

**ESTUDIO Y APLICACIONES DE LAS REDES SOM  
(MAPAS AUTO - ORGANIZATIVOS)**

MARCELA RAMOS VARELA  
MARILUZ PÉREZ CORTÉS

UNIVERSIDAD TECNOLÓGICA DE BOLÍVAR  
FACULTAD DE INGENIERÍA ELECTRICA Y ELECTRONICA  
CARTAGENA DE INDIAS D. T. Y C.

2005

**ESTUDIO Y APLICACIONES DE LAS REDES SOM (MAPAS AUTO -  
ORGANIZATIVOS)**

MARCELA RAMOS VARELA  
MARILUZ PÉREZ CORTÉS

DIRECTOR  
ING. EDUARDO GÓMEZ VÁSQUEZ

UNIVERSIDAD TECNOLÓGICA DE BOLÍVAR  
FACULTAD DE INGENIERÍA ELECTRICA Y ELECTRONICA  
CARTAGENA DE INDIAS D. T. Y C.

2005

**ESTUDIO Y APLICACIONES DE LAS REDES SOM (MAPAS AUTO -  
ORGANIZATIVOS)**

MARILUZ PÉREZ CORTÉS

MARCELA RAMOS VARELA

Trabajo de monografía presentado como requisito para obtener el certificado del Minor en  
Automatización Industrial.

DIRECTOR

ING. EDUARDO GOMEZ VASQUEZ

Magister En Ciencias Computacionales

UNIVERSIDAD TECNOLÓGICA DE BOLÍVAR  
FACULTAD DE INGENIERÍA ELECTRICA Y ELECTRONICA  
CARTAGENA DE INDIAS D. T. Y C.

2005.

*Nota de aceptación*

---

---

---

---

*Jurado*

---

*Jurado*

---

Cartagena D. T. Y C., Diciembre de 2005.

Señores

**COMITÉ CURRICULAR**

**UNIVERSIDAD TECNOLÓGICA DE BOLÍVAR**

La ciudad.

Respetados señores:

Con toda atención me dirijo a ustedes con el fin de presentarles a su consideración, estudio y aprobación la monografía titulada **ESTUDIO Y APLICACIONES DE LAS REDES SOM (MAPAS AUTO - ORGANIZATIVOS)** como requisito para obtener el título de Ingeniero Electrónico.

Atentamente

---

MARILUZ PÉREZ CORTES

Cartagena D. T. Y C., Diciembre de 2005.

Señores

**COMITÉ CURRICULAR**

**UNIVERSIDAD TECNOLÓGICA DE BOLÍVAR**

La ciudad.

Respetados señores:

Con toda atención me dirijo a ustedes con el fin de presentarles a su consideración, estudio y aprobación la monografía titulada **ESTUDIO Y APLICACIONES DE LAS REDES SOM (MAPAS AUTO - ORGANIZATIVOS)** como requisito para obtener el título de Ingeniero Eléctrico.

Atentamente

---

MARCELA RAMOS VARELA

Cartagena D. T. Y C., Diciembre de 2005.

Señores

**COMITÉ CURRICULAR**

**UNIVERSIDAD TECNOLÓGICA DE BOLÍVAR**

La ciudad

Cordial saludo:

A través de la presente me permito entregar la monografía titulada **ESTUDIO Y APLICACIONES DE LAS REDES SOM (MAPAS AUTO - ORGANIZATIVOS)** para su estudio y evaluación, la cual fue realizada por los estudiantes MARCELA RAMOS VARELA y MARILUZ PÉREZ CORTÉS, de la cual acepto ser su director.

Atentamente,

---

ING. EDUARDO GÓMEZ VÁSQUEZ

Magíster en Ciencias Computacionales.

## **AUTORIZACIÓN**

Yo, MARCELA RAMOS VARELA, identificada con la cédula de ciudadanía número 45.554.930 de Cartagena, autorizo a la Universidad Tecnológica de Bolívar, para hacer uso de mi trabajo de monografía y publicarlo en el catalogo on- line de la biblioteca.

---

MARCELA RAMOS VARELA



## **AUTORIZACIÓN**

Yo, MARILUZ PÉREZ CORTÉS, identificada con la cédula de ciudadanía número 57.308.334 de Pivijay (Magdalena), autorizo a la Universidad Tecnológica de Bolívar, para hacer uso de mi trabajo de monografía y publicarlo en el catalogo on- line de la biblioteca.

---

MARILUZ PÉREZ CORTÉS

## AGRADECIMIENTOS

*Agradezco a Dios por todo lo que me ha dado.*

*Agradezco a mis profesores, mis compañeros, mis amigos, por su apoyo y colaboración.*

*Agradezco a mi familia: a mis padres Héctor y Norma, y a mi hermana Lilitiana. A ellos les dedico todos mis esfuerzos, todos mis logros.*

*Marcela Ramos Varela*

## AGRADECIMIENTOS

*Agradezco a Dios porque sin Él nuestra existencia es muerte.*

*Agradezco a Álvaro, que donde se encuentre, siempre me impulsa a vivir.*

*Agradezco a Mauricio, su presencia cambió mi vida.*

*Agradezco a mi familia: a mi madre Norma, a mi hermano Álvaro José, a mis abuel@s, tí@s y prim@s, ellos siempre han estado ahí para apoyarme. A ellos les dedico este logro.*

*Agradezco a mis amigos, compañeros y profesores, sin su apoyo y colaboración este logro no tendría sentido.*

*Mariluz*

## TABLA DE CONTENIDO

|  |           |
|--|-----------|
| <b>LISTA DE GRAFICAS</b>   | <b>14</b> |
| <b>INTRODUCCIÓN</b>  | <b>16</b> |
| <b>CAPITULO 1 Generalidades de las Redes Neuronales Artificiales</b> | <b>18</b> |
| <b>1.1 Definiciones de una red neuronal.</b>                         | <b>19</b> |
| <b>1.2 Ventajas que ofrecen las redes neuronales.</b>                | <b>19</b> |
| <b>1.3 Elementos básicos que componen una red neuronal.</b>          | <b>20</b> |
| <b>1.4 Función de entrada (input function).</b>                      | <b>22</b> |
| <b>1.5 Función de activación (activation function).</b>              | <b>22</b> |
| <b>1.6 Función de salida (output function)</b>                       | <b>24</b> |
| <b>1.7 Mecanismos de aprendizaje</b>                                 | <b>25</b> |
| 1.7.1 Aprendizaje supervisado.                                       | 27        |
| 1.7.2 Aprendizaje no supervisado.                                    | 27        |
| <b>1.8 Principales Topologías</b>                                    | <b>28</b> |
| <b>CAPÍTULO 2 Mapas Auto- Organizados SOM</b>                        | <b>30</b> |
| <b>2.1 Redes de Kohonen</b>  | <b>31</b> |
| <b>2.2 Generalidades</b>   | <b>32</b> |
| 2.2.1 Redes Autoorganizadas  | 36        |
| 2.2.2 Tipos de Redes Autoorganizadas (No Supervisadas)               | 38        |
| 2.2.3 Redes Competitivas   | 39        |
| <b>2.3 Arquitectura SOM</b>  | <b>43</b> |
| <b>2.4 Algoritmo SOM</b>   | <b>44</b> |
| 2.4.1 Etapa de funcionamiento  | 45        |

|   |            |
|---|------------|
| 2.4.2 Etapa de aprendizaje _____  | 47         |
| 2.4.3 Interpretación del algoritmo de aprendizaje _____                     | 51         |
| <b>CAPÍTULO 3 Ejemplos de ejecución de las SOM. _____</b>                   | <b>58</b>  |
| <b>3.1 Demostración SOM_DEMO1 Comportamiento y propiedades de SOM _____</b> | <b>58</b>  |
| 3.1.1 Objetivos: _____  | 58         |
| 3.1.2 Funciones _____   | 58         |
| 3.1.3 Desarrollo _____  | 59         |
| <b>3.2 Demostración SOM_DEMO2 Uso Básico del ToolBox SOM _____</b>          | <b>68</b>  |
| 3.2.1 Objetivo: _____   | 68         |
| 3.2.2 Funciones _____   | 69         |
| 3.2.3 Desarrollo _____  | 69         |
| <b>3.3 Demostración SOM_DEMO3 Visualización _____</b>                       | <b>80</b>  |
| 3.3.1 Objetivo: _____   | 80         |
| 3.3.2 Funciones _____   | 81         |
| 3.3.3 Desarrollo _____  | 81         |
| <b>3.4 Demostración SOM_DEMO4 Análisis de datos usando SOM _____</b>        | <b>82</b>  |
| 3.4.1 Objetivo: _____   | 82         |
| 3.4.2 Desarrollo _____  | 82         |
| <b>3.5 Programas simuladores de redes neuronales _____</b>                  | <b>90</b>  |
| <b>CAPÍTULO 4 Aplicaciones de las SOM. _____</b>                            | <b>95</b>  |
| <b>4.1 Visualmaps De Xia Lin. _____</b>                                     | <b>95</b>  |
| <b>4.2 ET-MAP _____</b>   | <b>98</b>  |
| <b>4.3 WEBSOM _____</b>   | <b>100</b> |
| <b>4.4 Los mapas autoorganizados aplicados a la bibliometría. _____</b>     | <b>106</b> |
| <b>CONCLUSIONES _____</b>   | <b>112</b> |
| <b>BIBLIOGRAFÍA _____</b>   | <b>115</b> |

## LISTA DE GRAFICAS

|   |    |
|---|----|
| <i>FIGURA 1 EJEMPLO DE UNA RED NEURONAL TOTALMENTE CONECTADA.</i>                             | 20 |
| <i>FIGURA 2 FUNCIÓN DE ACTIVACIÓN LINEAL.</i>   | 23 |
| <i>FIGURA 3 FUNCIÓN DE ACTIVACIÓN SIGMOIDEA.</i>  | 24 |
| <i>FIGURA 4 FUNCIÓN DE ACTIVACIÓN TANGENTE HIPERBÓLICA.</i>                                   | 24 |
| <i>FIGURA 5 ARQUITECTURA LVQ.</i>   | 31 |
| <i>FIGURA 6 ARQUITECTURA SOM.</i>   | 32 |
| <i>FIGURA 7 TOPOLOGIA RED WINNER- TAKE- ALL.</i>  | 41 |
| <i>FIGURA 8 ARQUITECTURA DE LA CAPA DE SALIDA DE SOM.</i>                                     | 43 |
| <i>FIGURA 9 PROCESO DE APRENDIZAJE EN DOS DIMENSIONES.</i>                                    | 51 |
| <i>FIGURA 10 RADIOS DE VECINDAD DE LA FUNCIÓN ESCALÓN.</i>                                    | 54 |
| <i>FIGURA 11 FORMAS DE LA FUNCIÓN DE VECINDAD: FUNCIÓN GAUSSIANA (I).</i>                     | 55 |
| <i>FIGURA 12 FORMAS DE LA FUNCIÓN DE VECINDAD (II).</i>                                       | 55 |
| <i>FIGURA 13 CONFIGURACIONES DE NEURONAS: HEXAGONAL Y RECTANGULAR.</i>                        | 60 |
| <i>FIGURA 14 MAPAS CORRESPONDIENTES AL ESPACIO DE ENTRADAS Y AL ESPACIO DE SALIDAS.</i>       | 61 |
| <i>FIGURA 15 VISUALIZACIÓN DEL ENTRENAMIENTO DEL MAPA.</i>                                    | 62 |
| <i>FIGURA 16 DATOS A TRABAJAR, Y NUEVO MAPA.</i>  | 63 |
| <i>FIGURA 17 VISUALIZACIÓN DEL ENTRENAMIENTO LUEGO DE 300 ITERACIONES.</i>                    | 64 |
| <i>FIGURA 18 DATOS DEL ESPACIO DE ENTRADAS CÚBICO.</i>  | 65 |
| <i>FIGURA 19 VISUALIZACIÓN DE RESULTADOS DESPUÉS DE LA INICIALIZACIÓN Y EL ENTRENAMIENTO.</i> | 66 |
| <i>FIGURA 20 CÁLCULO DE LA BMU.</i>   | 67 |
| <i>FIGURA 21 HISTOGRAMAS Y GRÁFICAS DE LAS VARIABLES IRIS.</i>                                | 70 |
| <i>FIGURA 22 VISUALIZACIÓN BÁSICA DEL SOM.</i>  | 73 |
| <i>FIGURA 23 INDICACIÓN DE UNA UNIDAD SOBRE CADA EJE.</i>                                     | 74 |
| <i>FIGURA 24 VISUALIZACIÓN DE U-MATRIX</i>  | 75 |
| <i>FIGURA 25 VISUALIZACIÓN DE U-MATRIX Y DE LAS ETIQUETAS ADICIONADAS.</i>                    | 75 |
| <i>FIGURA 26 HISTOGRAMA DE EVENTOS SOBRE U-MATRIX.</i>  | 76 |

|  |            |
|--|------------|
| <i>FIGURA 27 HISTOGRAMAS CORRESPONDIENTES A CADA TIPO DE FLOR.....</i>   | <i>77</i>  |
| <i>FIGURA 28 VISUALIZACIONES OBTENIDAS. ....</i>   | <i>79</i>  |
| <i>FIGURA 29 PRIMERA VISUALIZACIÓN DE LOS CLUSTERS. ....</i>   | <i>83</i>  |
| <i>FIGURA 30 VISUALIZACIÓN DEL CÓDIGO DE COLORES, PROYECCIÓN Y CATEGORÍAS.<br/>.....</i>   | <i>84</i>  |
| <i>FIGURA 31 SCATTERED PLOTS E HISTOGRAMA DE EVENTOS.....</i>  | <i>85</i>  |
| <i>FIGURA 32 INFORMACIÓN DE LA CLASIFICACIÓN Y RESULTADOS DE LA DIVISIÓN. ..</i>   | <i>87</i>  |
| <i>FIGURA 33 RESULTADO DE LA FUNCIÓN DE PROBABILIDAD PARA LA PRIMERA<br/>MUESTRA. ....</i>   | <i>88</i>  |
| <i>FIGURA 34 CLASIFICADOR PARA EL CONJUNTO DE DATOS IRIS.....</i>  | <i>89</i>  |
| <i>FIGURA 35. VISUAL SITEMAP REALIZADO POR XIA LIN DE LOS DOCUMENTOS<br/>PERTENECIENTES A LA CATEGORÍA SPACE SCIENCE DE YAHOO<br/>[HTTP://LISLIN.GWS.UKY.EDU/SITEMAP/SPACE SMALL.HTM] .....</i>                          | <i>98</i>  |
| <i>FIGURA 36. MAPA SENSITIVO GENERAL REALIZADO EN EL PROYECTO ET-MAP BAJO<br/>LA DIRECCIÓN DEL PROFESOR CHEN<br/>[HTTP://AI2.BPA.ARIZONA.EDU/ENT/ENTERTAIN1/].....</i>   | <i>100</i> |
| <i>FIGURA 37 VISIÓN PARCIAL DEL MAPA DE CARACTERÍSTICAS DE PALABRAS<br/>GENERADO POR EL EQUIPO DE KOHONEN A PARTIR DE UNA COLECCIÓN DE<br/>CUENTOS DE LOS HERMANOS GRIMM. ....</i>                                       | <i>101</i> |
| <i>FIGURA 38 MAPA DE PRIMER NIVEL CORRESPONDIENTE AL WEBSOM APLICADO<br/>AL GRUPO DE DISCUSIÓN DE USENET D COMP.AI.NEURONAL-NETS<br/>[HTTP://WEBSOM.HUT.FI/WEBSOM/COMP.AI.NEURAL-NETS-NEW/HTML/ROOT.<br/>HTML] .....</i> | <i>104</i> |
| <i>FIGURA 39 MAPA DE SEGUNDO NIVEL (ENFOQUE) .....</i>   | <i>105</i> |
| <i>FIGURA 40 MAPA DE SEGUNDO NIVEL (ENFOQUE) .....</i>   | <i>106</i> |
| <i>FIGURA 41 FUNCIONAMIENTO DEL VISCOVER SOMINE.....</i>   | <i>109</i> |
| <i>FIGURA 42 POSICIÓN TECNOLÓGICA DE LAS INSTITUCIONES SEGÚN LAS<br/>CITACIONES.....</i>   | <i>110</i> |

## INTRODUCCIÓN

El hombre se ha caracterizado siempre por la búsqueda constante de nuevas vías para mejorar sus condiciones de vida, desde la reducción del trabajo en aquellas operaciones en las que la fuerza juega un papel primordial, hasta la creación de máquinas calculadoras que ayuden a resolver de forma automática y rápida determinadas operaciones que resultan tediosas cuando se realizan a mano. Sin embargo, en la construcción de las aplicaciones que brindan ayuda al hombre en la solución de problemas, se observa una limitación importante: ¿qué ocurre cuando el problema que se quiere resolver no admite un tratamiento algorítmico? Este ejemplo demuestra que la construcción de nuevas máquinas más versátiles requiere un enfoque del problema desde otro punto de vista. Los desarrollos actuales de los científicos se dirigen al estudio de las capacidades humanas como una fuente de nuevas ideas para el diseño de las nuevas máquinas. Así, la inteligencia artificial es un intento por descubrir y describir aspectos de la inteligencia humana que pueden ser simulados mediante máquinas.

La inteligencia artificial IA, la cual es entendida ampliamente como el modelado y la simulación de las actividades cognitivas complejas humanas se ha dividido en dos ramas. Una de ellas corresponde a la de las Redes Neuronales Artificiales. La idea de las redes neuronales fue concebida originalmente como un intento de modelar la biofisiología del cerebro humano, esto es, entender y explicar como funciona y opera el cerebro. La meta era crear un modelo capaz de emular el proceso humano de razonamiento.

Existen evidencias que demuestran que en el cerebro existen neuronas que se organizan en muchas zonas, de forma que las informaciones captadas del entorno a través de los órganos sensoriales se representan internamente en



forma de capas bidimensionales. A partir de estas ideas, Teuvo Kohonen presentó en 1982 un sistema con un comportamiento semejante. Se trataba de un modelo de Redes Neuronales con capacidad para formar mapas de características de manera similar a como ocurre en el cerebro. El objetivo de Kohonen era demostrar que en un estímulo externo o información de entrada por sí solo, suponiendo una estructura propia y una descripción funcional del comportamiento de la red, era suficiente para forzar la formación de los mapas correspondientes sobre la red neuronal.

El presente trabajo es una compilación de la información referente a este tipo de redes creada por Kohonen, y que reciben el nombre de mapas autoorganizados o SOM (Self Organized Map). El primer capítulo contiene los fundamentos teóricos de las redes neuronales artificiales, sus principales elementos y las topologías existentes. En el segundo capítulo, nos centramos en la teoría de las redes SOM, su organización y el mecanismo de aprendizaje que las caracteriza. El tercer capítulo es la ejecución de las demostraciones que ofrece el programa MatLab dentro de su ToolBox dedicado a este tipo de redes. Por último, el capítulo cuatro reúne las principales aplicaciones que tienen las redes SOM en los distintos campos de la vida humana, haciendo especial énfasis en la más conocida y la de mayor tamaño, denominada WebSOM, y dirigida por Teuvo Kohonen.



# **Capítulo 1**

## Generalidades de las Redes Neuronales Artificiales

*Información de las Redes Neuronales Artificiales, relativa a su definición, sus diferentes componentes, y demás conceptos relacionados con la explicación de su comportamiento, tales como entrenamiento y topologías.*

## 1.1 Definiciones de una red neuronal.<sup>1</sup>

Existen numerosas formas de definir a las redes neuronales; desde las definiciones cortas y genéricas hasta las que intentan explicar más detalladamente qué son las redes neuronales. Por ejemplo:

- Una nueva forma de computación, inspirada en modelos biológicos.
- Un modelo matemático compuesto por un gran número de elementos procesales organizados en niveles.
- Un sistema de computación compuesto por un gran número de elementos simples, elementos de procesos muy interconectados, los cuales procesan información por medio de su estado dinámico como respuesta a entradas externas.
- Redes neuronales artificiales son redes interconectadas masivamente en paralelo de elementos simples (usualmente adaptativos) y con organización jerárquica, las cuales intentan interactuar con los objetos del mundo real del mismo modo que lo hace el sistema nervioso biológico.

## 1.2 Ventajas que ofrecen las redes neuronales.

Debido a su constitución y a sus fundamentos, las redes neuronales artificiales presentan un gran número de características semejantes a las del cerebro. Por ejemplo, son capaces de aprender de la experiencia, de generalizar de casos anteriores a nuevos casos, de abstraer características

---

<sup>1</sup> Comparar también con la pagina: [http://es.wikipedia.org/wiki/Red\\_neuronal\\_artificial](http://es.wikipedia.org/wiki/Red_neuronal_artificial) enciclopedia de libre consulta, o con la pagina [http://www.iiia.csic.es/~mario/rna/tutorial/RNA\\_intro.html](http://www.iiia.csic.es/~mario/rna/tutorial/RNA_intro.html) tutorial de Mario Gomez Martinez del Instituto de investigación de inteligencia artificial.

esenciales a partir de entradas que representan información irrelevante, etc. Esto hace que ofrezcan numerosas ventajas y que este tipo de tecnología se esté aplicando en múltiples áreas. Entre las ventajas se incluyen:

- **Aprendizaje Adaptativo:** Capacidad de aprender a realizar tareas basadas en un entrenamiento o en una experiencia inicial.
- **Auto-organización:** Una red neuronal puede crear su propia organización o representación de la información que recibe mediante una etapa de aprendizaje.
- **Tolerancia a fallos:** La destrucción parcial de una red conduce a una degradación de su estructura; sin embargo, algunas capacidades de la red se pueden retener, incluso sufriendo un gran daño.
- **Operación en tiempo real:** Los cómputos neuronales pueden ser realizados en paralelo; para esto se diseñan y fabrican máquinas con hardware especial para obtener esta capacidad.
- **Fácil inserción dentro de la tecnología existente:** Se pueden obtener chips especializados para redes neuronales que mejoran su capacidad en ciertas tareas. Ello facilitará la integración modular en los sistemas existentes.

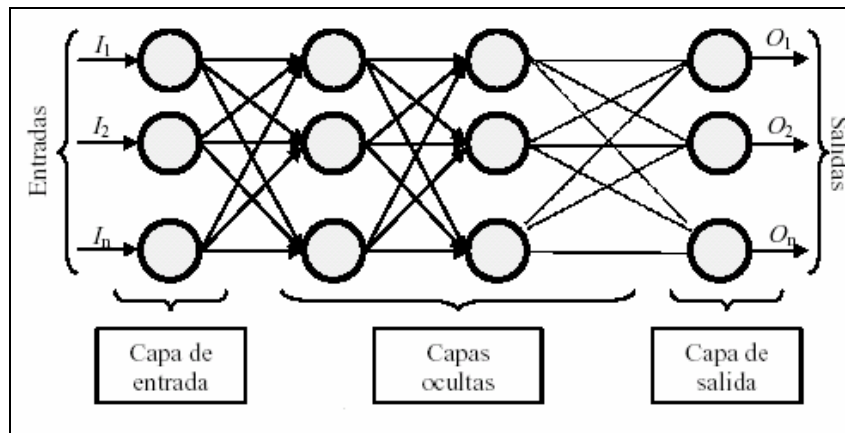
### **1.3 Elementos básicos que componen una red neuronal.<sup>2</sup>**

A continuación se puede ver en la Figura 1 un esquema de una red neuronal:

*Figura 1 Ejemplo de una red neuronal totalmente conectada.*

---

<sup>2</sup> Comparar también con él artículo: “**Implicancias del Data Mining**” capítulo vii - redes neuronales y algoritmos genéticos, autor: *ASS Silvia M. Aranguren - L. Muzachiodi*



La misma está constituida por neuronas interconectadas y arregladas en tres capas (esto último puede variar). Los datos ingresan por medio de la “*capa de entrada*”, pasan a través de la “*capa oculta*” y salen por la “*capa de salida*”. Cabe mencionar que la capa oculta puede estar constituida por varias capas. Entonces, La distribución de neuronas dentro de la red se realiza formando niveles o capas, con un número determinado de dichas neuronas en cada una de ellas. A partir de su situación dentro de la red, se pueden distinguir tres tipos de capas:

- De entrada: es la capa que recibe directamente la información proveniente de las fuentes externas de la red.
- Ocultas: son internas a la red y no tienen contacto directo con el entorno exterior. El número de niveles ocultos puede estar entre cero y un número elevado. Las neuronas de las capas ocultas pueden estar interconectadas de distintas maneras, lo que determina, junto con su número, las distintas topologías de redes neuronales.
- De salidas: transfieren información de la red hacia el exterior.

#### 1.4 Función de entrada (input function).<sup>3</sup>

La neurona trata a muchos valores de entrada como si fueran uno solo; esto recibe el nombre de *entrada global*. Por lo tanto, ahora nos enfrentamos al problema de cómo se pueden combinar estas simples entradas ( $in_1, in_2, \dots$ ) dentro de la entrada global,  $gin_i$ . Esto se logra a través de la *función de entrada*, la cual se calcula a partir del *vector entrada*. La función de entrada puede describirse como sigue:

$$input_1 = (in_{t1} \bullet w_{t1}) * (in_{t2} \bullet w_{t2}) * \dots (in_{tn} \bullet w_{tn})$$

Donde: \* representa al operador apropiado (por ejemplo: máximo, sumatoria, productoria, etc.),  $n$  al número de entradas a la neurona  $N_i$  y  $w_i$  al peso. Los valores de entrada se multiplican por los pesos anteriormente ingresados a la neurona. Por consiguiente, los pesos que generalmente no están restringidos cambian la medida de influencia que tienen los valores de entrada. Es decir, que permiten que un gran valor de entrada tenga solamente una pequeña influencia, si estos son lo suficientemente pequeños.

#### 1.5 Función de activación (activation function).<sup>4</sup>

Las neuronas artificiales tienen diferentes estados de activación; algunas de ellas solamente dos, activa e inactiva, al igual que las neuronas biológicas, pero otras pueden tomar cualquier valor dentro de un conjunto determinado. La *función activación* calcula el estado de actividad de una neurona; transformando la entrada global (menos el umbral,  $\Theta_i$ ) en un valor (estado) de activación, cuyo rango normalmente va de (0 a 1) o de (-1 a 1). Esto es así,

---

<sup>3</sup> Comparar también con él artículo: Redes Neuronales: Conceptos Básicos y Aplicaciones.  
Del autor: Damián Jorge Matich

<sup>4</sup> Comparar también con Redes Neuronales: Inteligencia por aprendizaje de Pascual  
Campoy U.P.M. - DISAM

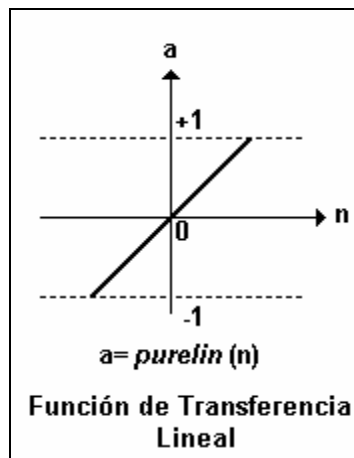
porque una neurona puede estar totalmente inactiva (0 o -1) o activa (1). La función activación, es una función de la entrada global  $gin_i$  menos el umbral  $\Theta_i$ . Las funciones de activación más comúnmente utilizadas se detallan a continuación:

- Función lineal: Los valores de salida obtenidos por medio de esta función de activación serán:

$$f(x) = \begin{cases} -1 & x \leq -1/a \\ a * x & -1/a < x \leq 1/a \\ 1 & x \geq 1/a \end{cases}$$

con  $x = gin_i - \Theta_i$ , y  $a > 0$

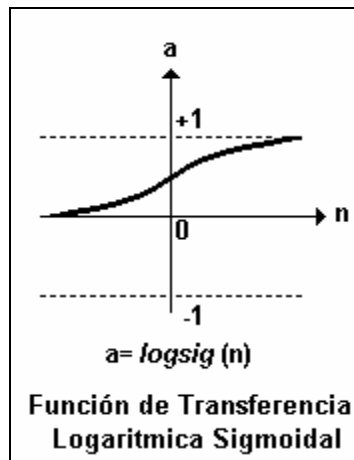
Figura 2 Función de activación lineal.



