

6

Sistemas Autoorganizativos

6.1 Introducción

Las redes de neuronas artificiales con aprendizaje no supervisado se han aplicado con éxito a problemas de reconocimiento de patrones y detección de señales. Estas redes construyen clases o categorías a partir de los datos de entrada utilizando correlaciones o medidas de similitud y tratan de identificar particiones “óptimas” en el conjunto de datos de entrada.

En una red neuronal competitiva las unidades de salida compiten entre sí para activarse; sólo se activa la de mayor potencial sináptico. La idea del aprendizaje competitivo está ya trazada en los primeros trabajos de von der Malsburg (1973) sobre la autoorganización de las células nerviosas de la corteza cerebral. En 1975, Fukushima propuso el *cognitron* que es una red competitiva multicapa y autoorganizada. Willshaw y von der Malsburg (1976) trabajaron sobre la formación de las conexiones neuronales mediante autoorganización y Grossberg (1972, 1976) sobre la clasificación adaptativa de patrones.

Rumelhart y Zisper (1985) especificaron los tres elementos básicos de una regla de aprendizaje competitiva:

- Un conjunto de neuronas (unidades de proceso) que se activan o no en respuesta a un conjunto de patrones de entrada (estímulos) y que difieren en los valores de un conjunto de pesos sinápticos específico de cada neurona.
- Un *límite* impuesto sobre la “fuerza” de cada neurona.
- Un mecanismo que permite competir a las neuronas para responder a un subconjunto de entradas de tal manera que una y sólo una neurona por grupo se activa.

En una red neuronal competitiva simple las neuronas individuales aprenden a especializarse sobre conjuntos de patrones similares y, por lo tanto, llegan a ser detectoras de características de los patrones de entrada.

El algoritmo de aprendizaje competitivo simple se puede contemplar como un método aproximado para la reconstrucción de vectores de representación, también llamados de reproducción, prototipos o códigos, de manera no supervisada. Ahalt, Krishnamurthy and Chen (1990) han llevado a cabo la aplicación de las redes neuronales competitivas a la cuantificación vectorial (VQ) y han desarrollado un nuevo algoritmo no supervisado para el diseño de la tablas de códigos (vectores de representación o prototipos) que conducen a resultados óptimos o casi óptimos. Además, las experiencias computacionales muestran un conjunto de ventajas de dicho algoritmo frente al algoritmo tradicional LBG (Linde et al., 1980) para el diseño de cuantificadores vectoriales. Yair, Zeger y Gersho (1992) han demostrado propiedades de la convergencia de la red autoorganizada de Kohonen aplicada al diseño de

cuantificadores vectoriales y proponen condiciones sobre los parámetros de aprendizaje. Pal, Bezdek y Tsao (1993) han propuesto una generalización de la técnica de aprendizaje de cuantificación vectorial (LVQ) para la formación de grupos que evita la necesidad de definir un esquema de vecindad y donde los *centroides* finales no parece que sean sensibles a los valores iniciales. Xu, Krzyzak y Oja (1993) han desarrollado un nuevo algoritmo, llamado aprendizaje competitivo con rivales penalizados, donde para cada entrada no sólo se modifica la unidad de proceso ganadora para adaptarla al patrón de entrada sino que también sus rivales se modifican separándolas del patrón de entrada (desaprenden) con una tasa de aprendizaje menor. Ueda y Nakano (1994) han presentado un nuevo algoritmo de aprendizaje competitivo con un mecanismo de selección basado en el principio de *equidistorsión* para diseñar cuantificadores vectoriales óptimos; el mecanismo de selección le permite al sistema escapar de los mínimos locales. Mao y Jain (1996) han propuesto una red neuronal autoorganizada para agrupaciones hiperelipsoidales que aplican a problemas de segmentación de texturas.

Por otra parte, el análisis de grupos clásico (cluster analysis) trata de formar automáticamente grupos o categorías (clusters) a partir de un conjunto de datos de manera que a cada dato o entrada le asigna una única etiqueta o grupo. La agrupación va a suponer una partición de los datos en categorías o clases con características similares y se lleva a cabo asignando datos o patrones con atributos similares a la misma clase. Cuando se elige el criterio de mínimos cuadrados (mínima distorsión o principio de los M mejores centroides) la formación de grupos basada en particiones se puede realizar por el algoritmo clásico de las K -MEDIAS propuesto por McQueen (1967). Uchiyama y Arbib (1994) han demostrado la relación que hay entre la formación de grupos (clustering) basada en particiones y la cuantificación vectorial; así, el problema de la formación de grupos basada en el principio de mínimos cuadrados es el mismo que el problema de la selección óptima de los vectores de representación (también llamados de reproducción o prototipos). Además, presentan un algoritmo de aprendizaje competitivo que genera unidades donde la densidad de vectores de entrada es más alta y muestran su eficiencia como una herramienta para la formación de grupos en el espacio de color que permite la segmentación de imágenes de color basada en el criterio de mínimos cuadrados.

6.2 Redes Neuronales Competitivas no supervisadas

Una **unidad de proceso binaria** (neuronal artificial) es un dispositivo simple de cálculo que solo puede presentar dos estados, activo (encendido) e inactivo (apagado). El estado que presenta depende de las señales que le lleguen de los sensores de entrada o de otras unidades de proceso. Cada unidad de proceso binaria, i , va a tener asociado un vector de pesos sinápticos $(w_{i1}, w_{i2}, \dots, w_{iN})$, con el que va a ponderar los valores que le lleguen de los sensores de entrada.

Comenzaremos definiendo lo que se entiende por el **potencial sináptico** de una unidad de proceso. Si a la unidad de proceso i le llegan N señales, dadas por el vector (x_1, x_2, \dots, x_N) , y el vector de pesos sinápticos de dicha unidad es $(w_{i1}, w_{i2}, \dots, w_{iN})$, entonces el potencial sináptico viene dado por la expresión:

$$h_i = w_{i1}x_1 + w_{i2}x_2 + \dots + w_{iN}x_N - \theta_i \quad (1)$$

donde $\theta_i = \frac{1}{2}(w_{i1}^2 + w_{i2}^2 + \dots + w_{iN}^2)$.

Definición 1

Una red competitiva está constituida por N **sensores de entrada**, M **unidades de proceso** (neuronas artificiales), y **conexiones** entre cada sensor y cada unidad de proceso, de manera que la conexión entre el sensor j y la unidad de proceso i tiene asociado un valor w_{ij} .

Para cada entrada recogida por los sensores solamente una unidad de proceso se activa, aquella que tiene el mayor potencial sináptico, que se le considera como la unidad ganadora.

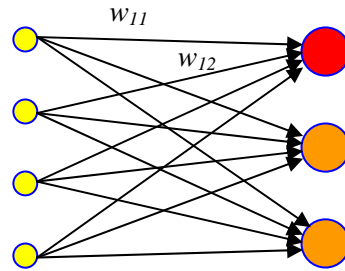


Figura 1. Arquitectura de la red.

Por lo tanto, si representamos el estado de la unidad de proceso i por la variable binaria y_i , que toma el valor 1 cuando dicha unidad está activada y cero en caso contrario, la **dinámica de la computación** de la red viene dada por la expresión:

$$y_i = \begin{cases} 1 & \text{si } h_i = \max_k \{h_1, h_2, \dots, h_M\} \\ 0 & \text{en otro caso} \end{cases}, \quad i=1,2,\dots,M \quad (2)$$

Cada entrada a la red, es un vector $(x_1, x_2, \dots, x_N) \in R^N$, que viene recogido por los sensores de entrada, y para el cual se activa una sola unidad de proceso, permaneciendo las restantes desactivadas. Así, podemos decir que la red neuronal competitiva es una función de R^N en el conjunto $\{1, 2, \dots, M\}$, que aplica un punto $(x_1, x_2, \dots, x_N) \in R^N$ en el valor $r \in \{1, 2, \dots, M\}$, cuando r sea la unidad ganadora. Dicha función produce una partición del espacio de los datos (patrones) de entrada en M regiones disjuntas. Dicho de otra forma, la red competitiva agrupa el conjunto de datos de entrada en M grupos o clases.

¿Cómo se determinan los pesos sinápticos? Mediante un proceso de aprendizaje no supervisado. Se pretende que se active aquella unidad de proceso cuyo vector de pesos sinápticos sea el “más parecido” al vector de entrada. De manera que los pesos sinápticos de cada unidad de proceso sean la “mejor representación” del conjunto de patrones que hacen que esa unidad de proceso sea ganadora. Para ello sólo tenemos que demostrar que la unidad ganadora es aquella cuyo vector de pesos sinápticos es el que más se parece al vector de entrada. El parecido entre el vector de entrada $\mathbf{x}=(x_1, x_2, \dots, x_N)'$ y el vector de pesos sinápticos de la unidad de proceso i , $\mathbf{w}_i=(w_{i1}, w_{i2}, \dots, w_{iN})'$, vendrá dado por la distancia euclídea entre dichos vectores, es decir,

$$d(\mathbf{x}, \mathbf{w}_i) = \|\mathbf{x} - \mathbf{w}_i\| = \sqrt{(x_1 - w_{i1})^2 + \dots + (x_N - w_{iN})^2}$$

A continuación vamos a demostrar que la unidad ganadora es aquella cuyo vector de pesos sinápticos es el que más se parece al vector de entrada.

