Sistemas Autoorganizados II

Modelos de la computación (Sistemas Autoorganizados)

Francisco Fernández Navarro

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

Índice de contenidos

¿Qué son las Redes de Kohonen?

- Las Redes de Kohonen, o Mapas Autoorganizados de Kohonen (SOM, por sus siglas en inglés), son un tipo de red neuronal artificial.
- Fueron desarrolladas por Teuvo Kohonen en la década de 1980.
- Se utilizan para la clasificación y visualización de datos de alta dimensión.

Características de las Redes de Kohonen

- Aprendizaje no supervisado: las SOM no requieren etiquetas en los datos de entrenamiento.
- Topología bidimensional: las neuronas se organizan en una cuadrícula 2D.
- Clustering y reducción de dimensionalidad: las SOM pueden agrupar datos similares y visualizar relaciones entre ellos.
- Aplicaciones en reconocimiento de patrones, minería de datos, y más.

Funcionamiento de las Redes de Kohonen

- Cada neurona en la red representa un punto en el espacio de entrada.
- Durante el entrenamiento, las neuronas se ajustan para representar regiones de datos similares.
- Las neuronas ganadoras se activan en respuesta a patrones de entrada.
- Las SOM utilizan vecindades para propagar ajustes entre neuronas.
- Modelo sencillo para la formación autoorganizada de mapas de características de los datos de entrada.

Diferencia con respecto a las redes no supervisadas competitivas

- En una red neuronal competitiva, los prototipos representan directamente a cada clase que conforma el conjunto de entrada.
- En un mapa de Kohonen, las neuronas están relacionadas topológicamente, de manera que si consideramos una neurona cualquiera, sus vecinas representarán prototipos similares.

Índice de contenidos

Arquitectura

- La arquitectura es la misma de una red competitiva, es decir, una única capa donde cada neurona tendrá tantos sensores como dimensión tengan los patrones.
  - No existen pesos sinápticos entre diferentes neuronas.
  - La topología y la función de vecindad determinarán cómo se relacionan entre sí las diferentes neuronas de la red.

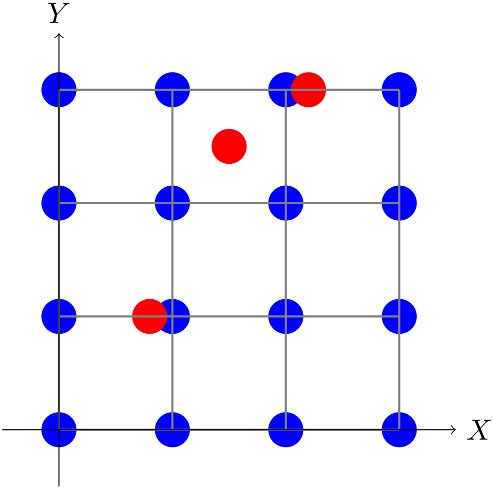
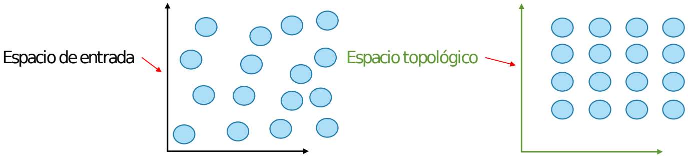


Figure: Red de Kohonen con Neuronas y Patrones de Entrada.

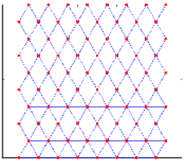
Arquitectura

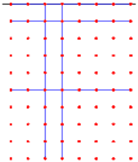
- La posición en el **espacio de entrada** de una neurona está determinada por sus **pesos sinápticos**, al igual que en la red competitiva, **varía conforme la red aprende.**
- La posición de cada neurona en el **espacio topológico** determina la topología de la red. **No evoluciona**, es constante a lo largo del tiempo.

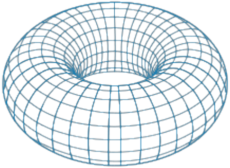


Arquitectura

- Normalmente una SOM está dispuesta siguiendo una topología de malla o rejilla. La topología, por medio de la función de vecindad, determina el grado de vecindad entre neuronas, pero NO sus conexiones sinápticas.
  - Las topologías más habituales son las rectangulares y las hexagonales.
  - También es habitual encontrar topologías sin bordes, como por ejemplo aquella que forma un toroide.