- Función sigmoidea: Los valores de salida que proporciona esta función están comprendidos dentro de un rango que va de 0 a 1. Al modificar el valor de  $g$  se ve afectada la pendiente de la función de activación.

$$f(x) = \frac{1}{1 + e^{-gx}}, \text{ con } x = gin_i - \Theta_i$$

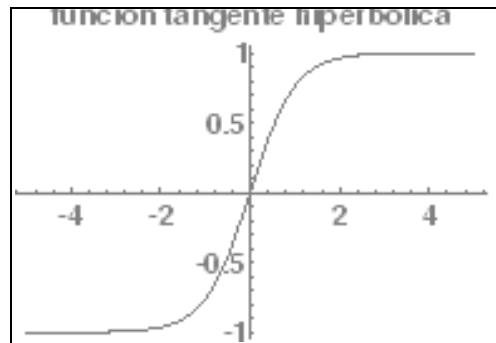
Figura 3 Función de activación sigmoidea



- Función tangente hiperbólica: Los valores de salida de la función tangente hiperbólica están comprendidos dentro de un rango que va de -1 a 1. Al modificar el valor de  $g$  se ve afectada la pendiente de la función de activación.

$$f(x) = \frac{e^{gx} - e^{-gx}}{e^{gx} + e^{-gx}}, \text{ con } x = gin_i - \Theta_i$$

Figura 4 Función de activación tangente hiperbólica.



### 1.6 Función de salida (output function)

El último componente que una neurona necesita es la *función de salida*. El valor resultante de esta función es la salida de la neurona  $i$ ,  $out_i$ ; por ende, la



función de salida determina que valor se transfiere a las neuronas vinculadas. Si la función de activación está por debajo de un umbral determinado, ninguna salida se pasa a la neurona subsiguiente. Normalmente, no cualquier valor es permitido como una entrada para una neurona, por lo tanto, los valores de salida están comprendidos en el rango  $[0, 1]$  o  $[-1, 1]$ . También pueden ser binarios  $\{0, 1\}$  o  $\{-1, 1\}$ . Dos de las funciones de salida más comunes son:

- Ninguna: este es el tipo de función más sencillo, tal que la salida es la misma que la entrada. Es también llamada *función identidad*.
- Binaria

$$\begin{cases} 1 & \text{si } act_i \geq \xi_i \\ 0 & \text{de lo contrario} \end{cases}, \text{ donde } \xi_i \text{ es el umbral.}$$

## 1.7 Mecanismos de aprendizaje

Se ha visto que los datos de entrada se procesan a través de la red neuronal con el propósito de lograr una salida. Una red neuronal debe entonces aprender a calcular la salida correcta para cada constelación (arreglo o vector) de entrada en el conjunto de ejemplos. Este proceso de aprendizaje se denomina: *proceso de entrenamiento o acondicionamiento*. El conjunto de datos (o conjunto de ejemplos) sobre el cual este proceso se basa es, por ende, llamado: *conjunto de datos de entrenamiento*. Si la topología de la red y las diferentes funciones de cada neurona (entrada, activación y salida) no pueden cambiar durante el aprendizaje, mientras que los pesos sobre cada una de las conexiones si pueden hacerlo; el aprendizaje de una red neuronal significa: *adaptación de los pesos*.

En otras palabras el aprendizaje es el proceso por el cual una red neuronal modifica sus pesos en respuesta a una información de entrada. Los cambios

que se producen durante el mismo se reducen a la destrucción, modificación y creación de conexiones entre las neuronas. En los sistemas biológicos existe una continua destrucción y creación de conexiones entre las neuronas. En los modelos de redes neuronales artificiales, la creación de una nueva conexión implica que el peso de la misma pasa a tener un valor distinto de cero. De la misma manera, una conexión se destruye cuando su peso pasa a ser cero. Durante el proceso de aprendizaje, los pesos de las conexiones de la red sufren modificaciones, por lo tanto, se puede afirmar que este proceso ha terminado (la red ha aprendido) cuando los valores de los pesos permanecen estables ( $dw_{ij}/dt = 0$ ). Un aspecto importante respecto al aprendizaje de las redes neuronales es el conocer cómo se modifican los valores de los pesos, es decir, cuáles son los criterios que se siguen para cambiar el valor asignado a las conexiones cuando se pretende que la red aprenda una nueva información.

Hay dos métodos de aprendizaje<sup>5</sup> importantes que pueden distinguirse, según la necesidad de un maestro o modelo a seguir:

- Aprendizaje supervisado.
- Aprendizaje no supervisado.

Otro criterio que se puede utilizar para diferenciar las reglas de aprendizaje se basa en considerar si la red puede aprender durante su funcionamiento habitual o si el aprendizaje supone la desconexión de la red, es decir, su inhabilitación hasta que el proceso termine. En el primer caso, se trataría de un aprendizaje *on line*, mientras que el segundo es lo que se conoce como *off line*. Cuando el aprendizaje es *off line*, se distingue entre una *fase de aprendizaje o entrenamiento* y una *fase de operación o funcionamiento*, existiendo un conjunto de datos de entrenamiento y un conjunto de datos de

---

<sup>5</sup> Comparar también con: Redes Neuronales: Conceptos básicos y aplicaciones, Matich Damian, Universidad Tecnológica Nacional (Argentina).

test o prueba, que serán utilizados en la correspondiente fase. Además, los pesos de las conexiones permanecen fijos después que termina la etapa de entrenamiento de la red. Debido precisamente a su carácter estático, estos sistemas no presentan problemas de estabilidad en su funcionamiento.

#### 1.7.1 Aprendizaje supervisado.

El aprendizaje supervisado se caracteriza porque el proceso de aprendizaje se realiza mediante un entrenamiento controlado por un agente externo (supervisor, maestro) que determina la respuesta que debería generar la red a partir de una entrada determinada. El supervisor controla la salida de la red y en caso de que ésta no coincida con la deseada, se procederá a modificar los pesos de las conexiones, con el fin de conseguir que la salida obtenida se aproxime a la deseada. En este tipo de aprendizaje se suelen considerar, a su vez, tres formas de llevarlo a cabo, que dan lugar a los siguientes aprendizajes supervisados:

- Aprendizaje por corrección de error.
- Aprendizaje por refuerzo.
- Aprendizaje estocástico.

#### 1.7.2 Aprendizaje no supervisado.<sup>6</sup>

Las redes con aprendizaje no supervisado (también conocido como autosupervisado) no requieren influencia externa para ajustar los pesos de las conexiones entre sus neuronas. La red no recibe ninguna información por parte del entorno que le indique si la salida generada en respuesta a una determinada entrada es o no correcta. Estas redes deben encontrar las

---

<sup>6</sup> Comparar también con: " Redes no supervisadas: Mapas autoorganizados" de David Gómez Saavedra y Lorena Vélez Fernández.

características, regularidades, correlaciones o categorías que se puedan establecer entre los datos que se presenten en su entrada.

Existen varias posibilidades en cuanto a la interpretación de la salida de estas redes, que dependen de su estructura y del algoritmo de aprendizaje empleado. En algunos casos, la salida representa el grado de familiaridad o similitud entre la información que se le está presentando en la entrada y las informaciones que se le han mostrado hasta entonces (en el pasado). En otro caso, podría realizar una clusterización (clustering) o establecimiento de categorías, indicando la red a la salida a qué categoría pertenece la información presentada a la entrada, siendo la propia red quien debe encontrar las categorías apropiadas a partir de las correlaciones entre las informaciones presentadas.

En cuanto a los algoritmos de aprendizaje no supervisado, en general se suelen considerar dos tipos, que dan lugar a los siguientes aprendizajes:

- Aprendizaje hebbiano.
- Aprendizaje competitivo y comparativo.

## **1.8 Principales Topologías<sup>7</sup>**

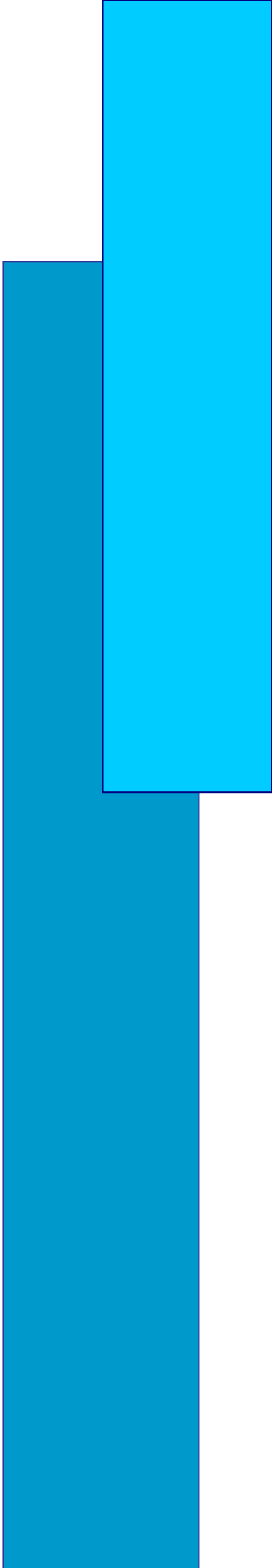
La topología o arquitectura de una red neuronal consiste en la organización y disposición de las neuronas en la misma, formando capas o agrupaciones de neuronas más o menos alejadas de la entrada y salida de dicha red. En este sentido, los parámetros fundamentales de la red son: el número de capas, el número de neuronas por capa, el grado de conectividad y el tipo de conexiones entre neuronas.

---

<sup>7</sup> Comparar también con la pagina web:  
[http://www.doc.ic.ac.uk/%7End/surprise\\_96/journal/vol4/cs11/report.html](http://www.doc.ic.ac.uk/%7End/surprise_96/journal/vol4/cs11/report.html) de Christos Stergiou y Dimitrios Siganos, resumen de la teoría de redes neuronales en ingles.

- Redes monocapa: En las redes monocapa, se establecen conexiones entre las neuronas que pertenecen a la única capa que constituye la red. Las redes monocapas se utilizan generalmente en tareas relacionadas con lo que se conoce como autoasociación (regenerar información de entrada que se presenta a la red de forma incompleta o distorsionada).
- Redes multicapa: Las redes multicapas son aquellas que disponen de un conjunto de neuronas agrupadas en varios (2, 3, etc.) niveles o capas. En estos casos, una forma para distinguir la capa a la que pertenece una neurona, consistiría en fijarse en el origen de las señales que recibe a la entrada y el destino de la señal de salida.

Normalmente, todas las neuronas de una capa reciben señales de entrada desde otra capa anterior (la cual está más cerca a la entrada de la red), y envían señales de salida a una capa posterior (que está más cerca a la salida de la red). A estas conexiones se las denomina *conexiones hacia adelante o feedforward*. Sin embargo, en un gran número de estas redes también existe la posibilidad de conectar la salida de las neuronas de capas posteriores a la entrada de capas anteriores; a estas conexiones se las denomina *conexiones hacia atrás o feedback*. Estas dos posibilidades permiten distinguir entre dos tipos de redes con múltiples capas: las redes con conexiones hacia adelante o *redes feedforward*, y las redes que disponen de conexiones tanto hacia adelante como hacia atrás o *redes feedforward/feedback*.



## **Capítulo 2**

### **Mapas Autoorganizados SOM**

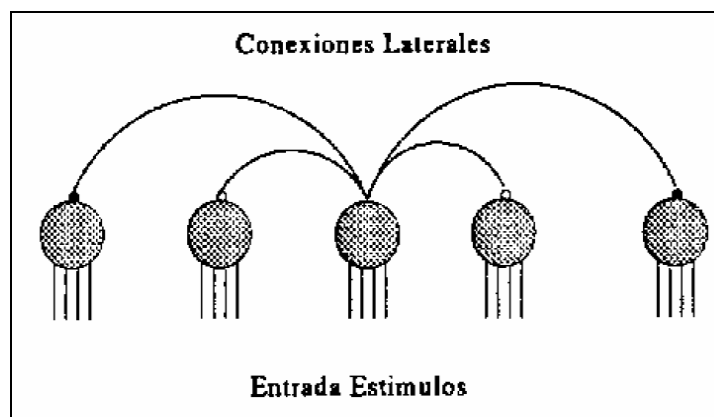
*Generalidades de los Mapas Autoorganizados SOM, haciendo énfasis en sus características de funcionamiento, entrenamiento y de organización, para finalmente trabajar el algoritmo de aprendizaje de este tipo de redes, y sus diferentes etapas.*

## 2.1 Redes de Kohonen

El objetivo de Kohonen, en la creación de su modelo de red neuronal artificial, era demostrar que un estímulo externo (información de entrada) por sí solo, suponiendo una estructura propia y una descripción funcional del comportamiento de la red, era suficiente para forzar la formación de mapas topológicos de las informaciones recibidas del exterior. El modelo tiene dos variantes:

- **LVQ<sup>8</sup>** de Learning Vector Quantization, la cual se trata de una red de dos capas con N neuronas de entrada y M de salida. Cada una de las N neuronas de entrada se conecta a las M de salida mediante conexiones hacia delante. Entre las neuronas de las capas de salida existen conexiones laterales, ya que cada una de ellas tiene influencia sobre sus vecinas a la hora de calcular los pesos de las conexiones hacia delante entre la capa de salida y la capa de entrada.

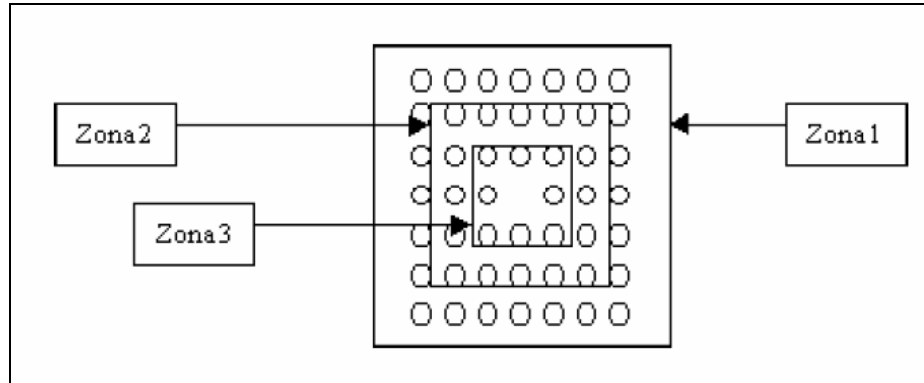
Figura 5 Arquitectura LVQ.



<sup>8</sup> Comparar también con la pagina: [http://www-etsi2.ugr.es/depar/ccia/rf/www/tema3\\_00-01\\_www/node23.html](http://www-etsi2.ugr.es/depar/ccia/rf/www/tema3_00-01_www/node23.html) "Técnicas supervisadas II: Aproximación no paramétrica" de Francisco José Cortijo Bon,

- **TPM** o **SOM** de Topologic Preserving Map o Self Organizing Map, donde los vectores de entrada a la red pueden ser de diferentes dimensiones, siendo en el caso de LVQ de una dimensión, y en el caso de SOM bidimensionales o incluso tridimensionales.

Figura 6 Arquitectura SOM.



## 2.2 Generalidades <sup>9</sup>

En el campo del estudio y aplicación de las Redes Neuronales, existe un tipo de redes que se crean para realizar una proyección  $f$  presentando a la red un número de ejemplos  $(x^P, d^P)$  donde  $d^P = f(x^P)$ . Sin embargo, cuando se realizan operaciones de este tipo, pueden aparecer problemas si los patrones de entrenamiento no están disponibles, y la única información que se proporciona corresponde a los vectores de entrada  $x^P$ . En estos casos la información relevante hay que buscarla dentro de los patrones de entrenamiento  $x^P$ . Este tipo de aprendizaje se denomina *aprendizaje no supervisado*. A diferencia de lo que sucede en el aprendizaje supervisado, en el no supervisado (o autoorganizado) no existe ningún maestro externo que indique si la red neuronal está operando correcta o incorrectamente, pues no se dispone de ninguna salida objetivo hacia la cual la red neuronal deba tender. Así, durante el proceso de aprendizaje la red autoorganizada debe descubrir por sí misma rasgos comunes, regularidades, correlaciones o



categorías en los datos de entrada, e incorporarlos a su estructura interna de conexiones. Se dice, por tanto, que las neuronas deben autoorganizarse en función de los estímulos (datos) procedentes del exterior. Los algoritmos no supervisados están normalmente basados en algún *método de competición* entre las neuronas.

En el aprendizaje competitivo las neuronas compiten unas con otras con el fin de llevar a cabo una tarea dada. Con este tipo de aprendizaje, se pretende que cuando se presente a la red un patrón de entrada, sólo una de las neuronas de salida (o un grupo de vecinas) se active. Por tanto, las neuronas compiten por activarse, quedando finalmente una como neurona vencedora y el resto anuladas, las cuales son forzadas a sus valores de respuesta mínimos.

El objetivo de este aprendizaje es categorizar (clusterizar) los datos que se introducen en la red. De esta forma, las informaciones similares son clasificadas formando parte de la misma categoría y, por tanto, deben activar la misma neurona de salida. Las clases o categorías deben ser creadas por la propia red, puesto que se trata de un aprendizaje no supervisado, a través de las correlaciones entre los datos de entrada.

Una de las propiedades del cerebro humano, es su capacidad de asociar conceptos con categorías. Se ha observado que en el córtex de los animales superiores aparecen zonas donde las neuronas detectoras de rasgos se encuentran topológicamente ordenadas, de forma que las informaciones captadas del entorno a través de los órganos sensoriales, se representan internamente en forma de mapas bidimensionales. Por ejemplo, en el área somatosensorial, las neuronas que reciben señales de sensores que se

---

<sup>9</sup> Comparar también con la página:

<http://www.gc.ssr.upm.es/inves/neural/ann2/anntutorial.html> Tutorial de redes neuronales del Grupo de investigación "Circuitos S.S.R" de la Universidad Politécnica de Madrid.

encuentran próximos en la piel se sitúan también próximas en el córtex<sup>10</sup>, de manera que reproducen --de forma aproximada--, el mapa de la superficie de la piel en una zona de la corteza cerebral.

Aunque en gran medida esta organización neuronal está predeterminada genéticamente, es probable que parte de ella se origine mediante el aprendizaje. Esto sugiere, por tanto, que el cerebro podría poseer la capacidad inherente de formar mapas topológicos de las informaciones recibidas del exterior. Por otra parte, también se ha observado que la influencia que una neurona ejerce sobre las demás es función de la distancia entre ellas, siendo muy pequeña cuando están muy alejadas. Sobre la base de este conjunto de evidencias, el modelo de red autoorganizado presentado por Kohonen pretende mimetizar de forma simplificada la capacidad del cerebro de formar mapas topológicos a partir de las señales recibidas del exterior. Según esto se puede observar que los mapas autoorganizados están más ligados a las redes biológicas que otras redes<sup>11</sup>, como por ejemplo las BAM, Hopfield o retropropagación. En la década de los 80, inspirado por algunos resultados obtenidos en estudios neurofisiológicos, Kohonen mostró cómo construir una red neuronal artificial capaz de extraer la estructura, en términos de categorías, de un conjunto de datos. De esta forma se desarrolló una red neuronal capaz de realizar tareas de clasificación. Las características principales de este tipo de red son:

- Poseen un solo nivel con muchas conexiones.
- Se necesita iniciar los pesos de estas conexiones.

---

<sup>10</sup> En el neuroencéfalo se llama cortex, o corteza, al área más superficial del cerebro constituida por estratos o capas de redes neuronales. Tales redes neuronales en el córtex macroscópicamente (a simple vista) se observan como materia gris. El córtex se encuentra particularmente desarrollado en las aves y sobre todo en los mamíferos, siendo tal área histológica la responsable de las actividades psíquicas más elaboradas.

<sup>11</sup> Comparar también con la pagina:  
[http://www.tdcat.cesca.es/TESIS\\_UPC/AVAILABLE/TDX-0416102-75520/26ApendiceD.PDF](http://www.tdcat.cesca.es/TESIS_UPC/AVAILABLE/TDX-0416102-75520/26ApendiceD.PDF)  
apéndice D “Redes neuronales y teoría de los conjuntos difusos”

- Las neuronas compiten de forma que las triunfadoras son las únicas que modifican los pesos asociados a sus conexiones.
- Estas redes agrupan las neuronas en diferentes clases, de tal forma que la topología de la red define cómo proyectar un espacio de entrada en otro, sin cambiar la configuración geométrica relativa del espacio de entrada.
- El aprendizaje es posible con la identificación de la estructura de los datos: correlaciones, agrupaciones, redundancia, etc.

Cuando los patrones de entrenamiento se presentan a la red, los pesos de las neuronas de salida se adaptan de forma que la clasificación presente en el espacio de entrada  $R^N$ , se preserva en la salida. Esto quiere decir que los patrones de entrenamiento que están próximos entre sí en el espacio de entrada (donde la proximidad se determina por alguna métrica, normalmente la distancia euclídea), deben proyectarse en las neuronas de salida que están también próximas unas a otras. Así, si las entradas están uniformemente distribuidas en  $R^N$ , y se mantiene el orden, la dimensión de  $S$  (espacio de salida) debe ser por lo menos  $N$ . Sin embargo, si las entradas están restringidas a un subespacio de  $R^N$ , se pueden usar mapas autoorganizados de menor dimensión.

Normalmente, los patrones de entrenamiento son muestras aleatorias de  $R^N$ . En un tiempo  $t$ , se genera una muestra  $x(t)$  que se presenta a la red. Usando el algoritmo de aprendizaje que se explica más adelante, se determina la neurona ganadora y se modifican sus pesos y los de sus vecinas. Con este tipo de entrenamiento se consigue que, sin necesidad de conocer la salida deseada, la respuesta de la red sea similar para vectores similares en el espacio de entrada. Este tipo de aprendizaje se denomina aprendizaje no supervisado<sup>12</sup> (al no requerir un supervisor externo), para distinguirlo del

---

<sup>12</sup> Comparar también con: "**Redes de Neuronas Artificiales. Un Enfoque Práctico**". Pedro Isasi Viñuela e Inés M. Galván León (edts). Prentice Hall. 2004 o su pagina web: <http://www.lfcia.org/~cipenedo/cursos/scx/Tema2/nodo2-1.html>

aprendizaje supervisado de las redes Perceptrón, ADALINE o Backpropagation, donde se realiza la comparación entre las salidas actuales y las salidas deseadas suministradas por un supervisor externo.

Los métodos no supervisados se suelen usar en lo denominado análisis de datos exploratorio, es decir, en una fase del análisis de los datos, cuando no se sabe de antemano cuáles son los grupos naturales que se forman, y se quiere visualizar la abundancia y la relación que hay entre los grupos "naturales"; se puede decir que una de sus principales aplicaciones es la visualización de datos multidimensionales, porque un algoritmo no supervisado actúa como una proyección de un espacio multidimensional a otro de dimensiones visualizables.

### 2.2.1 Redes Autoorganizadas<sup>13</sup>

Como se mencionó en la sección anterior, el grupo de modelos neuronales autoorganizados se caracteriza porque en su entrenamiento no se presentan salidas objetivo que se desean asociar a cada patrón de entrada. La principal aplicación de estos modelos será la realización de agrupamiento de patrones o clustering. La autoorganización consiste entonces en la modificación repetida de conexiones en respuesta a patrones de activación y siguiendo unas reglas preestablecidas, hasta el desarrollo final de la estructura o sistema. La organización en las redes neuronales tiene lugar en 2 niveles diferentes, los cuales están íntimamente interconectados en forma de lazo:

- Actividad: Ciertos patrones de actividad son producidos por la estructura en respuesta a señales de entrada.
- Conectividad: Los Pesos de las diferentes interconexiones, son modificados en respuesta a señales producidas por los patrones de entrada PE.

---

<sup>13</sup> Comparar también con Haykin Simon. Neural Networks. A comprehensive foundation. Prentice Hall 2nd. edition, 1999 [RL]

Las redes autoorganizadas se rigen bajo diferentes principios:

- Principio 1: Modificaciones en los pesos tienden a auto-amplificarse. El proceso de auto-amplificación está relacionado con el hecho de que las modificaciones de los pesos están basadas en señales variables localmente.
- Principio 2: Limitaciones debido a la competitividad establecen la selección de grupos de interconexiones fuertes, a expensas de otras.
- Principio 3: Modificaciones en los pesos tienden a una cooperación entre todos ellos.

El contexto de las redes autoorganizadas se define así:

- Existen problemas donde el conjunto de entrenamiento está formado por un conjunto de patrones de entrada y donde las salidas deseadas no están disponibles.
- En estos casos la información relevante debe de ser localizada en los propios patrones de entrada (redundancia).

En los modelos neuronales no supervisados, red autoorganizada debe descubrir por si misma rasgos comunes, regularidades, correlaciones o categorías en los datos de entrada, e incorporarlos a su estructura interna de conexiones (pesos). Se dice que las neuronas se autoorganizan en función de los estímulos procedentes del exterior. Hay varios propósitos para los que este tipo de redes son adecuados. Entre estos tenemos:

- Análisis de similitud entre patrones: Este tipo de procesamiento consta de una única neurona cuya salida es continua, indicando el grado de similitud o parecido entre el patrón de entrada actual y el promedio de los presentados en el pasado.
- Análisis de componentes principales: Si el caso anterior se extiende a varias neuronas de salida continua, el proceso de aprendizaje supone

encontrar una cierta base del espacio de entrada, representada por el conjunto de los vectores de pesos sinápticos de todas las neuronas, que se corresponderían con los rasgos más sobresalientes del espacio sensorial.

- Agrupamiento/ clasificación (clustering)<sup>14</sup>: La red realiza tareas de agrupamiento o clasificación cuando se compone de neuronas de salida discreta {0, 1}, donde cada una representa una categoría y solo una de ellas puede activarse. Ante un patrón de entrada, la neurona que se activa indica a que categoría o grupo (cluster) pertenece.
- Memoria asociativa: Es un caso general del anterior. Ahora la neurona activada no tiene salida {0, 1} sino un vector prototipo de la clase en cuestión.
- Codificación: Análogo a lo anterior pero la salida aparece codificada (Ej.: etiqueta) empleando menos bits.
- Mapas de rasgos: Son modelos no supervisados en los que las neuronas se ordenan geoméricamente (Ej.: forma de matriz bidimensional), llevando a cabo una proyección del espacio sensorial de entrada sobre la red (mapa).

