Sistemas Autoorganizados I

Modelos de la computación (Sistemas Autoorganizados)

Francisco Fernández Navarro

Departamento de Lenguajes y Ciencias de la Computación
Área: Ciencias de la Computación e Inteligencia Artificial

Índice de contenidos

1. Sistemas Autoorganizados
   - Introducción
   - Redes Neuronales Competitivas no supervisadas

Introducción

- Los primeros trabajos de Von der Malsburg (1973) sobre la autoorganización de las células nerviosas en la corteza cerebral sirvió como punto de partida para comprender que la competencia entre neuronas permite que cada una de ellas se especialice en diferentes entradas.
- En 1975, Fukushima propuso el cognitron, que es una red competitiva de múltiples capas y autoorganizada. Willshaw y von der Malsburg (1976) trabajaron en la formación de conexiones neuronales a través de la autoorganización y Grossberg (1972, 1976) en la clasificación adaptativa de patrones.
- Modelos más cercanos a la neurobiología. Arquitectura simple: en general, redes de una sola capa con procesamiento hacia adelante.

Introducción

¿Qué son los sistemas autoorganizados?

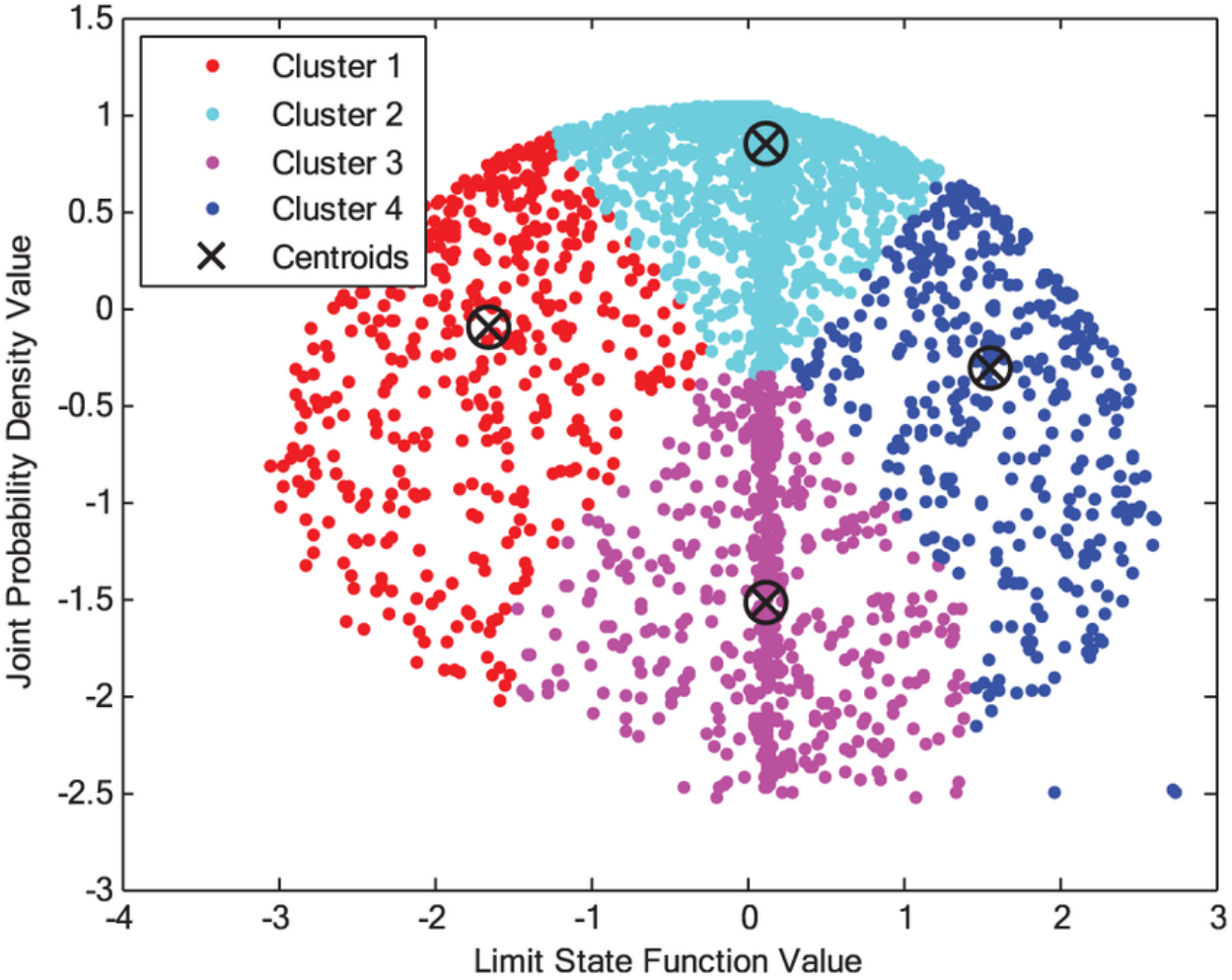
- La red aprende por sí misma (autoorganización) una representación de los patrones de entrenamiento (prototipo).
- En cada iteración hay una única neurona ganadora, el resto son "perdedoras" de ahí el adjetivo "competitivo".
- Suelen ser arquitecturas relativamente simples, por lo tanto es habitual que requieran tiempos de procesamiento cortos (si lo comparamos, por ejemplo, con las redes multicapa).

