MAPAS AUTOORGANIZADOS DE KOHONEN

PROBLEMAS NO LINEALES

- **ENTRADAS (3) (PROBLEMA) Problema**
- **PATRONES: (3) (PROBLEMAS)** problema REGISTROS QUE SE TIENEN PARA SOLUCIONAR EL PROBLEMA – **ENTRENAMIENTO (70% Y 85%) PRUEBAS SIMULACION 30% Y EL 15% (GENERALIZAR)**

PREPROCESAMIENTO Y PROCESAMIENTO DE LOS DATOS DE ENTRADA

- ADECUACION DE LOS DATOS DE ENTRADAS (VECTORES DE DATOS), FILTROS Y OTRA SERIE DE PROCESOS A LOS DATOS DE ENTRADA
- NORMALIZACION DE LOS DATOS DE ENTRADA

CONFIGURACION DE LA RED KOHONEN

- **NUMERO DE NEURONAS EN LA CAPA DE PROCESAMIENTO (CUALQUIER NUMERO MENOS LOS NUMEROS PRIMOS) Y DEBE SER COMO MINIMO EL DOBLE DE LA CANTIDAD DE ENTRADAS (4) USUARIO**
- TIPO DE COMPETENCIA (USUARIO)
    1. COMPETENCIA BLANDA
       En este tipo de competencia actualizan los pesos sinápticos o se activan la neurona vencedora y sus vecinas
    2. COMPETENCIA DURA
       En este tipo de competencia solo actualiza los pesos sinápticos o se activa la neurona vencedora
- COEFICIENTE DE VECINDAD (0.2) USUARIO

1. (0..1] (0<CV<=1) ES EL VALOR PROMEDIO DE LAS DISTANCIAS ENTRE LAS NEURONAS QUE SE ENCUENTRAN EN EL PLANO MULTIDIMENSIONAL

- RATA DE APRENDIZAJE DINAMICA -> INICIA EN (1) PARA LA PRIMERA ITERACION, PERO A MEDIDA QUE SE VA ENTRENANDO LA RED, LA RATA VA CAMBIANDO ITERACION X ITERACION (RA=1/IT)
- INICIALIZACION DE LOS PESOS SINAPTICOS -> [-1..1]
- LA MATRIZ DE PESOS SE DIMENSIONA TENIENDO EN CUENTA EL NUMERO DE ENTRADAS Y EL NUMERO DE NEURONAS CONFIGURADAS. EL TAMAÑO DE LA MATRIZ DE PESOS ES **W[ENTRADAS][NEURONAS]**
- ALGOTIRMO DE ENTRENAMIENTO ->KOHONEN
  1. Wnuevo[j,i]=Wactual[j,i]+RA*(D neurona vencedora)
- NUMERO DE ITERACIONES (100) USUARIO
- **********
- **ENTRENAMIENTO**
-

X1	X2	X3
0.3	0.6	0.9
0.9	0.8	0.2
0.8	0.9	0.9

- **MATRIZ DE PESOS W[ENTRADAS][NEURONAS] [3][4]**
- **CONV. W[M][N]**
- **j=1 HASTA M (M) NUMERO DE ENTRADAS 3**
- **i=1 HASTA N (N) NUMERO DE NEURONAS 4**

INICIALIZAMOS LA MATRIZ DE PESOS SINAPTICOS

0.3	0.3	0.4	0.4
0.7	0.3	0.2	0.8
0.5	0.1	0.4	1

N1	N2	N3	N4

- **PRESENTAMOS EL PRIMER PATRON DE ENTRADA**

• X1	• X2	• X3
• 0.3	• 0.6	• 0.9

- **CALCULAR LA DISTANCIA EUCLIDIANA EXISTENTE ENTRE EL PATRON DE ENTRADA Y CADA UNA DE LAS NEURONAS QUE SE ENCUENTRAN EN EL PLANO**
- D[i]=∑((X[j]-W[J][i])^2)^1/2 D[i]=$\sqrt{\sum(Xj - Wji)^2}$
- i=1 HASTA NUMERO DE NEURONAS (N)
- $Di = \sqrt{\sum \ (\mathrm{X[j]} - \mathrm{W[J][i]})^2}$
- D1=raíz cuadrada ((X1-W1,1)^2+(X2-W2,1) ^2+(X3-W3,1) ^2)
- D2=raíz cuadrada ((X1-W1,2)^2+(X2-W2,2) ^2+(X3-W3,2) ^2)
- D3=raíz cuadrada ((X1-W1,3)^2+(X2-W2,3) ^2+(X3-W3,3) ^2)
- D4=raíz cuadrada ((X1-W1,4)^2+(X2-W2,4) ^2+(X3-W3,4) ^2)