### 2.2.2 Tipos de Redes Autoorganizadas (No Supervisadas)<sup>15</sup>

Estos tipos de redes se pueden clasificar en dos grandes grupos:

- *Redes no supervisadas hebbianas*: Su aprendizaje es de tipo Hebb. Su característica más relevante es que puede haber un número elevado de neuronas de salida que pueden activarse simultáneamente. La regla de Hebb se cita de la siguiente manera: cuando un axón presináptico causa

---

<sup>14</sup> Comparar también con: "Clustering of the Self-Organizing Map" de Juha Vesanto y Esa Alhoniemi, *puede obtener el documento en:* <http://lib.tkk.fi/Diss/2002/isbn951226093X/article4.pdf>

<sup>15</sup> Comparar también con la pagina: <http://electronica.com.mx/neural/informacion/> Derechos reservados ©1999 TREC Internet

la activación de cierta neurona postsináptica, la eficacia de la sinapsis que las relaciona se refuerza.

De una manera general, se denomina aprendizaje hebbiano a aquellas formas de aprendizaje que involucra una modificación en los pesos  $\Delta w_{ij}$  proporcional al producto de la entrada por la salida  $i$  de la neurona siendo  $\epsilon$  el denominado ritmo de aprendizaje, que suele ser una cantidad entre 0 y 1.

Esta expresión puede considerarse la representación matemática del modelo de aprendizaje descrito por Hebb. Muchos otros algoritmos más complejos (y más potentes) lo toman como punto de partida.

$$\Delta W_{ij} = \epsilon y_i x_j$$

- *Redes no supervisadas competitivas:* A diferencia de las anteriores solo una neurona (o un grupo de vecinas) puede ser activada. La base de estos modelos es la competición entre las neuronas. En este proceso de competición la neurona más activada conseguirá vencer a las demás. Durante la fase de aprendizaje las neuronas obtienen un premio reforzando sus conexiones sinápticas.

### 2.2.3 Redes Competitivas

Hasta ahora hemos estudiado redes en las que muchas salidas están normalmente activas a la vez. En el aprendizaje competitivo sólo una unidad de salida está activa en cada momento. Las unidades de salida compiten entre sí para ser la que se activa como respuesta a una entrada determinada. Cada neurona se especializa en un área diferente del espacio

de entradas, y sus salidas se pueden utilizar para representar de alguna manera la estructura del espacio de entradas.

Vamos a introducir topologías y reglas de aprendizaje específicos que están dirigidas a la competición. La competición se muestra como un método rápido de organización automática de los recursos de la red y se ha mostrado muy efectiva en la práctica. Stephen Grossberg introdujo la mayor parte de las ideas de las redes competitivas. Sus redes trabajan en tiempo continuo y se definen en términos matemáticos complejos. Teuvo Kohonen introduce un estilo más próximo a la ingeniería y adopta una serie de principios fácilmente implementables en sistemas digitales

La competición puede ser de dos tipos:

- Competición dura, sólo una neurona consigue los recursos
- Competición blanda, hay un vencedor claro, pero sus vecinos comparten un pequeño porcentaje de los recursos del sistema.

Una red de neuronas competitiva típica consiste en una capa de neuronas en la que todas reciben la misma entrada. La neurona que presenta la mejor salida (la máxima o la mínima según el criterio) es declarada vencedora. El concepto de elegir un vencedor a menudo requiere un controlador global que compare cada salida con todas las demás. Deseamos construir una red que encuentre la mayor o la menor salida, sin necesidad de control global. Este sistema se denomina a menudo *winner-take-all* (el ganador lo consigue todo).

### 2.2.3.1 Redes *winner-take-all*

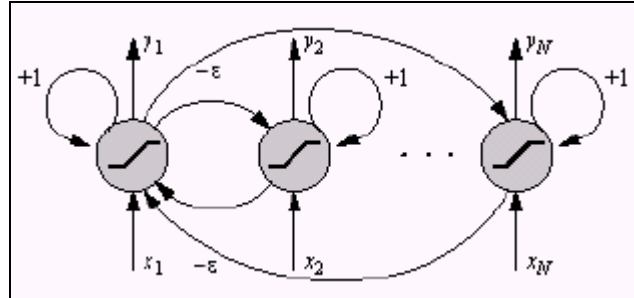
Supongamos N entradas  $x_1, x_2, \dots, x_N$ . Queremos crear una red de N salidas que dé una única salida positiva en la neurona que corresponde a la entrada más grande, mientras todas las demás neuronas presentan salida 0.

$$y_k = \begin{cases} 1 & x_k \text{ es el mayor} \\ 0 & \text{en otro caso} \end{cases}$$



Una posible topología sería:

Figura 7 Topología Red winner- take- all



Cada neurona tiene una autoalimentación con peso fijo 1 y está conectada lateralmente a todas las demás neuronas por pesos negativos  $\epsilon$  con  $0 < \epsilon < 1/N$ . Suponemos todas las entradas positivas, la pendiente de las neuronas igual a 1 y la condición inicial sin entrada es salida 0. Al presentar una entrada, la conexión lateral lleva a todas las neuronas hacia 0. Según las más pequeñas se aproximan a 0, la más grande será menos afectada por las conexiones laterales. A la vez, la autoalimentación eleva su salida, lo que refuerza la tendencia a 0 de las otras neuronas. Las neuronas compiten por la actividad de salida y sólo una lo consigue.

La salida que queremos obtener de la red es

$$y_i = \begin{cases} 1 & i=i^* \text{ la neurona vencedora} \\ 0 & \text{para todas las otras neuronas} \end{cases}$$

Por tanto la regla de aprendizaje será

$$w_{i^*}(n+1) = w_{i^*}(n) + \eta(x(n) - w_{i^*}(n))$$

Donde  $i^*$  es la neurona ganadora. Todas las demás mantienen sus pesos anteriores. El tamaño del paso  $\eta(0 < \eta < 1)$  controla el tamaño de la actualización en cada paso. Si es grande la red convergerá rápido pero será

menos estable. Se puede usar un paso variable comenzando por uno grande y disminuyéndolo progresivamente.

Las principales características de este tipo de redes se pueden resumir así:

- Sólo una salida está activada en cada momento
- Cada salida se distingue de las demás respondiendo a alguna característica.
- La red amplifica las diferencias creando un mecanismo selector que puede ser aplicado a muy diferentes problemas.
- Una aplicación típica es colocar una red de este tipo en la salida de otra red para seleccionar la salida más grande.
- De esta manera la red toma una decisión basada en la respuesta más probable.

Es trivial implementar esta operación en un sistema digital, por lo que esta arquitectura no se usa en la práctica.

#### *2.2.3.2 Competencia blanda*

Hasta ahora en la competición sólo había un ganador, una neurona permanece activa y todas las demás están inactivas. La competición blanda permite no sólo a la ganadora, sino también a sus vecinas estar activas, es decir, crea una bolsa de actividad en el espacio de salida donde la neurona más próxima es la más activa y sus vecinas están menos activas. Una red de este tipo se puede crear usando alimentación lateral. En este caso los pesos laterales varían según la distancia a la neurona vencedora. Las neuronas cercanas se excitan unas a otras, mientras que inhiben a las alejadas. Se actualizan varios pesos por cada patrón que se presenta pero atenuándose la actualización según la distancia.

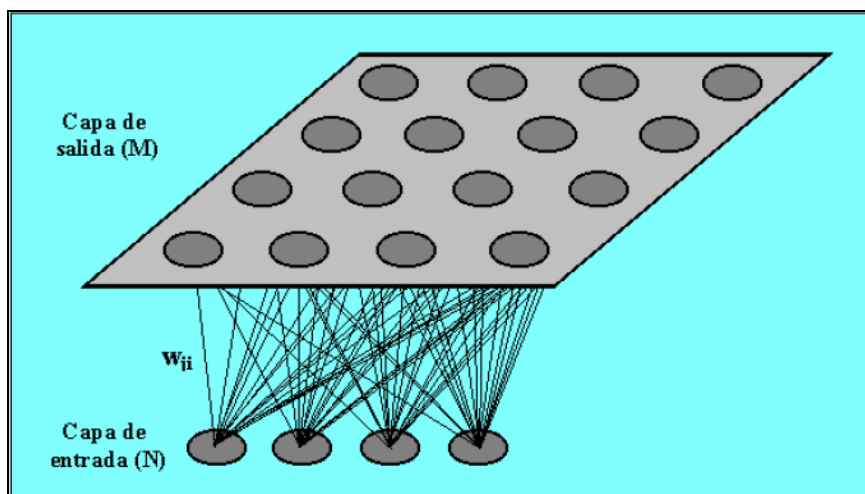
La competición blanda construye relaciones de vecindad entre las neuronas, establece una especie de métrica en el espacio de salida. Al contrario que

las estructuras anteriores, posibilita la existencia de aplicaciones topológicas, que conservan las relaciones de vecindad. Una de las aplicaciones importantes es la reducción de datos conservando la información de proximidad entre ellos. Las proyecciones creadas con esta técnica conservan en detalle la estructura celular de los datos. Si dos entradas se transforman en salidas próximas, aquellas deben también ser próximas en el espacio de entrada.

### 2.3 Arquitectura SOM

Un modelo SOM está compuesto por dos capas de neuronas. La capa de entrada (formada por  $N$  neuronas, una por cada variable de entrada) se encarga de recibir y transmitir a la capa de salida la información procedente del exterior. La capa de salida (formada por  $M$  neuronas) es la encargada de procesar la información y formar el mapa de rasgos. Normalmente, las neuronas de la capa de salida se organizan en forma de mapa bidimensional como se muestra en la figura 8, aunque a veces también se utilizan capas de una sola dimensión (cadena lineal de neuronas) o de tres dimensiones (paralelepípedo).

FIGURA 8 Arquitectura de la capa de salida de SOM.



Las conexiones entre las dos capas que forman la red son siempre hacia delante, es decir, la información se propaga desde la capa de entrada hacia la capa de salida. Cada neurona de entrada  $i$  está conectada con cada una de las neuronas de salida  $j$  mediante un peso  $w_{ji}$ . De esta forma, las neuronas de salida tienen asociado un vector de pesos  $W_j$  llamado vector de referencia (o *codebook*), debido a que constituye el vector prototipo (o promedio) de la categoría representada por la neurona de salida  $j$ .

Entre las neuronas de la capa de salida, puede decirse que existen conexiones laterales de excitación e inhibición implícitas, pues aunque no estén conectadas, cada una de estas neuronas va a tener cierta influencia sobre sus vecinas. Esto se consigue a través de un proceso de competición entre las neuronas y de la aplicación de una función denominada de vecindad.

## 2.4 Algoritmo SOM<sup>16</sup>

El algoritmo de la red neuronal SOM trata de establecer una correspondencia entre los datos de entrada y un espacio bidimensional de salida, de modo que ante datos con características comunes se activen neuronas situadas en zonas próximas de la capa de salida. El aprendizaje en el modelo de Kojonen es del tipo off-line, por lo que se distinguen dos etapas: una de funcionamiento donde se presenta, ante la red entrenada, un patrón de entrada y éste se asocia a la neurona o categoría cuyo vector de referencia es el más parecido y, por otro lado, una etapa de entrenamiento o aprendizaje donde se organizan las categorías que forman el mapa mediante un proceso no supervisado a partir de las relaciones descubiertas en el conjunto de los datos de entrenamiento.

---

<sup>16</sup> Comparar también con la página:  
<http://koti.mbnet.fi/~phodju/nenet/SelfOrganizingMap/Theory.html> de Timo Honkela, 2 de junio 1998, explicación ilustrativa del algoritmo en inglés (resumen)

#### 2.4.1 Etapa de funcionamiento

La idea básica es que las redes SOM incorporan a la regla de aprendizaje competitivo, un cierto grado de sensibilidad con respecto al conjunto de las neuronas vecinas a la ganadora. Esto hace que el proceso de aprendizaje no sea global, sino local (la modificación de los pesos se realiza en una mayor o menor grado dependiendo de la proximidad a la neurona ganadora), y esto ayuda a que se destaquen propiedades topológicas que aparecen en la proyección de características. Supongamos que un vector de entrada tiene  $N$  características, representado por un vector  $\mathbf{x}$  de un espacio  $N$ -dimensional. La red trata de proyectar el espacio de entrada sobre el de salida. El entrenamiento debe realizarse de forma que la proyección preserve el orden topológico. Kohonen propuso que las neuronas de salida interactuaran lateralmente, llegando así a los mapas de características autoorganizados. La propiedad más importante del modelo, es el concepto de aprendizaje en un vecindario próximo a la neurona ganadora. El tamaño del vecindario decrece en cada iteración.

En la etapa de funcionamiento, lo que se pretende es encontrar el vector de referencia más parecido al vector de entrada para averiguar qué neurona es la vencedora y, sobre todo, en virtud de las interacciones excitatorias e inhibitorias que existen entre las neuronas, para averiguar en qué zona del espacio bidimensional de salida se encuentra tal neurona. Por tanto, lo que hace la red SOM es realizar una tarea de clasificación, ya que la neurona de salida activada ante una entrada representa la clase a la que pertenece dicha información de entrada. Además, como ante otra entrada parecida se activa la misma neurona de salida, u otra cercana a la anterior, debido a la semejanza entre las clases, se garantiza que las neuronas topológicamente próximas sean sensibles a entradas físicamente similares. Por este motivo, la

red es especialmente útil para establecer relaciones, desconocidas previamente, entre conjuntos de datos<sup>17</sup>.

Lo que se realiza es que cuando se presenta un patrón  $p$  de entrada  $X_p$ :  $x_{p1}, \dots, x_{pi}, \dots, x_{pN}$ , éste se transmite directamente desde la capa de entrada (la cual tiene  $p$  neuronas de entrada) hacia la capa de salida (formada por  $M$  neuronas). En esta capa, cada neurona calcula la similitud entre el vector de entrada  $X_p$  y su propio vector de pesos  $W_j$  o vector de referencia según una cierta medida de distancia o criterio de similitud establecido, hasta encontrar una situación estable. A continuación, simulando un proceso competitivo, se declara vencedora la neurona cuyo vector de pesos es el más similar al de entrada.

La siguiente expresión matemática representa cuál de las  $M$  neuronas se activará al presentar el patrón de entrada  $X_p$ :

$$y_{pj} = \begin{cases} 1 & \min \|X_p - W_j\| \\ 0 & \text{resto} \end{cases}$$

Donde  $y_{pj}$  representa la salida o el grado de activación de las neuronas de salida en función del resultado de la competición (1 = neurona vencedora, 0 = neurona no vencedora),  $\|X_p - W_j\|$  representa una medida de similitud entre el vector o patrón de entrada  $X_p$ :  $x_{p1}, \dots, x_{pi}, \dots, x_{pN}$  y el vector de pesos  $W_j$ :  $w_{j1}, \dots, w_{ji}, \dots, w_{jN}$ , de las conexiones entre cada una de las neuronas de entrada y la neurona de salida  $j$ . La neurona vencedora es la que presenta la diferencia mínima.

---

<sup>17</sup> Hilera y Martínez, 1995

#### 2.4.2 Etapa de aprendizaje<sup>18</sup>

Antes que nada, se debe tener en cuenta que no existe un algoritmo de aprendizaje estándar para la red SOM. A continuación presentaremos el algoritmo más usual utilizado en estas redes.

Mediante la presentación de un conjunto de patrones de entrenamiento, el algoritmo de aprendizaje trata de constituir, las diferentes clases, una por neurona de salida, que se usaran en la etapa de funcionamiento para realizar categorizaciones de nuevos patrones de entrada.

La intención del algoritmo de aprendizaje consiste **en organizar los pesos en el espacio de salida, en regiones (locales)** que actuarán como clasificadores de los datos de entrada, formándose una especie de mapa topográfico que es organizado de manera autónoma mediante un proceso cíclico de comparación de patrones de entrada (entrenamiento) y vectores de pesos.

El efecto de la etapa de aprendizaje no es otro que acercar en una pequeña cantidad el vector de pesos  $w$  de la neurona de mayor actividad (ganadora) al vector de entrada  $x$ , es decir hacer los valores de los vectores lo mas similar posible.

De forma simplificada, el proceso de aprendizaje se desarrolla de la siguiente manera. Una vez presentado y procesado un vector de entrada, se establece a partir de una medida de similitud, la neurona vencedora, una vez encontrada esta, los pesos se actualizan no solo para la neurona vencedora si no para todas las de su entorno. De este modo, ante el mismo patrón de entrada, dicha neurona responderá en el futuro todavía con más intensidad. El proceso se repite para un conjunto de patrones de entrada los cuales son

---

<sup>18</sup> Comparar también con la pagina:  
<http://scitec.uwichill.edu.bb/cmp/online/p21h/Tutorial3/tut3.htm> de Janak Sodha , 1997  
ultima modificación: enero 22, 2003. Ejemplo con aplets.

presentados repetidamente a la red, de forma que al final los diferentes vectores de pesos sintonizan con uno o varios patrones de entrada y, por tanto, con dominios específicos del espacio de entrada. Si dicho espacio está dividido en grupos, cada neurona se especializará en uno de ellos, y la operación esencial de la red se podrá interpretar como un análisis de clases clusters.

A continuación presentaremos los pasos que se llevan a la hora de entrenar a la red:

### *1. Iniciación de los pesos y el número de iteraciones*

Para comenzar la etapa de entrenamiento, por primera vez, se debe realizar una inicialización de los pesos sinápticos  $W_{ijk}$ , en general, no existe discusión en este punto y los pesos se inicializan con pequeños valores aleatorios, por ejemplo, entre -1 y 1 ó entre 0 y 1<sup>19</sup>, también se puede partir en  $t = 0$  de diferentes configuraciones: pesos nulos<sup>20</sup>, o aleatorios de pequeño valor absoluto<sup>21</sup>.

El número de iteraciones se establece empíricamente, Kohonen indica que al menos es necesarias 500 veces el número de neuronas de salida.

### *2. Presentación de ejemplo*

En cada iteración se realiza una presentación de un patrón  $x_t$ , tomado de acuerdo con la función de distribución  $P_x$  del espacio sensorial de entrada (si el espacio es finito basta con tomar al azar uno de ellos). Es decir presentar a las neuronas de entrada un ejemplo que se debe propagar a la capa de proceso, en otras palabras avanzar a la salida. De esta presentación de muestras va a depender el número de iteraciones que se vayan a realizar.

---

<sup>19</sup> Kohonen, 1990

<sup>20</sup> Martín del Brío y Serrano, 1993

<sup>21</sup> SPSS Inc., 1997



### 3. Determinación de vecindad

Una vez presentada la muestra esta se propaga por la capa de proceso, hasta estabilizarse, buscando las neuronas más parecidas, agrupándose como por categorías o regiones (clusters), lo que lleva a no considerar a las neuronas aisladas, si no que se encuentran relacionadas con las del alrededor mediante el concepto de vecindario.

La vecindad va a depender de la topología de la red y el tiempo. Existen diferentes funciones de vecindad (las que determinan como se agrupan las redes) que se expondrán mas adelante. La más utilizada es la del espacio Euclideo: Cada neurona  $i = (i, j)$  del mapa calcula en paralelo la similitud entre su vector de pesos sinápticos  $w_{ij}$  y el actual vector de entrada  $x$ . Un criterio de similitud pudiera ser la distancia euclídea.

$$d^2(w_{ij}, x) = \sum_{k=1}^n (w_{ijk} - x_k)^2$$

Esta función de vecindad debe cubrir el espacio de salidas suficiente para que neuronas de pesos similares tengan acceso a acercarse entre sí. Y el coeficiente de aprendizaje debe ser alto (mayor que 0.1) para permitir a la red organizarse.

### 4. Determinación de la neurona ganadora

Luego se determina una neurona ganadora  $g = (g_1, g_2)$ , cuya distancia sea la menor de todas. En este punto el radio de vecindad se va disminuyendo, creándose una competencia entre las neuronas por aprender y de esta forma encontrar el camino que más se acerque a la entrada, los pesos se van ajustando quedando algunas neuronas sin activarse.

Sin embargo, se debe advertir que el criterio de similitud y la regla de aprendizaje que se utilicen en el algoritmo deben ser métricamente compatibles. Si esto no es así, estaríamos utilizando diferentes métricas para la identificación de la neurona vencedora y para la modificación del vector de pesos asociado, lo que podría causar problemas en el desarrollo del mapa<sup>22</sup>.

#### 5. Actualización de los pesos sinápticos

Al cabo de terminar la neurona ganadora se realiza una actualización de los pesos sinápticos de la neurona ganadora  $g = (g_1, g_2)$ , y los de sus neuronas vecinas. La regla mas empleada es:

$$w_{ijk}(t+1) = w_{ijk}(t) + \alpha(t)h(|i-g|)(x_k(t) - w_{ijk}(t))$$

Donde  $\alpha(t)$  es un parámetro o factor denominado tasa de aprendizaje, existen varias posibilidades para esta función, desde una constante hasta algún tipo de función monótona decreciente con el tiempo. La función  $h$  se denomina de vecindad, puesto que establece que neuronas son las vecinas de la ganadora. Esta función depende de la distancia entra a neurona  $i$  y la ganadora  $g$ , valiendo cero cuando  $i$  no pertenece a la vecindad de  $g$  y un numero positivo cuando si pertenece. Además esta función decrece con el número de iteraciones del proceso de aprendizaje, reduciendo paulatinamente en cada iteración la zona de vecindad alrededor de la neurona ganadora.

#### 6. Fin del aprendizaje

Si se ha alcanzado el número máximo de iteraciones establecido, entonces el proceso de aprendizaje finaliza. En caso contrario se vuelve al paso 2. Se puede realizar a continuación una segunda fase en el aprendizaje, en la que se produce el ajuste fino del mapa, de modo que la distribución de los pesos sinápticos se ajuste más a las de las entradas. El proceso es similar al

---

<sup>22</sup> Demartines y Blayo, 1992

anterior, tomando  $\alpha(t)$  constante y un valor pequeño (por ejemplo, 0.01) y radio de vecindad constante e igual a uno.

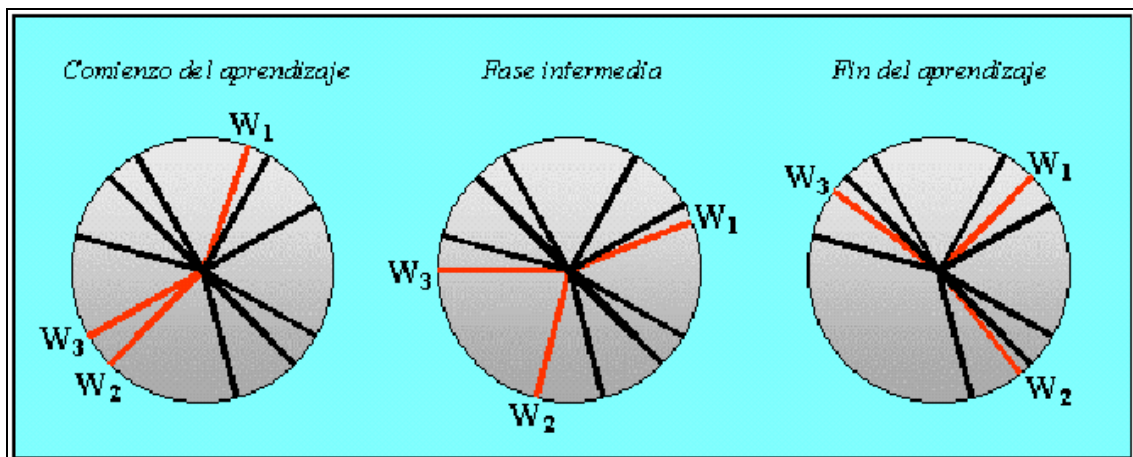
Una ventaja que presenta este algoritmo, es que los resultados que se obtienen aun en presencia de ruido son relativamente buenos porque el número de clases está fijo y los pesos se adaptan lentamente. Por lo tanto, este algoritmo es una alternativa cuando se requiere un cuantificador de vectores y se puede especificar el número de grupos.

#### 2.4.3 Interpretación del algoritmo de aprendizaje

Una forma de interpretar el algoritmo de aprendizaje es la geométrica, como hemos dicho anteriormente lo que busca el algoritmo, es acercarse de forma iterativa el vector de pesos de mayor actividad al vector de pesos de la entrada. Así, en cada iteración el vector de pesos de la neurona vencedora rota hacia el de entrada, y se aproxima a él en una cantidad que depende del tamaño de una tasa de aprendizaje.

A continuación se muestra cómo opera la regla de aprendizaje para el caso de varios patrones pertenecientes a un espacio de entrada de dos dimensiones, representados en la figura por los vectores de color negro.

FIGURA 9 Proceso de aprendizaje en dos dimensiones.



Supongamos que el número de neuronas de la red es tres y que los vectores del espacio de entrada se agrupan en tres clusters. Al principio del entrenamiento los vectores de pesos de las tres neuronas (representados por vectores de color rojo) son aleatorios y se distribuyen por la circunferencia. Conforme avanza el aprendizaje, éstos se van acercando progresivamente a las muestras procedentes del espacio de entrada, para quedar finalmente estabilizados como centroides de los tres clusters.

Al finalizar el aprendizaje, el vector de referencia de cada neurona de salida se corresponderá con el vector de entrada que consigue activar la neurona correspondiente.

En el caso de existir más patrones de entrenamiento que neuronas de salida, como en el ejemplo expuesto, más de un patrón deberá asociarse con la misma neurona, es decir, pertenecerán a la misma clase. En tal caso, los pesos que componen el vector de referencia se obtienen como un promedio (centroide) de dichos patrones.

Este proceso de aprendizaje no es global si no local, ya que la modificación de los pesos se realiza en una mayor o menor grado dependiendo de la proximidad a la neurona ganadora, el ritmo de este depende del parámetro  $\alpha(t)$  (tasa de aprendizaje), que la mayoría de las veces como habíamos mencionado es monótonamente decreciente con el tiempo, la cual determina la magnitud del cambio en los pesos ante la presentación de un patrón de entrada. La tasa de aprendizaje, con un valor inicial entre 0 y 1, por ejemplo, 0.6, decrece con el número de iteraciones (t), de forma que cuando se ha presentado un gran número de veces todo el juego de patrones de aprendizaje, su valor es prácticamente nulo, con lo que la modificación de los pesos es insignificante. Normalmente, la actualización de este parámetro se realiza mediante una de las siguientes funciones<sup>23</sup>:

---

<sup>23</sup> Hilera y Martínez, 1995.

$$\alpha(t) = \alpha_0 + (\alpha_f - \alpha_0) \frac{t}{t_\alpha}$$

Con  $\alpha_0$  el ritmo de aprendizaje inicial ( $< 1.0$ ),  $\alpha_f$  el final ( $= 0.01$ ) y  $t_\alpha$  el máximo número de iteraciones hasta llegar a  $\alpha_f$ .

$$\alpha(t) = \alpha_0 \left( \frac{\alpha_f}{\alpha_0} \right)^{\frac{t}{t_\alpha}}$$

El empleo de una u otra función no influye en exceso en el resultado final.