Teorema 1

Si r es la unidad de proceso ganadora cuando se introduce el patrón de entrada, $\mathbf{x}=(x_1, x_2, \dots, x_N)$, entonces

$$d(\mathbf{x}, \mathbf{w}_r) \leq d(\mathbf{x}, \mathbf{w}_k), \quad k = 1, 2, \dots, M$$

Demostración:

En efecto,

$$\begin{aligned} (d(\mathbf{x}, \mathbf{w}_r))^2 &= \|\mathbf{x} - \mathbf{w}_r\|^2 = (\mathbf{x} - \mathbf{w}_r)'(\mathbf{x} - \mathbf{w}_r) = \mathbf{x}'\mathbf{x} - 2\mathbf{w}_r'\mathbf{x} + \mathbf{w}_r'\mathbf{w}_r \\ &= \mathbf{x}'\mathbf{x} - 2h_r \\ &\leq \mathbf{x}'\mathbf{x} - 2h_k = (d(\mathbf{x}, \mathbf{w}_k))^2 \quad \blacksquare \end{aligned}$$

Vamos a determinar los pesos sinápticos de la red utilizando un conjunto de p patrones de entrenamiento, que representaremos por $\mathbf{x}(k)=(x_1(k), x_2(k), \dots, x_N(k))$, $k=1, 2, \dots, p$, y siguiendo una regla de aprendizaje, es decir, una ecuación matemática que me especifique cómo se actualizan los pesos sinápticos cada vez que introduzco, como entrada, un patrón de entrenamiento. Los patrones de entrenamiento están agrupados en M clases, C_1, C_2, \dots, C_M , desjuntas entre sí, de manera que cada patrón de entrenamiento pertenece a una sola clase. Las clases no son conocidas pero tienen que estar formadas por los patrones más cercanos (próximos) entre si, es decir, más similares. El objetivo de la red competitiva con aprendizaje no supervisado (puesto que no conocemos las clases) es descubrir por sí misma los grupos o clases que forman los patrones de entrenamiento.

Para ello, vamos a elegir un criterio o principio. Dicho criterio va a ser el criterio de mínimos cuadrados. Según este criterio se trata de encontrar M vectores de pesos sinápticos $\mathbf{w}_1, \mathbf{w}_2, \dots, \mathbf{w}_M$, tales que la función suma de errores cuadráticos sea mínima, es decir, se trata de minimizar la expresión:

$$E = \sum_{i=1}^M \sum_{j=1}^p a_{ij} \|\mathbf{x}(j) - \mathbf{w}_i\|^2 \quad (3)$$

donde

$$a_{ij} = \begin{cases} 1 & \text{si } \mathbf{x}(j) \in C_i \\ 0 & \text{en otro caso} \end{cases}$$

la función a_{ik} me indica si el patrón de entrada $\mathbf{x}(k)$ es, o no, de la clase C_i , pero dicha función no es conocida.

Vamos a determinar los vectores de pesos sinápticos siguiendo un proceso iterativo que minimice la función de error cuadrático en cada paso, es decir, que el nuevo vector determinado por dicho proceso disminuya el error cuadrático E . A dicho proceso lo llamaremos **regla de aprendizaje**.

La regla de aprendizaje puede ser de dos formas, según que actualicemos los pesos sinápticos cada vez que introducimos un patrón de entrada a la red, en cuyo caso

diremos que el aprendizaje es individualizado o en línea, o actualizar los pesos sinápticos después de introducir todos los patrones de entrada, en cuyo caso diremos que el aprendizaje es por lotes. Supongamos primero que el aprendizaje es en línea. Sea $\mathbf{x}(k)$ el patrón que introducimos en la red en la iteración k . Se trata de modificar los vectores de pesos sinápticos de modo que se minimice la expresión del error que depende de dicho patrón:

$$E(k) = \sum_{i=1}^M a_{ik} \|\mathbf{x}(k) - \mathbf{w}_i(k)\|^2$$

Si en la iteración k los vectores de pesos sinápticos son $\mathbf{w}_1(k), \mathbf{w}_2(k), \dots, \mathbf{w}_M(k)$ entonces vamos a determinar los nuevos vectores en la iteración $k+1$ siguiendo el método del descenso del gradiente que viene dado por la expresión:

$$\Delta \mathbf{w}_i(k) = -\lambda \frac{\partial E}{\partial \mathbf{w}_i(k)}$$

donde λ es el parámetro que regula la longitud del paso en sentido opuesto al gradiente. Teniendo en cuenta que

$$\frac{\partial E}{\partial \mathbf{w}_i(k)} = -2a_{ik} (\mathbf{x}(k) - \mathbf{w}_i(k)) \quad (4)$$

y que el patrón $\mathbf{x}(k)$ sólo puede pertenecer a una de las M clases, vamos a estimar los valores desconocidos $a_{1k}, \dots, a_{ik}, \dots, a_{Mk}$ que son todos nulos menos uno, mediante la expresión

$$\hat{a}_{rk} = \begin{cases} 1 & \text{si } \|\mathbf{x}(k) - \mathbf{w}_r(k)\| < \|\mathbf{x}(k) - \mathbf{w}_i(k)\|, \forall i \neq r \\ 0 & \text{en otro caso} \end{cases} \quad (5)$$

A la unidad de proceso r que le corresponde $\hat{a}_{rk} = 1$, 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, como se desprende de la expresión (4). Por lo tanto la regla de aprendizaje es la siguiente:

$$\mathbf{w}_i(k+1) = \mathbf{w}_i(k) + \Delta \mathbf{w}_i(k) \quad (6)$$

donde

$$\Delta \mathbf{w}_i(k) = \begin{cases} \eta(k)(\mathbf{x}(k) - \mathbf{w}_i(k)) & \text{si } i = r \\ 0 & \text{si } i \neq r \end{cases} \quad (7)$$

y r es la unidad ganadora, es decir, la de mayor potencial sináptico (según el teorema 1) Note que $\eta = 2\lambda$. Al parámetro η lo llamaremos tasa de aprendizaje, pues conforme mayor sea, más se modifican los pesos sinápticos.

Para entender mejor dicha regla de aprendizaje vamos a realizar la siguiente interpretación geométrica. Si r es la unidad ganadora entonces

$$\mathbf{w}_r(k+1) = \mathbf{w}_r(k) + \eta(k)(\mathbf{x}(k) - \mathbf{w}_r(k))$$

es decir,

$$\mathbf{w}_r(k+1) = (1 - \eta(k))\mathbf{w}_r(k) + \eta(k)\mathbf{x}(k)$$

Por lo tanto, el nuevo vector de pesos sinápticos $\mathbf{w}_r(k+1)$ es una combinación lineal de los vectores $\mathbf{w}_r(k)$ y $\mathbf{x}(k)$. Quiere decir que el vector de pesos sinápticos se modifica acercándose al patrón de entrada, como se muestra en la figura 2. Conforme mayor es el valor del parámetro de aprendizaje más se acerca.

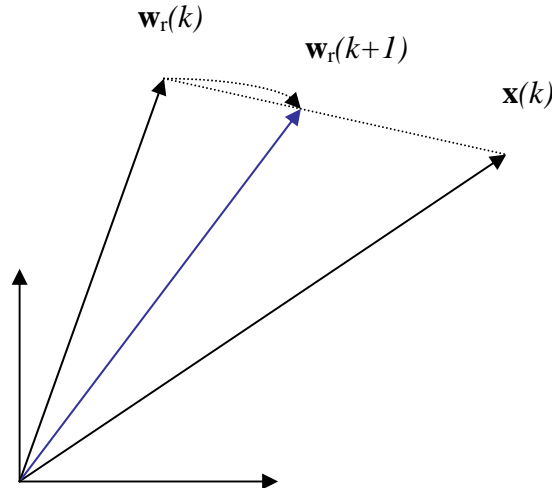


Figura 2. El nuevo vector de pesos sinápticos.

En cada iteración se introduce un patrón de entrada seleccionado aleatoriamente. El proceso continua hasta realizar un número total de T iteraciones, es decir, después de que cada patrón se haya introducido en la red un número determinado de veces (por ejemplo 10 veces).

Los valores iniciales de los vectores de pesos sinápticos pueden ser M patrones de entrada seleccionados aleatoriamente. El parámetro de aprendizaje debe de ir disminuyendo a lo largo de proceso de aprendizaje hasta alcanzar el valor cero para el cual la red deja de aprender. Inicialmente se puede elegir un valor $\eta_0 \in (0,1)$. El valor de dicho parámetro en la iteración k puede venir dado por la expresión:

$$\eta(k) = \eta_0 \left(1 - \frac{k}{T}\right), \quad k = 1, 2, \dots \quad (8)$$

que supone un decrecimiento lineal con respecto al número de iteración. El parámetro T es el número total de iteraciones del algoritmo hasta concluir el proceso de aprendizaje.

ALGORITMO DE APRENDIZAJE COMPETITIVO INDIVIDUALIZADO

Paso 0 Elegir como vectores de pesos sinápticos iniciales M patrones de entrenamiento y poner $k=1$.

Paso 1 Elegir aleatoriamente un patrón de entrenamiento.

Paso 2 Calcular los potenciales sinápticos

$$h_1(k), h_2(k), \dots, h_M(k).$$

Paso 3 Determinar la neurona ganadora r , es decir, la de mayor potencial sináptico

$$h_r(k) = \max_{1 \leq i \leq M} \{h_i(k)\}$$

Paso 4 Actualizar \mathbf{w}_r como sigue:

$$\mathbf{w}_r(k+1) = \mathbf{w}_r(k) + \eta(k)[\mathbf{x}(k) - \mathbf{w}_r(k)]$$

Paso 5 Calcular la nueva tasa de aprendizaje según la expresión

$$\eta(k) = \eta_0 \left(1 - \frac{k}{T}\right)$$

Paso 6 Si $k=T$ parar. Hemos encontrado los vectores sinápticos. En otro caso poner $k=k+1$ e ir al paso 1.

Note que la condición de parada se puede establecer fijando el número total de iteraciones T o estableciendo un valor suficientemente pequeño de la tasa de aprendizaje.

