

# Mapas auto-organizativos

Francisco José Ribadas Pena

INTELIGENCIA ARTIFICIAL

5º Informática

ribadas@uvigo.es

14 de diciembre de 2005

# Aprendizaje no supervisado y aprendizaje competitivo

## Aprendizaje no supervisado

- Los algoritmos de aprendizaje no supervisado no necesitan de un *supervisor externo* que juzgue (a priori o sobre la marcha) los resultados del proceso de aprendizaje.
  - No se presentan las salidas objetivo que se quieren asociar al patrón de entrada
  - Los algoritmos de aprendizaje solo manejan patrones de entrada
- Se pretende que la red descubra por si misma rasgos comunes, regularidades, correlaciones o categorías en los datos de entrada y los incorpora a su estructura interna de conexiones (pesos).
  - Se dice que la red se auto-organiza
  - La unica info. que se usa son las similitudes y diferencias entre las entradas
  - Este tipo de aprendizaje exige que en los datos de entrada exista cierta "redundancia" para poder identificar esas regularidades.
- Regla de Hebb: primera aproximación al aprendizaje no supervisado (sin info. externa)
  - Basado en evidencias fisiológicas
  - *Cuando una axon de una célula A está lo suficientemente cerca para excitar a una célula B, y toma parte repetidamente en el proceso de disparo de dicha célula, se produce un cambio metabólico en una o ambas células, que hace que la eficacia con la que A dispara a B se vea incrementada.*
  - De forma abreviada: *cuando una neurona activa a otra, la sinapsis entre ambas queda reforzada*
  - No depende de factores externos, las células simplemente se influyen unas a otras.
  - Múltiples formas de traducir esta idea a mecanismos de ajuste de pesos

## Aprendizaje competitivo

- Familia de modelos de RNA que soportan aprendizaje supervisado
  - Normalmente estructuradas en 2 capas:  $\left\{ \begin{array}{l} \text{capa de entrada} \\ \text{capa de competicion} \end{array} \right.$
- **Objetivo:** aprender a categorizar/ agrupar los datos de entrada
  - Se persigue que datos parecidos hagan reaccionar a las mismas neuronas
  - Se consigue haciendo que cada neurona se especialice en determinado "tipo" de patrones de entrada
  - Las neuronas juegan el papel de "prototipos" de los datos de entrada
- **Idea:** Para cada patrón de entrada se restringe la actualización de pesos **sólo** a la/las neuronas de la capa de competicion cuyo grado de activación haya sido más alto
  - Neuronas ganadoras = neuronas con "mayor" nivel de activación
  - Neuronas ganadoras se refuerzan a si mismas (opcionalmente, tb. a sus vecinas)
- Las neuronas compiten por la entrada
- La neurona ganadora (junto con sus pesos) representa al prototipo que se le asigna al dato de entrada