La función de vecindad<sup>24</sup>  $h(|i-g|, t)$  define en cada iteración  $t$  si una neurona  $i$  pertenece o no a la vecindad de la vencedora  $g$ . La vecindad es simétrica y centrada en  $g$ :

$$|i-g| = \sqrt{(i-g_1)^2 + (j-g_2)^2}$$

Pudiendo adoptar una forma circular, cuadrada, hexagonal o cualquier otro polígono regular. En general  $h$  decrece con la distancia de la vencedora y depende de un parámetro denominado radio de vecindad  $R(t)$ , que representa el tamaño de la vecindad actual. La función de vecindad más simple es el tipo escalón, que denominamos rectangular:

$$h(|i-g|, t) = \begin{cases} 0 & \text{si } |i-g| > R(t) \\ 1 & \text{si } |i-g| \leq R(t) \end{cases}$$

Por tanto, en este caso una neurona  $i = (i, j)$  pertenece a la vecindad de la ganadora solamente si su distancia es inferior a  $R(t)$ . La actualización de los pesos queda:

$$\Delta w_{ijk}(t) = \begin{cases} 0 & \text{si } |i-g| > R(t) \\ \alpha(t)(x_k(t) - w_{ijk}(t)) & \text{si } |i-g| \leq R(t) \end{cases}$$

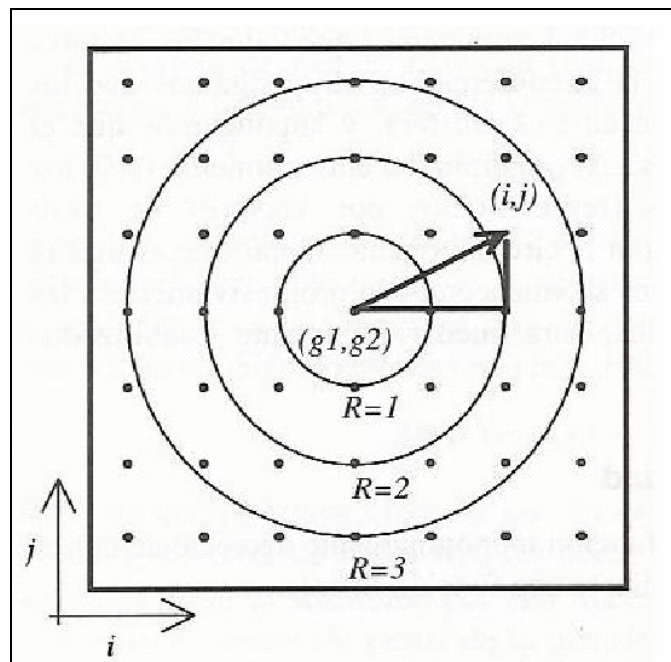
<sup>24</sup> Comparar también con la página:

<http://diwww.epfl.ch/mantra/tutorial/english/rbf/html/index.html> de Jesse W. Hong del Instituto de Tecnología de Massachussets. Applet de la función gaussiana

La función de vecindad posee una forma definida, pero su radio varia con el tiempo. Se parte de un valor inicial  $R_0$  grande, que determina vecindades amplias, con el fin de lograr la ordenación global del mapa.  $R(t)$  disminuye monótonamente con el tiempo, hasta alcanzar un valor final  $R_f = 1$ . Una posible función sería:

$$R(t) = R_f + (R_0 - R_f) \frac{t}{t_f}$$

FIGURA 10 Radios de Vecindad de la función escalón.



Con la función escalón, las vecindades adquieren una forma (cuadrada, circular, hexagonal, etc.) de bordes nítidos, en torno a la vencedora. También se utilizan a veces funciones gaussianas o en forma de sombrero mexicano continuas y derivables en todos sus puntos, que al delimitar vecindades decrecientes en el dominio espacial establecen niveles de pertenencia en lugar de fronteras nítidas.

FIGURA 11 Formas de la función de vecindad: Función gaussiana (I).

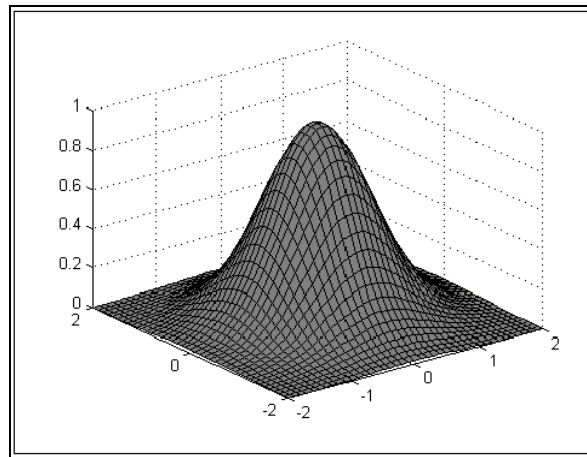
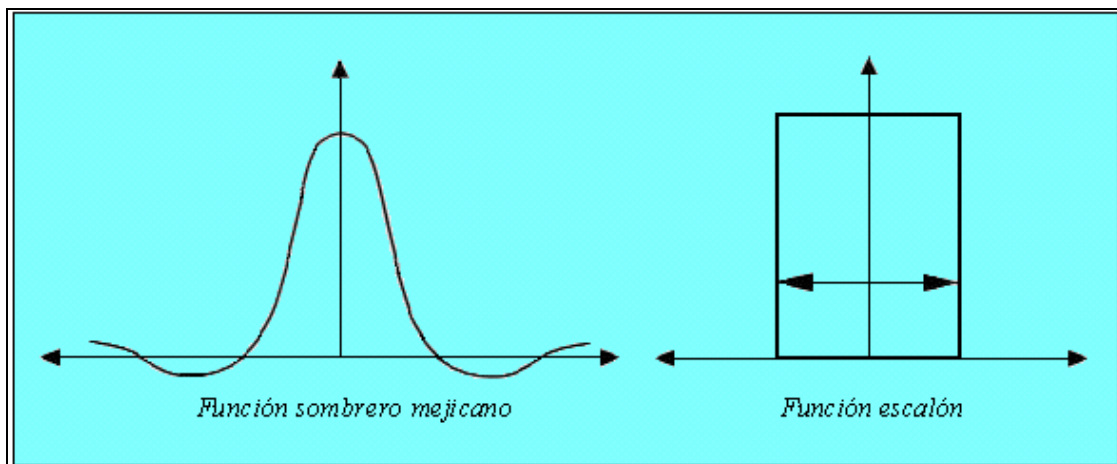


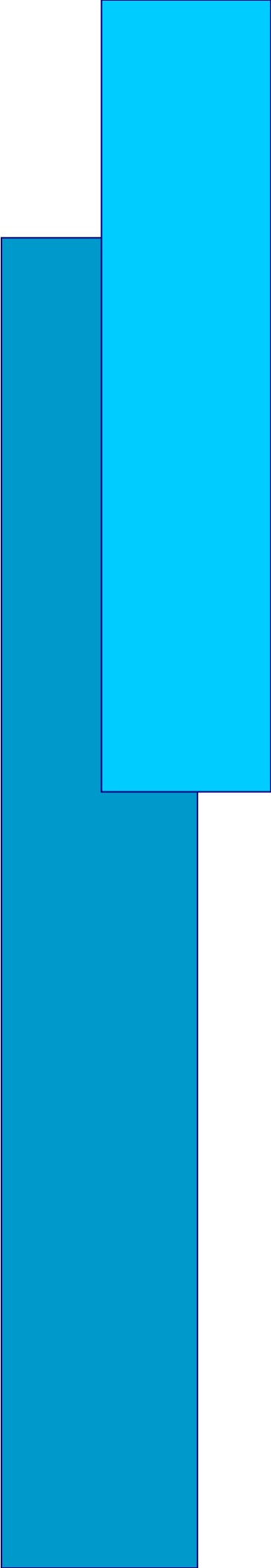
FIGURA 12 Formas de la función de vecindad (II).



Cuando trabajamos en sistemas no supervisados no disponemos de los ejemplos clasificados y por lo tanto no podemos utilizar los criterios de errores habituales, falsos positivos y falsos negativos. Por esto utilizamos una aproximación distinta el error de cuantización. Este error se basa en calcular la distancia euclídea de la neurona ganadora con el ejemplo a clasificar, así cuanto menor sea esta distancia mejor será el mapa. Este error se computa de la siguiente forma  $\|x - m_c\|$ . Se puede utilizar de igual forma una aproximación usando la función de vecindad  $h$  de la siguiente manera  $\sum h_{ci} \|x - m_c\|^2$  siendo  $h_{ci}$  la función de vecindad.

Con el objeto de obtener un mapa lo más adecuado posible, el entrenamiento se debería comenzar en múltiples ocasiones, cada vez utilizando una configuración de parámetros de aprendizaje diferentes y así, el mapa que obtenga el error cuantificador promedio (distancia euclídea) más bajo será el seleccionado para pasar a la fase de funcionamiento normal de la red.





## **Capítulo 3**

### Ejemplos de ejecución de los SOM

*Análisis de los Mapas Autoorganizados a partir de las herramientas del programa MatLab para redes SOM, que abarcan su comportamiento y propiedades, y el uso básico del toolbox de MatLab, en cuanto a las formas de visualización de los mapas y el análisis de los datos obtenidos con la implementación del SOM.*

En este capítulo se explican las cuatro demostraciones que hacen parte del Toolbox<sup>25</sup> de MatLab sobre Mapas Auto Organizados, o como los llamaremos en adelante, SOM. Estos demos se organizan de la siguiente manera: inician con una breve explicación de lo que se quiere lograr, seguido de una lista que reúne las principales funciones ejecutadas dentro del mismo, si aplica al caso. Finalmente, se explican en forma detallada los pasos seguidos para el funcionamiento particular del ejemplo SOM en cuestión.<sup>26</sup>

### 3.1 Demostración SOM\_DEMO1 Comportamiento y propiedades de SOM

#### 3.1.1 Descripción:

Con esta primera demostración, MatLab presenta las funciones a utilizar en el diseño, inicialización y entrenamiento de una red SOM, haciendo énfasis en las clases de inicialización de los vectores prototipo de las neuronas de salida según sea aleatorio o lineal, y en los algoritmo de entrenamiento posibles. Para esto, plantea dos casos: el primero se basa en datos de entrada bidimensionales, al igual que un mapa de salida bidimensional. Para el segundo caso, las dimensiones del mapa de salida se conservan, pero el espacio de entrada es ahora de tres dimensiones. Finalmente, establece comparaciones entre los resultados obtenidos para cada caso, bajo el título de calidad del mapa.

#### 3.1.2 Funciones<sup>27</sup>

`som_make`

Creación, inicialización y entrenamiento de SOM.

---

<sup>25</sup> Este Toolbox ha sido anexado al medio magnético que complementa este trabajo. Se trabaja en forma similar a cualquier otra herramienta demostrativa de MatLab 5.3.

<sup>26</sup> Para una información más completa y mejor visualización de la secuencia, se sugiere la ejecución de los listados de comandos con el programa Matlab.

|                             |  |
|-----------------------------|--|
| <code>som_randinit</code>   | Creación, inicialización de SOM  |
| <code>som_lininit</code>    | Creación, inicialización de SOM  |
| <code>som_seqtrain</code>   | Entrenamiento de SOM.  |
| <code>som_batchtrain</code> | Entrenamiento de SOM.  |
| <code>som_bmus</code>       | Encontrar las Unidades de Mejor Coincidencia (Best-matching units (BMUs)). |
| <code>som_quality</code>    | Medición de la calidad de SOM.   |

### 3.1.3 Desarrollo

- Mapas Auto Organizativos SOM

Un mapa auto organizado se puede definir como un "mapa" donde se organizan datos de entrenamiento de forma tal que se hace denso en las zonas donde coincida la mayor cantidad de datos y delgado donde la densidad de los mismos es menor. Las neuronas se organizan formando mapas bidimensionales de figuras geométricas como hexágonos o rectángulos. Los siguientes comandos son empleados en la formación de dichas configuraciones. Las configuraciones obtenidas se observan en la figura 13.

Para obtener una figura hexagonal

```
subplot(1,2,1)
som_cplane('hexa',[10 15],'none')
title('Hexagonal SOM grid')
```

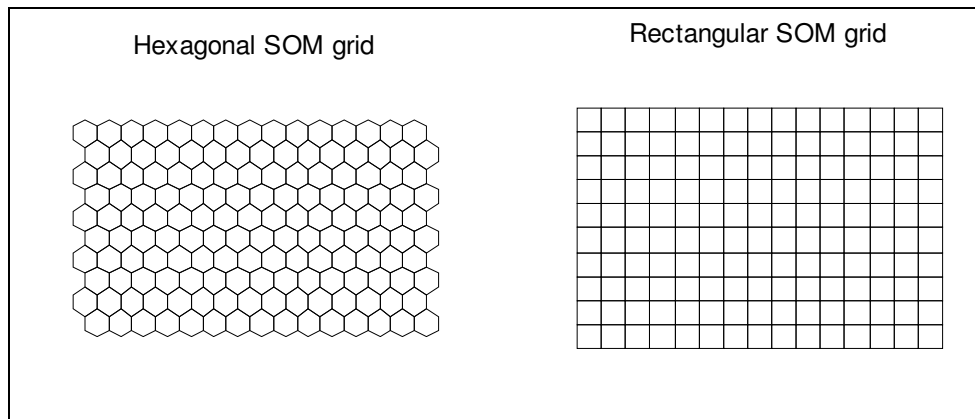
Y una figura rectangular con:

```
subplot(1,2,2)
som_cplane('rect',[10 15],'none')
title('Rectangular SOM grid')
```

---

<sup>27</sup> Se pueden encontrar definiciones más amplias de estas funciones consultando el HelpDesk de MatLab.

FIGURA 13 Configuraciones de neuronas: hexagonal y rectangular.



Cada neurona tiene un vector prototipo asociado. Después del entrenamiento, las neuronas vecinas tendrán vectores prototipos similares. Los SOM pueden ser utilizados para la visualización de datos, clasificación, estimaciones y una gran variedad de otros propósitos.

Tenemos a continuación 300 muestras de datos, tomadas aleatoriamente con el siguiente comando:

```
D = rand(300,2);
```

El mapa será una cuadrilla bidimensional con un tamaño de 10 x 10.

```
msize = [10 10];
```

Los comandos SOM\_RANDOMINIT y SOM\_LININIT pueden ser empleados para inicializar los vectores prototipo en el mapa. El tamaño del mapa es en realidad un argumento opcional. Si se omite, será determinado automáticamente con base en la cantidad de vectores de datos y de eigenvectores presentes en los mismos. A continuación se emplea un algoritmo de iniciación aleatoria:

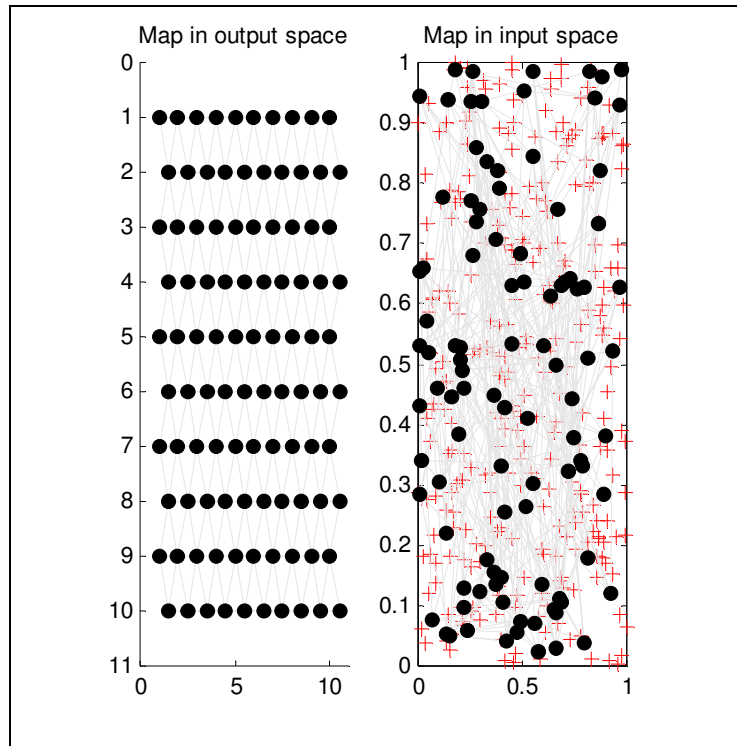
```
sMap = som_randinit(D, 'msize', msize);
```

En realidad, cada unidad del mapa puede ser analizada como el resultado de dos coordenadas: (1) en el espacio de entradas: los vectores prototipo, (2) en

el espacio de salidas: la posición en el mapa. En dichos espacios, las gráficas correspondientes se logran empleando los siguientes comandos. Los mapas obtenidos se muestran en la figura 14.

```
subplot(1,3,1)
som_grid(sMap)
axis([0 11 0 11]), view(0,-90), title('Map in output space')
subplot(1,3,2)
plot(D(:,1),D(:,2),'+r'), hold on
som_grid(sMap,'Coord',sMap.codebook)
title('Map in input space')
```

FIGURA 14 Mapas correspondientes al espacio de entradas y al espacio de salidas.

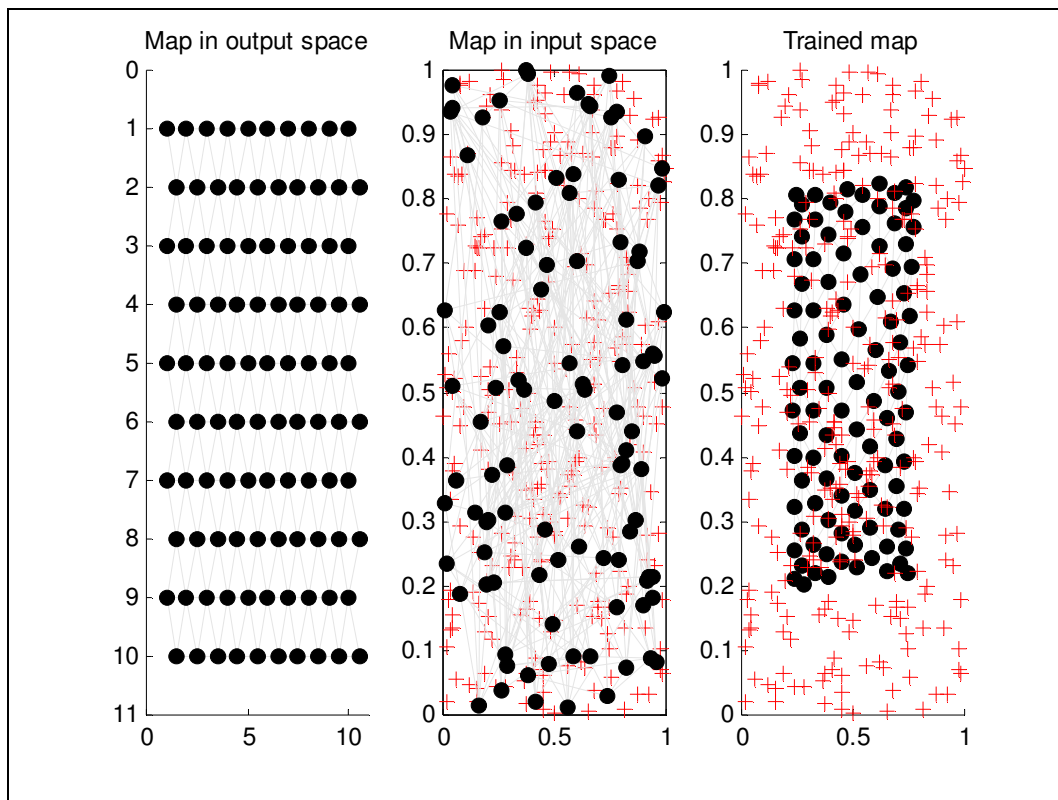


Los puntos negros que se observan en la figura muestran las posiciones de las unidades del mapa, y las líneas grises muestran las conexiones entre las unidades vecinas. Como el mapa fue inicializado aleatoriamente, las

posiciones en el espacio de salida están completamente desorganizadas. Las cruces rojas corresponden a datos de entrenamiento. El entrenamiento se realiza con los comandos que se escriben a continuación, y la gráfica del resultado de dicho entrenamiento se puede observar en la figura 15.

```
sMap = som_seqtrain(sMap,D,'radius',[5 1],'trainlen',10);
subplot(1,3,3)
som_grid(sMap,'Coord',sMap.codebook)
hold on, plot(D(:,1),D(:,2),'+r')
title('Trained map')
```

FIGURA 15 Visualización del entrenamiento del mapa.



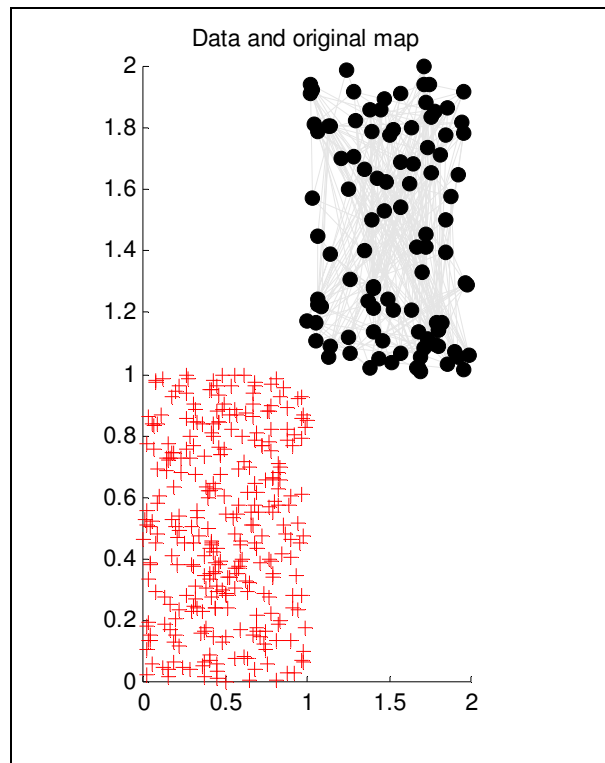
- Entrenamiento Del SOM

Para obtener una mejor idea de lo que ocurre durante el entrenamiento, analizaremos como el mapa gradualmente se despliega y se autoorganiza. A continuación, inicializaremos nuevamente el mapa de modo que se aleje lo

más posible de los datos anteriores. Los nuevos espacios con los que se va a trabajar se muestran en la figura 16.

```
sMap = som_randinit(D,'msize',msize);  
sMap.codebook = sMap.codebook + 1;  
subplot(1,2,1)  
som_grid(sMap,'Coord',sMap.codebook)  
hold on, plot(D(:,1),D(:,2),'+r'), hold off  
title("Data and original map")
```

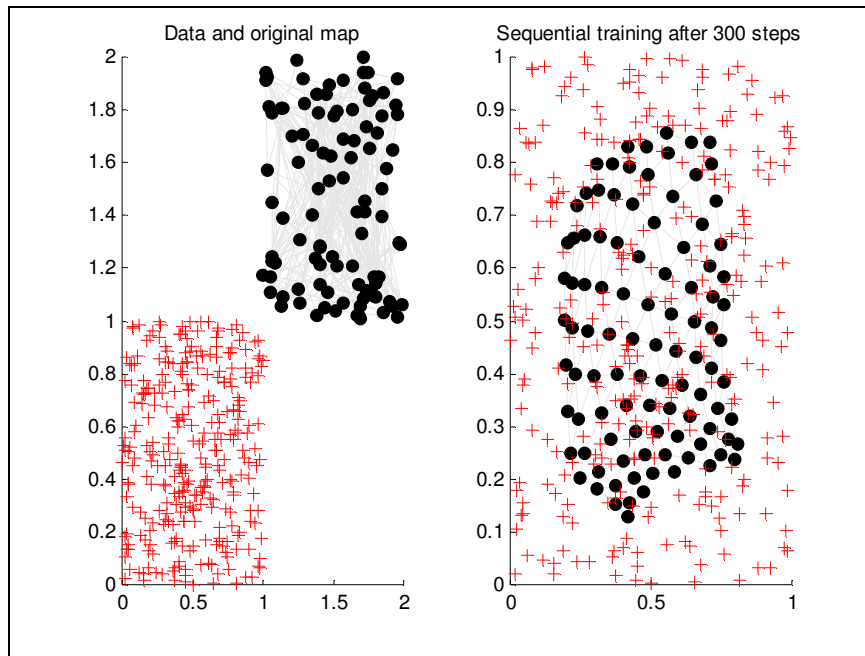
FIGURA 16 Datos a trabajar, y nuevo mapa.



El entrenamiento está basado en dos principios: el del aprendizaje competitivo, donde el vector prototipo que más se parece a un vector de datos es el que se modifica para mantener aún más este parecido, de forma que el mapa aprenda la posición respectiva en la nube de datos. El otro principio es el del aprendizaje cooperativo, en el cual la vecindad de dicho vector también es modificada, de modo que se logre la organización del

mapa. La gráfica 17 muestra el mapa resultante después del entrenamiento, y que corresponde a dicho espacio de entradas:

FIGURA 17 Visualización del entrenamiento luego de 300 iteraciones.



- Datos De Entrenamiento: Unidad Cúbica

En los casos anteriores, la dimensión del mapa era igual a la dimensión del espacio de entrada, es decir, ambos correspondían a configuraciones bidimensionales. Típicamente, las dimensiones del espacio de entrada son mucho más grandes que las del mapa bidimensional, caso en el cual el mapa no puede seguir con fidelidad dichos datos, sino que deberá encontrar un equilibrio entre la precisión de la representación de los mismos, y la de la topología a alcanzar.

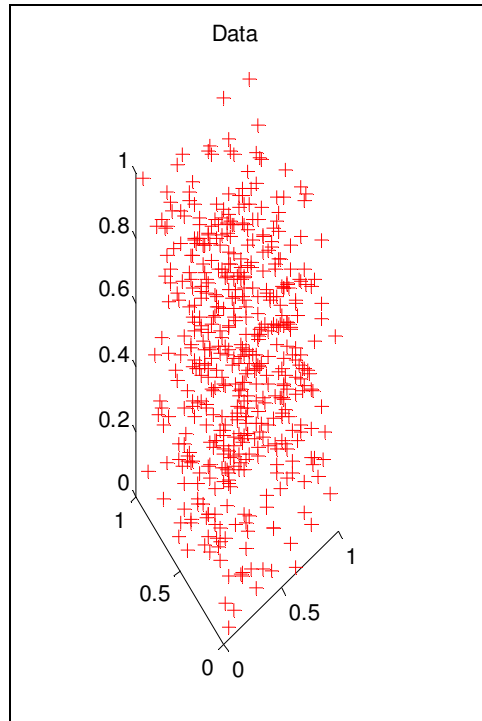
A continuación tenemos 500 muestras de datos tomadas aleatoriamente con el siguiente comando, y que corresponde a un espacio cúbico, mostrado en la figura 18:

```
D = rand(500,3);
```



```
subplot(1,3,1), plot3(D(:,1),D(:,2),D(:,3),'+r')  
view(3), axis on, rotate3d on  
title('Data')
```

FIGURA 18 Datos del espacio de entradas cúbico.



