



03063  
UNIVERSIDAD NACIONAL  
AUTONOMA DE MEXICO

U. A. C. P. Y P. DEL C. C. H.

I. I. M. A. S.

FALLA DE ORIGEN

IMPLEMENTACION DE UNA RED NEURONAL  
MULTI - CAPA PARA EL RECONOCIMIENTO DE  
PATRONES

T E S I S

que para obtener el grado de

MAESTRA EN CIENCIAS DE LA COMPUTACION

presenta

ARANZAZU FERNANDEZ AYALA

México, D.F.

1991.



Universidad Nacional  
Autónoma de México



## **UNAM – Dirección General de Bibliotecas Tesis Digitales Restricciones de uso**

### **DERECHOS RESERVADOS © PROHIBIDA SU REPRODUCCIÓN TOTAL O PARCIAL**

Todo el material contenido en esta tesis está protegido por la Ley Federal del Derecho de Autor (LFDA) de los Estados Unidos Mexicanos (México).

El uso de imágenes, fragmentos de videos, y demás material que sea objeto de protección de los derechos de autor, será exclusivamente para fines educativos e informativos y deberá citar la fuente donde la obtuvo mencionando el autor o autores. Cualquier uso distinto como el lucro, reproducción, edición o modificación, será perseguido y sancionado por el respectivo titular de los Derechos de Autor.

## INDICE

	PAG.
PROLOGO .....	1
INTRODUCCION .....	3
CAPITULO I .....	6
1.1-Bases Neuro-Fisiológicas, y su relación con las redes neuronales.	
REFERENCIAS .....	
CAPITULO II .....	10
2.1-Red de McCulloch-Pitts	
2.2-Perceptrón	
2.3-Red de Hopfield	
2.4-Cognitrón y Neocognitrón	
REFERENCIAS .....	
CAPITULO III .....	17
3.1-Neuronas Adaptables	
3.2-Algoritmo LMS	
3.3-Regla MRPI	
REFERENCIAS .....	
CAPITULO IV .....	25
4.1-Implantación de la Red	
4.1.1--Estructura	
4.1.2-Inmunidad ante Traslación y Rotación	
4.1.3-Implantación	

<b>CAPITULO V</b> .....	<b>29</b>
5.1-Análisis de Parámetros	
5.2-Rango de Pesos	
5.3-Redes con distintas configuraciones	
<b>REFERENCIAS</b>	
<b>CAPITULO VI</b> .....	<b>59</b>
6.1-Comparación de los resultados con un método estadístico de reconocimiento	
6.2-Conclusiones	
<b>ANEXO</b> .....	<b>66</b>
Programa en Fortran-77	

**PALABRAS CLAVE :**

**Redes neuronales, Adaline, neuronas, neuronas naturales, adaptable, retroalimentación, aprendizaje.**

## PROLOGO

Este trabajo tiene como objetivo el desarrollo y estudio de una *red neuronal* multi-capa capaz de reconocer imagenes binarias de caracteres alfanumericos.

En la introducción se da una breve explicación de lo que es una *red neuronal*. En primer lugar se define el concepto de *neurona* y todos aquellos parametros necesarios para la elaboración de una *neurona*. Una vez establecida la definición de *neurona* se explica de forma general lo que entendemos por *red neuronal*, indicando algunos tipos de redes existentes que son detallados en capitulos posteriores. Por último en esta sección se habla de las diferentes aplicaciones que se le han dado a las *redes neuronales*.

En los capitulos primero y segundo se presenta una serie de antecedentes históricos de las *redes neuronales* actuales y algunos de los modelos más importantes de dichas redes. Se tratan aquellos aspectos de la visión humana en los que se han basado las *redes neuronales*. Posteriormente se menciona brevemente los modelos matemáticos que han servido como punto de partida para algunas redes, y dos algoritmos distintos que están siendo utilizados en la etapa de *aprendizaje* de las redes. Al final del capítulo segundo se presenta un resumen de cinco tipos de redes, a saber: Red de McCulloch y Pitts, Perceptron, Red de Hopfield, Cognitrón y Neocognitrón.

El capítulo tercero detalla los fundamentos matemáticos de la *red neuronal* descrita por Winter-Widrow, mismos que han sido empleados para la implantación de la red que se describe en el capítulo cuarto. En esta sección se ve la aplicación del algoritmo LMS (Winter-Hoff) para el ajuste de pesos de la red en el proceso de *aprendizaje*. También se describe la regla MREI que ha sido aplicada en la implantación de la red.

Una vez establecidas las bases matemáticas, se pasa, en el capítulo cuarto, a explicar la implantación de la red. Dicha red, aunque guarda una similitud con la descrita por Winter-Widrow, presenta diferencias estructurales y cambios en los valores numéricos de los parámetros. En este capítulo se anexa una copia del programa empleado para la implantación de la red y otra de los veinte patrones que se utilizaron para las etapas de aprendizaje y reconocimiento para la red. De hecho se realizaron dos programas : uno en Turbo-Pascal para su uso en PC y otro en Fortran77 que permite una mayor transportabilidad a otras computadoras en nuestro caso una Burroughs-A9, este último es el programa que se detalla. En cuanto a los patrones, ya que se implantaron redes de distintos tamaños ( 49, 100, 144 y 225 neuronas por capa), se hicieron cuatro series de patrones de las cuales solo presentaremos la correspondiente a la red de 49 neuronas por capa. El resto de las series se intentó que fueran lo mas parecido a ésta, variando solo el tamaño aunque no fue posible guardar una escala rigurosa.

Los resultados experimentales obtenidos se describen en el capítulo quinto. En él se discuten los valores asignados a los parámetros de forma experimental con la ayuda de una serie de gráficas, fruto de los resultados. Se estudia la constante de convergencia de la red, el intervalo de pesos inicial que se le asigna de forma artificial y la relación entre su sensibilidad y su tamaño.

Finalmente, el capítulo sexto versa sobre el estudio comparativo de los resultados obtenidos con la red y la similitud que presentan los patrones por otro método. Asimismo se darán las conclusiones y sugerencias para experimentos posteriores.

## INTRODUCCION

Ya que este trabajo tiene como objetivo el mostrar la implantación de una red neuronal, debemos en primer lugar aclarar nuestro concepto de neurona. En una neurona natural perteneciente a un ser vivo, se pueden distinguir tres partes fundamentales, a saber: las dendritas, el axón y el soma. La neurona natural manda su información por el axón, misma que es recibida por otras neuronas naturales en los puntos de contacto (sinapsis) con sus dendritas o con su soma. Cada neurona natural recibe información de cientos o miles de otras neuronas naturales y en respuesta manda información a otras tantas neuronas naturales. En este proceso la información de entrada es transformada por la neurona natural obteniéndose un impulso de salida.

En las computadoras se ha intentado simular el proceso realizado por el cerebro dando lugar a neuronas artificiales. Dichas neuronas constan de unos receptores de la señal de entrada, cada uno de los cuales contribuye en una cierta medida o peso ( $w$ ) a la salida de la neurona. En nuestro caso la señal de entrada corresponde al patrón de entrada, esto es una matriz de  $N \times N$  pixels con valores de 0 ó 1. De ahora en adelante denominaremos al patrón de entrada como vector  $X(x_1, x_2, \dots, x_{N \times N})$ , teniendo la neurona tantos receptores como componentes tenga el vector  $X$ . Los pesos asignados a cada entrada conformarán el vector de pesos  $W(w_1, w_2, \dots, w_{N \times N})$ . La neurona una vez

recibida la señal de entrada realiza la siguiente operación: 
$$\sum_{i=1}^{i=N \times N} x_i w_i$$

Este resultado constituye la señal de salida analógica de la neurona. En los casos mas sencillos cada neurona produce su salida realizando el producto interno del vector de entrada por el vector de pesos. Ahora bien, generalmente es necesario obtener una señal binaria de salida por lo que se le aplica a la señal de salida analógica una función como

## INTRODUCCION

Ya que este trabajo tiene como objetivo el mostrar la implantación de una red neuronal, debemos en primer lugar aclarar nuestro concepto de neurona. En una neurona natural perteneciente a un ser vivo, se pueden distinguir tres partes fundamentales, a saber: las dendritas, el axón y el soma. La neurona natural manda su información por el axón, misma que es recibida por otras neuronas naturales en los puntos de contacto (sinapsis) con sus dendritas o con su soma. Cada neurona natural recibe información de cientos o miles de otras neuronas naturales y en respuesta manda información a otras tantas neuronas naturales. En este proceso la información de entrada es transformada por la neurona natural obteniéndose un impulso de salida.

En las computadoras se ha intentado simular el proceso realizado por el cerebro dando lugar a neuronas artificiales. Dichas neuronas constan de unos receptores de la señal de entrada, cada uno de los cuales contribuye en una cierta medida o peso ( $w$ ) a la salida de la neurona. En nuestro caso la señal de entrada corresponde al patrón de entrada, esto es una matriz de  $N \times N$  pixels con valores de 0 o 1. De ahora en adelante denominaremos al patrón de entrada como vector  $X(x_1, x_2, \dots, x_{N,N})$ , teniendo la neurona tantos receptores como componentes tenga el vector  $X$ . Los pesos asignados a cada entrada conformarán el vector de pesos  $W(w_1, w_2, \dots, w_{N,N})$ . La neurona una vez

recibida la señal de entrada realiza la siguiente operación: 
$$\sum_{i=1}^{i=N,N} x_i w_i$$

Este resultado constituye la señal de salida analógica de la neurona. En los casos mas sencillos cada neurona produce su salida realizando el producto interno del vector de entrada por el vector de pesos. Ahora bien, generalmente es necesario obtener una señal binaria de salida por lo que se le aplica a la señal de salida analógica una función como

por ejemplo:

$$g(x) = \frac{1}{1+e^{-x}} .$$

Este tipo de funciones son las que denominaremos en el futuro como "cuantificador" de la señal de salida, o en algunos casos función de umbral "θ".

Cuando una neurona se interconecta con otras, de forma que dado un vector de señales de entrada se origine otro de salida, entonces se dice que dichas neuronas forman una red neuronal. La forma de interconexión entre neuronas varía de unos modelos a otros. Existen redes que actúan con propagación hacia delante (feed-forward), esto es, el flujo de información siempre va desde las neuronas de una capa a las neuronas de la capa siguiente. También hay redes con retroalimentación (feed-backward), esto es, flujo de información de las neuronas de una capa a las neuronas de capas anteriores, y otros tipos en los que la información fluye dentro de una misma capa.

Aparte de las distintas formas en que viajan las señales dentro de la red, existen diversas formas de conectar a las neuronas dentro de la red. Las neuronas pueden estar todas interconectadas o pueden agruparse en capas con distinto número de neuronas existiendo conexiones sólo entre neuronas de distintas capas, o a veces sólo entre grupos de neuronas de distintas capas (generalmente subsecuentes). El tipo de conexión elegido puede facilitar o hacer prácticamente imposible la solución del problema que se quiere resolver.

Aunque en el caso de las redes neuronales la difusión comercial todavía no es muy amplia, ya se han obtenido buenos resultados en el reconocimiento visual de patrones, en el reconocimiento fonético, detección de movimiento y reconocimiento adaptativo de patrones [ 1 ], lo que configura nuestro tema de estudio. En teoría, un sistema neuronal debe ser capaz de reaccionar en forma favorable ante situaciones desconocidas, tras un periodo de entrenamiento [2,3 y 4].

## REFERENCIAS

- 1.-King T., "Using Neural Networks for Pattern Recognition", Dr. Dobb's Journal, Enero de 1989, pp. 14-28.
- 2.-Widrow B. y Winter R., "Neural Nets for Adaptive Filtering and Adaptive Pattern Recognition", Computer, Marzo de 1988, pp. 25-39.
- 3.-Fukushima K., Miyake S. e Ito T., "Neocognitron: a neural network model for a mechanism of visual pattern recognition", IEEE Transactions on Systems, Man and Cybernetics SMC-13, 1983, pp. 826-834.
- 4.-Fukushima K., "A Neural Network for Visual Pattern Recognition", Computer, Marzo 1988, pp. 65-75.

## CAPITULO I

### 1.1-BASES NEURO-FISIOLOGICAS, Y SU RELACION CON LAS REDES NEURONALES.

Ya que las redes neuronales intentan simular algunas de las funciones del cerebro, vamos a explicar brevemente el funcionamiento de éste.

La función cerebral depende de las interconexiones entre varios billones de células especializadas llamadas neuronas [3], a las que denominaremos *neuronas naturales* para no confundirlas con las neuronas simuladas. Cada *neurona natural* recibe información de cientos o miles de otras *neuronas naturales* y en respuesta manda información a otras tantas *neuronas naturales*. Estas conexiones se van haciendo durante el crecimiento del cerebro y se fijan en la edad adulta.

Aunque no existen dos *neuronas naturales* exactamente iguales, si se pueden diferenciar distintas clases de *neuronas naturales* con base en su forma. De hecho se ha comprobado que la actividad neuronal depende de la forma del area que ocupan [1]; pudiéndose representar una *neurona natural* matemáticamente, por ejemplo la transformada tridimensional de Fourier  $f(x,y,z)$ , donde  $f$  especifica la densidad en cada punto del area. Los coeficientes en la expansión de Maclaurin de la función característica de la *neurona natural* sirven como parámetros para la especificación de la forma neuronal [1].

La corteza cerebral humana envuelve casi totalmente al resto del cerebro, tendiendo a oscurecer a las otras partes. Un humano sin corteza es prácticamente un vegetal, sin habla, ciego e insensible. Los elementos básicos de la corteza son grandes cantidades de *neuronas naturales* interconectadas por medio de sinapsis. Si nos fijamos en la corteza primaria visual podemos ver que morfológicamente es bastante uniforme en dos de sus dimensiones. Un acercamiento por microscopio a

las células muestra que a pesar de la uniformidad existen variaciones estructurales que van en concordancia con las subdivisiones clínicas. Se ve como hay un mapeo del cuerpo hacia áreas somáticas sensoriales o motrices; en el caso del mundo visual éste se ve reflejado en la corteza visual primaria [2].

La primera información sobre la organización cortical fue, a fines del siglo diecinueve, el reconocimiento de la existencia de áreas con funciones totalmente distintas, y con una tendencia a un mapeo ordenado. Hoy en día se habla de 50 a 100 áreas distintas en la corteza.

A principio del siglo veinte, D. Santiago Ramón y Cajal [2,3] descubrió que las operaciones que realiza la corteza con la información que le llega son locales. Las conexiones entre entradas y salidas varían de un área a otra, pero dentro de una misma zona siguen un patrón.

La información llevada a la corteza por una fibra puede atravesar en profundidad la corteza en tres o cuatro sinapsis, mientras que la propagación lateral que se da a través de las dendritas y los axones se limita a unos pocos milímetros, lo que es una proporción muy pequeña del total de la corteza. Esto indica que la respuesta de la corteza es posiblemente local.

La corteza visual primaria no es el punto final en el camino visual. Es simplemente una etapa inicial en cuanto al grado de abstracción de la información que maneja. Una de las tareas fundamentales que realiza es la reorganización de la información de entrada de forma que la mayoría de sus células respondan a los segmentos lineales específicamente orientados y no a las zonas luminosas.

Las neuronas naturales con orientación específica varían en su complejidad; las más sencillas, a las que llamaremos células simples, actúan como si recibieran señales provenientes de células cuyos campos mantienen una simetría circular. Las células complejas actúan como si

recibieran su información de células simples todas con la misma orientación del campo receptor pero con alguna variación en la localización de su campo.

En el caso de una imagen difusa sólo se disparan las células simples cuyo campo de acción comprenda los bordes de la imagen. Podemos encontrar toda la gama de dominio ocular en las células ; desde el dominio total de un ojo pasando por la igualdad entre ambos ojos hasta llegar al dominio del otro ojo. Gran parte de las conexiones interneuronales se dan antes del nacimiento, aunque las conexiones binoculares se realizan en su mayoría después del mismo.

Se han realizado experimentos con animales a los que previamente se les inyecta desoxiglucosa como material marcado radiactivamente (método debido a Louis Sokoloff). Wiesel realizó un experimento [2] en el que después de 45mn. de mostrarle al animal un dibujo consistente en rayas verticales blancas y negras se analiza la corteza, observándose radiactividad en bandas estrechas perpendiculares a la superficie mostrada, a lo largo de toda la corteza. En los cortes paralelos se observo un conjunto complejo de bandas espaciadas periódicamente. De este y otros experimentos realizados con aminoácidos marcados, se deduce que la corteza visual primaria esta formada por bloques de un milímetro cuadrado por dos milímetros de profundidad. Las *neuronas naturales* se excitan o no en base a estímulos específicos y las diversas transformaciones que sufre la información se lleva a cabo por distintos grupos de *neuronas naturales*.

En el caso de las *redes neuronales* se ha tratado de seguir en lo posible el comportamiento de la corteza cerebral. Se intenta que sean grupos de *neuronas* los que respondan a distintos impulsos externos o señales de entrada. También se implantan las redes como capas de *neuronas* de forma que las *neuronas* de una capa incidan directamente en las *neuronas* de la capa siguiente.

## REFERENCIAS

- 1.-Stevens C.F., "Quantitative Specification of Neuron Form",  
Lectures on Mathematics in the Life Sciences, Vol.15,1982.
- 2.-Hubel D.H. and Wiesel T.N., "Brain Mechanisms of Vision",  
Scientific American, Vol.241,Sept.1979, pp. 150-162.
- 3.-Arbib M.A., "Brains, Machines and Mathematics", Springer-Verlag,  
2nd. Edition,1987.

## CAPITULO II

### 2.1-RED NEURONAL DE MCCULLOCH Y PITTS.

Las primeras neuronas simuladas datan de 1943, cuando Warren McCulloch y Walter Pitts propusieron su modelo [1,5]. Ellos muestran como cualquier maquina de estado finito puede simularse mediante una red conformada con las neuronas por ellos propuestas. Estas constan de  $m$  entradas  $x_1, x_2, \dots, x_m$  ( $m \geq 1$ ) y una salida  $y$ . Se caracteriza por  $m+1$  números, un umbral  $\theta$  y los pesos  $w_1, w_2, \dots, w_m$  donde  $w_i$  esta asociado con  $x_i$ . Suponiendo que la neurona opere en una escala de tiempo  $n = 1, 2, 3, \dots$  su salida en el tiempo  $n+1$  esta determinada por las entradas en el tiempo  $n$  siguiendo la siguiente regla : se emite un impulso en el tiempo  $n+1$  si la suma de las entradas en el tiempo  $n$  sobrepasa al umbral  $\theta$  de la neurona. Esto es,

$$y(n+1)=1 \quad \text{ssi} \quad \sum_1^m x_i(n) \geq \theta$$

Con esta neurona se pueden implantar las funciones lógicas AND, OR y NOT [1].

