Los mapas auto-organizados de Kohonen (*SOM*)

Introducción

En 1982 T. Kohonen presentó un modelo de red denominado mapas auto-organizados o *SOM* (Self-Organizing Maps), basado en ciertas evidencias descubiertas a nivel cerebral. Este tipo de red posee un aprendizaje no supervisado competitivo.

No existe ningún maestro externo que indique si la red neuronal está operando correcta o incorrectamente porque no se dispone de ninguna salida objetivo hacia la cual la red neuronal deba tender.

La red auto-organizada debe descubrir 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 auto-organizarse en función de los estímulos (datos) procedentes del exterior.

En el aprendizaje competitivo las neuronas compiten unas con otras con el fin de llevar a cabo una tarea dada. 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 anuladas el resto, que son forzadas a sus valores de respuesta mínimos.

El objetivo de este aprendizaje es categorizar los datos que se introducen en la red. Se clasifican valores similares en 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.

Ideas intuitivas sobre el algoritmo del SOM

- El SOM es, en realidad, un tipo de algoritmo para clasificar observaciones.
- Se elige un gran número de clusters y se colocan en forma de una red bidimensional. La idea es que los representantes de cada grupo (o pesos, según la notación de Kohonen) estén correlacionados espacialmente, de modo que los puntos más próximos en la rejilla sean más parecidos entre sí que los que estén muy separados.
- Este proceso es conceptualmente similar al MDS que transforma observaciones similares en puntos cercanos del espacio bidimensional.
- Si se discretiza el espacio bidimensional dividiéndolo, por ejemplo, en una rejilla de componentes rectangulares se puede definir una aplicación desde el espacio de alta dimensiones original sobre dicho espacio bidimensional.
- Además, se puede tomar la media de los elementos que se encuentran en cada elemento de la rejilla para definir representantes de las clases de la rejilla. Los representantes que están en clases próximas se parecen entre sí. La idea básica del SOM es, así, proporcionar una versión discreta del MDS.
- Kohonen afirma: *I just wanted an algorithm that would effectively map similar patterns (pattern vectors close to each other in the input signal space) onto contiguous locations in the output space.* (Kohonen, 1995, p. VI.)

Fundamentos biológicos

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.

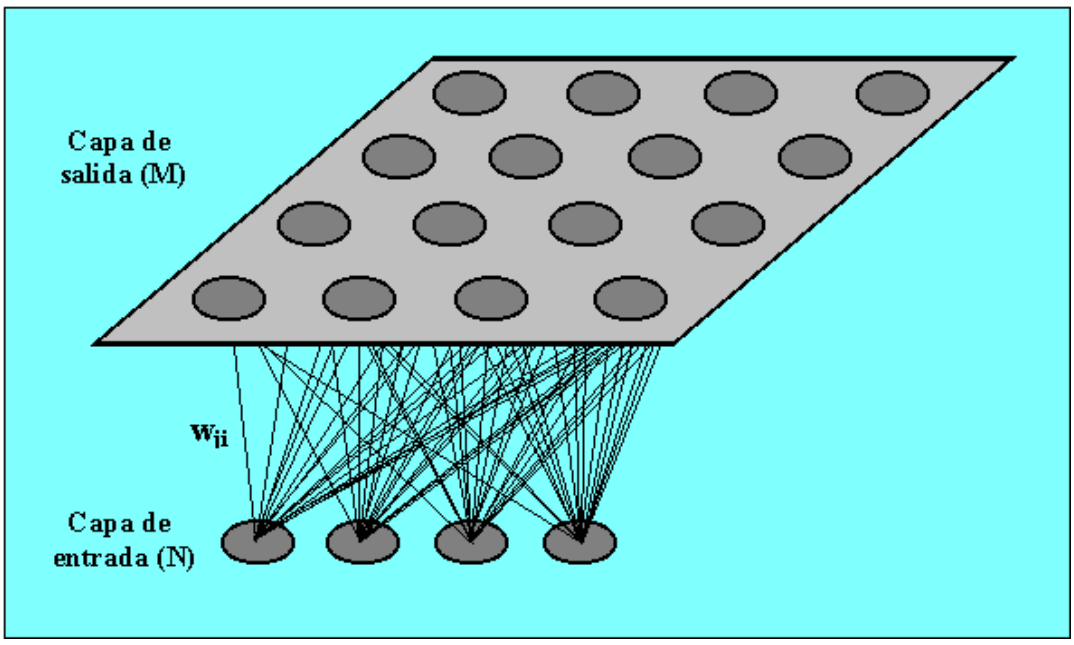
Aunque en gran medida esta organización neuronal está predeterminada genéticamente, es probable que parte de ella se origine mediante el aprendizaje. Esto sugiere, por tanto, que el cerebro podría poseer la capacidad inherente de formar mapas topológicos a partir de las informaciones recibidas del exterior.