- Proceso de entrenamiento por default

En el caso de arriba, la inicialización fue hecha de forma aleatoria y el entrenamiento se realizó con la función secuencial SOM\_SEQTRAIN. Por default, la inicialización es lineal, y empleamos el algoritmo de entrenamiento de corridas. Además, el entrenamiento se hace en dos fases: primero, con amplios radios de vecindad, y luego se van llevando estos radios a valores cada vez más pequeños. La función SOM\_MAKE puede emplearse tanto para inicializar como para entrenar el mapa usando parámetros ya establecidos:

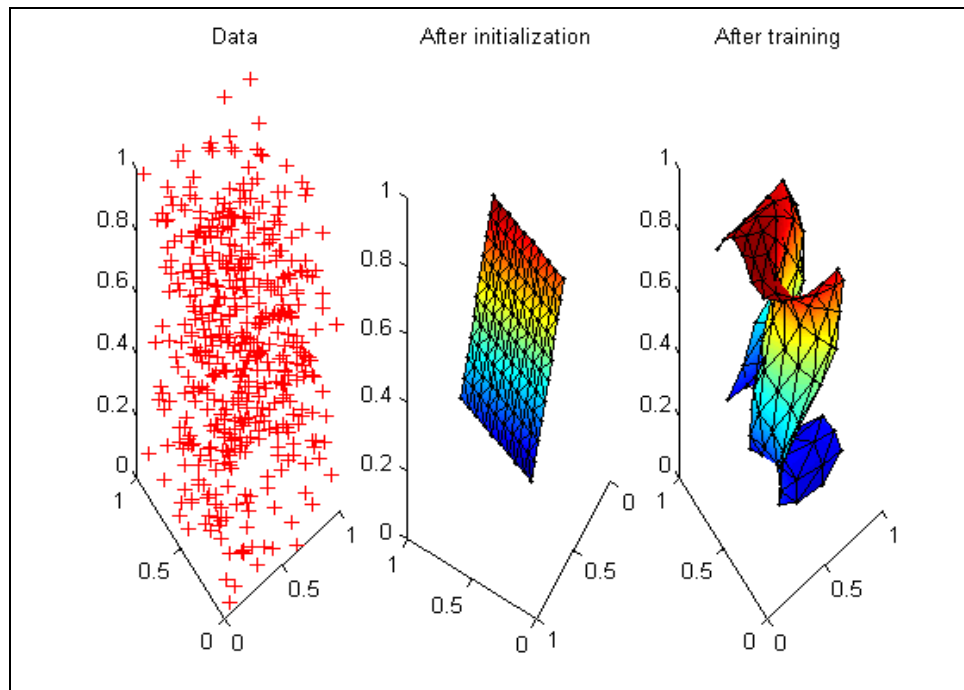
```
map size [11, 10]
```

A continuación, se realiza nuevamente la inicialización lineal de modo que se puedan comparar los resultados obtenidos

```
sMap0 = som_lininit(D);  
subplot(1,3,2)  
som_grid(sMap0,'Coord',sMap0.codebook,...  
axis([0 1 0 1 0 1]), view(-120,-25), title('After initialization')  
subplot(1,3,3)  
som_grid(sMap,'Coord',sMap.codebook,...  
axis([0 1 0 1 0 1]), view(3), title('After training'), hold on
```

En la figura 19 a continuación se puede observar como el mapa bidimensional se ha desplegado a un espacio tridimensional para lograr asemejar o capturar los datos por completo, que correspondían a un espacio en tres dimensiones.

FIGURA 19 Visualización de resultados después de la inicialización y el entrenamiento.



- Unidades de Mejor Coincidencia (Best-Matching Units (BMU))

Antes de analizar la calidad de los resultados, es importante introducir el concepto de Unidades de Mejor Coincidencia, o BMU como las llamaremos en adelante. La BMU de un vector de datos corresponde a la unidad en el mapa cuyo vector modelo mejor se asemeja al vector de datos. En la práctica, la similitud es medida como la distancia mínima entre el vector de datos y cada vector modelo en el mapa. La BMU puede ser calculada a través de la función SOM\_BMUS. Esta función entrega el índice de la unidad.

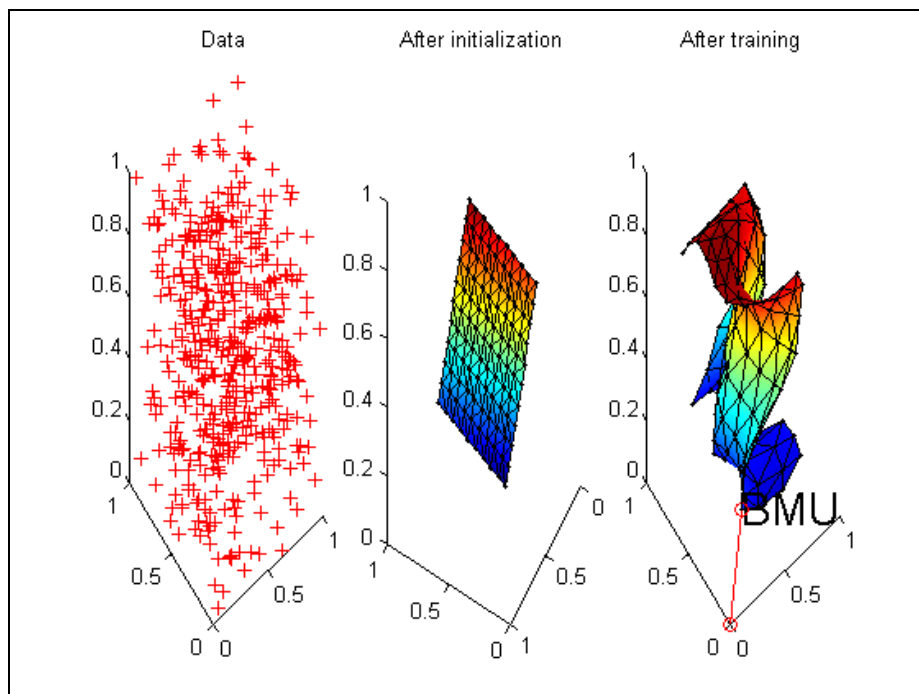
A continuación, la BMU es buscada desde el punto de origen (desde el mapa entrenado) con el siguiente comando:

```
bmu = som_bmus(sMap,[0 0 0]);
```

La unidad correspondiente es mostrada en la figura 20, y los comandos empleados son:

```
co = sMap.codebook(bmu,:);
text(co(1),co(2),co(3),'BMU','FontSize',20)
plot3([0 co(1)],[0 co(2)],[0 co(3)],'ro-')
```

FIGURA 20 Cálculo de la BMU.



Dentro de la medición de la calidad se implementa la función SOM\_QUALITY, a partir del porcentaje de vectores de datos para los que la primera y segunda BMU no son unidades adyacentes. A continuación se muestran las medidas de calidad para el mapa entrenado:

$[q,t] = \text{som\_quality}(\text{sMap},D)$

$q = 0.1340$

$t = 0.0360$

Y las correspondientes al mapa inicial son:

$[q_0,t_0] = \text{som\_quality}(\text{sMap}_0,D)$

$q_0 = 0.2686$

$t_0 = 0$

Como se puede observar, el despliegue del SOM ha logrado reducir el error promedio, pero se ha afectado enormemente la capacidad de representar la topología requerida. Mediante el uso de radios de vecindad mayores al final del entrenamiento, el mapa se vuelve más rígido, y preserva de mejor forma la topología de los datos.

## **3.2 Demostración SOM\_DEMO2** Uso Básico del ToolBox SOM

### 3.2.1 Descripción:

Con esta demostración, MatLab plantea los cinco pasos básicos para la creación de una red SOM: la construcción de los datos de entrada, y su normalización; el entrenamientos de la red, la visualización de resultados y el análisis de los mapas obtenidos. Utiliza además el famoso caso de estudio de las flores del Iris<sup>28</sup> para la explicación de cada uno de estos pasos.

---

<sup>28</sup> Este es uno de los problemas más conocidos en el ámbito del aprendizaje automático y del reconocimiento de patrones. El conjunto se compone de 150 ejemplos de lirios que pertenecen a una de las variedades siguientes: setosa, versicolor o virgínica. La primera publicación en la que se hace referencia a este problema es [Fisher, 36].

### 3.2.2 Funciones<sup>29</sup>

|                              |   |
|------------------------------|---|
| <code>som_data_struct</code> | Creación de una estructura de datos.                              |
| <code>som_read_data</code>   | Lectura de datos desde un archivo.                                |
| <code>som_normalize</code>   | Normalización de datos.   |
| <code>som_denormalize</code> | Desnormalización de datos.  |
| <code>som_make</code>        | Inicialización, entrenamiento de mapas.                           |
| <code>som_show</code>        | Visualización de mapas.   |
| <code>som_show_add</code>    | Adición de marcadores en la visualización <code>som_show</code> . |
| <code>som_grid</code>        | Visualización con coordenadas libre.                              |
| <code>som_autolabel</code>   | Establece etiquetas o categorías dentro del mapa.                 |
| <code>som_hits</code>        | Cálculo del histograma de eventos para el mapa.                   |

### 3.2.3 Desarrollo

- Uso Básico del ToolBox SOM

El uso básico del ToolBox SOM, en MatLab, es el siguiente:

1. Construcción del conjunto de datos.
2. Normalización de dicho conjunto.
3. Entrenamiento del mapa.
4. Visualización del mapa.
5. Análisis de resultados.

Los cuatro primeros ítems son (en caso de emplear opciones preestablecidas) operaciones bastante simples, cada una ejecutable con un solo comando. Para el último ítem existen diversas clases de funciones dentro del ToolBox, ya que los diferentes tipos de análisis posible no permiten preestablecer parámetros.

- Primer paso: Construcción del conjunto de datos.

---

<sup>29</sup> Se pueden encontrar definiciones más amplias de estas funciones consultando el HelpDesk de MatLab.

El ToolBox SOM posee una estructura especial, denominada estructura de datos, la cual se emplea para agrupar información reuniendo el conjunto de datos en un solo lugar. A continuación se genera una estructura de datos empleando la función SOM\_DATA\_STRUCT. El primer argumento es la misma matriz de datos, seguido del nombre dado al conjunto de datos, y luego los nombres de los componentes (variables) en la matriz de datos.

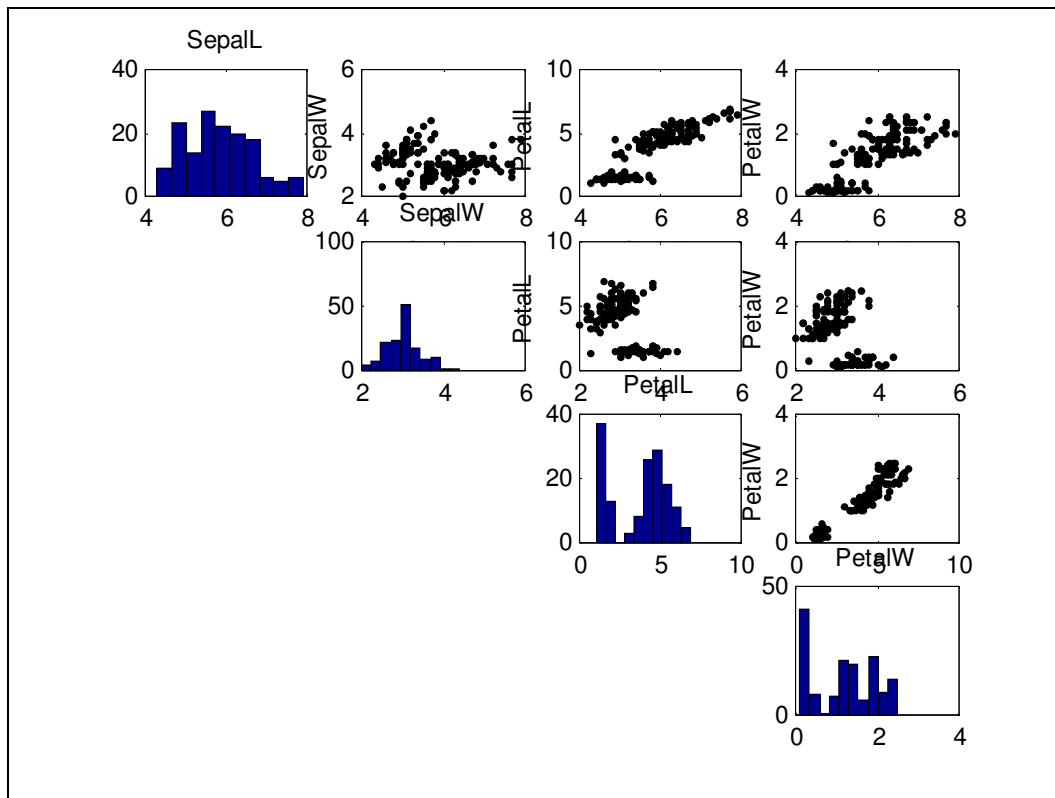
```
D = rand(1000,3); 1000 samples from unit cube
sData = som_data_struct(D,'name','unit cube','comp_names',{'x','y','z'});
```

Otra opción es leer los datos directamente de un archivo ASCII. A continuación, el conjunto de datos IRIS es descargado desde un archivo (asegúrese bien de que el archivo se encuentre disponible):

```
sDiris = som_read_data('iris.data');
```

En la figura 21 se muestran los histogramas y las gráficas de las cuatro variables correspondientes

FIGURA 21 Histogramas y gráficas de las variables IRIS.



- Segundo paso: Normalización de los datos

Como el algoritmo SOM esta basado en distancias Euclídeas, la escala de variables es muy importante a la hora de determinar como será el mapa. Si el rango de valores de alguna variable es mucho más grande que el de otras variables, esa variable probablemente dominará por completo la organización del mapa. Por esta razón, los componentes del conjunto de datos son usualmente normalizados, y esto se realiza a través de la función SOM\_NORMALIZE, con el siguiente comando:

```
sDiris = som_normalize(sDiris,'var');
```

Esta función posee además otros métodos de normalización. Sin embargo, la interpretación de los valores puede ser más difícil después de la normalización. Para esto, la operación puede ser revertida con la función SOM\_DENORMALIZE:

```
x = sDiris.data(1,:)  
x = -0.8977  1.0286 -1.3368 -1.3086  
orig_x = som_denormalize(x,sDiris)  
orig_x = 5.1000  3.5000  1.4000  0.2000
```

- Tercer paso: Entrenamiento del mapa

La función SOM\_MAKE es utilizada para el entrenamiento de SOM. Por default, primero se determina el tamaño del mapa, luego se inicializa el mapa a través de mecanismos lineales, y finalmente se usa el algoritmo de corrida para el entrenamiento de dicho mapa. Los códigos que se emplean son

```
sMap = som_make(sDiris);  
map size [11, 6]  
Initialization...  
Training using batch algorithm...
```

El conjunto de datos de IRIS posee además etiquetas asociadas con las muestras de datos. En realidad, el conjunto de datos consiste de 50

muestras de tres especies de flores (en total serían 150 muestras), las cuales representan mediciones de ancho y alto de las hojas del sépalo y pétalo. La etiqueta asociada con cada muestra es la información sobre la especie: "Setosa", "Versicolor" o "Virginica".

Ahora, el mapa puede ser etiquetado con estas categorías. LA BMU de cada categoría se encuentra desde el mapa, y las etiquetas de cada especie son asignadas a las unidades del mapa. Para esto empleamos la función SOM\_AUTOLABEL, como en las siguientes líneas:

```
sMap = som_autolabel(sMap,sDiris,'vote');
```

- Cuarto paso: Visualización del SOM: SOM\_SHOW

La visualización básica del SOM se realiza con la función SOM\_SHOW. La figura 22 muestra dicha visualización.

```
colormap(1-gray)  
som_show(sMap,'norm','d')
```

Note que los nombres de los componentes son incluidos como los títulos de los subplots. Además, los valores de las variables han sido desnormalizados al rango y la escala originales. Los planos componentes (PetalL, PetalW, SepalL y SepalW) muestran que clases de valores poseen los vectores prototipo de las unidades del mapa. El valor es indicado con color, y la barra de colores sobre la derecha muestra el significado de cada uno de ellos.

U-Matrix<sup>30</sup> muestra las distancias entre unidades vecinas, además de la visualización de la estructura de clusters (grupos) del mapa. Note que la visualización de dicha matriz posee muchos más hexágonos que los planos

---

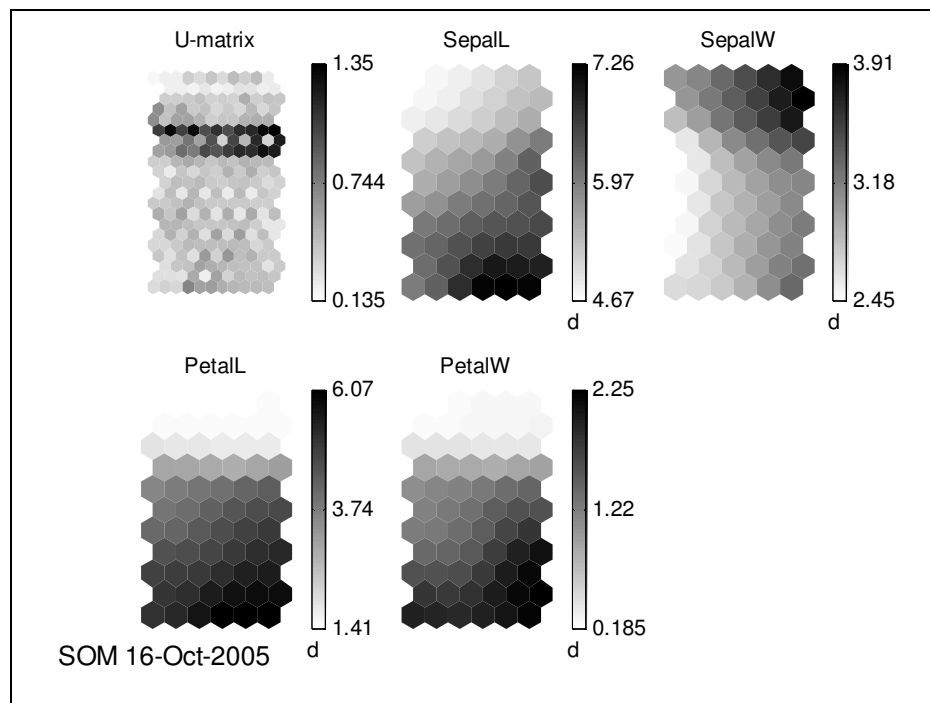
<sup>30</sup> UMatrix, herramienta que permite apreciar el agrupamiento de datos, es un algoritmo, que se usa sobre el mapa ya calibrado, y da una medida de la distancia entre dos elementos del mapa, y la distancia media de cada elemento a los que lo rodean. Esto permite apreciar los clusters: son zonas "claras" separadas por "zonas oscuras". Es una forma un tanto visual de verlo, pero es tan objetivo como asignar un umbral determinado, y decir que toda distancia por encima de ese umbral hace que se separen los clusters.



componentes. Esto se debe a que se pueden ver las distancias entre las unidades, y no sólo el valor de la distancia en las unidades del mapa.

Los altos valores que se observan en U-Matrix significan grandes distancias entre las unidades vecinas del mapa, además de indicar los límites de cada cluster. Dichos clusters son típicamente áreas uniformes de bajos valores. Si se hace referencia a la barra de colores, se puede observar cuáles colores indican altos valores. En el mapa de IRIS se podrían observar dos clusters.

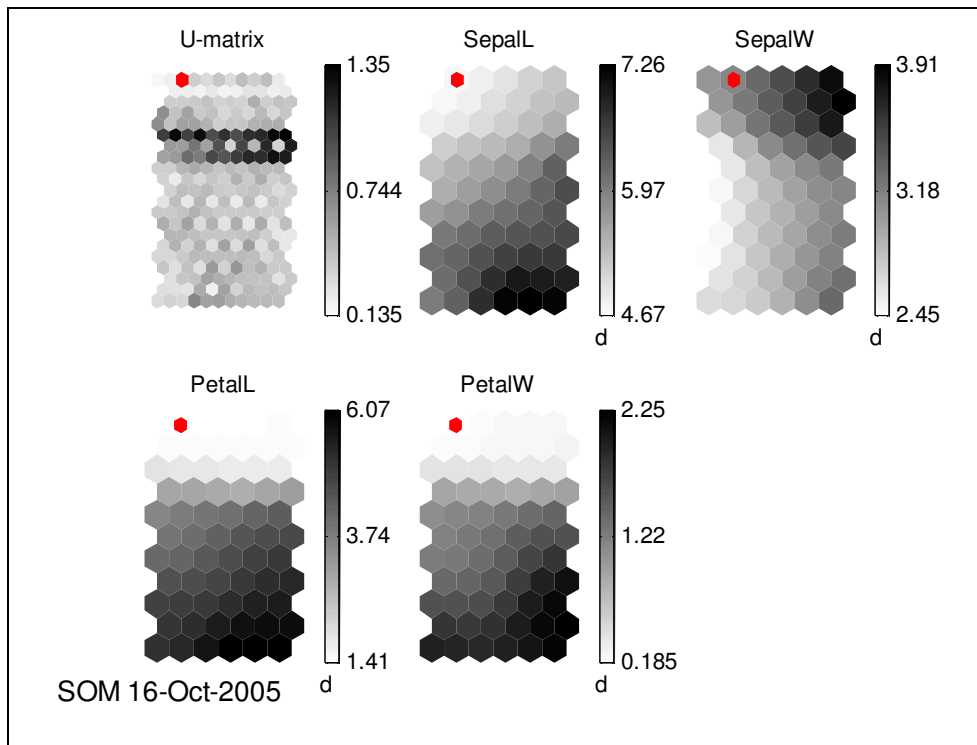
FIGURA 22 Visualización básica del SOM.



Las imágenes son enlazadas a través de posiciones similares. En cada eje, una unidad particular del mapa se encuentra siempre en el mismo lugar. Por ejemplo, con las siguientes líneas se hace que el marcador rojo de la figura 23 indique la misma unidad sobre cada eje

```
h=zeros(sMap.topol.msize); h(1,2) = 1;
som_show_add('hit',h(:),'markercolor','r','markersize',0.5,'subplot','all')
```

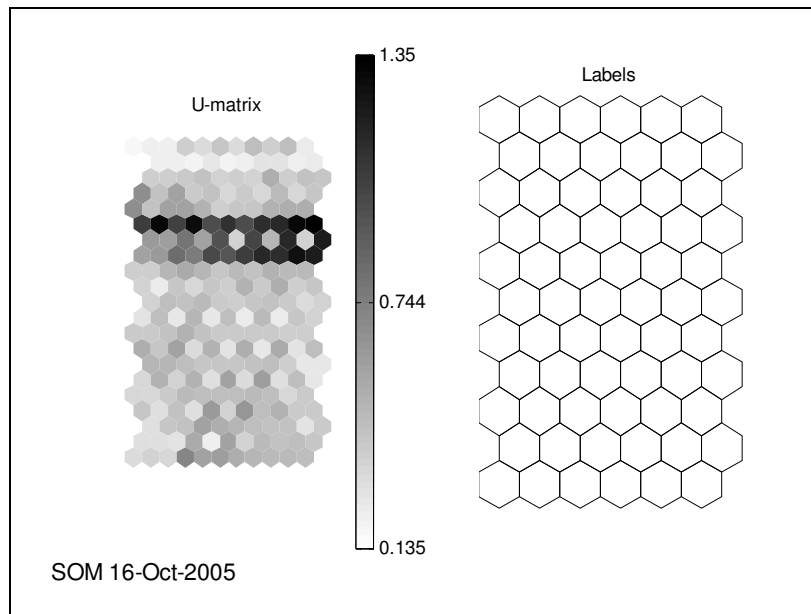
FIGURA 23 Indicación de una unidad sobre cada eje.