Índice de contenidos

Vecindad

- Para establecer una noción de la proximidad entre unidades de proceso, vamos a definir una función de distancia $d(\mathbf{p}_i, \mathbf{p}_j) = ||\mathbf{p}_i - \mathbf{p}_j||$, que nos da la distancia topológica entre la unidad $\mathbf{p}_i$ y la $\mathbf{p}_j$. Es posible usar diferentes funciones de distancia:
  - Distancia euclídea: $d(\mathbf{p}_i, \mathbf{p}_j) = \sqrt{(p_{i1} - p_{j1})^2 + (p_{i2} - p_{j2})^2}$.
  - Distancia rectangular: $d(\mathbf{p}_i, \mathbf{p}_j) = |p_{i1} - p_{j1}| + |p_{i2} - p_{j2}|$.
- La función de vecindad (o función ventana) tendrá valores más altos conforme más cercanas sean la unidades de proceso, es decir, es una función decreciente respecto de la distancia entre neuronas.

Vecindad

- Un ejemplo de función de vecindad es: $\Lambda(\mathbf{p}_i, \mathbf{p}_j) = e^{-d(\mathbf{p}_i, \mathbf{p}_j)}$.
- Es habitual reducir el radio de vecindad conforme avanzan las iteraciones.

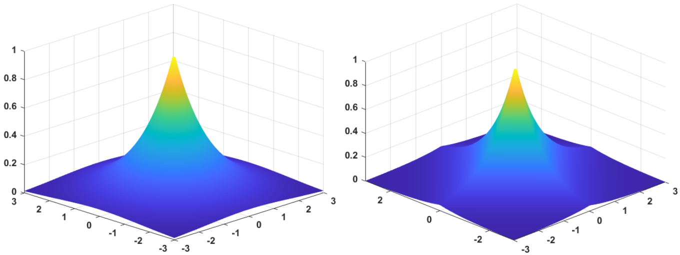


Figure: Representación vecindad para distancia euclidea y rectangular

Ejemplos de función de vecindad mediante ventana o plantilla

Ejemplo Vecindad 1:

$$\begin{bmatrix} 0 & 0.5 & 0 \\ 0.5 & 1 & 0.5 \\ 0 & 0.5 & 0 \end{bmatrix}$$

Ejemplo Vecindad 2:

$$\begin{bmatrix} 0.25 & 0.5 & 0.25 \\ 0.5 & 1 & 0.5 \\ 0.25 & 0.5 & 0.25 \end{bmatrix}$$

Índice de contenidos

Dinámica de computación

- La dinámica de la computación es análoga a la de la red competitiva $\rightarrow$ Existe una única neurona ganadora para cada patrón de entrada, que es la que más cerca está de dicho patrón.
- La neurona ganadora, denotada como r, será aquello que tenga 1 como salida estimatada:

$$\hat{y}_r(\mathbf{x}_n) = \begin{cases} 1, & \text{si } h_r(\mathbf{x}_n) = \max(h_1(\mathbf{x}_n), h_2(\mathbf{x}_n), \ldots, h_{N_1 \times N_2}(\mathbf{x}_n)) \\ 0, & \text{en otro caso} \end{cases}$$

  donde N_1 es el número de neuronas consideradas por filas, N_2 número de neuronas por columnas, y:

$$h_r(\mathbf{x}_n) = w_{1r}x_{n1} + w_{2r}x_{n2} + \ldots + w_{Kr}x_{nK} - \theta_r,$$

  con $\theta_r = \frac{1}{2}(w_{1r}^2 + w_{2r}^2 + \ldots + w_{Kr}^2)$.
- El número de neuronas del modelo lo determina el tamaño de la rejilla $N_1 \times N_2$.

Dinámica de computación

- La posición de las neuronas en el espacio de entrada evoluciona conforme aprende la red.
  - La neurona ganadora r se adapta al patrón de entrada de forma similar a la red competitiva, pero en este caso, los vecinos de la ganadora también se adaptan, pero en menor medida.

$$\mathbf{w}_j(i,n) = \mathbf{w}_j(i,n-1) + \eta(i,n)\Lambda(\mathbf{p}_j,\mathbf{p}_r)(\mathbf{x}_n - \mathbf{w}_j(i,n-1)) \quad (1)$$

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

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

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

Dinámica de computación

A tener en cuenta:

Si la función de vecindad viene definida por una matriz del estilo de la siguiente:

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

entonces la red de kohonen es equivalente a la red competitiva no supervisada estudiada en la sesión anterior.