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.

El modelo de red auto-organizado presentado por Kohonen pretende mimetizar de forma simplificada la capacidad del cerebro de formar mapas topológicos a partir de las señales recibidas del exterior.

Arquitectura del 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:



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. Así, el SOM define una proyección desde un espacio de datos en alta dimensión a un mapa bidimensional de neuronas.

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 , que produce la topología o estructura del mapa. Las topologías más frecuentes son la rectangular y la hexagonal.

Las neuronas adyacentes pertenecen a una vecindad N_j de la neurona j. La topología y el número de neuronas permanece fijo desde el principio. El número de neuronas determina la suavidad de la proyección, lo cual influye en el ajuste y capacidad de generalización del SOM.

Durante la fase de entrenamiento, el SOM forma una red elástica que se pliega dentro de la nube de datos originales. El algoritmo controla la red de modo que tiende a aproximar la densidad de los datos. Los vectores de referencia del codebook se acercan a las áreas donde la densidad de datos es alta.

Eventualmente unos pocos vectores el codebook están en áreas donde existe baja densidad de datos.

El algoritmo del SOM

El proceso de aprendizaje del SOM es el siguiente:

n += eta * exp(-d**2 / sigma**2) * (c - n) # kohonen.Map pythonwhere eta is the learning rate, sigma is called the neighborhood siz

Paso 1. Un vector x es seleccionado al azar del conjunto de datos y se calcula su distancia (similitud) a los vectores del codebook, usando, por ejemplo, la distancia euclídea:

$$\|x - m_c\| = \min_j \{\|x - m_j\|\}$$

Paso 2. Una vez que se ha encontrado el vector más próximo o *BMU* (best matching unit) el resto de vectores del codebook es actualizado. El *BMU* y sus vecinos (en sentido topológico) se mueven cerca del vector x en el espacio de datos. La magnitud de dicha atracción está regida por *la tasa de aprendizaje*.

Mientras se va produciendo el proceso de actualización y nuevos vectores se asignan al mapa, la tasa de aprendizaje decrece gradualmente hacia cero. Junto con ella también decrece el radio de vecindad ~~también~~.

La regla de actualización para el vector de referencia dado i es la siguiente:

(1-\alpha(t))m_j(t) + \alpha(t) x(t)

$$m_j(t+1) = \begin{cases} m_j(t) + \alpha(t)(x(t) - m_j(t)) & j \in N_c(t) \\ m_j(t) & j \notin N_c(t) \end{cases}$$

Los pasos 1 y 2 se van repitiendo hasta que el entrenamiento termina. El número de pasos de entrenamiento se debe fijar ~~antes~~ a priori, para calcular la tasa de convergencia de la función de vecindad y de la tasa de aprendizaje.

Una vez terminado el entrenamiento, el mapa ha de ordenarse en sentido topológico: n vectores topológicamente próximos se aplican en n neuronas adyacentes o incluso en la misma neurona.

Medidas de calidad del mapa y precisión del mapa

Una vez que se ha entrenado el mapa, es importante saber si se ha adaptado adecuadamente a los datos de entrenamiento. Como medidas de calidad de los mapas se considera la precisión de la proyección y la preservación de la topología.

La medida de precisión de la proyección describe cómo se adaptan o *responden* las neuronas a los datos. Habitualmente, el numero de datos es mayor que el número de neuronas y el error de precisión es siempre diferente de 0.

Para calcular la precisión de la proyección se usa el error medio de cuantificación sobre el conjunto completo de datos:

$$\varepsilon_q = \frac{1}{N} \sum_{i=1}^{N} \|x_i - m_c\|$$

La medida de preservación de la topología describe la manera en la que el SOM preserva la topología del conjunto de datos. Esta medida considera la estructura del mapa. En un mapa que esté *retorcido* de manera extraña, el error topográfico es grande incluso si el error de precisión es pequeño.

Una manera simple de calcular el error topográfico es:

$$\varepsilon_t = \frac{1}{N} \sum_{k=1}^{N} u\left(x_k\right)$$

donde $u\left(x_k\right)$ es igual a 1 si el primer y segundo BMUs de x_k no están próximos el uno al otro. De otro modo, $u\left(x_k\right)$ es igual a 0.