La función SOM\_SHOW\_ADD puede ser empleada para la adición de marcadores, etiquetas y trayectorias sobre las figuras creadas con SOM\_SHOW. La función SOM\_SHOW\_CLEAR será empleada para desaparecer estas aplicaciones. A continuación en la figura 24, U-Matrix es mostrada a la izquierda, y una cuadrícula vacía con el nombre de "Labels" (etiquetas) es mostrada a la derecha. Los comandos a utilizar son:

```
som_show(sMap,'umat','all','empty','Labels')
```

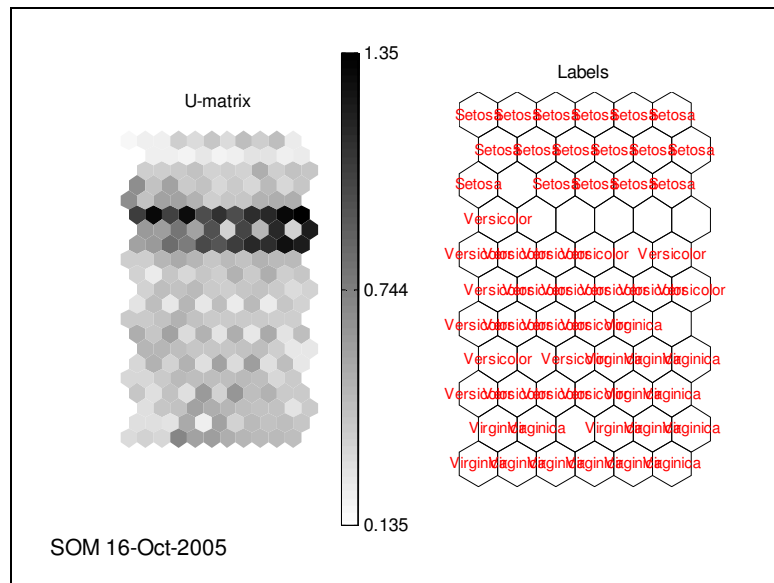
FIGURA 24 Visualización de U-Matrix



En la figura 25, sobre la cuadrícula vacía, se observan las etiquetas adicionadas al mapa con la función SOM\_AUTOLABEL, según se describe en la siguiente línea:

```
som_show_add('label',sMap,'Textsize',8,'TextColor','r','Subplot',2)
```

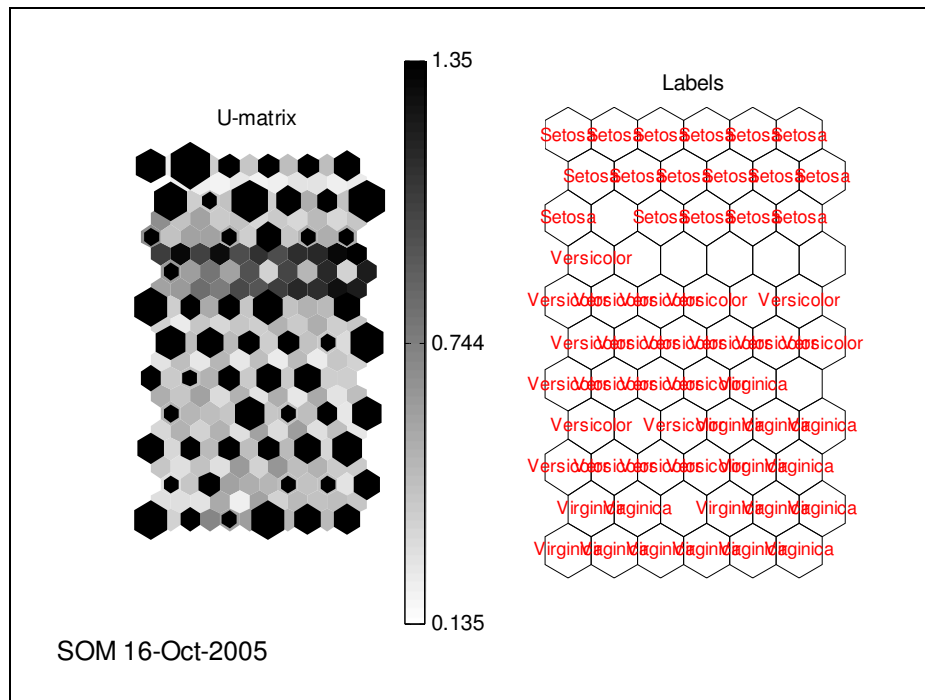
FIGURA 25 Visualización de U-Matrix y de las etiquetas adicionadas



Una herramienta importante en el análisis de datos empleando SOM es lo que se conoce como histograma de eventos. Estos se forman tomando un conjunto de datos, encontrando luego la BMU de cada muestra de datos del mapa, e incrementando un contador en una unidad del mapa cada vez que esta corresponde a la BMU. El histograma de eventos muestra la distribución del conjunto de datos sobre el mapa. A continuación, el histograma de eventos para el conjunto completo de datos es calculado y visualizado sobre U-Matrix. La gráfica corresponde a la figura 26. Los comandos a utilizar son:

```
h = som_hits(sMap,sDiris);
som_show_add('hit',h,'MarkerColor','w','Subplot',1)
```

FIGURA 26 Histograma de eventos sobre U-Matrix



Los histogramas de eventos múltiples pueden ser mostrados en forma simultánea. A continuación se calculan y se muestran en la figura 27 los tres histogramas correspondientes a las tres especies de flores Iris. El primer paso será borrar el histograma que se realizó en primera instancia, con la siguiente línea:

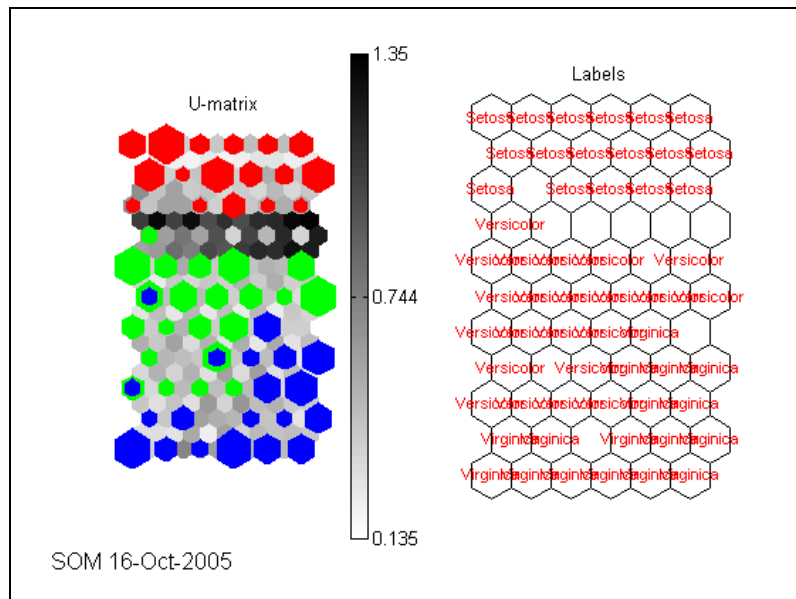
```
som_show_clear('hit',1)
```

A continuación se procede a calcular los nuevos histogramas. Las primeras 50 muestras en el conjunto de datos corresponden a la especie “Setosa”, las 50 siguientes a la especie “Versicolor” y las 50 últimas a la especie “Virginica”. Los comandos empleados son:

```
h1 = som_hits(sMap,sDiris.data(1:50,:));  
h2 = som_hits(sMap,sDiris.data(51:100,:));  
h3 = som_hits(sMap,sDiris.data(101:150,:));  
som_show_add('hit',[h1, h2, h3], 'MarkerColor',[1 0 0; 0 1 0; 0 0  
1], 'Subplot',1)
```

De la figura obtenida se puede deducir que el color rojo corresponde a “Setosa”, el verde corresponde a “Versicolor” y el azul a “Virginica”. Además, se observa que las tres especies están bastante separadas, aunque Versicolor y Virginica presenten una pequeña mezcla o unión.

FIGURA 27 Histogramas correspondientes a cada tipo de flor.



Existe además otra función empleada en la visualización: SOM\_GRID. Esta permite la visualización de SOM en coordenadas de libre especificación, como por ejemplo el espacio de entrada (máximo hasta tres dimensiones). Esta función posee gran cantidad de opciones, por lo que es considerada como muy flexible. Básicamente, con SOM\_GRID se puede visualizar la red SOM: se muestra cada unidad con un marcador, y las conexiones con la vecindad se muestran con líneas. El usuario posee el control sobre: las coordenadas de cada unidad (2D o 3D), el tipo de marcador (color y tamaño) de cada unidad y el tipo de línea (color y espesor) de las conexiones.

A continuación se presentan cuatro visualizaciones hechas con SOM\_GRID. Estas corresponden a:

La cuadrícula del mapa en el espacio de salidas

```
subplot(2,2,1)
som_grid(sMap,'Linecolor','k')
view(0,-90), title('Map grid')
```

Una vista superficial de la matriz de distancias: tanto el color como la coordenada z indican distancias promedio a las unidades vecinas en el mapa. Esto está altamente relacionado con U-Matrix.

```
subplot(2,2,2)
Co=som_unit_coords(sMap);
U=som_umat(sMap); U=U(1:2:size(U,1),1:2:size(U,2));
som_grid(sMap,'Coord',[Co, U(:)],'Surf',U(:),'Marker','none');
view(-80,45), axis tight, title('Distance matrix')
```

La cuadrícula del mapa en el espacio de salidas. Hay tres primeros componentes que determinan las coordenadas en 3D de la unidad del mapa, y el tamaño del marcador es determinado por un cuarto componente. Note que los valores han sido desnormalizados.

```
subplot(2,2,3)
M = som_denormalize(sMap.codebook,sMap);
som_grid(sMap,'Coord',M(:,1:3),'MarkerSize',M(:,4)*2)
```

```
view(-80,45), axis tight, title('Prototypes')
```

Una cuadrícula similar a la que se obtiene arriba, pero los datos originales han sido graficados también: las coordenadas muestran los valores de los tres primeros componentes y el color indica la especie de cada muestra. El cuarto componente no se muestra.

```
subplot(2,2,4)
```

```
som_grid(sMap,'Coord',M(:,1:3),'MarkerSize',M(:,4)*2)
```

```
hold on
```

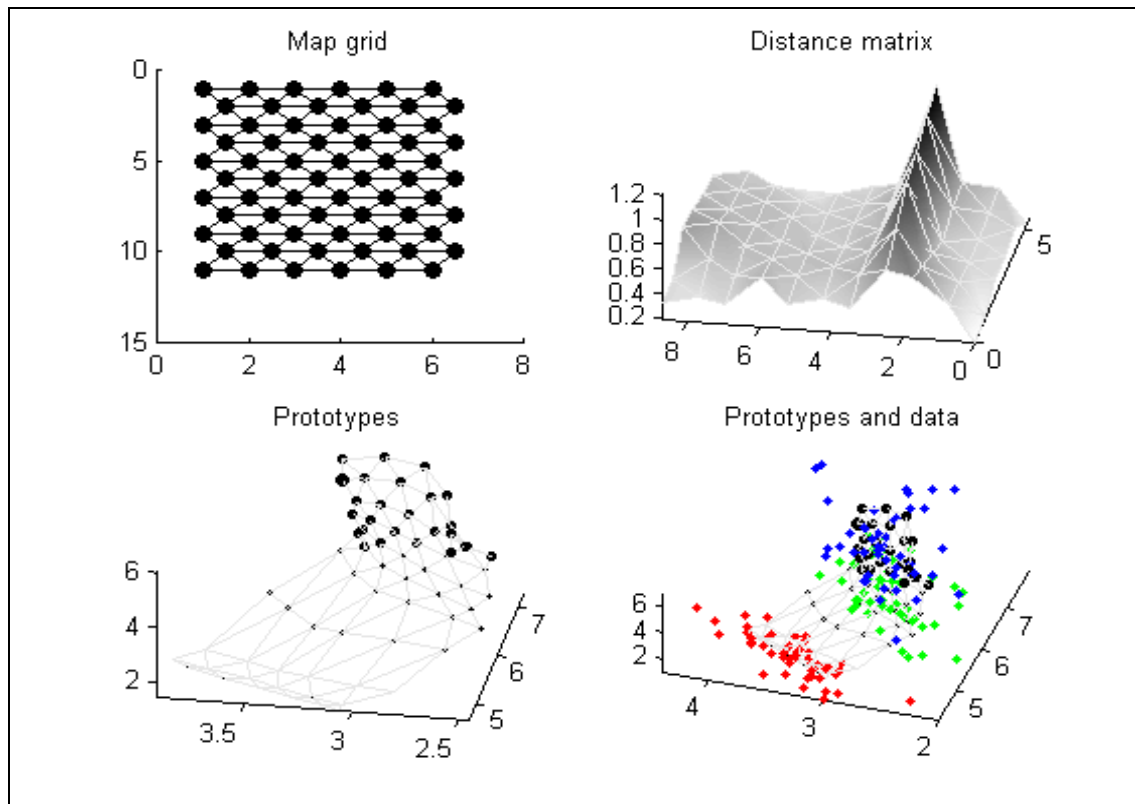
```
D = som_denormalize(sDiris.data,sDiris);
```

```
plot3(D(1:50,1),D(1:50,2),D(1:50,3),'r.',...
```

```
view(-72,64), axis tight, title('Prototypes and data')
```

Los resultados obtenidos se pueden observar en la figura 28 a continuación:

FIGURA 28 Visualizaciones obtenidas.



- Paso cinco: Análisis de resultados

El propósito de este paso depende enormemente del propósito que se quiera alcanzar en el análisis de datos: si es segmentación, modelamiento, detección de rasgos, clasificación, entre otros. Por esta razón no existe una función única de análisis, sino que existe un número de funciones individuales que pueden o no ser usadas en una aplicación o caso específico. La visualización sin duda alguna hace parte del análisis de resultados, junto con el examen de etiquetas o categorías y los histogramas de eventos, además de la validación de la calidad del SOM, que fue estudiada en la demostración anterior.

Dentro del ToolBox existe una gran cantidad de funciones que ayudan a la aplicación del análisis requerido por el usuario. Para mayor información sobre las técnicas de análisis, el lector puede remitirse a la demostración 4 del ToolBox SOM de MatLab.

### **3.3 Demostración SOM\_DEMO3 Visualización**

#### **3.3.1 Descripción:**

Es el complemento para el paso “4” planteado en la demostración 2. En esta nueva demostración se presentan las principales herramientas que ayudan a la visualización de los mapas, en cuanto al manejo de los códigos de colores, las proyecciones de datos, las identificaciones de las categorías con etiquetas y marcadores, entre otros.



### 3.3.2 Funciones<sup>31</sup>

|                             |  |
|-----------------------------|--|
| <code>som_show</code>       | Visualizar mapa.   |
| <code>som_grid</code>       | Visualización con coordenadas libres.                                      |
| <code>som_show_add</code>   | Adición de marcadores sobre la visualización <code>som_show</code> .       |
| <code>som_show_clear</code> | Borrar marcadores de la visualización <code>som_show</code> .              |
| <code>som_recolorbar</code> | Actualizar la barra de colores en la visualización <code>som_show</code> . |
| <code>som_cplane</code>     | Visualizar componentes/colores/ plano U-matrix.                            |
| <code>som_pieplane</code>   | Visualizar vectores prototipo como gráficos circulares.                    |
| <code>som_barplane</code>   | Visualizar vectores prototipo como gráficos de barras.                     |
| <code>som_plotplane</code>  | Visualizar vectores prototipo como gráficos lineales.                      |
| <code>Pcaproj</code>        | Proyección a componentes espaciales principales.                           |
| <code>Cca</code>            | Proyección con Análisis de componentes curvilíneos.                        |
| <code>Sammon</code>         | Proyección con mapeo de Sammon.  |
| <code>som_umat</code>       | Calculo de la U-matrix.  |
| <code>som_colorcode</code>  | Codificación de colores para el mapa.                                      |
| <code>som_hits</code>       | Histogramas de eventos para el mapa.                                       |

### 3.3.3 Desarrollo

Las funciones básicas para la visualización de SOM son `SOM_SHOW` y `SOM_GRID`. La primera posee tres funciones auxiliares, que son `SOM_SHOW_ADD`, `SOM_SHOW_CLEAR` y `SOM_RECOLORBAR`, las cuales son utilizadas para adicionar o remover marcadores, y controlar la barra de colores. Además, la función `SOM_SHOW` utiliza en realidad la función `SOM_CPLANE` para la realización de las visualizaciones. También la función `SOM_{PIE,BAR,PLOT}PLANE` puede ser utilizada para visualizar SOMs. El resto de funciones listadas no ejecutan ninguna visualización

---

<sup>31</sup> Puede encontrar una definición más amplia de estas funciones consultando el HelpDesk del programa MatLab.

directa, pero sus resultados se emplean en la adecuación de las visualizaciones.

En forma general los objetivos que se pueden alcanzar con este demo son: la visualización de datos multidimensionales, el enlace de visualizaciones, la visualización de datos utilizando SOM, que incluye a su vez la visualización de los clusters en cuanto a la matriz de distancias, además los componentes del mapa (tales como planos componentes, cluster o grupos, búsqueda de correlaciones), permite observar los datos sobre el mapa (por medio de histogramas de eventos, trayectorias), y por último, la aplicación de detalles sobre el manejo del color, y de las formas de los mapas.

### **3.4 Demostración SOM\_DEMO4** Análisis de datos usando SOM

#### 3.4.1 Descripción:

Constituye el complemento para el paso "5" planteado en la segunda demostración. Esta última demostración busca enseñar las diferentes técnicas de análisis de datos, tanto en aspectos de visualización, e imágenes, como de técnicas estadísticas, con diagramas de dispersión e histogramas de eventos.

#### 3.4.2 Desarrollo

En este demo, el conjunto de datos de IRIS es analizado usando SOM. Primero, los datos son leídos desde el archivo ascii, son normalizados, y el mapa es entrenado. Como los datos tenían etiquetas, el mapa será categorizado. Los comandos utilizados en la ejecución de estos pasos son los que siguen:

```
sD = som_read_data('iris.data');  
sD = som_normalize(sD,'var');  
sM = som_make(sD);
```

```
map size [11, 6]
sM = som_autolabel(sM,sD,'vote');
```

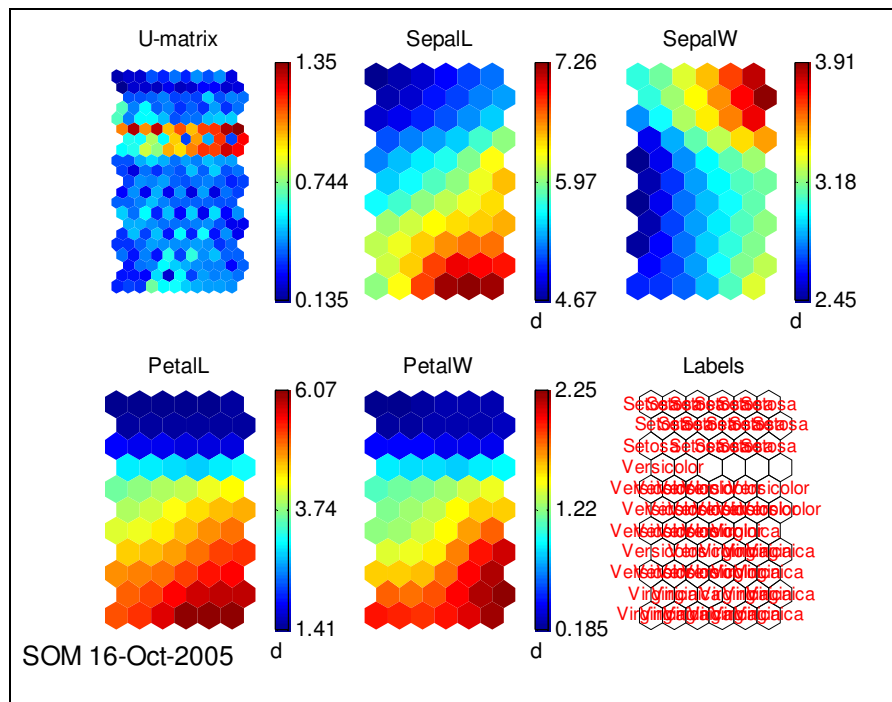
- Inspección visual del mapa.

El primer paso en el análisis de un mapa es la inspección visual. A continuación presentamos a U-Matrix, los planos componentes y etiquetas, con los siguientes comandos:

```
som_show(sM,'umat','all','comp',[1:4],'empty','Labels','norm','d');
som_show_add('label',sM.labels,'textsize',8,'textcolor','r','subplot',6);
```

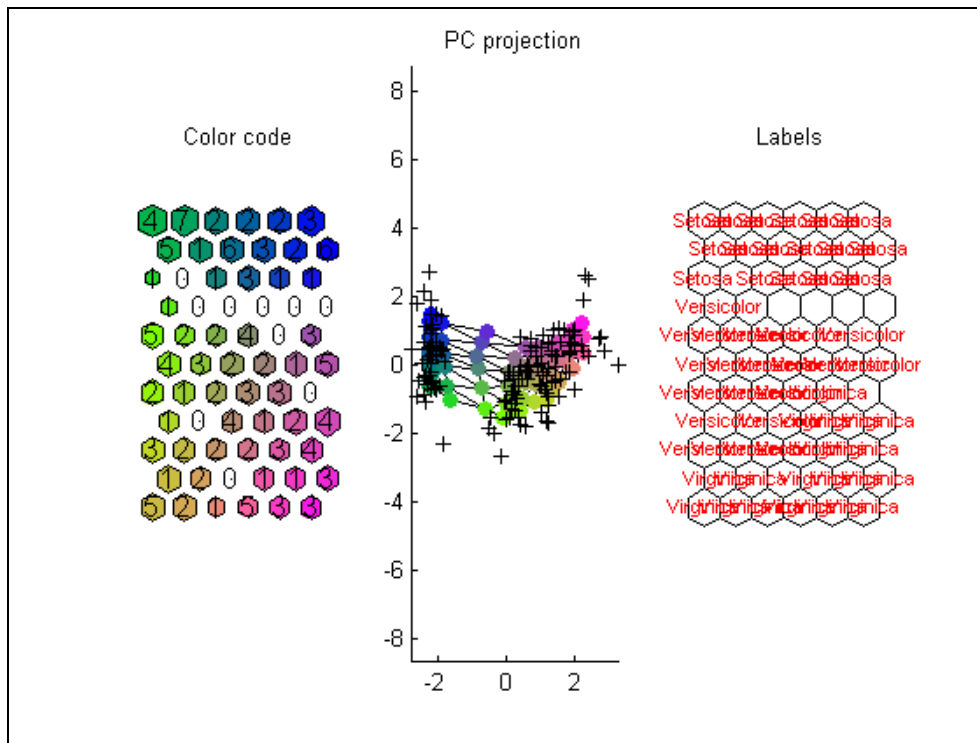
De la primera visualización, en la figura 29, se puede observar lo siguiente: hay esencialmente dos clusters (grupos); PetalL y PetalW están altamente correlacionados, mientras que SepalL está solo un poco. Uno de los grupos corresponde a la especie Setosa y exhibe pétalos pequeños y sépalos cortos y gruesos. El otro grupo corresponde a Virginica y Versicolor, de modo que Versicolor posee hojas más pequeñas (sépalos y pétalos) que Virginica.

FIGURA 29 Primera visualización de los clusters.



A continuación, la proyección del conjunto de datos es investigada. La proyección de un componente principal se realiza para los datos, y luego se aplica al mapa. El color del mapa se hace introduciendo un mapa de colores en la proyección. La información de la matriz de distancias es extraída de U-Matrix, y es modificada por el conocimiento de unidades con cero eventos (interpolación). Finalmente, se muestran tres visualizaciones: el código de colores, con información de grupos y el número de eventos de cada unidad, la proyección y las categorías (etiquetas). Estas visualizaciones se observan en la figura 30. Además, de dicha figura podemos concluir lo siguiente: la proyección confirma la existencia de dos grupos o clusters diferentes, las unidades interpoladas parecen dividir las flores Virginica en dos clases, diferenciándose en el tamaño de las hojas de los sépalos.

FIGURA 30 Visualización del código de colores, proyección y categorías.

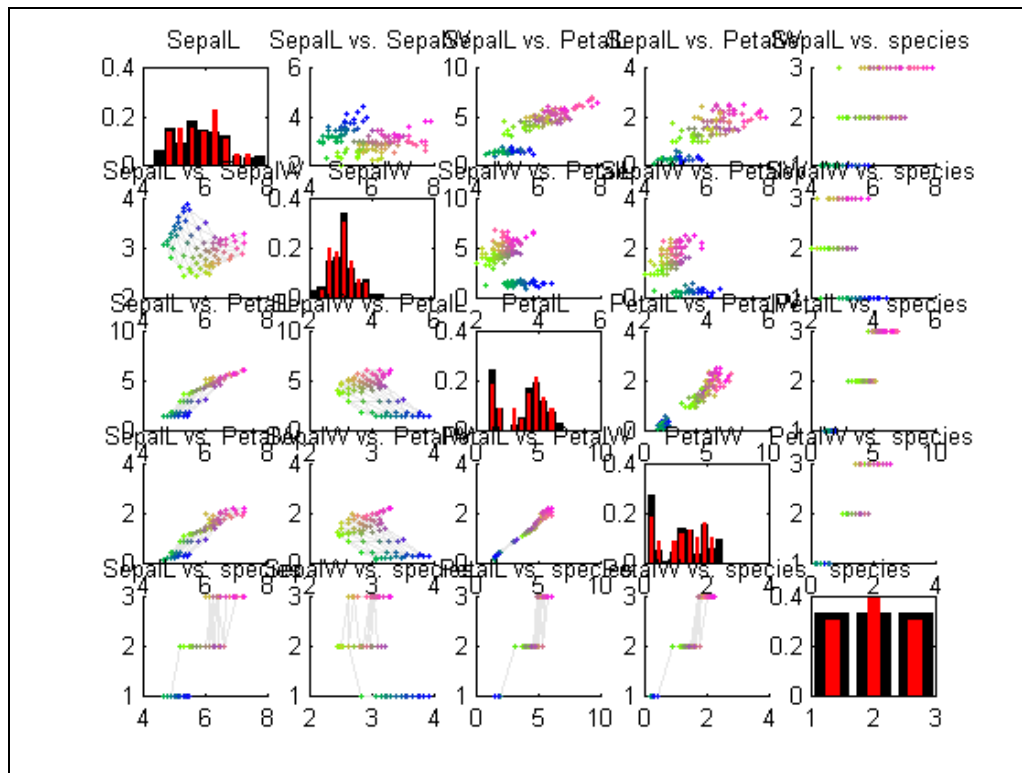


Finalmente, la figura que ofrece la mayor información de todas: una simple scatter plots y el histograma de todas las variables. La información de las

especies es codificada en una quinta variable: 1 para Setosa, 2 para Versicolor y 3 para Virginica. Los puntos de datos originales se encuentran en el triángulo superior, los valores prototipo del mapa en el triángulo inferior, y los histogramas sobre las diagonales: negro para el conjunto de datos y rojo para los valores prototipo del mapa. La codificación de colores de las muestras de datos ha sido copiada desde el mapa (desde la BMU de cada muestra). Note que los valores de las variables han sido desnormalizados.

De esta visualización podemos conformar ciertas conclusiones iniciales, por ejemplo: existen dos clusters: Setosa (azul, negro, verde oscuro) y Virginica / Versicolor (verde claro, amarillo, rojos). PetalL y PetalW poseen una correlación altamente lineal. SepalL esta correlacionada (al menos en el cluster más grande) con PetalL y PetalW. SepalL y SepalW poseen una clara correlación lineal, pero esta un poco alejada de los dos clusters principales. Todo esto se puede observar en la figura 31 a continuación:

FIGURA 31 Scattered plots e histograma de eventos



- Clasificación del mapa

Después de la inspección visual hemos llegado a determinar que existen por lo menos dos grupos o clusters en los datos, y que las propiedades de dichos grupos son diferentes entre sí (específicamente la relación de SepalL y SepalW). Para investigaciones futuras, el mapa necesitará ser dividido. A continuación la función `KMEANS_CLUSTERS` se utiliza para encontrar una división inicial. Las líneas a continuación muestran el índice de grupos Davies-Boulding, el cual es minimizado con la mejor clasificación.

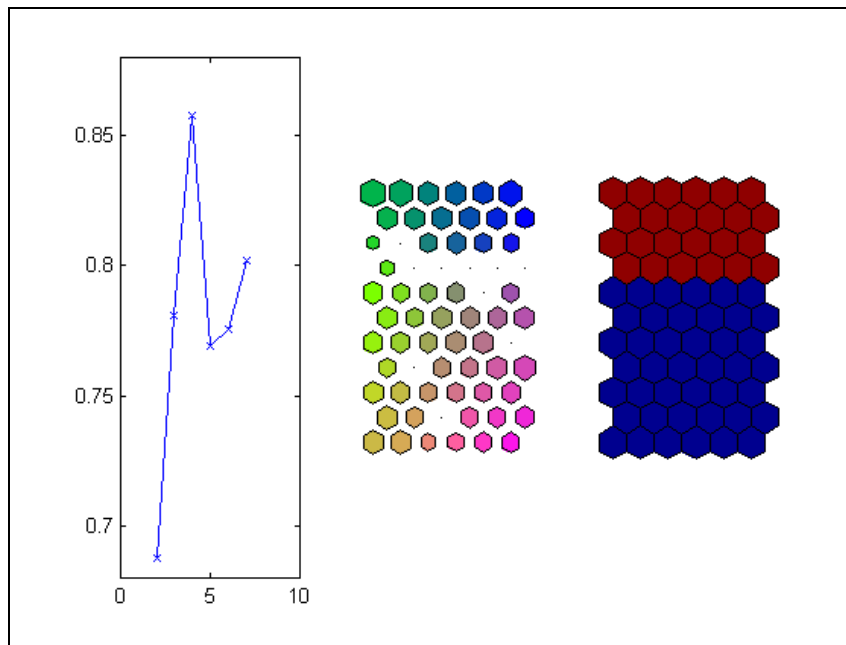
```
subplot(1,3,1)
[c,p,err,ind] = kmeans_clusters(sM, 7); find at most 7 clusters
plot(1:length(ind),ind,'x-')
[dummy,i] = min(ind)
dummy = 0.6873
i = 2
cl = p{i};
```

El índice de Davies-Boulding parece indicar la existencia de dos clusters en el mapa. A continuación, en la figura 32, se muestra la información de la clasificación calculada previamente, y los resultados de la división.

```
subplot(1,3,2)
som_cplane(sM,Code,Dm)
subplot(1,3,3)
som_cplane(sM,cl)
```

Se pudo haber utilizado también la función `SOM_SELECT` para modificar manualmente la división. Después de esto, el análisis procedería con la sumatoria de los resultados, y el análisis de cada cluster individualmente. Sin embargo, el ToolBox SOM no cuenta con las funciones para la realización de esta actividad.

FIGURA 32 Información de la clasificación y resultados de la división.