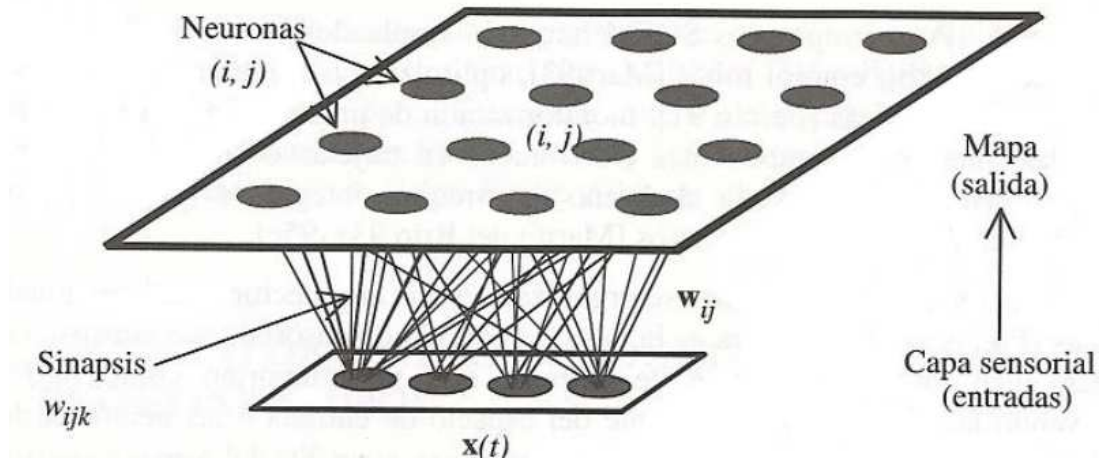
# Mapas auto-organizativos

## Fundamentos

- En la corteza cerebral existen zonas especializadas en ciertas tareas
  - Neuronas asociadas a características similares son vecinas
  - Se generan "mapas" donde se "ordenan" las características de los estímulos recibidos
- Teuvo Kohonen(1982) diseñó un modelo de red competitiva que emula esta idea
  - Los SOM (*self-organizing map*) reciben un conjunto de datos y construyen una representación (mapa neuronal) de menor dimensión
  - Neuronas asociadas con vectores similares tendrán pesos similares
- Características generales de los SOM
  - Los pesos sinápticos son representativos de determinados tipos de patrones de entrada
  - Los patrones de entrada son presentados a todas las neuronas simultáneamente
  - Uso de aprendizaje competitivo: solo la neurona con mejor respuesta es tomada en cuenta
  - Ajuste de pesos basado en los conceptos de auto-refuerzo y vecindad

## Estructura de los SOM de Kohonen

- Red neuronal de 2 capas:  $\left\{ \begin{array}{l} \text{capa de entrada} \\ \text{capa de competicion} \end{array} \right.$



- Capa de entrada recibe la señal de entrada a la red (no hay procesamiento)
  - Su dimension,  $n$ , depende del problema
  - Sera un vector  $\vec{X} = (x_1, x_2, \dots, x_n)$
- Capa de competicion: formada por  $m$  neuronas
  - Cada neurona de competicion esta conectada con todas la neuronas de entrada
  - Los pesos de la neurona  $i$  formaran un vector de  $n$  dimensiones:  $\vec{W}_i = (w_{1i}, w_{2i}, \dots, w_{ni})$
  - No hay conexion entre las neuronas de competicion
  - Si existe una relacion de *vecindad* usada en el aprendizaje

## Funcionamiento de los SOM de Kohonen

- En el modo de operación **normal** permanecen fijos los pesos.
  - Cada neurona  $i$  calcula la similitud entre el vector de entrada  $\vec{X}$  y su vector de pesos  $\vec{W}_i$
  - Vence aquella con mayor similitud.
- En la fase de **aprendizaje**, la neurona vencedora ajusta sus pesos aproximándose cada vez más a los de  $\vec{X}$ .
  - Por la función de vecindad también actualizan sus pesos neuronas vecinas a la vencedora

# Aprendizaje en los mapas auto-organizativos (I)

- Inicialmente los pesos de cada neurona se establecen aleatoriamente
- Durante entrenamiento, se elige al azar un vector de entrada con el que se realizan 2 tareas
  - determinar neurona ganadora
  - modificacion de pesos

## ■ (1) Selección de la neurona ganadora

- Al recibir un patron  $\vec{X}$  cada neurona compara su vector de pesos con el vector de entrada
  - distintas funciones de distancia posible
  - la mas usual es la *distancia euclidea*

$$d(\vec{X}, \vec{W}_i) = \sqrt{\sum_{j=1}^n (x_j - w_{ji})^2} \quad \forall 1 \leq i \leq m$$

- La única neurona ganadora es aquella con los pesos mas parecidos al patron
  - solo se activara la neurona cuya distancia sea la menor

$$g(\text{neurona}_i) = \begin{cases} 1 & \text{si } d(\vec{X}(t), \vec{W}_i) = \min_k \{d(\vec{X}, \vec{W}_k)\} \\ 0 & \text{en otro caso} \end{cases}$$

## ■ (2) Ajuste de pesos

- Solo se realizara ajuste de pesos en la neurona ganadora y sus "vecinas"
- La idea es que dicha neurona se "especialice" en patrones similares
- Se ajustan los pesos para hacerlos mas parecidos al patron de entrada que provoco la activacion