Introducción

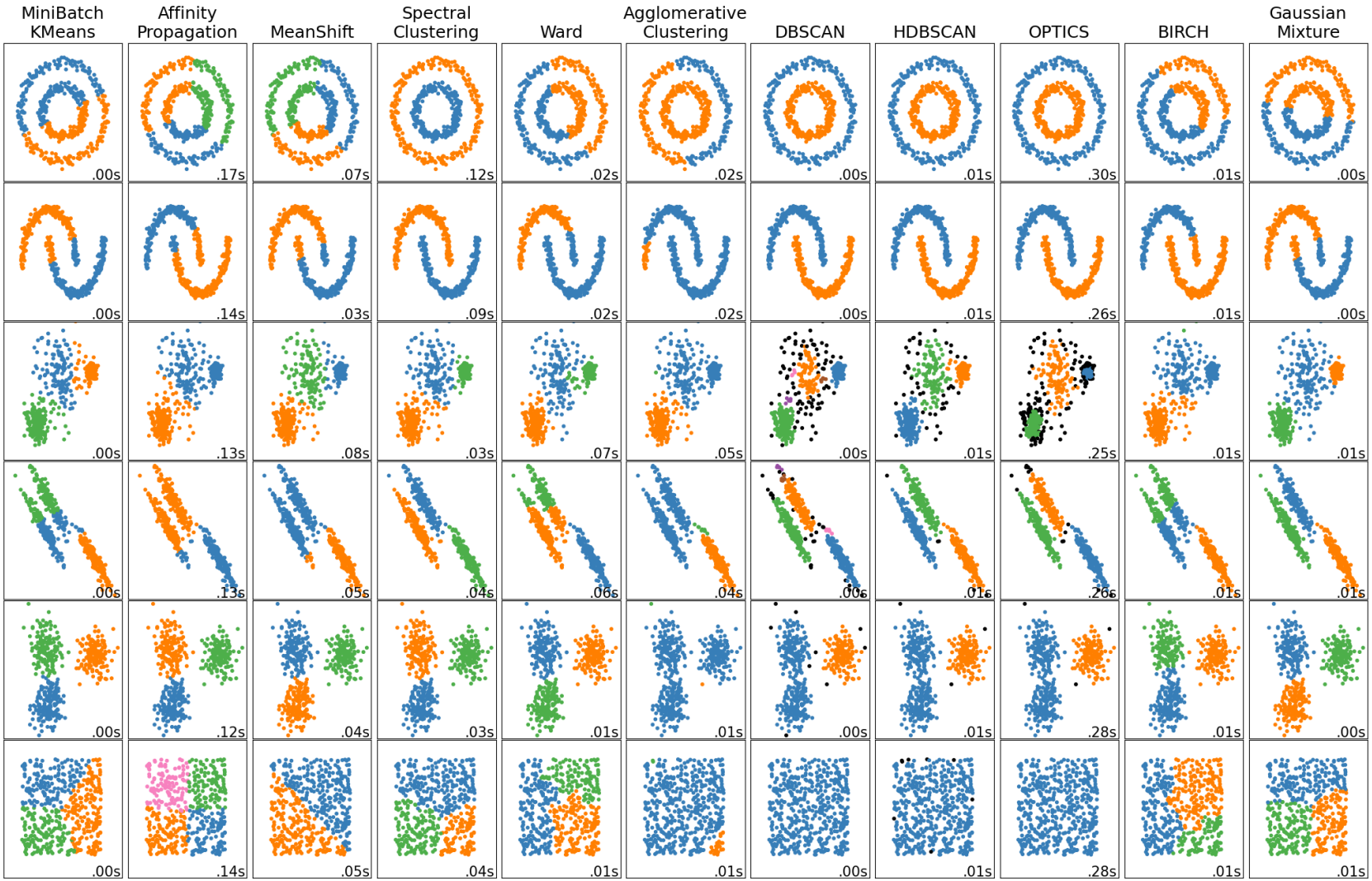
¿Para qué se usan?

- Agrupamiento (cluster analysis).
- Reducción de la dimensionalidad: PCA, Análisis de Componentes Principales. Como hemos visto, puede ser útil para reducir la complejidad de las redes con aprendizaje supervisado.
- Prototipado: ¿qué características suelen tener los patrones de una misma clase?
- Reconocimiento de similitud: ¿a qué patrones se parece el nuevo patrón de entrada? ¿cómo de parecido es?