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

$$E(k) = \sum_{i=1}^M \sum_{j=1}^p a_{ij} \|\mathbf{x}(j) - \mathbf{w}_i(k)\|^2 \quad (9)$$

Para ello utilizamos también el método del descenso del gradiente. En este caso,

$$\frac{\partial E}{\partial \mathbf{w}_i} = -2 \sum_{j=1}^p a_{ij} (\mathbf{x}(j) - \mathbf{w}_i(k))$$

y la regla de aprendizaje es la siguiente:

$$\mathbf{w}_i(k+1) = \mathbf{w}_i(k) + \Delta \mathbf{w}_i(k)$$

donde

$$\Delta \mathbf{w}_i(k) = \eta(k) \sum_{j=1}^p a_{ij} (\mathbf{x}(j) - \mathbf{w}_i(k)) \quad (10)$$

y los coeficientes de pertenencia de patrones a clases, a_{ij} se estiman según la expresión (5).

Puede observarse que cuando

$$\mathbf{w}_i = \frac{\sum_{j=1}^p \hat{a}_{ij} \mathbf{x}(j)}{\sum_{j=1}^p \hat{a}_{ij}}$$

es decir, cuando el vector de pesos sinápticos de la unidad de proceso i es el patrón promedio de todos los patrones de entrada asignados a dicha unidad i (aquellos que la hacen ganadora por su proximidad con su vector sináptico) entonces $\Delta \mathbf{w}_i(k+1)=0$, y así no modifica sus pesos, pues ha encontrado el valor buscado.

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. Para tratar de evitar este problema se pueden seguir algunas pautas, como elegir los vectores sinápticos iniciales de las unidades de proceso iguales a patrones del conjunto de entrenamiento, en lugar de elegir valores aleatorios cualesquiera, o introducir un mecanismo de conciencia que evite que haya unidades de proceso que ganen con demasiada frecuencia; esto se puede hacer restándole un valor al potencial sináptico de las unidades que ganan con más frecuencia para evitar que ganen, y dicho valor se puede ir incrementando proporcionalmente al número de veces que ganen.

ALGORITMO DE APRENDIZAJE COMPETITIVO POR LOTES

Paso 0 Elegir como vectores de pesos sinápticos iniciales M patrones de entrenamiento.

Paso 2 Calcular los potenciales sinápticos

$$h_1(k), h_2(k), \dots, h_M(k)$$

para cada patrón de entrada $\mathbf{x}(k)$, $k=1, 2, \dots, p$.

Paso 3 Determinar la neurona ganadora r , es decir, la de mayor potencial sináptico,

$$h_r(k) = \max_{1 \leq i \leq M} \{h_i(k)\}$$

para cada patrón de entrada $\mathbf{x}(k)$, $k=1, 2, \dots, p$. Poner

$$\hat{a}_{ij} = \begin{cases} 1 & \text{si } i \text{ es la unidad ganadora para } \mathbf{x}(j) \\ 0 & \text{en otro caso} \end{cases}$$

Paso 4 (Regla de aprendizaje)

Actualizar cada \mathbf{w}_i como sigue:

$$\mathbf{w}_i(k+1) = \mathbf{w}_i(k) + \eta(k) \sum_{j=1}^p \hat{a}_{ij} [\mathbf{x}(j) - \mathbf{w}_i(k)]$$

Paso 5 Calcular la nueva tasa de aprendizaje según la expresión

$$\eta(k) = \eta_0 \left(1 - \frac{k}{T}\right)$$

Paso 6 Si $k=T$ parar. Hemos encontrado los vectores sinápticos. En otro caso poner $k=k+1$ e ir al paso 1.

Note que la condición de parada se puede establecer fijando el número total de iteraciones T o estableciendo un valor suficientemente pequeño de la tasa de aprendizaje.

La elección del criterio de mínima suma de errores cuadráticos (9) es apropiada cuando los grupos forman nubes compactas que están bien separadas unas de otras. Sin embargo, no es apropiada cuando hay una gran diferencia entre el tamaño de los grupos, es decir, entre el número de elementos que forman cada grupo. En la figura 3 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.

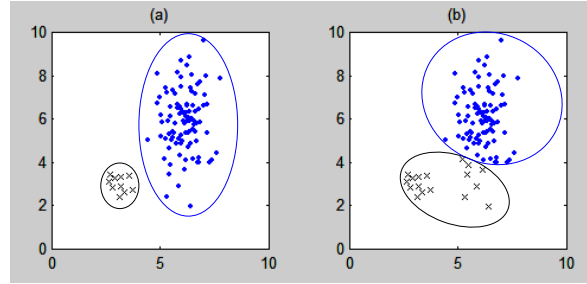


Figura 3. (a) Agrupación natural con valor de E grande.
(b) Agrupación con valor de E pequeño.

Esta situación se presenta también con la presencia de patrones atípicos que producen agrupaciones que no son adecuadas.

Un criterio alternativo es minimizar el error cuadrático medio de representación dentro de cada grupo, es decir, minimizar la función:

$$E(k) = \sum_{i=1}^M \frac{1}{n_i} \sum_{j=1}^p a_{ij} \|\mathbf{x}(j) - \mathbf{w}_i(k)\|^2$$

siendo n_i el número de patrones del grupo i , es decir, $n_i = \sum_{j=1}^p a_{ij}$. En este caso, la regla de aprendizaje por lotes viene dada por la expresión:

$$\Delta \mathbf{w}_i(k) = \eta(k) \frac{1}{\hat{n}_i} \sum_{j=1}^p \hat{a}_{ij} (\mathbf{x}(j) - \mathbf{w}_i(k)) \quad (11)$$

donde $\hat{n}_i = \sum_{j=1}^p \hat{a}_{ij}$. Es decir,

$$\begin{aligned} \mathbf{w}_i(k+1) &= \mathbf{w}_i(k) + \eta(k) \frac{1}{\hat{n}_i} \sum_{j=1}^p \hat{a}_{ij} [\mathbf{x}(j) - \mathbf{w}_i(k)] \\ &= (1 - \eta(k)) \mathbf{w}_i(k) + \eta(k) \frac{1}{\hat{n}_i} \sum_{j=1}^p \hat{a}_{ij} \mathbf{x}(j) \end{aligned}$$

Obsérvese que el nuevo valor del peso sináptico es una combinación lineal de su valor actual con el patrón promedio de los patrones asignados a la clase i .

Asimismo, la regla de aprendizaje en línea viene dada por la siguiente expresión:

$$\Delta \mathbf{w}_i(k) = \begin{cases} \eta(k) \frac{1}{\hat{n}_i} (\mathbf{x}(k) - \mathbf{w}_i(k)) & \text{si } i = r \\ 0 & \text{si } i \neq r \end{cases} \quad (12)$$

siendo r la unidad ganadora en esta iteración. Asimismo, teniendo en cuenta que

$$\begin{aligned} \mathbf{w}_r(k+1) &= \mathbf{w}_r(k) + \eta(k) \frac{1}{\hat{n}_r} (\mathbf{x}(k) - \mathbf{w}_r(k)) \\ &= (1 - \eta(k) \frac{1}{\hat{n}_r}) \mathbf{w}_r(k) + \eta(k) \frac{1}{\hat{n}_r} \mathbf{x}(k) \end{aligned}$$

vemos que también el nuevo valor del vector sináptico es una combinación lineal de su valor actual con el patrón de entrada, ponderado de forma inversa con el tamaño de la clase a la que pertenece, como parece lógico. Es decir, se modifica menos el vector sináptico cuantos más patrones tenga la clase r a la que ha sido asignado el patrón de entrada.

6.3 Redes neuronales autoorganizativas: Mapas de Kohonen

En el aprendizaje competitivo no hemos tenido en cuenta para nada la posición física de las unidades de proceso. Sin embargo, no ocurre así en el cerebro humano, donde neuronas próximas físicamente presentan características y comportamiento similares. Así, el desarrollo de estos modelos está motivado por la manera de organizarse que tiene el cerebro. Las diferentes áreas de la corteza cerebral están caracterizadas por la delgadez de sus capas y por el tipo de neuronas que hay dentro de ellas; así tenemos las áreas visual, auditiva, motora, etc. Cada entrada sensorial es aplicada al área correspondiente de la corteza cerebral de una manera ordenada. Por lo tanto, la localización espacial de una neurona dentro de un mapa topográfico va a corresponder a un dominio o característica particular de los datos de entrada. Así, inspirados en estas ideas las unidades de proceso se van a estar colocadas sobre una cuadrícula o rejilla rectangular dentro de la cual cada unidad de proceso va a tener un conjunto de unidades vecinas, de manera que los pesos sinápticos de las unidades vecinas deberán ser parecidos. Esta idea está inspirada en los estudios pioneros que hizo von der Malsburg (1973) indicando que un modelo de la corteza visual puede no estar completamente predeterminado genéticamente sino que un proceso de autoorganización por aprendizaje puede ser responsable de la ordenación local de las neuronas. Kohonen (1982) presentó un modelo sencillo para la formación autoorganizada de mapas de características de los datos de entrada del que nos ocuparemos en este capítulo.

La **autoorganización** es un fenómeno observado en la naturaleza mediante el cual se alcanza un orden global a partir de interacciones locales (Turing 1952). Dicho orden global conduce a un **comportamiento coherente** que es la esencia de la autoorganización. La autoorganización es un proceso de aprendizaje no supervisado mediante el cual se descubren características o patrones significativos en los datos de entrada.

