T.P. 05

INTELIGENCIA ARTIFICIAL 2021

Tema: REDES NEURONALES ARTIFICIALES - APLICACIONES 26/may/2021

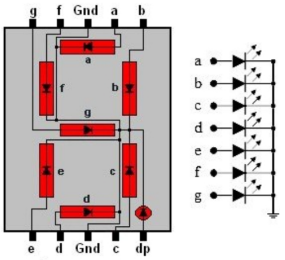
Grupo: Integran:

Profesor Mg. Ing. Sergio L. Martínez – Aux. Ing. Miguel A. Azar

1*. Conversor neuronal de código BCO a 7 segmentos

La tabla siguiente muestra la decodificación del código BCO (3 bits) a octal, y a su equivalente en código de 7 segmentos. A la derecha, se tiene la gráfica de un dígito de 7 segmentos, mostrando la ubicación de cada segmento.

Octal	B2	B1	B0	a	b	c	d	e	f	g
0	0	0	0	1	1	1	1	1	1	0
1	0	0	1	0	1	1	0	0	0	0
2	0	1	0	1	1	0	1	1	0	1
3	0	1	1	1	1	1	1	0	0	1
4	1	0	0	0	1	1	0	0	1	1
5	1	0	1	1	0	1	1	0	1	1
6	1	1	0	1	0	1	1	1	1	1
7	1	1	1	1	1	1	0	0	0	0



a) Configurar un decodificador neuronal utilizando 7 perceptrones, de tal forma que cada perceptrón maneje un segmento del display.

b) Desarrollar el proceso de entrenamiento de los perceptrones sobre un script de Matlab.

c) Desde otro script, mostrar una ejecución de todos los perceptrones, alimentados con todos los datos BCO.

d) Dibujar un esquema del sistema decodificador.

e) Emitir conclusiones del trabajo realizado.

2*. Clasificación con red RBF

La red RBF de la figura (a) se va a utilizar como herramienta de agrupamiento (clustering). Se implementará mediante las funciones de Matlab con un script.

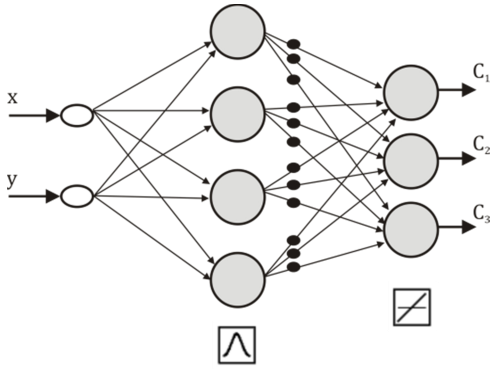


Figura (a)

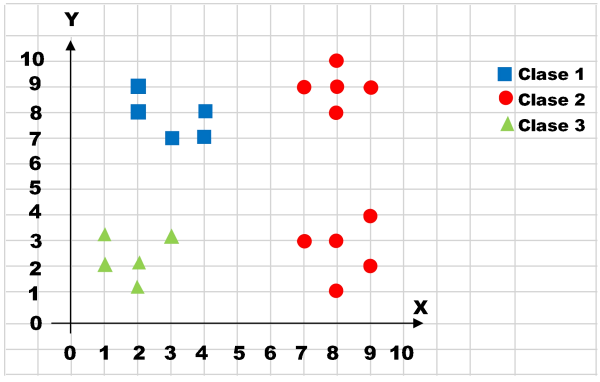


Figura (b)

a) Utilizar los puntos de la figura (b) para crear los patrones de entrenamiento de la red. La respuesta deseada debe ser un '1' en la neurona de salida que represente a la clase y un '0' en las restantes neuronas de salida.

b) Crear tres redes RBF para verificar la operación; una con 3 neuronas ocultas, otra con 4 neuronas ocultas (como en la figura a) y la tercera sin dar valor al parámetro de la cantidad de neuronas, en este último caso la función de creación de la red tomará la cantidad de neuronas necesarias (ver Problema Resuelto RBF). Seleccionar un valor de "spread" conveniente (por defecto utiliza el valor 1). Se recomienda un error objetivo igual a 0.