j=1,2,3 i=1,2,3,4

$\mathrm{D}_1 = \sqrt{(0.3-0.3)^2 + (0.6-0.6)^2 + (0.9-0.9)^2}$

D1= 0

$D_2 = \sqrt{(0.3-0.3)^2 + (0.6-0.3)^2 + (0.9-0.1)^2}$

D2= 0.8544

$D_3 = \sqrt{(0.3-0.4)^2 + (0.6-0.2)^2 + (0.9-0.4)^2}$

D3= 0.648

$D4 = \sqrt{(0.3-0.4)^2 + (0.6-0.8)^2 + (0.9-1)^2}$

D4= 0.2449

- BUSCAR LA DISTANCIA DE LA NEURONA MAS CERCANA AL VECTOR DE ENTRADA (MENOR DISTANCIA)
- LA NEURONA VENCEDORA ES LA 4 DV ->(0.2449)

- SI SELECCIONARON COMPETENCIA BLANDA TENEMOS QUE CONSEGUIR LAS NEURONAS VECINAS A LA VENCEDORA (UTILIZANDO EL COEFICIENTE DE VECINDAD), PERO SI SELECCIONAN COMPETENCIA DURA SOLO SE ACTUALIZAN LOS PESOS DE LA NEURONA VENCEDORA
- SI SELECCIONAMOS LA COMPETENCIA BLANDA TENEMOS QUE CONSEGUIR LAS VECINAS A LA NEURONA VENCEDORA
- DT=Dv+COEFICIENTE DE VECINDAD
- DT=0.2449+0.2 -> 0.4449
- TODAS LAS DISTANCIAS MENORES A ESTE VALOR (DV -> 0.4449) SON VECINAS DE LA VENCEDORA
- D1 (0.4123)< 0.4449 SI ES VECINA
- D2 (0.8544)< 0.4449 NO NO ES VECINA
- D3 (0.648)< 0.4449 NO NO ES VECINA
- SOLO SE ACTIVA LA VENCEDORA Y SUS VECINA (LAS QUE CUMPLEN CON LA CONDICION ESTABLECIDA)
- **Wnuevo[j,i]=Wactual[j,i]+RA*(D neurona vencedora)**
  Actualizando los pesos de la neurona vecina Neurona 1
- W[1,1]= 0.3+1*(**0.2449**) -> 0.5499
- W[2,1]= 0.7+1*(**0.2449**) -> 0.9449
- W[3,1]= 0.5+1*(**0.2449**)-> 0.7449
- Actualizando los pesos de la neurona vencedora Neurona 4
- W[1,4]= 0.4+1*(**0.2449**) -> 0.6449
- W[2,4]= 0.8+1*(**0.2449**) -> 1.0449
- W[3,4]= 1+1*(**0.2449**)-> 1.2449

0.5499	0.3	0.4	**0.6449**
0.9449	0.3	0.2	**1.0449**
0.7449	0.1	0.4	**1.2449**
N1	**N2**	**N3**	**N4**

- PRESENTAMOS EL SIGUIENTE PATRON

• X1	• X2	• X3
• 0.9	• 0.8	• 0.2

- **CALCULAR LA DISTANCIA EUCLIDIANA EXISTENTE ENTRE EL PATRON DE ENTRADA Y CADA UNA DE LAS NEURONAS QUE SE ENCUENTRAN EN EL PLANO**
- D[i]=∑((X[j]-W[J][i])^2)^1/2
- i=1 HASTA NUMERO DE NEURONAS (N)
- $D\_l = \sqrt{\sum \ (\mathrm{X[j]} - \mathrm{W[j][i]})^2}$
- $D_1 = \sqrt{\left(0.9 - \mathbf{0.5499}\right)^2 + \left(0.8 - \mathbf{0.9449}\right)^2 + \left(0.2 - \mathbf{0.7449}\right)^2}$