Consideremos $M_1 \times M_2$ unidades de proceso colocadas sobre una rejilla rectangular (figura 4) de manera que el vector $\mathbf{p}_i = (p_{i1}, p_{i2})$ nos da la posición de la unidad de proceso i . Para establecer la noción de proximidad entre las unidades de proceso definiremos una función distancia, $d(\mathbf{p}_i, \mathbf{p}_j) = \|\mathbf{p}_i - \mathbf{p}_j\|$, que nos da la distancia que hay entre la unidad \mathbf{p}_i y la unidad \mathbf{p}_j . Podemos utilizar la distancia euclídea,

$$d(\mathbf{p}_i, \mathbf{p}_j) = \sqrt{(p_{i1} - p_{j1})^2 + (p_{i2} - p_{j2})^2},$$

la distancia rectangular,

$$d(\mathbf{p}_i, \mathbf{p}_j) = |p_{i1} - p_{j1}| + |p_{i2} - p_{j2}|$$

o cualquier otra función distancia.

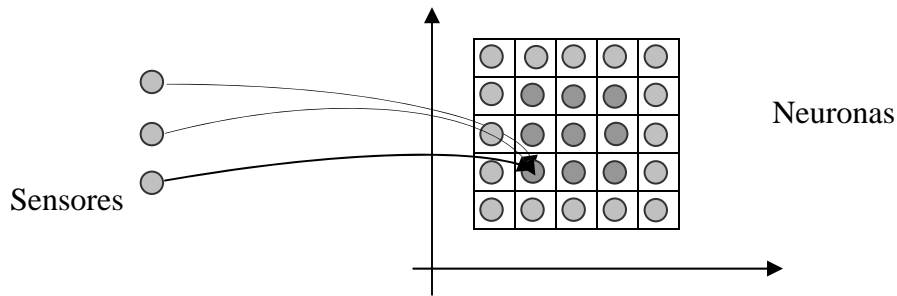


Figura 4. Rejilla de colocación de las unidades de proceso.

A continuación definimos una **función de vecindad** (o función de ventana) que tomará valores mayores conforme más próximas estén las dos unidades de proceso, es decir, es cualquier función decreciente de la distancia entre las mismas. Un ejemplo de función de vecindad cuando las unidades de proceso están sobre un segmento es

$$\Lambda(\mathbf{p}_i, \mathbf{p}_r) = \begin{cases} 1 & \text{si } i = r \\ 0.5 & \text{si } |i - r| = 1 \\ 0 & \text{en otro caso} \end{cases}$$

Si las unidades de proceso estuvieran sobre una cuadrícula se podría utilizar una función de vecindad determinada por una plantilla o ventana, como la siguiente:

0	0.5	0
0.5	1	0.5
0	0.5	0

Es decir, la función de vecindad vale 1 cuando \mathbf{p}_j coincide con \mathbf{p}_i ; vale 0.5 cuando \mathbf{p}_j es la unidad vecina que está encima, debajo, a la derecha o a la izquierda de \mathbf{p}_i , y vale cero para el resto de los casos.

En general, cuando las unidades de proceso estuvieran ubicadas espacialmente se puede emplear como función de vecindad:

$$\Lambda(\mathbf{p}_i, \mathbf{p}_r) = e^{-d(\mathbf{p}_i, \mathbf{p}_r)}$$

Además, también se puede utilizar funciones de vecindad dinámicas, es decir, que se van modificando en cada iteración. Por ejemplo, la función de vecindad dinámica:

$$\Lambda(\mathbf{p}_i, \mathbf{p}_r) = e^{-\frac{d(\mathbf{p}_i, \mathbf{p}_r)}{\sigma(k)^2}}$$

donde $\sigma(k) = \sigma_0(1+k)^{-1/2}$. Dicha función va reduciendo su radio de vecindad conforme k aumenta aproximándose sucesivamente a un aprendizaje competitivo simple.

La tarea que pretendemos realizar es la siguiente: Dado un conjunto de patrones del espacio de entradas, vamos a construir una aplicación entre dicho espacio y el espacio de colocación de las unidades de proceso de manera que cada patrón de entrada se asigna a una unidad de proceso (la unidad ganadora) de forma que patrones de entrada vecinos según la topología definida en el espacio de entradas se correspondan con unidades de proceso vecinas según la topología definida entre las unidades de proceso. Para ello seguiremos una la regla de aprendizaje similar a la regla competitiva donde habrá una unidad ganadora pero ahora se modifican también los vectores sinápticos de las unidades de proceso vecinas, aunque en menor medida, según su proximidad a la unidad ganadora, acercándose al patrón de entrada. Ello garantiza que las unidades de proceso vecinas tengan sus vectores sinápticos parecidos, es decir, se preserva la topología del espacio de entrada. Por ejemplo, si se trata de agrupar fonemas, las señales de sonido correspondientes a la pronunciación del fonema “be” será asignarán a una unidad de proceso lejana de la unidad de proceso correspondiente al fonema “tu” pero vecina de la unidad correspondiente al fonema “ve”.

Por lo tanto, si \mathbf{w}_i es el vector de pesos sinápticos de la unidad de proceso i correspondiente a la conexión entre el sensor de entrada y dicha unidad, la actualización del mismo se realiza según la siguiente regla de aprendizaje:

$$\mathbf{w}_i(k+1) = \mathbf{w}_i(k) + \eta(k)\Lambda(\mathbf{p}_r, \mathbf{p}_i)(\mathbf{x}(k) - \mathbf{w}_i(k)) \quad , i=1,2,\dots,M_1 \times M_2 \quad (13)$$

siendo r la unidad de proceso ganadora que, de manera similar a la regla del aprendizaje competitivo, es la de mayor potencial sináptico. Para la tasa de aprendizaje podemos elegir una función lineal y decreciente, como la función $\eta(k) = (1 - \frac{k}{T})\eta_0$, donde $\eta_0 \in (0,1]$ es el valor inicial de dicha tasa y T el número total de iteraciones, o una función no lineal y decreciente como la función $\eta(k) = (1+k)^{-1/2}$.

La interpretación de esta regla de aprendizaje es la siguiente: Para cada patrón de entrada se ajusta el vector de pesos sinápticos de la unidad ganadora de manera que sea más parecido a dicho patrón, es decir, se acerca al mismo; también se ajustan los vectores sinápticos de las unidades vecinas, pero en menor medida, dependiendo de su proximidad.

Se obtiene así una rejilla sobre la que están colocadas las unidades de proceso (neuronas). Cada unidad de proceso tiene un vector de pesos sinápticos que representa a todos los patrones de entrada que hacen ganadora a dicha unidad, ya que es el vector de pesos sinápticos más próximo (similar) a dichos patrones de entrada. A dicha rejilla, con los correspondientes vectores sinápticos de sus neuronas determinados por la regla de aprendizaje (13), la llamaremos **Mapa Autoorganizativo de características** (en inglés Self-Organizing Feature Map, acrónimo SOFM, o simplemente SOM). Una SOM trata de proyectar los patrones (puntos) de entrada sobre una rejilla de manera que los patrones (puntos) próximos en el espacio de entrada se asignen a neuronas próximas en la rejilla (preservación de la topología). Las rejillas pueden ser rectangulares, hexagonales, circulares, etc. En la figura 5 mostramos una rejilla hexagonal, en la que

cada neurona tiene 6 neuronas vecinas, y una rejilla rectangular, donde cada neurona puede tener 4 u 8 neuronas vecinas, según se defina el tamaño del entorno.

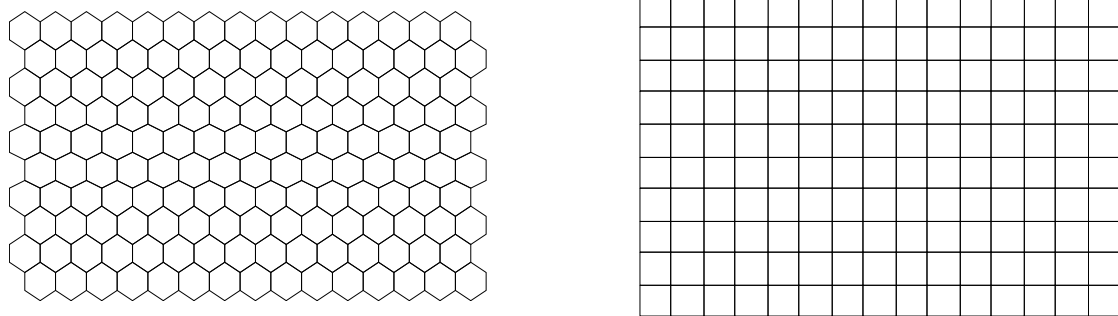


Figura 5 a) Rejilla hexagonal. b) rejilla rectangular.

El proceso de entrenamiento de un Mapa Autoorganizativo consta de 2 etapas:

- a) En la primera etapa, llamada **fase de ordenación global**, tiene lugar la ordenación topológica de los vectores sinápticos (vectores de referencia o prototipos) de las unidades de proceso. Se utiliza una tasa de aprendizaje alta (0.9) y un radio de vecindad grande similar al diámetro del mapa. A medida que avanza el aprendizaje, tanto la tasa de aprendizaje como el radio de vecindad se van reduciendo forma lineal hasta alcanzar unos valores mínimos, 0.02 y 1, respectivamente.
- b) La segunda etapa, llamada **fase de ajuste fino**, tiene como objetivo conseguir la convergencia de los vectores sinápticos a los prototipos de los grupos que construye la red. La tasa de aprendizaje se toma pequeña (0.05) y se puede mantener constante o reduciéndose suavemente. El radio de vecindad se toma igual a 1 (mínimo).

Se repite el proceso de entrenamiento 10 veces, cada una de ellas con una configuración inicial diferente de los vectores sinápticos. Se calcula el **error de representación** para cada uno de los mapas obtenidos y se selecciona el de menor error.