Visualización del SOM

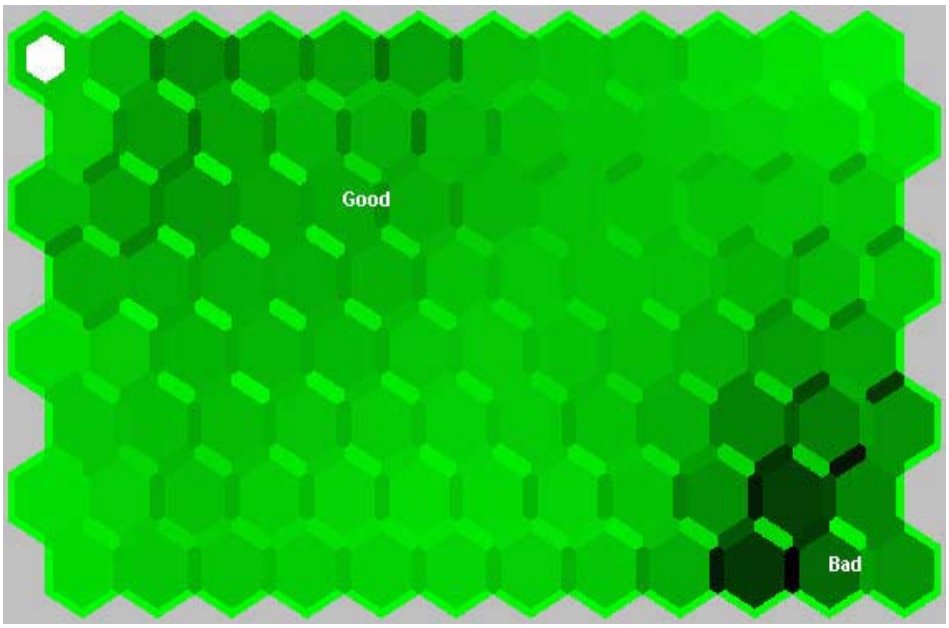
El SOM es fácil de visualizar y, además, en los últimos años se han desarrollado diferentes técnicas de visualización, tanto para los vectores de referencia como para los histogramas de datos.

La proyección de Sammon representa el SOM de la manera gráfica que se muestra en la figura siguiente. Esta trata de encontrar una proyección no lineal óptima para los datos en alta dimensión, de manera que los vectores que se proyectan en la superficie bidimensional, conservan la misma distancia euclídea relativa entre ellos que la que tenían en alta dimensión.

La matriz unificada de distancias, o matriz U, es el método más popular para mostrar el SOM. Representa el mapa como una rejilla regular de neuronas, el tamaño y topología del mapa se puede observar en el gráfico donde cada elemento representa una neurona.

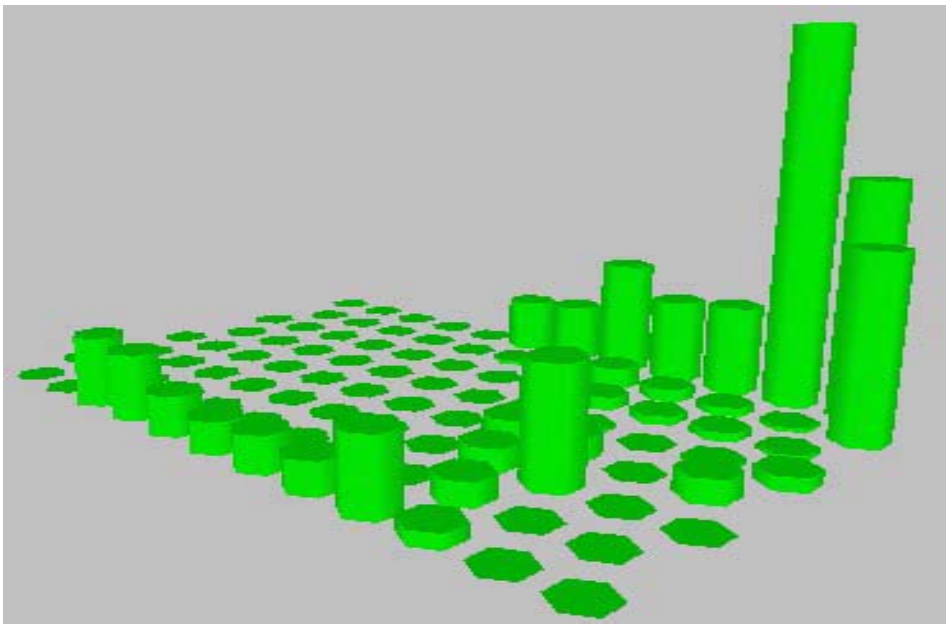
Cuando se genera la matriz U se calcula, a su vez, una matriz de distancias entre los vectores de referencia de neuronas adyacentes en el mapa bidimensional. Después se

selecciona algún tipo de representación gráfica, por ejemplo una escala de grises. Los colores en la figura se seleccionan de modo que cuanto más oscuro es el color entre dos neuronas, menor es la distancia entre ellas.



