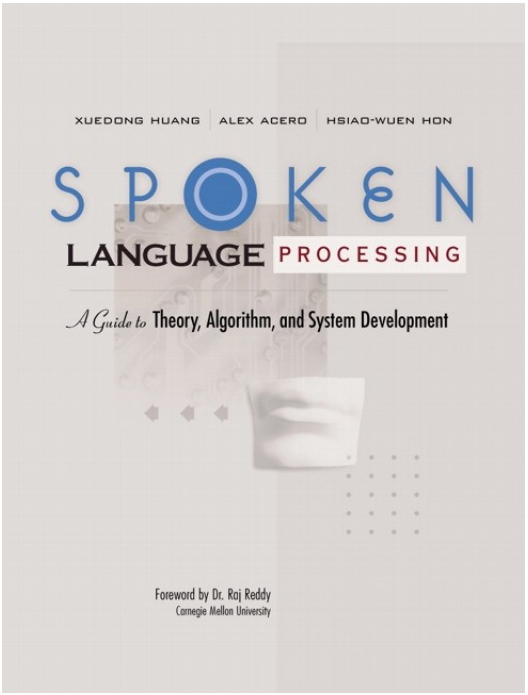
A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition

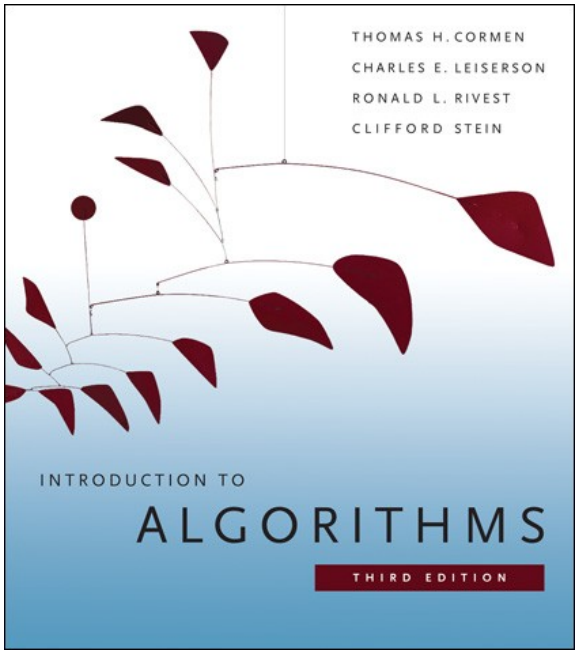
LAWRENCE R. RABINER, FELLOW, IEEE

Referencias

- Por donde empezar?







alumnos 2007

- Proyectos realizados
  - *Extracción de características de la señal de voz utilizando LPC-Cepstrum - Jorge Velarde, Jhon Franko, Pretel Jesús, Alicia Isolina*
  - *Prediccion Lineal Perceptual PLP - Alan Alfredo Collantes Arana Dany Richard Sari Bustos*
  - *Audio files compression through wavelets - Fredy Carranza-Athó*
  - *Máquinas de Sopoerte Vectorial en el Reconocimiento Automático del Habla - Juan Carlos Federico Roeder Moreno*
  - *Efectos de las diferentes transformadas del coseno en RAH Márquez Fernández, Luz Victoria*

alumnos 2008

- **Proyectos realizados**
  - **Extracción de características de palabras aisladas usando MFCC y MFCC con pesos, Nils Murrugarra Llerena**
  - **Reconocimiento automático de palabras aisladas mediante el uso de los extractores de características: MFCC y MODGDF, Jorge Valverde Rebaza**
  - **Uso del método de extracción de características MFCC con formas arbitrarias a nivel de filtros para el reconocimiento de palabras aisladas, Pedro Shiguihara Juárez**
  - **Algoritmo N-Best: Eficiente procedimiento para la búsqueda de las N hipotesis de frases más probables, Luis Mostacero Zárate**
  - **Predicción y Entropia de Textos en Inglés, Juan Grados Vásquez**
  - **Aplicación del algoritmo MFCC-DTW en el reconocimiento de comandos activados por voz, Pedro Linares Kcomt**

http://eventos.seccperu.org/sraa2009/index.html

nculos | A Hybrid ANN-HM... | Jahmm - An imple... | Wall Street Journal (... | torrents.to search fo... | postgrado | www.977music.com | MoviStar:::..