Por lo tanto, el proceso de aprendizaje de un Mapa Autoorganizativo se puede dividir en tres fases. Una primera fase **competitiva** en la que se determina la neurona ganadora, es decir, la mejor emparejada con el patrón de entrada según la distancia euclidiana; una segunda fase **cooperativa** establecida en términos de una función de vecindad entre la neurona ganadora y las demás, y, finalmente, una tercera fase **adaptativa** que actualiza los vectores sinápticos de la unidad ganadora y sus vecinas según la regla de aprendizaje (13).

6.3.1 Precisión de la transformación

Cada unidad de proceso tiene asignado el conjunto de patrones de entrada que la hace ganadora. Dicho conjunto viene representado por el vector de pesos sinápticos de la unidad de proceso. Una **medida de precisión** de la transformación de un mapa autoorganizativo describe cómo de correcta es la representación de dicho conjunto de patrones por el vector de pesos sinápticos de la unidad de proceso que los representa (unidad ganadora). Es decir, la precisión de las respuestas de las unidades de proceso a

un conjunto de patrones (datos) dado. Una medida que calcula la precisión de la transformación es el **error medio de cuantificación** sobre un conjunto de N datos que viene dada por la expresión:

$$EMC = \frac{1}{N} \sum_{k=1}^N \|\mathbf{x}_k - \mathbf{w}_{(k)}\|^2$$

siendo $\mathbf{w}_{(k)}$ el vector sináptico de la unidad de proceso ganadora cuando la entrada es el patrón \mathbf{x}_k .

6.3.2. Preservación de la topología

Con una red autoorganizativa se persigue la preservación topológica en el plano de la proyección. Es decir, que los puntos próximos en el espacio original estén también próximos en el plano de la proyección. Una medida de la preservación de la topología del conjunto de patrones (datos) en el mapa autoorganizado es el error topográfico que viene dado por la expresión:

$$ET = \frac{1}{N} \sum_{k=1}^N u(\mathbf{x}_k)$$

donde

$$u(\mathbf{x}_k) = \begin{cases} 1 & \text{si la primera y segunda unidades ganadoras para } \mathbf{x}_k \text{ están próximas} \\ 0 & \text{en otro caso} \end{cases}$$

Por otro lado, la **proyección de Sammon** es un algoritmo que proyecta un espacio de alta dimensión en un espacio de baja dimensión (generalmente en una región bidimensional) de manera que se minimice la siguiente función de error:

$$E = \frac{1}{\sum_{i < j} d_{ij}^*} \sum_{i < j} \frac{(d_{ij}^* - d_{ij})^2}{d_{ij}^*}$$

siendo d_{ij}^* la distancia entre los objetos i -ésimo y j -ésimo en el espacio original, y d_{ij} la distancia entre sus proyecciones. Podemos utilizar también la expresión anterior para comprobar la adecuación de la proyección realizada por la red autoorganizada, es decir, el grado de preservación topológica de los objetos (puntos).

6.3.3. Aplicaciones

Aplicación 1: Visualización de datos

Una de las aplicaciones de los Mapas autoorganizativos es a la **visualización de datos** ya que proyectan puntos (patrones) de un espacio de alta dimensionalidad a puntos sobre una rejilla intentando mantener la topología (puntos próximos en el espacio de entrada se corresponde con puntos próximos en la rejilla). Una **matriz de distancias unificada (U-matriz)** es una visualización 2D de datos multivariantes que utiliza los vectores sinápticos (prototipos, vectores de representación o vectores código) de las unidades de proceso de un Mapa Autoorganizativo. Para ello, se genera una matriz en la que cada elemento de la misma está asociado a una unidad de proceso y el valor de cada

elemento viene determinado por la distancia media del vector de pesos sinápticos de la unidad de proceso a los vectores sinápticos de sus unidades de proceso vecinas. Por ejemplo, consideremos un mapa autoorganizado de tamaño 5×1 , siendo \mathbf{p}_i , $i=1,2,\dots,5$, las unidades de proceso. La U -matriz es el vector $(u_1, u_2, u_3, u_4, u_5)$ donde

$$u_i = \frac{1}{2} \sum_{j \in N(i)} d(\mathbf{w}_i, \mathbf{w}_j), \quad i = 1, 2, \dots, 5.$$

y $N(i)$ es el conjunto de unidades de proceso vecinas con la unidad i .

Los valores altos de una U -matriz representan una región frontera entre clusters mientras que los valores bajos representan un alto grado de similitud entre las neuronas de esa región (clusters). Podemos utilizar algún sistema de colores o tonos de gris para realizar una representación gráfica de dicha matriz. Por ejemplo, utilizar tonos de gris claros para valores altos de la distancia y oscuros para valores bajos. Así, una tonalidad clara entre neuronas corresponde a una gran distancia entre sus vectores sinápticos (hay un gran espacio entre ellos). Una oscura indica que los vectores sinápticos están próximos entre sí en el espacio de entrada. Las áreas oscuras se pueden interpretar como agrupaciones y las claras como separaciones de grupos. Ello nos puede ayudar a definir los grupos (clusters) cuando no tenemos información a priori de los mismos. Por ejemplo, vamos a construir una U -matriz para el conjunto de datos IRIS formada por 150 patrones, cada uno constituido por los valores de 4 características de la hojas (ancho y largo del pétalo y del sépalo) de tres variedades de lirios (setosa, versicolor y virginica). Hay 50 patrones de cada variedad. Si utilizamos una rejilla cuadrada de 40×40 unidades de proceso se puede obtener una U -matriz como la de la figura 6. Las neuronas de la rejilla que se activan para lirios de la variedad setosa tienen un punto de color azul, para la variedad versicolor el punto es verde y para la variedad virginica es rojo. Los elementos más claros de la matriz (montañas) corresponden a grandes distancia y por lo tanto marcan las fronteras de los grupo mientras que los elementos oscuros (valles) se corresponden con patrones similares. Los lirios de la variedad setosa (punto azul) están claramente separados de las otras dos variedades.

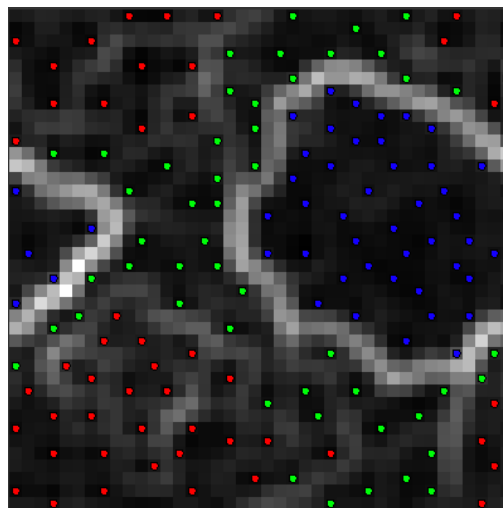


Figura 6. Mapa Autoorganizativo de Kohonen 40×40 para datos IRIS.

Aplicación 2: ¿Cómo se puede tratar de resolver el problema del viajante con una **Red autoorganizada de Kohonen**? Teniendo en cuenta que la solución del problema del viajante es un recorrido (ruta) que pasa por cada una de las N ciudades sólo una vez y regresa a la ciudad de partida, vamos a utilizar una red autoorganizada con un número

mayor de unidades de proceso (por ejemplo, $3N$) que ciudades a visitar. Las unidades de proceso van a estar colocadas sobre una circunferencia e igualmente espaciadas (ver la figura 7), de manera que la circunferencia nos determinará la ruta a seguir y los pesos sinápticos de ciertas unidades de proceso van a llegar a ser las coordenadas de los pueblos que representan.

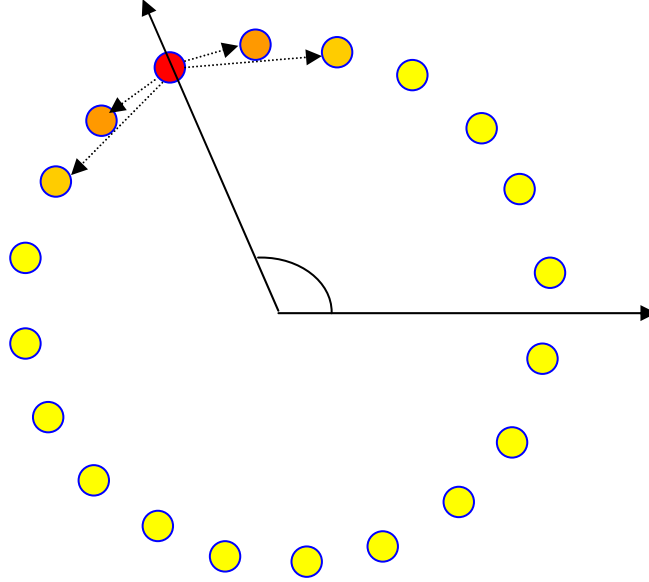


Figura 7 Topología de la red autoorganizada.

Vamos a tomar como *función de vecindad*:

$$\Lambda(\theta_r, \theta_j) = \begin{cases} 1 & \text{si } |\theta_r - \theta_j| = 0 \\ 1/2 & \text{si } |\theta_r - \theta_j| = \frac{2\pi}{3N} \\ 1/4 & \text{si } |\theta_r - \theta_j| = \frac{4\pi}{3N} \\ 0 & \text{en otro caso} \end{cases}, \quad i=1,2,\dots,3N$$

siendo r la neurona ganadora y θ_i el ángulo, en radianes, que determina la posición de la neurona artificial i con respecto al origen y el eje de abscisas. Con esta función la unidad ganadora comparte la mitad de sus ganancias con cuatro neuronas vecinas. Así, conseguiremos que los vectores sinápticos de neuronas vecinas sean también próximos.