- Modelamiento

Se pueden construir además modelos sobre los SOM, los cuales corresponden típicamente a modelos de vecindades cercanas o locales. A continuación, SOM es utilizado para estimación de la densidad de probabilidad. Cada prototipo del mapa es el centro de una distribución gaussiana, cuyos parámetros son estimados de los datos. El modelo gaussiano es estimado a partir de la función SOM\_ESTIMATE\_GMM y las probabilidades pueden ser calculadas con SOM\_PROBABILITY\_GMM. Los comandos son los siguientes:

```
[K,P] = som_estimate_gmm(sM,sD);
```

```
[pd,Pdm,pmd] = som_probability_gmm(sD,sM,K,P);
```

A continuación se muestra el valor de la función de densidad de probabilidad para la primera muestra de datos ( $x=sD.data(:,1)$ ) en términos de cada unidad del mapa. La figura 33 muestra los resultados.

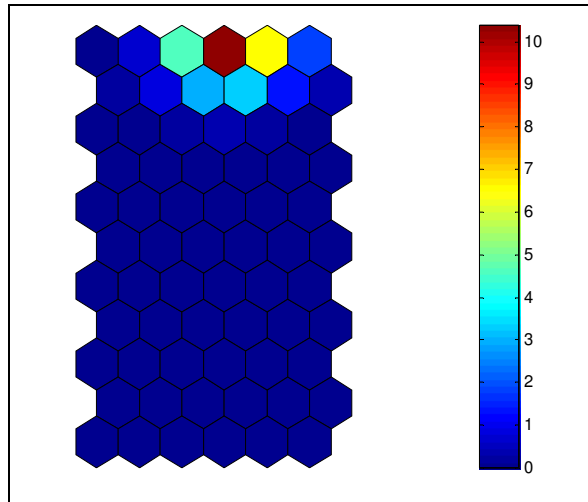
```
som_cplane(sM,Pdm(:,1))
```

```

colorbar
title('p(x|m)')

```

FIGURA 33 Resultado de la función de probabilidad para la primera muestra.



En este caso, SOM es utilizado para clasificación. Aunque SOM puede cumplir este objetivo, se debe recordar que SOM no utiliza información de clases, por lo que la clasificación puede ser algo defectuosa. Sin embargo, con pequeñas modificaciones, la red puede llevar las clases a una cuenta, a través de la función SOM\_SUPERVISED. A continuación, dicha función es empleada en la creación de un clasificador para el conjunto de datos IRIS. Las líneas son las siguientes, y la grafica asociada corresponde a la figura 34:

```

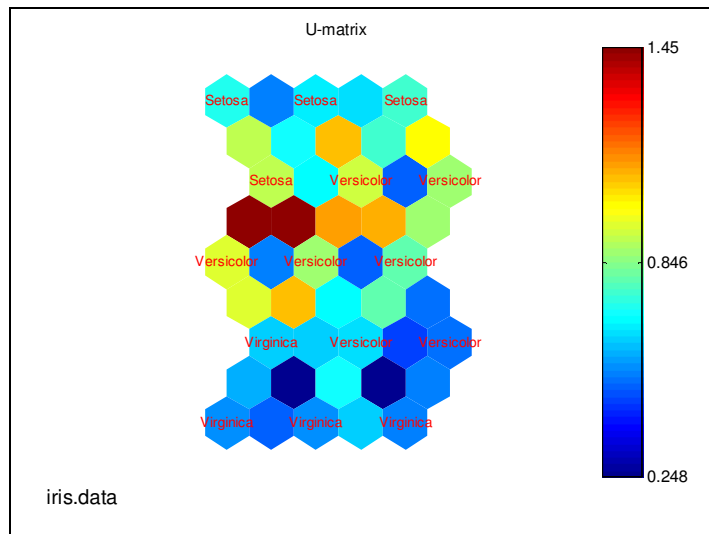
sM = som_supervised(sD,'small');
map size [5, 3]
Training using batch algorithm...
som_show(sM,'umat','all');
som_show_add('label',sM.labels,'TextSize',8,'TextColor','r')
sD2 = som_label(sD,'clear','all');
sD2 = som_autolabel(sD2,sM);      classification

```



```
ok = strcmp(sD2.labels,sD.labels); errors
100*(1-sum(ok)/length(ok))      error percentage ()
ans = 8.0000
```

FIGURA 34 Clasificador para el conjunto de datos IRIS.



### 3.5 Programas simuladores de redes neuronales

En la Web se encuentran multitud de programas simuladores de redes neuronales de libre distribución (gratuitos o *sharewares*). Existen varias empresas que los ofrecen, como por ejemplo se encuentra el listado ofrecido por el grupo de noticias sobre redes neuronales [comp.ai.neural-nets](http://comp.ai.neural-nets), por otro lado, el listado ofrecido por el Pacific Northwest National Laboratory. A continuación se mencionan algunos de los simuladores más utilizados:

- **LVQ\_PAK, SOM\_PAK**

Estos son paquetes de Learning Vector Quantization y Self Organizing maps, respectivamente. Han sido programados por el equipo de programación LVQ/SOM de la universidad tecnológica de Helsinki. Por el grupo de Kohonen para el trabajo y experimentación con el modelo SOM. Existen Versiones para UNIX y MS-DOS. Es uno de los más destacados.

- **GENESIS**

GEneral NEural Simulation System. Es una plataforma de simulación desarrollada para soportar la simulación de sistemas neuronales desde complejos modelos de neuronas sencillas hasta simulaciones de grandes redes constituidas por componentes neuronales más abstractos. Ejecutable en la mayor parte de las plataformas UNIX.

- **Nenet v1.0**

Es una aplicación a 32 bits para Windows 95 y NT 4.0 diseñada para facilitar el uso de un algoritmo Self-Organizing Map. Consta de una interfaz muy eficiente y atractiva.

- **Multi-Module Neural Computing Environment (MUME)**

Es un simulador para computación neuronal multi-modular. Proporciona una herramienta orientada a objetos para la simulación y ejecución de múltiples redes con varias arquitecturas y algoritmos de aprendizaje. Puede utilizarse para simulaciones de RN de gran escala

puesto que proporciona un soporte para aprendizaje en entornos multi-redes. Soporte para X Windows. Distribución libre para instituciones educativas tras enviar la licencia firmada.

- **Adaptive Logic Network Educational Kit**

Permite desarrollar aplicaciones simples usando Redes Lógicas adaptativas (ALN). La versión Educacional Kit es la misma que la comercial excepto que no permite más que dos variables de entrada y que se registra sólo para uso educativo.

- **AINET**

Es una aplicación de RN probabilística ejecutable bajo Windows 95/NT. Fue diseñada específicamente para facilitar las tareas de modelización en los problemas de las RN. Es muy rápida, no hay límite en el número de variables, ni en el tamaño de la muestra. Puede descargarse también un manual.

- **Aspirin/MIGRAINES**

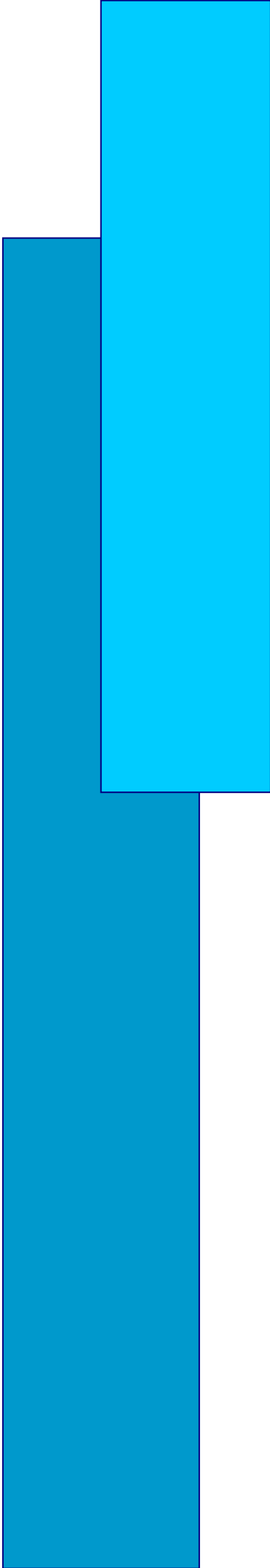
Consiste en un generador de código que construye simuladores de RN tras leer una descripción de la red (escrita en un lenguaje llamado "Aspirin") y genera una simulación en C. Una interfase permite exportar datos desde la RN a herramientas visuales. Funciona en varias plataformas.

- **BIOSIM**

Es un simulador de RN con orientación biológica. De dominio público, ejecutable en UNIX, fácil de instalar, en inglés y alemán, tiene una interfase gráfica de usuario, diseñado para investigación y enseñanza, proporciona herramientas de ayuda online y una DEMO.

- **DemoGNG**

Este simulador está escrito en Java y debería ejecutarse sin compilación en todas las plataformas con un intérprete Java. Implementa varios algoritmos. Consta de una interfaz gráfica donde se introducen los parámetros del modelo. No permite introducir datos del usuario, por lo que está orientado a la enseñanza.



## Capítulo 4

### Aplicaciones de los SOM

*Conocer las principales aplicaciones de los Mapas Auto-Organizados SOM, haciendo énfasis en la organización topológica de documentos, la cual ha presentado un gran desarrollo consistente con el incremento extraordinario de la información disponible para cualquier usuario.*

Las redes neuronales pueden utilizarse en un gran número y variedad de aplicaciones, tanto comerciales como militares. Se pueden desarrollar redes neuronales en un periodo de tiempo razonable, con la capacidad de realizar tareas concretas mejor que otras tecnologías. Cuando se implementan mediante hardware (redes neuronales en chips VLSI), presentan una alta tolerancia a fallos del sistema y proporcionan un alto grado de paralelismo en el procesamiento de datos. Esto posibilita la inserción de redes neuronales de bajo coste en sistemas existentes y recientemente desarrollados.

Hay muchos tipos diferentes de redes neuronales; cada uno de los cuales tiene una aplicación particular más apropiada. La ventaja que presentan las redes neuronales frente a otros procesos reside en el procesado paralelo, adaptativo y no lineal.

El dominio de aplicación de las redes de SOM va encaminado a la clasificación y/o reconocimiento de patrones y la extracción de características, basándose en una regla de aprendizaje con cierto grado de sensibilidad con respecto al vecindario, haciendo que el número de neuronas que no aprenda desaparezca, aumentando así su capacidad para extraer o mapear características topológicas de los datos.

Los mapas de Kohonen son aplicados<sup>32</sup> en diferentes ciencias como:

- *Visión y análisis de imágenes:* visión de robots, transmisión de imágenes, generación de imágenes médicas, codificación, compresión y segmentación de imágenes.
- *Aplicaciones ópticas:* Caracteres ópticos, reconocimiento de caracteres manuscritos, etc.

---

<sup>32</sup> Estas aplicaciones pueden ser consultadas en el anexo magnético de este trabajo investigativo.

- *Análisis y reconocimiento de voz*: reconocer palabras y habla continua, identificar personas por el habla, etc.
- *Estudios acústicos y música*: reconocer señales acústicas, análisis de contaminación acústica en ciudades, etc.
- *Procesamiento de señales y sistemas de medida de radar*:
- *Sistemas de medida*: detección de minas, detectar condiciones de fallos en anestesia, enrutamiento de tráfico, etc.
- *Control de procesos*: Identificación del estado de proceso, detección y diagnóstico de errores, etc.
- *Robótica*: Sistemas de navegación para robots, etc.
- *Química*: clasificar estructuras químicas complejas y proteínas, extraer características de cromosomas, etc.
- *Física*: Espectro de rayos infrarrojos, problemas de inversión geofísica, clasificar fenómenos sísmicos, etc.
- *Diseño de circuitos electrónicos*: Diseño de circuitos VLSI, eliminación de componentes defectuosos, etc.
- *Aplicaciones médicas sin procesamiento de imágenes*: análisis de encefalogramas, estudios biomagnéticos, etc.
- *Procesamiento de datos*: Análisis de datos financieros y económicos, compresión de datos, etc.
- *Problemas lingüísticos y de inteligencia artificial*: Clasificar palabras en categorías, formar y reconocer frases.
- *Problemas matemáticos*: Clustering, aproximar funciones, predicción de series temporales, TSP, etc.
- *Investigación neurofisiológica*: Clasificación del sueño.

En esta oportunidad se quiere hacer énfasis en la organización topológica de documentos por ser esta aplicación la que está tomando mayor auge, debido, entre otras cosas, al incremento extraordinario de la información disponible, muchos grupos de investigación han desarrollado sus trabajos

basándose en las características de SOM, algunas de las investigaciones disponibles son las siguientes:

#### **4.1 Visualmaps De Xia Lin.**

Xia Lin, investigador de la Escuela de Biblioteconomía de la Universidad de Kentucky, utiliza estos mapas con el fin de generar una salida visualizable de una determinada colección de documentos. La representación documental la obtiene de la siguiente forma:

- 1.- Construcción de una lista que incluya todos los términos que aparecen en los títulos y resúmenes de todos los documentos de la colección.
- 2.- Supresión de términos irrelevantes del lenguaje a partir de una lista de palabras vacías.
- 3.- Transformación de los términos en la raíz mediante un algoritmo de stemming para reducir la lista.
- 4.- Eliminación de términos de frecuencias altas y bajas. Como dijo Luhn la significación de un texto está depositada sobre las palabras de frecuencias intermedias.
- 5.- Construcción de un vector para cada documento con tantas componentes como términos han quedado en la lista. Las componentes se generan de tres formas diferentes (dependiendo de la que se escoja se obtienen distintas visiones de la base):
  - a) Dígitos binarios (uno sí el término correspondiente está en el documento y cero sino está)
  - b) Pesos proporcionales a la frecuencia del término en el documento.
  - c) Pesos proporcionales a la frecuencia del término en el documento e inversamente proporcionales al número de documentos en los que aparece el término.

Terminado el entrenamiento se le asigna a cada neurona el término más cercano a su vector de pesos. Uniendo las neuronas cuyo término asignado es el mismo tenemos la rejilla dividida en distintas zonas (que se pueden etiquetar con el término en cuestión). De esta forma finalmente tenemos los documentos clasificados en una superficie dividida en una serie de zonas etiquetadas por términos.

Lin utiliza esta clasificación como interfaz para el browsing de bases documentales pequeñas, que pueden incluso ser los resultados de una búsqueda realizada. Proporciona prototipos que pueden ser consultados en su propia página personal <http://www.ukv.edu/~xLin/> esta pagina tiene como función la ubicación de documentos (únicamente en el idioma ingles) mediante el uso de índices de contenidos basada en el lenguaje JAVA.

Varios sitios web conocidos cuentan con un mapa de este tipo como índice automático de contenidos: Yahoo, McDonnell-Douglas, etc.<sup>33</sup> Una vez etiquetadas las distintas zonas de la base, se puede seleccionar aquella que resulte de nuestro interés. Dichos prototipos están realizados en *java*, de forma que incorpora, además, *dos controles*, barras de desplazamiento:

- La horizontal regula la aparición de puntos que representan a los documentos. Estos puntos aparecen en el lugar que le corresponde a cada documento dentro de la rejilla. Aparecen inicialmente aquellos documentos que logran una mayor activación de la neurona en la que se encuentran clasificados. A medida que se desplaza la barra disminuye el umbral que necesitan superar los documentos para aparecer, de modo que aparece un mayor número.

- La barra de desplazamiento vertical tiene la misma función, pero, en este caso para los términos. Es decir, a medida que se desplaza disminuye el

---

<sup>33</sup> Lin 1995 y 1996, 1997

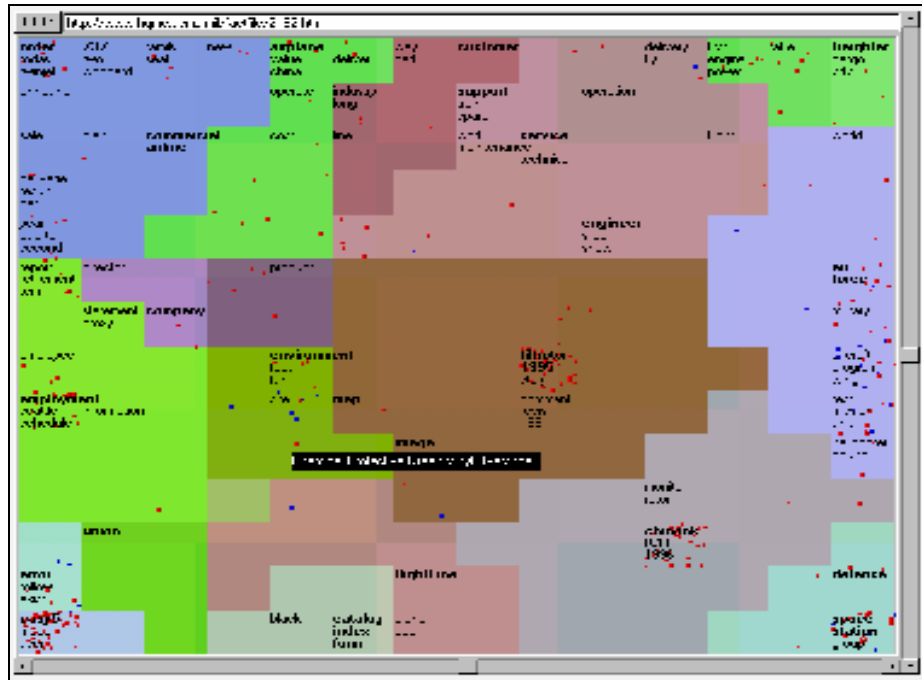


umbral de activación que tienen que superar para aparecer en el mapa. Con ello aumenta el número de los que aparecen, dejando así el mapa más etiquetado.

La forma de proceder para Lin sería:

- Comenzar con muy pocos términos en el mapa, de forma que sea fácil localizar aquellos que describan mejor nuestras necesidades.
- A medida que vayamos necesitando términos más específicos podemos *ir aumentando el número de estos*.
- Con el otro control podemos ver también el número de documentos presentes en la zona para ampliar o reducir la recuperación.
- Si no se encuentran documentos que satisfagan nuestras necesidades de información se puede ampliar la búsqueda en la dirección conveniente. Con el *ratón* se puede *seleccionar* la zona que se quiera, apareciendo en ese caso una ventana con enlaces que contienen el título, a los distintos documentos. Una imagen de este interfaz correspondiente a una base generada con todos los documentos clasificados en la categoría Space Science, la podemos ver en la figura 35.

FIGURA 35. Visual Sitemap realizado por Xia Lin de los documentos pertenecientes a la categoría Space Science de Yahoo [<http://lislin.gws.uky.edu/Sitemap/spaceSmall.htm>]



Lin establece cuales son las limitaciones del modelo, entre las que se encuentran: incapacidad para trabajar con grandes volúmenes de información y alto costo de procesamiento de la información. No obstante, la aplicación de este modelo parece ser una de las aplicaciones más prometedoras en la clasificación automática mediante redes neuronales. En <http://www.uky.edu/~xLin/> es posible encontrar más información sobre el asunto, incluyendo una lista de sitios web clasificados por este método.

#### 4.2 ET-MAP<sup>34</sup>

El profesor Chen en el Laboratorio de Inteligencia Artificial de la Universidad de Arizona supervisa la realización de un proyecto llamado ET-Map. En él se

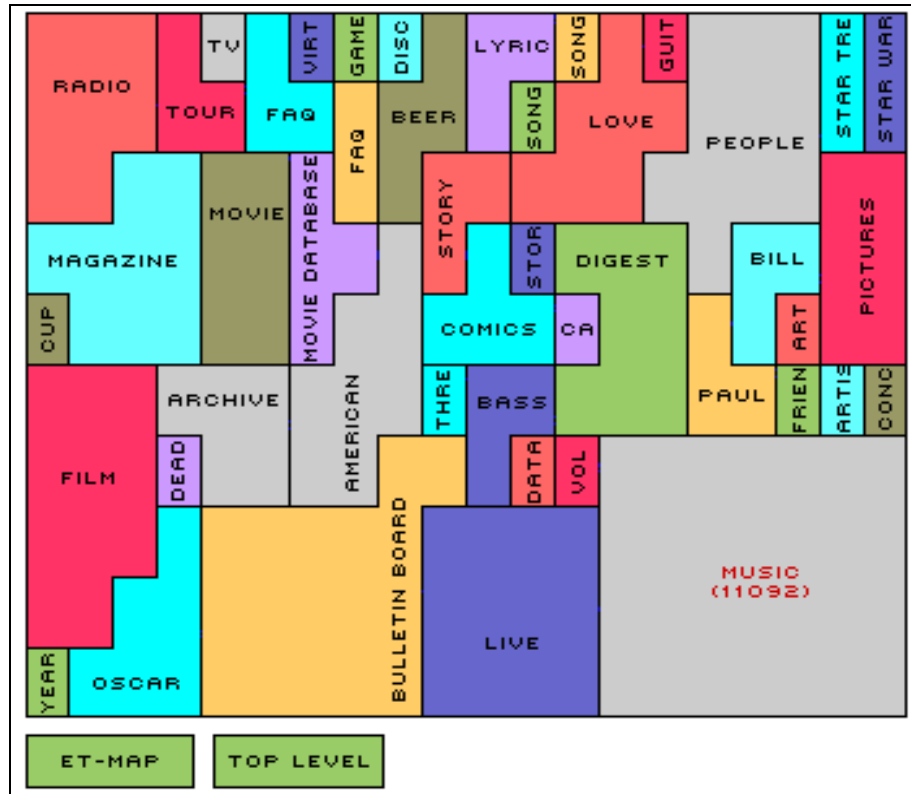
<sup>34</sup> Comparar también con la pagina: [http://mappa.mundi.net/maps/maps\\_009/](http://mappa.mundi.net/maps/maps_009/) de Martin Dodge, contien un pequeño resumen de lo que es ET\_MAP en ingles

ha hecho un proceso similar, de todos documentos pertenecientes a la subcategoría de “Entertainment” del índice Yahoo. Está disponible en Internet [<http://ai2.BPA.arizona.edu/ent>] esta pagina tiene el uso restringido pero representa una diversidad de acoplamientos de sitios web proporcionando una herramienta visual intuitiva de mucha información. En este caso la interfaz es un mapa sensitivo donde se pueden observar las distintas regiones de la rejilla (generadas de la misma forma que en el caso anterior). En ellas solamente se indica el término ganador en toda la región, así como el número de elementos que contiene, no pudiendo así aumentar el número de términos presentes como permite el interfaz de Lin. ET-Mapa, se explora, usando el estilo familiar de navegación de la Web, con un solo “clic” del mouse puedes encontrar la información de tu interés.

Este mapa tiene dos niveles, es decir, al seleccionar una región aparece otro mapa de los documentos pertenecientes a la misma. Si las regiones resultantes contienen un gran número de documentos se genera otro mapa. Cuando se accede a una región del mapa correspondiente al último nivel aparecen los enlaces a los distintos documentos. En resumidas cuentas en este sugerente esquema el nivel superior es como un **mosaico en el que los distintos dominios adoptan formas poligonales de lados paralelos**. Cada dominio tiene una palabra que define la categoría. Si pulsamos sobre un dominio determinado se abre una segunda pantalla que contiene otro mapa similar pero ahora ya restringido a los documentos de ese dominio particular. El proceso se repite hasta que llegamos a un nivel de detalle suficiente como para que aparezca un listado tradicional con los documentos que componen ese subdominio concreto.

Han realizado experimentos de browsing comparándolo con el correspondiente índice humano de Yahoo. Los resultados son muy parecidos si no se va buscando nada en particular, siendo peores si se busca algo específico. Podemos ver una imagen del interfaz en la figura 36.

FIGURA 36. Mapa sensitivo general realizado en el proyecto ET-Map bajo la dirección del profesor Chen [<http://ai2.bpa.arizona.edu/ent/entertain1/>].



Cada región es un resumen visual de un grupo de páginas web de contenido similar, la diversidad de colores se usa para distinguir las regiones, la etiqueta identifica el tema y el tamaño de cada región se relaciona directamente con el número de las páginas del Web en esa categoría.

#### 4.3 WEBSOM<sup>35</sup>

El propio Teuvo Kohonen dirige un grupo finlandés perteneciente al Centro de Investigación en Redes Neuronales de la Universidad Tecnológica de Helsinki que está utilizando este tipo de redes tanto para hacer clasificaciones de términos como de documentos. El *Word Category Map*



Como hemos dicho anteriormente también han realizado un sistema de clasificación documental, que se puede utilizar como interfaz para acceder a la información. Este lo han denominado *WEBSOM*<sup>36</sup>, y es un sistema pensado para clasificar un gran número de documentos, que lo han aplicado a Internet.

Para la representación de cada documento se siguen los siguientes pasos:

- 1.- Se eliminan las palabras de alta y baja frecuencia, con el fin de reducir el procesamiento computacional y eliminar ruido.
- 2.- Con estas palabras se genera *un mapa de categorías de palabras* (Word category map).
- 3.- Se construye un histograma para cada documento indicando el número de palabras de cada categoría que contiene.

El algoritmo funciona de la siguiente forma: primero, se crea un mapa de palabras; el objetivo es desambiguar los significados de las palabras, detectando sinónimos y otras relaciones semánticas. Para ello, se entrena un mapa de Kohonen con cada palabra representada por su contexto promediado: se toma cada palabra y se le asigna un vector n-dimensional aleatorio. Una vez hecho eso, se calcula para cada palabra la que le antecede y sucede, teniendo en cuenta finales de frase y principios como si fuera otra palabra. Una vez tomadas todos los antecesores y sucesores, se promedian; la representación de la palabra que se usará será los vectores antecesores y sucesores promedio yuxtapuestos.

Cualquier otro método que sirva para asignar un vector único dependiente del contexto a cada palabra sirve también. Por ejemplo, si tenemos los textos

---

<sup>36</sup>Comparar también con la pagina: <http://geneura.ugr.es/~jmerelo/tutoriales/bioinfo/Kohonen.html> Tutorial de mapas autoorganizativos de Kohonen de J.J Merelo

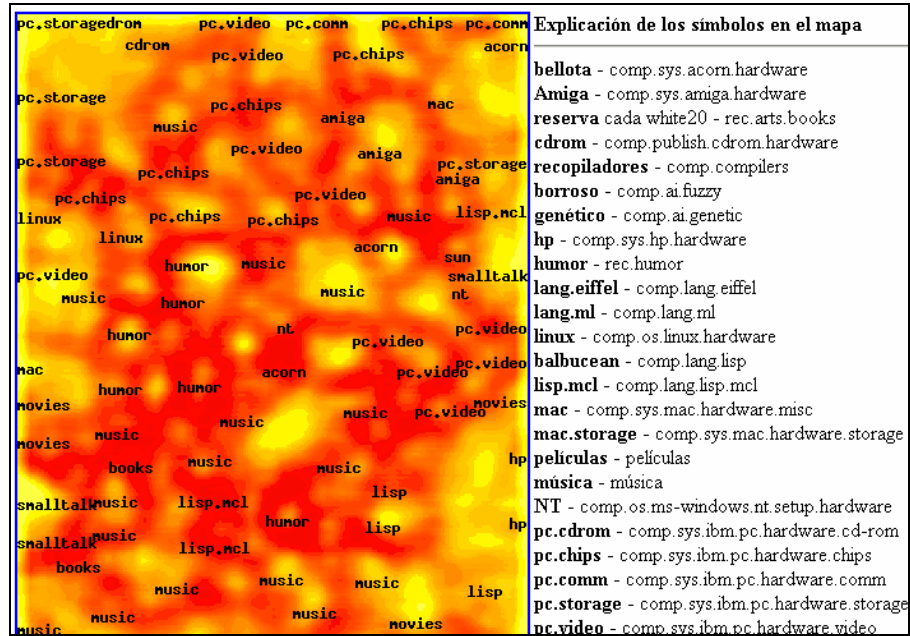
categorizados (por ejemplo, secciones de un periódico), se pueden usar para cada palabra un vector de las frecuencias con las que aparece en cada sección.