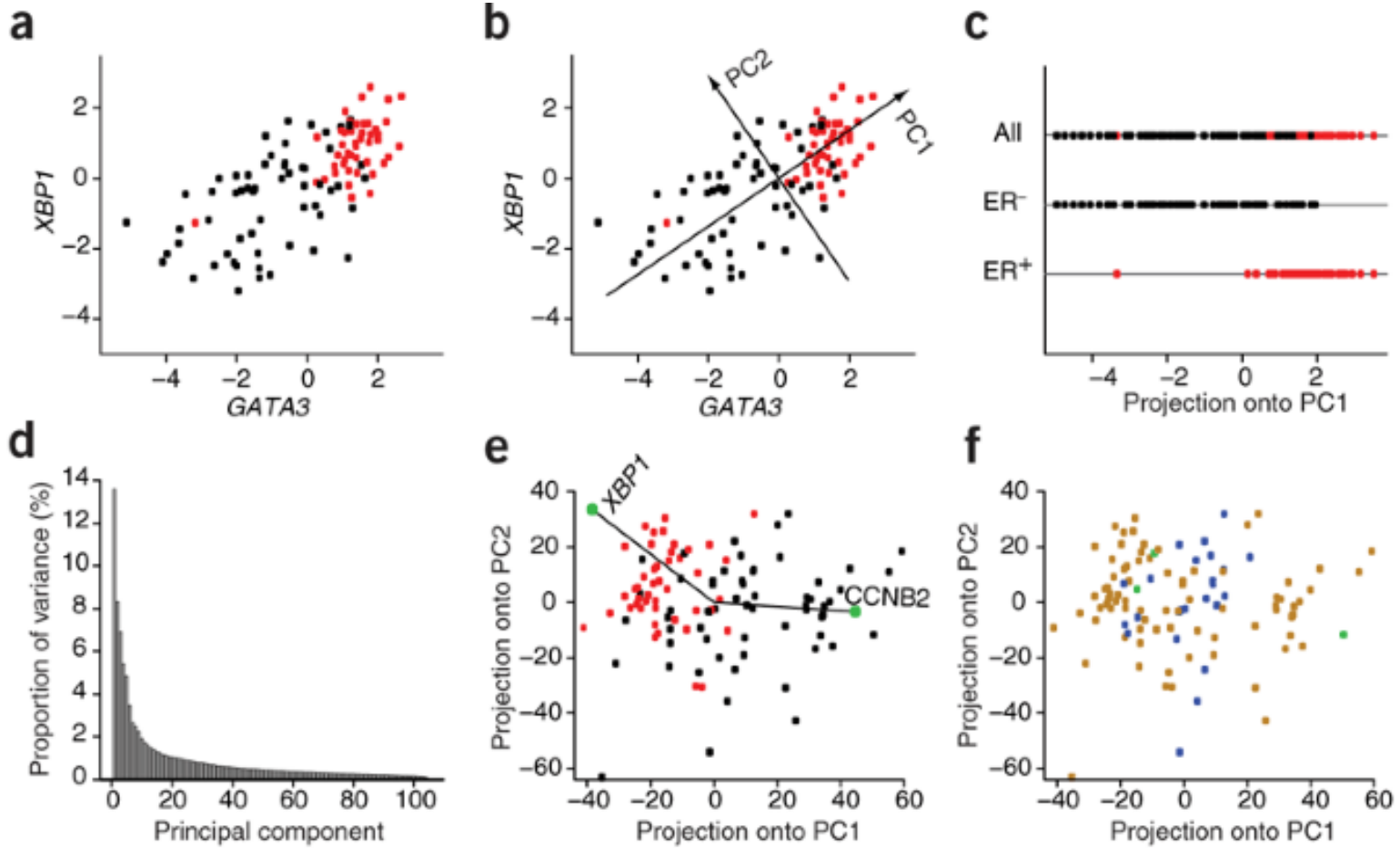
Ejemplo clustering



Ejemplo clustering (scikit-learn)



Ejemplo Análisis de componentes principales



Características del aprendizaje en sistemas autoorganizados

Entonces... ¿cuál es el objetivo de un sistema autoorganizado?
Categorizar/clasificar los datos de entrada.
Características del aprendizaje:

- Cada categoría de los patrones de entrada estará representada por un prototipo ideal (desconocido para la red).
- Los patrones pertenecerán a la categoría que represente el prototipo ideal al que más se parezcan.
- Existe una capa de clasificación compuesta con tantas neuronas como categorías hay en los datos (o una estimación).
- Cada neurona va ajustándose en función de los datos de entrada similares a la categoría que intenta representar.
- Pueden usar aprendizaje no supervisado o supervisado con refuerzo.

Aprendizaje no supervisado

¿Qué es el aprendizaje no supervisado?

