Redes Neuronales. Práctica Guiada.

- Manejo del simulador de redes neuronales SNNS.
- Construcción de una red neuronal para aproximar la función lógica **XOR**.

La función lógica **XOR** básica tiene 2 entradas y una salida, todas ellas binarias.

C = XOR (A , B)

Su tabla de verdad es la siguiente.

Entrada A	Entrada B	Salida C
0	0	0
0	1	1
1	0	1
1	1	0

El objetivo de la práctica es construir una red con el modelo de retropropagación que sea capaz de aproximar dicha función. La red empleada tendrá un tamaño 2x2x1. 2 entradas, 2 neuronas ocultas y 1 neurona de salida.

Los datos necesarios para el entrenamiento están en el fichero **xor1.pat** [1].

Pasos:

1 Desde el shell de Unix, llamar al simulador SNNS con el comando **snns**.

2 Aparecerán dos ventanas. Pinchar con el ratón en la ventana/panel informativo y desaparecerá.

3 En el panel llamado **snns-manager** elegiremos la opción **bignet** para crear una red.

Notación: **bignet**[snns-manager].

4 En el menú que aparece al pinchar en **bignet**, hay que seleccionar **general**.

5 Aparece un nuevo panel que sirve para diseñar las redes por capas (en la parte de arriba). Cada capa puede tener las neuronas dispuestas en un plano de dos dimensiones (**x** e **y**). El número total de neuronas será **x*****y**. Los pasos para definir una capa son:

5.1 Introducir el tamaño de la capa en la dirección x (**x-direction**). Si la disposición debe ser lineal, como en este caso, introduciremos un 1.

[1] Disponible en la página web de las prácticas: http://scalab.uc3m.es/~docweb/it-sup/practicas.

5.2 Introducir el tamaño de la capa en la dirección y (**y-direction**). Si la disposición debe ser lineal, como en este caso, introduciremos el número de neuronas de la capa.

5.3 Seleccionar en el botón **type** el tipo de capa, a elegir entre **input**, **hidden** y **output**.

5.4 Pulsar el botón **enter** (de arriba) para aceptar los datos y pasar a la siguiente capa.

5.5 La secuencia para diseñar nuestra red será: **type**: input, **x**:1, **y**:2, **enter**; **type**: hidden **x**:1, **y**:2, **enter**; **type**: output **x**:1, **y**:1, **enter**.

6 Una vez introducidos los datos de las capas, hay que interconectar las neuronas. En el panel de abajo pulsar **full conection**.

7 Pulsando **create network**, generamos la nueva red. Minimizar la ventana de bginet.

8 En **display**[snns-manager] (es decir, en el panel de snns-manager) podemos abrir un nuevo panel que muestra la red recién diseñada.

9 La información mostrada en este panel se puede alterar y configurar pulsando **setup**[display].

10 Activando links **on**[setup] y links **-2.35**[setup] añadiremos al gráfico líneas que representan las conexiones y los pesos correspondientes. Cerrar el panel de [setup] pulsando **done**[setup].

11 Pulsar en **control**[snns-manager]. Aparece el panel de control del aprendizaje de redes.

12 Seleccionar en el primer botón **sel func**[control] el algoritmo de aprendizaje Std_Backpropagation (el primero de la lista).

13 Observar en el panel los avisos de **Training Pattern File?**, **Validation Pattern File**. Hay que cargar los patrones de entrenamiento antes de proceder al proceso de aprendizaje.

14 Pulsar **file**[snns-manager]. El nuevo panel permite leer y grabar ficheros con descripciones de redes o de patrones.

15 Pulsar en **pat**[file], seleccionar el fichero **xor1** y pulsar **load**[file]. Pulsar **done**[file] para cerrar el panel de ficheros.

16 En el panel de control han desaparecido los mensajes de aviso. Pulsar en **use**[control] para ver el menú de ficheros de patrones disponibles (sólo debería haber uno: **xor1**).

17 Inicializar la red con **init**[control]. Observar los cambios que produce en el panel de display. Observar el rango de inicialización en el panel de control abajo, junto a la palabra Init (por omisión, de -1 a 1).