Visualización de histogramas de datos

Se trata de mostrar cómo los vectores de datos son clasificados por el SOM. El histograma de datos muestran cuántos vectores pertenecen a un *cluster* definido por cada neurona. El histograma se genera usando un SOM entrenado y el conjunto de datos.



Ejemplo: Clasificación de la planta del Iris

Como ejemplo ilustrativo se puede utilizar los datos de Fisher, sobre el problema de clasificación de las plantas del género Iris.

Se tiene una muestra de 150 plantas donde cada ejemplar posee cuatro características y la tarea consiste en determinar el tipo de planta Iris en base a esas características. Las caracterísitcas son: longitud del sépalo, ancho del sépalo, longitud del pétalo y ancho del pétalo. Hay una 50 ejemplares para cada tipo de planta del género Iris: setosa, versicolor y virginica.

Estos datos se encuentran en la librería `datasets` de `R`, y tienen como nombre *iris*:

```
library(datasets)
help(iris)
```

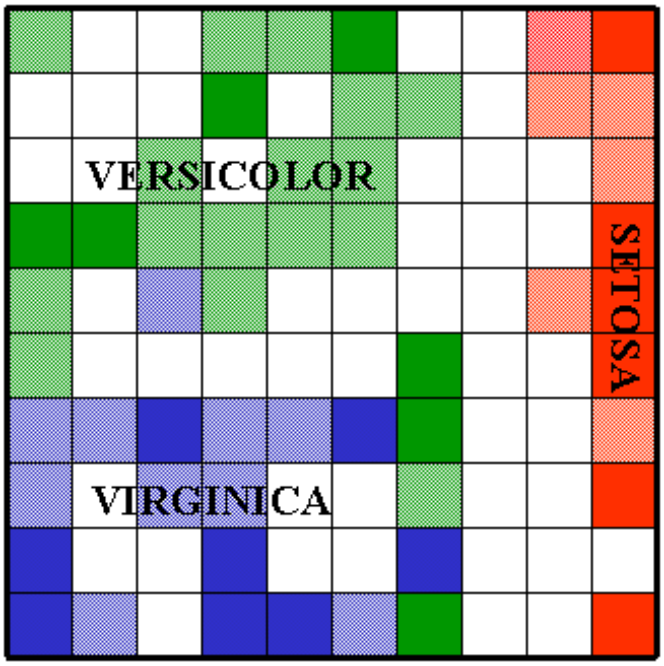
A partir de esta matriz de datos, se trata de comprobar si un modelo SOM es capaz de agrupar en el mapa los tres tipos de planta, proporcionándole únicamente los datos sobre las cuatro características citadas. Por tanto, las categorías deberían ser creadas de

forma no supervisada por la propia red a través de las correlaciones descubiertas entre los datos de entrada, puesto que no se proporciona la categoría de pertenencia del ejemplar.

Se utiliza como criterio de similitud, la distancia euclídea en la etapa de entrenamiento. Como salida de la red se determina un mapa bidimensional 10×10; por tanto, el mapa está compuesto por 100 neuronas de salida.

Se entrena un total de 10 modelos, cada uno con una configuración inicial de pesos diferente (con rango comprendido entre 0 y 1), pero siguiendo todos el mismo esquema de aprendizaje. Así, el entrenamiento de cada mapa se organiza en dos fases. En la primera fase, cuyo objetivo consiste en organizar el mapa, se utiliza una tasa de aprendizaje alta igual a 1 y un radio de vecindad grande igual al diámetro del mapa, es decir, igual a 10. A medida que avanza el aprendizaje, tanto la tasa de aprendizaje como el radio de vecindad se van reduciendo de forma lineal hasta alcanzar unos valores mínimos, 0,05 y 1, respectivamente. En la segunda fase, cuyo objetivo es el ajuste fino del mapa, se utiliza una tasa de aprendizaje pequeña y constante igual a 0,05 y un radio de vecindad constante y mínimo igual a 1.