La red neuronal está compuesta por una serie de neuronas de McCulloch-Pitts con la misma escala de tiempo, interconectadas de forma que la salida de una neurona actúe como entrada para una o mas de las otras neuronas. La salida de la red corresponde a la salida de aquellas neuronas que no sirven de entrada para otras. Este modelo sencillo de red sentó las bases para el desarrollo de redes mas sofisticadas.

### 2.2-PERCEPTRON.

En los cincuentas, Frank Rosenblott y sus colegas [1,5] en la Universidad de Cornell desarrollaron una red neuronal denominada perceptron [1]. La diferencia básica con el modelo de McCulloch-Pitts, es que ellos permiten que los pesos de cada neurona cambien con el

tiempo. De esta forma se le permite a la red la posibilidad de ir cambiando (aprendiendo) con el tiempo.

Un perceptrón con una sola capa puede utilizarse tanto con entradas de tipo analógico como con entradas binarias. Esta red sencilla causo gran expectativa cuando se desarrollo debido a su capacidad de aprendizaje en el reconocimiento de patrones sencillos.

De acuerdo con lo que se expuso en el capitulo I sobre el sistema nervioso, podemos hacer la analogía entre dicho sistema y el perceptrón. La retina equivale a los receptores del perceptrón; estos al ser estimulados transmiten el impulso a las *neuronas* situadas en capas intermedias (preprocesadores o unidades asociativas). Si el total de estímulos que llegan a la *neurona* excede el umbral de la misma, ésta se activa enviando un impulso a las *neuronas* a las que se encuentra conectada (conexión hacia delante). De igual forma en el sistema nervioso los impulsos son transmitidos a lo largo de las capas neuronales. Por último en el sistema nervioso se activan aquellas *neuronas* cuya salida controla la señal de respuesta, lo que es equivalente a la salida de las *neuronas* dentro de la red que ya no están conectadas a otras (*neuronas* de la última capa).

El perceptrón actúa como si tuviera una memoria a largo plazo. Esto quiere decir que es capaz de modificar los pesos matemáticos de las *neuronas* en base a una corrección por error. Si la *neurona* responde erróneamente ante un estímulo sus pesos matemáticos son cambiados. Estos y la medida de umbral se pueden fijar o adaptar utilizando diversos algoritmos. El procedimiento original de convergencia para el ajuste de pesos, desarrollado por Rosenblatt [5] se puede resumir como sigue: a) inicialización de pesos y  $\theta$  con valores aleatorios pequeños, b) presentación de un patron de entrada junto con la señal deseada de salida, c) calculo de la señal de salida y en el tiempo  $t$ , donde

$$y(t) = f_h \left( \sum_{i=0}^{N-1} w_i(t) x_i(t) - \theta \right)$$

d) ajuste de pesos, solo en aquellos casos en los que exista error, de la siguiente manera

$$w_i(t+1) = w_i(t) + \eta [d(t) - y(t)] x_i(t)$$

donde  $0 \leq \eta \leq 1$ , y  $\eta$  es un factor positivo de ganancia menor que 1. Este proceso se repite hasta que se alcanza la convergencia de la red.

Para que el perceptrón aprenda una serie de patrones, se le tiene que presentar dicha serie tantas veces como sea necesario para que, con el vector de pesos resultante después de los ajustes, los patrones sean identificados correctamente. Esto puede dar lugar a una etapa de entrenamiento algo tediosa pero en definitiva muy efectiva.

Mediante una modificación del método de convergencia del perceptrón se puede llegar al algoritmo de mínimos cuadrados (LMS) debido a Widrow-Hoff.

### 2.3-RED DE HOPFIELD.

Como ya es sabido el sistema óptico cerebral maneja la información en paralelo con la ayuda de unidades procesadoras sencillas (*neuronas*) interconectadas entre si. Basándose en esta idea, Hopfield realizó un modelo neuronal en 1986 [2]. En él, las *neuronas* participan a través de una operación de umbral y la información se guarda en las conexiones entre *neuronas*. Dependiendo del vector de entrada a la red, algunas *neuronas* se activan.

En el caso de las *neuronas* de McCulloch-Pitts, cada *neurona* procesaba las entradas para determinar una salida en ese lapso de tiempo. La red de Hopfield se rige por la siguiente regla de actualización: "Selecciona una *neurona* de forma aleatoria. Si la suma de pesos en las conexiones con otras *neuronas* activas es positiva, activa la *neurona*, en caso contrario, desactívala". Hopfield asume que los pesos son simétricos ( $w_{ij} = w_{ji}$ ), lo que está de acuerdo con la mayoría de los algoritmos en visión.

La inclusión de retroalimentación en estas redes las diferencia de las redes tipo perceptrón, que enfatizan la conectividad hacia adelante.

Las *neuronas* son sistemas dinámicos y continuos y por lo tanto los

modelos neuronales deben tener la capacidad de describir cantidades continuas. Las neuronas de McCulloch-Pitts son elementos lógicos de decisión descritos por una variable de dos estados (activo-inactivo), y cuyas entradas provienen de unas pocas neuronas. Este modelo adolece de dos de las características de las neuronas biológicas, a saber : procesamiento analógico y alta interconectividad. En la red de Hopfield la función de activación de la neurona oscila de forma continua entre 0 y el valor de umbral  $\theta$  (generalmente es una función de forma sigmoidea y monótonica ); además la red presenta una estructura de conexiones masivas en paralelo [3].

Una de las mayores aportaciones de Hopfield fue la inclusión de la medida de Energía, que se puede describir como :

$$E = -1/2 \sum_{i,j} s_i s_j w_{ij} + \sum_i s_i \theta_i$$

La inclusión del concepto de energía equivale a decir : " Se toma aleatoriamente una neurona. Se cambia su estado (de 0 a 1, o viceversa) únicamente en aquellos casos en los que el cambio disminuya la energía ".

El concepto de una función de Energía y su uso en el diseño de circuitos nos facilita la rápida resolución de problemas de optimización. El estado de cada neurona cambia en el tiempo de forma simple en función del estado de las neuronas a las que está conectada, pero la organización de las sinapsis determina la dinámica colectiva que minimiza la función de energía correspondiente, al problema de optimización.

#### 2.4-COGNITRON Y NEOCOGNITRON.

En 1975 Fukushima, Miyake e Ito[4] propusieron un modelo de red neuronal multicapa llamado cognitron y en 1983 le dieron una nueva orientación a la arquitectura del modelo, conociéndose esta nueva versión como el neocognitrón. Por último, en 1988 Fukushima mejoró el modelo con el fin de discernir entre varios patrones presentados simultáneamente a la red. Este modelo se va perfeccionando constantemente.

La red tipo cognitrón utiliza el método de aprendizaje sin maestro (aprendizaje no supervisado). El procedimiento empleado en este modelo se ha clasificado bajo el paradigma de aprendizaje competitivo. El cognitrón no tiene la capacidad de reconocer patrones distorsionados en su forma o desplazados de lugar.

El neocognitrón ya es capaz de identificar correctamente los patrones desplazados o distorsionados. Ahora bien, cuando se le presentan dos o más patrones simultáneamente, disminuye la probabilidad de identificación correcta.

En un sistema de aprendizaje, generalmente se tiene acceso a las entradas y salidas del sistema. Se pueden cambiar las conexiones entre los elementos de entrada y salida utilizando una regla de Hebb, o suministrando una señal de error mediante la diferencia entre la salida deseada y la real como se establece en la regla de Widrow-Hoff. Ahora bien, el ajuste de las neuronas de las capas ocultas, esto es capas que no son ni de entrada ni de salida, es difícil de realizar. Las máquinas de Boltzmann y el algoritmo de retroalimentación nos da una solución explícita al problema de aprendizaje en redes multicapa para una serie de casos.

En el neocognitrón se presupone que antes de construir la red ya se tiene una idea del tipo de resultado que se desea obtener y de la estrategia a seguir en el procesamiento de información. La estructura del neocognitrón es similar a la del modelo visual propuesto por Hubel-Wiesel. En esta red las neuronas están conectadas en cascada (la red consta de 9 capas de neuronas) y tiene la tendencia a responder selectivamente a los distintos patrones. A medida que las capas son más profundas dentro de la red su campo de recepción es mayor y por consecuencia son más insensibles a los cambios de posición del patrón de entrada. Todas las neuronas utilizadas son de tipo analógico y se van alternando neuronas S cuyos campos de recepción provienen aproximadamente de la misma zona del patrón de entrada y neuronas C. Estas últimas están conectadas a varias neuronas S y se activan al activarse cualquiera de las neuronas S que le sirvan de entrada. De

esta forma las neuronas C siguen estimulándose aunque el patrón cambie de posición. La diferencia entre las neuronas tipo S y las C estriba en las distintas entradas que las alimenta y sus conexiones de salida, ya que el tratamiento de la información es el mismo por ambos tipos de neuronas.

Con la distribución neuronal descrita, el neocognitrón es capaz de reconocer, después de una etapa de entrenamiento, números cursivos incluso en el caso de que se encuentren distorsionados.

Para mejorar la respuesta del neocognitrón, en aquellos casos en los que se le presentan varios patrones simultáneamente, Fukushima añadió al sistema conexiones retroalimentadas. El modelo mejorado es capaz de discernir entre los diversos patrones además de tener la facultad de restaurar aquellos patrones que se encuentren incompletos y de eliminar el ruido que acompañe a la imagen.

En la actualidad existen múltiples modelos de *redes neuronales*; cada uno de ellos enfocado a la resolución de un tipo específico de problemas. Generalmente los modelos propuestos son redes multicapa con neuronas adaptativas y retroalimentación. En los capítulos siguientes expondremos un modelo con dichas características y discutiremos los valores experimentales obtenidos.

## REFERENCIAS

- 1.-Arbib M.A., "Brains, Machines and Mathematics", Springer-Verlag, 2nd. Edition, 1987.
- 2.-Hopfield J.J. and Tank D.W., "Computing with Neural Circuits: A Model", Science, Vol. 233, Agosto de 1986, pp. 625-633.
- 3.-Farhat N.H., Psaltis D., Prata A. and Pack E., "Optical Implementation of Hopfield Model", Optical Society of America, 1985, pp. 653-656.
- 4.-Fukushima K., "A Neural Network for Visual Pattern Recognition", Computer, Marzo 1988, pp. 65-75.
- 5.-Lippmann R.P., "An Introduction to Computing with Neural Nets", IEEE ASSP Magazine, Vol. 3, No. 4, Abril 1987, pp. 4-22.

### CAPITULO III

En la introducción se describió una *neurona* sencilla que no era capaz de automodificarse. Ahora bien, para que la *neurona* pueda aprender es necesario que pueda cambiar su vector de pesos de acuerdo a las necesidades. Esto es, se necesitan *neuronas* adaptables para que sean inteligentes.

En la mayoría de los sistemas adaptables, esto es, sistemas con la capacidad de automodificarse en un proceso de retroalimentación de la señal de salida, el elemento básico es el "combinador lineal adaptable" (ALC) [1 y 2]. Este elemento, unido a un cuantificador, es capaz de simular el comportamiento de las *neuronas*. Posee la propiedad de retener información en un proceso de aprendizaje para utilizarla posteriormente cuando se le presenta una situación desconocida.

El ALC forma parte tanto de los filtros adaptables [2] como de las *neuronas* adaptables. Los filtros adaptables han tenido gran éxito en el procesamiento de señales; tal es el caso de las llamadas telefónicas de larga distancia y en las comunicaciones via satélite, donde se utilizan supresores de eco adaptables, permitiendo la comunicación simultánea en ambos sentidos [3].

A continuación vamos a explicar el funcionamiento de las *neuronas* adaptables.

#### 3.1-NEURONAS ADAPTABLES.

En la figura 3-1 podemos observar el esquema básico de una *neurona* adaptable. La salida de la misma es la combinación lineal de las diversas señales de entrada, y se obtiene realizando el producto entre el vector de pesos y el vector formado por las señales de

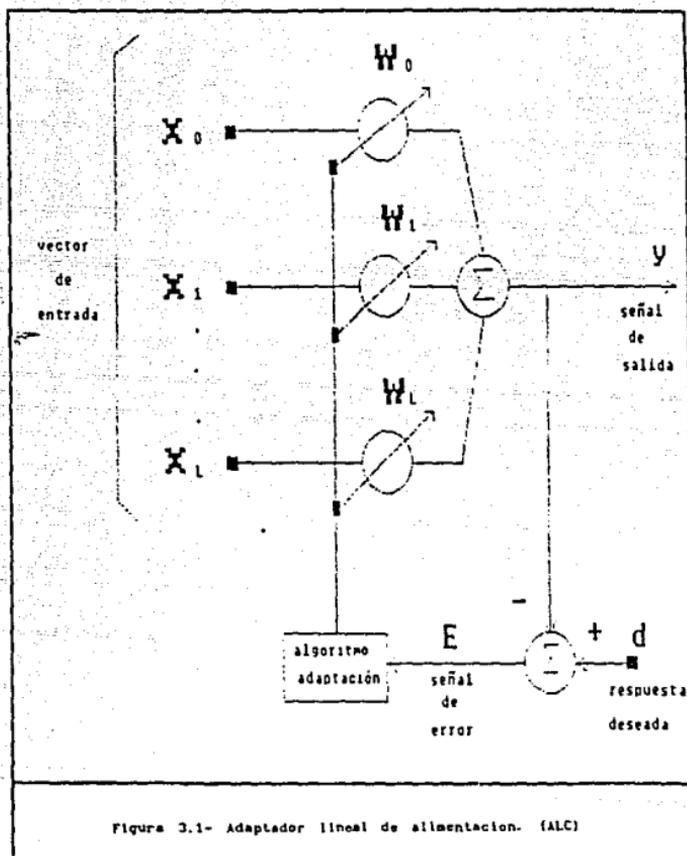


Figura 3.1- Adaptador lineal de alimentación. (ALC)

entrada. Este resultado se compara con la señal de salida deseada y la diferencia entre ambos es la señal de error. Para ajustar los los valores del vector de pesos, se utiliza el algoritmo de mínimos cuadrados (LMS) de Winter-Hoff [1].

Cuando la señal de salida de la neurona se pasa por un cuantificador se obtiene una señal binaria. Al conjunto formado por el cuantificador y el ALC se le denomina "adaline", siendo éste el elemento básico de la red neuronal (Figura 3-2). Los pesos en el adaline semejan las sinapsis cerebrales; el vector de entrada actúa como las entradas dendríticas y la salida binaria es similar a la salida axonal [4,6].

El adaline puede ser "entrenado" para que con un cierto vector de entrada se obtenga el vector de salida deseado (matriz compuesta por +1 y -1 combinados en forma tal que representen al patrón que la red está aprendiendo).

Con un vector de entrada de componentes binarias y una salida binaria, el adaline es capaz de realizar ciertas funciones lógicas, con las distintas combinaciones de los valores posibles de los pesos [1,5]. Para este tipo de aplicaciones es necesario dejar fijo uno de los pesos ( $w_0$ ) y la entrada correspondiente ( $x_0$ ) hacerla igual a +1 (Figura 3-3).

Una neurona aislada sólo puede realizar aquellas funciones que sean linealmente independientes. Para efectuar funciones separables no lineales, es necesario utilizar una red de neuronas. Las primeras redes tipo Madaline (múltiples adalines), constaban de un vector de entradas múltiples y dos capas de neuronas. La primera capa de neuronas era adaptable mientras que la otra era fija. Con esta configuración es posible implantar las funciones lógicas AND, OR y MAY (mayoría) [1].

Actualmente las redes constan de muchas capas, todas ellas adaptables. Para las neuronas de la última capa la adaptación es sencilla ya que la señal de salida deseada es conocida, pero para las neuronas de las capas internas la señal deseada de salida es

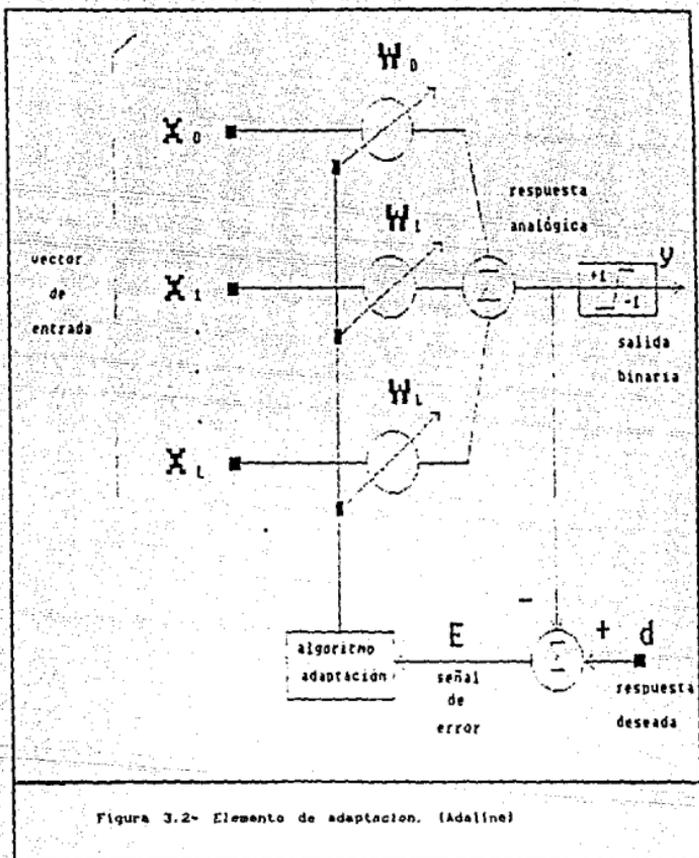


Figura 3.2- Elemento de adaptación. (Adaline)

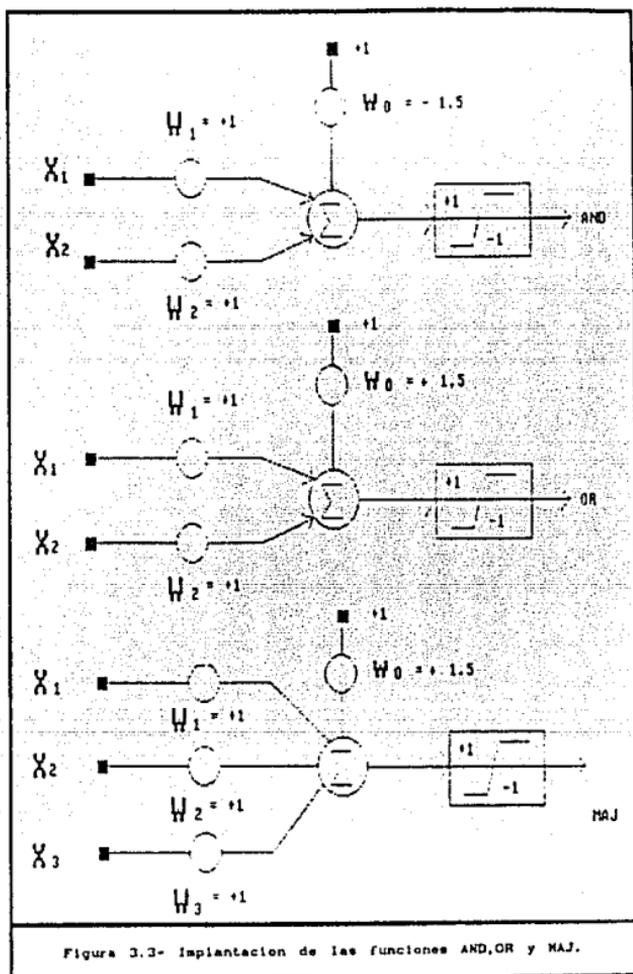


Figura 3.3- Implantación de las funciones AND, OR y MAJ.

desconocida. Para solventar este problema Winter y Widrow [1] elaboraron un algoritmo basándose en la primitiva regla de adaptación del Madaline. El algoritmo establece que la red debe ser perturbada lo menos posible con cada nuevo patrón de entrenamiento. De esta forma la red es capaz de retener lo aprendido con patrones anteriores.