**Idea base:** acercar vector de pesos al vector de entrada
- Ajuste de pesos controlado por dos parametros que varian con el tiempo (num. patrones procesados)
  - funcion de vecindad y amplitud del "vecindario"
  - tasa de aprendizaje

# Aprendizaje en los mapas auto-organizativos (II)

## Ajuste de pesos

- Nuevos pesos despues de la iteracion  $t$  para la neurona  $i$

$$\vec{W}_i^{t+1} = \vec{W}_i^t + \alpha(t+1) H_g(t, |i - G|) [\vec{X} - \vec{W}_i^t]$$

- Con  $\alpha(t+1)$  el valor de la tasa de aprendizaje para la presente iteracion
  - Se va reduciendo con el tiempo (una función determina como decrece a medida que aumentan las iteraciones)
  - Al principio: valores grandes, provocan cambios relativamente bruscos en la organiz. de las neuronas
  - Al final: valores casi nulos, para que la red se estabilice y converga
- Con  $H_g(t, |i - G|)$  una media que indica el grado de vecindad entre la neurona  $i$  y la neurona ganadora  $G$ 
  - Valdra 1 para la neurona ganadora  $G$  y 0 para las que no esten en su vecindad
  - Su valor se va haciendo menor a medida que se reduce la vecindad con  $G$
  - Su efecto tb. disminuye con el tiempo (a medida que aumentan las iteraciones)

**Nota:**  $|i - G|$  denota la distancia entre las neuronas  $i$  y  $G$  en la capa competitiva en función del tipo de vecindad considerado

- **Idea base:** acercar los pesos al patron de entrada
  - El nuevo peso es el resultado de sumar al antiguo una fraccion (determinada por  $\alpha$  y  $H_g$ ) de la diferencia entre el peso antiguo y su componente correspondiente del vector de entrada.



# Esquema general del algoritmo de aprendizaje en SOM

1. Inicializar pesos
  - Asignar a los pesos valores aleatorios pequeños
2. Presentar una entrada
  - El conjunto de aprendizaje se presenta repetidas veces hasta llegar a la convergencia de la red
  - Actualizar  $\alpha$  (reducir su valor)
3. Propagar el patron de entrada hasta la capa de competicion
  - Obtener los valores de salida (distancias) de las neuronas de dicha capa
4. Seleccionar la neurona ganadora  $G$ 
  - La de menor distancia al patron
5. Actualizar conexiones entre capa de entrada y la neurona  $C$ 
  - Actualizar tambien los pesos de sus vecinas segun el grado de vecindad
6. Si  $\alpha$  se mantiene por encima del umbral de parada, volver a 2, en caso contrario FIN

# Aplicaciones típicas

Tipos de problemas que pueden resolver los SOM

- Condicionará como se interprete lo que representa la salida de un SOM

**Agrupamiento (*clustering*)** A partir de un conjunto de entrada se desea determinarse si se puede dividir ese conjunto en diferentes clases.

- Permite determinar qué clases existen
- Permite decidir a qué clase pertenece cada dato de entrada  
→ determinando su neurona ganadora
- Permite caracterizar cada una de esas clases  
→ mediante los pesos de cada neurona ( $\approx$  clase)

**Prototipado** Similar al anterior, en lugar de conocer la clase del dato, interesa obtener un prototipo de la clase a la que pertenece.

- Usa los pesos de la ganadora para determinar ese prototipo

**Codificación** Se obtiene a la salida de la red una versión codificada del dato de entrada.

- Se busca una salida de menor dimensión de la entrada

**Análisis de componentes principales** Se trata de detectar qué vectores de conj. de entrada caracterizan en mayor grado ese conjunto de datos.

- Los demás vectores podrán eliminarse sin una pérdida significativa de info

**Extracción y relación de características** Se pretende organizar los vectores de entrada en un mapa topológico

- A partir de la red entrenada, patrones parecidos producirán respuestas similares en neuronas cercanas
- Si existe una organización global de patrones de entrada, se verá reflejada en la salida de la red.
- Se ubican entradas parecidas y/o relacionadas en zonas próximas de la red