Por lo tanto, la dinámica de computación es la siguiente:

Se determina la unidad ganadora, r , es decir, la de mayor potencial sináptico:

$$h_r \geq h_i$$

siendo

$$h_i = \sum_{j=1}^N w_{ij} x_j - \sum_{j=1}^N w_{ij}^2 / 2$$

y se sigue la siguiente *regla de aprendizaje*:

$$\Delta \mathbf{w}_i(k) = \eta \Lambda(\theta_r, \theta_i) (\mathbf{x} - \mathbf{w}_i), \quad i=1,2,\dots,N$$

tomando como conjunto de patrones de entrenamiento el conjunto de N puntos: $\{(x_i, y_i), i=1,2,...,N\}$ que corresponden a las coordenadas de las ciudades. Los vectores sinápticos serán atraídos por los puntos donde están situadas las ciudades y cuando se estabilice la red los pesos sinápticos serán las coordenadas de las ciudades y la ruta viene determinada por la secuencia de ciudades sobre la circunferencia.

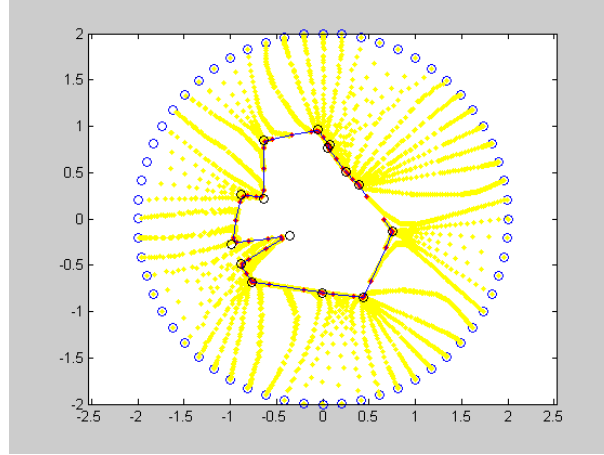


Figura 8. Evolución de los pesos sinápticos.

En la figura 8 presentamos la trayectoria de los pesos sinápticos y el resultado obtenido cuando el número de ciudades es igual a 15 y el número de unidades de proceso es igual a 45. ■

Aplicación 3: Se dispone de 322 puntos (x_i, y_i) , $i=1,2,...,322$, que configuran el contorno difuso de un *vaso sanguíneo* en una mamografía (ver la figura 9). Se trata de **diseñar una red neuronal** que construya el contorno poligonal de 30 vértices que “mejor” se ajusta al contorno del vaso sanguíneo.

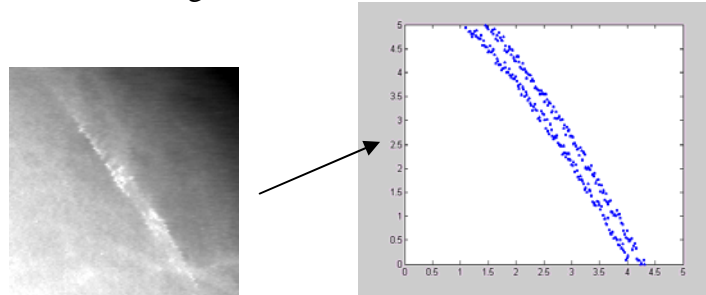


Figura 9. Vaso capilar en una mamografía.

Se puede utilizar una **red autoorganizada de Kohonen** con tantas neuronas artificiales como vértices tenga el contorno poligonal. Como no conocemos la posición ni la orientación del vaso sanguíneo, las neuronas artificiales estarán colocadas sobre una circunferencia e igualmente espaciadas (figura 10).

Vamos a tomar como *función de vecindad*:

$$\Lambda(\theta_r, \theta_j) = \begin{cases} 1 & \text{si } |\theta_r - \theta_j| = 0 \\ 1/2 & \text{si } |\theta_r - \theta_j| = \frac{\pi}{10}, i=1,2,...,N \\ 0 & \text{en otro caso} \end{cases}$$

siendo r la neurona ganadora y θ_i el ángulo, en radianes, que determina la posición de la neurona artificial i con respecto al origen y el eje de abscisas. Con esta función la unidad ganadora comparte la mitad de sus ganancias con dos neuronas vecinas. Así, conseguiremos que los vectores sinápticos de neuronas vecinas sean también próximos.

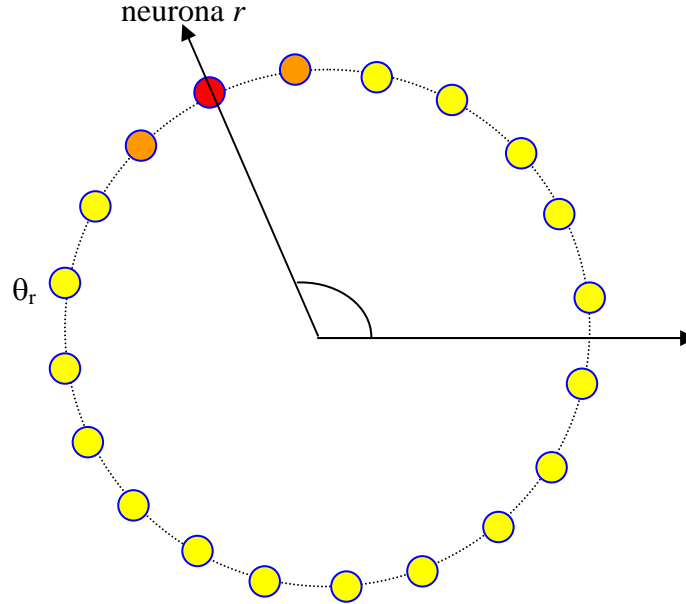


Figura 10. Topología de la Red autoorganizada.

La dinámica de computación es la siguiente: Se determina la unidad ganadora, r , es decir, la de mayor potencial sináptico, $h_r \geq h_i$, siendo

$$h_i = \sum_{j=1}^N w_{ij} x_j - \sum_{j=1}^N w_{ij}^2 / 2$$

Se utiliza la *regla de aprendizaje*

$$\Delta \mathbf{w}_i(k) = \eta \Lambda(\theta_r, \theta_i) (\mathbf{x} - \mathbf{w}_i), \quad i=1,2,\dots,N,$$

tomando como conjunto de patrones de entrenamiento el conjunto de 322 puntos: $\{(x_i, y_i), i=1,2,\dots,322\}$ que corresponden a las posiciones de los píxeles del contorno en la imagen de bordes. Los vectores sinápticos serán atraídos por los puntos que configuran el contorno del vaso sanguíneo y nos darán los vértices del contorno poligonal (figura 11).

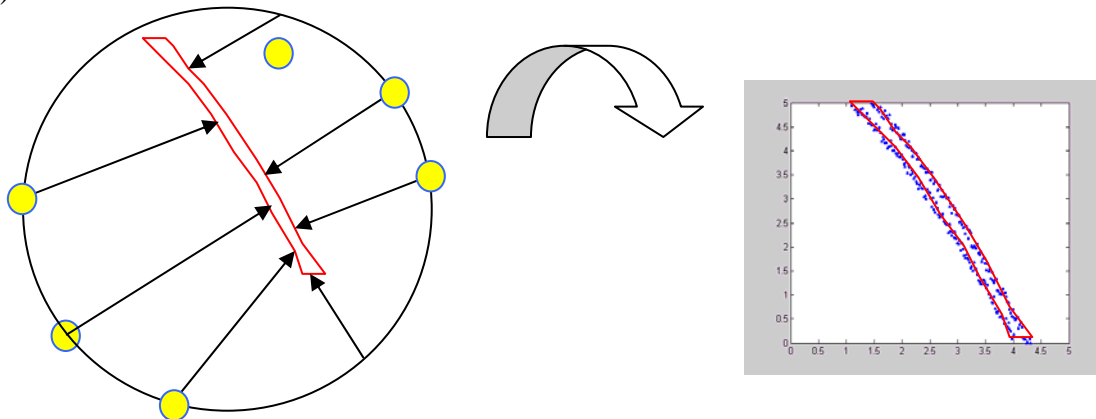


Figura 11. Ajuste al contorno de vaso sanguíneo de una mamografía.

6.4 Redes Neuronales ART (Adaptive Resonance Theory)

En las redes competitivas es necesario que la tasa o ritmo de aprendizaje vaya decreciendo gradualmente hasta hacerse igual a cero para garantizar la estabilidad de la red, pues en caso contrario puede ocurrir que el grupo que se va asignando a un mismo patrón de entrada vaya cambiando indefinidamente en etapas sucesivas. Sin embargo, esta estrategia conduce a que cuando la tasa de aprendizaje llegue a ser pequeña la red no se adapta bien a los nuevos patrones de entrada, es decir, va perdiendo plasticidad (capacidad de reaccionar para adaptarse a nuevos patrones). Esto conduce al **dilema de estabilidad-plasticidad** planteado por Grossberg.

Los modelos de redes neuronales ART permiten formar grupos o categorías a partir de una secuencia de patrones de entrada no etiquetados, es decir, que no conocemos la clase a la que pertenecen, y además resuelven el *dilema de la estabilidad-plasticidad* de Grossberg, pues la red permite adaptarse a nuevos patrones evitando que su información sea aniquilada por la información de todos los patrones de entrenamiento anteriormente utilizados cuando trata de buscar la estabilidad del proceso. Estas redes van a realizar la agrupación según la similitud de los patrones de entrada, de manera que los patrones que constituyan un mismo grupo sean similares entre sí. Asimismo, tampoco se tiene que establecer a priori el número de grupos que forman los patrones de entrada, ya que la propia red se encargará de determinarlos siguiendo un proceso que se controla mediante un parámetro de vigilancia.