### 3.2-ALGORITMO LMS.

Para adaptar los pesos de una neurona aislada se emplea el algoritmo de mínimos cuadrados LMS. Dicho algoritmo se puede denotar matemáticamente como sigue:

$$W_{k+1} = W_k + \alpha c_k X_k / |X_k|^2 \quad (1)$$

El vector de pesos en el ciclo de adaptación  $k+1$  es igual al del ciclo  $k$  mas una constante  $\alpha$  (que controla la estabilidad y velocidad de convergencia de la red) multiplicada por el error  $c_k$  definido como la diferencia entre la señal deseada y la señal analógica de salida de la neurona en el ciclo  $k$ ; y por el cociente  $X_k / |X_k|$ , donde  $X_k$  es el vector de entrada. Por otra parte, sabemos que el error  $c_k$  es la diferencia entre la señal de salida deseada  $d_k$  y la señal analógica de salida  $X_k^T W_k$  [6]; por lo tanto

$$c_k = d_k - X_k^T W_k. \quad (2)$$

En cada ciclo de adaptación los pesos toman nuevos valores y el error se va reduciendo

$$\Delta c_k = \Delta(d_k - X_k^T W_k) = -X_k^T \Delta W_k \quad (3)$$

donde  $\Delta W_k$  es:

$$\Delta W_k = W_{k+1} - W_k = \alpha c_k X_k / |X_k|^2 \quad (4)$$

Sustituyendo esta expresión en la ecuación (3), obtenemos

$$\Delta c_k = -\alpha c_k \quad (5)$$

Experimentalmente se ha visto que se debe cumplir que  $1.0 > \alpha > 0.1$  para que el sistema converja y se estabilice [1]. Estos datos reportados por Winter-Widrow no coinciden con los que se obtuvieron al

implantar la red y serán estudiados posteriormente.

### 3.3-REGLA MR11.

El proceso de adaptación detallado anteriormente es para una sola neurona, pero la red consta de tres capas, cada una de ellas con un número prefijado de neuronas. El proceso de adaptación para el conjunto debe ser congruente con el principio de mínima perturbación y para conseguirlo se emplea la regla MR11, que se detalla a continuación.

Cuando se le presenta a la red el primer patrón de entrenamiento, se tiene que realizar un proceso de adaptación. En primer lugar, se toma la neurona de la primera capa cuyo valor analógico de salida este más próximo a cero en valor absoluto. Se fuerza a dicha neurona a invertir su señal de salida binaria y, con este cambio, se hace un recorrido de la red comparando las salidas de la última capa con las salidas deseadas (patrón que queremos que aprenda). Si el cambio repercutió positivamente en la salida, se recalculan los pesos de la neurona afectada para que el cambio se haga efectivo. Ahora bien, si el cambio no produjo ninguna mejora en la salida, entonces se mantienen los pesos inalterados y se repite el proceso para la siguiente neurona cuya salida analógica esté más próxima a cero.

El proceso se repite para las neuronas aisladas y luego para las neuronas en pares, tercias, etc. hasta un límite cualquiera preestablecido. El proceso se aplica a la primera capa de neuronas y después a las subsiguientes. La última capa se adapta directamente en base a la salida deseada. Acto seguido, se le presenta a la red otro patrón y se vuelve a aplicar la regla MR11.

## REFERENCIAS

- 1.-Widrow B. y Winter R., "Neural Nets for Adaptive Filtering and Adaptive Pattern Recognition", Computer, Marzo 1988, pp. 25-39.
- 2.-Widrow B., "Generalization and Information Storage in Networks of Adaline Neurons", Self-Organizing Systems 1962, M.C. Yovitz, G.T. Jacobi and Goldstein eds., Spartan Books, Washington DC, pp. 435-461.
- 3.-Kailath T., "Lectures on Wiener and Kalman Filtering", Springer Verlag, New York, 1981.
- 4.-Hubel D. y Wiesel T., "Brain Mechanisms of Vision", Scientific American, Vol 241, Sept. 1979, pp. 130-145.
- 5.-Lewis P.M. y Coates C.L., "Treshold Logic", John Wiley and Sons, New York, 1967.
- 6.-Fukushima K., Miyake S. y Ito T., "Neocognitron: a neural network model for a mechanism of visual pattern recognition", IEEE Transactions on Systems, Man and Cibernetics SMC-13, pp. 826-834.

## CAPITULO IV

### 4.1-IMPLANTACION DE LA RED.

#### 4.1.1-ESTRUCTURA.

La red se dividió en tres capas de neuronas adaptables, contando cada capa con un número igual de neuronas. Las neuronas de la primera capa reciben la información del vector de entrada, posteriormente cada neurona transmite su señal de salida binaria como entrada a cada una de las neuronas de la capa siguiente (capa dos). Las neuronas de la segunda capa actúan de forma análoga a las neuronas de la capa uno, transmitiendo su salida a la capa tercera y última. La capa tres emite el vector de salida de la red. En esta red cada neurona se encuentra conectada con todas las neuronas de la capa siguiente dando lugar a un flujo de información "feed-forward" (Figura 4-1). En el proceso de aprendizaje la red consta de retroalimentación ó alimentación "feed-backward". Esto es, a la neurona se le proporciona la salida deseada, para que en base a ésta pueda modificar su vector de pesos para que su salida coincida con la deseada.

En una red como la propuesta aquí, la sensibilidad de las capas de neuronas aumenta gradualmente, haciéndose máxima en la última capa. Las capas internas u ocultas de la red actúan, en cierta medida, como codificadoras de información, mientras que la última capa hace el papel de decodificador.

Como patrón de entrada se le dio a la red de 49 neuronas/capa una matriz de  $7 \times 7$  (49 pixels), a la red de 100 neuronas/capa una matriz de  $10 \times 10$ , a la red de 144 neuronas/capa una matriz de  $12 \times 12$  y por último a la red de 225 neuronas/capa una matriz de  $15 \times 15$ . La matriz de entrada, en todos los casos, consta de ceros y unos correspondiendo los unos a los pixels con la información del carácter y los ceros al fondo de la imagen. La red ofrece como salida de la misma un vector con el mismo número de componentes que el vector de entrada (patrón de entrada).

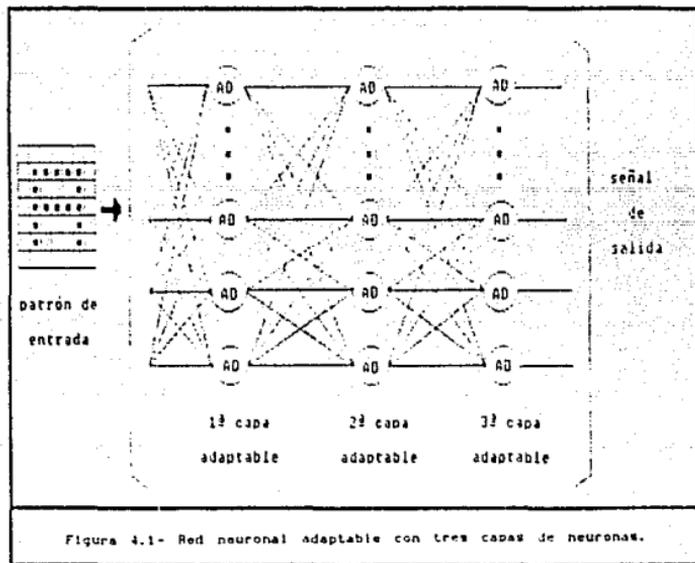
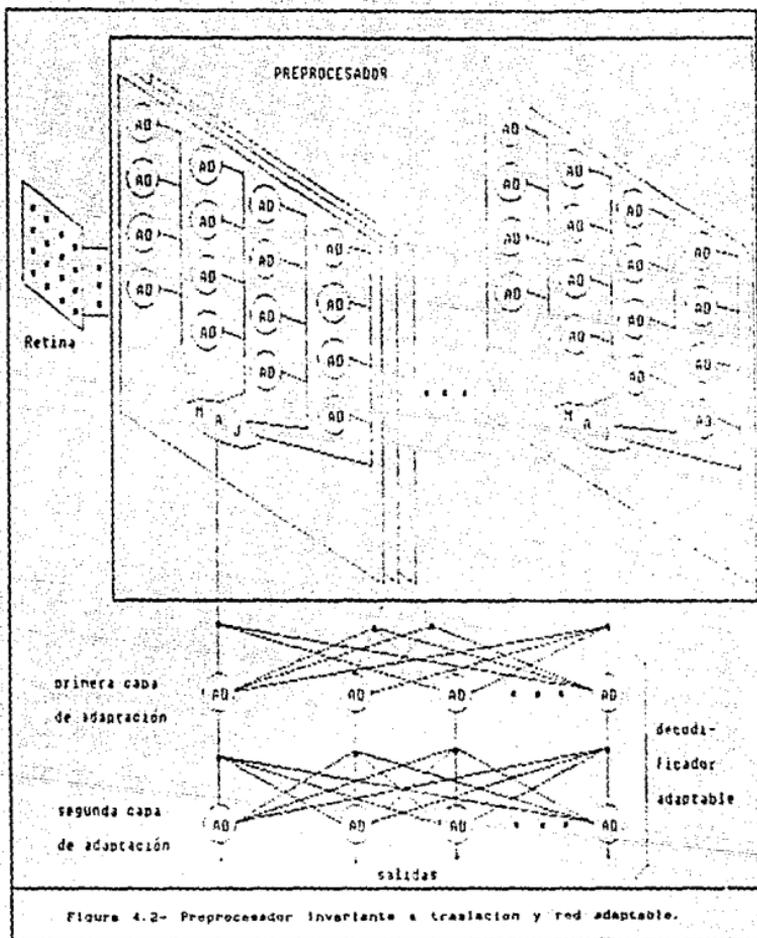


Figura 4.1- Red neuronal adaptable con tres capas de neuronas.



El comportamiento de la red varia en función del patrón de entrada, de la selección de parámetros y del tamaño de la misma.

#### 4.1.2-INMUNIDAD ANTE TRASLACION, ROTACION .

Para conseguir que el comportamiento, respuesta de la red sea invariante frente a la rotación y traslación, Winter y Widrow estiman necesario anteponer al cuerpo de la red un procesador formado por varias capas de *neuronas* que actúen como codificador de la información de entrada (Figura 4-2). En nuestro caso, el problema se solventó enseñándole a la red una serie completa del patrón rotado y trasladado dentro de la matriz. Se optó por esta solución para facilitar el uso de la red en sistemas de cómputo pequeños, ya que por cada capa adicional de *neuronas* es necesario guardar tres matrices, una para pesos, otra para valores analógicos y la última para valores binarios. En la etapa de *aprendizaje* es necesario utilizar todas las matrices simultáneamente. Esto origina problemas en cuanto a capacidad de memoria.

#### 4.1.3-IMPLANTACION.

Para la red de 49*neuronas*/capa se dio como matrices de entrada los siguientes veinte patrones:

0000000	0000000	0000000	0000000	0100010	0100010	0000000
0111110	0111110	0111110	0100010	0111110	0000000	1001111
0100010	0101000	0001010	0100010	0100010	0000000	0001010
0111110	0101000	0001010	0111110	0111110	0111110	0001010
0100010	0101000	0001010	0100010	0000000	0100010	0001010
0100010	0111110	0111110	0111110	0000000	0111110	1001111
0000000	0000000	0000000	0000000	0100010	0100010	0000000
pat-1	pat-2	pat-3	pat-4	pat-5	pat-6	pat-7

0000000	0000000	0000000	0000000	0000000	0000000	0000000
1111001	0111110	0111110	0111100	0111110	0111110	0111110
0101000	0100010	0100000	0100010	0100000	0100000	0100000
0101000	0111100	0100000	0100010	0111000	0111000	0100110
0101000	0100010	0100000	0100010	0100000	0100000	0100010
1111001	0111110	0111110	0111100	0111110	0100000	0111110
0000000	0000000	0000000	0000000	0000000	0000000	0000000

pat-8	pat-9	pat-10	pat-11	pat-12	pat-13	pat-14
-------	-------	--------	--------	--------	--------	--------

0000000	0000000	1111111	1000000	1010000	1111111
0100010	0111110	1000001	0000000	0000000	1000001
0100010	0001000	1000001	0000000	0100000	1011101
0111110	0001000	1000001	0000000	0000000	1000001
0100010	0001000	1000001	0000000	0000010	1011101
0100010	0111110	1000001	0000000	0000000	1011101
0000000	0000000	1111111	0000000	0000000	1111111

pat-15	pat-16	pat-17	pat-18	pat-19	pat-20
--------	--------	--------	--------	--------	--------

Teniendo en cuenta la estructura de la red y los algoritmos necesarios para la etapa de aprendizaje se diseñó el programa que se detalla en el anexo 1. Los parámetros se fueron variando en las distintas pruebas ; los resultados se pueden ver en el capítulo V.

## CAPITULO V

### 5.1-ANALISIS DE PARAMETROS.

Una vez establecido el diseño estructural de la red en el capítulo IV, se hicieron una serie de pruebas experimentales para determinar el valor de los parámetros que estaban indeterminados.

Para los pesos iniciales se hicieron cinco pruebas distintas; la primera fue con pesos iniciales aleatorios entre 0 y 1. En este caso, y fijando el resto de los parámetros que posteriormente enunciaremos, se vió que la red convergia en la etapa de *aprendizaje*, siendo capaz de identificar en la etapa de reconocimiento al patron ya aprendido.

La segunda prueba se hizo con pesos aleatorios entre -1 y +1. La razón para tomar este intervalo fue el generar pesos inhibitorios asociados a los valores negativos. Aunque teóricamente parecía razonable tener dichos valores negativos, experimentalmente se vió que la red así configurada no era capaz de converger. Creemos que este hecho se debe a que la matriz de entrada consta de +1 y -1, donde -1 actúa como inibidor; se utilizaron valores negativos de entrada en lugar de ceros para que dichas entradas tuvieran incidencia en la salida de la *neurona*. Ahora bien, al multiplicar pesos inhibitorios (negativos) por valores inhibitorios de entrada (negativos) se tiene como resultado una salida excitatoria (positiva) distorsionando con ello el comportamiento de la red.

En cualquier caso, autores como Winter y Widrow [1] reportan resultados favorables utilizando pesos negativos.

Otro de los parámetros que inciden ampliamente en el comportamiento de la red es  $\alpha$ . Como ya se dijo anteriormente,  $\alpha$  influye directamente en la velocidad de convergencia y estabilidad de la red. Algunos autores [1] establecen un intervalo para  $\alpha$  entre 1.0 y 0.1. En

nuestro caso se vio experimentalmente que valores inferiores a 0.5 implicaban la no convergencia de la red.

Para la red que se está estudiando el valor idóneo es de 0.7. Por debajo de este valor la red tarda mucho en converger o puede no converger, y por arriba del mismo la red converge rápidamente pero su nivel de retención en el aprendizaje disminuye notablemente.

En el proceso de adaptación de cada *neurona*, se estableció un factor de tolerancia para el error de  $10^{-6}$  y un número máximo de iteraciones igual a  $10^3$ .

En la red con 49 *neuronas* por capa se hicieron diversas pruebas para valores de  $\alpha$  entre 0.1 y 0.9 ( gráfica 5-8.1 ). Como se puede ver en la gráfica los tiempos de convergencia de la red aumentan considerablemente para valores de  $\alpha$  por debajo de 0.7. Al aumentar el número de *neuronas* de la red los tiempos de convergencia aumentan de forma exponencial como se puede observar en la curva de la gráfica 5-8.2, donde para redes de 225 *neuronas* por capa los tiempos de convergencia para la red, siendo  $\alpha = 0.7$ , se encuentran alrededor de 1hr. 15 min en una Burroughs-A9. Para valores de  $\alpha$  por debajo de 0.7 los tiempos requeridos para la convergencia de la red son excesivamente altos y la mejora en la sensibilidad de la red es prácticamente inapreciable (del orden de  $10^{-5}$ ). Por esta razón se optó por trabajar con  $\alpha = 0.7$ . Valores superiores a 0.7 no mejoran los tiempos de convergencia como se puede apreciar en la curva ( gráfica 5-8.1 ) y sin embargo la red pierde algo de sensibilidad.

Una vez fijada la constante de convergencia en el valor óptimo de 0.7 se dió paso al estudio de otros parámetros.

## 5.2-INTERVALO DE PESOS.

En un principio se pensó que al ampliar el intervalo de pesos se podría conseguir una mayor sensibilidad en la red.

Se hicieron distintas pruebas para pesos iniciales comprendidos

entre 0-1, 0-5 y 0-10 (tablas 5-5, 5-6 y 5-7). En las gráficas 5-5.1 a la 5-7.3 podemos ver los pesos promedio y las correspondientes desviaciones estandar que se obtuvieron para las tres capas de las redes y los tres distintos intervalos de pesos. Podemos ver que la desviación es menor en las redes de mayor tamaño. En cuanto a las capas de la red se aprecia una disminución en la desviación a medida que pasamos de capa (menor para la capa 3).

Como se puede apreciar en las curvas (gráficas 5-1 a la 5-4), en la red de 49neuronas/capa la sensibilidad mejora apreciablemente al pasar de una escala de 0-1 a la escala de 0-5, la que difiere muy poco del intervalo de 0-10 (tablas 5-1 a la 5-4). En la gráfica 5-1 vemos como para un intervalo de 0-1 la red identifica solo tres patrones (4, 9 y 20) como distintos del aprendido (en las gráficas se expresa como patrón aprendido). En el momento de contrastar el resultado que se obtiene al recorrer la red con cada uno de los 20 distintos patrones, con ellos mismos (en las gráficas patrón existente), sólo un 15% de los patrones son identificados en menos del 50% mientras que el 70% de los patrones se identifican en más del 70%.