- Aprender sin ninguna información sobre cómo debería ser la salida, es decir, sin conocimiento de lo cerca que ha estado de acertar y ni tan siquiera de si ha fallado o no. Es un aprendizaje completamente "a ciegas".
- En cada iteración, la red neuronal recibe uno o varios patrones y, únicamente en función de esos estímulos, modifica sus pesos sinápticos.

Aprendizaje supervisado con refuerzo

¿Qué es el aprendizaje supervisado con refuerzo?

- Aprender corrigiendo o reforzando los pesos sinápticos en función de si se equivoca o acierta.
- En este caso sabremos si acierta o no, pero no sabremos lo cerca que ha estado de acertar, es decir, lo cerca que se ha quedado el prototipo actual de aquel ideal que representaría correctamente al patrón de entrada. Es posible que ni siquiera sepamos cuál debería ser el prototipo ideal.
- En cada iteración la red neuronal recibe un patrón y, en función de esos estímulos y de si acierta o falla al clasificarlo, modifica sus pesos sinápticos.

Índice de contenidos

1. Sistemas Autoorganizados
   - Introducción
   - Redes Neuronales Competitivas no supervisadas

Estructura de la red

- Espacio de las variables de entrada con K componentes.
- Espacio de las neuronas de salida con J componentes.
- Debemos estimar un vector de pesos, $\mathbf{W} \in \mathbb{R}^{K \times J}$.
- Las J neuronas de salida representan a los J grupos considerados por el problema. En principio el valor de J es desconocido.
- El prototipo de cada clase j-ésima lo define el vector $\mathbf{w}_j \in \mathbb{R}^K$, que representan a los pesos que conectan a la neurona de salida j con las variables del espacio de entrada. Así $\mathbf{W} = (\mathbf{w}_1 \mathbf{w}_2 \dots \mathbf{w}_J)$

Estructura de la red

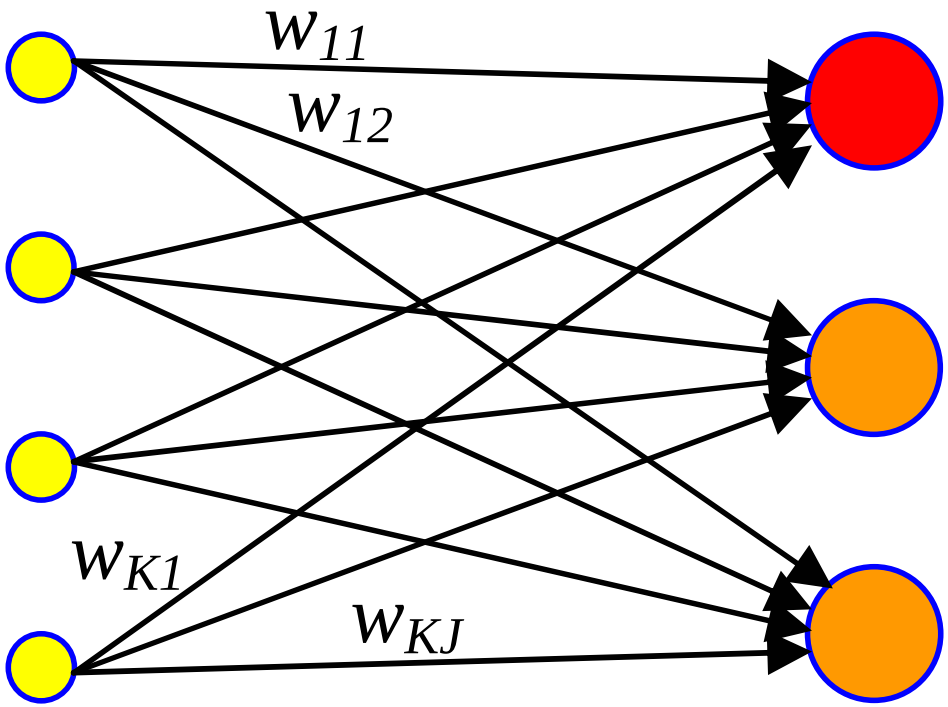


Figure: Arquitectura de red competitiva no supervisada. K variables de entrada, J de salida

Potencial sináptico de las neuronas de salida

La salida de la neurona j-ésima para el patrón n-ésimo, viene definida como:

$$h_j(\mathbf{x}_n) = w_{1j}x_{n1} + w_{2j}x_{n2} + \ldots + w_{Kj}x_{nK} - \theta_j,$$

con $\theta_j = \frac{1}{2}(w_{1j}^2 + w_{2j}^2 + \ldots + w_{Kj}^2)$.
La salida estimada en discreto, $\hat{y}_j(\mathbf{x}_n)$, de la neurona j para el patrón n-esimo será:

$$\hat{y}_j(\mathbf{x}_n) = \begin{cases} 1, & \text{si } h_j(\mathbf{x}_n) = \max(h_1(\mathbf{x}_n), h_2(\mathbf{x}_n), \ldots, h_J(\mathbf{x}_n)) \\ 0, & \text{en otro caso} \end{cases}$$

En el caso improbable de empate de potenciales, habría que elegir una ganadora al azar.