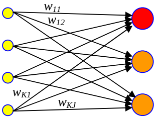
Flujo algorítmico

Red Kohonen (en línea) ($\mathbf{X}$, $\eta_0 > 0, N_1, N_2$):

for $j = 1$ until $N1 \times N_2$ **do**
    $\mathbf{w}_j(1,0) \leftarrow 2 \times \text{rand}(K,1) - 1, 0 \leq \text{rand}() < 1.$
end for
for $i = 1$ until I **do**
    for $n = 1$ until N **do**
        $\eta(i,n) \leftarrow \eta_0 \left(1 - \frac{((i-1)\cdot N) + n}{N \cdot I}\right)$
        $\hat{y}_r(\mathbf{x}_n) \leftarrow \begin{cases} 1, & \text{si } h_r(\mathbf{x}_n) = \max(h_1(\mathbf{x}_n), h_2(\mathbf{x}_n), \ldots, h_{N_1 \times N_2}(\mathbf{x}_n)) \\ 0, & \text{en otro caso} \end{cases}$
        for $j = 1$ until $N_1 \times N_2$ **do**
            $\mathbf{w}_j(i,n) \leftarrow \mathbf{w}_j(i,n-1) + \eta(i,n)\Lambda(\mathbf{p}_j, \mathbf{p}_r)(\mathbf{x}_n - \mathbf{w}_j(i,n-1)).$
        end for
    end for
end for

Índice de contenidos

Arquitectura y dinámica



- Idéntico al modelo competitivo no supervisado.
- Potencial sináptico:

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

con $\theta_j = \dfrac{1}{2}(w_{1j}^2 + w_{2j}^2 + \ldots + w_{Kj}^2)$.

- Dinámica de la computación $\rightarrow$ Para cada patrón sólo hay una neurona ganadora:

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

Índice de contenidos

Regla de modificación de parámetros

- Para cada patrón conocemos siempre la clase a la que realmente pertenece.
- Regla de aprendizaje: Los pesos se modifican de la siguiente manera:
  - Si acierta, se sigue el sentido opuesto al gradiente (igual que el aprendizaje no supervisado).
  - Si falla, se sigue el mismo sentido del gradiente.
- Si en la iteración i con el patrón n, r es la neurona ganadora, la actualización de pesos es:

$$\mathbf{w}_r(i,n) = \begin{cases} \mathbf{w}_r(i,n-1) + \eta(i,n)(\mathbf{x}_n - \mathbf{w}_r(i,n-1)), & \text{si } \mathbf{x}_n \in \mathcal{C}_r \\ \mathbf{w}_r(i,n-1) - \eta(i,n)(\mathbf{x}_n - \mathbf{w}_r(i,n-1)), & \text{si } \mathbf{x}_n \notin \mathcal{C}_r \end{cases}$$

- Las que no ganan no modifican su vector de pesos: $\mathbf{w}_j(i,n) = \mathbf{w}_j(i,n-1), \forall j \neq r$.

Índice de contenidos

Tasa de aprendizaje

- Una tasa de aprendizaje dinámica en una red de Kohonen significa que:
  - Cuando la red acierta, los cambios en los pesos son suaves y pequeños, estabilizando el aprendizaje.
  - Si la red comete un error, la tasa de aprendizaje aumenta para permitir cambios más significativos en los pesos, lo que ayuda a corregir el error de manera más rápida y efectiva.
- La expresión sería:

$$\eta(i,n) = \begin{cases} \frac{\eta(i,n-1)}{1+\eta(i,n-1)}, & \text{si } \mathbf{x}_n \in \mathcal{C}_r \\ \frac{\eta(i,n-1)}{1-\eta(i,n-1)}, & \text{si } \mathbf{x}_n \notin \mathcal{C}_r \end{cases}$$

Tasa de aprendizaje

- La modificación de la tasa de aprendizaje siguiendo la ecuación anterior puede llevar a valores absurdos.
  - Imaginemos que se equivoca varias veces consecutivas partiendo de una tasa inicial de 0.1:

$$0.1 \to 0.111 \to 0.125 \to 0.143 \to \ldots \to 0.333 \to 0.50 \to 1 \to \infty$$