Estudiaremos a continuación un ejemplo para poder entender estas cifras. En nuestro caso el patrón aprendido va a ser el carácter "A". Cuando la red pasa la etapa de aprendizaje y se le presenta de nuevo el patrón "A", la red lo identifica en un 100%. Ahora bien, si a la red se le suministra la información de entrada correspondiente al patrón número dos ("A" rotada), la red de 49neuronas/capa ofrece una salida que solapa en un 100% con el patrón aprendido "A". Dicha salida comparada con el patrón de entrada nos da un nivel de identificación del 79.7%. En este caso la red no es capaz de discernir entre el carácter uno y el dos.

Al ampliar el intervalo de pesos a 0-5, con la misma red, vemos que el número de patrones que la red es capaz de distinguir como distintos del aprendido aumenta en un 10%, pasando a ser el 25% del total de los patrones. En este caso la red ha aumentado considerablemente su sensibilidad. Al aumentar nuevamente el intervalo de pesos a 0-10, la sensibilidad de la red casi no mejora. El número de

patrones que la red es capaz de discernir es el mismo que para el intervalo de 0-5 y sólo se aprecia una mejoría mínima en el porcentaje de identificación de los patrones distintos al aprendido.

Estas diferencias, que son evidentes para la red de 49neuronas/capa, se van haciendo cada vez menores para redes con mayor número de neuronas. En la red de 100neuronas/capa todavía se aprecian cambios al pasar de un intervalo de 0-1 a otro de 0-5, pero para las redes de 144neuronas/capa y 225neuronas/capa los cambios son prácticamente inapreciables (graficas y tablas 5-1 a la 5-4 ).

De estos estudios podemos concluir que la variación en el intervalo de pesos influye más en redes pequeñas. La red mejora su comportamiento al aumentar el intervalo, tendiéndose a un equilibrio después de un cierto aumento del intervalo. Como se ha visto experimentalmente, no se mejora casi nada el comportamiento de la red en el cambio de intervalo de 0-5 al de 0-10.

### 5.3-REDES CON DISTINTAS CONFIGURACIONES.

En un principio se pensó que si la red era capaz de procesar más información de entrada, el resultado final sería mejor, esto es, la red aumentaría su sensibilidad. Por esta razón, se realizaron experimentos con redes de 49neuronas/capa, 100neuronas/capa, 144neuronas/capa y 225neuronas/capa.

En contra de lo esperado los resultados experimentales indican que la red más sensible es la de 49neuronas/capa, empleándose un intervalo de 0-5 para los pesos iniciales. Esta red es la que mejor discierne entre los patrones distintos del aprendido. Ahora bien, al aumentar el tamaño de la red los patrones que no le habían sido enseñados a la red son reconocidos como ellos mismos en un mayor porcentaje. Esto es una "B" se reconoce como "B" en un 44% en la red de 49neuronas/capa; mientras que la red de 225neuronas/capa identifica a la "B" en un 56%. Estos porcentajes no se pueden considerar de forma absoluta ya que el patrón no guarda una escala rigurosa al pasar de la matriz de 7x7 a la de 15x15. Los resultados con las distintas redes se pueden apreciar en

las gráficas y tablas 5-1 a la 5-4 .

RED NEURONAL  
49 NEURONAS POR CAPA  
RECONOCIMIENTO PORCENTUAL

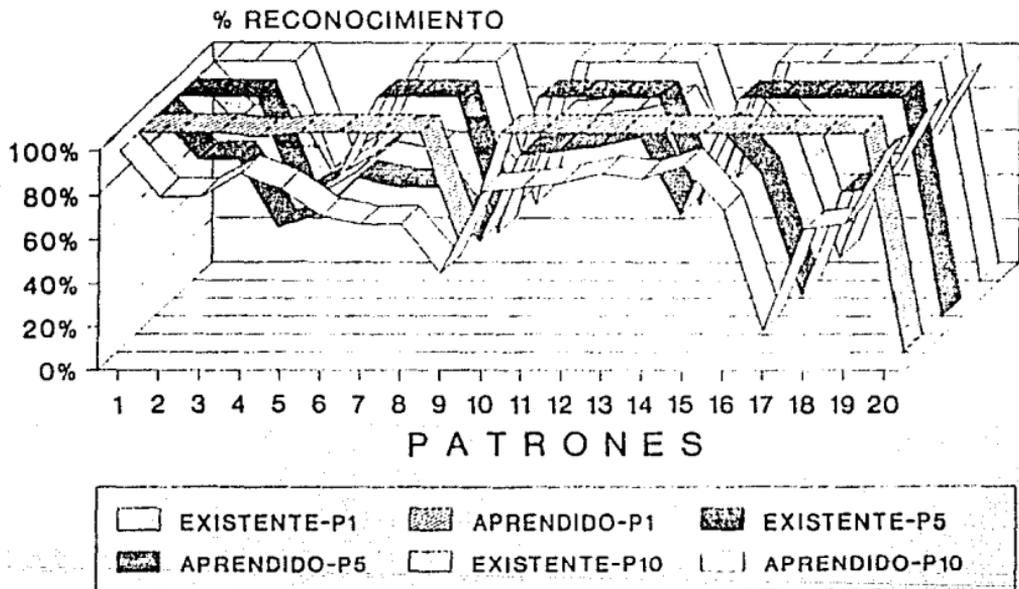
TABLA 5-1

	RANGO PESOS #1		RANGO PESOS #2		RANGO PESOS #3	
	EXISTENTE	APRENDIDO	EXISTENTE	APRENDIDO	EXISTENTE	APRENDIDO
1	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%
2	79.60%	100.00%	79.50%	100.00%	79.50%	100.00%
3	79.60%	100.00%	79.50%	100.00%	79.50%	100.00%
4	89.90%	97.90%	48.90%	53.00%	46.90%	51.00%
5	83.00%	100.00%	55.10%	63.20%	57.10%	61.20%
P 6	71.00%	100.00%	71.40%	100.00%	71.40%	100.00%
A 7	67.30%	100.00%	67.30%	100.00%	67.30%	100.00%
T 8	67.30%	100.00%	67.30%	100.00%	67.30%	100.00%
R 9	44.90%	48.90%	42.80%	38.70%	42.80%	34.60%
O 10	81.60%	100.00%	81.60%	100.00%	81.60%	100.00%
N 11	83.70%	100.00%	83.60%	100.00%	83.60%	100.00%
E 12	83.70%	100.00%	85.70%	100.00%	85.70%	100.00%
S 13	89.80%	100.00%	89.70%	100.00%	89.70%	100.00%
14	87.80%	100.00%	55.10%	51.00%	51.00%	46.90%
15	93.90%	100.00%	93.80%	100.00%	93.80%	100.00%
16	73.50%	100.00%	73.40%	100.00%	73.40%	100.00%
17	18.40%	100.00%	18.30%	100.00%	18.30%	100.00%
18	65.30%	100.00%	65.30%	100.00%	65.30%	100.00%
19	67.30%	100.00%	67.30%	100.00%	67.30%	100.00%
20	100.00%	.10%	100.00%	.00%	100.00%	.00%

RANGO PESOS #1 -> PESOS INICIALES DE CERO A UNO  
RANGO PESOS #2 -> PESOS INICIALES DE CERO A CINCO  
RANGO PESOS #3 -> PESOS INICIALES DE CERO A DIEZ

# RED NEURONAL

## 49 NEURONAS



GRAFICA 5-1

RED NEURONAL  
100 NEURONAS POR CAPA  
RECONOCIMIENTO PORCENTUAL

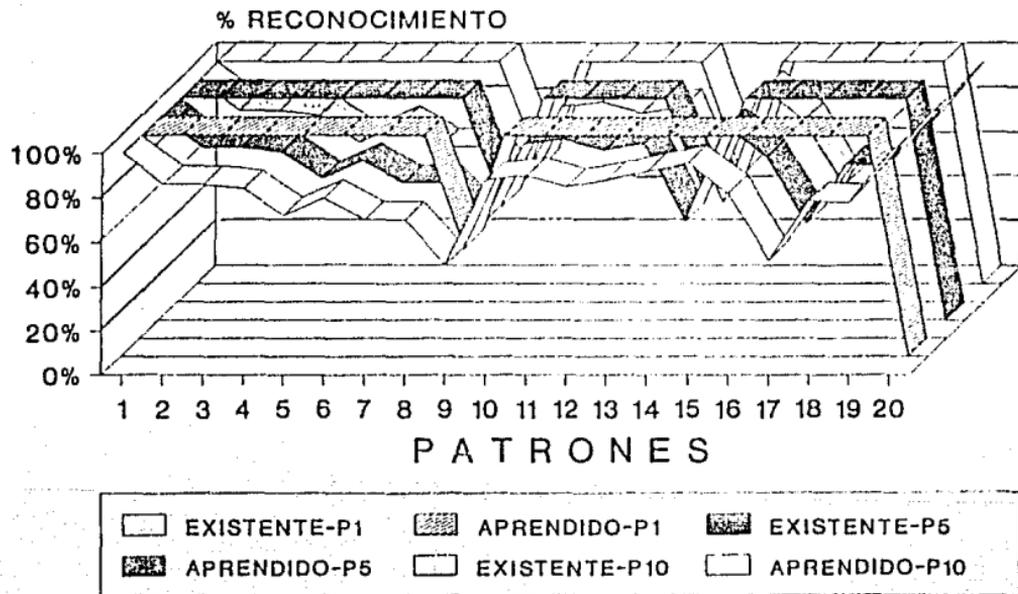
TABLA 5-2

	RANGO PESOS #1		RANGO PESOS #2		RANGO PESOS #3	
	EXISTENTE	APRENDIDO	EXISTENTE	APRENDIDO	EXISTENTE	APRENDIDO
1	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%
2	86.00%	100.00%	86.00%	100.00%	86.00%	100.00%
3	86.00%	100.00%	86.00%	100.00%	86.00%	100.00%
4	84.00%	100.00%	84.00%	100.00%	84.00%	100.00%
5	72.00%	100.00%	72.00%	100.00%	72.00%	100.00%
6	80.00%	100.00%	80.00%	100.00%	80.00%	100.00%
7	70.00%	100.00%	70.00%	100.00%	70.00%	100.00%
8	70.00%	100.00%	70.00%	100.00%	70.00%	100.00%
9	50.00%	49.00%	52.00%	51.00%	53.00%	52.00%
10	88.00%	100.00%	88.00%	100.00%	88.00%	100.00%
11	90.00%	100.00%	90.00%	100.00%	90.00%	100.00%
12	85.00%	100.00%	85.00%	100.00%	85.00%	100.00%
13	88.00%	100.00%	88.00%	100.00%	88.00%	100.00%
14	92.00%	100.00%	53.00%	57.00%	45.00%	49.00%
15	96.00%	100.00%	96.00%	100.00%	96.00%	100.00%
16	82.00%	100.00%	82.00%	100.00%	82.00%	100.00%
17	52.00%	100.00%	52.00%	100.00%	52.00%	100.00%
18	79.00%	100.00%	79.00%	100.00%	79.00%	100.00%
19	78.00%	100.00%	78.00%	100.00%	78.00%	100.00%
20	100.00%	.00%	100.00%	.00%	100.00%	.00%

RANGO PESOS #1 -> PESOS INICIALES DE CERO A UNO  
RANGO PESOS #2 -> PESOS INICIALES DE CERO A CINCO  
RANGO PESOS #3 -> PESOS INICIALES DE CERO A DIEZ

# RED NEURONAL

## 100 NEURONAS



GRAFICA 5-2

RED NEURONAL  
144 NEURONAS POR CAPA  
RECONOCIMIENTO PORCENTUAL

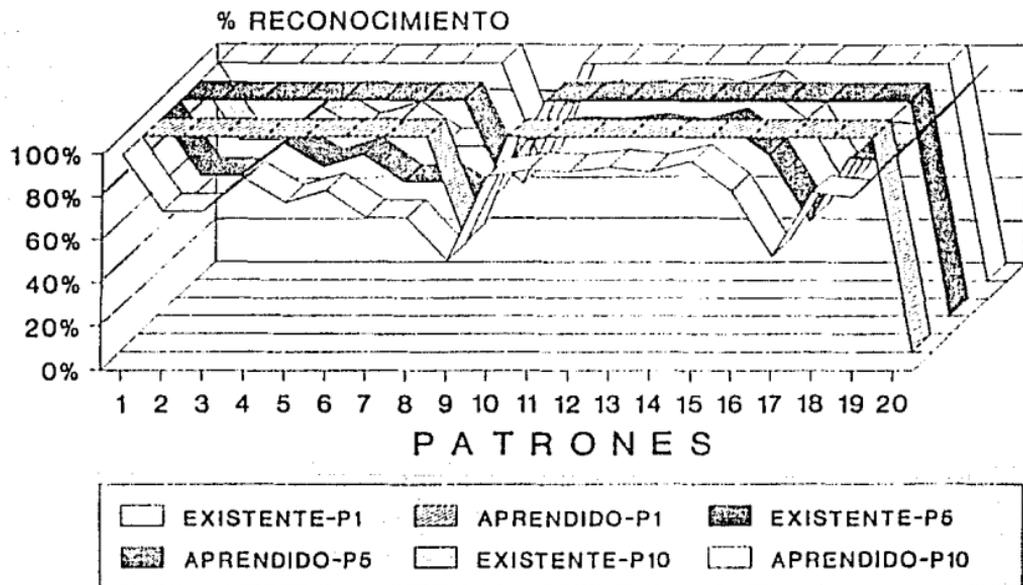
TABLA 5-3

	RANGO PESOS #1		RANGO PESOS #2		RANGO PESOS #3	
	EXISTENTE	APRENDIDO	EXISTENTE	APRENDIDO	EXISTENTE	APRENDIDO
1	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%
2	73.60%	100.00%	73.60%	100.00%	73.60%	100.00%
3	73.60%	100.00%	73.60%	100.00%	73.60%	100.00%
4	68.80%	100.00%	68.80%	100.00%	68.80%	100.00%
5	77.70%	100.00%	77.70%	100.00%	77.70%	100.00%
6	83.30%	100.00%	83.30%	100.00%	83.30%	100.00%
7	70.80%	100.00%	70.80%	100.00%	70.80%	100.00%
8	70.80%	100.00%	70.80%	100.00%	70.80%	100.00%
9	50.60%	51.30%	59.70%	61.80%	53.40%	54.10%
10	90.20%	100.00%	90.20%	100.00%	90.20%	100.00%
11	93.00%	100.00%	93.00%	100.00%	93.00%	100.00%
12	92.30%	100.00%	92.30%	100.00%	92.30%	100.00%
13	94.40%	100.00%	94.40%	100.00%	94.40%	100.00%
14	92.30%	100.00%	92.30%	100.00%	92.30%	100.00%
15	97.20%	100.00%	97.20%	100.00%	97.20%	100.00%
16	83.30%	100.00%	83.30%	100.00%	83.30%	100.00%
17	52.70%	100.00%	52.70%	100.00%	52.70%	100.00%
18	82.60%	100.00%	82.60%	100.00%	82.60%	100.00%
19	80.50%	100.00%	80.50%	100.00%	80.50%	100.00%
20	100.00%	.00%	100.00%	.00%	100.00%	.00%

RANGO PESOS #1 -> PESOS INICIALES DE CERO A UNO  
RANGO PESOS #2 -> PESOS INICIALES DE CERO A CINCO  
RANGO PESOS #3 -> PESOS INICIALES DE CERO A DIEZ

# RED NEURONAL

## 144 NEURONAS



GRAFICA 5-3

RED NEURONAL  
225 NEURONAS POR CAPA  
RECONOCIMIENTO PORCENTUAL

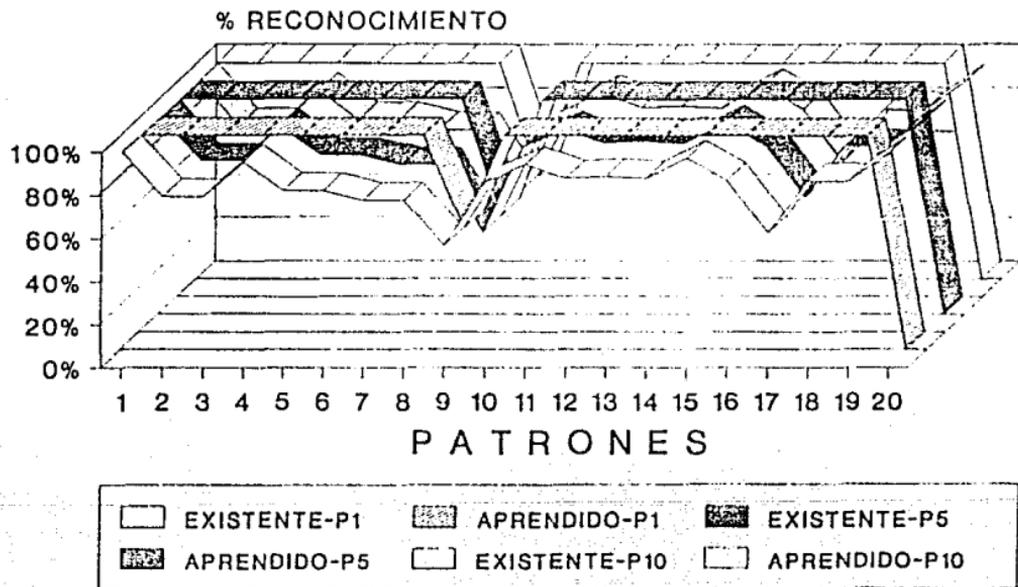
TABLA 5-4

	RANGO PESOS #1		RANGO PESOS #2		RANGO PESOS #3	
	EXISTENTE	APRENDIDO	EXISTENTE	APRENDIDO	EXISTENTE	APRENDIDO
1	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%
2	79.60%	100.00%	79.60%	100.00%	79.50%	100.00%
3	79.60%	100.00%	79.60%	100.00%	79.50%	100.00%
4	95.60%	100.00%	95.60%	100.00%	95.50%	100.00%
5	82.20%	100.00%	82.20%	100.00%	82.20%	100.00%
6	82.20%	100.00%	82.20%	100.00%	82.20%	100.00%
7	77.80%	100.00%	77.80%	100.00%	77.70%	100.00%
8	77.60%	100.00%	77.80%	100.00%	77.70%	100.00%
9	56.90%	56.40%	46.20%	45.80%	48.60%	47.50%
10	86.20%	100.00%	86.20%	100.00%	86.20%	100.00%
11	94.70%	100.00%	94.70%	100.00%	94.60%	100.00%
12	88.00%	100.00%	88.00%	100.00%	88.00%	100.00%
13	88.90%	100.00%	88.90%	100.00%	88.80%	100.00%
14	88.00%	100.00%	88.00%	100.00%	88.00%	100.00%
15	97.80%	100.00%	97.80%	100.00%	97.70%	100.00%
16	87.10%	100.00%	87.10%	100.00%	87.10%	100.00%
17	62.70%	100.00%	62.70%	100.00%	62.60%	100.00%
18	87.10%	100.00%	87.10%	100.00%	87.10%	100.00%
19	86.70%	100.00%	86.70%	100.00%	86.60%	100.00%
20	100.00%	.00%	100.00%	.00%	100.00%	.00%