Dinámica de computación

La unidad de proceso ganadora es aquella cuyo peso sináptico $\mathbf{w}_j$ esté más cerca del patrón de entrada $\mathbf{x}_n$:

$$d(\mathbf{x}_n, \mathbf{w}_j) = \sqrt{(x_{n1} - w_{1j})^2 + (x_{n2} - w_{2j})^2 + \ldots + (x_{nK} - w_{Kj})^2}$$

Teorema: Si j, $\mathbf{w}_j$, es la unidad de proceso ganadora cuando se introduce el patrón de entrada $\mathbf{x}_n$ entonces:

$$d(\mathbf{x}_n, \mathbf{w}_j) \leq d(\mathbf{x}_n, \mathbf{w}_p), \forall p \neq j$$

Demostración:

$$\begin{aligned}
(d(\mathbf{x}_n, \mathbf{w}_j))^2 &= ||\mathbf{x} - \mathbf{w}_j||^2 = \mathbf{x}_n'\mathbf{x}_n - 2\mathbf{w}_j'\mathbf{x}_n + \mathbf{w}_j'\mathbf{w}_j \\
&= \mathbf{x}_n'\mathbf{x}_n - 2h_j(\mathbf{x}_n) \\
&\leq \mathbf{x}_n'\mathbf{x}_n - 2h_p(\mathbf{x}_n) = (d(\mathbf{x}_n, \mathbf{w}_p))^2.
\end{aligned}$$

Problema de optimización

Problema de optimización

Los parámetros se estiman según siguiente problema de optimización:

$$\min_{\mathbf{W}\in\mathbb{R}^{K\times J}} \quad E(\mathbf{W}) = \sum_{j=1}^{J}\sum_{n=1}^{N} a_{nj}||\mathbf{x}_n - \mathbf{w}_j||^2,$$

donde $\mathbf{W} \in \mathbb{R}^{K\times J}$ es la matriz de pesos de la red y a_{nj} es una variable binaria que será uno si el patrón n-ésimo pertenece a la clase j-ésima y 0 en caso contrario:

$$a_{nj} = \begin{cases} 1, & \text{si } \mathbf{x}_n \in \mathcal{C}_j \\ 0, & \text{en otro caso} \end{cases}$$

¿Se os ocurre otro criterio que se podría utilizar? Alejar a los que no pertenecen a la clase.

Aprendizaje en línea

El procedimiento será iterativo ($i = 1, \ldots, I$) y las modificaciones se realizarán de patrón a patrón ($1 \leq n \leq N$), por lo que será necesario definir:

- $\mathbf{w}_j(i, n) \in \mathbb{R}^K$: Vector de pesos de la neurona de salida j para el patrón n, en la iteración i.
- Al ser patrón a patrón, la función de error para el patrón n-ésimo en la iteración i-ésima quedaría definida como:

$$\min_{\mathbf{W} \in \mathbb{R}^{K \times J}} \quad E(\mathbf{W}) = \sum_{j=1}^{J} a_{nj} ||\mathbf{x}_n - \mathbf{w}_j(i, n)||^2.$$

Estimación de los parámetros en línea

Estimación $w_j, j = \{1, \ldots, J\}$

La regla de modificación de los pesos sinápticos de la capa de salida será:

$$\mathbf{w}_j(i, n+1) = \mathbf{w}_j(i, n) + \Delta\mathbf{w}_j(i, n), \tag{1}$$

donde

$$\Delta\mathbf{w}_j(i, n) = -\lambda \frac{\partial E(\mathbf{W})}{\partial \mathbf{w}_j(i, n)} = 2\lambda a_{nj}(\mathbf{x}_n - \mathbf{w}_j(i, n)).$$

Teniendo en cuenta que para cada patrón deberemos estimar el valor de a_{nj} como:

$$a_{nj} = \begin{cases} 1, & \text{si } ||\mathbf{x}_n - \mathbf{w}_j||^2 \leq ||\mathbf{x}_n - \mathbf{w}_p||^2, \forall p \neq j \\ 0, & \text{en otro caso} \end{cases}$$

Estimación de los parámetros en línea

A tener en cuenta...

A la unidad de proceso j que le corresponde $a_{nj} = 1$, es decir, aquella cuyo vector de pesos sinápticos está más cerca del patrón de entrada, diremos que es la unidad ganadora, y será la única que modifique su vector de pesos sinápticos, las demás no lo modifican. Así, si j es la neurona ganadora, y llamamos a $\eta = 2\lambda$ la tasa de aprendizaje tenemos que:

$$\mathbf{w}_j(i, n+1) = \mathbf{w}_j(i, n) + \eta(i, n)(\mathbf{x}_n - \mathbf{w}_j(i, n)).$$

Estimación de los parámetros en línea

A tener en cuenta...