Una vez entrenados los 10 modelos, se calcula para cada uno de ellos, el error cuantificador promedio y se selecciona el modelo cuyo error sea más pequeño. A continuación, se muestra el mapa del modelo finalmente seleccionado:



En la figura, las neuronas de salida que representan ejemplares de la variedad versicolor aparecen de color verde, las neuronas que representan ejemplares de la variedad virginica aparecen de color azul y, finalmente, las neuronas que representan ejemplares de de la variedad setosa aparecen de color rojo. Las neuronas de color blanco no representan patrón alguno. Con el objeto de poder analizar la distribución de frecuencias de cada clase de planta en el mapa, cada color puede aparecer con dos posibles intensidades según la frecuencia de patrones asociados a una determinada neurona de salida. Así, cuando el color es poco intenso, hay pocos patrones (1 ó 2) asociados a la neurona; cuando el color es intenso, hay muchos patrones (de 3 a 10) asociados a la neurona.

Se puede observar que los ejemplares de la variedad setosa han sido perfectamente discriminados respecto a las otras dos categorías, quedando agrupados todos en la parte derecha del mapa. Por su parte, los ejemplares de versicolor y virginica se reparten la parte superior e inferior izquierda del mapa, respectivamente; compartiendo zonas adyacentes. Esto parece indicar que la variedad setosa se diferencia perfectamente de los otros dos tipos, mientras que las variedades versicolor y virginica mantienen características más similares, aunque bien diferenciables.

A continuación, podríamos pasar a la etapa de funcionamiento de la red donde se presentarían nuevos ejemplares y, mediante el proceso de competición, podríamos observar en qué zona del mapa está ubicada la neurona de salida vencedora asociada a cada nuevo ejemplar

Tutoriales:

`http://www.bibliopsiquis.com/psicologiacom/vol6num1/3301/`

`http://www.ai-junkie.com/ann/som/som1.html`

Programación

En `R` existen unos comandos básicos, en la librería `class`, que producen unas salidas gráficas más bien pobres. Mejora bastante la librería `kohonen`.

En Tanagra se tiene, en el menú *Clustering*, un procedimiento llamado *Kohonen-SOM* pero es bastante pobre. Su utilización es sencilla: basta seguir los pasos básicos que se seguían en el tema de redes neuronales, usando Tanagra.

Ejemplo: Cangrejos

Esta base de datos contiene 200 observaciones y 8 variables. Cada observación es un cangrejo de sobre el que se mide 5 medidas morfológicas. Los cangrejos son de dos posibles sexos y dos especies:

masculino Blue	B
femenino blue	b
masculino naranja	O
femenino naranja	o

En `R` de manera básica se se puede hacer:

```
library(class)
# Cargo los datos
data(crabs, package = "MASS")
# Se toman logaritmos
lcrabs <- log(crabs[, 4:8])
```

```
# Cuatro tipos de Cangrejos (2 colores X 2 Sexos)
crabs.grp <- factor(c("B", "b", "O", "o")[rep(1:4, rep(50,4))])
# Tipo de Zona de vecinidad
gr <- somgrid(topo = "hexagonal")
# Calculo la red som
crabs.som <- batchSOM(lcrabs, gr, c(4, 4, 2, 2, 1, 1, 1, 0, 0))
plot(crabs.som)

bins <- as.numeric(knn1(crabs.som$code, lcrabs, 0:47))
# Representacion de la red
plot(crabs.som$grid)
symbols(crabs.som$grid$pts[, 1], crabs.som$grid$pts[, 2],
circles = rep(0.4, 48), inches = FALSE, add = TRUE)
text(crabs.som$grid$pts[bins, ] + rnorm(400, 0, 0.1),
as.character(crabs.grp))
```

Con la librería kohonen se obtiene una salida más completa.

Ejemplo: Vinos italianos

Se estudia el conjunto de datos `wine` contiene observaciones sobre 177 vinos italianos procedentes de tres cosecheros y 13 variables, como concentraciones de alcohol y *flavonoides* entre otras.

Se usa la librería kohonen.

```
library(kohonen)
data(wines)
set.seed(7)
wines.sc <- scale(wines)
```

```
wine.som <- som(data = wines.sc, grid = somgrid(5, 4, "hexagonal"))
names(wine.som)
summary(wine.som)
wine.som$unit.classif
wine.som$codes
plot(wine.som, main = "Datos de Vinos")
```

En el llamado SOM supervisado existe también hay una variable dependiente para comparar los resultados (categórica o continua)

```
library(kohonen)
data(wines)
set.seed(7)
kohmap <- xyf(scale(wines), classvec2classmat(wine.classes),
grid = somgrid(5, 5, "hexagonal"))

par(mfrow=c(2,2))
plot(kohmap, type="codes", main=c("Distribuci\'{o}n de variables",
"Clases de C\'{o}digos"))
plot(kohmap, type="counts")
plot(kohmap, type="mapping",labels=wine.classes, col=wine.classes+1,
main="Mapa de clases")
```