RANGO PESOS #1 -> PESOS INICIALES DE CERO A UNO  
RANGO PESOS #2 -> PESOS INICIALES DE CERO A CINCO  
RANGO PESOS #3 -> PESOS INICIALES DE CERO A DIEZ

# RED NEURONAL

## 225 NEURONAS



GRAFICA 5-4

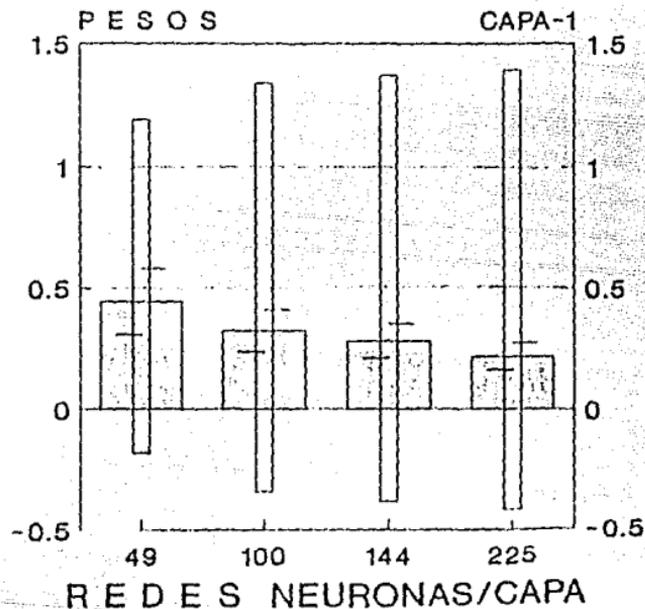
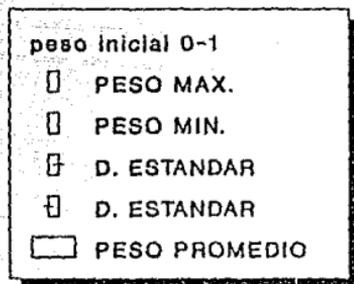
VALORES CARACTERISTICOS DE LOS PESOS  
REDES CON 49-100-144-225 NEURONAS POR CAPA

TABLA 5-5

	NEURONAS POR CAPA	PESO MAXIMO	PESO MINIMO	PESO PROMEDIO	DESVIACION ESTANDAR	
C	1	49	1.191	-.181	.445	0.445±0.135
	1	100	1.339	-.337	.325	0.335±0.085
	1	144	1.370	-.360	.283	0.283±0.07
	1	225	1.393	-.413	.221	0.221±0.056
P	2	49	.594	-.497	.020	0.02±0.14
	2	100	.596	-.528	.010	0.01±0.099
	2	144	.597	-.531	.007	0.007±0.083
A	2	225	.545	-.546	.004	0.004±0.066
S	3	49	.563	-.717	.007	0.007±0.142
	3	100	.522	-.722	.006	0.006±0.099
	3	144	.548	-.670	.005	0.005±0.083
	3	225	.515	-.539	.003	0.003±0.066

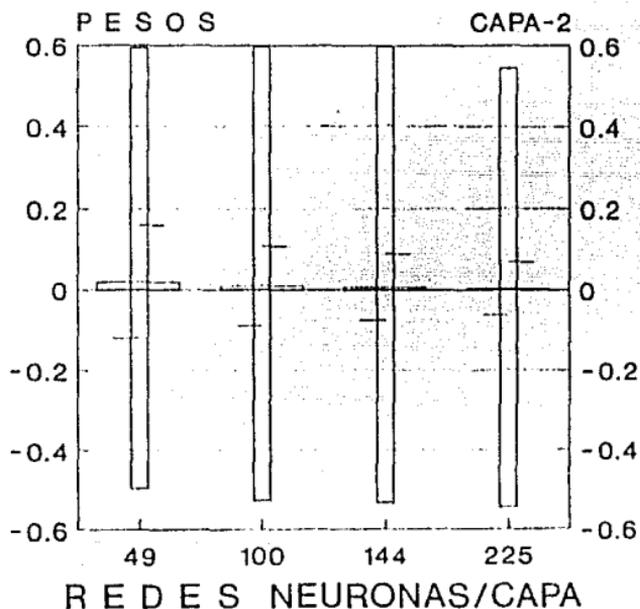
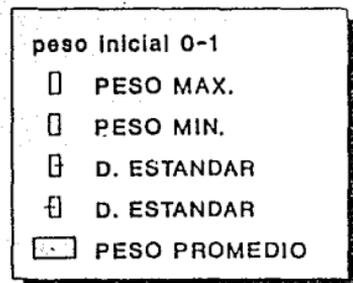
RANGO DE PESOS INICIALES DE CERO A UNO

# VALORES CARACTERISTICOS DE LOS PESOS



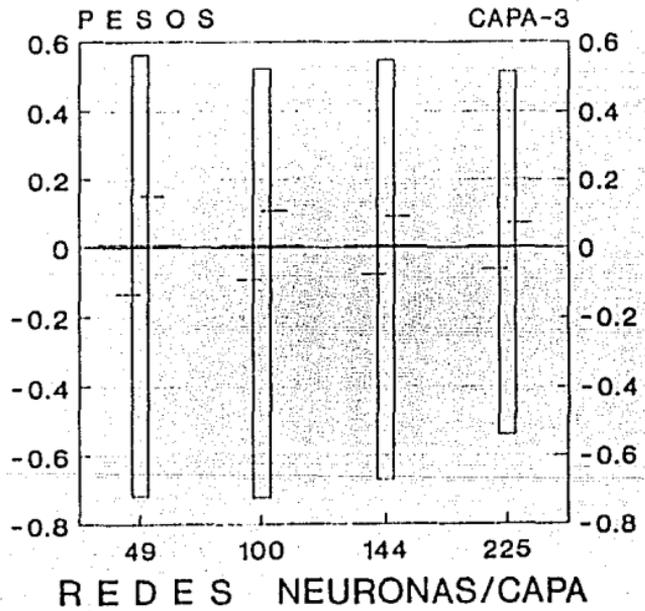
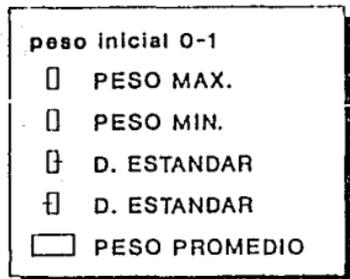
GRAFICA 5-6.1

# VALORES CARACTERISTICOS DE LOS PESOS



GRAFICA 5-5.2

# VALORES CARACTERISTICOS DE LOS PESOS



GRAFICA 5-5.3

45

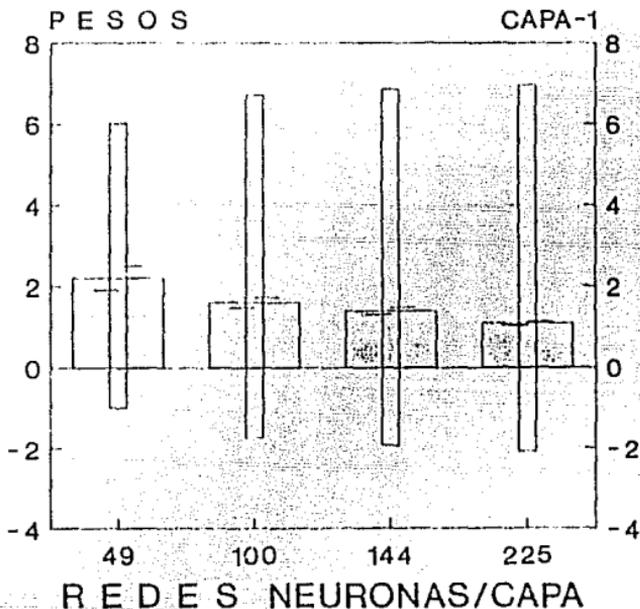
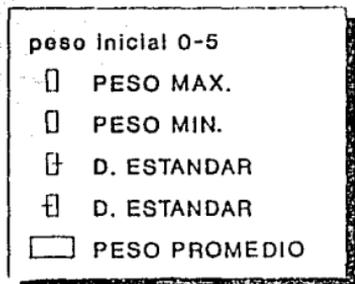
VALORES CARACTERISTICOS DE LOS PESOS  
REDES CON 49-100-144-225 NEURONAS POR CAPA

TABLA 5-6

	NEURONAS POR CAPA	PESO MAXIMO	PESO MINIMO	PESO PROMEDIO	DESVIACION ESTANDAR	
C	1	49	6.038	-.987	2.198	2.198±0.296
	1	100	6.733	-1.725	1.599	1.599±0.128
	1	144	6.877	-1.930	1.394	1.394±0.087
	1	225	6.982	-2.065	1.093	1.093±0.05
P	2	49	2.890	-2.568	.020	0.020±0.14
	2	100	2.942	-2.681	.010	0.010±0.099
	2	144	2.959	-2.696	.007	0.007±0.083
R	2	225	2.706	-2.747	.004	0.004±0.066
	S	3	49	2.736	-3.670	.007
3		100	2.572	-3.651	.006	0.006±0.099
3		144	2.710	-3.382	.005	0.005±0.083
3		225	2.561	-2.713	.003	0.003±0.066

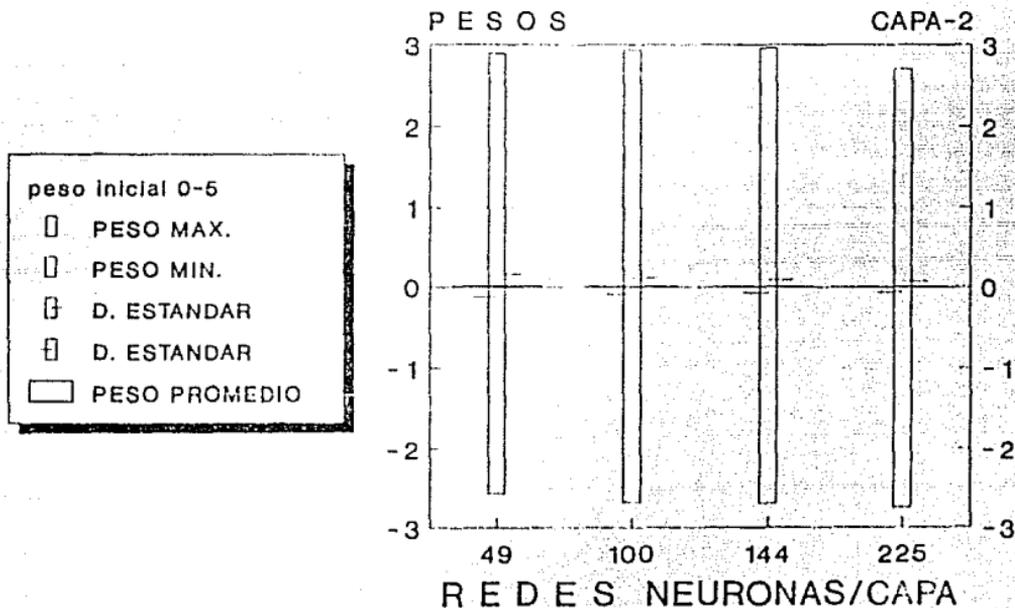
RANGO DE PESOS INICIALES DE CERO A CINCO

# VALORES CARACTERISTICOS DE LOS PESOS



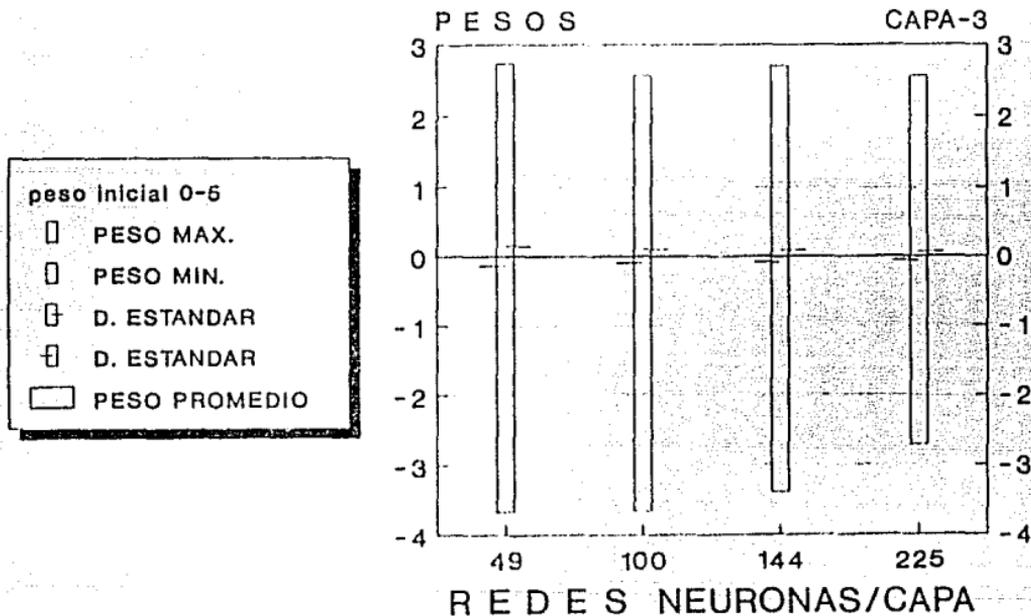
GRAFICA 5-6.1

# VALORES CARACTERISTICOS DE LOS PESOS



GRAFICA 5-6.2

# VALORES CARACTERISTICOS DE LOS PESOS



GRAFICA 5-6.3

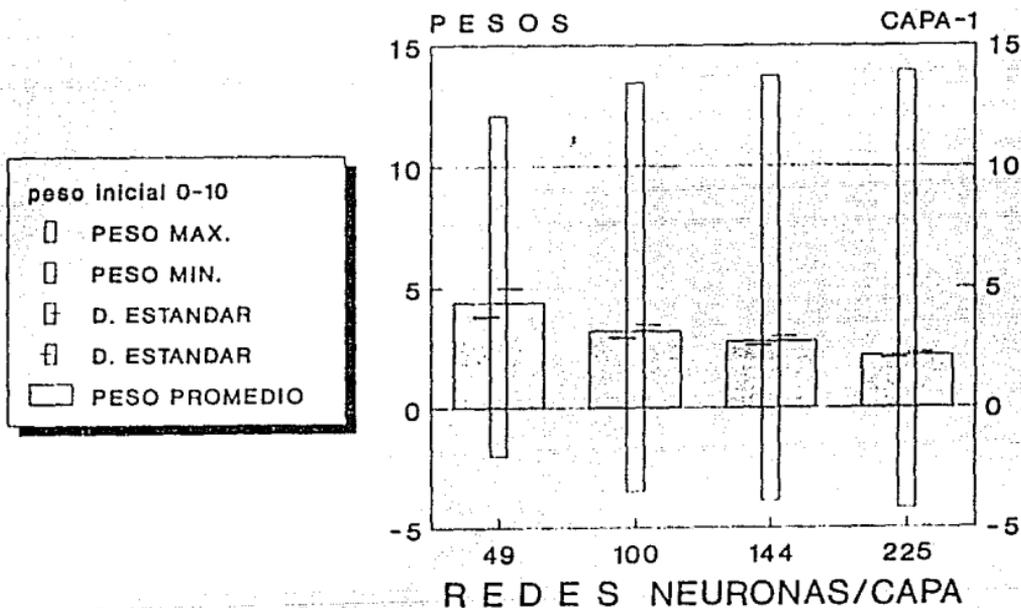
VALORES CARACTERISTICOS DE LOS PESOS  
REDES CON 49-100-144-225 NEURONAS POR CAPA

TABLA 5-7

	NEURONAS POR CAPA	PESO MAXIMO	PESO MINIMO	PESO PROMEDIO	DESVIACION ESTANDAR
C	49	12.095	-1.994	4.390	4.390±0.593
	100	13.475	-3.460	3.191	3.191±0.271
	144	13.761	-3.066	2.784	2.784±0.187
	225	13.969	-4.175	2.182	2.182±0.105
A	49	5.760	-5.157	.020	0.020±0.14
	100	5.873	-5.372	.010	0.010±0.099
	144	5.911	-5.400	.007	0.007±0.083
	225	5.408	-5.499	.004	0.004±0.066
S	49	5.452	-7.361	.007	0.007±0.142
	100	5.133	-7.311	.006	0.006±0.099
	144	5.414	-6.771	.005	0.005±0.083
	225	5.127	-5.430	.003	0.003±0.066