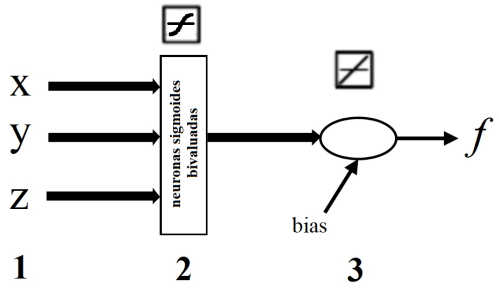
c) Comprobar el funcionamiento de las redes con al menos 10 puntos (de la grilla - fig. B - valores enteros), no utilizados para el entrenamiento. Volcar los resultados sobre el mismo espacio de datos de la figura (b), uno para cada red.

d) Dar conclusiones respecto de los resultados obtenidos.

3. Aproximación de funciones con red feedforward-backpropagation

Se desea aproximar la función dada a continuación, con una red feedforward-backpropagation, de tres capas, cuyo esquema genérico se muestra en la figura de la derecha.

$$f(x,y,z)=\sin(x)+\cos(y)+z$$
$$\text{con } x \in [0, 2\pi]$$
$$y \in [0, 2\pi]$$
$$z \in [-1, 1]$$



a) Utilizando la ecuación de la función anterior, generar un conjunto de entrenamiento de 20 puntos, distribuidos uniformemente en el espacio de datos indicado.

b) Determinar la arquitectura de la red feedforward-backpropagation, de acuerdo a los datos de entrada-salida. Para las neuronas ocultas, utilizar la heurística de cálculo dada por Manry.

c) Generar un conjunto de datos de comprobación, de por lo menos 30 puntos, que no hayan sido utilizados durante el entrenamiento.

d) Alcanzar un error de aproximación (MSE) de por los menos 10^{-4}. Luego, calcular el error de comprobación, comparar ambos errores y comentar al respecto.

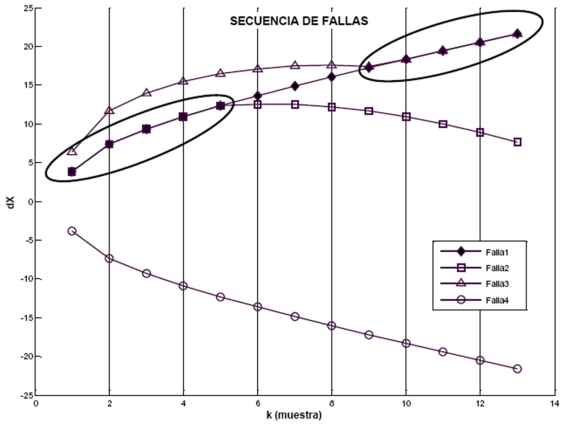
e) El proceso de aproximación se puede desarrollar desde la línea de comandos de Matlab o con asistencia de la GUI *nntool*.

f) Dibujar las superficie que se generan con los datos calculados por la red y la superficie exacta, dada por la ecuación *f*(x,y,z), utilizado en ambos casos los mismos puntos de las variables de entrada. Investigar sobre la graficación de superficies en Matlab (función *mesh()* u otras). Para poder graficar debe seleccionar dos de lastres variables independientes.

4. Identificación de patrones con red LVQ

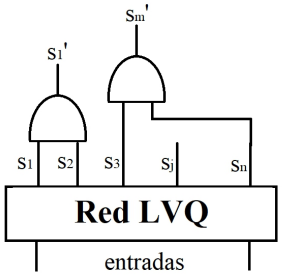
El archivo *Patrones_fallas.mat* contiene cuatro patrones F1, F2, F3 y F4 correspondientes a estados de fallas de un determinado proceso, cuyas imágenes gráficas se muestran en la figura siguiente.

Tales secuencias muestran rasgos similares para diferentes fallas (dentro de las elipses), que es difícil de clasificar.



a) Utilizando la aplicación gráfica ***nntool***, crear una red LVQ de 13+4, para clasificar tales patrones.

b) En caso de que la red LVQ no pueda asociar correctamente los patrones a cada una de las cuatro salidas, aumentar la cantidad de salidas y componer los resultados parciales mediante compuertas AND (ver figura).



c) El sistema Matlab, dispone de una aplicación gráfica específica para el reconocimiento de patrones, que se activa con el comando ***nprtool***. Utilizando esta GUI, comprobar el resultado obtenido en los pasos anteriores.