  D1=

- CUANDO SE TERMINEN LA PRESENTACION DE TODOS LOS PATRONES DE ENTRADA SE ESTABLECE QUE SE REALIZO UNA ITERACION
- SE ACTUALIZA LA RATA DE APRENDIZAJE **RA=1/ITERACION**
- CONDICIONES DE PARADA
    1. Número de iteraciones (USUARIO)
    2. Que la red se entrene satisfactoriamente
    3. Cada vez que termine una iteración se debe calcular la sumatoria de las distancias vencedora de cada neurona de cada patrón y promediarla
       Dm=∑Dv/número de patrones
    4. Cuando este valor Dm registre (0) o muy parecido a (0) (0.1) ,(0.01), este valor se analiza iteración por iteración y lo debemos ir graficando en tiempo real (Grafica etiquetada)

5. En esta arquitectura de red es necesario (Obligatorio) ir graficando el comportamiento de los pesos sinápticos (Tiempo en real)
6. Cuando Alcanzamos el objetivo trazado en el entrenamiento DM se haga o se aproxime a (0), cuando se cumple esta condición **almacenamos los pesos sinápticos de forma permanente (Entrenamiento de la red – configuración de la red neuronal)**
7. SINO SE CUMPLE LA CONDICION QUE DM SE HAGA (0) O SE APROXIME A (0) Y SE TERMINA EL NUMERO DE ITERACIONES, DEBEMOS INICIAR UN NUEVO ENTRENAMIENTO
8. QUE SE DEBE TENER EN CUENTA:
    - **SI EL COMPORTAMIENTO DE LOS (DM) VA DISMINUYENDO, PODEMOS SEGUIR UTILIZANDO LOS PESOS DEL ENTRENAMIENTO ANTERIOR Y QUE SEAN LOS PESOS INICIALES PARA UN PROXIMO ENTRENAMIENTO (NO MODIFICAMOS EL NUMERO DE NEURONAS QUE TENEMOS EN EL PLANO)**
    - **PERO SI EL COMPORTAMIENTO DE DM ES DE IR AUMENTADO O SE ESTANCA (MINIMO LOCAL), NECESARIAMENTE DEBEMOS REINICIALIZAR LA RED (AUMENTAR EL NUMERO DE NEURONAS QUE TENEMOS EN EL PLANO), POR ENDE, DEBEMOS REINICIALIZAR ALEATORIAMENTE LOS PESOS SINAPTICOS**

SIMULACION

- Cargar la red entrenada – pesos y configuración óptimos

- Presentar un patrón de entrada (de los que participaron en el proceso de entrenamiento o patrones que la red no conoce)
- PATRONES QUE PARTICIPARON EN EL ENTRENEMIENTO
- PATRONES QUE NO PARTICIPARON EN EL ENTRENAMIENTO (GENERALIZAR)
- PATRONES CALCULADOS EN TIEMPO REAL (DISPOSITIVO - SENSOR) (GENERALIZAR)

• 0.3	• 0.6	• 0.9

- **CALCULAR LA DISTANCIA EUCLIDIANA EXISTENTE ENTRE EL PATRON DE ENTRADA Y CADA UNA DE LAS NEURONAS QUE SE ENCUENTRAN EN EL PLANO UTILIZANDO LOS PESOS OPTIMOS**
- $D\_l = \sqrt{\sum \ (\mathrm{X[j]} - \mathrm{W[J][i]})^2}$
- D1=raíz cuadrada ((X1-W14)^2+(X2-W2,4) ^2+(X3-W3,4) ^2)

j=1,2,3 i=1,2,3,4

$\mathrm{D}_1 = \sqrt{(0.3-0.1)^2 + (0.6-0.5)^2 + (0.9-0.8)^2}$

D1= 0.24

$D_2 = \sqrt{(0.3-0.3)^2 + (0.6-0.3)^2 + (0.9-0.9)^2}$

D2= 1.04

$D_3 = \sqrt{(0.3-0.2)^2 + (0.6-0.8)^2 + (0.9-0.9)^2}$

D3= 0.22

$4 = \sqrt{(0.3-0.2)^2 + (0.6-0.9)^2 + (0.9-0.9)^2}$

D4= 0.31

- CUAL ES LA SALIDA O LA RESPUESTA QUE DEBE ENTREGAR LA RED NEURONAL
- BUSCAMOS LA MENOR DISTACIA PARA CONSEGUIR LA NEURONA VENCEDORA (LA MAS CERCANA A CERO (0))
- LA NEURONA VENCEDORA SE CONVIERTE EN LA SALIDA DE LA RED (Los pesos sinápticos conectados a la vencedora)

0.32	**0.3**	**0.3**	**0.42**
0.72	0.3	**0.6**	1.12
1.02	**0.1**	**0.9**	**1.12**
N1	**N2**	**N3**	**N4**

-