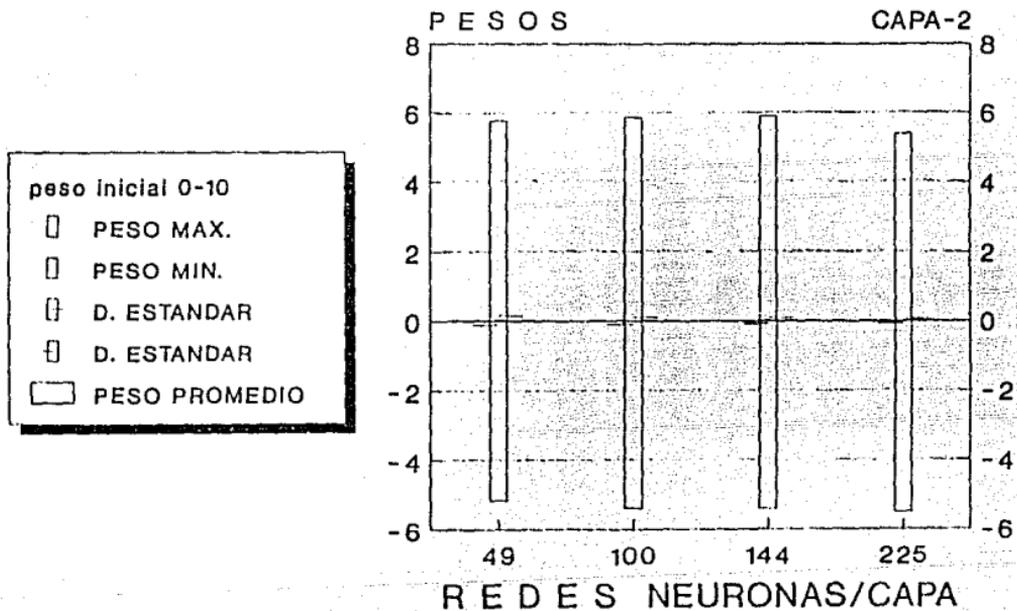
RANGO DE PESOS INICIALES DE CERO A DIEZ

# VALORES CARACTERISTICOS DE LOS PESOS



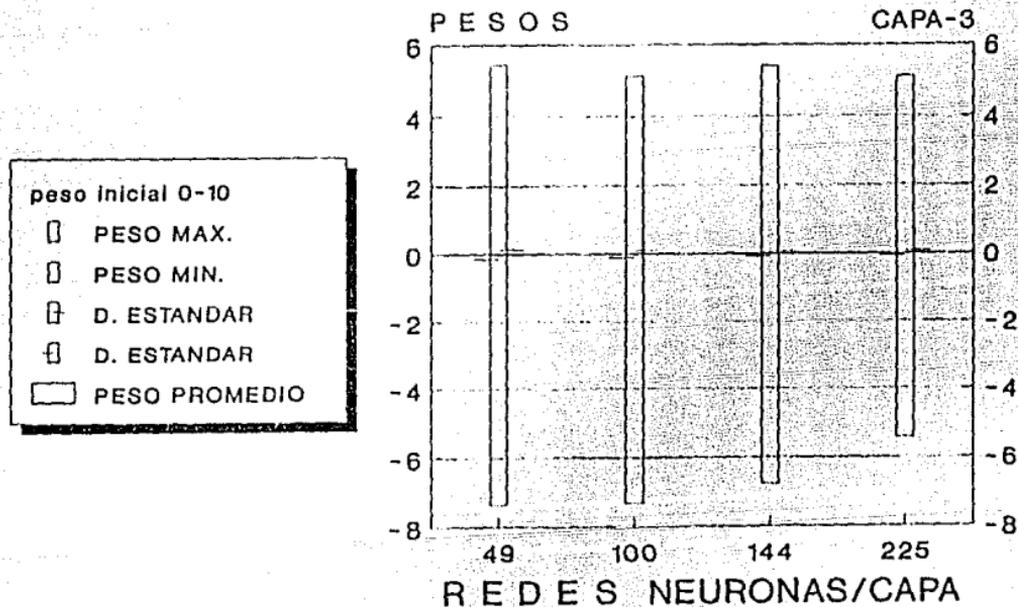
GRAFICA 5-7.1

# VALORES CARACTERISTICOS DE LOS PESOS



GRAFICA 5-7.2

# VALORES CARACTERISTICOS DE LOS PESOS



GRAFICA 5-7.3

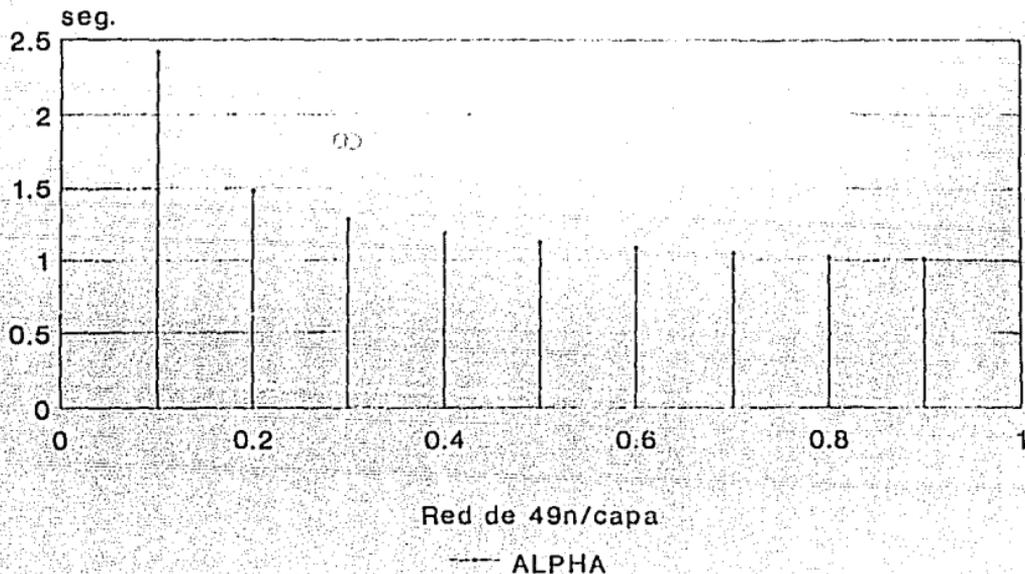
TABLA 5-8

RED NEURONAL  
TIEMPOS DE CONVERGENCIA  
EN UNA BURROUGHS-A9

		ALFA	TIEMPO DE CONVERGENCIA (MIN)
# D E N E U R O N A S	49	.1	2.418
	49	.2	1.477
	49	.3	1.284
	49	.4	1.185
	49	.5	1.122
	49	.6	1.085
	49	.7	1.045
	49	.8	1.021
	49	.9	1.004
	49	.7	1.045
	100	.7	7.057
	144	.7	20.353
	225	.7	75.191

# RED NEURONAL

## TIEMPOS DE CONVERGENCIA

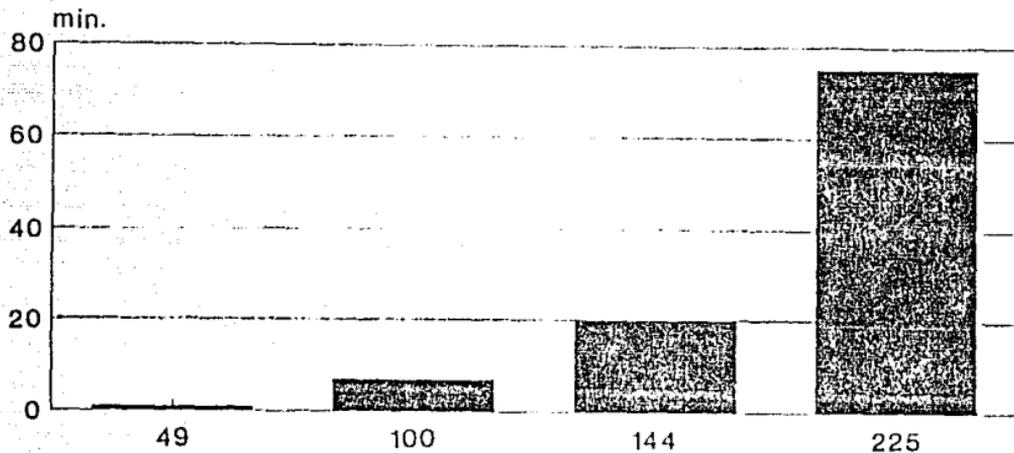


55

GRAFICA 5-8.1

# RED NEURONAL

## TIEMPOS DE CONVERGENCIA



ALPHA = 0.7

n./capa

GRAFICA 5-8.2

## REFERENCIAS

- 1.-Widrow B. y Winter R., "Neural Nets for Adaptive Filtering and Adaptive Pattern Recognition", Computer, Marzo 1988, pp. 25-39.

## CAPITULO VI

### 6.1-COMPARACION DE LOS RESULTADOS CON UN METODO ESTADISTICO DE RECONOCIMIENTO.

Ya que el análisis de la validez de los resultados obtenidos con las *redes neuronales* aquí propuestas, era casi imposible de efectuar al no tener ningún punto de referencia más que los intuitivos, se optó por realizar una comparación contra un método estadístico de correlación de los patrones. Aunque no se trata de hacer una comparación rigurosa si es bueno mostrar las similitudes y diferencias que presentan las *redes* frente a otros métodos de identificación de caracteres.

El método estadístico de comparación utilizado como referencia ha sido diseñado por el Dr. Angel Kuri Morales. En él se generan aleatoriamente 2500 muestras del patrón base (patrón de aprendizaje) rotadas y trasladadas. Una vez obtenidas las muestras se ajusta una curva mediante polinomios de Chebyshev. Por último, dado el intervalo de desviación estándar deseado y el grado máximo del polinomio que se quiere, se genera el polinomio de descriptores específico para cada patrón. Una vez obtenido el polinomio descriptor del patrón base se pueden efectuar las correlaciones entre dicho patrón y el resto de los patrones.

En la tabla-6.1 (gráfica 6.1) se pueden ver los resultados obtenidos al correlacionar cuatro patrones base con el resto de los patrones. Se tomaron estos patrones base por ser los que muestran mayores diferencias con los resultados obtenidos en las *redes neuronales* (gráfica 6.2).

En la gráfica 6.1, tenemos en el eje de ordenadas la serie completa de patrones que se le habían presentado previamente a la *red neuronal* de 100 neuronas por capa. En el eje de abscisas se encuentran los porcentajes de correlación, y cada una de las líneas en la gráfica

corresponde a cada uno de los patrones base. Como podemos apreciar con los patrones base "A" (patrón 1) y "G" (patrón 14) aproximadamente la mitad de los patrones se identifican casi en su totalidad con el patrón base, mientras que patrones radicalmente distintos como es el caso del patrón correspondiente a un sólo punto al ser tomados como patrón base presentan una correlación muy baja con el resto de los patrones.

En el caso de las *redes neuronales* la clasificación de los patrones no es tan marcada, aunque sí se da la misma tendencia para la serie de patrones aprendidos. Cuando los caracteres son distintos del aprendido las diferencias entre los sistemas de identificación son notables. Patrones como la "B" y la "G" que por el método estadístico presentan una correlación alta (del 100% en el caso de la "E"), son identificados por la red como muy distintos del patrón base (carácter "A"). El patrón con un sólo punto (patrón 18) que presenta una correlación muy baja con el patrón base ( 5% ) es identificado por la red como igual al patrón base. Por último el negativo del caracter base ( patrón 20 ) presenta una correlación arriba del 20% y la red lo interpreta como totalmente distinto al aprendido (reconocimiento del 0% ). En este caso la red identifica el caracter inverso como si fuera el base, esto es, como si se le hubiera enseñado en la etapa de aprendizaje.

Los resultados obtenidos nos llevan a creer que en el caso de las redes no sólo es importante el carácter si no también su relación con el resto de la imagen. Esto es, el fondo de la imagen también esta contribuyendo de forma activa en el tratamiento de la imagen. En el estudio estadístico el fondo no es tomado en cuenta y por lo tanto al darse variaciones en la relación fondo/imagen el comportamiento de la red tiende a diferir del estadístico.

## 6.2-CONCLUSIONES.

Tras haber visto los resultados comparativos con el método estadístico podemos concluir que el grado de reconocimiento de la red no guarda una relación directa con el grado de correlación entre los

patrones. Esto se debe en gran medida al peso excesivo correspondiente al fondo de la imagen. Una sugerencia para mejorar la red en base a los resultados es utilizar pesos tanto positivos como negativos con matrices de entrada integradas por ceros y unos. De esta forma el fondo no afectaría puesto que sólo puede dar lugar a valores de cero y serían los pesos los que originarían las salidas inhibitorias al tener valores negativos.

Después de analizar los resultados obtenidos con las distintas redes y los distintos intervalos de pesos iniciales vemos que al aumentar el intervalo de pesos mejora el comportamiento de las redes tendiéndose a alcanzar un equilibrio. De esta forma se puede obtener un intervalo óptimo de pesos para cada red; en las redes en estudio el intervalo óptimo ha sido el mismo en todos los casos.

El aumento del número de *neuronas* de la red no mejora el comportamiento de la misma. Es más, la red más pequeña (49 *neuronas* por capa) es la que alcanza a diferenciar entre un mayor número de patrones. Una sugerencia al respecto es hacer un aumento selectivo de *neuronas*. Esto es, agrupar las *neuronas* de forma que no todas las *neuronas* reciban la misma información. Seguramente un aumento del número de *neuronas* en una red con estas características sí daría lugar a un mejoramiento de la misma.

Las redes necesitan un entrenamiento más exhaustivo. Se le deben presentar los patrones de aprendizaje repetidamente a fin de que la información que se le haya perdido en el transcurso de las sesiones de aprendizaje se restablezca. Este comportamiento está de acuerdo con el humano que necesita proyecciones repetidas de una imagen antes de memorizarla.

Las redes, al estabilizarse, tienden a obtener valores promedio de pesos cercanos a cero siendo su desviación estándar muy baja. Esto ocurre para todas las redes independientemente del número de *neuronas* y es más patente en las últimas capas de la red.

Por último haremos notar que sería muy interesante implantar

distintas funciones de umbral y estudiar el comportamiento de las redes en los diversos casos.

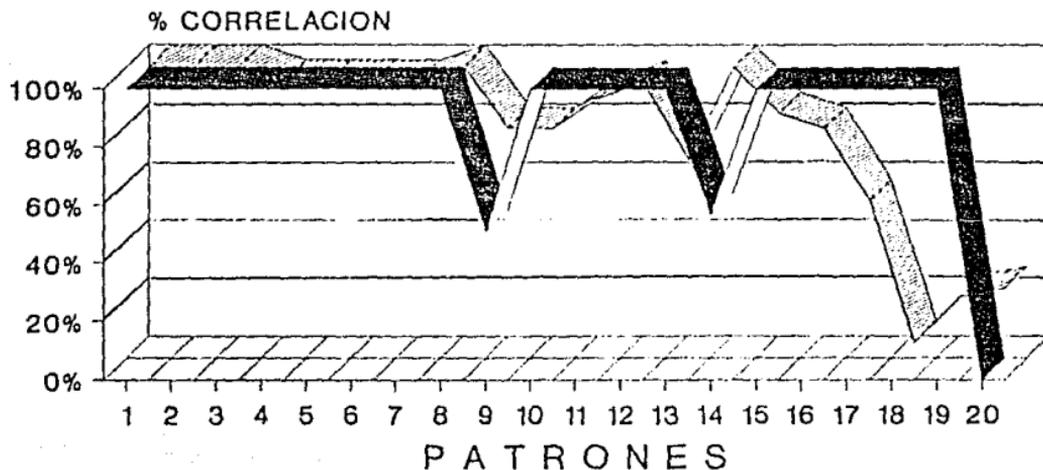
TABLA 6-1

TABLA COMPARATIVA  
RED NEURONAL / METODO ESTADISTICO

	RED 100 n/CAPA % RECONOCIMIENTO	% CORRELACION METODO ESTADISTICO			
		P-BASE-1	P-BASE-9	P-BASE-14	P-BASE-18
1	100.00%	100.00%	80.00%	100.00%	5.55%
2	100.00%	100.00%	80.00%	100.00%	5.55%
3	100.00%	100.00%	80.00%	100.00%	5.26%
4	100.00%	95.00%	76.00%	95.00%	6.25%
5	100.00%	95.00%	76.00%	100.00%	5.26%
6	100.00%	95.00%	80.00%	100.00%	5.00%
7	100.00%	95.00%	80.00%	100.00%	5.55%
8	100.00%	100.00%	80.00%	100.00%	5.55%
9	51.00%	79.16%	100.00%	80.00%	4.34%
10	100.00%	78.94%	64.00%	100.00%	7.42%
11	100.00%	89.47%	72.00%	90.00%	6.25%
12	100.00%	94.73%	76.00%	95.00%	5.88%
13	100.00%	68.42%	56.00%	70.00%	8.33%
14	57.00%	100.00%	80.00%	100.00%	5.55%
15	100.00%	84.21%	64.00%	80.00%	6.66%
16	100.00%	78.94%	64.00%	100.00%	7.14%
17	100.00%	54.28%	78.12%	64.51%	3.44%
18	100.00%	5.26%	4.00%	5.00%	100.00%
19	100.00%	21.05%	16.00%	20.00%	25.00%
20	.00%	24.05%	32.89%	26.66%	1.40%

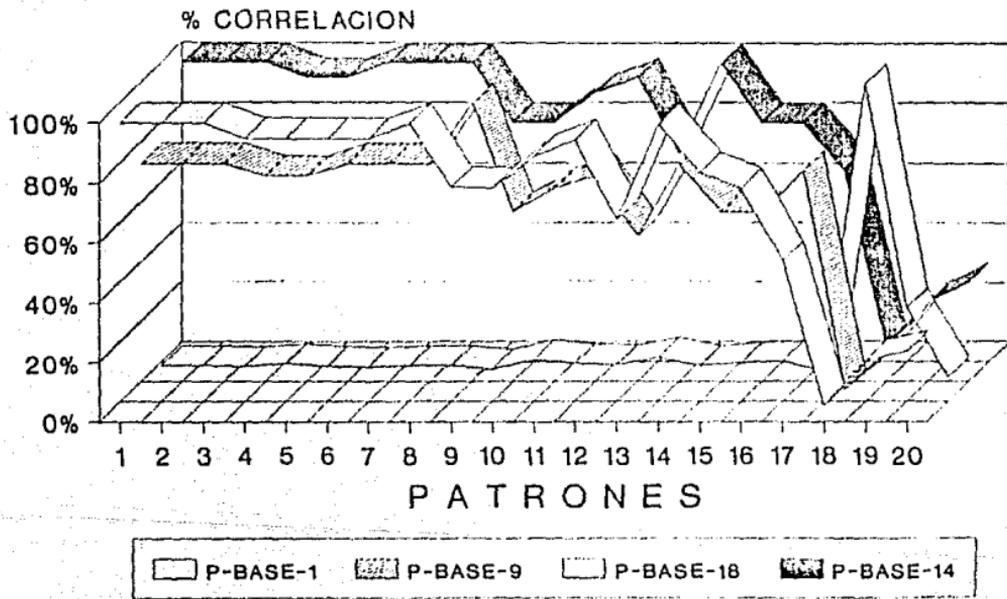
RANGO DE PESOS INICIALES PARA LA RED DE CERO A CINCO

# DIAGRAMA COMPARATIVO RED NEURONAL/METODO ESTADISTICO



GRAFICA 6-1

# ANALISIS ESTADISTICO COMPARACION DE PATRONES



GRAFICA 6-2

ANEXO 1.