También se pueden considerar contextos más extensos: de 4 palabras, en vez de dos palabras. O se puede representar cada palabra con un vector que represente concurrencia de una palabra en un documento determinado; esta codificación se usa para el análisis semántico latente. Dependiendo de la aplicación, será necesario uno u otro.

La diferencia con respecto a los otros es que para aliviar el proceso de cálculo reduce el número de componentes de los vectores documentales representados en función de las categorías en lugar de las palabras. Los vectores generados se utilizan también para entrenar una red de Kohonen que los organiza temáticamente en dos dimensiones. Tiene un interfaz gráfico bastante agradable, en el que también se han incorporado etiquetas descriptivas generadas automáticamente puede consultarse un prototipo del mismo en Internet <http://websom.hut.fi/websom/> esta pagina te lleva a un demo de WEBSOM, en donde encontraras documentos sobre el newsgroup de USENET. Y más información sobre la exploración de documentos con WEBSOM.

En la figura 38 podemos ver el mapa general realizado para una colección de 4600 artículos de USENET pertenecientes al grupo de discusión de *comp.ai.neural-nets*. Tenemos que tener en cuenta que junto a este mapa se facilita un índice donde se especifica el significado de cada etiqueta.

FIGURA 38 Mapa de primer nivel correspondiente al WEBSOM aplicado al grupo de discusión de usenet d comp.ai.neuronal-nets [<http://websom.hut.fi/websom/comp.ai.neural-nets-new/html/root.html>]

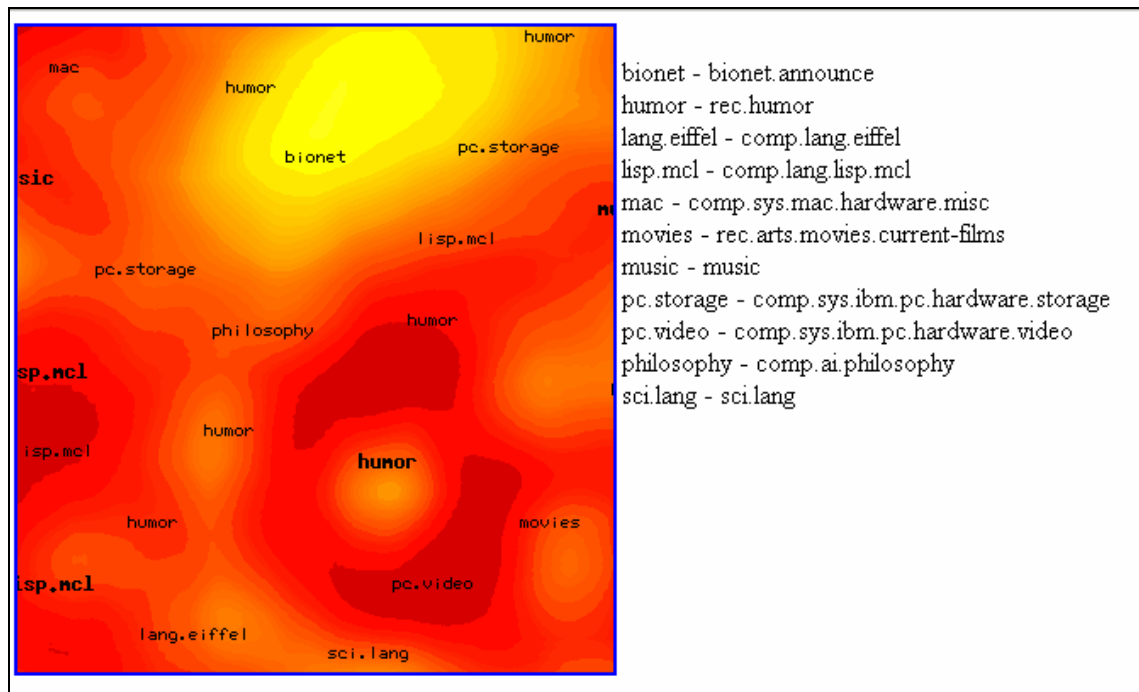


En WEBSOM, cada documento se representa en el mapa del documento como punto de una manera tal que la distancia mutua entre cualquier dos puntos de la representación refleje la semejanza de los dos histogramas correspondientes. Por lo tanto los documentos similares se encuentran cerca uno del otro en el mapa del documento, como los libros en los estantes de una biblioteca bien organizada.

En la figura 39 podemos observar un zoom del centro del mismo mapa que es el mapa enfocado:



FIGURA 39 Mapa de segundo nivel (enfoque)

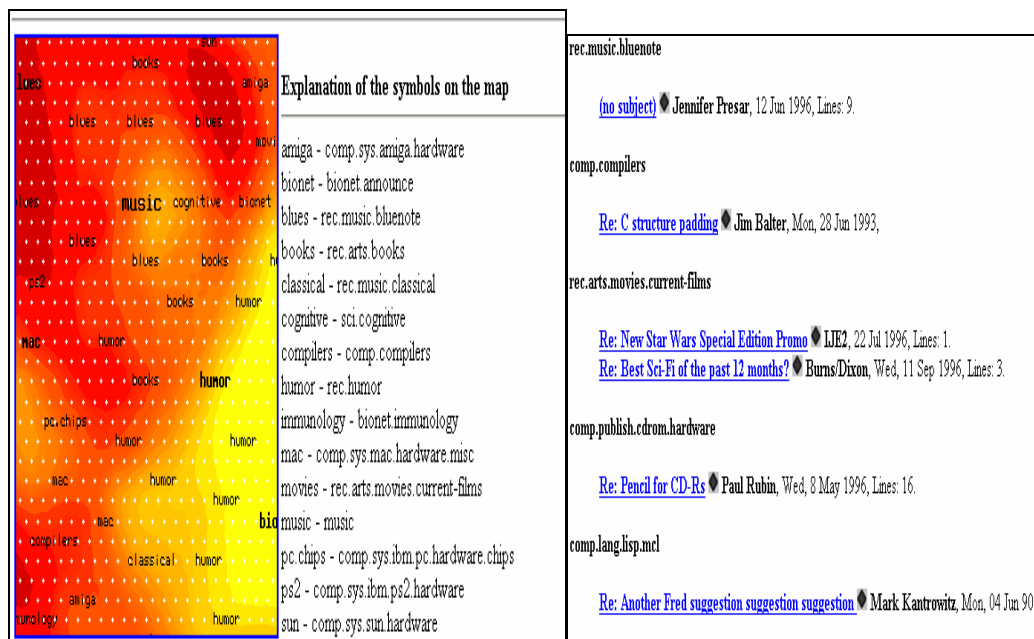


Mediante la intensidad del color, que se puede ver en las figuras, se indica las distancias entre los prototipos de los distintos nodos de cada zona. De este modo se puede interpretar de manera similar a un mapa topográfico:

- Una superficie rugosa donde los colores oscuros representan distancias mayores entre los nodos.
- Los colores claros representarían los valles, zonas con nodos más cercanos.

Según el equipo de desarrollo es en los valles donde se pueden encontrar las discusiones más intensas. En el nodo del mapa también se aprecian como pequeñas estrellas blancas los nodos existentes. Estas son las poblaciones donde se agrupan los documentos. Si se selecciona uno de estos nodos se obtiene una página con enlaces a cada uno de los documentos ubicados en ella. Como se puede apreciar en la figura 40:

FIGURA 40 Mapa de segundo nivel (enfoque)



Lo que se persigue con esta aplicación no es solo la clasificación de documentos, sino también el crear un mapa de un corpus de documentos, de forma que se pueda navegar por él entre documentos similares, y, además, ver la estructura a gran escala del grupo de documentos, con el objeto de detectar grupos naturales (los clusters de los que hemos hablado anteriormente).

#### 4.4 Los mapas autoorganizados aplicados a la bibliometría.

La bibliometría es una disciplina que estudia los aspectos cuantitativos de la información registrada. Para ello se han creado una serie de modelos estadísticos que aportan datos numéricos sobre el comportamiento de la actividad científica. También se han adaptado modelos de otras disciplinas para facilitar los análisis y representar los resultados desarrollados a partir de la bibliometría. Los mapas auto-organizados (SOM) o modelo de Kohonen

son una de estas herramientas. En los estudios métricos la aplicación de las redes neuronales, y específicamente los SOM, están asociados en lo fundamental con la clasificación de información, ósea, la formación de cluster y su representación en mapas bidimensionales de conceptos y más específicamente con el descubrimiento de información (*data mining*). Este último vinculado con la recuperación de la información con "ruido" e incompleta o con el tratamiento de información que incluye diferentes tipos de datos (números, texto, registros estructurados, etc.). Los SOM facilitan que el conocimiento tácito se haga explícito, a partir de la extracción no-trivial (a partir de los datos) de conocimientos implícitos potencialmente útiles desconocidos previamente. Se podrán encontrar patrones o estructuras en el conocimiento tácito. Las investigaciones bibliométricas, a través de la utilización de las redes neuronales, incursionan en:

- la selección de variables,
- clasificación de información o formación de cluster,
- regresión,
- relaciones entre variables,
- cambios y desviaciones,
- Representación de las variables.

Lo anterior se puede ejemplificar a partir de algunas aplicaciones prácticas relacionadas con la evaluación de páginas web y trabajos relacionados con la clasificación de revistas en un determinado campo temático. Se conocen, además, investigaciones relacionadas con la minería de textos (*text mining*) sobre todo aplicado a la asociación de palabras o co-word. En todos estos ejemplos se utiliza como variante de las RNA el modelo de los mapas autoorganizativos (*self-organizing map*, SOM). En un análisis, realizado por los autores sobre el tema, se examinaron cerca de 56 documentos sobre redes neuronales aplicadas al análisis de información, con ello se constato que la mayoría utilizaban el modelo SOM como herramienta de estudio.

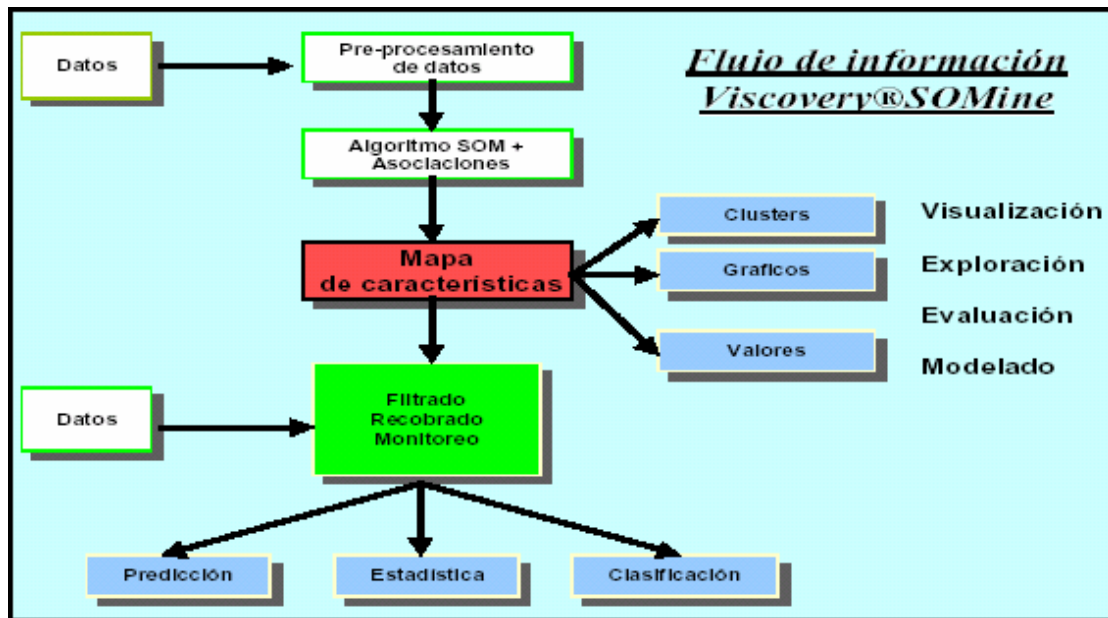
Un ejemplo de SOM podría ser el estudio de una temática determinada, para este caso en un mapa cada documento (artículo de revista, podría ser una patente, una tesis, etc.) va a ocupar un lugar en el espacio, en función de su contenido temático. Cada área del mapa va a reflejar un contenido específico y los tópicos van variando levemente a lo largo del mismo. Las diferentes tonalidades indican la densidad de documentos, cuanto más oscura más documentos se encuentran. Este uso frecuente de los SOM quizás se deba a lo amigable de la interfaz de los mapas para los usuarios finales y a la diversidad de sus utilidades prácticas, estas representaciones son válidas para poder identificar, además de los desarrollos temáticos antes mencionados, relaciones entre áreas temáticas y publicaciones, alianzas estratégicas y características de la cooperación. Permite, también, visualizar los avances tecnológicos que tienen lugar en un período, conocer la evolución de una tecnología a través del tiempo e identificar campos emergentes.

La arquitectura SOM se ha extendido a infinidad de aplicaciones, es por ello que este algoritmo y sus modelos se han automatizado para dar vida a varios software como son el Viscovery SOMine<sup>37</sup> y el WEBSOM. Ambos son utilizados en el análisis y filtrado de información, el Viscovery ha sido validado tanto en estudios de mercado como análisis financiero o proyecciones urbanísticas. Actualmente está incursionando en el tratamiento de la información usando herramientas diseñadas por la bibliometría. Este sistema es utilizado por un equipo de trabajo del Instituto Finlay para elaborar mapas científico-tecnológicos. La lógica de funcionamiento del Viscovery SOMinẽse muestra en la Figura 41.

---

<sup>37</sup> Comparar también con él artículo *Viscovery@SOMine*, Visual Information Discovery and Exploration por Eudaptics

FIGURA 41 Funcionamiento del Viscover SOMine



El punto de partida, para el uso de esta herramienta, es la entrada de un conjunto de datos numéricos (datos multivariables, variables, “nodos”). Estos datos necesitan ser preprocesados con el objetivo de “organizarlos” en forma de matrices. Los datos son convertidos hasta obtener una información visual en forma de mapa, para ello se aplica un número de técnicas de evaluación como coeficientes de correlación entre variables o factores discriminantes. Los mapas serán amigables a la vista del usuario final, en ellos se identificarán dependencias entre parámetros, cluster y gráficos que facilitarán diferentes predicciones o el proceso de monitoreo.

Se estima que a pesar de las limitaciones técnicas, las redes neuronales aplicadas a la Bibliometría constituyen un campo de investigación muy prometedor. Un ejemplo es presentado a continuación.

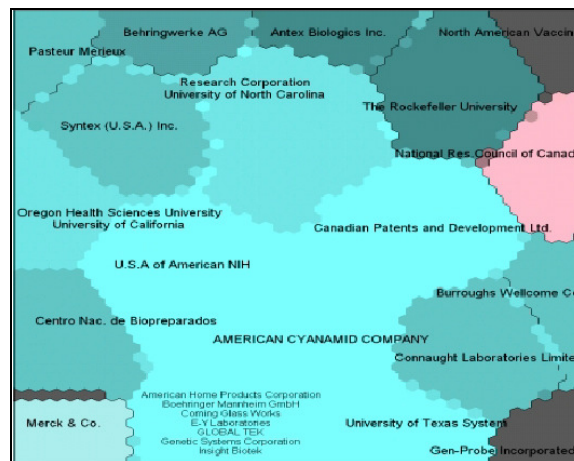
La multidisciplinaridad de las redes neuronales es aplicada en esta sección, donde se asume a la producción de los documentos de patentes como

indicador de la capacidad de desarrollo industrial. El objetivo es identificar posibles competidores, alianzas estratégicas, dependencia tecnológica, etc.

Se escogió para el primer ejemplo<sup>38</sup> la representación de la situación tecnológica de la *Neisseria meningitidis* (entre las bacterias causantes de la meningitis cerebroespinal, la *Neisseria meningitidis* es el agente causal más frecuentemente encontrado. La enfermedad se presenta en todo el mundo y se manifiesta de forma endémica o epidémica).

Las diferentes instituciones en la primera hoja de sus patentes hacen referencias a otras patentes, a partir de estos datos se puede inferir el impacto que produce una tecnología o institución en otra. Con el objetivo de determinar la dependencia tecnológica entre instituciones, se realizó un análisis de citas, estas formaron los cluster que aparecen en la figura 42.

FIGURA 42 Posición tecnológica de las instituciones según las citaciones



El mapa tecnológico presentado en la figura anterior representa a tres cluster: cluster 1 formado solamente por la *Merck & Co.*, un cluster 2 formado

<sup>38</sup> Comparar también con el artículo Contribución al estudio de las revistas de América Latina y el Caribe mediante el mapeo autoorganizado, por Oscar Saavedra, Gilberto Sotolongo y María V Guzmán, 2002.

únicamente por el *National Res. Council of Canadá* y el cluster 3 que incluye al resto de las instituciones. Este último grupo está formado por una gran cantidad de instituciones que tienen igual estrategia de citación, sobre todo las representadas con colores más claros y sin límites de separación. Se presupone que estas firmas se basan para su desarrollo en su propia base tecnológica, pues citan poco a otras instituciones. El cluster 2 evidencia un alto nivel de autocitación, cuando esto sucede, algunos investigadores en el tema señalan que probablemente esta institución tenga un nicho de protección cerrada sobre un espacio tecnológico. Puede estar ocurriendo que exista una patente importante, la cual se ha rodeado de invenciones mejoradas. El cluster formado por la *Merck & Co.* indica un mayor nivel de citación, esto presupone una estrategia balanceada: absorbe tecnología externa y produce tecnología propia. La cercanía de los cluster también es una evidencia sobre las instituciones que tienen estrategias parecidas a la de otras. La *Rockefeller University* hace frontera con el cluster que incluye a *North American Vaccine*, y el *National Res. Council de Canadá*; estas instituciones forman un colegio tecnológico invisible que basa sus desarrollos en la misma innovación tecnológica.

## CONCLUSIONES

Las redes autoorganizadas son redes neuronales que se organizan por sí mismas y están concebidas para clasificar conjuntos de datos para los que no se tiene ningún tipo de organización. De este modo, la salida ante un dato concreto de entrada no se contrasta con ninguna referencia. El principal objetivo de estas redes es el de deducir automáticamente la clasificación más natural de ciertos datos, a través de la aplicación de un aprendizaje no supervisado y competitivo.

En una red neuronal con aprendizaje competitivo los prototipos pueden acabar siendo representados de cualquier forma por las neuronas. En la red de Kohonen, o SOM como han sido llamadas en el presente documento, se supone que las neuronas de la red están relacionadas topológicamente, de manera tal que puede hablarse de  $n$  grado de vecindad entre ellas, y se pretende que prototipos semejantes vengan representados por neuronas cercanas, emulando de esta forma la actividad de razonamiento humano que se lleva a cabo en el cerebro.

El algoritmo de los SOM se basa en el aprendizaje competitivo, y es aplicado durante la etapa de entrenamiento de la red. Con este tipo de aprendizaje se proporciona un mapeo de preservación topológica a partir de un espacio multidimensional para mapear u organizar las neuronas, formando una rejilla de dos dimensiones. La propiedad de preservación topológica significa que en los SOM se agrupan vectores de datos de entrada similares en neuronas: los puntos que se encuentran cercanos unos de otros en el espacio de entrada son mapeados en el SOM, en unidades del mapa que son cercanas.



El efecto de la etapa de aprendizaje no es otro que acercar en una pequeña cantidad el vector de pesos de la neurona de mayor actividad al vector de entrada, es decir hacer los valores de los vectores lo mas similar posible. Para esto, el entrenamiento de una red SOM se hace mediante la aplicación de los siguientes pasos:

1. Inicialización de los pesos y el número de iteraciones.
2. Presentación del ejemplo.
3. Determinación de vecindad.
4. Determinación de la neurona ganadora.
5. Actualización de los pesos sinápticos.
6. Fin del aprendizaje, si se ha alcanzado el número máximo de iteraciones establecido.

En cuanto a la topología, los mapas autoorganizados consisten de dos capas de unidades de procesamiento: la primera es una capa de entrada que contiene unidades de procesamiento para cada elemento en el vector de entrada; la segunda es la capa de salida o rejilla de unidades de procesamiento, que esta completamente conectada con la capa de entrada, y el número de unidades de procesamiento que la conforma es determinado por el usuario, a partir de la forma y el tamaño inicial del mapa deseado. Además no existe ninguna capa ni unidad de procesamiento oculto.

Las redes SOM a pesar de que son muy robustas pueden ser eficientes en su implantación en los algoritmos computacionales, convirtiéndose en una de las RNA mas completas.

En cuanto a las aplicaciones, los mapas autoorganizados ofrecen una fácil visualización, imponen pocos presupuestos y restricciones, y son capaces de manipular grandes conjuntos de datos para detectar en estos estructuras y patrones aislados. Por ello, los mapas autoorganizados han cobrado

creciente interés, y sus principales aplicaciones se pueden resumir como sigue: visión y análisis de imágenes, aplicaciones ópticas, análisis y reconocimiento de voz, estudios acústicos y música, procesamiento de señales y sistemas de medida de radar, sistemas de medida, control de procesos, robótica, química, física, diseño de circuitos electrónicos, aplicaciones médicas sin procesamiento de imágenes, procesamiento de datos, problemas lingüísticos y de inteligencia artificial, problemas matemáticos, investigación neurofisiológica, entre otras. Las aplicaciones mas relevantes se dirigen hacia la recopilación de inmensas cantidades de documentos, permitiendo obtener resúmenes visuales en un gran cuadro, una pantalla de computador.

## BIBLIOGRAFÍA

GONZÁLEZ PENEDO, Manuel; “Sistemas Conexionistas”, tema 5: “Redes Autoorganizativas”.

CAMPOY, Pascual; “Redes Neuronales: Inteligencia por aprendizaje”.

ARANGUREN, Silvia; MUZACHIODI, Silvia; “Implicancias del Data Mining”, capítulo 7: “Redes Neuronales y Algoritmos Genéticos”.

GÓMEZ, David; VÉLEZ, Lorena; “Redes no supervisadas: Mapas Autoorganizados”.

PALMER, A., MONTAÑO, J.J. y JIMÉNEZ, R. “Tutorial sobre Redes Neuronales Artificiales: Los Mapas Autoorganizados de Kohonen”

BISHOP, C.M. (1995). Neural networks for pattern recognition. Oxford: Oxford University Press.

GUERRERO, VICENTE P. “Redes neuronales artificiales aplicadas a la visualización de la información”

MATICH, Damián; “Redes Neuronales: Conceptos Básicos y Aplicaciones”.

Haykin Simon. Neural Networks. A comprehensive foundation. Prentice Hall 2<sup>nd</sup> edition, 1999

Rumelhart David E, Mc Clelland James L., and the PDP research group. Parallel Distributed Processing, Explorations in the microstructure of cognition. MIT Press 1986

<http://electronica.com.mx/neural/teoriabasica> Tutorial sobre redes neuronales

<http://www.cis.hut.fi/%7Esami/thesis/node2.html> de Sami Kaski, marzo de 1997

<http://diwww.epfl.ch/mantra/tutorial/english/rbf/html/index.html> de Jesse W. Hong del Instituto de Tecnología de Massachusetts

<http://geneura.ugr.es/~jmerelo/tutoriales/bioinfo/Kohonen.html> Tutorial de mapas autoorganizativos de Kohonen de J.J Merelo

[Text mining with the WEBSOM](#) de Krista Lagus, 2000 (abstrac)

[http://mappa.mundi.net/maps/maps\\_009/](http://mappa.mundi.net/maps/maps_009/) de Martin Dodge,

<http://scitec.uwichill.edu.bb/cmp/online/p21h/Tutorial3/tut3.htm> de Janak Sodha , 1997 ultima modificación: enero 22, 2003

<http://koti.mbnet.fi/~phodju/nenet/SelfOrganizingMap/Theory.html> de Timo Honkela, 2 de junio 1998.

[http://www-etsi2.ugr.es/depar/ccia/rl/www/tema3\\_00-01\\_www/node23.html](http://www-etsi2.ugr.es/depar/ccia/rl/www/tema3_00-01_www/node23.html)  
"Técnicas supervisadas II: Aproximación no paramétrica" de **Francisco José Cortijo Bon,**

<http://www ldc.usb.ve/~mcastro/docs/CLASE8completa.pdf> del Ing. Migue Castro, grupo inteligencia artificial Universidad Simon Bolívar. Derechos

reservados ©1999 TREC Internet

<http://www.lfcia.org/~cipenedo/cursos/scx/Tema2/nodo2-1.html> "*Redes de Neuronas Artificiales. Un Enfoque Práctico*". Pedro Isasi Viñuela e Inés M. Galván León (eds). Prentice Hall. 2004

[http://www.tdcat.cesca.es/TESIS\\_UPC/AVAILABLE/TDX-0416102-075520/26ApendiceD.PDF](http://www.tdcat.cesca.es/TESIS_UPC/AVAILABLE/TDX-0416102-075520/26ApendiceD.PDF) apéndice D "Redes neuronales y teoría de los conjuntos difusos"

<http://www.gc.ssr.upm.es/inves/neural/ann2/anntutorial.html> Tutorial de redes neuronales del Grupo de investigación "Circuitos S.S.R" de la Universidad Politécnica de Madrid.

[http://www.doc.ic.ac.uk/%7End/surprise\\_96/journal/vol4/cs11/report.html](http://www.doc.ic.ac.uk/%7End/surprise_96/journal/vol4/cs11/report.html) de Christos Stergiou y Dimitrios Siganos

[http://es.wikipedia.org/wiki/Red\\_neuronal\\_artificial](http://es.wikipedia.org/wiki/Red_neuronal_artificial) ,enciclopedia

[http://www.iiia.csic.es/~mario/rna/tutorial/RNA\\_intro.html](http://www.iiia.csic.es/~mario/rna/tutorial/RNA_intro.html) un tutorial de Mario Gómez Martínez del Instituto de investigación de inteligencia artificial.

<http://www.hut.fi/Yksikot/Kirjasto/Diss/2000/isbn9512252600/isbn9512252600.pdf> Text Mining with the WEBSOM

<http://lib.hut.fi/Diss/2000/isbn9512252600/article1.pdf>  
Browsing digital libraries with the aid of self-organizing maps.

<http://websom.hut.fi/websom/doc/ps/honkela96tr.ps>

Newsgroup exploration with WEBSOM method and browsing interface

<http://www.cis.hut.fi/research/som-research/som.shtml>

SOM RESEARCH

<http://websom.hut.fi/websom/>

WEBSOM

<http://www.uwasa.fi/stes/step96/step96/lagus/>

WEBSOM STATUS REPORT