La **red neuronal ART1** consta de una capa de sensores de entrada binarios, es decir, los patrones de entrada tienen que ser binarios, $\mathbf{x} \in \{0,1\}^N$, y las unidades de proceso son lineales con vectores sinápticos también binarios que representan patrones prototipo. Se activa sólo la unidad de proceso cuyo vector de pesos sinápticos (prototipo) es más parecido al patrón de entrada. Si la unidad de proceso activada (ganadora) no representa bien a dicho patrón de entrada (lo que se comprueba mediante un test de similitud) entonces se añade una nueva unidad de proceso cuyo vector de pesos sinápticos es el propio patrón de entrada; si la unidad de proceso activada representa bien al patrón de entrada, según el test de similitud, entonces su vector de pesos sinápticos se modifica acercándolo al patrón de entrada.

Para cada patrón de entrada, $\mathbf{x} \in \{0,1\}^N$, presentado a la red, se calcula el potencial sináptico normalizado de cada unidad de proceso. El potencial sináptico normalizado de la unidad de proceso i viene dado por la expresión:

$$h_i = \frac{\mathbf{w}_i' \mathbf{x}}{\|\mathbf{w}_i\|^2} = \bar{\mathbf{w}}_i' \mathbf{x}$$

siendo $\|\mathbf{w}_i\|^2 = w_{i1}^2 + w_{i2}^2 + \dots + w_{iN}^2 = w_{i1} + w_{i2} + \dots + w_{iN}$, ya que los pesos sinápticos son también vectores binarios, y $\bar{\mathbf{w}}_i = \frac{\mathbf{w}_i}{\|\mathbf{w}_i\|^2}$ es el vector sináptico normalizado de la unidad

de proceso i . La unidad ganadora es la de mayor potencial sináptico normalizado, h_i , $i=1,2,\dots$. Sea r la unidad ganadora. Con el fin de comprobar si el vector sináptico de la unidad ganadora, \mathbf{w}_r , representa de forma adecuada al patrón de entrada \mathbf{x} (está lo bastante próximo) utilizamos un primer test, que consiste en comprobar si

$$h_i = \frac{\mathbf{w}_i' \mathbf{x}}{\|\mathbf{w}_i\|^2} > \frac{\|\mathbf{x}\|^2}{N}$$

Es decir, si hay una fracción suficiente de unos en el vector \mathbf{w} emparejados con los unos de \mathbf{x} , puesto que $\|\mathbf{w}_i\|^2$ nos da el número de unos que tiene el vector \mathbf{w}_i .

Si no se pasa este primer test es porque el vector de entrada no está bien representado por los vectores sinápticos existentes, por lo que se habilita una nueva unidad de proceso que tendrá como vector sináptico el propio patrón de entrada \mathbf{x} . En caso contrario (pasa el primer test) comprobamos si además pasa un segundo test que se ocupa también de verificar el emparejamiento adecuado del patrón de entrada con el vector sináptico de la unidad ganadora. Se pasa este segundo test si

$$\frac{\mathbf{w}_r' \mathbf{x}}{\|\mathbf{x}\|^2} \geq \rho$$

Es decir, se declara el vector \mathbf{w}_r emparejado con el vector de entrada \mathbf{x} si una fracción significativa de unos de \mathbf{x} (determinada por el parámetro ρ) están también en \mathbf{w}_r . El parámetro $\rho \in (0,1)$, llamado **parámetro de vigilancia**, controla la granularidad de la agrupación que forma la red. Así, valores pequeños de ρ permiten grandes desviaciones del prototipo que conducen a un número pequeño de grupos (grupos más grandes), mientras que valores grandes de ρ conducen a agrupaciones con muchos grupos pero de menos elementos, y más próximos entre sí los patrones de cada grupo.

Por lo tanto, el primer test tiene en cuenta el número de unos de \mathbf{w}_r que también tiene \mathbf{x} con respecto al número de unos de \mathbf{w}_r mientras que el segundo test tiene en cuenta el número de unos de \mathbf{x} que también están en \mathbf{w}_r con respecto al número de unos de \mathbf{x} .

Cuando se pasa también este segundo test se dice que la red está en **resonancia** y entonces se modifica el vector sináptico de la unidad ganadora según la siguiente regla de aprendizaje:

$$\mathbf{w}_r^{nuevo} = \mathbf{w}_r \wedge \mathbf{x}$$

donde \wedge es la operación conjunción lógica (AND) componente a componente. Así, el nuevo vector sináptico preserva su naturaleza binaria y sólo conserva los unos que tiene en común con \mathbf{x} . Esta reducción sucesiva de unos en el proceso de aprendizaje puede conducir a que un patrón anterior salga de un grupo porque otro patrón se ha unido a él.

Cuando no se pasa este segundo test (habiendo pasado el primero) entonces la unidad de proceso ganadora se desactiva y se activa como ganadora aquella que tiene mayor potencial sináptico h_i entre la restantes y se repite el proceso de verificación con los tests. Si esta situación persiste después de que se haya ido tomando sucesivamente cada una de las unidades de proceso entonces se habilita una nueva unidad de proceso cuyo vector sináptico es el patrón de entrada.

Una característica importante de esta red es su capacidad para aprender continuamente al mismo tiempo que es estable para un conjunto finito de patrones de entrenamiento, independientemente del valor elegido del parámetro de vigilancia.

Por lo tanto, la red ART1 no sólo realiza una agrupación no supervisada de los patrones de entrada sino que determina también el número de grupos que la configura.

Ahora vamos a estudiar una red de resonancia adaptativa, llamada ART2, que permite que las entradas de la red y los pesos sinápticos sean números analógicos (números reales, no necesariamente números binarios).

La red neuronal parte de un número reducido M de unidades de proceso. Como vectores sinápticos iniciales de las mismas se toman vectores (patrones) de entrada. Se activa sólo una unidad de proceso, llamada unidad ganadora, aquella que su vector

sináptico está más próximo al patrón de entrada, \mathbf{x} , es decir, se activa la unidad r (ganadora) si

$$\|\mathbf{w}_r - \mathbf{x}\| \leq \|\mathbf{w}_i - \mathbf{x}\|, \quad \forall i \neq r$$

A continuación la red comprueba si el patrón de entrada está bien representado por el vector sináptico de la unidad ganadora aplicando el siguiente **test de vigilancia**: La unidad de proceso r representa adecuadamente al patrón de entrada si

$$\|\mathbf{w}_r - \mathbf{x}\| < \rho$$

donde el parámetro ρ controla el grado de precisión de la representación y es el radio del grupo que se forma entorno al vector sináptico de la unidad ganadora. Conforme dicho parámetro sea mayor se formarán menos grupos y de mayor tamaño; conforme sea menor se formarán más grupos y de menor tamaño.

Si no se pasa el test se habilita una nueva unidad de proceso h con peso $\mathbf{w}_h = \mathbf{x}$, que representaría a dicho patrón y se vuelve a introducir un nuevo patrón de entrada.

Si se pasa el test entonces se actualiza sólo el vector sináptico de la unidad ganadora r según la expresión (regla de aprendizaje):

$$\mathbf{w}_r(k+1) = \frac{\mathbf{x} + |\text{grupo } r| \mathbf{w}_r(k)}{1 + |\text{grupo } r|}$$

donde $|\text{grupo } r|$ es el número de elementos del grupo configurado por la unidad de proceso r , es decir, el constituido por todos los patrones de entrada que tienen a la unidad de proceso r como unidad ganadora (\mathbf{w}_r es el vector prototipo que mejor los representa).

La regla de aprendizaje anterior nos dice que modificamos el vector sináptico de la unidad ganadora acercándolo al patrón de entrada y que dicha modificación es menor cuanto más grande sea el tamaño del grupo que configura la misma, es decir, se acerca más a \mathbf{x} cuanto mayor se la cantidad $1/(1+|\text{grupo } r|)$. Obsérvese que si $\mathbf{w}_r(k)$ es el centroide (media) del grupo r entonces $\mathbf{w}_r(k+1)$ es el centroide del nuevo grupo de $r+1$ patrones, al incorpora el patrón de entrada \mathbf{x} , pues

$$\frac{1}{r+1} \sum_{i=1}^{r+1} \mathbf{x}_i = \frac{\mathbf{x}_{r+1} + r \left(\frac{1}{r} \sum_{i=1}^r \mathbf{x}_i \right)}{r+1}.$$

El proceso se continúa hasta que se establezca la red, es decir, no se modifiquen ya los pesos sinápticos después de haber introducido, uno a uno, todos los patrones de entrenamiento.

Por lo tanto, la red ART2 no sólo realiza una agrupación no supervisada de los patrones de entrada analógicos, sino que determina también el número de grupos que la configuran.

6.5 Redes Neuronales Competitivas Supervisadas

Como vimos con anterioridad, una red competitiva está constituida por N **sensores de entrada**, M **unidades de proceso** (neuronas artificiales), y **conexiones** entre cada sensor y cada unidad de proceso, de manera que la conexión entre el sensor j y la unidad

de proceso i tiene asociado un valor w_{ij} . Para cada entrada recogida por los sensores se activa solamente una unidad de proceso, aquella que tiene el mayor potencial sináptico. Por lo tanto, la **dinámica de la computación** de la red viene dada por la expresión:

$$y_i = \begin{cases} 1 & \text{si } h_i = \max_k \{h_1, h_2, \dots, h_M\} \\ 0 & \text{en otro caso} \end{cases}, \quad i=1,2,\dots,M \quad (14)$$

donde el potencial sináptico de la unidad de proceso i viene dado por la expresión

$$h_i = w_{i1}x_1 + w_{i2}x_2 + \dots + w_{iN}x_N - \theta_i \quad (15)$$

con $\theta_i = \frac{1}{2}(w_{i1}^2 + w_{i2}^2 + \dots + w_{iN}^2)$.