PROGRAMA EN FORTRAN77.

```
% SIMULACION DEL COMPORTAMIENTO DE UNA RED NEURONAL DE APRENDIZAJE
% Y RECONOCIMIENTO DE PATRONES

%
% PROGRAM REDES
FILE 8(FILE="PAT15.",STATUS="OLD",KIND="DISK",RECL=2,ACCESS="DIRECT")
% ARCHIVO DE PATRONES PARA APRENDIZAJE Y RECONOCIMIENTO
FILE 9(STATUS="NEW",KIND="DISK",MAXRECSIZE=90,UNITS=1,
% ACCESS="DIRECT")
% SE UTILIZA EN LA GENERACION DE ARCHIVOS DE INFORMACION, CADA UNO
% DE ESTOS ARCHIVOS RECIBE UN NOMBRE Y UNA CLAVE DE ACCESO QUE SE
% GUARDA EN EL ARCHIVO DE CLAVES
FILE 10(FILE="CLAVE.",KIND="DISK",STATUS="OLD",RECL=8,
% AREAS=40,AREASIZE=10,ACCESS="DIRECT")
% ARCHIVO DONDE SE GUARDAN LAS LLAVES DE ACCESO A LOS ARCHIVOS
% DE INFORMACION QUE SE ORIGINAN EN EL PROCESO DE APRENDIZAJE.
FILE 20(KIND="PRINTER")
FILE 30(FILE="GNOM.",STATUS="NEW",KIND="DISK",ACCESS="DIRECT")
% ARCHIVO PARA LA GENERACION DE NOMBRES DE ARCHIVOS
PARAMETER (N = 225)
% (* NUMERO DE NEURONAS POR CAPA *)
PARAMETER (NM = 15)
% (* CAPA DE NM x NM NEURONAS *)
PARAMETER (TOLER = 0.000001)
% TOLERANCIA MAXIMA PARA EL ERROR EN EL AJUSTE DE LAS NEURONAS
PARAMETER (NUMITE = 1000)
% NUMERO DE ITERACIONES MAXIMO EN EL PROCESO DE ADAPTACION DE
% NEURONAS
%
```

```
COMMON CAL1, CAPA(5,N), WEIGHT(3,N,N), I, J, K, PN, ALFA  
COMMON XT2, SUMA(3,N), ORDEN(N), AEXT2, DESV(3), DERROR  
COMMON PMAX(3), PMIN(3), PROM(3), NP, NA, OPCION, RPORC(21), RANGO
```

```
REAL DESV, PMAX, PMIN, PROM, NA  
INTEGER RANGO, CAL1, I, J, K, PN, CAPA, ORDEN, NP  
REAL ALFA, DERROR, XT2, AEXT2, WEIGHT, SUMA
```

```
PROGRAMA PRINCIPAL
```

```
CALL CORRE
```

```
CLOSE(10)
```

```
STOP
```

```
END
```

```
SUBROUTINE CORRE
```

```
PARAMETER (N = 225)
```

```
PARAMETER (NM = 15)
```

```
PARAMETER (TOLER=0.000001)
```

```
PARAMETER (NUMITE=1000)
```

```
DIMENSION B(NM, NM)
```

```
COMMON CAL1, CAPA(5,N), WEIGHT(3,N,N), I, J, K, PN, ALFA  
COMMON XT2, SUMA(3,N), ORDEN(N), AEXT2, DESV(3), DERROR  
COMMON PMAX(3), PMIN(3), PROM(3), NP, NA, OPCION, RPORC(21), RANGO
```

```
REAL DESV, PMAX, PMIN, PROM, NA  
INTEGER RANGO, CAL1, I, J, K, PN, CAPA, ORDEN, NP  
REAL ALFA, DEPROR, XT2, AEXT2, WEIGHT, SUMA  
CHARACTER NOM1*06, NOM2*06, P*06
```

```
INICIALIZACION DE PARAMETROS
```

```
X='S'
```

```
Y='S'
```

```
P='A'
```

```
CAL1=0
```

```
PN=0
```

```
NP=0
```

```

NA=0
FANGO=1
CPEN(10)
OPEN(8)
‡ LOOP PARA PRUEBA CON DISTINTOS VALORES DE ALFA
DO 10 II=1,1
‡   ALFA=(II+3)/10
   ALFA=0.7
DO 20 JJ=1,1
‡   LECTURA DE LOS PATRONES DE ENTRADA EN ESTE CASO SOLO SE LEE
‡   UN PATRON, PERO SE PUEDE LEER UNA SERIE HACIENDO JJ=1,*
   READ(8,2)((B(L,M),L=1,NM),M=1,NM)
2   FORMAT(15,(15I1,/))
DO 36 J=1,NM
DO 36 I=1,NM
   IF (B(I,J) .EQ. 0) THEN
     CAPA(1,I+((J-1)*NM))=-1
     IF (JJ .EQ. 1) THEN
       CAPA(5,I+((J-1)*NM))=-1
     END IF
   ELSE
     CAPA(1,I+((J-1)*NM))=B(I,J)
     IF (JJ .EQ. 1) THEN
       CAPA(5,I+((J-1)*NM))=B(I,J)
     END IF
   END IF
‡ EN CAPA 5 SE GUARDA EL PATRON CONTRA EL QUE SE VA A
‡ CONTRASTAR EL RESULTADO DEL RECORRIDO POR LA RED
36 CONTINUE
   IF (JJ .EQ. 1) THEN
     CALL GENPES
     CALL FORRED
‡ FASE DE ADAPTACION DE LAS NEURONAS
DO 12 IR=1,3
DO 12 IS=1,N
   CALL ADALIN(IR,IS)
12 CONTINUE

```

```

ELSE
  CALL FORRED
END IF
  CALL MR11
20  CONTINUE
*  FASE DE RECONOCIMIENTO
  REWIND(8)
  DO 40 IK=1,20
READ(8,2)((B(L,M),L=1,NM),M=1,NM)
  DO 46 J=1,NM
  DO 46 I=1,NM
    IF (B(I,J) .EQ. 0) THEN
      CAPA(1,I+((J-1)*NM))=-1
*  CUANDO SE QUIERE COMPARAR CONTRA EL PATRON DE ENTRADA
      CAPA(5,I+((J-1)*NM))=-1
    ELSE
      CAPA(1,I+((J-1)*NM))=B(I,J)
      CAPA(5,I+((J-1)*NM))=B(I,J)
    END IF
46  CONTINUE
  CALL FORRED
  IF (CAL1 .EQ. 0) THEN
    RPORC(IK)=0
  ELSE
    RPORC(IK)=(CAL1/N)*100
  END IF
*  RPORC(IK) ES EL PORCENTAJE DE RECONOCIMIENTO DEL PATRON IK
40  CONTINUE
  CALL GCLAVE
WRITE(20)' ALFA NEU  RANGO  PATR  PROM-1  PROM-2  PROM-3'
WRITE(20,3)ALFA,N,RANGO,P,PROM(1),PROM(2),PROM(3)
3  FORMAT(F6.3,1X,I3,2X,I5,2X,A2,1X,F10.5,2X,F10.5,2X,F10.5)
WRITE(20,4)PMIN(1),PMIN(2),PMIN(3),PMAX(1),PMAX(2),PMAX(3),
*  DESV(1),DESV(2),DESV(3)
4  FORMAT(9(F10.5,1X),/)
WRITE(20,5)(RPORC(L),L=1,20)
5  FORMAT(20(F10.5,2X))

```

```

REWIND(8)
10 CONTINUE
END

%
SUBROUTINE GCLAVE
PARAMETER (N = 225)
PARAMETER (NM = 15)
PARAMETER (TOLER=0.000001)
PARAMETER (NUMITE=1000)

%
COMMON CALL, CAPA(5,N), WEIGHT(3,N,N), I, J, K, PN, ALFA
COMMON XT2, SUMA(3,N), ORDEN(N), AEXT2, DESV(3), DERROR
COMMON PMAX(3), PMIN(3), PROM(3), NP, NA, OPCION, RPORC(21), RANGO

%
%
REAL DESV, PMAX, PMIN, PROM, NA
INTEGER RANGO, CALL, I, J, K, PN, CAPA, ORDEN, NP
REAL ALFA, DERROR, XT2, AEXT2, WEIGHT, SUMA
CHARACTER NOM1*06, NOM2*06, P*06
READ(10,3,REC=1)J
PAT=1
I=(ALFA*10)+(RANGO*100)+(PAT*10000)+(NM*1000000)
WRITE(10,3,REC=J)I
3 FORMAT(I8)
WRITE(10,3,REC=1)J+1
IF (J .GT. 99) THEN
WRITE(30,5)"P",J,"."
5 FORMAT(A3,I3,A1)
ELSE
IF (J .GT. 9) THEN
WRITE(30,6)"P",J,"."
6 FORMAT(A4,I2,A1)
ELSE
WRITE(30,7)"P",J,"."
7 FORMAT(A5,I1,A1)
END IF
END IF
END IF

```

```

CALL GDATA
END
%
% (***** LLENADO DE LA RED CON PESOS ALEATORIOS *****)
SUBROUTINE GENPESOS
PARAMETER (N = 225)
PARAMETER (NM = 15)
PARAMETER (TOLER=0.000001)
PARAMETER (NUMITE=1000)
%
COMMON CALL, CAPA(5,N), WEIGHT(3,N,N), I, J, K, PN, ALFA
COMMON XT2, SUMA(3,N), ORDEN(N), AEXT2, DESV(3), DERROR
COMMON PMAX(3), PMIN(3), PROM(3), NP, NA, OPCION, RPORC(21), RANGO
%
%
REAL DESV, PMAX, PMIN, PROM, NA
INTEGER RANGO, CALL, I, J, K, PN, CAPA, ORDEN, NP
REAL ALFA, DERROR, XT2, AEXT2, WEIGHT, SUMA
INTEGER SEED
      DO 10 I =1,3
        DO 10 J =1,N
          DO 10 K =1,N
            SEED = K*J+I
            WEIGHTS(I,J,K) = RANGO*RANDOM(SEED)
10    CONTINUE
      RETURN
END
%
%
% (***** ALGORITMO DE ADAPTACION DE LA NEURONA *****)
SUBROUTINE ADALINE(I1,J1)
PARAMETER (N = 225)
PARAMETER (NM = 15)
PARAMETER (TOLER=0.000001)
PARAMETER (NUMITE=1000)
%
COMMON CALL, CAPA(5,N), WEIGHT(3,N,N), I, J, K, PN, ALFA

```

```
COMMON XT2, SUMA(3,N), ORDEN(N), AEXT2, DESV(3), DERROR
COMMON PMAX(3), PMIN(3), PROM(3), NP, NA, OPCION, RPORC(21), RANGO
```

```
REAL DESV, PMAX, PMIN, PROM, NA
INTEGER RANGO, CALL, I, J, K, PN, CAPA, ORDEN, NP
REAL ALFA, DERROR, XT2, AEXT2, WEIGHT, SUMA
```

```
INTEGER NI, ACUMUL, I1, J1
REAL TOL1
CHARACTER X, Y
CHARACTER *20 NOM1, NOMBRE
CHARACTER *2 P
```

```
NI = 0
DERROR = (CAPA(I1+2, J1) - SUMA(I1, J1))
TOL1 = SUMA(I1, J1)
AEXT2=ALFA/N
27 IF (NI .LT. NUMITE) THEN
  DO 20 K =1, N
    WEIGHT(I1, J1, K) = WEIGHT(I1, J1, K) + (DERROR* AEXT2*CAPA(I1, K))
20 CONTINUE
  SUMA(I1, J1)=0.0
  DO 30 K=1, N
    SUMA(I1, J1) = SUMA(I1, J1) + (CAPA(I1, K) * WEIGHT(I1, J1, K))
30 CONTINUE
  IF (SUMA(I1, J1) .GE. 0) THEN
    CAPA(I1+1, J1) = 1
  ELSE
    CAPA(I1+1, J1) = -1
  END IF
  DERROR = CAPA(I1+2, J1) - SUMA(I1, J1)
  IF (SUMA(I1, J1)-TOL1 .LT. TOLER) THEN
    J=NI
    NI = NUMITE
  ELSE
    J=NI
    TOL1 = SUMA(I1, J1)
```

```

END IF
  NI= NI+1
  GO TO 27
END IF
END

%
%
%  (***** PRIMER RECORRIDO POR LA RED CON PESOS ALEATORIOS *****)
SUBROUTINE FORRED
  PARAMETER (N = 225)
  PARAMETER (NM = 15)
  PARAMETER (TOLER=0.000001)
  PARAMETER (NUMITE=1000)

%
COMMON CAL1, CAPA(5,N), WEIGHT(3,N,N), I, J, K, PN, ALFA
COMMON XT2, SUMA(3,N), ORDEN(N), AEXT2, DESV(3), DERROR
COMMON PMAX(3), PMIN(3), PROM(3), NP, NA, OPCION, RPORC(21), RANGO

%
REAL DESV, PMAX, PMIN, PROM, NA
INTEGER RANGO, CAL1, I, J, K, PN, CAPA, ORDEN, NP
REAL ALFA, DERROR, XT2, AEXT2, WEIGHT, SUMA

%
DO 2 I=1,3
  DO 2 J=1,N
    SUMA(I,J)=0
2  CONTINUE

DO 10 I =1,3
  DO 20 J =1,N
    DO 30 K =1,N
      SUMA(I,J) = SUMA(I,J) + (WEIGHT(I,J,K)*CAPA(I,K))
30 CONTINUE
  IF (SUMA(I,J) .GE. 0) THEN
    CAPA(I+1,J) = 1
  ELSE
    CAPA(I+1,J) = -1
  END IF
20 CONTINUE

```

```

10  CONTINUE
    CAL1=0
    DO 60 K=1,N
        IF (CAPA(4,K) .EQ. CAPA(5,K)) THEN
            CAL1 = CAL1 +1
        ELSE
            CONTINUE
        END IF
60  CONTINUE
$   (* CALIFICA POR PRIMERA VEZ LA RED *)
END
$
$   (***** INDICA SI SE DEBEN REAJUSTAR LOS PESOS DE LA RED *****)
FUNCTION CALIF(CAL2,I2)
PARAMETER (N = 225)
PARAMETER (NM = 15)
PARAMETER (TOLER=0.000001)
PARAMETER (NUMITE=1000)
$
COMMON CAL1, CAPA(5,N), WEIGHT(3,N,N), I, J, K, PN, ALFA
COMMON XT2, SUMA(3,N), ORDEN(N), AEXT2, DESV(3), DERROR
COMMON PMAX(3), PMIN(3), PROM(3), NP, NA, OPCION, RPORC(21), RANGO
$
$
REAL DESV, PMAX, PMIN, PROM, NA
INTEGER RANGO, CAL1, I, J, K, PN, CAPA, ORDEN, NP
REAL ALFA, DERROR, XT2, AEXT2, WEIGHT, SUMA
$
PN=I2+1
DO 10 W = PN,3
DO 10 V = 1,N
    SUMA(W,V) = 0
    DO 20 R = 1,N
        SUMA(W,V) = SUMA(W,V) + CAPA(W,I)*WEIGHT(W,V,R)
20  CONTINUE
    IF SUMA(W,V) .GE. 0) THEN
        CAPA(W+1,V) = 1

```

```

        ELSE
            CAPA(W+1,V) = -1
10    CONTINUE
        CONTAD = 0
        DO 30 R = 1,N
            IF (CAPA(5,R) .EQ. CAPA(4,R)) THEN
                CONTAD = CONTAD +1
            END IF
30    CONTINUE
        IF (CONTAD .GT. CAL2) THEN
            CAL1 = CONTAD
            CAL2 = CAL1
            CALIF = 1
        RETURN
        ELSE
            CAL1 = CAL2
            CALIF = 0
        RETURN
    END IF
END

%
% (***** ORDENAMIENTO POR CAPAS CON BASE AL VALOR ABSOLUTO
% ANALOGICO DE C/NEURONA *****)
SUBROUTINE ORDENA(L)
    PARAMETER (N = 225)
    PARAMETER (NM = 15)
    PARAMETER (TOLER=0.000001)
    PARAMETER (NUMITE=1000)

%
COMMON CAL1, CAPA(5,N), WEIGHT(3,N,N), I, J, K, PN, ALFA
COMMON XT2, SUMA(3,N), ORDEN(N), AEXT2, DESV(3), DERROR
COMMON PMAX(3), PMIN(3), PROM(3), NP, NA, OPCION, RPORC(21), RANGO

%
%
REAL DESV, PMAX, PMIN, PROM, NA
INTEGER RANGO, CAL1, I, J, K, PN, CAPA, ORDEN, NP
REAL ALFA, DERROR, XT2, AEXT2, WEIGHT, SUMA

```

```
REAL W
INTEGER C, D, E, L, R, CAMBIO
```

```
DO 10 J=1, N
  ORDEN(J)=J
  DO 10 I=1, J-1
    W= SUMA(L, ORDEN(J))
    IF (ABS(W) .LT. ABS(SUMA(L, ORDEN(I)))) THEN
      K=ORDEN(I)
      ORDEN(I)=ORDEN(J)
      ORDEN(J)=K
    END IF
  CONTINUE
RETURN
END
```

```
(*ALGORITMO QUE EJECUTA EL PRINCIPIO DE MINIMA PERTURBACION *)
```

```
SUBROUTINE MRII
```

```
PARAMETER (N = 225)
PARAMETER (NM = 15)
PARAMETER (TOLER=0.000001)
PARAMETER (NUMITE=1000)
```

```
COMMON CAL1, CAPA(5, N), WEIGHT(3, N, N), I, J, K, PN, ALFA
COMMON XT2, SUMA(3, N), ORDEN(N), AEXT2, DESV(3), DERROR
COMMON PMAX(3), PMIN(3), PROM(3), NP, NA, OPCION, RPORC(21), RANGO
```

```
REAL DESV, PMAX, PMIN, PROM, NA
INTEGER RANGO, CAL1, I, J, K, PN, CAPA, ORDEN, NP
REAL ALFA, DERROR, XT2, AEXT2, WEIGHT, SUMA
```

```
INTEGER C2(5, N)
INTEGER L, M, R, S, I2, J2, IP, K1, IK, CALIF, CAL2, CAMBIO
REAL W2(3, N, N), S2(3, N)
```

```

CAMBIO = 0
CAL2 = CAL1
DO 5 I = 1,3
  DO 5 J2=1,N
    S2(I,J2)=SUMA(I,J2)
    C2(I+1,J2)=CAPA(I+1,J2)
  DO 6 K=1,N
    W2(I,J2,K)=WEIGHT(I,J2,K)
6 CONTINUE
5 CONTINUE
  DO 17 I=1,N
    C2(1,I)=CAPA(1,I)
    C2(5,I)=CAPA(5,I)
17 CONTINUE
  DO 10 I2=1,2
    IF (CAL2 .LT. N) THEN
      CALL ORDENA(I2)
    DO 20 J2=1,N
      IK=ORDEN(J2)
      CAPA(I2+1,IK)=-1*CAPA(I2+1,IK)
      IF (CALIF(CAL2,I2) .EQ. 0) THEN
        DO 15 I=1,3
          DO 15 J=1,N
            SUMA(I,J)=S2(I,J)
            CAPA(I+1,J)=C2(I+1,J)
          DO 16 K=1,N
            WEIGHT(I,J,K)=W2(I,J,K)
16 CONTINUE
            CAPA(1,J) = C2(1,J)
            CAPA(5,J) = C2(5,J)
15 CONTINUE
          END IF
          IF (CALIF(CAL2,I2) .EQ. 1) THEN
            CAPA(I2+2,IK) = CAPA(I2+1,IK)
            CAPA(I2+1,IK) = -1*CAPA(I2+1,IK)
          DO 25 I=1,3
            DO 25 J=1,N