SRAA 2009 Trujillo

I Evento de Speech Recognition: Algoritmos y Aplicaciones - Junio 29

search...

Menú Principal

- Presentación
- Objetivos
- Conferencistas
- Conferencias
- Organización
- Inscripciones
- Enlazar
- Exposiciones

Presentación SRAA 2009

I Evento de Reconocimiento del Habla 2009

El 29 de Junio del 2009 se realizará el **I Evento en Speech Recognition** en el norte del país, una multiconferencia llevada a cabo en la *Sala de Conferencias de la Universidad Privada del Norte*, Trujillo, Perú. El evento presentará investigaciones realizadas en esta área y la necesidad de mostrar cuál es su importancia e influencia sobre nuestra sociedad actual. La Sociedad de Estudiantes de Ciencia de la Computación hace presente esta invitación a todos los universitarios, profesionales, y comunidad estudiantil en general.

El I Evento en Speech Recognition se realizará en la Sala de Conferencias de la Universidad Privada del Norte, sito en Pabellón A - 1º Piso, Av. Del Ejército 920 - Urb. El Molino - Trujillo. El ingreso es totalmente libre. El pago por solicitud de certificado es de S/10.00 recaudados al inicio del evento.

RAH según Wikipedia

El Reconocimiento Automático del Habla (RAH) o Reconocimiento Automático de voz es una parte de la Inteligencia Artificial que tiene como objetivo permitir la comunicación hablada entre seres humanos y computadoras electrónicas. El problema que se plantea en un sistema de RAH es el de hacer cooperar un conjunto de informaciones que provienen de diversas fuentes de conocimiento (acústica, fonética, fonológica, léxica, sintáctica, semántica y pragmática), en presencia de ambigüedades, incertidumbres y errores inevitables para llegar a obtener una interpretación aceptable del mensaje acústico recibido.

Conferencistas del SRAA 2009

Jorge Guevara Diaz

- Mg. Ciencias de la Computacion
- Universidad Nacional de Trujillo

Luis Miguel Mostacero Zárate

- BSc. Ciencias de la Computacion
- Universidad Nacional de Trujillo

Jorge Valverde Rebaza

- BSc. Ciencias de la Computacion
- Universidad Nacional de Trujillo

Nils Murrugara Llerena

- BSc. Ciencias de la Computacion
- Universidad Nacional de Trujillo

Pedro Shiguihara Juárez

- BSc. Ciencias de la Computacion
- Universidad Nacional de Trujillo

Juan Grados Vasquez

- BSc. Ciencias de la Computacion
- Universidad Nacional de Trujillo

Pedro Linares Kcomt

- BSc. Ciencias de la Computacion
- Universidad Nacional de Trujillo

Clusterización para la inicialización de HMM en un ASR	Mg. Jorge Guevara Diaz
Extracción de características de palabras aisladas usando MFCC y MFCC con pesos	BSc. Nils Murrugarr Llerena
Reconocimiento automático de palabras aisladas mediante el uso de los extractores de características: MFCC y MODGDF	BSc. Jorge Valverde Rebaza
Uso del método de extracción de características MFCC con formas arbitrarias a nivel de filtros para el reconocimiento de palabras aisladas	BSc. Pedro Shiguihara Juárez
Algoritmo N-Best: Eficiente procedimiento para la búsqueda de las N hipotesis de frases más probables	BSc. Luis Mostacerc Zárate
Predicción y Entropia de Textos en Inglés	BSc. Juan Grados Vásquez
Aplicación del algoritmo MFCC-DTW en el reconocimiento de comandos activados por voz	BSc. Pedro Linares Kcomt
Domótica con Speech Recognition	Roger Castañeda y Diego Castañeda

GRACIAS!!!