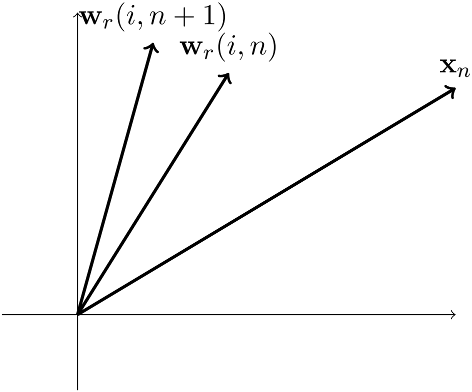
- En general es mejor utilizar la siguiente regla de modificación de la tasa de aprendizaje:

$$\eta(i,n) = \begin{cases} \frac{\eta(i,n-1)}{1+\eta(i,n-1)}, & \text{si } \mathbf{x}_n \in \mathcal{C}_r \\ \min\left\{\frac{\eta(i,n-1)}{1-\eta(i,n-1)}, \eta_{\max}\right\}, & \text{si } \mathbf{x}_n \notin \mathcal{C}_r \end{cases}$$

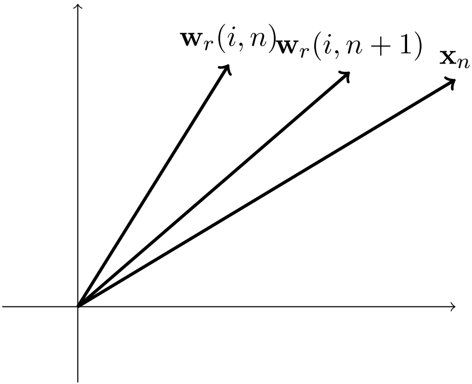
donde $\eta_{\max}$ es el valor máximo a considerar para la tasa de aprendizaje.

Índice de contenidos

Interpretación de la regla de aprendizaje



Fallo en la clasificación, $\mathbf{x}_n \notin \mathcal{C}_r$.



Acierto en la clasificación, $\mathbf{x}_n \in \mathcal{C}_r$.

En redes competitivas supervisadas, cuando la neurona ganadora coincide con la clase del patrón, el vector de pesos se acerca al patrón; de lo contrario, se aleja del patrón.

Índice de contenidos

Flujo algorítmico

Red competitiva supervisada (en línea) $(\mathbf{X}, \eta_0 > 0)$:

for $j = 1$ until J **do**
  $\mathbf{w}_j(1,0) \leftarrow 2 \times \text{rand}(K,1) - 1, 0 \leq \text{rand}() < 1.$
end for
for $i = 1$ until I **do**
  for $n = 1$ until N **do**
    $\hat{y}_r(\mathbf{x}_n) \leftarrow \begin{cases} 1, & \text{si } h_r(\mathbf{x}_n) = \max(h_1(\mathbf{x}_n), h_2(\mathbf{x}_n), \ldots, h_J(\mathbf{x}_n)) \\ 0, & \text{en otro caso} \end{cases}$
    $\mathbf{w}_r(i,n) \leftarrow$
    $\begin{cases} \mathbf{w}_r(i,n-1) + \eta(i,n-1)(\mathbf{x}_n - \mathbf{w}_r(i,n-1)), & \text{si } \mathbf{x}_n \in \mathcal{C}_r \\ \mathbf{w}_r(i,n-1) - \eta(i,n-1)(\mathbf{x}_n - \mathbf{w}_r(i,n-1)), & \text{si } \mathbf{x}_n \notin \mathcal{C}_r \end{cases}$
    $\eta(i,n) \leftarrow \begin{cases} \frac{\eta(i,n-1)}{1+\eta(i,n-1)}, & \text{si } \mathbf{x}_n \in \mathcal{C}_r \\ \min\left\{\frac{\eta(i,n-1)}{1-\eta(i,n-1)}, \eta_{\max}\right\}, & \text{si } \mathbf{x}_n \notin \mathcal{C}_r \end{cases}$
  end for
end for

En el código suponemos que tenemos un prototipo por clase, pero podríamos tener más.

¡Gracias por vuestra atención!