```

```

        S2(I,J)=SUMA(I,J)
        C2(I+1,J)=CAPA(I+1,J)
    DO 26 K=1,N
        W2(I,J,K)=WEIGHT(I,J,K)
        C2(1,K)=CAPA(1,K)
        C2(5,K)=CAPA(5,K)
26     CONTINUE
25     CONTINUE
        END IF
20     CONTINUE
        END IF
10     CONTINUE
DO 100 I2=1,2
    IF (CAL2 .LT. N) THEN
        DO 30 J = 1,N
            DO 30 K1 =J+1,N
                CAPA(I2+1,J) =-1*CAPA(I2+1,J)
                CAPA(I2+1,K1) =-1*CAPA(I2+1,K1)
            IF (CALIF(CAL2,I2) .EQ. 0) THEN
                DO 35 I=1,3
                    DO 35 IP=1,N
                        SJMA(I,IP)=S2(I,IP)
                        CAPA(I+1,IP)=C2(I+1,IP)
                    DO 36 K=1,N
                        WEIGHT(I,IP,K)=W2(I,IP,K)
                        CAPA(1,K)=C2(1,K)
                        CAPA(5,K)=C2(5,K)
36     CONTINUE
35     CONTINUE
                    ELSE
                        CAPA(I2+2,J) = CAPA(I2+1,J)
                        CAPA(I2+1,J) = -1*CAPA(I2+1,J)
                        CAPA(I2+2,K1) = CAPA(I2+1,K1)
                        CAPA(I2+1,K1) =-1*CAPA(I2+1,K1)
                        CALL ADALIN(I2,K1)
                DO 11 I=1,3
                    DO 11 IP=1,N

```

ESTA TESIS NO DEBE  
SALIR DE LA BIBLIOTECA

```
S2(I,IP)=SUMA(I,IP)
C2(I+1,IP)=CAPA(I+1,IP)
DO 12 K=1,N
W2(I,IP,K)=WEIGHT(I,IP,K)
C2(1,K)=CAPA(1,K)
C2(5,K)=CAPA(5,K)
12 CONTINUE
11 CONTINUE
END IF
30 CONTINUE
END IF
100 CONTINUE
DO 40 J=1,N
CALL ADALIN(3,J)
40 CONTINUE
I= CALIF(CALL,1)
NA= CALL/N
END

%
% (***** RECORRIDO POR LA RED CON BASE A UN PATRON POR RECONOCER **
SUBROUTINE RPAT
PARAMETER (N = 225)
PARAMETER (NM = 15)
PARAMETER (TOLER=0.000001)
PARAMETER (NUMITE=1000)
DIMENSION B(N),CPA(5,N)

%
COMMON CALL, CPA(5,N), WEIGHT(3,N,N), I, J, K, PN, ALFA
COMMON XT2, SUMA(3,N), ORDEN(N), AEXT2, DESV(3), DERROR
COMMON PMAX(3), PMIN(3), PROM(3), NP, NA, OPCION, RPORC(21), RANGO

%
%
REAL DESV, PMAX, PMIN, PROM, NA
INTEGER RANGO, CALL, I, J, K, PN, CPA, ORDEN, NP
REAL ALFA, DERROR, XT2, AEXT2, WEIGHT, SUMA
CHARACTER *20 NOM1, NOMBRE

%
```

```

REAL CUENTA
INTEGER B, CPA
*
OPEN(5, FILE=NOMBRE, STATUS='OLD')
READ(5, 10) (B(L), L=1, N)
10  FORMAT(9I2)
CLOSE(5)
DO 16 I=1, N
    CAPA(1, I)=B(I)
16  CONTINUE
DO 35 I=1, 3
    DO 30 J=1, N
        SUMA(I, J) = 0
        DO 20 K=1, N
            SUMA(I, J) = SUMA(I, J) - CAPA(I, K)*WEIGHT(I, J, K)
        IF (SUMA(I, J) .GE. 0) THEN
            CAPA(I+1, J) = 1
        ELSE
            CAPA(I+1, J) = -1
        END IF
    END IF
20  CONTINUE
30  CONTINUE
35  CONTINUE
OPEN(5, FILE=NOMBRE, STATUS='OLD')
READ(5, 11) (B(L), L=1, N)
11  FORMAT(9I2)
CLOSE(5)
DO 36 I=1, N
    CAPA(5, I)=B(I)
36  CONTINUE
CUENTA = 0
DO 40 I=1, N
    IF (CAPA(4, I) .EQ. CAPA(5, I)) THEN
        CUENTA = CUENTA + 1
    END IF
40  CONTINUE
CUENTA=(CUENTA*100)/N

```

```

IF (CUENTA .EQ. 100) THEN
  NP=NP+1
END IF
END

```

```

(***** LECTURA DE LOS PATRONES DE APRENDIZAJE Y APLICACION DE MRI
SUBROUTINE LEEPAT

```

```

  PARAMETER (N = 225)
  PARAMETER (NM = 15)
  PARAMETER (TOLER=0.000001)
  PARAMETER (NUMITE=1000)

```

```

COMMON CAL1, CAPA(5,N), WEIGHT(3,N,N), I, J, K, PN, ALFA
COMMON XT2, SUMA(3,N), ORDEN(N), AEXT2, DESV(3), DERROR
COMMON PMAX(3), PMIN(3), PROM(3), NP, NA, OPCION, RPORC(21), RANGO

```

```

REAL DESV, PMAX, PMIN, PROM, NA
INTEGER RANGO, CAL1, I, J, K, PN, CAPA, ORDEN, NP
REAL ALFA, DERROR, XT2, AEXT2, WEIGHT, SUMA

```

```

CALL RPAT
CALL MR11
END

```

```

(***** ALMACENAMIENTO DE LOS PESOS DEFINITIVOS *****)

```

```

SUBROUTINE GDATA
  PARAMETER (N = 225)
  PARAMETER (NM = 15)
  PARAMETER (TOLER=0.000001)
  PARAMETER (NUMITE=1000)

```

```

COMMON CAL1, CAPA(5,N), WEIGHT(3,N,N), I, J, K, PN, ALFA
COMMON XT2, SUMA(3,N), ORDEN(N), AEXT2, DESV(3), DERROR
COMMON PMAX(3), PMIN(3), PROM(3), NP, NA, OPCION, RPORC(21), RANGO

```

```

REAL DESV, PMAX, PMIN, PROM, NA

```

```
INTEGER RANGO,CAL1, I, J, K, PN, CAPA, ORDEN, NP  
REAL ALFA, DERROR, XT2, AEXT2, WEIGHT, SUMA
```

```
CHARACTER X,Y  
CHARACTER *7 NOM1,NOMBRE  
CHARACTER *2 P
```

```
NP=0
```

```
CALL PESOCL
```

```
REWIND(30)
```

```
READ(30,88)NOMBRE
```

```
88 FORMAT(A7)
```

```
OPEN(9,FILE=NOMBRE)
```

```
WRITE(9,11)(DESV(L),L=1,3),(PMAX(I),I=1,3),(PMIN(J),J=1,3)  
*,PROM(K),K=1,3),((WEIGHT(L,M,K),L=1,3),M=1,N),K=1,N)
```

```
11 FORMAT(3F10.5,/,3F10.5,/,3F10.5,/,3F10.5,
```

```
*,/,3(9F10.5,/,/))
```

```
CLOSE(9)
```

```
END
```

```
(***** LECTURA DE LOS PESOS DEFINITIVOS *****)
```

```
SUBROUTINE LEEPES
```

```
PARAMETER (N = 225)
```

```
PARAMETER (NM = 15)
```

```
PARAMETER (TOLER=0.000001)
```

```
PARAMETER (NUMIT!=1000)
```

```
COMMON CAL1, CAPA(5,N), WEIGHT(3,N,N), I, J, K, PN, ALFA  
COMMON XT2, SUMA(3,N), ORDEN(N), AEXT2, DESV(3), DERROR  
COMMON PMAX(3), PMIN(3), PROM(3), NP, NA, OPCION, RPORC(21), RANGO
```

```
REAL DESV, PMAX, PMIN, PROM, NA
```

```
INTEGER RANGO,CAL1, I, J, K, PN, CAPA, ORDEN, NP
```

```
REAL ALFA, DERROR, XT2, AEXT2, WEIGHT, SUMA
```

```
CHARACTER *20 NOM1,NOMBRE
```

```

OPEN(5,FILE=NOMBRE,STATUS='OLD')
READ(5,11)(DESV(L),L=1,3),(PMAX(I),I=1,3),(PMIN(J),J=1,3),
* (PROM(K),K=1,3),((WEIGHT(L,M,K),L=1,3),M=1,N),K=1,N)
11 FORMAT(3F10.5,/,3F10.5,/,3F10.5,/,3F10.5,
* /,3(9F10.5,/,/))
CLOSE(5)
END

*
* (***** LECTURA DE LOS PESOS DEL PATRON APRENDIDO *****)
SUBROUTINE REECP
PARAMETER (N = 225)
PARAMETER (NM = 15)
PARAMETER (TOLER=0.000001)
PARAMETER (NUMITE=1000)
DIMENSION B(N)

*
COMMON CAL1, CAPA(5,N), WEIGHT(3,N,N), I, J, K, PN, ALFA
COMMON XT2, SUMA(3,N), ORDEN(N), AEXT2, DESV(3), DERROR
COMMON PMAX(3), PMIN(3), PROM(3), NP, NA, OPCION, RPORC(21), RANGO

*
*
REAL DESV, PMAX, PMIN, PROM, NA
INTEGER RANGO, CAL1, I, J, K, PN, CAPA, ORDEN, NP
REAL ALFA, DERROR, XT2, AEXT2, WEIGHT, SUMA

*
CHARACTER X, Y
CHARACTER *20 NOM1, NOMBRE
CHARACTER *2 P

*
CALL LEEPES
100 WRITE(*,*)' QUIERES RECONOCER ALGUN PATRON (S/N) ? : '
READ(*,58)X
58 FORMAT(A)
IF (X.EQ. 's') THEN
X='S'
END IF
IF (X.EQ. 'S') THEN

```

```

CALL RPAT
GOTO 100
END IF
CALL MENU
END

```

```

SUBROUTINE PESOCL
PARAMETER (N = 225)
PARAMETER (NM = 15)
PARAMETER (TOLER=0.000001)
PARAMETER (NUMITE=1000)
DIMENSION B(N)
DIMENSION SUM(3)

```

```

COMMON CAL1, CAPA(5,N), WEIGHT(3,N,N), I, J, K, PN, ALFA
COMMON XT2, SUMA(3,N), ORDEN(N), AEXT2, DESV(3), DERROR
COMMON PMAX(3), PMIN(3), PROM(3), NP, NA, OPCION, RPORC(21), RANGO

```

```

REAL DESV, PMAX, PMIN, PROM, NA
INTEGER RANGO, CAL1, I, J, K, PN, CAPA, ORDEN, NP
REAL ALFA, DERROR, XT2, AEXT2, WEIGHT, SUMA

```

```

DO 77 I=1,3
  PMAX(I)=WEIGHT(I,1,1)
  PMIN(I)=WEIGHT(I,1,1)
  PROM(I)=0
77 CONTINUE
DO 10 I=1,3
  DO 20 J=1,N
    DO 30 K=1,N
      IF (PMAX(I) .LT. WEIGHT(I,J,K)) THEN
        PMAX(I)=WEIGHT(I,J,K)
      END IF
      IF (PMIN(I) .GT. WEIGHT(I,J,K)) THEN
        PMIN(I)=WEIGHT(I,J,K)

```

```

        END IF
        PROM(I)=PROM(I)+WEIGHT(I,J,K)
30    CONTINUE
20    CONTINUE
10    CONTINUE
        DO 55 I=1,3
            IF (PROM(I) .EQ. 0) THEN
                PROM(I)=0
            ELSE
                PROM(I)=PROM(I)/(N*N)
            END IF
55    CONTINUE
        DO 65 I=1,3
            DO 75 J=1,N
                SUM(I)=SUM(I)+CAPA(I,J)-PROM(I)**2
75    CONTINUE
            DESV(I)=SQRT(SUM(I))
            DESV(I)=DESV(I)/N
65    CONTINUE
        RETURN
    END

```

‡

‡

(\*\*\*\*\* INICIALIZA VARIABLES \*\*\*\*\*)

SUBROUTINE INICIA

```

    PARAMETER (N = 225)
    PARAMETER (NM = 15)
    PARAMETER (TOLER=0.000001)
    PARAMETER (NUMITE=1000)
    DIMENSION B(NM,NM)

```

‡

```

COMMON CALL, CAPA(5,N), WEIGHT(3,N,N), I, J, K, PN, ALFA
COMMON XT2, SUMA(3,N), ORDEN(N), AEXT2, DESV(3), DERROR
COMMON PMAX(3), IMIN(3), PROM(3), NP, NA, OPCION, RPORC(21), RANGO

```

‡

‡

```

REAL DESV, PMAX, IMIN, PROM, NA
INTEGER RANGO, CALL, I, J, K, PN, CAPA, ORDEN, NP

```

```

REAL ALFA, DEPROR, XT2, AEXT2, WEIGHT, SUMA
3
CHARACTER X,Y
CHARACTER *20 NOM1,NOMBRE
CHARACTER *2 P
3
X = 'S'
Y = 'S'
XT2= N
CALL = 0
PN = 0
ALFA = 0.7
NP = 0
NA = 0
RANGO= 1
AEXT2= ALFA/XT2
DO 10 I=1,3
DO 10 J=1,N
SUMA(I,J) = 0
10 CONTINUE
OPEN(8)
READ(8,12)((B(L,M),L=1,NM),M=1,NM)
12 FORMAT(15(15I1,/))
CLOSE(8)
DO 36 J=1,NM
DO 36 I=1,NM
IF (B(J,I) .EQ. 0 ) THEN
CAPA(1,I+((J-1)*NM))=-1
CAPA(5,I+((J-1)*NM))=-1
ELSE
CAPA(1,I+((J-1)*NM))=B(J,I)
CAPA(5,I+((J-1)*NM))=B(J,I)
END IF
36 CONTINUE
END
3
SUBROUTINE ENTREN

```

```

PARAMETER (N = 225)
PARAMETER (NM = 15)
PARAMETER (TOLER=0.000001)
PARAMETER (NUMITE=1000)

```

```

COMMON CALL, CAPA(5,N), WEIGHT(3,N,N), I, J, K, PN, ALFA
COMMON XT2, SUMA(3,N), ORDEN(N), AEXT2, DESV(3), DERROR
COMMON PMAX(3), PMIN(3), PROM(3), NP, NA, OPCION, RPORC(21), RANGO

```

```

REAL DESV, PMAX, PMIN, PROM, NA
INTEGER RANGO, CALL, I, J, K, PN, CAPA, ORDEN, NP
REAL ALFA, DERROR, XT2, AEXT2, WEIGHT, SUMA

```

```

CHARACTER X, Y
CHARACTER *20 NOM1, NOMBRE
CHARACTER *2 P

```

```

100 WRITE(*,*)'          ** MENU PARA ENTRENAMIENTO **'
    WRITE(*,*)'          A.-NUEVO PATRON A RECONOCER'
    WRITE(*,*)'          B.-MAS PATRONES DE ENTRENAMIENTO'
    WRITE(*,*)'          C.-FIN DEL ENTRENAMIENTO'
    WRITE(*,*)'          QUE OPCION DESEA?'
    READ(*,68)X
68  FORMAT(A)
    IF (X .EQ. 'A') THEN
    WRITE(*,*)'          D.- USO EL MISMO ARCHIVO DE PATRONES APRENDIDO ?
    WRITE(*,*)'          E.- GENERO NUEVO ARCHIVO DE PESOS ?'
    WRITE(*,*)'          OPCION >'
    READ(*,58)X
58  FORMAT(A)
    IF (X .EQ. 'D') THEN
        CALL INICIA
        CALL LEEPES
        CALL FORRED
        CALL MRII
        CALL GCLAVE
    END IF

```

```

IF (X .EQ. 'E') THEN
  CALL INICIA
  CALL GENPES
  CALL FORRED
  CALL MRRI
  CALL GCLAVE
END IF
RETURN
END IF
IF (X .EQ. 'B') THEN
  CALL INICIA
  CALL LEEPES
  CALL FORRED
  CALL MRRI
  CALL INICIA
RETURN
END IF
IF (X .EQ. 'C') THEN
  RETURN
ELSE
  GOTO 100
END IF
END

```

```

%
%
% (***** MENU GENERAL DE OPCIONES *****)

```

```

SUBROUTINE MENU

```

```

  PARAMETER (N = 225)
  PARAMETER (NM = 15)
  PARAMETER (TOLER=0.000001)
  PARAMETER (NUMITE=1000)

```

```

%
%
% COMMON CAL1, CAPA(5,N), WEIGHT(3,N,N), I, J, K, PN, ALFA
COMMON XT2, SUMA(3,N), ORDEN(N), AEXT2, DESV(3), DERROR
COMMON PMAX(3), PMIN(3), PROM(3), NP, NA, OPCION, RPORC(21), RANGO
%
%

```

```
REAL DESV, PMAX, PMIN, PROM, NA  
INTEGER RANGO, CALL1, I, J, K, PN, CAPA, ORDEN, NP  
REAL ALFA, DERROR, XT2, AEXT2, WEIGHT, SUMA
```

```
CHARACTER X, Y  
CHARACTER *20 NOM1, NOMBRE  
CHARACTER *2 P
```

```
200 WRITE(*,*)' ** MENU GENERAL **'  
WRITE(*,*)' A.- ENTRENAMIENTO DE LA RED'  
WRITE(*,*)' B.- RECONOCE PATRON APRENDIDO'  
WRITE(*,*)' C.- APRENDIZAJE Y RECONOCIMIENTO AUTOM  
WRITE(*,*)' D.- FIN DEL PROGRAMA'  
WRITE(*,*)' OPCION > '  
READ(*,58)X  
58 FORMAT(A)  
IF (X .EQ. 'A') THEN  
CALL ENTREN  
GOTO 200  
END IF  
IF (X .EQ. 'B') THEN  
CALL LEEPES  
CALL RPAT  
GOTO 200  
END IF  
IF (X .EQ. 'D') THEN  
WRITE(*,*)' FIN DEL PROGRAMA'  
WRITE(*,*)' ADIOS'  
RETURN  
ELSE  
GOTO 200  
END IF  
END
```