18 Ajustar los parámetros de aprendizaje, junto a Learn. En principio, sólo interesa la tasa de aprendizaje (0.2).

19 Pulsar repetidamente el botón **test**[control] observando al mismo tiempo los cambios en el panel de [display]. Identificar la secuencia de patrones en las entradas y salidas de la red.

20 Ajustar el número de ciclos a 100. Cada ciclo (iteración) supone presentar todos los patrones una vez a la red.

21 El aprendizaje se lanza pulsando **all**[control]. Durará 100 iteraciones. En la ventana del terminal se pueden ver la evolución del error cometido por la red durante el progreso del aprendizaje. Los errores son:

SSE: Error cuadrático medio acumulado para todos los patrones.

MSE: SSE medio sobre el número de patrones

SSE/o-units: SSE medio sobre el número de salidas.

22 Probar de nuevo el paso 19. La red debería dar en la salida valores próximos a 0.5, es decir, que no ha aprendido todavía.

23 Ajustar los ciclos a 5000, lanzar otro proceso de aprendizaje con **all**[control]. Los errores en la ventana del terminal deberían descender hacia 0. Caso de no suceder, repetir el proceso de aprendizaje. Si se sigue resistiendo, incializar la red (**init**[control]) y repetir el proceso de aprendizaje (**all**[control]).

24 Repitiendo el paso 19/22, deberíamos observar que la red aproxima muy bien los patrones.

25 Pulsar **graph**[snns-manager], para abrir un panel de representación gráfica (tipo osciloscopio).

26 Inicializar la red (**init**[control]), lanzar un proceso de aprendizaje (**all**[control]). Observar la evolución del error en el gráfico.

27 Ajustar las escalas **x** e **y** en [graph]. Elegir errores diferentes al SSE. Probar el botón de **clear**[graph].

28 Grabar la red ya aprendida en **file**[snns-manager].

Segunda Parte.

Partiendo de la red ya entrenada para aproximar la función **XOR**, supongamos que ahora queremos comprobar que ha aprendido fuera del los puntos (0,0), (0,1), (1,0) y (1,1).

¿Podemos suponer que ha aprendido a aproximar una función similar a C = | A - B |?

(Esta función genera los mismos resultados que la XOR en los puntos (0,0), (0,1), (1,0) y (1,1).

Hay dos ficheros con patrones de esta función: **abs1.pat** y **abs2.pat**, cada uno con 50 patrones distintos.

29 Cargar el fichero **abs1.pat** en **file**[snns-manager].

30 Seleccionar como fichero de entrenamiento **xor1.pat** (en **use**[snns-control] el de arriba).

31 Seleccionar como fichero de validación **abs1.pat** (en **use**[snns-control] el de abajo).

32 Escribir en el campo **Valid**[snns-control] un 1. Esto quiere decir que por cada ciclo de aprendizaje se va a realizar una validación de todos los patrones.

33 Abrir el **graph**[snns-manager] o borrar su contenido y lanzar el aprendizaje. En negro se puede observar la evolución del error de aprendizaje y en rojo el de validación [2].

34 Juzgar si la red está aprendiendo algo similar a la función de valor absoluto.

35 Intercambiar los ficheros de validación y de aprendizaje (con los **use**[snns-control]). Ahora se llevará a cabo el aprendizaje con **abs1.pat** y la validación con **xor1.pat**.

36 Realizar pruebas similares con los ficheros **abs1.pat** para aprendizaje y **abs2.pat** para la validación.

36 Editar los ficheros **xor1.pat** y **abs1.pat** para observar su estructura. . Para crear un nuevo fichero de patrones (para otro problema, por ejemplo) hay que conservar la cabecera, modificar los datos correspondientes a número de patrones, número de entradas y número de salidas y hay que sustituir los datos restantes (conservando siempre las etiquetas # i y # o para identificar los datos de entrada y los de salida.

[2] Es probable que los errores de aprendizaje y validación tengan magnitudes muy distintas. Esto se produce siempre que la cantidad de patrones en uno y otro conjunto sea diferente. Hay que ajustar la escala del **graph** para poder visualizarlos.