La tasa de aprendizaje debe ser una función decreciente del número de iteraciones:

$$\eta(i,n) = \eta_0 \left(1 - \frac{((i-1)\cdot N) + n}{N \cdot I}\right).$$

donde $N \cdot I$ nos indica el número de ejecuciones del algoritmo (por iteración y patrón), y $((i-1)\cdot N) + n$ nos indica la iteración en la que nos encontramos. Finalmente, $\eta_0 \in (0, 1]$.

Flujo algorítmico

Red competitiva no supervisada (en línea) ($\mathbf{X}$, $\eta_0 > 0$):

for $j = 1$ until J **do**
  $\mathbf{w}_j(1,0) \leftarrow 2 \times \text{rand}(K,1) - 1, 0 \leq \text{rand}() < 1.$
end for
for $i = 1$ until I **do**
  for $n = 1$ until N **do**
    $\eta(i,n) \leftarrow \eta_0 \left(1 - \frac{((i-1)\cdot N) + n}{N \cdot I}\right)$
    for $j = 1$ until J **do**
      $a_{nj} \leftarrow \begin{cases} 1, & \text{si } ||\mathbf{x}_n - \mathbf{w}_j||^2 \leq ||\mathbf{x}_n - \mathbf{w}_p||^2, \forall p \neq j \\ 0, & \text{en otro caso} \end{cases}$
      $\Delta\mathbf{w}_j(i,n) \leftarrow \eta(i,n) a_{nj} (\mathbf{x}_n - \mathbf{w}_j(i,n))$
      $\mathbf{w}_j(i,n) \leftarrow \mathbf{w}_j(i,n-1) + \Delta\mathbf{w}_j(i,n).$
    end for
  end for
end for

Aprendizaje por lotes

Si actualizamos los vectores de pesos sinápticos después de introducir todos los patrones de entrada entonces tendremos que minimizar en cada iteración la expresión:

$$\min_{\mathbf{W}\in\mathbb{R}^{K\times J}} \quad E(\mathbf{W}) = \sum_{j=1}^{J}\sum_{n=1}^{N} a_{nj}||\mathbf{x}_n - \mathbf{w}_j(i)||^2,$$

donde ya desaparece el subíndice n al realizar la actualización por lotes. La variación en cada iteración:

$$\begin{aligned}\Delta\mathbf{w}_j(i) &= -\lambda\frac{\partial E(\mathbf{W})}{\partial\mathbf{w}_j(i)} = 2\lambda\sum_{n=1}^{N} a_{nj}(\mathbf{x}_n - \mathbf{w}_j(i)) \\ &= \eta(i)\sum_{n=1}^{N} a_{nj}(\mathbf{x}_n - \mathbf{w}_j(i)).\end{aligned}$$

con $\eta(i) = \eta_0\left(1 - \frac{i}{I}\right)$

Flujo algorítmico

Red competitiva no supervisada (por lotes) ($\mathbf{X}$, $\eta_0 > 0$):

for $j = 1$ until J **do**
  $\mathbf{w}_j(0) \leftarrow 2 \times \text{rand}(K, 1) - 1, 0 \leq \text{rand}() < 1.$
end for
for $i = 1$ until I **do**
  $\eta(i) \leftarrow \eta_0 \left(1 - \frac{i}{I}\right).$
  $\Delta\mathbf{w}_j(i) \leftarrow \mathbf{0}_K.$
  for $n = 1$ until N **do**
    for $j = 1$ until J **do**
      $$a_{nj} \leftarrow \begin{cases} 1, & \text{si } ||\mathbf{x}_n - \mathbf{w}_j||^2 \leq ||\mathbf{x}_n - \mathbf{w}_p||^2, \forall p \neq j \\ 0, & \text{en otro caso} \end{cases}$$
      $\Delta\mathbf{w}_j(i) \leftarrow \Delta\mathbf{w}_j(i) + \eta(i) a_{nj} (\mathbf{x}_n - \mathbf{w}_j(i)).$
    end for
  end for
  $\mathbf{w}_j(i) \leftarrow \mathbf{w}_j(i-1) + \Delta\mathbf{w}_j(i).$
end for

Posibles limitaciones del algoritmo

- Un problema que se puede presentar es que algunas unidades de proceso no ganen nunca, es decir, no se activen, lo que quiere decir que no representan a ningún patrón de entrada y se les llama unidades de proceso muertas.
- Ello se debe a que sus vectores sinápticos no están próximos a ningún patrón de entrada.
- **Posible solución**: Elegir los vectores sinápticos iniciales de las unidades de proceso iguales a patrones del conjunto de entrenamiento.

Posibles limitaciones del algoritmo

La minimización del ECM no es apropiada cuando hay una gran diferencia entre el tamaño de los grupos. En la figura observamos que la agrupación (a), que corresponde a un mayor valor del error cuadrático total, es la natural, mientras que la (b), que tiene un menor error cuadrático total, no lo es.