Para una entrada $\mathbf{x}=(x_1, x_2, \dots, x_N)' \in R^N$ se activa la unidad de proceso i cuyo vector de pesos sinápticos $\mathbf{w}_i=(w_{i1}, w_{i2}, \dots, w_{iN})'$ está más próximo a \mathbf{x} . Así, se puede decir que la red neuronal competitiva es una función de R^N en el conjunto $\{1, 2, \dots, M\}$, que aplica un punto $(x_1, x_2, \dots, x_N)' \in R^N$ en el valor $r \in \{1, 2, \dots, M\}$, cuando r sea la unidad ganadora. Dicha función produce una partición del espacio de los datos (patrones) de entrada en M regiones disjuntas. Dicho de otra forma, la red competitiva agrupa el conjunto de datos de entrada en M grupos o clases.

¿Cómo se determinan los pesos sinápticos? En este caso disponemos de un conjunto de patrones de entrenamiento etiquetados, $\{(\mathbf{x}_i, z_i), i = 1, 2, \dots, p\}$, es decir, podemos saber si la unidad que se activa corresponde, o no, a la clase a la que pertenece el patrón de entrada. Para incorporar dicha información al proceso de aprendizaje nos basamos en la idea de acercar el vector sináptico al patrón de entrada siempre y cuando la unidad ganadora sea la de la clase correcta (como en el aprendizaje no supervisado), y en caso contrario, cuando la asignación sea incorrecta, alejamos el vector sináptico de la unidad ganadora del vector de entrada \mathbf{x} . Por lo tanto, en el proceso de aprendizaje el vector sináptico de la unidad ganadora es atraído por patrón de entrada si la clasificación es correcta, y repelido si la clasificación es incorrecta. Concretamente, cuando el patrón de entrada \mathbf{x} es de la clase s , y la unidad ganadora es la r , entonces la **regla de aprendizaje supervisado** es la siguiente:

$$\mathbf{w}_i(k+1) = \mathbf{w}_i(k) + \Delta \mathbf{w}_i(k) \quad (16)$$

donde

$$\Delta \mathbf{w}_r(k) = \begin{cases} \eta_r(k)(\mathbf{x}(k) - \mathbf{w}_r(k)) & \text{si } r = s \\ -\eta_r(k)(\mathbf{x}(k) - \mathbf{w}_r(k)) & \text{si } r \neq s \end{cases} \quad (17)$$

$$\Delta \mathbf{w}_i(k) = 0, \quad i \neq r$$

Al parámetro η_r lo llamaremos tasa de aprendizaje, pues conforme mayor sea más se modifican los pesos sinápticos. Sin embargo, la evolución de dicho parámetro durante el proceso de entrenamiento de la red no va a ser siempre decreciente como en el caso del aprendizaje no supervisado. Para mejorar la velocidad de convergencia cada unidad de proceso tiene su propia tasa de aprendizaje y evoluciona según la siguiente regla propuesta por Kohonen (1990):

$$\eta_r(k) = \begin{cases} \frac{\eta_r(k)}{1 + \eta_r(k)} & \text{si } r = s \\ \frac{\eta_r(k)}{1 - \eta_r(k)} & \text{si } r \neq s \end{cases} \quad (18)$$

Así, se disminuye la tasa de aprendizaje cuando la clasificación ha sido correcta y se aumenta en caso contrario.

Por lo tanto, vamos a tener una red neuronal con m unidades de proceso. El vector sináptico de la unidad de proceso i viene dado por un prototipo de la clase correspondiente. Los vectores sinápticos (prototipos) se van modificando según la anterior regla de aprendizaje. Sin embargo, puede ocurrir que patrones de una misma clase no estén agrupados formando un único grupo, en cuyo caso utilizar un único prototipo, es decir, una única unidad de proceso para representar una clase no sería adecuado. Si los patrones de la clase i están agrupados en K_i grupos entonces deberíamos utilizar K_i unidades de proceso para esta clase. Por lo tanto, la red neuronal estaría formada por $\sum_{i=1}^m K_i$ unidades de proceso. Se obtiene el siguiente algoritmo:

Algoritmo de aprendizaje competitivo supervisado

Paso 1: Elegir K_i prototipos iniciales para la clase i , $i=1,2,\dots,m$ (se puede utilizar para ello la red competitiva no supervisada o el algoritmo de las K -medias) que constituirán los vectores sinápticos de las unidades de proceso.

Paso 2 (k-ésima iteración): Seleccionar aleatoriamente (con reemplazamiento) un patrón del conjunto de entrenamiento, $\mathbf{x}(k)$.

Paso 3: Determinar la unidad ganadora mediante la expresión

$$h_r = \max_{j=1,\dots,M} \{h_j\}$$

Paso 4 (Fase de aprendizaje):

- Si r es la unidad ganadora y corresponde a la misma clase que la entrada $\mathbf{x}(k)$ entonces se modifica el vector sináptico de la misma según la expresión:

$$\mathbf{w}_r(k+1) = \mathbf{w}_r(k) + \eta_r(k)(\mathbf{x}(k) - \mathbf{w}_r(k))$$

Las demás unidades de proceso no modifican sus pesos.

- En otro caso se modifica el vector sináptico de la unidad ganadora según la expresión:

$$\mathbf{w}_r(k+1) = \mathbf{w}_r(k) - \eta_r(k)(\mathbf{x}(k) - \mathbf{w}_r(k))$$

Las demás unidades de proceso no modifican sus pesos.

Paso 5: Repetir el paso 2 modificando la tasa de aprendizaje de la unidad ganadora según la expresión (18).

El algoritmo fue propuesto por Kohonen (1989) y se suele llamar Aprendizaje de la Cuantificación Vectorial (Learning Vector Quantization) o algoritmo LVQ. Para entender mejor dicha regla de aprendizaje vamos a realizar la siguiente interpretación geométrica. Si r es la unidad ganadora y el patrón de entrada $\mathbf{x}(k)$ es de la misma clases entonces

$$\begin{aligned}\mathbf{w}_r(k+1) &= \mathbf{w}_r(k) + \eta(k)(\mathbf{x}(k) - \mathbf{w}_r(k)) \\ &= (1 - \eta(k))\mathbf{w}_r(k) + \eta(k)\mathbf{x}(k)\end{aligned}$$

Es decir, el nuevo vector de pesos sinápticos $\mathbf{w}_r(k+1)$ es una combinación lineal de los vectores $\mathbf{w}_r(k)$ y $\mathbf{x}(k)$. Quiere decir que el vector de pesos sinápticos se modifica acercándose al patrón de entrada, como se muestra en la figura 12. Conforme mayor es el valor del parámetro de aprendizaje más se acerca.

En otro caso, entonces

$$\begin{aligned}\mathbf{w}_r(k+1) &= \mathbf{w}_r(k) - \eta(k)(\mathbf{x}(k) - \mathbf{w}_r(k)) \\ &= (1 + \eta(k))\mathbf{w}_r(k) - \eta(k)\mathbf{x}(k)\end{aligned}$$

Es decir, el nuevo vector sináptico se retira del patrón de entrada en la dirección opuesta, como se muestra en la figura 13.

La utilización de la regla (18) para modificar la tasa de aprendizaje puede conducir a valores altos de la misma, puesto que si una unidad de proceso ganadora se equivoca varias veces consecutivas en la asignación de la clase, entonces si vale 0.100, va pasando a los valores 0.111, 0.125, 0.143, 0.167, 0.200, 0.250, 0.333, 0.50, 1, ∞ . Por ello, es mejor utilizar la siguiente regla:

$$\eta_r(k) = \begin{cases} \frac{\eta_r(k)}{1 + \eta_r(k)} & \text{si } r = s \\ \min\left\{\frac{\eta_r(k)}{1 - \eta_r(k)}, \eta_r(0)\right\} & \text{si } r \neq s \end{cases} \quad (19)$$

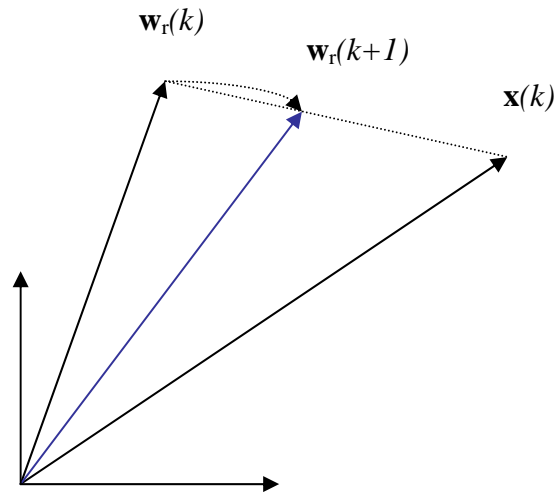


Figura 12. El nuevo vector de pesos sinápticos en caso de atracción.

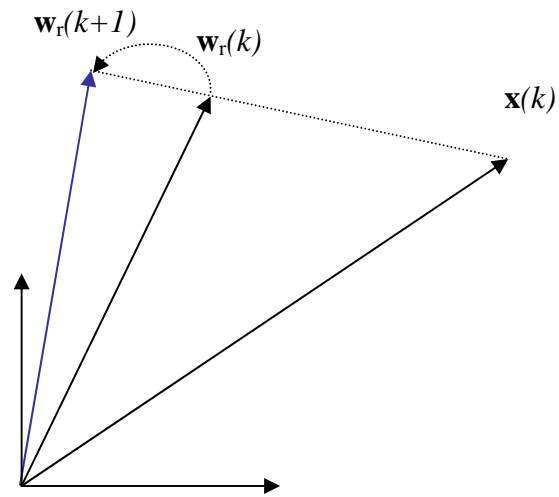


Figura 13. El nuevo vector de pesos sinápticos en caso de repulsión.