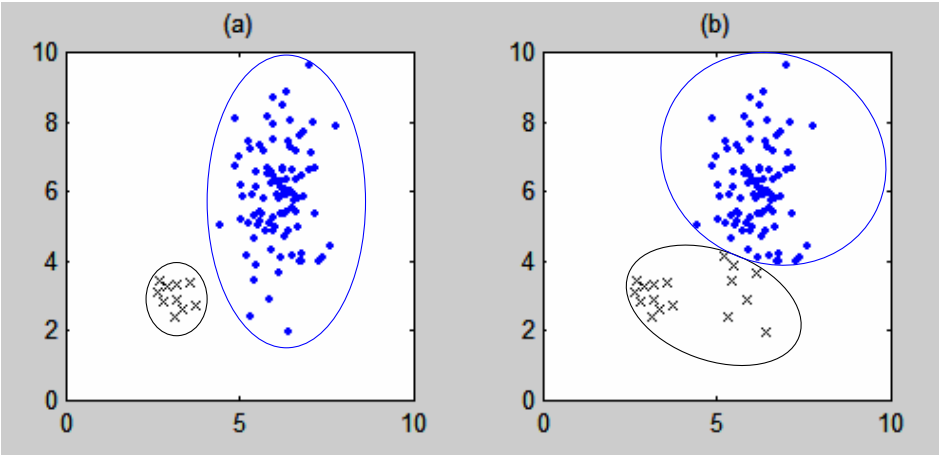


Figure: (a) Agrupación natural con valor de E grande. (b) Agrupación con valor de E pequeño.

Posibles limitaciones del algoritmo

Una posible solución al problema anterior es es minimizar el error cuadrático medio de representación dentro de cada grupo:

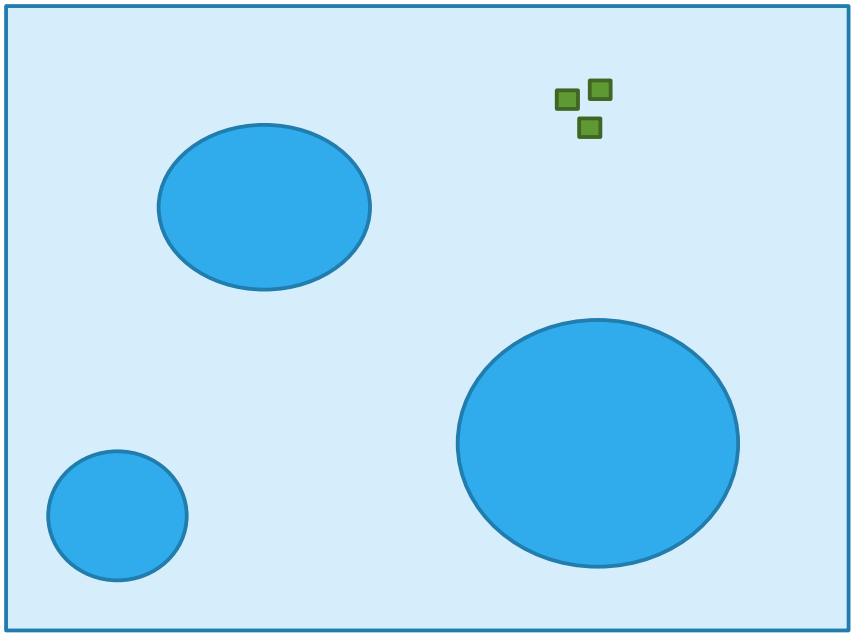
$$\min_{\mathbf{W}\in\mathbb{R}^{K\times J}} \quad E(\mathbf{W}) = \sum_{j=1}^{J} \frac{1}{n_j} \sum_{n=1}^{N} a_{nj} ||\mathbf{x}_n - \mathbf{w}_j(i)||^2,$$

donde n_j el número de patrones del grupo j, es decir, $n_j = \sum_{n=1}^{N} a_{nj}$. En este caso, la regla de aprendizaje por lotes:

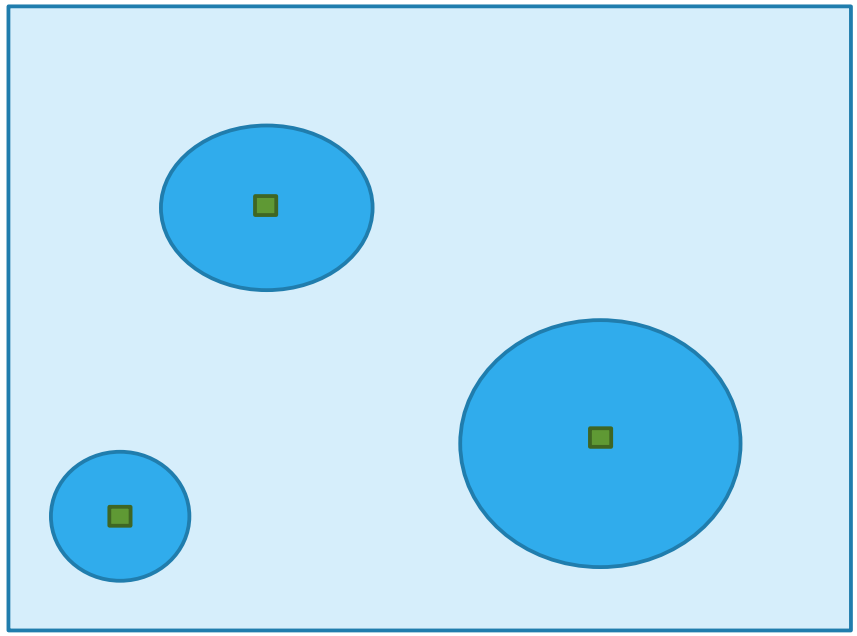
$$\Delta \mathbf{w}_j(i) = \eta(i) \frac{1}{n_j} \sum_{n=1}^{N} a_{nj} (\mathbf{x}_n - \mathbf{w}_j(i)).$$

Ejemplo 1

¿Cómo evolucionará la red?



(a) Estado inicial

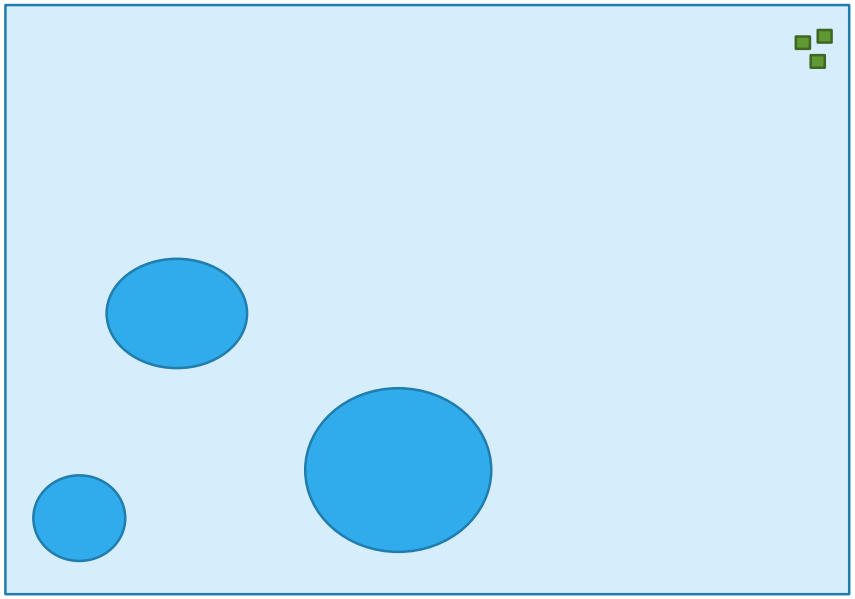


(b) Estado final

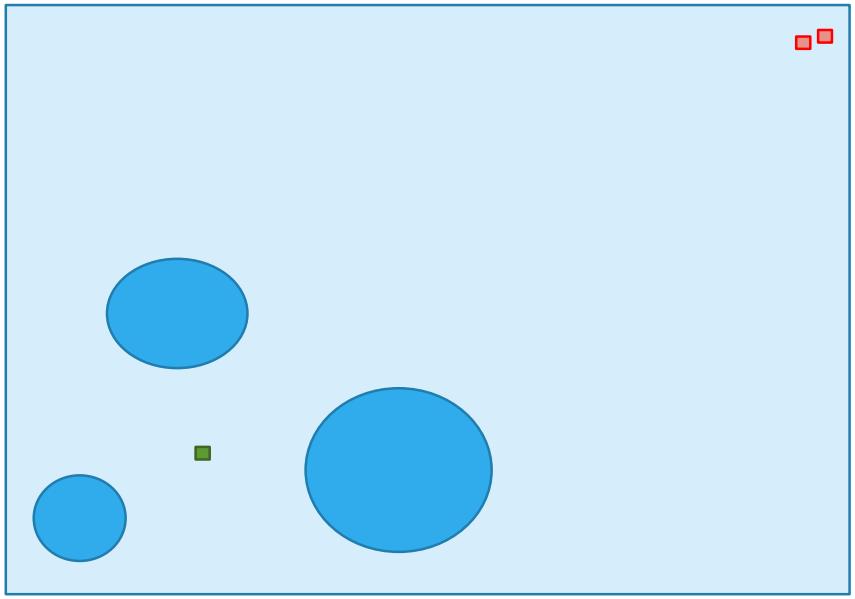
Figure: Primer ejemplo.

Ejemplo 2

¿Cómo evolucionará la red?



(a) Estado inicial



(b) Estado final

Figure: Segundo ejemplo.

¡Gracias por vuestra atención!